WIZ - EXHIBIT 1004 (1)
Wiz, Inc. v. Orca Security Ltd.

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	ISSUE DATE	PATENT NO.	ATTORNEY DOCKET NO.	CONFIRMATION NO.
17/657,981	04/11/2023	11627154	15635.0008-00000	2397

22852 7590 03/22/2023

FINNEGAN, HENDERSON, FARABOW, GARRETT & DUNNER
LLP
901 NEW YORK AVENUE, NW
WASHINGTON, DC 20001-4413

ISSUE NOTIFICATION

The projected patent number and issue date are specified above.

Determination of Patent Term Adjustment under 35 U.S.C. 154 (b)
(application filed on or after May 29, 2000)

The Patent Term Adjustment is 0 day(s). Any patent to issue from the above-identified application will include an indication of the adjustment on the front page.

If a Continued Prosecution Application (CPA) was filed in the above-identified application, the filing date that determines Patent Term Adjustment is the filing date of the most recent CPA.

Applicant will be able to obtain more detailed information by accessing the Patent Application Information Retrieval (PAIR) WEB site (http://pair.uspto.gov).

Any questions regarding the Patent Term Extension or Adjustment determination should be directed to the Office of Patent Legal Administration at (571)-272-7702. Questions relating to issue and publication fee payments should be directed to the Application Assistance Unit (AAU) of the Office of Patents Stakeholder Experience (OPSE), Stakeholder Support Division (SSD) at (571)-272-4200.

INVENTOR(s) (Please see PAIR WEB site http://pair.uspto.gov for additional inventors):

Avi SHUA, Tel Aviv-Yafo, ISRAEL;

APPLICANT(s) (Please see PAIR WEB site http://pair.uspto.gov for additional applicants):

Orca Security, Tel Aviv-Yafo, ISRAEL;

The United States represents the largest, most dynamic marketplace in the world and is an unparalleled location for business investment, innovation, and commercialization of new technologies. The USA offers tremendous resources and advantages for those who invest and manufacture goods here. Through SelectUSA, our nation works to encourage and facilitate business investment. To learn more about why the USA is the best country in the world to develop technology, manufacture products, and grow your business, visit SelectUSA.gov.

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Use as many sheets as necessary)				Complete if Known	
				Application Number	17/657,981
				Filing Date	April 5, 2022
				First Named Inventor	Avi SHUA
				Art Unit	OPAP
				Examiner Name	CENTRAL, DOCKET
Sheet	2	of	3	Attorney Docket Number	**15635.0008-00000**

U.S. PUBLISHED PATENT APPLICATIONS

Examiner Initials*	Cite No.	Document Number Number-Kind Code (if known)	Publication Date MM-DD-YYYY	Name of Patentee or Applicant of Cited Document	Pages, Columns, Lines, Where Relevant Passages or Relevant Figures Appear
		US-2022/0182294A1	06-2022 ~~06-09-2022~~	Li et al.	
		US-2022/0100869A1	03-31-2022	BERGER et al.	
		US-2020/0396254A1	12-17-2020	Crabtree et al.	
		US-2016/0330237A1	11-10-2016	Edlabadkar	
		US-2018/0205755A1	07-19-2018	Kavi et al.	
		US-2021/0342297A1	11-04-2021	Gupta et al.	
		US-2022/0239681A1	07-28-2022	Newman et al.	
		US-2022/0124110A1	04-21-2022	CHHABRA et al.	
		US-2021/0099478A1	04-01-2021	Seetharamaiah et al.	
		US-2020/0228560A1	07-16-2020	Murthy et al.	

Change(s) applied to document, /J.G./ 12/9/2022

Note: Submission of copies of U.S. Patents and published U.S. Patent Applications is not required.

FOREIGN PATENT DOCUMENTS

Examiner Initials*	Cite No.	Foreign Patent Document Country Code Number Kind Code (if known)	Publication Date MM-DD-YYYY	Name of Patentee or Applicant of Cited Document	Pages, Columns, Lines, Where Relevant Passages or Relevant Figures Appear	Translation
		WO 2020/106973A1	05-28-2020	ARAALI NETWORKS INC.		N/A
		WO 2019/051507A1	03-14-2019	CARBON BLACK INC.		N/A
		WO 2017/153983A1	09-14-2017	B. G. NEGEV TECHNOLOGIES AND APPLICATIONS LTD.		N/A

NON PATENT LITERATURE DOCUMENTS

Examiner Initials*	Cite No.	Include name of the author (in CAPITAL LETTERS), title of the article (when appropriate), title of the item (book, magazine, journal, serial, symposium, catalog, etc.), date, page(s), volume-issue number(s), publisher, city and/or country where published.	Translation
		Non-Final Office Action Application No. 17/716,491 dated August 3, 2022, in the United States Patent and Trademark Office.	N/A
		Non-Final Office Action Application No. 17/657,986 dated August 17, 2022, in the United States Patent and Trademark Office.	N/A
		Non-Final Office Action Application No. 17/658,156 dated July 18, 2022, in the United States Patent and Trademark Office.	N/A
		Non-Final Office Action Application No. 17/716,583 dated August 18, 2022, in the United States Patent and Trademark Office.	N/A
		Non-Final Office Action Application No. 17/657,972 dated June 21, 2022, in the United States Patent and Trademark Office.	N/A
		Final Office Action Application No. 17/657,972 dated September 1, 2022, in the United States Patent and Trademark Office.	N/A
		Ex Parte Quayle Action Application No. 17/716,231 dated June 26, 2022, in the United States Patent and Trademark Office.	N/A

Examiner Signature	/SARAH SU/	Date Considered	10/13/2022

*EXAMINER: Initial if reference considered, whether or not citation is in conformance with MPEP 609. Draw line through citation if not in conformance and not considered. Include copy of this form with next communication to applicant.

02

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.S/

UNITED STATES
PATENT AND TRADEMARK OFFICE

P.O. Box 1450
Alexandria, VA 22313 - 1450
www.uspto.gov

ELECTRONIC PAYMENT RECEIPT

APPLICATION #	RECEIPT DATE / TIME	ATTORNEY DOCKET #
17/657,981	03/07/2023 03:37:14 PM ET	15635.0008-00000

Title of Invention

Forward and Rearward Facing Attack Vector Visualization

Application Information

APPLICATION TYPE	Utility - Nonprovisional Application under 35 USC 111(a)	PATENT #	-
CONFIRMATION #	2397	FILED BY	Wanda Whittington
PATENT CENTER #	61702287	AUTHORIZED BY	Christopher Johns
CUSTOMER #	22852	FILING DATE	04/05/2022
CORRESPONDENCE ADDRESS	-	FIRST NAMED INVENTOR	Avi SHUA

Payment Information

PAYMENT METHOD	PAYMENT TRANSACTION ID	PAYMENT AUTHORIZED BY
CARD / 1007	E202337F38109560	Wanda Whittington

FEE CODE	DESCRIPTION	ITEM PRICE($)	QUANTITY	ITEM TOTAL($)
2501	UTILITY ISSUE FEE	600.00	1	600.00
			TOTAL AMOUNT:	$600.00

This Acknowledgement Receipt evidences receipt on the noted date by the USPTO of the indicated documents, characterized by the applicant, and including page counts, where applicable. It serves as evidence of receipt similar to a Post Card, as described in MPEP 503.

New Applications Under 35 U.S.C. 111

If a new application is being filed and the application includes the necessary components for filing date (see 37 CFR 1.53(b)-(d) and MPEP 506), a Filing Receipt (37 CFR 1.54) will be issued in due course and the date shown on this Acknowledgement Receipt will establish the filing date of the application

National Stage of an International Application under 35 U.S.C. 371

If a timely submission to enter the national stage of an international application is compliant with the conditions of 35 U.S.C. 371 and other applicable requirements a Form PCT/DO/EO/903 indicating acceptance of the application as a national stage

submission under 35 U.S.C. 371 will be issued in addition to the Filing Receipt, in due course.

New International Application Filed with the USPTO as a Receiving Office

If a new international application is being filed and the international application includes the necessary components for an international filing date (see PCT Article 11 and MPEP 1810), a Notification of the International Application Number and of the International Filing Date (Form PCT/RO/105) will be issued in due course, subject to prescriptions concerning national security, and the date shown on this Acknowledgement Receipt will establish the international filing date of the application.

PART B - FEE(S) TRANSMITTAL

Complete and send this form, together with applicable fee(s), by mail or fax, or via EFS-Web.

By mail, send to: Mail Stop ISSUE FEE
Commissioner for Patents
P.O. Box 1450
Alexandria, Virginia 22313-1450

By fax, send to: (571)-273-2885

INSTRUCTIONS: This form should be used for transmitting the ISSUE FEE and PUBLICATION FEE (if required). Blocks 1 through 5 should be completed where appropriate. All further correspondence including the Patent, advance orders and notification of maintenance fees will be mailed to the current correspondence address as indicated unless corrected below or directed otherwise in Block 1, by (a) specifying a new correspondence address; and/or (b) indicating a separate "FEE ADDRESS" for maintenance fee notifications.

CURRENT CORRESPONDENCE ADDRESS (Note: Use Block 1 for any change of address)

22852 7590 12/07/2022

FINNEGAN, HENDERSON, FARABOW, GARRETT & DUNNER LLP
901 NEW YORK AVENUE, NW
WASHINGTON, DC 20001-4413

Note: A certificate of mailing can only be used for domestic mailings of the Fee(s) Transmittal. This certificate cannot be used for any other accompanying papers. Each additional paper, such as an assignment or formal drawing, must have its own certificate of mailing or transmission.

Certificate of Mailing or Transmission

I hereby certify that this Fee(s) Transmittal is being deposited with the United States Postal Service with sufficient postage for first class mail in an envelope addressed to the Mail Stop ISSUE FEE address above, or being transmitted to the USPTO via EFS-Web or by facsimile to (571) 273-2885, on the date below.

(Typed or printed name)
(Signature)
(Date)

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
17/657,981	04/05/2022	Avi SHUA	15635.0008-00000	2397

TITLE OF INVENTION: Forward and Rearward Facing Attack Vector Visualization

APPLN. TYPE	ENTITY STATUS	ISSUE FEE DUE	PUBLICATION FEE DUE	PREV. PAID ISSUE FEE	TOTAL FEE(S) DUE	DATE DUE
nonprovisional	UNDISCOUNTED	$1200	$0.00	$0.00	$1200	03/07/2023

EXAMINER	ART UNIT	CLASS-SUBCLASS
SU, SARAH	2431	726-025000

1. Change of correspondence address or indication of "Fee Address" (37 CFR 1.363).

☐ Change of correspondence address (or Change of Correspondence Address form PTO/AIA/122 or PTO/SB/122) attached.

☐ "Fee Address" indication (or "Fee Address" Indication form PTO/AIA/47 or PTO/SB/47; Rev 03-02 or more recent) attached. **Use of a Customer Number is required.**

2. For printing on the patent front page, list
(1) The names of up to 3 registered patent attorneys or agents OR, alternatively,
(2) The name of a single firm (having as a member a registered attorney or agent) and the names of up to 2 registered patent attorneys or agents. If no name is listed, no name will be printed.

1 Finnegan, Henderson,
2 Farabow, Garrett &
3 Dunner LLP

3. ASSIGNEE NAME AND RESIDENCE DATA TO BE PRINTED ON THE PATENT (print or type)

PLEASE NOTE: Unless an assignee is identified below, no assignee data will appear on the patent. If an assignee is identified below, the document must have been previously recorded, or filed for recordation, as set forth in 37 CFR 3.11 and 37 CFR 3.81(a). Completion of this form is NOT a substitute for filing an assignment.

(A) NAME OF ASSIGNEE: ORCA SECURITY LTD.

(B) RESIDENCE: (CITY and STATE OR COUNTRY): TEL-AVIV-YAFO, ISRAEL

Please check the appropriate assignee category or categories (will not be printed on the patent) : ☐ Individual ☒ Corporation or other private group entity ☐ Government

4a. Fees submitted: ☒ Issue Fee ☐ Publication Fee (if required) ☐ Advance Order - # of Copies _____

4b. Method of Payment: *(Please first reapply any previously paid fee shown above)*

☒ Electronic Payment via EFS-Web ☐ Enclosed check ☐ Non-electronic payment by credit card (Attach form PTO-2038)

☒ The Director is hereby authorized to charge the required fee(s), any deficiency, or credit any overpayment to Deposit Account No. 06-0916

5. **Change in Entity Status** (from status indicated above)

☐ Applicant certifying micro entity status. See 37 CFR 1.29

☐ Applicant asserting small entity status. See 37 CFR 1.27

☐ Applicant changing to regular undiscounted fee status.

NOTE: Absent a valid certification of Micro Entity Status (see forms PTO/SB/15A and 15B), issue fee payment in the micro entity amount will not be accepted at the risk of application abandonment.
NOTE: If the application was previously under micro entity status, checking this box will be taken to be a notification of loss of entitlement to micro entity status.
NOTE: Checking this box will be taken to be a notification of loss of entitlement to small or micro entity status, as applicable.

NOTE: This form must be signed in accordance with 37 CFR 1.31 and 1.33. See 37 CFR 1.4 for signature requirements and certifications.

Authorized Signature /Christopher C. Johns/ Date 7 March 2023

Typed or printed name Christopher C. Johns Registration No. 68,664

UNITED STATES
PATENT AND TRADEMARK OFFICE

P.O. Box 1450
Alexandria, VA 22313 - 1450
www.uspto.gov

ELECTRONIC ACKNOWLEDGEMENT RECEIPT

APPLICATION #	RECEIPT DATE / TIME	ATTORNEY DOCKET #
17/657,981	**03/07/2023 03:37:14 PM ET**	**15635.0008-00000**

Title of Invention

Forward and Rearward Facing Attack Vector Visualization

Application Information

APPLICATION TYPE	Utility - Nonprovisional Application under 35 USC 111(a)	PATENT #	-
CONFIRMATION #	2397	FILED BY	Wanda Whittington
PATENT CENTER #	61702287	FILING DATE	04/05/2022
CUSTOMER #	22852	FIRST NAMED INVENTOR	Avi SHUA
CORRESPONDENCE ADDRESS	-	AUTHORIZED BY	Christopher Johns

Documents

TOTAL DOCUMENTS: 1

DOCUMENT	PAGES	DESCRIPTION	SIZE (KB)
15635.0008-00000 Issue Fee Transmittal-DRAFT.pdf	1	Issue Fee Payment (PTO-85B)	225 KB

Digest

DOCUMENT	MESSAGE DIGEST(SHA-512)
15635.0008-00000 Issue Fee Transmittal-DRAFT.pdf	6A53C5F30FF1A5F0B6D4C4EC1D77A871294471D140FE657BB A8F8393F78F61B4A4282B89141FCB1370FB86E646989942D5C 255B6D951A83813C28F0C55002893

This Acknowledgement Receipt evidences receipt on the noted date by the USPTO of the indicated documents, characterized

by the applicant, and including page counts, where applicable. It serves as evidence of receipt similar to a Post Card, as described in MPEP 503.

New Applications Under 35 U.S.C. 111

If a new application is being filed and the application includes the necessary components for filing date (see 37 CFR 1.53(b)-(d) and MPEP 506), a Filing Receipt (37 CFR 1.54) will be issued in due course and the date shown on this Acknowledgement Receipt will establish the filing date of the application

National Stage of an International Application under 35 U.S.C. 371

If a timely submission to enter the national stage of an international application is compliant with the conditions of 35 U.S.C. 371 and other applicable requirements a Form PCT/DO/EO/903 indicating acceptance of the application as a national stage submission under 35 U.S.C. 371 will be issued in addition to the Filing Receipt, in due course.

New International Application Filed with the USPTO as a Receiving Office

If a new international application is being filed and the international application includes the necessary components for an international filing date (see PCT Article 11 and MPEP 1810), a Notification of the International Application Number and of the International Filing Date (Form PCT/RO/105) will be issued in due course, subject to prescriptions concerning national security, and the date shown on this Acknowledgement Receipt will establish the international filing date of the application.

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
17/657,981	04/05/2022	Avi SHUA	15635.0008-00000	2397

22852 7590 02/15/2023

FINNEGAN, HENDERSON, FARABOW, GARRETT & DUNNER LLP
901 NEW YORK AVENUE, NW
WASHINGTON, DC 20001-4413

EXAMINER
SU, SARAH

ART UNIT	PAPER NUMBER
2431	

NOTIFICATION DATE	DELIVERY MODE
02/15/2023	ELECTRONIC

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Notice of the Office communication was sent electronically on above-indicated "Notification Date" to the following e-mail address(es):

regional-desk@finnegan.com

Corrected ***Notice of Allowability***	**Application No.** 17/657,981	**Applicant(s)** SHUA, Avi	
	Examiner SARAH SU	**Art Unit** 2431	**AIA (FITF) Status** Yes

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address--

All claims being allowable, PROSECUTION ON THE MERITS IS (OR REMAINS) CLOSED in this application. If not included herewith (or previously mailed), a Notice of Allowance (PTOL-85) or other appropriate communication will be mailed in due course. **THIS NOTICE OF ALLOWABILITY IS NOT A GRANT OF PATENT RIGHTS.** This application is subject to withdrawal from issue at the initiative of the Office or upon petition by the applicant. See 37 CFR 1.313 and MPEP 1308.

1. ☑ This communication is responsive to IDS filed 8 February 2023.
   ☐ A declaration(s)/affidavit(s) under **37 CFR 1.130(b)** was/were filed on _____.

2. ☐ An election was made by the applicant in response to a restriction requirement set forth during the interview on _____; the restriction requirement and election have been incorporated into this action.

3. ☑ The allowed claim(s) is/are 81-100 . As a result of the allowed claim(s), you may be eligible to benefit from the **Patent Prosecution Highway** program at a participating intellectual property office for the corresponding application. For more information, please see **http://www.uspto.gov/patents/init_events/pph/index.jsp** or send an inquiry to **PPHfeedback@uspto.gov.**

4. ☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

   Certified copies:

   a) ☐ All b) ☐ Some* c) ☐ None of the:

   1. ☐ Certified copies of the priority documents have been received.
   2. ☐ Certified copies of the priority documents have been received in Application No. _____ .
   3. ☐ Copies of the certified copies of the priority documents have been received in this national stage application from the International Bureau (PCT Rule 17.2(a)).

   * Certified copies not received: _____ .

Applicant has THREE MONTHS FROM THE "MAILING DATE" of this communication to file a reply complying with the requirements noted below. Failure to timely comply will result in ABANDONMENT of this application.
THIS THREE-MONTH PERIOD IS NOT EXTENDABLE.

5. ☐ CORRECTED DRAWINGS (as "replacement sheets") must be submitted.
   ☐ including changes required by the attached Examiner's Amendment / Comment or in the Office action of Paper No./Mail Date _____ .

   Identifying indicia such as the application number (see 37 CFR 1.84(c)) should be written on the drawings in the front (not the back) of each sheet. Replacement sheet(s) should be labeled as such in the header according to 37 CFR 1.121(d).

6. ☐ DEPOSIT OF and/or INFORMATION about the deposit of BIOLOGICAL MATERIAL must be submitted. Note the attached Examiner's comment regarding REQUIREMENT FOR THE DEPOSIT OF BIOLOGICAL MATERIAL.

Attachment(s)

1. ☐ Notice of References Cited (PTO-892)
2. ☑ Information Disclosure Statements (PTO/SB/08), Paper No./Mail Date _____.
3. ☐ Examiner's Comment Regarding Requirement for Deposit of Biological Material _____.
4. ☐ Interview Summary (PTO-413), Paper No./Mail Date. _____.
5. ☐ Examiner's Amendment/Comment
6. ☐ Examiner's Statement of Reasons for Allowance
7. ☐ Other _____.

/SARAH SU/
Primary Examiner, Art Unit 2431

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Use as many sheets as necessary)				Complete if Known	
				Application Number	17/657,981
				Filing Date	April 5, 2022
				First Named Inventor	Avi Shua
				Art Unit	2431
				Examiner Name	SU, SARAH
Sheet	1	of	1	Attorney Docket Number	15635.0008-00000

U.S. PATENTS

Examiner Initials*	Cite No.	Document Number Number-Kind Code (if known)	Issue Date MM-DD-YYYY	Name of Patentee or Applicant of Cited Document	Pages, Columns, Lines, Where Relevant Passages or Relevant Figures Appear

U.S. PUBLISHED PATENT APPLICATIONS

Examiner Initials*	Cite No.	Document Number Number-Kind Code (if known)	Publication Date MM-DD-YYYY	Name of Patentee or Applicant of Cited Document	Pages, Columns, Lines, Where Relevant Passages or Relevant Figures Appear
		US-2021/0112091-A1	04-15-2021	Compton	
		US-2021/0044563-A1	02-11-2021	Reyes et al.	
		US-2021/0365564-A1	11-25-2021	Ewaida et al.	
		US-2022/0046051-A1	02-10-2022	Stopel et al.	
		US-2009/0187989-A1	07-23-2009	Kim et al.	
		US-2004/0078384-A1	04-22-2004	Keir et al.	

Note: Submission of copies of U.S. Patents and published U.S. Patent Applications is not required.

FOREIGN PATENT DOCUMENTS

Examiner Initials*	Cite No.	Foreign Patent Document Country Code Number Kind Code (if known)	Publication Date MM-DD-YYYY	Name of Patentee or Applicant of Cited Document	Pages, Columns, Lines, Where Relevant Passages or Relevant Figures Appear	Translation
		WO-03060717-A1	07-24-2003	FOUNDSTONE, INC.		N/A
		WO-2021108569-A1	06-03-2021	RELIAQUEST HOLDING, LLC		N/A

NON PATENT LITERATURE DOCUMENTS

Examiner Initials*	Cite No.	Include name of the author (in CAPITAL LETTERS), title of the article (when appropriate), title of the item (book, magazine, journal, serial, symposium, catalog, etc.), date, page(s), volume-issue number(s), publisher, city and/or country where published.	Translation
		Notice of Allowance No. 17/657,972 dated February 1, 2023, in the United States Patent and Trademark Office. (15635.0005-00000)	N/A

Examiner Signature	/SARAH SU/	Date Considered	02/13/2023

*EXAMINER: Initial if reference considered, whether or not citation is in conformance with MPEP 609. Draw line through citation if not in conformance and not considered. Include copy of this form with next communication to applicant.

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.S/

INFORMATION DISCLOSURE STATEMENT BY APPLICANT *(Use as many sheets as necessary)*				***Complete if Known***	
				Application Number	17/657,981
				Filing Date	April 5, 2022
				First Named Inventor	Avi Shua
				Art Unit	2431
				Examiner Name	SU, SARAH
Sheet	1	of	1	*Attorney Docket Number*	15635.0008-00000

U.S. PATENTS

Examiner Initials*	Cite No.	Document Number Number-Kind Code *(if known)*	Issue Date MM-DD-YYYY	Name of Patentee or Applicant of Cited Document	Pages, Columns, Lines, Where Relevant Passages or Relevant Figures Appear

U.S. PUBLISHED PATENT APPLICATIONS

Examiner Initials*	Cite No.	Document Number Number-Kind Code *(if known)*	Publication Date MM-DD-YYYY	Name of Patentee or Applicant of Cited Document	Pages, Columns, Lines, Where Relevant Passages or Relevant Figures Appear
		US-2021/0112091-A1	04-15-2021	Compton	
		US-2021/0044563-A1	02-11-2021	Reyes et al.	
		US-2021/0365564-A1	11-25-2021	Ewaida et al.	
		US-2022/0046051-A1	02-10-2022	Stopel et al.	
		US-2009/0187989-A1	07-23-2009	Kim et al.	
		US-2004/0078384-A1	04-22-2004	Keir et al.	

Note: Submission of copies of U.S. Patents and published U.S. Patent Applications is not required.

FOREIGN PATENT DOCUMENTS

Examiner Initials*	Cite No.	Foreign Patent Document Country Code Number Kind Code *(if known)*	Publication Date MM-DD-YYYY	Name of Patentee or Applicant of Cited Document	Pages, Columns, Lines, Where Relevant Passages or Relevant Figures Appear	Translation
		WO-03060717-A1	07-24-2003	FOUNDSTONE, INC.		N/A
		WO-2021108569-A1	06-03-2021	RELIAQUEST HOLDING, LLC		N/A

NON PATENT LITERATURE DOCUMENTS

Examiner Initials*	Cite No.	Include name of the author (in CAPITAL LETTERS), title of the article (when appropriate), title of the item (book, magazine, journal, serial, symposium, catalog, etc.), date, page(s), volume-issue number(s), publisher, city and/or country where published.	Translation
		Notice of Allowance No. 17/657,972 dated February 1, 2023, in the United States Patent and Trademark Office. (15635.0005-00000)	N/A

Examiner Signature		Date Considered	

*EXAMINER: Initial if reference considered, whether or not citation is in conformance with MPEP 609. Draw line through citation if not in conformance and not considered. Include copy of this form with next communication to applicant.

(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
24 July 2003 (24.07.2003)

PCT

(10) International Publication Number
WO 03/060717 A1

(51) International Patent Classification[7]: G06F 11/30

(21) International Application Number: PCT/US02/01093

(22) International Filing Date: 15 January 2002 (15.01.2002)

(25) Filing Language: English

(26) Publication Language: English

(71) Applicant: **FOUNDSTONE, INC.** [US/US]; 2 Venture Street, Suite 100, Irvine, CA 92618 (US).

(72) Inventors: **MCCLURE, Stuart, C.**; 7 Welbe Circle, Ladera Ranch, CA 92694 (US). **KURTZ, George**; 19 Douglass Drive, Coto de Caza, CA 92679 (US). **KEIR, Robin**; 2718 Cipres, Mission Viejo, CA 92692 (US). **BEDDOE, Marshall, A.**; 306 Avenida Del Mar #2, San Clemente, CA 92672 (US). **MORTON, Michael, J.**; 6260 Via Ribazo, Anaheim Hills, CA 92807 (US). **PROSISE, Christopher, M.**; 27815 Tirante, Mission Viejo, CA 92692 (US). **COLE, David, M.**; 8171 Racepoint Drive #206, Huntington Beach, CA 92646 (US). **ABAD, Christopher**; 1007 Russia Avenue, San Francisco, CA 94112 (US).

(74) Agent: **DELANEY, Karoline, A.**; KNOBBE, MARTENS, OLSON AND BEAR, LLP, 620 Newport Center Drive, 16th Floor, Newport Beach, CA 92660 (US).

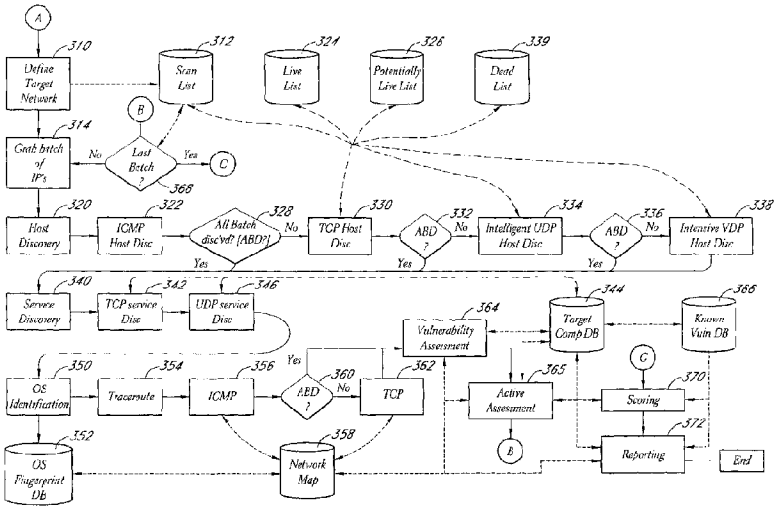
(81) Designated States *(national)*: AE, AG, AL, AM, AT (utility model), AT, AU, AZ, BA, BB, BG, BR, BY, BZ, CA, CH, CN, CO, CR, CU, CZ (utility model), CZ, DE (utility model), DE, DK (utility model), DK, DM, DZ, EC, EE (utility model), EE, ES, FI (utility model), FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ, NO, NZ, OM, PH, PL, PT, RO, RU, SD, SE, SG, SI, SK (utility model), SK, SL, TJ, TM, TN, TR, TT, TZ, UA, UG, UZ, VN, YU, ZA, ZM, ZW.

(84) Designated States *(regional)*: ARIPO patent (GH, GM, KE, LS, MW, MZ, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE, TR), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Published:
— *with international search report*

[Continued on next page]

(54) Title: SYSTEM AND METHOD FOR NETWORK VULNERABILITY DETECTION AND REPORTING



(57) Abstract: A system and method provide comprehensive and highly automated testing of vulnerabilities to intrusion on a target network (310), including identification of operating system, identification of target network topology and target computers, identification of open target ports, assessment of vulnerabilities (364) on target ports, active assessment of vulnerabilities based on information acquired from target computers (344), quantitative assessment of target network security and vulnerability, and hierarchical graphical representation of the target network, target computers, and vulnerabilities in a test report. The system and method employ minimally obtrusive techniques to avoid interference with or damage to the target network during or after testing.

WO 03/060717 A1

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

SYSTEM AND METHOD FOR NETWORK VULNERABILITY DETECTION AND REPORTING

Background of the Invention

Field of the Invention

[0001] This invention relates to network system security, and more particularly relates to systems and methods for automatic detection, monitoring and reporting of network vulnerabilities.

Description of the Related Art

[0002] The reliability and security of a network is essential in a world where computer networks are a key element in intra-entity and inter-entity communications and transactions. Various tools have been used by network administrators, government, security consultants, and hackers to test the vulnerabilities of target networks, such as, for example, whether any computers on a network can be accessed and controlled remotely without authorization. Through this intensive testing, a target network can be "hardened" against common vulnerabilities and esoteric attacks. Existing testing systems, however, produce inconsistent results, use techniques that are unproven or that damage the target network, fail to respond to changing network environments or to detect new vulnerabilities, and report results in difficult to understand, text-based reports.

[0003] Well-known network security tools now exist to test network paths for possible intrusion. From a testing point, simple commands such as traceroute and ping can be used to manually map a network topography, and determine roughly what network addresses are "alive" with a computer "awake" on the network (i.e., determine which computers are on and are responding to network packets). A tool such as a port scanner can be used to test an individual target computer on the target network to determine what network ports are open. If open ports are found, these ports may provide access for possible intrusion, and potentially represent a vulnerability that can be exploited by a malicious hacker.

[0004] Some suites combining various network tools attempt to follow a quasi-automated process to test target computers on a target network. These suites provide

variations on the tools described above, and provide long-form text-based output based on the outcome of this testing. The output of these security tests are extremely technical, and require extensive knowledge of network communications in order to interpret and provide advice based on the results. Thus, these partially automated suites do not provide comprehensive security to an entity seeking to "harden" its network.

[0005] Further, some security suites actually risk substantial damage to the target network. For example, while the use of malformed network packets to test a target computer can provide extensive information from the target and feedback on the security of the target, these malformed packets can destabilize the target computer in unpredictable ways. This sometimes results in a short-term loss of information to the target computer or, in more serious cases, a complete crash of the target computer operating system or hardware.

[0006] In other cases, the testing method used by existing suites is not reliable. If a network port scanning method employed on a target computer is, for example, 80% accurate over time, then a complete test of all 2^{16} ports on a single computer may result in approximately 13,000 ports incorrectly identified as potentially running vulnerable services. Over an entire target network, such "false positives" make it virtually impossible to determine the true security level of the target network.

[0007] Existing testing methods lack a standard, quantitative method for objectively comparing the security of a target network or target computer to other systems. Typically, a target network or target computer is ranked only as "high risk," "medium risk," or "low risk." However, such a three-tier system alone provides very little substantive feedback or comparative information about changes in the network over time, the relative weight of different vulnerabilities in determining the resulting risk level, or objective assessments of network security among otherwise heterogeneous network environment.

Summary of the Invention

[0008] The present invention solves these problems and more through a comprehensive network vulnerability testing and reporting method and system. Specifically, the testing system features include a selected combination of: (1) a non-destructive identification of target computer operating system; (2) a multiple-tier port scanning method for determination of what network addresses are active and what ports

are active at those addresses; (3) a comparison of collected information about the target network with a database of known vulnerabilities; (4) a vulnerability assessment of some vulnerabilities on identified ports of identified target computers; (5) an active assessment of vulnerabilities reusing data discovered from previously discovered target computers; (6) an application of a quantitative score to objectively and comparatively rank the security of the target network; and, (7) reduction of detailed results of the information collected into hierarchical, dynamic and graphical representations of the target network, target computers, and vulnerabilities found therein. Other features are foreseen and disclosed herein, as well.

[0009] In its preferred embodiment, the testing system operates over a modern multi-layer packet network such as a corporate intranet or the Internet. The network typically includes one or more computers, where a computer includes a desktop station running any operating system, a router, a server, and/or any other networked device capable of sending and receiving packets through standard internet protocols such as TCP/IP (Transmission Control Protocol/Internet Protocol), UDP (User Datagram Protocol), and the like. The system and method can be run remotely from a monitoring computer outside the target network, or can be run by a monitoring computer included within the target network. The target network itself is typically defined as an interconnected set of computers, bounded by a specific pre-designated sub-network address, range of IP addresses or sub-addresses, physical network boundaries, computer names or unique identifiers, presence or connection via a pre-determined network protocol, and the like. The target computers comprise all or a portion of the computers found within the target network. For example, a target computer with a simple connection to a WAN (Wide Area Network) can be tested remotely, as a single peer target network. In a more complicated example, a distributed network provider can have multiple sub-networks geographically distributed throughout the world but interconnected via an internal protocol, as a WAN target network with thousands of target computers.

[0010] A target network typically runs on one or more IP-based network protocols. Most commonly, the protocol will be TCP/IP and UDP. Similarly, the testing system is typically indifferent to the physical layer structure and topology of the target network. Only structural elements such as firewalls or routers that block, reroute, or change packets

will affect the testing system. The testing system, however, attempts to adapt to these structural elements and generally provides accurate results regardless of physical implementation.

[0011] TCP/IP is a fundamental protocol used for packet-based network communications on local area networks, wide area networks, and global telecommunications networks such as the Internet. A sample configuration of a TCP/IP SYN (synchronization) packet is shown in Table 1.

Source Port		*Destination Port*	
Sequence Number			
Acknowledgement Number			
Data Offset	*Reserved Data*	*Flags*	*Window*
Checksum		*Urgent Pointer*	
Options			*Padding*
Data			

Table 1: Typical TCP SYN packet

[0012] A computer typically runs on one or more operating systems. More commonly, these operating systems include those provided by Microsoft®, such as the Microsoft Windows® family of operating systems, MacOS® from Apple®, various flavors of UNIX including Linux®, NetBSD, FreeBSD, Solaris®, and the like. Additionally, devices on the target network may include router operating systems, mobile communication device operating systems, palmtop or handheld operating systems, appliance operating systems, set-top box operating systems, gaming operating systems, digital rights management systems, surveillance systems, smart card transaction systems, transportation management systems, and the like, that assign unique or temporary network addresses and are capable of sending and/or receiving traffic from the target network.

[0013] Target computers, in one embodiment, are identified by a unique or temporarily unique IP (Internet Protocol) address, typically in the form A.B.C.D, where each of A, B, C and D represent the Class A, Class B, Class C and Class D sub-networks and each has a value between 0 and 255. Typically, the target network is defined by one or more ranges of IP addresses controlled by the target network, but may contain

additional target computers or target sub-networks connected to the target network topographically but not part of the predetermined IP range or ranges.

[0014] UDP (User Datagram Protocol) is an alternative "connectionless" communications protocol that runs above IP (Internet Protocol). UDP lacks the error correction and receipt acknowledgment features of connection-based protocols such as TCP. ICMP (Internet Control Message Protocol) is another extension of IP which permits control communications (most commonly through a ICMP PING request) between hosts on an IP network.

[0015] Another aspect of the invention includes non-destructive and relatively non-intrusive identification of the target operating system of a target computer.

[0016] Another aspect of the invention includes parallel testing of multiple target computers on a target network.

[0017] Another aspect of the invention includes an improved testing method to determine whether particular target computers on a target network are alive.

[0018] Another aspect of the invention includes an improved method for determining whether a set of commonly used ports are open on a target computer.

[0019] Another aspect of the invention includes an improved method for reliably determining whether a set of commonly used UDP ports are open or closed on a target computer.

[0020] Another aspect of the invention includes a method for associating the ports found open on a target computer with a known set of vulnerabilities.

[0021] Another aspect of the invention includes parallel testing of multiple ports and multiple target computers simultaneously.

[0022] Another aspect of the invention includes active assessment of some known set of vulnerabilities at a target computer.

[0023] Yet another aspect of the invention includes application of an objective quantitative score to the vulnerabilities found on a target network.

[0024] Still another aspect of the invention includes compilation of a dynamic, graphical report representing the network topology, network computers, and network vulnerabilities in a hierarchical report including both overview and detail documents.

[0025] In one embodiment, the present invention is a system for determining an operating system of a target computer operably connected to a network. The system comprises (1) first and second data packets, the first and second data packets compliant with a protocol supported by the network, the first and second data packets transmitted via the network to the target computer; (2) first and second operating system fingerprints comprising data bits stored in a computer-readable medium, the first and second operating system fingerprints associated with a first operating system; (3) a first target computer fingerprint comprising data bits stored in a computer-readable medium, the first target computer fingerprint including a representation of at least a portion of data received in response to the transmission of the first data packet; (4) a second target computer fingerprint comprising data bits stored in a computer-readable medium, the second target computer fingerprint including a representation of at least a portion of data received in response to the transmission of the second data packet; and (5) fingerprint comparison instructions executable by a computer to compare the first operating system fingerprint and the first target computer fingerprint, to compare the second operating system fingerprint and the second target computer fingerprint, and to generate a result indicative of whether the first operating system was running on the target computer. In a preferred aspect, the invention further comprises: (6) a third data packet, the third data packet compliant with the protocol, the first range of bits of the third data packet representing a third parameter value different from the first and second parameter values, the third data packet transmitted via the network to the target computer; (7) a third operating system fingerprint comprising data bits stored in a computer-readable medium, the third operating system fingerprint associated with the first operating system, the third operating system fingerprint differing from the first and second operating system fingerprints; and (8) a third target computer fingerprint comprising data bits stored in a computer-readable medium, the third target computer fingerprint including a representation of at least a portion of data received in response to the transmission of the first data packet, the comparison instructions executable by a computer to compare the third operating system fingerprint and the third target computer fingerprint before generating the result. In a further preferred aspect, the invention further comprises: (9) fourth, fifth and sixth operating system fingerprints comprising data bits stored in a computer-readable medium,

the fourth, fifth and sixth operating system fingerprints associated with a second operating system, at least one of the fourth, fifth and sixth operating system fingerprints differing from a respective one of the first, second and third operating system fingerprints; the comparison instructions executable by a computer to compare the fourth operating system fingerprint and the first target computer fingerprint, to compare the fifth operating system fingerprint and the second target computer fingerprint, to compare the sixth operating system fingerprint and the third target computer fingerprint, and to generate a second result indicative of whether the second operating system was running on the target computer. Preferred aspects of this embodiment are ones wherein (10) the first parameter value is obtained by setting no bits, the second parameter value is obtained by setting one bit, and the third parameter value is obtained by setting two bits, or (11) wherein the first parameter value is 0, the second parameter value is 128, and the third parameter value is 128 plus a multiple of 256.

[0026] In another embodiment, the present invention is a system for determining an operating system of a target computer accessible via a network. The system comprises: (1) a plurality of data packets compliant with a protocol supported by the network, the plurality of data packets transmitted via the network to the target computer; (2) a first plurality of operating system fingerprints, each comprising data bits stored in a computer-readable medium, each associated with a first operating system; (3) a plurality of target computer fingerprints, each comprising data bits stored in a computer-readable medium, each including a representation of at least a portion of data received in response to the transmission of the plurality of data packets; and (4) fingerprint comparison instructions executable by a computer to compare the first plurality of the operating system fingerprint and the plurality of the target computer fingerprints, and to generate a result indicative of whether the first operating system was running on the target computer. A preferred aspect of the embodiment is one wherein the protocol is TCP/IP. Another preferred aspect of the embodiment further comprises (5) a second plurality of operating system fingerprints, each comprising data bits stored in a computer-readable medium, each associated with a second operating system, the fingerprint comparison instructions comparing the second plurality of the operating system fingerprints and the plurality of the target computer

fingerprints to generate a second result indicative of whether the second operating system was running on the target computer.

[0027] A further embodiment of the present invention is a method for determining an operating system of a target computer accessible via a network. The method comprises the steps of (1) transmitting to the target computer a plurality of data packets compliant with a protocol supported by the network; (2) generating a plurality of target computer fingerprints, each including at least a portion of data received via the network in response to the transmission of the plurality of data packets; (3) comparing the plurality of target computer fingerprints to a first set of predetermined operating system fingerprints, each of the first set of predetermined operating system fingerprints associated with a first operating system; and (4) generating a result indicative of whether the first operating system was running on the target computer. In a preferred aspect the embodiment comprises the further steps of (5) comparing the plurality of target computer fingerprints to a second set of predetermined operating system fingerprints, each of the second set of predetermined operating system fingerprints associated with a second operating system; and (6) generating a result indicative of whether the second operating system was running on the target computer. One preferred aspect of that embodiment is one wherein the protocol is TCP/IP and wherein the value of the MSS option of two of the plurality of data packets is divisible by 128. Another preferred aspect of that embodiment is one wherein a first of the plurality of data packets has a maximum segment size option of 0, wherein a second of the plurality of data packets has a maximum segment size option of 128, and wherein a third of the plurality of data packets has a maximum segment size option of 384.

[0028] A still further embodiment of the invention is a method for identifying an operating system of a target computer via a network, the method comprising the steps of: (1) sending a first data packet to the target computer via the network, the first data packet complying with a protocol of the network and having a first pattern of bits in a first range of bits; (2) generating a first response value representing at least a portion of data received via the network in response to the sending of the first data packet; (3) sending a second data packet to the target computer via the network, the second data packet complying with the protocol and having a second pattern of bits in a first range of bits, the second pattern

of bits different from the first pattern; (4) generating a second response value representing at least a portion of data received via the network in response to the sending of the second data packet; (5) sending a third data packet to the target computer via the network, the third data packet complying with the protocol and having a third pattern of bits in a first range of bits, the third pattern of bits different from the first or the second pattern; (6) generating a third response value representing at least a portion of data received via the network in response to the sending of the third data packet; (7) comparing the first response value to a first predetermined value associated with a first operating system; (8) comparing the second response value to a second predetermined value associated with the first operating system; (9) comparing the third response value to a third predetermined value associated with the first operating system; and (10) generating a value indicative of a relationship between the first operating system and the target computer. A preferred aspect of the embodiment comprises the further steps of: (11) comparing the first response value to a fourth predetermined value associated with a second operating system; (12) comparing the second response value to a fifth predetermined value associated with the second operating system; and (13) comparing the third response value to a sixth predetermined value associated with the second operating system. A preferred aspect of that embodiment is one wherein no bit is set in the first pattern of bits, wherein one bit is set in the second pattern of bits, and wherein two bits are set in the third pattern of bits. Another preferred aspect of that embodiment is one wherein the number of bytes in the second pattern of bits that have at least one bit set is greater than the number of bytes in the first pattern of bits that have at least one bit set, and wherein the number of bytes in the third pattern of bits that have at least one bit set is greater than the number of bytes in the second pattern of bits that have at least one bit set.

[0029] Yet another embodiment of the present invention is a system for determining whether a target computer is on a network, the system comprising: (1) a first set of port identifiers stored in a computer-readable medium, each of the first set of port identifiers representing a port used by computers to receive data packets compliant with a first protocol of the network, each of the first set of port identifiers representing a port associated with known network services; (2) a first set of data packets, each directed to a port represented by at least one of the first set of port identifiers, each of the first set of

data packets compliant with the first protocol and transmitted to the target computer via the network; (3) a first set of acknowledgement packets received via the network in response to the transmission of the first set of data packets, and (4) a list of host identifiers, each host identifier representing a computer on the network that transmits data in response to a packet sent to the respective computer, a host identifier representing the target computer added to the list of host identifiers if the first set of acknowledgment packets indicates a responsiveness of the target computer. An alternative preferred aspect of the embodiment further comprises: (5a) a second set of port identifiers stored in a computer-readable medium, each of the second set of port identifiers representing a port used by computers to receive data packets compliant with a second protocol of the network, each of the second set of port identifiers representing a port associated with known network services; (6a) a second set of data packets, each directed to a port represented by at least one of the second set of port identifiers, each of the second set of data packets compliant with the second protocol and transmitted to the target computer via the network, at least one of the second set of data packets including data associated with the known network services; (7a) a second set of acknowledgement packets received via the network in response to the transmission of the second set of data packets; and (8a) a host identifier representing the target computer added to the list of host identifiers if the second set of acknowledgment packets indicates a responsiveness of the target computer. A preferred aspect of that embodiment is one wherein the first protocol is TCP, wherein the second protocol is UDP, wherein the second set of acknowledgment packets is a nonzero set of UDP data response packets. Another alternative preferred aspect of the embodiment further comprises: (5b) a second set of port identifiers stored in a computer-readable medium, each of the second set of port identifiers representing a port used by computers to receive data packets compliant with a second protocol of the network, each of the second set of port identifiers representing a port associated with known network services; (6b) a second set of data packets, each directed to a port represented by at least one of the second set of port identifiers, each of the second set of data packets compliant with the second protocol and transmitted to the target computer via the network, at least one of the second set of data packets including data associated with the known network services; (7b) a second set of acknowledgement packets received via the network in

response to the transmission of the second set of data packets; and (8b) a host identifier representing the target computer added to a second list of host identifiers if the second set of acknowledgment packets does not indicate an unresponsiveness of the target computer, each of the second list of host identifiers representing a computer not known to be unresponsive. A preferred aspect of that embodiment is one wherein the first protocol is TCP, wherein the second protocol is UDP, wherein the second set of acknowledgment packets is an empty set of ICMP error packets. A further preferred aspect of either alternative embodiment further comprises: (9) a third set of data packets, each directed to a port represented by at least one of the second set of port identifiers, each compliant with the second protocol, the third set of data packets transmitted to the target computer throughout a predetermined maximum latency period; (10) a first response received first in time in response to the transmission of the third set of data packets; (11) a second response received second in time in response to the transmission of the third set of data packets, a time duration between the receipt of the first response and the receipt of the second response defining a target computer latency period. A further preferred aspect of the embodiment is one wherein each of the second set of data packets is transmitted continuously to the target computer for the duration of the target computer latency period.

[0030] A still further embodiment of the present invention is a system for testing the accessibility of a target computer via a network. The system comprises: (1) a set of port identifiers stored in a computer-readable medium, each of the set of port identifiers representing a UDP-compliant port, at least one of the port identifiers representing a port associated with known network services; (2) a set of UDP-compliant data packets, each associated with a port represented by at least one of the set of port identifiers, each of the UDP-compliant data packets transmitted continuously to the target computer for a duration approximately the same as the latency period of the target computer, at least one of the UDP-compliant data packets including data associated with the known network services; (3) a first list representing computers accessible via the network, the first list including the target computer if a nonzero set of UDP data response packets is received in response to the transmission of the data packets; and (4) a second list representing computers not known to be inaccessible via the network, the second list including the

target computer if an empty set of ICMP error packets is received in response to the transmission of the data packets.

[0031] Another embodiment of the present invention is a method for determining whether a target computer is accessible via a network. The method comprises the steps of: (1) identifying TCP ports; (2) sending first data packets to the TCP ports of the target computer, each of the first data packets compliant with TCP; (3) receiving first acknowledgment packets in response to the sending of the first data packets; and (4) adding a representation of the target computer to a list representing accessible computers if the first acknowledgment packets are nonzero. A preferred aspect of the embodiment comprises the further steps of: (5) identifying UDP ports associated with network services; (6) sending second data packets to the UDP ports of the target computer, at least one of the second data packets sent continuously to the target computer throughout a latency period of the target computer; (7) receiving second acknowledgment packets in response to the sending of the second data packets; and (8) adding a representation of the target computer to a list representing accessible computers if the second acknowledgment packets are nonzero UDP data response packets. A further preferred aspect of the embodiment comprises the further step of: (9) determining the latency period of the target computer by measuring the time between responses received in response to packets transmitted to the target computer. A further preferred aspect of the embodiment comprises the further step of: (10) adding a representation of the target computer to a list representing computers not known to be inaccessible via the network, the adding performed if the second acknowledgment packets comprise an empty set of ICMP error packets.

[0032] An additional embodiment of the present invention is a method for assessing the vulnerability of a target computer via a network. The method comprising the steps of: (1) discovering a set of responsive computers on a network by transmitting a set of ICMP packets, a set of TCP packets and a set of UDP packets to a group of computers on a network; (2) detecting services on each of the set of responsive computers by transmitting TCP packets to first ports of each of the set of responsive computers and by transmitting UDP packets to second ports of each of the set of responsive computers, the first and second ports commonly used by computers to receive data packets over a network, the

TCP packets including data associated with at least one computer-based service known to use one of the first ports, the UDP packets including data associated with at least one computer-based service known to use one of the second ports; and (3) generating a list of responsive ports using responses received in response to the transmission of the TCP packets and the UDP packets. A preferred aspect of the embodiment comprises the further step of: (4) determining an operating system used by each of the set of responsive computers by comparing predetermined values with portions of responses received from each of the set of responsive computers in response to transmission of a plurality of TCP-compliant packets to each of the set of responsive computers. A further preferred aspect of the embodiment comprises the further step of: (5) confirming the presence of vulnerabilities on the network by applying an automated vulnerability script to each responsive port represented in the list of responsive ports, each of the automated vulnerability scripts testing a vulnerability known to be associated with a computer configuration comprising a particular responsive port and a particular operating system. A still further preferred aspect of the embodiment comprises the further step of: (6) calculating an objective indicia of security of the network, the calculation based on a weighted summation of confirmed vulnerabilities. A preferred aspect of the embodiment comprises the further step of: (7) determining a topology of the network, the topology determination made by transmitting a set of ICMP packets with varying time to live (TTL) settings and by transmitting a set of TCP packets with varying TTL settings. Another preferred aspect of the embodiment comprises the further step of: (8) producing a graphical representation of the network, the representation including a topological map of the network, a color-based representation of weighted confirmed vulnerabilities, and an association between the graphical representation and information descriptive of confirmed vulnerabilities and computers on the network.

[0033] Another embodiment of the present invention is a method for creating a topological representation of a network. The method comprises the steps of: (1) identifying responsive computers on the network; (2) obtaining a plurality of sequences of IP addresses by sending to each responsive computer a sequence of packets having increasing TTL values, each sequence of IP addresses representing nodes in the network between a source computer and one of the responsive computers, adjacent IP addresses in

each sequence representing connected nodes, each of the nodes comprising a computer or a router; (3) generating a list of node structures, each of the node structures including data representing a node and data indicative of other nodes to which it directly connects, the list representing all IP addresses in the plurality of sequences; (4) determining for each IP address a distance count, the distance count representing a number of nodes between a node having the IP address and a source node; (5) creating a router structure for each node structure that represents a node comprising a router; (6) associating with each of the router structures connection data representative of each connecting node that connects to no other node except the router represented by the respective router structure; (7) for each router structure, visually depicting a graphical shape spatially related to one or more graphical shapes corresponding to connecting nodes represented by the connection data of the respective router structure; and (8) for each router structure, visually depicting a connection between a graphical shape associated with the respective router structure and another graphical shape associated with a different router structure when distance counts associated with the IP addresses of routers represented by the respective router structure and the different router structure indicate a direct connection. A preferred aspect of the embodiment comprises the further step of: (9) testing whether a router represented by a router structure and a connecting node represented in connection data comprise two network connections of one node. A further preferred aspect of this embodiment is one wherein the graphical shape representing a router is a sphere, and wherein each of the spatially related graphical shapes is a sphere orbiting the sphere representing the router.

[0034] Yet another embodiment of the present invention is a method for calculating an objective security score for a network. The method comprising the steps of: (1) determining a vulnerability value numerically representing a combination of known vulnerabilities of a network; (2) determining an exposure value numerically representing a combination of accessible ports of computers on the network; and (3) deriving a score by combining the vulnerability value and the exposure value. A preferred aspect of this embodiment is one wherein the combination of known vulnerabilities is a summation of weighted numeric expressions of particular vulnerabilities, the weighting based on an ease of exploitation ranking and on an access granted ranking for each vulnerability.

[0035] Still another embodiment of the present invention is a method for conducting an automated network vulnerability attack, the method comprising the steps of: (1) selecting a set of vulnerability attacks for each responsive computer on a network, each selected vulnerability attack for each responsive computer designed to expose a vulnerability associated with ports of the respective computer known to be accessible and also associated with an operating system used by the respective computer; (2) encoding the set of vulnerability attacks such that each is represented in a database by a unique identifier; (3) representing each of the set of vulnerability attacks using instructions of an automated scripting language; and (4) executing the vulnerability attacks by processing the instructions with a computer.

[0036] One more embodiment of the present invention is a hierarchical network vulnerability report. The report comprises: (1) a first report level comprising: (a) an objective score representing the security of the network; and (b) a graphical representation of a network topology, including a graphical representation of computers accessible via the network and a color-based graphical representation of the vulnerability of at least some of the computers; and (2) a second report level comprising: (a) a textual list describing the computers and their associated vulnerabilities; and (b) an exposure report describing accessible ports and services of the computers.

[0037] An additional embodiment of the present invention is a vulnerability assessment language. The vulnerability assessment language comprises: (1) a set of programming language statements used to create executable scripts, the scripts executed in a thread-safe execution architecture wherein all variables are stack variables and wherein a parse tree is treated as a read-only data structure; (2) a set of special scalar data types interchangeable with an integer data type in expressions, each of the set of special scalar data types having a set of constant values configured to support vulnerability assessment operations embodied in scripts; (3) a set of native objects declared in a metascope owning a script scope to make available the native objects to executable scripts, the native objects facilitating network communication, providing callable member functions for building lists of unique ports and directing script execution to certain hosts, and providing IP addresses for scripts; and (4) a vulnerability object behaving to copy itself into a global data area where other scripts may access its information to compromise

another machine, facilitating the use by one script of vulnerability data discovered by a different script.

[0038] A further embodiment of the present invention is a method for automated application of a known vulnerability on a target computer. The method comprises the steps of: (1) providing a database of known vulnerabilities, the database including a data object; (2) providing an executable script, the executable script associated with the data object; (3) applying the executable script to the target computer, the script performing the known vulnerability on a port of the target computer; and (4) returning a value representing at least one of the success, failure or other outcome of the executable script.

[0039] A still further embodiment of the present invention is a method for automated application of known vulnerabilities to target computers of a network. The method comprises the steps of: (1) providing a database of known vulnerabilities; (2) providing a set of executable scripts, each executable to apply a known vulnerability to a specified target computer; (3) executing first executable scripts to apply vulnerabilities on specified target computers; (4) monitoring return values representing a success, failure or other outcome of each of the first executable scripts; and (5) generating a report using the return values, the report representing a security level of the network. One preferred aspect of this embodiment comprises the further step of: (6) identifying execution time intervals wherein execution of the first executable scripts commences at the beginning of each of the time intervals and pauses at the end of each of the time intervals, until all of the first executable scripts have executed. A preferred aspect of the embodiment comprises the further step of: (7) automatically repeating the execution of the first executable scripts when the execution of the first executable scripts is completed. Another preferred aspect of the embodiment comprises the further steps of: (8) generating a report upon each completed execution of the first executable scripts; and (9) calculating a security trend for the network by comparing a plurality of the reports. An alternative preferred aspect of the embodiment comprises the further step of: (10) executing second executable scripts to apply vulnerabilities to a second network of computers during the execution of the first executable scripts. Another preferred aspect of the embodiment is one wherein the second network is a subset of the network. Still another preferred aspect of the embodiment is one wherein the first executable scripts are configured to apply

vulnerabilities to a first port of all of the target computers before applying vulnerabilities to a second port of all of the target computers. An additional preferred aspect of the embodiment comprises the further step of allocating a plurality of packet slots, each packet slot permitting asynchronous transmission of a packet by one of the executable scripts.

Brief Description of the Drawings

[0040] Preferred embodiments of the present invention are described below in connection with the attached drawings in which:

[0041] **Figure 1** illustrates one embodiment of a target network;

[0042] **Figure 2** illustrates one embodiment of a target computer on the target network;

[0043] **Figure 3** illustrates one embodiment of a comprehensive testing method;

[0044] **Figure 4** illustrates one embodiment of the operating system identification method;

[0045] **Figure 5** illustrates one example embodiment of the TCP SYN packet used in the operating system identification method of Figure 3;

[0046] **Figure 6** illustrates one embodiment of first phase scanning to determine what target computers are alive;

[0047] **Figure 7** illustrates one embodiment of second phase scanning to determine what ports are open on a target computer;

[0048] **Figure 8** illustrates one embodiment of active assessment of a vulnerability of a target computer on a target network;

[0049] **Figure 9** illustrates one embodiment of a methodology for determining the security score for a target network;

[0050] **Figure 10** illustrates one embodiment of a hierarchical security report, including a graphical representation of network topology and network vulnerabilities; and

[0051] **Figure 11** illustrates a second embodiment of a hierarchical security report in greater detail.

[0052] **Figure 12** illustrates a second embodiment of the comprehensive testing method.

Detailed Description of the Preferred Embodiment

I. Basic Implementation, Structure and Control Language

[0053] Figure 1 illustrates one embodiment of a target network. The network security system 100 of the present invention is, in one embodiment, at least one Intel-based server running on a Windows 2000 operating system, although any computer system or operating system capable of handling an IP network and capable of large-scale data processing can be used. The network security system 100 may be outside the target network 102 or inside the target network (not shown). In either case, the system 100 is connected to the target network 102 through a network such as the Internet, via one or more nodes 104. The target network 102, in one example, consists of an intranet with a central intranet hub 106. The target network 102 further includes a firewall 108 which blocks some incoming traffic to or outgoing network traffic leaving the target network 102. The target network further comprises a number of hosts 110, defined as within a predetermined range of Internet Protocol (IP) addresses. In some cases, external hosts 112 may lie outside the target network but may nonetheless be connected to the target network 102.

[0054] Figure 2 illustrates one embodiment of a target computer on the target network. In general, a host IP address represents a target computer, as more generally defined below, if the address is in use by the target network. In a simplified representation of a target computer 200 at a host 110, the target computer 200 is running an operating system 202. The operating system preferably contains at least one network TCP/IP stack 204 to provide packet transport, preferably including an interface to provide raw socket 206 connections between the target computer 200 and the network. The physical connection to the network 208 is provided, in one embodiment, by a Network Interface Card (NIC) 210. On an IP network, a packet can be received at any one of 65,536 logical ports 212 at the target computer 200. Similarly, any number of network services 214 can be provided.

[0055] Figure 3 illustrates one embodiment of a comprehensive testing method in accordance with embodiments of the present invention. Figure 3 presents an overview of the method. Additional details of the method are set forth below in connection with Figures 4-11.

[0056] In a first step or routine 310 in Figure 3, the testing method defines the target network and creates a scan list of IP addresses. The scan list is stored in a scan list database 312. Then, in a routine 314, the method obtains a first batch of IP addresses from the scan list database 312 and begins a detailed analysis of the target network by performing a host discovery routine that begins in a block 320.

[0057] The host discovery routine comprises an ICMP host discovery routine 322, which will be described in more detail below. During the ICMP host discovery routine 322, the method pings the target computers identified by the present batch of IP addresses. Based on the responses or lack of responses, the ICMP host discovery routine 322 is able to determine that certain target computers are "live" or "probably live," and the associated IP addresses are added to a respective live database 324 or probably live database 326 accordingly.

[0058] After completing the ICMP host discovery routine 322, the method performs a decision routine 328 wherein the method determines whether all the IP addresses in the current batch of IP addresses have been discovered (i.e., whether all the IP addresses have been assigned to the live database 324 or the probably live database 326. If any IP address has not been discovered, the method proceeds to a TCP host discovery routine 330, which will be described in more detail below. During the TCP host discovery routine 330, the method sends TCP packets to the remaining target computers identified by the present batch of IP addresses. Based on responses or lack of responses, the TCP host discovery routine 330 is able to determine that certain ones of the remaining target computers are "live" or "probably live," and the associated IP addresses are added to a respective live database 324 or probably live database 326 accordingly.

[0059] After completing the TCP host discovery routine 330, the method performs a decision routine 332 wherein the method determines whether all the IP addresses in the current batch of IP addresses have been discovered (i.e., whether all the IP addresses have been assigned to the live database 324 or the probably live database 326. If any IP address has not been discovered, the method proceeds to an intelligent UDP host discovery routine 334, which will be described in more detail below. During the intelligent UDP host discovery routine 334, the method sends UDP packets to the remaining target computers identified by the present batch of IP addresses. Based on

responses or lack of responses, the intelligent UDP host discovery routine 334 is able to determine that certain ones of the remaining target computers are "live" or "probably live," and the associated IP addresses are added to a respective live database 324 or probably live database 326 accordingly.

[0060] After completing the intelligent UDP host discovery routine 334, the method performs a decision routine 336 wherein the method determines whether all the IP addresses in the current batch of IP addresses have been discovered (i.e., whether all the IP addresses have been assigned to the live database 324 or the probably live database 326. If any IP address has not been discovered, the method, in one embodiment, proceeds to an intensive UDP host discovery routine 338, which will be described in more detail below. During the intensive UDP host discovery routine 338, the method sends additional UDP packets to the remaining target computers identified by the present batch of IP addresses. Based on responses or lack of responses, the intensive UDP discovery routine 338 is able to determine that certain ones of the remaining target computers are "live" or "probably live," and the associated IP addresses are added to a respective live database 324 or probably live database 326 accordingly. The intensive UDP host discovery routine 338 is optional and may not be included in all embodiments.

[0061] After completing the intensive UDP host discovery routine 338, the method preferably proceeds to a service discovery routine that begins in a block 340. Alternatively, in certain embodiments, the foregoing host discovery routines 322, 330, 334, 338 are advantageously repeated to determine whether additional IP addresses corresponding to target computers can be assigned to the live database 324 and the probably live database 326. In such alternative embodiments, the method repeats the host discovery routines a maximum of a predetermined times before continuing to the service discovery routine 340. Those IP addresses for which no response is received by any method are, in one embodiment, added to a dead list 339 of hosts.

[0062] In the service discovery routine, the method performs a TCP service discovery routine 342, which will be described in more detail below. In the TCP service discovery routine 342, the method sends TCP packets to selected ports of the discovered target computers in the live list 324 and the probably live list 326 and monitors the responses. Based on the responses or lack of responses, the TCP discovery routine 342 adds

information regarding open ports of the target computers to a target computer database 344.

[0063] After performing the TCP service discovery routine 342, the method performs a UDP service discovery routine 346, which will be described in more detail below. In the UDP service discovery routine 346, the method sends UDP packets to selected ports of the discovered target computers in the live list 324 and the probably live list 326 and monitors the responses. Based on the responses or lack of responses, the UDP discovery routine 346 adds information regarding open ports of the target computers to the target computer database 344. In alternative embodiments, the TCP service discovery routine 342 and the UDP service discovery routine 346 are advantageously repeated a limited number of times to determine whether additional open ports are discovered.

[0064] After completing the UDP service discovery routine 346 (or after completing the routine a limited number of times), the method proceeds to an operating system (OS) identification routine 350 wherein the method determines the type and version of operating system present on each of the live computers having open ports. As will be described in more detail below, the method sends two or more (preferably three) RFC compliant TCP packets to the target computers. The TCP packets have predetermined data in at least one selected field. Each target computer responds to each of the TCP packets. The response information from each computer is treated as a "fingerprint" for each computer. The fingerprint from each computer is compared with fingerprints in fingerprint database 352, and the results of the comparisons are used to identify the target computers with high degrees of accuracy.

[0065] After completing the operating system identification routine 350 for the target computers, the method proceeds to a traceroute routine that begins in a block 354. In the traceroute routine, the method first performs an ICMP traceroute routine 356 in which the method uses ICMP tracerouting techniques, described in more detail below. In particular, the method sends a plurality of ICMP echo request packets to the target computer with varying TTL (time to live) values in the TCP/IP header. The method creates a network topology based on the known TTL value, the number of "hops" between the system and the target computer, and the router/host at each "hop." The information from the ICMP traceroute routine 356 is added to a network map database 358. When all packets sent

have either arrived, failed to arrive, or timed out, the traceroute step is completed for that target computer. The ICMP traceroute steps are repeated until a complete trace is received or a selected predetermined maximum number of passes is completed.

[0066] After completing the predetermined number of ICMP traceroute passes, the method proceeds to a decision routine 360 wherein the method determines whether the trace of the target computer is complete or incomplete. If the trace is incomplete, the method proceed to a TCP traceroute routine 362. Otherwise, the method bypasses the TCP traceroute routine and proceeds directly to a vulnerability assessment routine 364.

[0067] As described in more detail below, the TCP traceroute routine 362 works similarly to ICMP traceroute routine 354, except that TCP SYN packets are sent to the target computers. As with ICMP tracerouting, the TTL value in each SYN packet is incrementally increased, and the return of ICMP unreachable packets and SYN ACK packets is monitored for all "hops" between the scanning system and the target host. Through the combination of ICMP tracerouting and TCP tracerouting, a complete map to each target computer, and collectively a relatively complete map of the target network topology is advantageously created and stored in the network map database 358.

[0068] After completing the TCP traceroute routine 362, the method proceeds to the vulnerability assessment routine 364. As described in more detail below, in the vulnerability assessment routine 364, the method executes vulnerability scripts that apply known vulnerabilities to open ports of the live target computers to determine whether the ports of the target computers exhibit the potential vulnerabilities. The method uses information stored in a known vulnerability database 366 to select the vulnerability scripts to executes for each active port. Information collected from vulnerable target computers is advantageously stored to the target computer database 344.

[0069] In one embodiment, the vulnerability assessment routine 364 preferably only performs vulnerability checks associated with the identified operating system and open ports of the target computer as determined by the operating system identification routine 350 and service discovery routine 340. If the operating system is not conclusively identified, typically the routine runs all known vulnerabilities for the open ports of the target computer.

[0070] In one embodiment, the method proceeds to an active assessment routine 365. As further described in detail below, the active assessment routine 365 uses information collected from target computers in the vulnerability assessment routine 364 to execute further vulnerability scripts to open ports of the live target computers. Specifically, the active assessment routine 365 reuses known vulnerabilities and information collected from target computers to continue to determine whether the ports of the target computers exhibit potential vulnerabilities, using information from the known vulnerability database 366 and information collected in the target computer database 344.

[0071] After completing the active assessment routine 365, the method proceeds to a decision routine 368 to determine whether the method has analyzed all the potential target computers. In particular, in the decision routine 368, the method determines whether the last batch of IP address has been analyzed. If further batches of target computers remain to be analyzed, the method proceeds to the routine 314 to obtain the next batch of IP addresses. Otherwise, the method proceeds to a scoring routine 370.

[0072] In the scoring routine 370, described in more detail below, the method establishes a vulnerability score for each target computer and for the network based on the results of the active assessment and based on the vulnerability information in the known vulnerability database 366. The method then proceeds to a reporting routine 372, also described in more detail below, wherein the method reports the results of the scanning, active assessment and scoring.

[0073] The method can be advantageously repeated continuously on a target network, and can be scheduled into pre-determined scanning window time periods for execution of the method over time. For example, in preferred embodiments, the method is scheduled to scan during off-peak hours when the network is less likely to be less heavily used. In particularly preferred embodiments, the method is interruptible at the end of a window of off-peak hours and resumes where it paused at the beginning of the next off-peak window. In particularly advantageous embodiments, the method operates on multiple target networks concurrently by threading to share network resources.

II. Non-Destructive Operating System Identification

[0074] Vulnerability of and access to a target computer may be heightened by knowing which particular operating system is running on the computer. Identifying a

target computer's operating system can be accomplished by examining the operating system's response to a data packet it receives over the network. The forms of the packets and the responses to the packets are generated in accordance with network protocols. The written definitions of the protocols used for communications on the Internet are set forth in Internet Request for Comment (RFC) documents. For example, the TCP/IP protocol is defined in part in RFC 793, incorporated herein by reference, and contains a standard model for TCP packet communications over a network. While virtually every operating system includes a TCP/IP stack, each TCP/IP stack is implemented in a slightly different manner. Thus, known responses from a TCP/IP stack unique to a particular operating system can serve as a "fingerprint" to determine the target computer's actual operating system.

[0075] Operating system detection traditionally has been performed by sending out a combination of RFC-compliant and non-RFC compliant TCP packets. The traditional system then collects the unique, non-standard response from the target computer and maps the response to a known database of TCP/IP stacks and related operating systems. However, this method tends to be inaccurate, highly dependent on the particular packet shape and TCP/IP stack of the target computer, and requires a large number of packets to identify the target operating system with any degree of reliability. This method may trigger security or firewall alarms at the target computer, and, more seriously, this method may interfere with or actually crash the target computer, in large part due to the non-RFC compliant packets sent to the target.

[0076] The present system typically employs a unique set of new features to maximize the accuracy of operating system detection while minimizing intrusiveness and interference with operations of the target computer. In one embodiment, the invention sends RFC-compliant TCP "SYN" (synchronization) packets to a target computer. The use of RFC-compliant TCP packets advantageously reduces the probability that the detection packets are blocked by a router or firewall, and greatly reduces the probability that the detection packets will cause damage or crashes at the target computer. In one particularly preferred embodiment, the invention uses just three RFC-compliant TCP packets. Thus, network strain is significantly reduced during detection of the operating systems of a large number of target computers on a target network.

[0077] In one embodiment, the first packet sent is a completely standard TCP SYN to an open port on the target computer. The MSS (maximum segment size) option in the options field of the first packet is set to 0 (i.e., no bits set in the MSS option). See Figure 5. When an acknowledgement packet, a SYN ACK packet, is received by the detection system from the target computer, certain bits from the packet are saved by the system. In one embodiment, for example, the TCP advertised window, TTL (time-to-live), options, flags, and the DF (don't fragment) fields are saved to a first fingerprint.

[0078] In this embodiment, a second packet is then sent. The second TCP SYN packet is again a standard TCP SYN packet; however, the MSS option is set to 128 in the second packet (corresponding to the setting of a single bit in the MSS option). Portions of the response SYN ACK from the target computer (preferably the TCP advertised window, TTL, and DF bits) are again saved to a second fingerprint. Finally, a third TCP SYN packet is sent. The third packet is also a standard TCP SYN packet; however, the MSS option is set to 384 in the third packet (corresponding to the setting of two bits in the MSS option). Again, portions of the response SYN ACK from the target computer (preferably the TCP advertised window, TTL, and DF bits) are once again saved to a third fingerprint.

[0079] In one embodiment, the fingerprint is saved in the following format:

$$\mathbf{AW_{MSS=0}:AW_{MSS=128}:AW_{MSS=384}:TTL:DF:OS}$$

where, for example,

AW = TCP advertised window

MSS = TCP Options Maximum Segment Size

TTL = TCP Options Time to Live

DF = TCP Options Don't Fragment Flag, and

OS = Operating System identification.

[0080] In another embodiment, the fingerprint is saved in the following format:

$$\mathbf{AW_{MSS=0}:AW_{MSS=128}:AW_{MSS=384}:OPT_{MSS=384}:OPT_{MSS=0}:OPT_{MSS=128}:TTL:DF:FL:OS}$$

where, for example,

OPT = TCP Options Bytes, and

FL = TCP Flags.

[0081] The fingerprints are then compared to a known database of fingerprints associated with various operating systems and operating system versions. Known

fingerprints can be compiled through application of the above methodology to various target computers known to have a particular operating system before testing. For example, testing of known computers running various versions of the Solaris® operating system provide the following operating system fingerprints:

AW_0	AW_{128}	AW_{384}	OPT_0	OPT_{128}	OPT_{384}	TTL	DF	Flags	OS
83E8	8380	8400	02040218	02040080	02040180	FF	0	SA--	Solaris 2.6
6050	6000	6000	02040564	02040564	02040564	40	0	SA--	Solaris 2.7
6050	6000	6000	020405B4	020405B4	020405B4	40	0	SA--	Solaris 8
4000	4000	4080	-	-	-	64	0	-	OpenBSD 2.x
4000	4000	4000	-	-	-	64	0	-	NetBSD 1.4.2

Table 2: Sample OS Fingerprints for Solaris and BSD operating systems

[0082] While more than one OS fingerprint may be associated with each operating system, collisions between fingerprints of distinct operating systems have been found to be highly unlikely. Tables can be compiled for other operating systems similar to that shown in Table 2. As operating system versions and popularity change over time, the fingerprint database is advantageously regularly updated to account for patches, version changes, and new operating systems. The fingerprint style shown above is only one embodiment of such a database, and any efficient method to store the operating system fingerprint can be used, based upon the TCP options altered, number of packets typically sent to a target computer, other TCP fields stored for recognition, and identification field used to represent a particular operating system and version, and the like. In one example, a unique data string for a particular operating system is compressed and stored using a digest algorithm such as MD5, and the like. For further example, perfect matching of fingerprints is not required: a system may employ a percentage match, such as, for example, 90% similarity between two fingerprints, as sufficient to identify a target computer as having a particular operating system or at least being in particular family of operating systems.

[0083] Below is an example exchange of packets when performing an OS identification. Three standard TCP SYN packets are sent to the remote host. The first packet is a SYN packet with no data and no IP or TCP options. Packet 2 is also a TCP SYN packet but a TCP Maximum Segment Size of 128 in the TCP options field is set. The third and final packet is again a TCP SYN packet but a TCP Maximum Segment Size of 384 is set in the TCP options field.

[0084] As noted above, by analyzing the replies from the 3 packets, a fingerprint is assembled that appears in a textual format as follows:

window1:window2:window3:options1:options2:options3:ttl:dontfrag:flags

where,

window1 = is the TCP window size received from the 1st response

window2 = is the TCP window size received from the 2nd response

window3 = is the TCP window size received from the 3rd response

options1 = the option bytes received from the 1st response

options2 = the option bytes received from the 2nd response

options3 = the option bytes received from the 3rd response

ttl = is the IP TTL field from the 1st response

dontfrag = is the IP dontFragment bit from the 1st response, and

flags = are single character representations of the TCP flags from 1st response.

[0085] In the example TCP packets shown below, the resultant fingerprint looks like this:

40E8:4000:4080:020405B4:020405B4:020405B4:80:1:SA----

[0086] From this, the fingerprint is compared to a database of known operating system fingerprints in order to find the closest match that will identify the remote operating system. In this example, three example TCP packets sent and three example TCP packets returned are shown below:

```
----------
     Packet 1 send

     TCP
          Source port: 272
          Destination port: 80
          Sequence: 0x01100000 (17825792)
          Acknowledgement: 0x00000000 (0)
          Header length: 0x05 (5) - 20 bytes
          Flags: SYN
               URG: 0
               ACK: 0
               PSH: 0
               RST: 0
               SYN: 1
               FIN: 0
          Window: 0x0040 (64)
          Checksum: 0x4518 (17688) - correct
```

```
          Urgent Pointer: 0x0000 (0)
          TCP Options: None
          Data length: 0x0 (0)
-----
     Packet 1 reply

     TCP
          Source port: 80
          Destination port: 272
          Sequence: 0x659A2C81 (1704602753)
          Acknowledgement: 0x01100001 (17825793)
          Header length: 0x06 (6) - 24 bytes
          Flags: SYN ACK
          URG: 0
          ACK: 1
          PSH: 0
          RST: 0
          SYN: 1
          FIN: 0
          Window: 0xE840 (59456)
          Checksum: 0x9A47 (39495) - correct
          Urgent Pointer: 0x0000 (0)
          TCP Options
          Maximum Segment Size: 0x5B4 (1460)
          Data length: 0x0 (0)
-----
     Packet 2 send

     TCP
          Source port: 528
          Destination port: 80
          Sequence: 0x03100000 (51380224)
          Acknowledgement: 0x00000000 (0)
          Header length: 0x07 (7) - 28 bytes
          Flags: SYN
          URG: 0
          ACK: 0
          PSH: 0
          RST: 0
          SYN: 1
          FIN: 0
          Window: 0x0040 (64)
          Checksum: 0x1E8A (7818) - correct
          Urgent Pointer: 0x0000 (0)
          TCP Options
          Maximum Segment Size: 0x80 (128)
          Data length: 0x0 (0)
------
     Packet 2 reply

     TCP
```

```
          Source port: 80
          Destination port: 528
          Sequence: 0x659ABB23 (1704639267)
          Acknowledgement: 0x03100001 (51380225)
          Header length: 0x06 (6) - 24 bytes
          Flags: SYN ACK
          URG: 0
          ACK: 1
          PSH: 0
          RST: 0
          SYN: 1
          FIN: 0
          Window: 0x0040 (64)
          Checksum: 0x098D (2445) - correct
          Urgent Pointer: 0x0000 (0)
          TCP Options
          Maximum Segment Size: 0x5B4 (1460)
          Data length: 0x0 (0)
-------
     Packet 3 send

     TCP
          Source port: 784
          Destination port: 80
          Sequence: 0x05100000 (84934656)
          Acknowledgement: 0x00000000 (0)
          Header length: 0x07 (7) - 28 bytes
          Flags: SYN
               URG: 0
               ACK: 0
               PSH: 0
               RST: 0
               SYN: 1
               FIN: 0
          Window: 0x0040 (64)
          Checksum: 0x1A8A (6794) - correct
          Urgent Pointer: 0x0000 (0)
          TCP Options
               Maximum Segment Size: 0x180 (384)
          Data length: 0x0 (0)
----------
     Packet 3 reply

     TCP
          Source port: 80
          Destination port: 784
          Sequence: 0x659B732B (1704686379)
          Acknowledgement: 0x05100001 (84934657)
          Header length: 0x06 (6) - 24 bytes
          Flags: SYN ACK
               URG: 0
```

```
            ACK: 1
            PSH: 0
            RST: 0
            SYN: 1
            FIN: 0
      Window: 0x8040 (32832)
      Checksum: 0x4E04 (19972) - correct
      Urgent Pointer: 0x0000 (0)
      TCP Options
            Maximum Segment Size: 0x5B4 (1460)
      Data length: 0x0 (0)
------------
```

[0087] While any number of iterations of the fingerprints described above can be derived for a target computer, it has been determined that three fingerprints provide the most accurate identification of operating system without undue repetitiveness. Similarly, while other TCP options flags may be altered to detect target operating systems, it has been found that alteration of the TCP advertised window, over a plurality of test SYN packets, is most effective, preferably with three test SYN packets with TCP option MSS values of 0, 128 and 384 respectively. Furthermore, alteration of the MSS value in the TCP advertised window is also less likely to interfere with target computer operation than alteration of other values in the packets.

[0088] The foregoing operating system identification method is summarized in Figure 4. As described above, in accordance with the method of Figure 4, a network security system 410 first sends a first RFC compliant TCP SYN packet to a target computer 412 via a first packet transmission represented by a line 420. The first TCP SYN packet has the TCP Options Maximum Segment Size (MSS) set to a value of 0 (i.e., all bits cleared). The target computer 412 responds to the first TCP SYN packet with a first SYN ACK packet represented by a line 422. As discussed above, at least a portion of the information included in the first SYN ACK packet received from the target computer 412 is determined by data in the TCP/IP stack within the target computer 412, and the data is determined, at least in part, by the particular operating system running on the target computer 412.

[0089] The network security system 410 next sends a second RFC compliant TCP SYN packet to a target computer 412 via a second packet transmission represented by a line 430. The first TCP SYN packet has the TCP Options Maximum Segment Size (MSS) set to a value of 128 (i.e., bit 7 set). The target computer 412 responds to the

second TCP SYN packet with a second SYN ACK packet represented by a line 432. As discussed above, at least a portion of the information included in the second SYN ACK packet received from the target computer 412 is also determined by data in the TCP/IP stack within the target computer 412, and the data is determined, at least in part, by the particular operating system running on the target computer 412.

[0090] Preferably, the network security system 410 next sends a third RFC compliant TCP SYN packet to a target computer 412 via a third packet transmission represented by a line 440. The first TCP SYN packet has the TCP Options Maximum Segment Size (MSS) set to a value of 384 (i.e., bits 7 and 8 set). The target computer 412 responds to the third TCP SYN packet with a third SYN ACK packet represented by a line 442. As discussed above, at least a portion of the information included in the third SYN ACK packet received from the target computer 412 is also determined by data in the TCP/IP stack within the target computer 412, and the data is determined, at least in part, by the particular operating system running on the target computer 412.

[0091] Together, the information in the three SYN ACK packets received by the network security system 410 from the target computer 412 in response to the TCP SYN packets comprise a fingerprint that is compared with the fingerprint database 352 described above in connection with Figure 3.

[0092] Figure 5 illustrates one example embodiment of the TCP SYN packet 500 used in the operating system identification method of Figure 3. On a higher (IP) level, the packet typically provides source and destination Internet Protocol addresses and unique network addresses (not shown). On the TCP level, the packet includes the source port 510 from which the packet was sent, and the destination port 512 on the target computer to which the packet is destined. The 32 bit sequence number 514 describes the starting point of the data contained in the packet in the data window, and the 32 bit acknowledgement number 516 contains the last byte received by the target computer. Data offset 518 and a reserved section 520 are also part of the packet.

[0093] The TCP flags 522 denote important information about the packet. In particular, SYN denotes the first packet in a new stream, and space in the sequences 514 and 516 is typically reserved for SYN flags besides the single bit in the TCP flags region 522. The window 524 describes how much data may be transferred into the buffer at one

of the end point computers in the TCP packet communication. The checksum 526 and urgent pointer 528 are included. The TCP options 530 typically include a maximum segment size. After packet padding 532, the actual data 534 carried by the packet is attached.

III. Multi-Tier Port Scanning for Target Computer Identification and Target Port Identification

[0094] Port scanning is an essential tool for ensuring network security. Typically, a would-be intruder will apply a port scanner to a chosen target computer to attempt to find open ports. Through these open ports, the intruder may hope to obtain access to the target computer through known or unknown vulnerabilities. In the network security context, applying an ordinary port scanner to all 2^{16} (65,536) ports on each target computer on a target network may significantly drain network resources, take an impracticable amount of time, and not provide an accurate accounting of actual vulnerabilities of target computers.

[0095] In one embodiment, the present system employs an iterated port scanning system with at least two phases of operation: host discovery and service discovery. For a particular set of IP address ranges in the target network (the "scan list"), host discovery determines which IP addresses represent live target computers (i.e., computers that respond to network packets) and adds each such address to the "live list;" determines which IP addresses represent computers that are partially responsive, as discussed below, and adds each such address to the "potentially live list;" and determines which IP addresses represent computers that are non-responsive, and adds each such address to the "dead list." In the service scan, each target computer reported as live in the host discovery phase is subjected to scanning of a known set of ports likely to be open to traffic.

A. Host Discovery

[0096] As described in more detail below, the host discovery phase applies one, two or three distinct tests to each IP address on the scan list. Preferably, the scan list is scanned in batches, where each batch of IP addresses is scanned in parallel (as described in more detail below) to identify host computers (i.e., live target computers).

i. First Test (ICMP Ping Request)

[0097] In a first host discovery test, a standard ICMP ping request is sent to each target computer. If a response is received, the target computer is removed from the scan list and placed on the live list. In one embodiment, this entails sending out an ICMP echo request packet to each host. Multiple ICMP packets can advantageously be sent out in parallel to more than one IP address in a batch. Typically, the system waits until an ICMP echo reply is received from all IP addresses in the batch or the ICMP echo request is timed out. As a result of this process, for each batch of IP addresses on the scan list, a list of IP addresses that responded to the ICMP echo request is removed from the scan list and placed on the live list.

ii. Second Test (Sending TCP Packets)

[0098] If no response is received from one or more IP addresses on the list in the first test, a set of TCP packets (either single SYN packets or full TCP connection sequences ("TCP full connect")) are sent to the remaining target computers in the scan list as a second host discovery test. Specifically, a list of "TCP discovery ports" is selected in one embodiment. The selection is based on the TCP ports that are most likely to be open. The TCP discovery port list is advantageously relatively short, and preferably includes well known service ports such as HTTP (hypertext transfer protocol), SMTP (simple mail transfer protocol) and the like. One non-exclusive example embodiment of a TCP host discovery list is shown in Table 3.

Service	Port
FTP	21
SSH	22
Telnet	23
SMTP	25
HTTP	80, 81, 8000
POP3	110
NetBIOS	139
SSL / HTTPS	443
PPTP	1723

Table 3: Sample TCP Discovery List

[0099] Other common ports can be added or removed from the list, and the list, for example, can be tailored to specific network environments where certain ports are more

likely to be in use than others. In this example, File Transfer Protocol (FTP), Secure Shell (SSH), Telnet, Simple Mail Transfer Protocol (SMTP), HyperText Transfer Protocol (HTTP), Post Office Protocol (POP3), NetBios, Secure Sockets Layer (SSL), and Point-to-Point Tunneling Protocol (PPTP) are advantageously used.

[0100] In one embodiment, a standard TCP SYN packet is sent to some or all of the ports on the TCP host discovery list for each target IP address (target computer.) As with the prior ICMP ping test, multiple IP addresses are advantageously tested in parallel (i.e., in batches) in a preferred embodiment. If a target computer responds with a TCP SYN ACK, then the target computer is added to the live list. Otherwise, the TCP SYN request to the target times out (i.e., a maximum time period passes without a response from the target computer).

[0101] In an alternative embodiment of the TCP scan test, a standard TCP full connect request initiated using the standard Window® Winsock interface. If the operating system confirms that a TCP three-way handshake has been completed, then the target computer is added to the live list. If the target responds with a TCP RST ACK, an ambiguous response, the target computer is added to the "potentially live" list. Otherwise, the TCP request to the target times out.

[0102] The foregoing tests result in a list of live target computers (IP addresses) on the live list. The target computers on the live list are removed from the scan list. If there are any IP addresses that have not been confirmed on the "live list" or "potentially live list," then a third step of scanning selected UDP ports on the target computer is performed for IP addresses remaining on the scan list.

iii. Third Test (Intelligent UDP Port Scanning)

[0103] If any IP addresses (i.e., target computers) remain on the scan list after the first two tests, then a third test of intelligent UDP port scanning is performed on the remaining addresses. As described below, intelligent UDP port scanning differs from traditional scanning of UDP ports, which is notoriously inaccurate. When scanning TCP ports, a response from the target computer signals that the target port on the target computer is open. In contrast, while scanning UDP ports, no response from the target computer signals that the target port is open, and a response (an ICMP error message) will only be sent if the UDP port scanned is closed. The traditional method of scanning UDP ports

thus results in a significant number of "false positives" where a UDP scan results in no response (suggesting an open port), but the UDP port scanned is actually closed. This may happen, for example, when a firewall or router blocks the ICMP error message from returning from the target computer or when the ICMP error message is simply lost while returning from the target computer. Over thousands of tests, such errors become likely.

[0104] Sometimes, in order to "force" a response from the target computer, an intruder may send a malformed packet to a target port. While this known technique increases the likelihood that an open UDP port on the target computer can be identified, this technique also substantially increases the likelihood that the malformed packet could damage the target computer. Also, firewalls or routers may detect and filter out malformed packets, and such packets can alert the target network of an attempted security breach.

[0105] The intelligent UDP port scanning test in accordance with this embodiment of the present invention employs an efficient, less intrusive and more accurate method for scanning UDP ports on a target computer. As with TCP scanning, a UDP host discovery list of commonly used UDP ports is created.

[0106] An example of an UDP discovery list is shown in Table 4.

Service	Port
DNS	53
DHCP	67
BootP	69
NTP	123
NetBIOS File/Printer Sharing RPC Pipes WINS Proxy	137
SNMP	161
IKE	500

Table 4: Sample UDP Host Discovery List

[0107] Unlike the data in traditional UDP port detection packets, the data contained within the UDP packets sent in accordance with the present invention are specifically designed to prompt a reply from the scanned host (i.e., target computer) based on knowledge of a service typically associated with the UDP port. If no information is

available about the UDP port, standard data (for example, the data representing a simple ASCII character return or control character) are placed in a UDP packet. In one embodiment, an exemplary UDP data probe packet is designed to solicit a response from a NetBIOS name service that typically runs on UDP protocol at port 137. An exemplary UDP data probe for UDP port 137 is shown in Table 5. In this case, the probe is advantageously a NetBIOS node status request, which typically results in a known response from the UDP port.

Service	Port	Data probe (hex)
NetBIOS	137	A2 48 00 00 00 01 00 00 00 00 00 00 20 43 4B 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 00 00 21 00 01 -- -- -- --

Table 5: Sample UDP Data Probe

[0108] Similar UDP data probe packets can be created for other UDP ports known to be associated with certain services, based on publicly known information in Internet Requests For Comment (RFCs), published specifications on private standards, analysis and monitoring of traffic on known ports, or well-established reverse engineering techniques. Particular protocols and particular UDP ports are subject to substantial change over time, due to changes in standards, technology, operating system, and the like. Preferably, when a data probe from a known protocol is not available for a certain UDP port, standard UDP packets with data representing at least one simple ASCII carriage return are sent.

[0109] As with TCP port scanning, multiple UDP ports can be advantageously scanned in parallel. Typically, the system sends a UDP data probe packet to a set of UDP ports on each target computer. The system waits until a UDP data reply is received from one or more target computers or until the packets sent have "timed out." If an ICMP unreachable reply is received, the hosts added to the "potentially live" list. Only those target computer IP addresses not yet determined to correspond to live or potentially live target computers after employing ICMP, TCP and UDP scanning are left on the scan list.

[0110] The three-step discovery phase, employing ICMP, TCP and UDP scanning steps, optionally is applied multiple times to increase the accuracy of identification of live target computers on the target network. In one embodiment, after an initial pass through all three steps, if there are any remaining target computer IP addresses on the scan list that

have not been identified on the "live list" or on the "potentially live list," at least the ICMP and TCP steps are repeated a predetermined number of times for those remaining scan list target computers. Optionally, the UDP step may also be repeated. In particularly preferred embodiments, more intensive UDP scanning techniques, employing more ports, different data probe packets or more data packets, can be applied to the target computers remaining on the scan list to provide a more definitive list of live target computers on the network. Alternatively, traditional UDP scanning with malformed packets can be attempted.

[0111] One obstacle to the usefulness of UDP scanning is that some target computers will limit the number of ICMP responses sent within a predetermined "response latency" time period, ranging from about 250 milliseconds to about one second. Thus, if a target computer is sent twenty UDP requests to various ports in one response latency period, it may only send one or two ICMP error responses indicating that the ports are closed, even if all twenty ports tested are closed. This results in substantial ambiguity as to UDP port status. In particular, when applying traditional UDP scanning techniques, many "false positives" (i.e., falsely reported open ports) are possible.

[0112] The present invention advantageously overcomes the foregoing problem by (1) determining dynamically the latency period of the target computer, and (2) continuing the UDP port scanning of each target port for at least one entire latency period (or until an ICMP error response or UDP data response is received) to ensure that a non-responsive UDP port is actually open and is not simply masked by the response latency. The latency period is determined by selecting a UDP port, preferably one that is known to be closed, and sending UDP datagram requests for a predetermined maximum response latency period (i.e., for a time at least as great as the target computer dead time period (approximately two seconds in particular embodiments)). The time between responsive ICMP error messages or the time between UDP response packets represents the standard latency period. This test can be repeated to confirm the latency period (dead time).

[0113] Once the latency period is determined, response validity is ensured by sending UDP requests continuously to the target port for at least the pre-determined latency time or until an ICMP error response or UDP data response is received. If an ICMP error response is received, the port associated with the prompting UDP request may be assumed

to be closed. If a UDP data response is received, the associated port may be assumed to be open. If no response is received for the entire latency period, the associated port may be assumed to be open unless a router, firewall or packet loss has interfered with UDP port response.

[0114] The foregoing three-step discovery phase and optional intensive UDP scanning step are illustrated by the process flowchart in Figure 6. As discussed above and as illustrated in Figure 6, the discovery phase begins with a scan list 610, which is advantageously parsed into batches of IP addresses 612. In a step 620, a batch of IP addresses is obtained for processing. In a step 630, the ICMP ping test is performed as discussed above. Depending on the outcome of the test, each IP address is added to a live list 632 or remains in the present batch of IP addresses 612. As illustrated, the process also operates with a potentially live list 634 and a dead list 636.

[0115] After performing the ICMP ping test, the process determines, in a decision step 638, whether any of the IP addresses in the current batch have not been added to the live list. If no IP addresses remain, the process proceeds to a decision step 640 where the process determines whether all batches of IP addresses have been scanned. If batches of IP addresses remain, the process returns to the step 620 and obtains a new batch of IP addresses. Otherwise, the process ends.

[0116] If, in the decision step 638, the process determines that one or more IP addresses have not been added to the live list, the process proceeds to a step 650 where the TCP discovery scan described above is performed using a TCP discovery port list 652. IP addresses are added to the live list 632 or to the potentially live list 634 in accordance with the results of the scan. Then, in a decision step 654, the process determines whether any of the IP addresses in the current batch has not been added to the live list or the potentially live list. If no IP addresses remain, the process proceeds to the decision step 640 discussed above.

[0117] If, in the decision step 654, the process determines that one or more IP addresses have not been added to the live list or the potentially live list, the process proceeds to a step 660 where the intelligent UDP discovery scan described above is performed using a UDP discovery port list 662. IP addresses are added to the live list 632, the potentially live list 634 or the dead list 636 in accordance with the results of the

scan. Then, in a decision step 664, the process determines whether any of the IP addresses of the current batch have not been added to the live list, the potentially live list or the dead list. If no IP addresses remain, the process proceeds to the decision step 640 discussed above.

[0118] If, in the decision step 662, the process determines that one or more IP addresses have not been added to the live list, the potentially live list or the dead list, the process proceeds to a step 670 where the intensive UDP discovery scan described above is performed using a second UDP port list 672, which advantageously includes additional ports to scan. In one embodiment, the second UDP port list 672 is the UDP service port list advantageously described below, but any combination of UDP ports may be used. IP addresses are again added to the live list 632, the potentially live list 634 and the dead list 636 in accordance with the results of the intensive scan. Then, in a decision step 674, the process determines whether the discovery testing needs to be repeated. If all IP addresses have been added to one of the three lists and no IP addresses remain, the process proceeds to the decision step 640 discussed above. If any remaining IP addresses remain that have not been added to one of the lists, then the process determines whether the scanning steps have been performed a predetermined number of times for the current batch of IP addresses. If the steps have been performed a predetermined number of times, the process proceeds to the decision step 640. Otherwise, the process returns to the ICMP ping test 630 to again process the remaining IP addresses.

[0119] As discussed above, the intensive UDP discovery scan 670 is optional. If the intensive UDP discovery scan 670 is not included, then the decision step 664 following the intelligent UDP discover scan 660 advantageously includes the additional decision of whether to repeat the scanning process based on whether the scanning process has been performed a predetermined number of times.

B. Service Discovery

[0120] In one embodiment, the present invention then proceeds to examine each host (i.e., target computer) in more detail using the live list and optionally using the potentially live list as well. In service discovery, a set of common TCP service ports and a set of common UDP service ports are scanned. The TCP service discovery list and the UDP service discovery list are typically substantially larger than the TCP host discovery list and

the UDP host discovery list. Each list typically includes some subset of ports that are commonly used for communications. For example, each list may include anywhere from a few to hundreds of ports. Typically, each list includes ports such as those shown in Table 6 (an exemplary list of default ports publicly reported by Microsoft® for Windows® 2000.) This list is not exhaustive, and changes in technology, protocols, network infrastructure and operating systems frequently change port requirements.

Service	UDP Ports	TCP Ports
NetBIOS	138, 137	
Client/Server		135
CIFS	445	139, 445
DCOM	135	135
DHCP		67, 68, 135
DNS	53	53, 139
Exchange 5.0		135
IMAP		143, 993
LDAP		389, 636
POP3		110, 995
RPC		135, 1500, 2500
SMTP		25
NNTP		119, 563
File Sharing	137	139
FTP		20, 21
HTTP		80, 443, 8000, 8080
IIS		80
IKE	500	
IRC		531
ISPMOD		1234
Kerberos	88, 464	88, 464, 543, 544, 2053
WinNT Login	137, 138	139
Macintosh File Services		548
MSN Chat/Messaging	1801, 3527	135, 569, 1801, 2101, 2103, 2105, 6665, 6667
NetMeeting		389, 522, 1503, 1720, 1731,
PPTP		1723
Printer Sharing	137	139
SNTP	162	
SQL/Pipes/RPC	137	139, 1024-5000, 1433
Telnet / Terminal		23, 3389
UNIX Printing		515
WINS	137	42,102, 135, 137

Table 6: Example Set of UDP and TCP ports for Service Discovery

[0121] Because there are 2^{16} (65,536) possible ports on each target computer, the selection of a subset of ports is preferred because time constraints generally preclude scanning of all ports on each target computer, especially on a large network. Similarly, random selection of ports on each target computer is unlikely to be fruitful because an average target computer will run less than a dozen, and in rare cases dozens or hundreds, of services, making the probability of hitting an open port through random selection of a port inefficient and inaccurate.

[0122] In accordance with preferred embodiments of the present invention, TCP service discovery uses the large list of TCP service scan ports, a few of which are shown above, and attempts to connect to each target port on each target computer. As with Host Discovery, described above, standard TCP SYN scanning requires waiting for a SYN ACK response from each target port, or TCP "full connect" scanning requires waiting for an operating system message indicating that a three-way handshake between the target computer and scanning system has been completed. Target computers that respond positively to either of these attacks are added to a list of vulnerable computers to undergo vulnerability assessment for each open target port found.

[0123] In accordance with preferred embodiments of the present invention, UDP service discovery uses the large list of UDP service scan ports, a few of which are shown above, and attempts to connect to each target port on each target computer. As with Host Discovery, described above, the present invention may advantageously use the improved UDP data probe packets for any port recognized to be commonly associated with a particular service. Alternatively, packets that include data representing standard characters, such as one or more ASCII character returns, may be used. Target computers that respond positively to this attack are added to a list of vulnerable computers to undergo vulnerability assessment for each open target port found. The vulnerabilities used are typically limited to those associated with the operating system of the target computer as discovered by the operating system identification system described previously, and by those vulnerabilities associated with the open ports found on the target computer. If the operating system of the target computer cannot be conclusively identified, then typically all vulnerabilities associated with the open ports found on the

target computer during the service discovery system described herein are tested against the target computer.

[0124] As with Host Discovery, a more intensive UDP scanning regime can be applied to target computers that do not respond to a simple UDP data probe packet. The more intensive UDP scanning regime advantageously uses, for example, traditional UDP port scanning in combination with the optional latency time resolution described above.

[0125] After vulnerability assessment, an optional active assessment of target computers takes place. Active assessment applies knowledge of target computer operating system, open ports, as well as information recovered from the target network during vulnerability assessment, to further test known vulnerabilities against each target computer.

Parallel Processing of Multiple Ports

[0126] In one preferred embodiment, the present invention advantageously performs port scanning in "batches," rather than completing serial scanning of individual ports, one after another. This allows small pieces of a large scan to be completed and saved to a database, not requiring the entire results of a large scan to be held in memory all at once. For example, if a class A network is tested, hundreds of millions of ports need to be scanned merely for host discovery. The system may be advantageously adapted to technical limitations in older network equipment to scan ports of computers in a very large target network with multiple, significant IP address ranges. In the event of network or system failure, the system resumes particularly large scans following the last known successfully completed batch.

[0127] During a port scanning process, preferred embodiments of the present invention identify small portions or "batches" of the entire scan range of IP addresses, comprising, for example, 64 target computers represented in the scan range. Host discovery begins using a first batch of target computers. When all live hosts in the first batch have been discovered, service discovery of TCP and UDP services is performed on the live hosts in the first batch, along with other testing services such as operating system identification, tracerouting of network topology, and active assessment of vulnerabilities, described herein. When a batch is completed, the results are stored in a vulnerability

database. Then, the next batch of 64 target computers are similarly processed, and so on, until the entire scan list has been processed for the target network.

[0128] Preferably, a large number of target computers are tested on one or a small number of ports in parallel, rather than a large number of ports being tested on a single or small number of target computer in parallel. Typically the former method prevents undue load on any single target computer, prevents network overload, and reduces the risk of triggering target network security alerts. Occasionally, the later method may be preferred if, for example, a few target computers are selectively singled out for scanning.

[0129] More particularly, in one exemplary embodiment, the present invention simultaneously uses a set of 640 packet slots. A pocket slot is an advantageously allocated space in memory that keeps track of a sent packet while waiting for a response or for the packet sent to time-out. Thus, when processing a batch of target computers, up to 640 ICMP request packets can be handled simultaneously, but it will be appreciated that a different number of slots, more or less than 640, may be allocated and used simultaneously. When an ICMP response is received for one of the packets sent, or after the packet has timed-out, the slot allocated for the packet is then reallocated to send a new packet and monitor any response or time-out. Thus, packets can be kept moving at a rapid rate since there is a small average delay between sending of one packet and receipt of a response and/or time out for another packet. It is more than likely, for example, that by the time the 640th ICMP request packet has been sent, a response to one of the sent packets will have been received, thereby freeing the associated slot for handling of another packet. A similar methodology applies to the handling of TCP SYN packets, UDP host discovery, and service discovery in general. As those of ordinary skill in the art will appreciate, operating system identification and tracerouting can use similar batched parallel port monitoring. For those processes, the number of slots used is not limited to 640 slots, and can be varied for programming or scanning efficiency.

[0130] As an example, one embodiment of the present invention uses a batch size of one hundred target computers for simultaneous scanning. For host discovery, the ICMP scanning process sends 100 ICMP echo requests and monitors responses and time-outs. Because this is less than this embodiment's 640 packet slots, all ICMP request packets can be sent out, and responses or time-outs for all packets can be monitored at the same

time. Assuming a TCP host discovery list of 10 TCP ports, the TCP host discovery scanner will ultimately send out 1000 packets (100 target computers × 10 ports = 1000 packets) to probe all listed ports of all listed computers. In this embodiment, the scanner will initially send out 640 packets in a sequential burst using all 640 packet slots. However, as discussed above, by the time the 640th packet is sent, it is probable (depending on time-out settings and how many responses the system receives) that at least one of the 640 packets sent earlier will have generated a response or a time-out, freeing that packet slot for sending one of the remaining 360 packets. By constantly monitoring the receipt of response packets and time-outs, the entire list of target computers can be tested with little wasted time.

[0131] Larger batch sizes and more packet slots can be used, but the efficiency of such use depends on how much of the target network comprises live hosts, time-out values, network response time, memory and other system resources, and the like. In general, very large batch sizes take correspondingly longer in the host discovery phase, such that by the time the system begins scanning the discovered hosts in the service scanning phase there is a greater possibility that those target computers may have changed in some way.

[0132] The above-described service discovery phase is illustrated by the process flowchart in Figure 7. The process starts with a live list 710, which advantageously corresponds to the live list 632 generated by the host discovery process illustrated in Figure 6. The process may also include the potentially live list 634 generated in Figure 6; however, for ease of explanation, the potentially live list is not shown in Figure 7. The IP addresses in the live list 710 are advantageously parsed into batches of IP addresses 712. As described below, the process of Figure 7 operates on a target computer vulnerability database 714 and a known vulnerability database 716.

[0133] In a first decision step 720 in Figure 7, the process determines whether TCP SYN scanning or TCP "full connect" scanning is being performed. As discussed above, the TCP full connect scanning process requires the process to wait for an operating system message indicating that a three-way handshake between the target computer and the scanning system has been established. In the TCP SYN scanning process, the process, waits for a SYN ACK response from each target port, in which case, the target computers

can be processed in parallel, as discussed above. In both cases, the process proceeds to a step 722 to obtain a first batch of IP addresses representing a first batch of live (or potentially live) target computers. If TCP full connect scanning is being performed, the process operates on a smaller number of ports on the target computers at one time. The process then proceeds to a decision step 730.

[0134] In the decision step 730, the process determines whether all the live target computers have been processed in TCP full connect scanning or whether all the batches of live target computers have been processed in TCP SYN scanning. If all the target computers or all the batches of target computers have been processed, the process ends. Otherwise, the process proceeds to a TCP service scan routine 740 wherein the process uses a TCP service discovery list 742 to identify the TCP service ports to be examined for each target computer. As described above, TCP packets are sent to the identified TCP service ports of each target computer, and the target computer vulnerability database 714 is updated for each target computer in accordance with whether a response is received or is not received from each target computer for each TCP service port scanned and using the known vulnerability database to obtain the vulnerability information for the particular TCP service ports that are determined to be open.

[0135] After performing the TCP service scan routine 740, the process proceeds to an optional UDP latency test 750 wherein the latency of each target computer is determined and stored in a latency database 752. The process proceeds from the latency test 750 or directly from the TCP service scan routine 740 to a UDP service scan 760 wherein the process uses a UDP service discovery list 762 to identify the UDP service ports to be examined for each target computer. As described above, UDP packets are sent to the identified UDP service ports of each target computer, and the target computer vulnerability database 714 is updated for each target computer in accordance with whether a response is received or is not received from each target computer for each UDP service port scanned and using the known vulnerability database to obtain the vulnerability information for the particular UDP service ports that are determined to be open.

[0136] After completing the UDP service scan routine 760, the process proceeds to a decision step 770 wherein the process determines whether a response has been received from all the scanned service ports. If not, the process determines whether the scanning of

the current target computer or batch of target computers has been performed a predetermined number of times. If all the scanned service ports have responded or if the scanning has been performed a predetermined number of times, the process returns to the step 720. Otherwise, the process returns to the TCP service scan routine 740 to scan the service ports that have not responded.

C. Banner Grabbing

[0137] "Banner grabbing" is a method of stripping relevant information from packets received from a target computer. In one embodiment, for each target computer in the live list, an attempt is made to perform banner grabbing against each open TCP port and each open UDP port discovered during the service scan phase. If information is successfully obtained from a responsive TCP or UDP packet, this information is stored in a database in association with the target computer from which the information is obtained. The stored information is subsequently used to collect vulnerability information on the target computer, and the vulnerability information is also stored in a database.

[0138] For each open TCP port located during the service discovery phase, a TCP data probe is sent to that port if the port is known to be typically associated with a particular service. Thus, for example, when sending a TCP data probe to a target TCP port 80 on a target computer, where TCP port 80 is well known as the common service port for HTTP, the system sends an HTTP-compliant GET request, and strips useful information from any TCP response packet sent back from the target computer. As noted in the above tables, similar commands can be used for other ports with well known services running thereon. The useful information obtained may advantageously be transformed or translated into a readable or processable form (e.g., text) and stored for later reporting.

[0139] For each UDP port discovered during the service discovery phase, a similar UDP data probe is sent to each UDP port on the target computer known to be typically associated with a service. Thus, for example, from the foregoing tables, UDP port 137 is known to be associated with a NetBIOS service. In one embodiment, the system sends a NetBIOS Node Status Command, for example, and strips and stores useful information from any response packet sent by the target computer. Again, the information obtained in

this manner may advantageously be transformed or translated into a readable or processable form (e.g., text) and stored for later reporting.

IV. Tracerouting

[0140] In one embodiment, for each target computer in the live list, an attempt is made to perform an ICMP traceroute between the system and the target computer. If the ICMP traceroute is incomplete, an attempt is made to perform a TCP traceroute between the system and the target computer. Based on the traceroute results for each target computer, a map of the target network topology is created and stored in a database.

[0141] Initially, traditional ICMP tracerouting is performed. A number of packets are sent to the target computer with varying TTL (time to live) values in the TCP/IP header, starting with a TTL of 1. If the ICMP echo request packet fails to reach the destination target computer, it will return an ICMP destination unreachable packet containing the IP address of the router/host that the packet was returned from. Packets that arrive at the target computer itself return an ICMP echo reply packet. Based on the known TTL value, the number of "hops" between the system and the target computer, and the router/host at each "hop" can be mapped out. When all packets have arrived or timed out, if there is a router/host and IP address associated with each "hop" between the system and the target computer, the traceroute step is completed for that target computer. The ICMP traceroute is continued for a number of pre-determined passes until a complete trace is received or the maximum number of passes is completed. If, after the pre-determined number of passes the tracerouting of the target computer is incomplete, then a TCP traceroute is attempted.

[0142] TCP tracerouting works similarly to ICMP tracerouting, except TCP SYN packets are used. In some instances, TCP packets are more reliable at completing otherwise incomplete traces, because, for example, ICMP packets are sometimes blocked by firewalls or routers. Typically, the TCP port chosen is taken from the list of open ports discovered for the target computer that is the subject of the traceroute. As with ICMP tracerouting, the TTL value in each SYN packet is incrementally increased, and the return of ICMP unreachable packets and SYN ACK packets is monitored for all "hops" between the scanning system and the target host. Through the combination of ICMP tracerouting

and TCP tracerouting, a complete map to each target computer, and collectively a relatively complete map of the target network topology, is advantageously created.

[0143] The network maps are an attempt to represent in the most concise way possible the topology of the network as discovered by tracerouting methods. Basically, packets are sent to each discovered host in the scanned network with the Time-To-Live field set to the hopcount at which a packet indicating failure to transmit due to that hopcount will be sent by the machine that is that many hops away from the scanning machine. By increasing the TTL and storing the IPs of the machines that respond until getting a response from the host in question, a string of IPs is built up that represents the machines that led up to that host. The procedure is far from perfect. A machine at a given hop may not respond within the time expected by the tracerouting algorithm, or it may not respond at all if it is a firewall that blocks the traceroute packets. Any approach that can exist must deal with this uncertainty in some way. In one embodiment, any unknowns are assumed to be firewalls.

[0144] The algorithm typically employed is presented with a set of discovered hosts and each host includes a IP address that represent what the tracerouting algorithm initially determines are the machines leading up to them. This string may include unknowns which are denoted by 0xFFFFFFFF or 255.255.255.255. The algorithm operates as follows:

- CONDENSE UNKNOWNS: Condense consecutive unknowns into a single hop with unknown IP address (0xFFFFFFFF). If a traceroute is performed to a given hop and an unknown is received, it is probable that every hop after that will also be unknown because the first unknown hop is a firewall blocking the packets.
- UNKNOWN RESOLUTION: Attempt to resolve unknowns by looking through the other routes to see if a route exists with the IP addresses on either side of the unknown in this route that are connected by all known machines. If found, replace the unknown with those machines. This attempts to eliminate spurious unknowns so they are not mislabeled as firewalls. It will not eliminate actual firewalls as they will typically be unknown.
- BUILD SET OF NODES: Build up a list of structures (structs) of type ROUTENODE. This is a list in which the IP addresses are guaranteed unique

(except for unknowns as detailed later) and represents the set of all IP addresses discovered in the network (i.e., the hosts discovered by the scanning process and the hops leading up to them discovered as by the tracerouting process.) The m_pHost member of each ROUTENODE struct will be NULL if the IP address is not one of the hosts discovered in the scanning process and will point to the host data discovered by that process otherwise. As mentioned, the known machines are unique by IP address. The unknowns are further qualified by the IP address immediately preceding them. As a result, unknowns with the same IP address preceding them are typically considered to be the same firewall and are therefore represented by a single ROUTENODE in the list.

- FILL IN CONNECTIONS: Each ROUTENODE struct has a list of ROUTENODE pointers in it which represent the machines that ROUTENODE is directly connected to. The algorithm next fills this in to guarantee that the IP addresses are unique in each list of connections. Note that these pointers point to other ROUTENODE structs in the list (i.e., the list is typically a self contained bunch of data.)
- COMPUTE DISTANCE FROM INTERNET: The traceroute information is once again traversed for each host and for each IP address. The corresponding ROUTENODE struct is looked up in the list and the struct's m_nDistFromInternet is set according to the hopcount in the route. This is typically done to represent a maximum hopcount. In other words, the m_nDistFromInternet fields are initialized to 0, and then, if the hopcount in a traceroute IP string is greater, the m_nDistFromInternet is set to that hopcount. For example, an IP address (let it be called "d," for example) may appear in one route as a-b-d and in some other route as a-b-c-d, where "a," "b," and "c" are other IP addresses on the route to the IP address c. When this part of the algorithm completes, d will have m_nDistFromInternet = 4. This information is used by the algorithm to know if one machine proceeds another. Note that this maximum of discovered hop lengths is a heuristic for the sake of making the problem reasonably computable. It could go awry if a particular machine should be drawn as connected to another machine (because it is directly connected to it) and precedes it in one route, but it is not

drawn that way because the machine is in some other route that bumps its m_nDistFromInternet to the same or more than that of the machine it is connected to in the other route. This situation is rare, and this heuristic is typically acceptable.

- BUILD ROUTER LIST: The algorithm then traverses the ROUTENODE list in order to build up a list of ROUTER structs. A ROUTER struct contains a pointer to a ROUTENODE believed to be a ROUTER and a list of ROUTENODE pointers that are leaf (end of route) machines it is connected to. A leaf is any ROUTENODE that is directly connected to only one machine (besides itself.) For the purposes of this algorithm, a ROUTER is that machine. This stage of the algorithm builds up a list of ROUTER structs, and, within each ROUTER, builds a list of leaf ROUTENODEs connected to that ROUTER. The algorithm also marks a Boolean field within the ROUTENODEs that are being designated ROUTERs as it will become necessary to know which machines in the ROUTENODE list are routers without wanting to consult the ROUTER list.

[0145] Next the refinements to the algorithm are discussed, but first it is useful to discuss how the map would be drawn if the algorithm finished here. At this point, the algorithm has a set of ROUTERs, and within each of those, has the set of leaves connected to that ROUTER. Prior to the refinements about to be detailed, the map renderer would draw each ROUTER as a sphere and all of its leaves as small spheres embedded in a ring in orbit around the ROUTER sphere. The algorithm would then look for other ROUTERs whose m_nDistFromInternet fields are less than this ROUTER's m_nDistFromInternet (i.e., look for a ROUTER that is directly connected to this ROUTER and precedes it.) The algorithm takes the greatest m_nDistFromInternet router that it finds and connects this ROUTER to it visually. If it finds no preceding ROUTERs, it connects this ROUTER to the internet cloud in the center of the map. The refined algorithm still does all that is described above, but typically not before doing a bit more processing.

- ROUTER PROMOTION: There are circumstances when the algorithm as it has been detailed thus far would produce a map in which a ROUTER has leaf nodes surrounding it all of which have the same subnet, for example they might all be

149.17.1.xxx. The first 1, 2, or 3 digits (8 bits each) will be the same. That ROUTER has a ROUTER of less m_nDistFromInternet directly connected to it that has exactly 1 leaf node and that leaf node has the same IP address digits in common as all the ones in the first ROUTER's leaf list. In this case, although the traceroute data suggests that the first ROUTER and that single leaf are different machines, they are likely two NICs (Network Interface Cards) on the same machine. Hence, the algorithm "promotes" that ROUTENODE by adding its IP address to the m_lnOtherNics member of the first ROUTER's ROUTENODE and then removing it from the other ROUTER's list of leaves and marking that ROUTER's ROUTENODE to be no longer a ROUTER. That ROUTER is also removed from the ROUTER list. On the map then, the ROUTER's primary IP address is represented as usual but now there is a grey bar and the other IP address underneath it. The algorithm supports doing multi-homed ROUTERs and will represent all the IPs that get promoted. The multi-homed case is easily imagined by extending the dual-homed example discussed here.

- OTHER TYPE OF ROUTER PROMOTION: The preferred algorithm performs another type of router promotion as well. If exactly one leaf around some ROUTER fulfills a heuristic, it will be promoted to another NIC on the ROUTER. In order to test for the heuristic, the machine needs to be a host discovered during the scanning process because that is where the information comes from. It is noteworthy that most leaves will be discovered hosts. Although the algorithm does not demand this, and traceroute information may produce non-discovered-host hops that only have one thing connected to them and thereby make leaves out of them, since tracerouting is being done to the discovered hosts, the leaves will tend to be discovered hosts. The heuristic is, in one embodiment, as follows: if TCP 2001, 4001, 6001, or 9001 is discovered, or if UDP 520, or both UDP 23 and 79 are discovered, or if the discovered operating system is Cisco, that ROUTENODE is assumed to be a router-IP address and it is promoted (IP address added to the ROUTER's m_lnOtherNics and ROUTENODE pointer removed from its leaves).

- FIREWALL PROMOTION: Firewall promotion is similar to the router promotion heuristic detailed above. In one embodiment, if exactly one leaf around a firewall satisfies the heuristic then it gets promoted into the firewall's other-NIC list resulting in the map printing "Firewall," but now the known IP address is underneath it separated by the gray bar. The heuristic, in one embodiment, is if TCP 256, 257, 264, 265 is found on a machine then it is assumed to be a firewall.
- NUMBER FIREWALLS: This is the last step of the algorithm. The firewalls are sorted in order of the IP address that precedes them and are then numbered so that the map can print "Firewall-1, Firewall-2, ..., etc." on the report.

[0146] In this manner, an entire network map can be relatively accurately stored internally, and translated into a visual representation for reporting.

V. Vulnerability Identification and Active Assessment

[0147] For each known TCP port, UDP port, and operating system, the known vulnerabilities for that configuration are stored in a vulnerability database based on vulnerability identification codes. Advantageously, for many vulnerabilities in the vulnerability identification database, a methodology for testing the vulnerability can be written into an automatic script, which will assess the actual weakness of the target system to the suspected vulnerability. In one embodiment, these scripts are prepared in a assessment security scripting language, and are preferably prepared in FASL. FASL is a scripting language based on C++ and Java implementation, in one embodiment. FASL provides an adjustable, automated language for security testing various vulnerabilities. Multiple FASL scripts advantageously may be run in parallel. For example, in one embodiment up to eight simultaneous scripts may be run. Each FASL script, for example, will respond with a success or failure result noting whether the target computer was vulnerable or not to the given vulnerability identification code. The information collected by the FASL script from the target computer, and the success or failure of the attempt, are stored in a database related to the target computer for later reporting, for use in additional vulnerability testing, or for repeating additional testing.

VI. FASL Scripting Language

[0148] The implementation of the FASL language is similar in construction to C++. In one embodiment, FASL includes member functions in structure objects, constructors and destructors in structure objects, inheritance in structure objects, arrays of scalar types, new scalar type "binary" and associated functions, string constants that support embedded hex codes including embedded zero bytes, RPCCheck() and SMBCheck() functions for RPC and Netbios checks, debugMessage() on all scalar types that produces hex output on binary type, recursion, function overloading, reference parameters, and support for Active-Assessment.

[0149] In one particular implementation of FASL, all variables are stack variables. In addition to permitting recursion, this also allows a parse tree to be built once and then used to execute the same script by multiple threads. The execution architecture in FASL treats the parse tree as a read-only data structure. Hence, other than considerations such as two instances of the same script attempting to instantiate a Socket on the same target computer IP and port (which should not happen in practice), FASL is completely thread-safe.

A. Scalar data types

[0150] Scalar data types are those native types in the language that take up one slot on the execution stack. Basically, any variable that can be declared is a scalar type, an array of a scalar type, or an instance of a structure. The scalar types are as follows:

Scalar Type	Definition
void	Function return type
int	64 bit signed integer
string	String of printable characters of arbitrary length, terminated by a null
Binary	String of arbitrary bytes of arbitrary length, terminated by a null. Type keeps track of its length
Char	8 bit signed integer (interchangeable with int)
Boolean	True or false (not interchangeable with int)

Table 7: FASL Data Types

[0151] Typically, a string is NULL terminated. This means that internally the string is represented as a string of characters with a zero marking the end. This zero's existence is always implicit, however. In other words, there is no string function that will return you the zero. The length of the string is computed by counting the number of characters up to

but not including the zero. This also makes it so that if a constant is assigned to a string of, for example, "this is a string\x0 with stuff beyond the zero." the value of that string will be "this is a string" and its length will be 16. Type binary typically does not use a zero or any other delimiting character in its internal representation. The string constant example with the embedded zero would have everything after the zero also in the binary.

[0152] For example,

```
binary b = "1234";          // length = 4.
string s = "1234";          // length = 4.
binary b = "zzzz\x0ssss";   // value = "zzzz\x0ssss", length = 9.
string s = "zzzz\x0ssss";   // value = "zzzz", length = 4.
```

[0153] Types which are equivalent to int (interchangeable in expressions with int) are as follows:

Type	Description
Attack	specifying type of attack
Operatingsystem	specifying target OS of a script
Protocol	specifying protocol used by a Socket
Returnvalue	specifying status of script (script return value)
Ipservice	IP type of service
Ipoptions	IP options
Ipoffset	IP offset

Table 8: Additional int-like FASL types

[0154] Keywords that indicate a constant value of a given type are as follows:

Constant Keyword	Value
null	0 (int, char, boolean)
true	1 (boolean)
false	0 (boolean)

Table 9: Additional FASL Constants

[0155] The types above that are equivalent to int also have keywords indicating constant values of those types. They are (respective to above) as follows, for example:

Type	Constants
attack	ATTACK_UNKNOWN, ATTACK_INTRUSIVE, ATTACK_DOS, ATTACK_NONINTRUSIVE
operatingsystem	OS_UNIX, OS_MAC, OS_WINDOWS, OS_UNKNOWN, OS_ROUTER
protocol	TCP, UDP, ICMP, IGMP

returnvalue	RETURN_SUCCESS, RETURN_FAILURE, RETURN_TIMEOUT
ipservice	LOWDELAY, THROUGHPUT, RELIABILITY, MINCOST, CONEXP, ECTCAP
ipoptions	EOL, NOP, RR, TS, SECURITY, LSRR, SATID, SSRR, RA
ipoffset	RESERVED, DONT, MORE

Table 10: FASL Type Constants

B. Statements

[0156] A FASL script is typically a list of statements. These statements are generally separated by semicolons. The exception to this rule is that statements that involve blocks (other statement lists enclosed by { and }) do not typically require semicolons to separate them. A semicolon constitutes a valid (empty) statement so it does not hurt anything to put semicolons after statements involving blocks but it accomplishes nothing. A new block represents a new scope. Any variables declared in that scope will be accessible only in that scope, and if they have destructors, those destructors will be called at the end of the scope. Variables can be declared with the same name as variables in scopes that enclose the scope in which they are declared. They will override the variables in the outer scopes which are otherwise accessible to the enclosed scope. Variables of the same name as other variables in the same scope will generate an error however. A statement can be one of the following, each of which will be detailed further below:

- Function declaration: A named list of statements with parameters that appear to the list of statements as declared variables in their scope but whose values are copied from expressions or else refer to variables supplied by a function-call which is a type of expression. Function declarations may only occur in the topmost scope of the script.
- Structure declaration: A declaration of an entity which can have both data members and member functions. The member functions all have an implicit first parameter which is a reference to the object instance upon which the function is being called.
- Variable declaration: A declaration of a variable which then exists in the scope in which it was declared. Variables can be simply declared or as part of their declaration they can be initialized by assigning them to an expression or

constructed by supplying constructor parameters in parenthesis after the variable. Note that scalar types have no constructors. On types that do have constructors, the parameter list must match a constructor that exists for that type.

- Expression: This can be a function call, a constant value, a variable, a member selection (which is either a member variable or member function), and arithmetic and logical operations on variables, constants, etc.
- While loop: This is a control structure that has a block/statementlist/scope that is executed for as long as a given condition (expression) resolves to true.
- Repeat loop: This is a control structure that executes its block a given integer amount of times that an expression resolves to.
- If statement: This is a control structure that executes a block upon an expression evaluating to true and executes an optional else block if it is false.
- For loop: This control structure has 3 lists of expressions separated by semicolons enclosed in parentheses. Within the lists (which can be empty, have one expression, or more than one) if there are more than one expression, they are separated by commas. The first list is executed initially. The next is a condition that must be true for the block to execute (in a list of expressions all must be true), and the last is the iterator list of expressions that gets executed each time through the loop.
- Block: Note that many statements own blocks. A block may also exist on its own without being owned by a control statement or function declaration. An unowned block may be coded in circumstances when construction and destruction of something needs to happen at a given point that cannot be accomplished in any of the "normal" scopes.

C. Function Declarations

[0157] Functions are declared using the "function" keyword. They are declared as follows:

function [<scalar-return-type] <functionname>(<argument-list>)

{

<body>

}

[0158] When a return type is not specified, int is implicitly assumed. The argument-list can be empty (i.e., "function <functionname>()...".) Every script typically needs a "function faslmain()" and will error if one is not found. The function faslmain() can be explicitly called from anywhere in the script but if no call to faslmain() is found in the topmost scope, a call is implicitly inserted at the end of the script. Functions may be overloaded, such that multiple functions with the same functionname can exist provided their argument-lists are different. The argument-list has the form "<argument>, <argument>, ... , <argument>". An argument can be the following forms:

- Pass by copy. <scalar-type> <distinct-variable-name>: Anything the function does to this variable is not reflected in the corresponding expression supplied in the call. The expression can be constant.
- Pass by reference. <scalar-type>& <distinct-variable-name>: Anything the function does to this variable is reflected in the corresponding variable supplied in the call. The call must supply an expression that reduces to a variable corresponding to this argument (i.e., only variables or structure member selections that reduce to variables.)

[0159] The (possibly empty) <body> consists of a list of statements. If a statement which is "return <expression>" is encountered anywhere in this body, the execution of the function ceases at that point and control is returned to the caller of the function with the value returned being the value that function presents to the expression in which it participates. Functions can be called without using the value they return as well.

[0160] For example,

```
----------------------
function faslmain()
{
     int  x;

     x = 5;
}

function string DoStringStuff(int x)
{
     return intToString(x);
}
```

```
function void DoStringThing(string& s)
{
     s = "the string";
     // DoStringThing(szThing) will set
     // szThing to "the string"
}
---------------
```

Variable Declarations

[0161] Variable declarations are declared as follows:

<typename> <var>[, <var>...];

[0162] In the case of structures, <typename> is typically expressed as "structure <structurename>" or, alternatively, a structure variable of type "OBJECT" can be declared as "OBJECT o;".

[0163] A <var> is one of the following:

- <identifier>: This is a simple declaration. If <typename> is a structure and the default (no arguments) constructor exists, it is called.
- <identifier> = <initializer expression>: The expression is evaluated and the result assigned to the variable at declaration time. Note that the type returned by the expression must be compatible with <typename>.
- <identifier>(<params>): <params> is a comma separated list of expressions. The constructor matching that signature in <typename> is called. Note that <typename> needs to be a structure.
- <identifier>[<array-dimension-expression>]: <typename> must indicate a scalar (non-structure) type. The expression must resolve to a constant integer (i.e., no variables or function calls in it). This declares <identifier> to be an array variable of given dimension of <typename>.

  For example,

```
------------
OBJECT o;
structure OBJECT o;
OBJECT o(5, 6, "something");
int x = 8, y, z[6 + 7];
-----------
```

D. Structure Declarations

[0164] Structures are declared using the "structure" keyword. They are declared as follows:

structure <structurename> [extends <parentstructurename>]

{

<member-list>

};

[0165] The parameter <member-list> is a possibly empty list of <member>s. A <member> is typically either a function declaration or a variable declaration. The only syntactic difference between these things when used in structure declarations versus not is that member variables cannot have initializer expressions or constructor parameters. Member variables can be arrays however.

[0166] When "extends" is used, this structure "inherits" all the members of the parent. Any member functions in this structure that are the same name and signature as one that is in the parent structure will override that parent structure's function.

[0167] A "constructor" is a member function that gets called when a structure variable is declared and a "destructor" is a member function that gets called when a variable goes out of scope.

[0168] Any member function whose name is <structurename> is a constructor and can have any or no parameters. There can be zero or one member function whose name is ~<structurename> and has no parameters, and this is the destructor. Some subtleties that are not immediately apparent are as follows: If this structure extends an object that has constructors and/or a destructor or has member variables that are structures that have constructors and/or a destructor, then each constructor in this object will implicitly contain a call to the default constructor in the parent and/or owned object(s) and similarly the destructor for this object will have call(s) to destructors of parent/owned object(s). Further, if a parent or owned object has a constructor and/or destructor and this object does not, one will be created in this object for the purpose of calling all the constructors/destructors of parent/owned objects. Constructors and destructors should be declared as "function <structurename>..." or "function ~<structurename>..." (i.e., no return type.)

[0169] Usage of declared structure variables is accomplished with member selection using the "." character. For example:

```
------------
OBJECT o;
o.m_intmember = 5;
o.m_ownedstructuremember.m_x = 8;
o.DoSomething(5, 6, "stuff");
------------
```

E. Expressions

[0170] Expressions are variables, function calls, constants, member-selections, and combinations of these things with operators serving as the connectors. For the purposes of this discussion, expressions may be defined recursively, such that, for example, <expression> + <expression> is an expression. In expressions involving operators (assignment, arithmetic, logical, etc.,) it is customary to refer to the expression on the left as the "lvalue" and the expression on the right as the "rvalue." In FASL as most languages, the lvalue and rvalue in any expression must return compatible types (usually the types must be the same but sometimes, as with int and char in FASL, different types can be compatible). Additionally, some operators only work on some types. There is also the notion of operator precedence, meaning that in the absence of overriding parentheses, some operators will be given precedence over others. So for example, 3 - 5 + 7 will be evaluated left to right but 3 - 5 * 7 will multiply 5 by 7 first and subtract the result from 3. A list of all the operators in order of precedence follows:

Functions and constants

- function calls: <function-name>([<argument-list>]); <argument-list> is a possibly empty, comma-separated list of expressions. There must be a function somewhere in the script which takes parameters that match the return types of these expressions in the order in which they occur. In the case of functions that take references, the corresponding expression in the <argument-list> of the function call must resolve to a variable (not a constant or arithmetic/logical/relational expression of multiple variables). The type of a function call expression is the return type of its corresponding function declaration.
- string constants: "sample string \x3f"; these are string values enclosed in quotes which mostly represent the literal characters that comprise the string. The

exceptions to this are the escape characters: "\t" is tab, "\n" is newline, "\r" is carriage return, and "\x[0-9a-fA-F][0-9a-fA-F]*" are embedded hex codes that resolve down to one character. Note the regular expression simply means for example "\x3F-sample" will resolve down to a 47 (0x3F) followed by a '-' followed by a 's,' etc. Any embedded zeros ("\x0") will terminate the string when the constant is used in string expressions, but when used in binary expressions, the total string will get processed to resolve the escape sequences and then converted to type binary. String constants typically have the same precedence as functions.

- char constants: For example, 'A'. These are treated the same as string constants, except they are single quotes and single character (which may be represented with escape sequences).
- int constants: These can be decimal digits as in "1048576" or a hexadecimal number as in "0x100F."

Unitary operators

- ++<variable>: increment value of variable and then return its value. Works on int and char.
- --<variable>: decrement value of variable and then return its value. Works on int and char.
- <variable>++: return value of variable then increment it. Works on int and char.
- <variable>--: return value of variable then decrement it. Works on int and char.
- -<expression>: negate whatever is returned by <expression> and return that value. Works on int and char.
- ~<expression>: flip bits of whatever is returned by <expression> and return that value. Works on int and char.
- !<expression>: logical negation of <expression>. Works on boolean.
- sizeof(<typename> or <variable>): returns int that is how many stack cells is occupied by the <typename> or <variable>.

Member select operator

- <structurevariable>.<member>[.<member>...]: returns whatever type/value the rightmost <member> is.

Power operator

- <expression> power <expression>: lvalue and rvalue must be int or char. If int and char mixed, promote char to int.

Multiply operators

- <expression> * <expression>: multiply... lvalue and rvalue must be int or char. If int and char are mixed, promote char to int.
- <expression> / <expression>: divide... lvalue and rvalue must be int or char. If int and char are mixed, promote char to int.
- <expression> % <expression>: modulo... lvalue and rvalue must be int or char. If int and char are mixed, promote char to int.

Addition operators

- <expression> + <expression>: add... same type rules as multiply operators but also does concatenation of two strings (returns string) or two binaries (returns binary). If one of the expressions is a string constant and the other is a binary variable, the string constant will be considered a binary constant.
- <expression> - <expression>: subtract... same type rules as multiply operators but also works on strings (not binaries as operator + does). In the case of strings, subtraction removes all substrings from lvalue that match rvalue and returns the resulting string.

Bitwise operators

- <expression> << <expression>: shift left one bit (effectively multiply by 2). Same type rules as multiply operators.
- <expression> >> <expression>: shift right one bit (effectively divide by 2 losing remainders). Same type rules as multiply operators. Most significant bit (sign bit) is repeated in next bit over.
- <expression> & <expression>: bitwise and. Same type rules as multiply operators.
- <expression> | <expression>: bitwise or. Same type rules as multiply operators.
- <expression> ^ <expression>: bitwise exclusive or. Same type rules as multiply operators.

Relational operators

[0171] All of these operators typically return boolean regardless of the types on which they operate (in contrast to most operators which return the same type as that on which they operate). Unless otherwise noted, lvalue and rvalue can be int/char and int/char or string and string. If they are strings then the comparison is alphabetic case sensitive.

- <expression> < <expression>: Return true if lvalue is less than rvalue.
- <expression> <= <expression>: Return true if lvalue is less than or equal to rvalue.
- <expression> > <expression>: Return true if lvalue is greater than rvalue.
- <expression> >= <expression>: Return true if lvalue is greater than or equal to rvalue.
- <expression> == <expression>: Return true if lvalue equals rvalue.
- <expression> != <expression>: Return true if lvalue does not equal rvalue.
- <expression> in <expression>: This only works on string and binary expressions. Return true if lvalue occurs as a substring/subbinary pattern in rvalue.

Logical operators

[0172] These operators expect lvalue and rvalue to return boolean and the operators return boolean.

- <expression> || <expression>: "or else"... if lvalue is true then return true without evaluating rvalue, else return whatever rvalue returns.
- <expression> && <expression>: "and then"... if lvalue is false then return false without evaluating rvalue, else return whatever rvalue returns.

Assignment operators

[0173] These operators insist on type compatibility which means equality of types most of the time. Exceptions are: when mixing ints and chars, lvalue type prevails, and when assigning string constants into binary variables the string constant becomes a binary constant. Lvalue and rvalue must resolve to scalar types and lvalue must resolve to a variable.

- <expression> = <expression>: Simple assignment: copy rvalue into lvalue.
- <expression> *= <expression>:

- <expression> /= <expression>:
- <expression> %= <expression>:
- <expression> += <expression>:
- <expression> -= <expression>:
- <expression> >>= <expression>:
- <expression> <<= <expression>:
- <expression> &= <expression>:
- <expression> |= <expression>:
- <expression> ^= <expression>: All the above perform the operator preceding the "=" on the rvalue according to the rules of that operator specified above and then put the result in the lvalue.

Conditional expression

[0174] There is also a construct identical in principle and syntax to the conditional expression in C/C++ . Its syntax is:

(<expression>) ? <expression-eval-if-true> : <expression-eval-if-false>;

If the expression in parentheses evaluates to true then the expression after the question mark is executed otherwise the expression after the colon is executed. The expression in parentheses must resolve to boolean and the other two expressions must be type compatible. The return type/value of a conditional expression is the return type/value of the expression that executes after evaluating the conditional expression.

F. Control Structures

While loops

A while loop is expresses as:

while (<expression>)

{

<statement-list>

}

or,

while (<expression>)

<statement>;

[0175] This evaluates <expression> which must return boolean and executes the <statement-list> or <statement> and then reevaluates <expression> for as long as <expression> returns true. If <expression> returns false the first time then the <statement-list> or <statement> is never executed.

Repeat loops

repeat (<expression>)
{
<statement-list>
}

or,

repeat (<expression>)
<statement>;

[0176] This evaluates <expression> once (in contrast to most loop constructs which execute their "iterator" expression each time before executing the block), and it must return an int or char indicating how many times to execute the block. The block is then executed that many times. In case the implications of this are not clear, note that in a while loop, for example, the body of the loop must do something that eventually causes the <expression> to evaluate to false, otherwise the loop goes on forever. But if the repeat loop's block does something that would cause the <expression> to evaluate differently, this makes no difference as the <expression> is only evaluated once before the block is executed. So "v = 2; repeat (v) {v = 5;}" will not cause an infinite loop. Rather, it will assign 5 to v twice.

If statements

if (<expression>)
{
<statement-list>
}

or,

if (<expression>)
{

<statement-list>

}

else

{

<statement-list>

}

[0177] For the sake of brevity, all possible options of using a single <statement> instead of {<statement-list>} have not been enumerated, but those options exist. If <expression>, which must resolve to boolean type, evaluates to true then the first block is executed. If it evaluates to false and the else block exists, it is executed, otherwise no block is executed.

For loops

for (<expression-list>; <expression-list>; <expression-list>)

{

<statement-list>

}

or,

for (<expression-list>; <expression-list>; <expression-list>)

<statement>;

[0178] The <expression-list> can be nothing, it can be a single <expression>, or it can be a comma-separated list of expressions. The first <expression-list> gets called initially and can be any mix of expressions as the return values are ignored. The middle <expression-list> is the condition that determines whether the block will be executed and whether the loop will continue to be iterated. All expressions in this list must return boolean and they all must return true in order for the block to be executed. The last <expression-list> can be any mix of expressions and it gets executed after each time the block has been executed.

[0179] The classic example is "for (x = 0; x < 8; x++) {<statement-list>}" This will set x to 0, test whether x is less than 8, find that to be true and execute the block, increment x to 1, test whether x is less than 8, find that to be true and execute the block... etc. until x is 8 which will cause the condition to be false and loop execution terminates.

G. Native Objects

[0180] The grammatical/syntactic elements of one embodiment of the FASL language have been specified above. Notice that these things are sufficient to perform calculations and implement algorithms, but there is nothing about them that allows for things like sending and receiving data over the network and there are certain operations on the scalar types that would be nice to have but are not expressible with the grammar thus far specified. For this reason there exist structures and functions declared in the scope that owns the script scope (a "meta-scope" if you will) that allow access to this functionality. As used herein "Faultline" refers to the overall network security system.

FASL Object

[0181] Every script has access to a variable named "FASL" whose type is "FASL_OBJECT." This variable's type is specified as follows:

```
--------------
structure FASL_OBJECT
{
private:
     string     m_szName,
                m_szDescription,
                m_szSummary,
                m_szReturnString;
     RETURNVALUE     m_eReturnCode;
     int        m_nDefaultPort,
                m_nIPAddress;
     ATTACK     m_eAttackType;
     OPERATINGSYSTEM
                m_eOSMajor,
                m_eOSMinor;
     PROTOCOL   m_eDefaultProtocol;
public:
     function FASL_OBJECT()
     {
          m_nDefaultPort = 0;
          m_nIPAddress = 0;
          m_szReturnString = "Return string not set."
     }
     function void setScriptName(string s)
     {
          m_szName = s;
     }
     function void setScriptVulnerabilityCode (int
nFaultlineID)
     {
```

```
        // This sets the vulnerability id as it exists in
the Faultline database that uniquely
        // identifies the vulnerability being sought by
the script.
        m_nFaultlineID = nFaultlineID;
    }
    function void setScriptDesc(string s)
    {
        m_szDescription = s;
    }
    function void setScriptSummary(string s)
    {
        m_szSummary = s;
    }
    function void setScriptAttackType(ATTACK e)
    {
        m_eAttackType = e;
    }
    function void setScriptReturn(string szReturnString,
RETURNVALUE eReturnCode)
    {
        m_szReturnString = szReturnString;
        m_eReturnCode = eReturnCode;
    }
    function void addValidPort(int n)
    {
        // When a script successfully compiles, it will
execute all these that it finds
        // in the main scope.  It builds a list of ports
on which to run the script.
        m_nValidPort = n;
    }
    function void setScriptPort(int n)
    {
        m_nDefaultPort = n;
    }
    function void setScriptOS(OPERATINGSYSTEM eMajor,
OPERATINGSYSTEM eMinor)
    {
        // When a script successfully compiles, it will
execute all these that it finds
        // in the main scope.  It will use this
information to decide whether this script
        // needs to be run on a given host.
        m_eOSMajor = eMajor;
        m_eOSMinor = eMinor;
    }
    function void setScriptProtocol(PROTOCOL e)
    {
        m_eDefaultProtocol = e;
    }
```

```
    function int getIPAddress()
    {
        return m_nIPAddress;
    }
    function int getScriptPort()
    {
        return m_nDefaultPort;
    }
    function string strTok(string& s, string szDelimiters)
    {
        // Like strtok in UNIX, this skips past any
characters at the beginning of the string
        // that are in szDelimiters.  Return the substring
that includes all the leftmost
        // characters up to but not including the next
instance of a character in
        // szDelimiters, or the end of the string
whichever comes first.  Remove the
        // returned string from s (note it is a
reference).
        return STRTOK(s, szDelimiters);
    }
};
-------------
```

[0182] When a script is successfully compiled, all statements of the form "FASL.memberfunction()" that are in the main scope are executed (and no other statements in the script are executed at this point). The information that these member functions set goes into the script object's data which allows the system to make some intelligent decisions on when and how to run the script. FASL.addValidPort(nPort) can be called any number of times and will result in a list of unique ports being built up. When the system is run, it will either find that FASL.addValidPort(nPort) was not called in the script's main scope in which case the script will be run once per host and FASL.getScriptPort() will return 0 within that script. If FASL.addValidPort(nPort) does occur in the main scope of the script, the system will execute the script once for each unique port on a given host, and FASL.getScriptPort() will return whatever port the system passes in for that execution. FASL.setScriptOS() operates along a similar principle (i.e., by calling it you are requesting that the script be executed only on hosts whose OS has been specifically determined whereas not calling it implies that the script is to be called regardless of OS). Note that calling this multiple times does not make a list

of OSes like the addValidPort makes a list of ports. The last call to setScriptOS is the one that gets used.

[0183] Upon entry into the script's scope, the m_nIPAddress member of the FASL variable has been set to the IP address of the target machine upon which the FASL script is to run. All network activity that takes place in a FASL script uses this IP address as its destination, hence in the functions and objects about to be specified, the IP address is never a parameter. Calling FASL.setScriptReturn(string, RETURNVALUE) sets the m_szReturnString member which is then printed by the script executor command line application "fasl.exe" upon completion of executing the script. The other calls, most notably setScriptOS(), set info that is used by Faultline.exe to determine whether or not to run the script on a given host. This is to say, the system will execute only the "FASL.xxx()" statements in the script, then the system examines the memberdata and decides whether or not to run the whole script on that IP address. In one embodiment, the constructor of FASL_OBJECT is actually not called on the FASL variable. Thus, any variable other than m_nIPAddress is sometimes not be initially set to anything. Structures may be derived from FASL_OBJECT and variables of type FASL_OBJECT may be initialized, in which case that constructor will be called. Note also that the member variables of FASL are not directly accessible by script code (declared as private).

Socket object

[0184] This is a structure declared in the meta-scope but there are no instances declared in the meta-scope as there is with FASL_OBJECT. It is mostly like a normal FASL structure, but the constructor creates a windows TCP or UDP socket depending on what you pass in for the first argument of the constructor and the destructor cleans up that object. The member functions permits data to be sent or received on that socket, assuming it connected properly. The structure is as follows (the data members are omitted as it is not necessary to know them):

```
--------------
structure Socket
{
     function Socket(PROTOCOL eProtocol, int nPort)
     {
          // eProtocol is either TCP or UDP and the port is
the IP port.
          // Creates a windows socket object.
```

```
    }
    function ~Socket()
    {
        // Cleans up the Windows socket object associated
with this socket.
    }
    function boolean socketOpen()
    {
        // Typically must be called before sending or
receiving data for either protocol.
        // if returns false then it could not open the
socket and communication
        // will fail.
    }
    function boolean socketClose()
    {
        // Close the socket. destructor will do this if
not done here.
    }
    function void BindPort(int nPort)
    {
        // Use when you want specify the source port
explicitly.
        BIND_PORT(nPort);
    }
    function string socketRead(int nBytes, int nTimeout)
    {
        // Read nBytes from Socket... returns empty string
if fail.
    }
    function string socketReadLine(int nBytes, int
nTimeout)
    {
        // Read nBytes or up to end of line whichever
comes first.
        // return empty string on fail.
    }
    function binary socketReadBinary(int nBytes, int
nTimeout)
    {
        // Read nBytes of raw binary data. empty binary on
fail.
    }
    function int socketWrite(string szBuffer, int nLength)
    {
        // Write nLength characters out of szBuffer.
return 0 on fail
        // or number of bytes written on success.
    }
    function int socketWrite(string szBuffer)
    {
```

```
          // Write the entire string (length implicit),
otherwise same as above.
     }
     function int socketWrite(binary bBuffer)
     {
          // Write the binary raw data... length implicit,
otherwise same as above.
     }
};
-------------
```

Vulnerability object for Active Assessment

[0185] This object is used for active assessment, and its custom behavior — atypical of general FASL script — is that it knows how to copy itself, and all the data it needs to find itself later, into a global data area where other scripts can access the data, and use the information to attempt to compromise another machine. The object is as follows:

```
--------------
structure Vulnerability
{
private:
     int  m_nFaultlineID,
          m_nIPAddress,
          m_nExploitIPAddress;
     string    m_szDescription;
public:
     function Vulnerability()
     {
          m_nFaultlineID = 0;
          m_nIPAddress = 0xFFFFFFFF;
          m_szDescription          =          \"Vulnerability
uninitialized.\";
     }
     function void addToExploitableData(
          int nFaultlineID,
          int nIPAddress,
          int nExploitIPAddress,
          string szDescription)
     {
          // This sets all the member variables of this
structure.  This function may be
          // called from a derived structure and this
function will know that.  It stores the
          // entire contents of the object as well as the
object's typename and size in the
          // global vulnerability data area.
     }
     function boolean getExploitableData(int nIndex)
     {
```

```
          //      this function searches the global
vulnerability
          // data area for the nIndex-the instance of a
variable of the same type as this (this
          // could be and probably will be a derived
structure from Vulnerability) and copies
          // its contents into this object.  If there is no
nIndex-th object, return false and
          // no change to this object's data will have
occurred.
     }
     // Accessor functions. the members of this structure
need to be read only once they
     // have been stored with addToExploitableData().
     function int getFaultlineID()
     {
          return m_nFaultlineID;
     }
     function int getIPAddress()
     {
          return m_nIPAddress;
     }
     function int getExploitIPAddress()
     {
          return m_nExploitIPAddress;
     }
     function string getDescription()
     {
          return m_szDescription;
     }
};
```

[0186] Behind this object is the concept that one script may discover something that another script can use to compromise another machine. Any vulnerability needs the information contained in this base class, in this embodiment. The m_nFaultlineID is the vulnerability id of the vulnerability discovered. The m_nIPAddress is the machine it was discovered on. The m_nExploitIPAddress is the machine on which the data was discovered that proved instrumental in finding this vulnerability. The m_szDescription is what you want to appear in the Active-Assessment report. The m_nExploitIPAddress should typically be set to -1 when no other vulnerability was used to find this vulnerability. When another vulnerability is used, its m_nIPAddress (typically using othervuln.getIPAddress()) is entered into this vulnerability's m_nExploitIPAddress. This will establish an audit trail which the report can represent graphically.

[0187] The general, usage of this is to derive a structure from Vulnerability, (i.e., "structure UnicodeVulnerability { // extra info specific to unicode vuln that can be used by other scripts};"). When a unicode vulnerability is found, for one example, a variable such as "UnicodeVulnerability uv;" is created, and its extra data is set and added via a call to "uv.AddToExploitableData(...)." After this call, another script that looks to make use of this particular vulnerability has code resembling the following:

```
----------
int             i;
UnicodeVulnerability     uv;

for (i = 0; uv.getExploitableData(i); i++)
{
     // Attempt to use this data to compromise this host.
     if (succeed at compromising this host)
     {
          // create a vulnerability representing this host's
vulnerability that was
          // found using the UnicodeVulnerability.   Note
that this vulnerability
          // may or may not be a UnicodeVulnerability... it
could be some other
          // vulnerability.  When you have populated it with
its specific data, call:
          newvuln.addToExploitableData(
               nNewVulnFaultlineID,
               FASL.getIPAddress(),
               uv.getIPAddress(),
               "we got this vuln by exploiting Unicode on
another machine");
     }
}
-----------
```

Debug messages

[0188] There is a function called "debugMessage(<scalar> e)" overloaded for all the scalar types. Mostly it prints what is expected from a debugging system. Int and char print as numbers, boolean prints as "true" or "false", strings print their contents. Binary produces a hex dump output, through debugMessage(binary b), which closely resembles what the MS-DOS DEBUG.EXE "d" option produces. That is, it outputs out lines representing 16 bytes each and the line format is "<4 digit hex offset>: <8 2-digit hex numbers separated by space> - <the other 8> <16 chars where printable chars are represented as is and non-printable are represented as periods>. For example:

"0060: 67 65 6C 73 20 6C 6F 6F - 6B 20 6C 69 6B 65 20 74 gels look like t"

[0189] With the functions about to be specified, it is easy to convert other types into strings so you can debugMessage entire expressions... i.e. debugMessage("here is what x equals: " + intToString(hex));. There is typically no need to remove or comment out debugMessage calls in scripts that get "signed off," as final as the system execution of scripts will typically ignore the debug output.

String functions

```
------------
function string leftTrim(string s)
{
     // Lops off whitespace up to the first non-whitespace
character.
}
function string rightTrim(string s)
{
     // Lops off all trailing whitespace.
}
function int stringLength(string s)
{
     // self-explanatory.
}
function string toLower(string s)
{
     // Makes any characters that are upper-case lower-case.
}
function string toUpper(string s)
{
     // Makes any characters that are lower-case upper-case.
}
function int stringToInt(string s)
{
     // similar to atoi in C.
}
function string intToString(int n)
{
     // i.e 1048576 becomes string "1048576"
}
function string intToString(char c)
{
     // Similar to intToString(int).
}
function string intToHexString(int n)
{
     // i.e. 16383 becomes "3FFF"
}
function int hexStringToInt(string s)
```

```
{
     // i.e. "3FFF-blahblahblah" becomes 16383.
}
function string intToBinaryString(int n)
{
     // i.e. 85 becomes "1010101"
}
function int binaryStringToInt(string s)
{
     // i.e. "1010101blahblahblah" becomes 85.
}
function string intToIPString(int n)
{
     i.e. 16777215 becomes "0.255.255.255"
}
function string grep(string s1, string s2)
{
     return \"to be implemented\";
}
function int locate(string s1, string s2)
{
     // return 0-based position of s1 in s2 or -1 if s1 not
in s2.
}
function   string   subString(string   s,   int   nStart,   int
nNumChars)
{
     // i.e. subString("one law for the lion and ox is
oppression", 4, 3) = "law"
     // it is smart about boundries... if you ask for more
characters than exist
     // you only get the ones that exist.
}
function string garbageString(char c, int nLength)
{
     // i.e. garbageString('A', 7) = "AAAAAAA"
}
function string garbageString(int nLength)
{
     // return string of nLength length whose characters are
random
     // upper and lower case alphanumerics.
}
----------
```

Binary functions:

```
----------------
function int binaryLength(binary b)
{
     // self-explanatory.
}
```

```
function char binaryNthByte(binary b, int n)
{
     // return nth byte numbered from 0.  If out of bounds,
     // return -1.
}
function boolean binaryChangeByte(binary& b, int n, char c)
{
     // Change byte #n in binary b to c.  If out of bounds,
     // do nothing to b and return false.
}
function binary binaryRight(binary b, int n)
{
     // return a binary which is the rightmost n bytes of b.
If there
     // are not n bytes in b, you get all of b (not padded
to n).
}
function binary binaryLeft(binary b, int n)
{
     // same like binaryRight except from left.
}
function binary binaryMid(binary b, int nStart, int nLength)
{
     // nStart is 0-based... return binary which is the
nLength bytes starting
     // at nStart.  If there are not nLength bytes, return
however many there are.
}
----------
```

General global functions:

```
--------------
function string getLocalHostIP()
{
     // return string "xxx.xxx.xxx.xxx" representing the IP
of the machine
     // on which the FASL script is running.
}
function string getTargetHostIP()
{
     // This is in here for compatibility reasons... you can
get the same
     // effect by doing:
     intToIPString(FASL.getIPAddress());
}
function int getTargetPort()
{
     return FASL.m_nDefaultPort;
}
function boolean RPCCheck(int nRPCProgNum)
{
```

```
    // Attempt to make an RPC call on the given nRPCProgNum
to see if it exists
    // on the target machine.  Return true if it does.
}
function boolean SMBCheck(string szUserName, string
szPassWord, string szShare)
{
    // Attempt to do a "net use" sort of thing on the given
share... i.e. "IPC$",
    // "\\HOSTNAME\C$", "\\xxx.xxx.xxx.xxx\C$", etc.  If it
succeeds then
    // promptly delete the share and return true, otherwise
return false.
}
--------------
```

[0190] This implementation of the FASL scripting language permits one to perform active assessment of potential vulnerabilities on a known target computer with known open ports. For example, a FASL script to test a unicode vulnerability has the form:

```
-----------
structure UnicodeVulnerability extends Vulnerability
{
    string    m_szUnicodeString;
};

function faslmain()
{
    UnicodeVulnerability    uv;

    uv.m_szUnicodeString = "The unicode string";
    uv.addToExploitableData(1,    FASL.m_nIPAddress,    "the
string for reporting purposes");
    debugMessage("vulnfind executed.");
}
structure UnicodeVulnerability extends Vulnerability
{
    string    m_szUnicodeString;
};

function faslmain()
{
    UnicodeVulnerability    uv;

    if (uv.getExploitableData(0))
    {
        debugMessage("getExploitableData()...
m_nFaultlineID = " + intToString(uv.m_nFaultlineID) + ",
m_nIPAddress = " + intToIPString(uv.m_nIPAddress) + ",
m_szDescription = " + uv.m_szDescription + ",
m_szUnicodeString = " + uv.m_szUnicodeString);
```

```
    }
    else
    {
        debugMessage("getExploitableData() failed.");
    }
}
-----------------
```

[0191] Notably, information collected from a successful vulnerability test, in one embodiment, is used to advantageously further test the vulnerability of the target network and individual target computers. In particular, in the event of a successful vulnerability test, additional ports or levels of account access are typically available on the target computer. Thus, even if other prior vulnerability tests failed, they are advantageously retried after a successful vulnerability test.

[0192] Figure 8 illustrates one embodiment of active assessment of a vulnerability of a target computer on a target network. For simplicity, the network is shown as having a target 1 computer 800 and a target 2 computer 802. For simplicity, a single vulnerability is assumed to apply to all TCP and UDP ports of the two computers. A single vulnerability "TEST" is tested on various TCP and UDP ports on both target computers. Prior host discovery and port discovery have provided target 1 data 804 and target 2 data 806 that includes an identification of open ports found on each target computer. Specifically, in the example illustrated in Figure 8, TCP port 80 and UDP ports 5000 and 53 were found open on the target 1 computer 800, and TCP ports 23 and 80 and UDP ports 500, 53, and 1721 were found open on the target 2 computer 802. The active assessment process starts in a step 806 and begins by executing a TEST routine 808 that applies the TEST vulnerability to each port on the target 1 computer 800. In the example of Figure 8, in a first round 810 of testing the target 1 computer 800, the testing of ports 80, 5000 and 53 result in no successful applications of the vulnerability. The system then moves in to a first round 812 of testing the target 2 computer, where testing of ports 80 and 53 are unsuccessful, but testing of ports 23, 5000 and 1721 are successful. Upon successful testing, an attempt is made to strip useful information from the target computer, and any information retrieved is stored in the target 2 data 806. Based on the new information retrieved, a second round 814 of testing the target 1 computer is attempted. In this attempt, testing of port 80 is still unsuccessful, but testing of ports 5000 and 53 is now successful with the inclusion of the additional information gathered from

the target 2 data 806. In this manner, a second round 816 of testing the target 2 computer is attempted. The rounds of testing are repeated until vulnerability assessments can be actively completed for the entire target network.

[0193] Known vulnerabilities are typically stored in a database for inclusion in vulnerability assessment testing, active assessment and reporting. In one embodiment, the "Vulns" table called "static data," which only changes when deploying new patches incorporating new vulnerabilities. This would happen, for example, as new vulnerabilities are discovered and existing in-the-field system installations need to be made aware of them. The present system uses, in one example, the VulnsFound table below to indicate which vulnerabilities it has found on a particular scan. VulnsFound is simply an index into the Vulns table which keeps us from otherwise repeating the same data a multitude of times and wasting space.

```
TABLE: Vulns
     COLUMN: BID, TYPE: varchar(10)
     COLUMN: CVE, TYPE: varchar(20)
          //"Common Vulnerabilities and Exposures" number.
     COLUMN: CyberID, TYPE: int(10)
     COLUMN: Description, TYPE: text(2147483647)
     COLUMN: ExploitDataType, TYPE: varchar(64)
          //Name of the FASL type, where applicable, that
          contains extra data for this vulnerability.
     COLUMN: ExploitDate, TYPE: smalldatetime(16)
     COLUMN: ExploitLink, TYPE: varchar(255)
          //Site where you can download the exploit/patch.
     COLUMN: FaultlineID, TYPE: int(10)
          //Primary key field... linked to VulnsFound table.
     COLUMN: Impact, TYPE: tinyint(3)
     COLUMN: ISSID, TYPE: int(10)
     COLUMN: LHF, TYPE: bit(1)
          //short term intrusion opportunity... true or
          false.
     COLUMN: Name, TYPE: varchar(128)
     COLUMN: NID, TYPE: int(10)
     COLUMN: Observation, TYPE: text(2147483647)
     COLUMN: Person, TYPE: varchar(50)
     COLUMN: Popularity, TYPE: tinyint(3)
     COLUMN: Recommendation, TYPE: text(2147483647)
     COLUMN: Risk, TYPE: tinyint(3)
     COLUMN: RiskText, TYPE: text(2147483647)
     COLUMN: Simplicity, TYPE: tinyint(3)
     COLUMN: Type, TYPE: varchar(50)
          //web, router, unix, trojan, etc.
```

```
TABLE: VulnsFound
     COLUMN: ConfigurationID, TYPE: int(10)
     COLUMN: CustomerID, TYPE: int(10)
     COLUMN: FaultlineID, TYPE: int(10)
     COLUMN: HostID, TYPE: int(10)
     COLUMN: JobID, TYPE: int(10)
     COLUMN: VulnFoundID, TYPE: int identity(10)
-------
```

VII. Quantitative Scoring of Target Network Vulnerability

[0194] Network vulnerabilities have traditionally been rated on a "low risk," "medium risk," and "high risk" scale. This subjective scale is based on the level of access to a target system granted by a vulnerability, the ease of detecting and exploiting the vulnerability, the public knowledge of a vulnerability, the percentage of computers subject to the vulnerability, and the like. However, this subjective scale lacks objective indicia allowing comparison of otherwise heterogeneous networks for the purpose of comparing relative security. Further, this subjective three-level scale provides little information about security improvement or added risks over time.

[0195] In one embodiment, the present system provides objective, quantitative indicia of the overall security of the target network. Advantageously, this quantitative indicia can be any quantitative scale of sufficient granularity to provide meaningful distinctions due to changes in network vulnerabilities over time, with a range of between 1-10, 1-100, and the like. Further, this objective indicia applies a standard formula to the various vulnerabilities discovered on the target network and network configuration, such that a valid comparison can be made between the security of otherwise heterogeneous network configurations, operating systems, and computers.

[0196] For example, the objective indicia is a risk measurement algorithm, such as a FoundScore **F**. In one embodiment illustrated by the flowchart in Figure 9, the FoundScore **F** is defined as:

$$\mathbf{F = 100 - V - E} \qquad \textbf{(Eq. 1)}$$

where,

F = FoundScore

V = Vulnerability Loss

E = Exposure Loss.

[0197] More specifically, in one preferred embodiment, the vulnerability loss **V** is defined as the sum of values assigned to each vulnerability found on the target network. For each of the **n** vulnerabilities found on the target network, that vulnerability is assigned a weight.

[0198] Thus, for a particular vulnerability V_x, where $1 \leq x \leq n$, the vulnerability weight $V_{w,x}$ is defined as:

$$V_{w,x} = \begin{cases} \{ 50 \mid PEA(V_x) \geq 2 \} \\ \{ 10 \mid 2 > PEA(V_x) \geq 1 \} \\ \{ 5 \mid 1 > PEA(V_x) \} \end{cases} \qquad \textbf{(Eq. 2)}$$

where:

$$PEA(V_x) = \text{floor}\left(\{P(V_x) + E(V_x) + A(V_x)\} / 3\right) \qquad \textbf{(Eq. 3)}$$

and:

floor(x) is the standard floor function,

$P(V_x)$ is the popularity of the vulnerability on a 1-3 scale,

$E(V_x)$ is the ease of exploitation of the vulnerability on a 1-3 scale,

$A(V_x)$ is the level of access/privileges granted by the vulnerability on a 1-3 scale,

where the higher the score on the scale the greater the risk.

[0199] Alternatively, the vulnerability can be simply defined as the risk level associated with the vulnerability, such as where:

$$PEA(V_x) = R(V_x) \qquad \textbf{(Eq. 3a)}$$

where,

$R(V_x)$ is the risk associated with the vulnerability on a 1-3 scale.

[0200] Typically, the factors used to determine the **PEA(V)** for a particular vulnerability are provided from the vulnerability database discussed previously. In other embodiments, the vulnerability base function (e.g., **PEA(V)**) can be defined from a number of different variables, including, for example, ease of distribution of a vulnerability, ease of preventing the vulnerability, and the like. Thus, the total vulnerability score V is, in one embodiment, equal to:

$$V = \textbf{min}\left(50, \Sigma_{(x=1 \to n)}\{V_{w,x}\}\right) \qquad \textbf{(Eq. 4)}$$

where:

n = number of vulnerabilities found on the target network,

Σ = the summation symbol,

$V_{w,x}$ = the vulnerability weight defined above, and,

min (x,y) = the standard minimum function.

[0201] The total exposure score **E** is, in one embodiment, defined as:

$$\mathbf{E = min\ (\ 50\ ,\ \Sigma_{(y=1\rightarrow q)}\ \{10\times U_y + 5\times I_y + N_y + 2\times M_y\}\ )} \qquad \textbf{(Eq. 5)}$$

where:

q = the number of live target computers found on the target network,

U_y = the open UDP ports found on the yth target computer, except DNS,

I_y = the open ICMP ports found on the yth target computer,

N_y = the non-essential services found on the yth target computer,

M_y = a penalty for target computers with no essential services,

Σ = the summation symbol, and,

min (x,y) is the standard minimum function.

[0202] In particular, the number of live target computers q is typically deduced from the host discovery process described above. The number of open UDP ports U_y found on the yth target computer is typically deduced from the service discovery and active assessment process described above. The number of non-essential services N_y found on the yth target computer is typically deduced from the service discovery of TCP and UDP ports, and in one preferred embodiment, counts all open ports found on each target computer, except for certain predetermined services. Table 11 lists examples of predetermined services that are not included in the count of nonessential services N_y in accordance with one embodiment of the present invention.

Essential Service	Ports
HTTP	80, 8000, 8080
SSL / HTTPS	443
FTP	21
DNS	53
SMTP	25

[0203] Table 11: Example Essential Services not Included in N_y count

[0204] Other combinations of essential services are also possible, based on changes in operating system, protocols, and the like. Finally, the no essential services penalty flag M_y is set to 1 for each target computer that has $N_y > 0$, and has no essential services running such as those shown in Table 7. Otherwise, M_y is set to 0.

[0205] As a result, the FoundScore $F = 100 - V - E$ provides a quantitative score between 0 and 100 representing an objective indicia of the relative security of the target network. Generally, the higher the Foundscore (i.e., the higher **F** is), the greater the relative security of the target network. Other embodiments are possible, such as, for example, where:

$$E_{alt} = [\ \Sigma_{(y=1\rightarrow q)}\ \{(5\times U_y + 2\times I_y + 2\times N_y + 2\times M_y)^2\}\ ]^{1/2} / q \qquad \text{(Eq. 7)}$$

$$V_{alt} = [\ \Sigma_{(x=1\rightarrow n)}\{(V_{w,x})^2\}\ ]^{1/2} / n \qquad \text{(Eq. 8)}$$

$$F_{alt} = E_{alt} + V_{alt} \qquad \text{(Eq. 9)}$$

[0206] In this alternative scoring regiment, the higher the F_{alt} score the worse the target network security.

[0207] Figure 9 illustrates one embodiment of a methodology for determining the security score for a target network. In a first decision step 900, the method determines whether all vulnerabilities found in the target network have been counted. If not, the method calculates $\mathbf{PEA(V_x)}$ (Eq. 3) above, or a variation thereof, in a step 902 for vulnerability number x based on predetermined values stored in a vulnerability database 904. The value of $\mathbf{PEA(V_x)}$ is used to calculate the weighted vulnerability $V_{w,x}$ (Eq. 2) for a vulnerability x in a step 906. Typically, the vulnerability counter is then incremented in a step 908 if necessary for a particular embodiment. Thereafter, the method returns to the decision step 900 and again determines if all the vulnerabilities have been counted.

[0208] After all the vulnerabilities have been counted, then in preferred embodiments, the total vulnerability V is calculated in a step 910. As discussed above, the total vulnerability V is the lesser of either 50 or the sum of all weighted vulnerability scores $V_{w,x}$ in this embodiment. Alternatively, other scores are possible, or the tabulation of the total score V can be combined with the prior loop.

[0209] Then, in a decision step 912, the method determines whether all target computers found in the target network have been counted. If not, the exposure values are

determined in a step 914. In preferred embodiments, the exposure values include U_y (the open UDP ports found on the yth target computer, except DNS on port 53), I_y (the open ICMP ports found on the yth target computer), and N_y (the non-essential services found on the yth target computer), are determined. The exposure values are determined dynamically based on the network security test, and are, in one embodiment, stored in the target computer vulnerability database 904 or in another database. In a step 916, a penalty M_y is determined for target computers with no essential services present. Preferably, the exposure counter is then incremented in a step 918, and the method returns to the decision step 912 to determine whether all the target computers have been counted.

[0210] When all the target computers are counted, the total exposure **E** is calculated in a step 920 as the lesser of either 50 or the sum of a weighted sum of the exposure values found. Then, in preferred embodiments, the score **F** is calculated in a step 992 as 100 minus the total vulnerability and exposure scores to generate a representation of a network security score. In preferred embodiments, a greater value of **F** implies greater network security on an objective scale.

[0211] Other exemplary embodiments apply various permutations of the factors associated with vulnerability scoring and exposure scoring, and are similarly foreseen as within the disclosure herein.

VIII. Graphical Hierarchical Reporting of Target Network Topology and Vulnerabilities

[0212] Generally, a conventional network security system provides reporting information in a text dump format. The collected data records regarding network topology, target computers, vulnerabilities, and the like are dumped into an ASCII file that requires substantial effort to interpret. Such data dumps are used conventionally because, other than alerting the network security user to the presence of high danger vulnerabilities, existing systems do not provide an interactive, hierarchical, and graphical representation of the data representing the network, target computers, and vulnerabilities found to assist the user in identifying and correcting the specific vulnerabilities.

[0213] The present system compiles the data discovered during security testing into a graphical, informationally hierarchical, and interactive set of documents for review at various levels of detail and documentation. Thus, in one embodiment of the present

invention, the reports engine produces (1) a dynamic graphical display representing the network topology, the target computers found, and the vulnerabilities found throughout the target network; (2) a comprehensive list of target computers, vulnerabilities found, and vulnerability explanations; (3) an objective scoring report describing the approximate security rating of the target network; (4) an exposure report describing the ports, services, and (5) detailed information that sets forth testing results on a per-machine, per-port, or per-vulnerability basis. Certain individuals within an organization may want different levels of detail. For example, upper management may only want the objective scoring report (3) that describes the approximate security rating of the target network. In contrast, the network administrator wants to receive all reports, particularly, the detailed information report (5) that enables the administrator to identify the machines and ports on the machines that need to be corrected.

[0214] For example, a functional representation of a report generated by the report engine typically includes a quantitative score representation including (1) the actual score, (2) a color representation of the score exposure, and (3) a graphical representation of the quantitative score. The actual score is the F score or an alternative embodiment, such as, for example, a 1 to 100 numerical rating as described previously. The color representation represents the overall vulnerability range of the score. Thus, in one embodiment, a score between 1 and 33 will have a red color signifying high vulnerability, a score between 34 and 66 will have a yellow color signifying medium vulnerability, and a score above 66 will have a green color signifying low vulnerability. Other colors, ranges, and representations, such as, for example, using icons or pictures to represent the vulnerability level, are foreseen.

[0215] Figure 10, for example, illustrates an embodiment of a hierarchical security report 1000 presented on a display for viewing by a user. The hierarchical security report includes a representation 1010 of the Foundstone score, showing a value of 15 corresponding to a relatively high risk. The security report further includes a graphical representation 1020 of the mapped network topology, including the live target computers and the services located on the target computers. The security report further includes a list 1030 of discovered hosts, and a list 1040 of discovered services.

[0216] The user may select any of the reports shown in Figure 10 (or other reports that are not shown) and request additional details. Figure 11, for example, illustrates an embodiment of a hierarchical security report 1100 that provides greater detail regarding the vulnerabilities of two target computers. For example, an upper portion of the display identifies a first target computer ("localhost.local.net" at IP address 255.255.255.255) having a first open port represented by a window 1110 and a second open port represented by a window 1120. Each of the open ports of the first target computer has a respective vulnerability and may identify a vulnerability patch to install on the target computer to reduce the vulnerability. As a further example, a lower portion of the security report 1100 identifies a second target computer ("localhost2.local.net" at IP address 254.255.255.255) having a single open port represented by a window 1130. The information in the window 1130 indicates that the service on the port is out of date, suggesting that it should be removed, and advantageously identifies a vulnerability patch to install if the service is to be retained.

[0217] There is an "object layer" implemented in C++ which represents the hierarchy of data objects that the system deals with, as described above with respect to the FASL scripting language. All objects follow the model of 1) containing the data directly relevant to that object as data members (these are 1-1 correspondent with a row in the database) and 2) possibly containing lists of other objects that also follow this convention. Each object has a Load() and Save() member that deals with the data for which that object is directly responsible by loading or saving it from the database. For the "child" object lists, the UpdateToDatabase() and UpdateFromDatabase() member functions are used. These call the object's Save() or Load() members respectively. They then recursively call the child lists' UpdateToDatabase() or UpdateFromDatabase() members. In this way, objects and their children can be selectively loaded from any point in the hierarchy. This was done because the objects are generally too unwieldy to deal with all at once.

[0218] Once the data is loaded, an object which represents a single scan has a GenerateReport member function which programmatically generates all the html, gif, jpg, and png files that constitute the report. The html files are generated by loading a "template" html file that contains boilerplate source and key words and replacing the key words with the html source representing the report data. The gifs and jpgs are simply

background graphics that do not change from report to report so they are written as is without the report mechanism having to know the details of gif and jpg encoding. For images that do need to be dynamically calculated, the png graphics format is used. This is a public (non-licensed) format for which there exists a public library which we are linking into the system executable. The images are drawn onto a Windows HDC (windows software object that represents a display device) and then converted into png by a custom class which wraps the object in the proper format.

[0219] An exemplary *Scan Summary Report* is set forth below in Appendix A. A corresponding exemplary *FoundScore Report* is set forth below in Appendix B. A corresponding *Network Topology Report* is set forth below in Appendix C.

[0220] Although the above description of the present invention includes at least one embodiment of the system, it should be understood that various changes, substitutions and alterations can be made to the system without departing from the spirit and scope of the invention as defined by the claims herein. For example, one alternative embodiment of the methodology described above is exemplified in Figure 12.

FoundScan for Acme Company: Scan Summary Report Page 1 of 5

Report Generated: 10-11-2001 15:10:07 Pacific Standard Time Navigate

Acme Company

Scan Name: Daily Scan

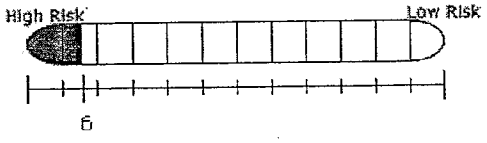
Scan Duration: 13 Minute(s), 23 Second(s)

Start Time: 19:00:00, July 20, 2001 **Stop Time:** 19:13:24, July 20, 2001

FoundScan results are driven by the Foundstone Assessment Methodology.

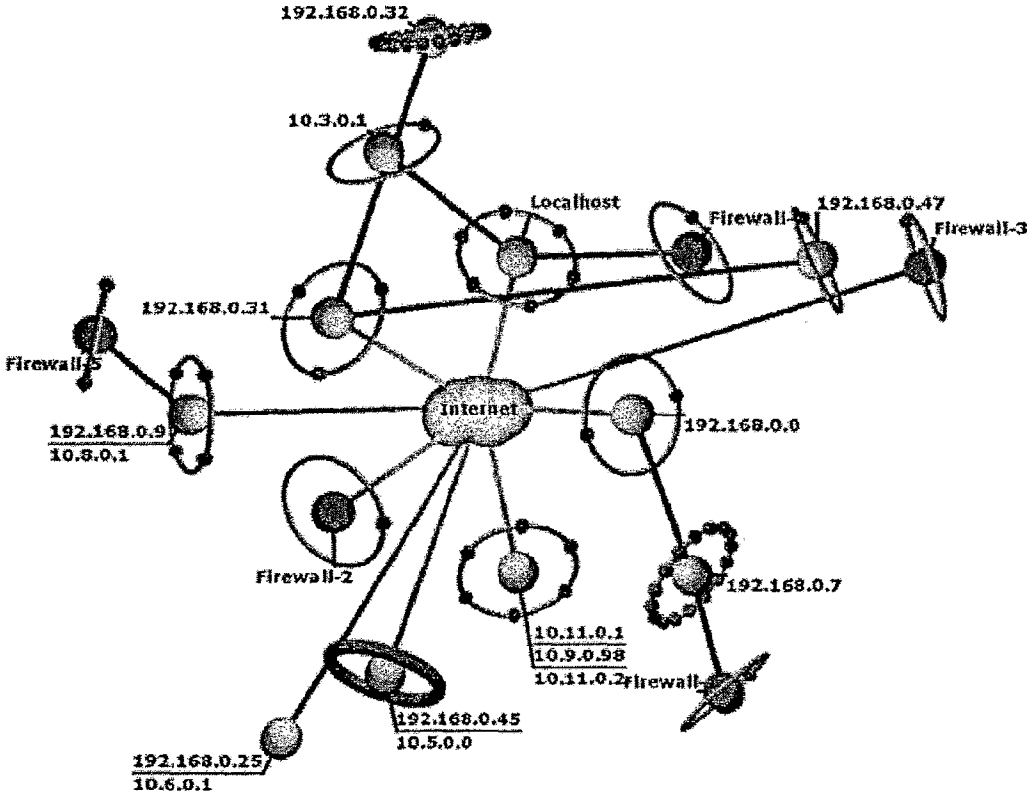
FoundScore Summary - Detailed Report

FoundScore is a numerical representation of your security exposure. For a detailed explanation of FoundScore, click here.



Network Map Summary - Detailed Report

Network Topology Summary: Acme Company

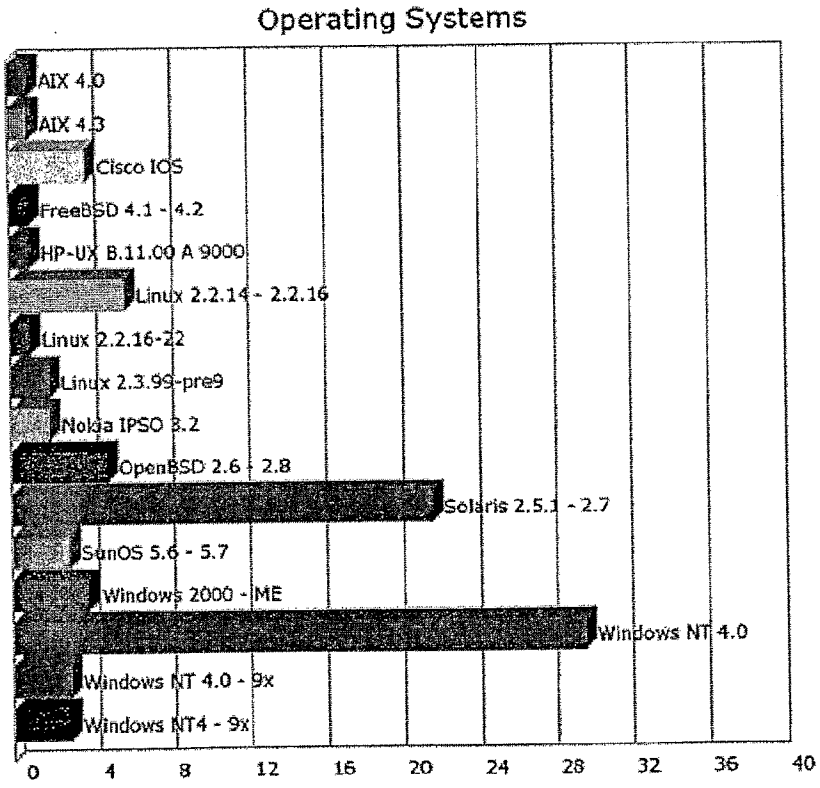


Discovered Hosts - Detailed Report

IP Address Range	Active IP Addresses	Potentially Active IP Addresses	Inactive IP Addresses	Total IP Addresses in Range
10.1.0.0-1	1	0	1	2
10.2.0.0-7	5	2	1	8
10.3.0.0-62	23	2	38	63
10.4.0.0-7	3	0	5	8
10.5.0.0-125	72	0	54	126
10.6.0.0-7	1	2	5	8
10.7.0.0-13	10	0	4	14
10.8.0.0-31	7	0	25	32
10.9.0.0-125	2	0	124	126
10.10.0.0-61	14	1	47	62
10.11.0.0-29	8	0	22	30

Total	**146**	**7**	**326**	**479**

Operating System Summary - Detailed Report

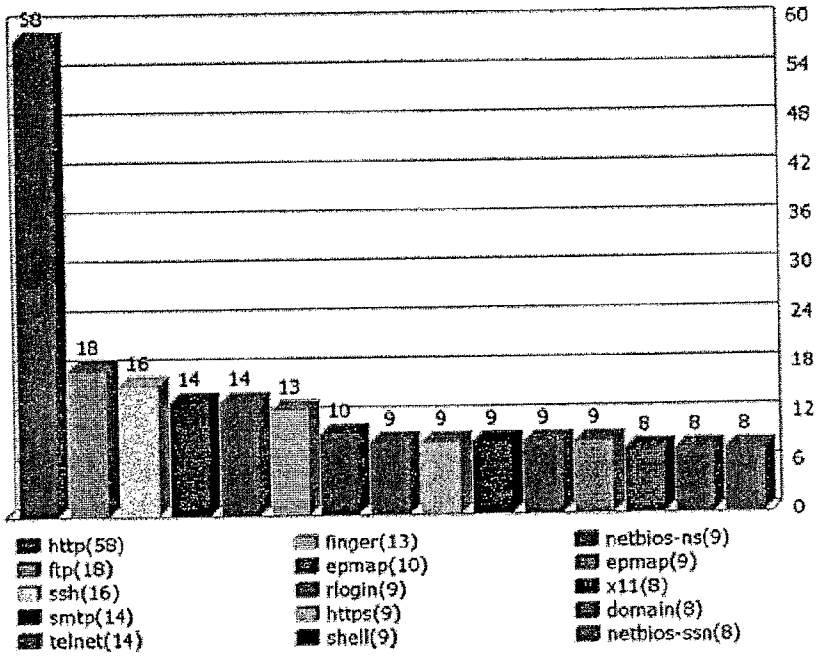


- AIX 4.0 (1)
- AIX 4.3 (1)
- Cisco IOS (4)
- FreeBSD 4.1 - 4.2 (1)
- HP-UX B.11.00 A 9000 (1)
- Linux 2.2.14 - 2.2.16 (6)
- Linux 2.2.16-22 (1)
- Linux 2.3.99-pre9 (2)
- Nokia IPSO 3.2 (2)
- OpenBSD 2.6 - 2.8 (5)
- Solaris 2.5.1 - 2.7 (22)
- SunOS 5.6 - 5.7 (3)
- Windows 2000 - ME (4)
- Windows NT 4.0 (30)
- Windows NT 4.0 - 9x (3)
- Windows NT4 - 9x (3)

Network Services Summary - Detailed Report

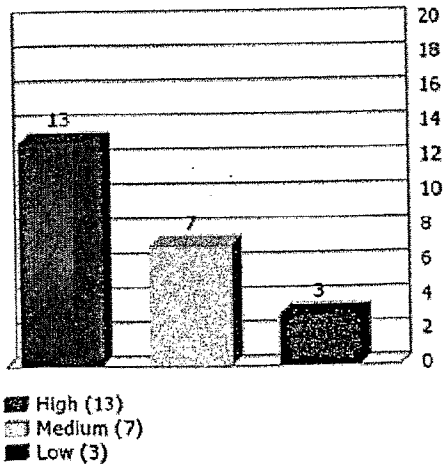
FoundScan for Acme Company: Scan Summary Report

Top 15 Services

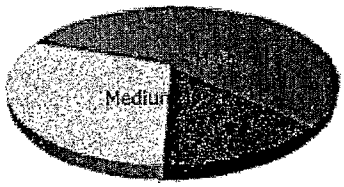


Vulnerabilities Report Summary - Detailed Report

Vulnerabilities By Severity



Percent of Vulnerabilities By Severity



Trend Report Summary - Detailed Report

Summary Comparison	7-19-2001	7-20-2001	Change
FoundScore Risk Rating:	11	6	-5
Total Vulnerabilities:	23	23	0

FoundScan for Acme Company: Scan Summary Report Page 5 of 5

Total Discovered Hosts:	42	146	**104**
Total Network Services:	196	274	**78**
Total Web Servers:	7	58	**51**
Total SSL Web Servers:	5	9	**4**
Total SMTP Servers:	10	14	**4**
Total FTP Servers:	10	18	**8**
Number of Telnet Servers:	14	14	**0**

(c)2000-2001 Foundstone Inc.

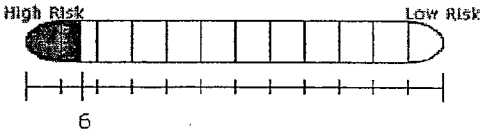
FoundScan for Acme Company: FoundScore Report Page 1 of 3

FoundScan
MANAGED SECURITY SERVICES
KNOW VULNERABILITIES

Report Generated: 10-11-2001 15:10:08 Pacific Standard Time Navigate

Acme Company
Scan Name: Daily Scan

FOUNDSCORE REPORT

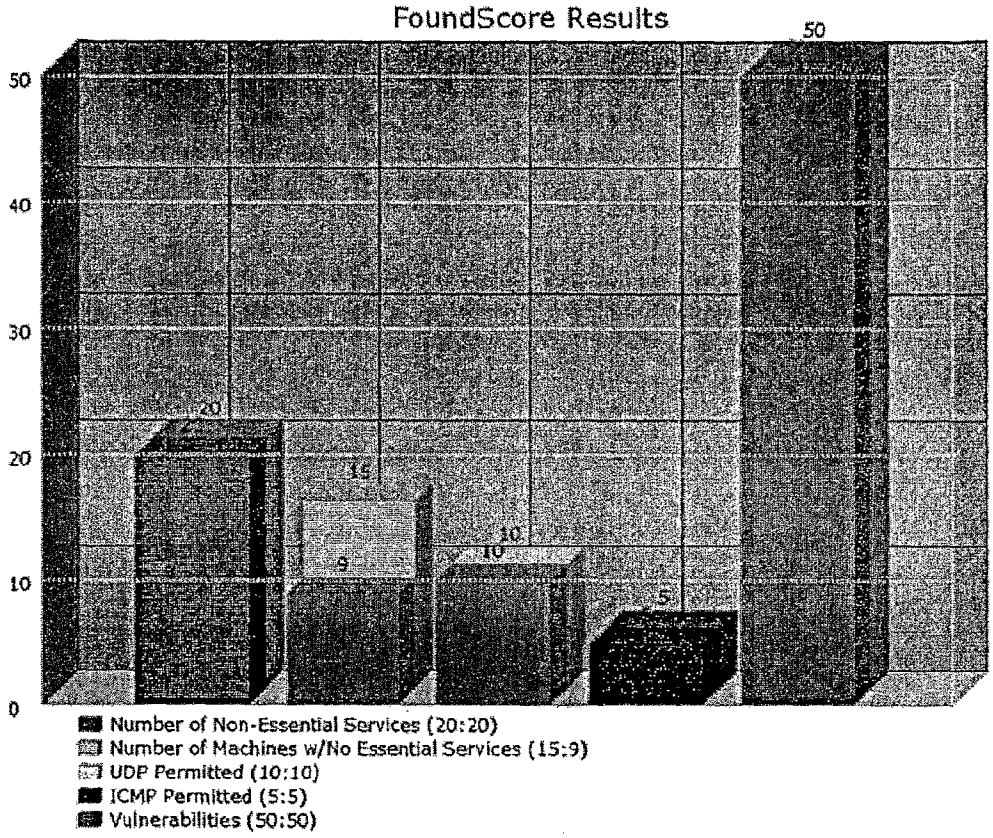


FoundScore Risk Rating

You scored a total of **6** out of a possible 100 points. Your FoundScore places you in the Poor category. Thus, relative to other organizations, your risk rating is poor. Please review the FoundScore Results graph and FoundScore Categories table below for details.

A Full Explanation of the FoundScore scoring methodology is also available.

FoundScan for Acme Company: FoundScore Report Page 2 of 3



FoundScore: Vulnerabilities	Your Results	Deductions	Running Score
High Risk Vulnerabilities	13	50	50
Medium Risk Vulnerabilities	7	0	50
Low Risk Vulnerabilities	3	0	50
	Score after Vulnerability Deductions:		**50**
FoundScore: Exposure	Your Results	Deductions	Running Score
Number of Non-Essential Services	150	20	30
Number of Machines with No Essential Services	9	9	21
UDP Permitted	Yes	10	11
ICMP Permitted	Yes	5	6
		Total Score	**6**
			Poor

To understand what systems and ports affected your FoundScore, click on the specific item of interest in the "Your Results" column in the above table.

FoundScan for Acme Company: FoundScore Report Page 3 of 3

(c)2000-2001 Foundstone Inc.

FoundScan
MANAGED SECURITY SERVICES
KNOW VULNERABILITIES

Report Generated: 10-11-2001 15:10:08 Pacific Standard Time

Navigate

Acme Company

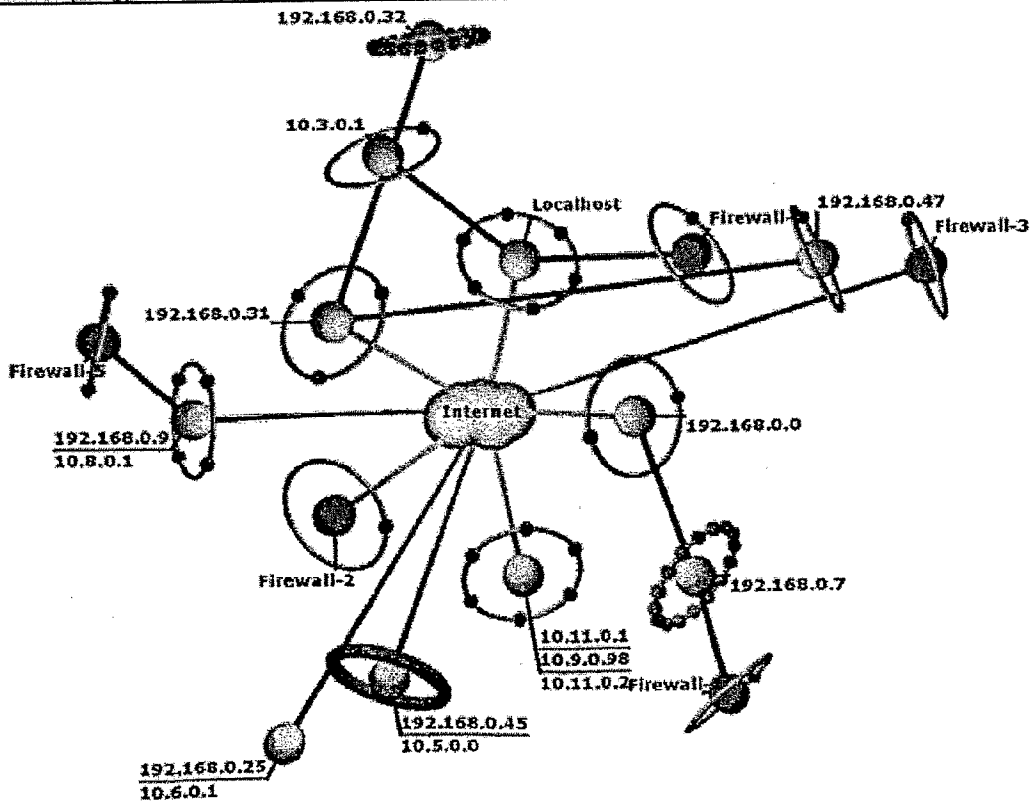
Scan Name: Daily Scan

NETWORK TOPOLOGY REPORT

The network map below is a graphical representation of the scanned environment. FoundScan created it by issuing a series of ICMP and TCP traceroute commands, analyzing the results, then mapping devices based on their subnet membership and distance from one another. The map provides a high-level overview of your environment so that you can easily identify all discovered subnets and devices. A detailed report of discovered hosts is also available.

NOTE: The traceroute procedure does not always return an IP at each hop. This is almost always due to a firewall blocking responses to the traceroute commands, and so we represent unknowns as firewalls on the map. It is possible, however, due to varying load conditions on the network and other factors out of our control that spurious unknowns might creep into the data. Our data gathering methods minimize this, but when it does happen, some machines may be connected to a firewall in the map that are in fact connected to a known router in the network.

Network Topology Summary: Acme Company

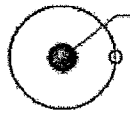


FoundScan for Acme Company: Network Topology Report

Detailed Subnets: Acme Company

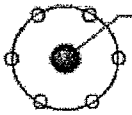
Abc Cisco Freebsd HP

Linux Openbsd Solaris Windows



10.3.0.1
1 Host

10.3.0.30
63.acme.net



10.11.0.1
10.9.0.98
10.11.0.2
6 Hosts

10.9.0.1
[Unknown]

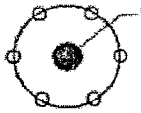
10.11.0.8
[Unknown]

10.11.0.9
[Unknown]

10.11.0.10
nathanbullfrog.acm...

10.11.0.13
wart.acme.com

10.11.0.19
[Unknown]



Localhost
6 Hosts

10.3.0.2
35.acme.net

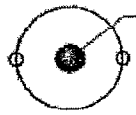
10.3.0.5
38.acme.net

10.3.0.7
40.acme.net

10.3.0.8
41.acme.net

10.3.0.12
45.acme.net

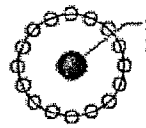
10.4.0.2
[Unknown]



192.168.0.0
2 Hosts

10.2.0.1
1.acme.net

10.3.0.32
gw.acme.com



192.168.0.7
16 Hosts

10.2.0.2
2.acme.net

10.2.0.4
4.acme.net

10.2.0.5
5.acme.net

10.3.0.33
fw.acme.com

10.3.0.34
ns1.acme.com

10.3.0.37
topaz.acme.com

10.3.0.39
feldspar.acme.com

10.3.0.40
diamond.acme.com

10.3.0.42
calcite.acme.com

10.3.0.43
gypsum.acme.com

10.3.0.45
granite.acme.com

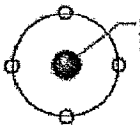
10.3.0.50
apatite.acme.com

10.3.0.51
beryl.acme.com

10.3.0.52
pearl.acme.com

10.3.0.53
ruby.acme.com

10.3.0.54
sapphire.acme.com



192.168.0.9
10.8.0.1
4 Hosts

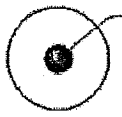
10.8.0.13
acme.com

10.8.0.14
[Unknown]

10.8.0.18
[Unknown]

10.8.0.25
[Unknown]

(c)2000-2001 Foundstone Inc.



192.168.0.25
10.6.0.1
0 Hosts

192.168.0.25

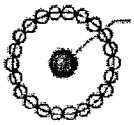


192.168.0.31
3 Hosts

10.7.0.0
bellcoqv.acme.com

10.10.0.0
[Unknown]

10.11.0.0
[Unknown]



192.168.0.32
22 Hosts

10.7.0.1
bellco2.acme.org

10.7.0.2
bellco3.acme.org

10.7.0.3
bellco4.acme.org

10.7.0.5
bellco6.acme.org

10.7.0.6
bellco7.acme.org

10.7.0.7
bellco8.acme.org

10.7.0.8
bellco9.acme.org

10.7.0.9
bellco10.acme.org

10.7.0.12
bellco13.acme.org

10.10.0.7
[Unknown]

10.10.0.8
[Unknown]

10.10.0.9
[Unknown]

10.10.0.10
[Unknown]

10.10.0.11
[Unknown]

10.10.0.12
[Unknown]

10.10.0.13
[Unknown]

10.10.0.14
[Unknown]

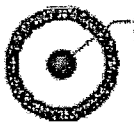
10.10.0.15
[Unknown]

10.10.0.16
[Unknown]

10.10.0.17
[Unknown]

10.10.0.18
[Unknown]

10.10.0.61
[Unknown]



192.168.0.45
10.5.0.0
71 Hosts

10.5.0.1
[Unknown]

10.5.0.5
[Unknown]

10.5.0.6
[Unknown]

10.5.0.7
[Unknown]

10.5.0.8
[Unknown]

10.5.0.9
[Unknown]

10.5.0.10
[Unknown]

10.5.0.11
[Unknown]

10.5.0.12
[Unknown]

10.5.0.13
[Unknown]

10.5.0.14
[Unknown]

10.5.0.15
[Unknown]

10.5.0.16
[Unknown]

10.5.0.17
[Unknown]

10.5.0.18
[Unknown]

10.5.0.19
[Unknown]

10.5.0.20
[Unknown]

10.5.0.21
[Unknown]

10.5.0.22
[Unknown]

10.5.0.23
[Unknown]

10.5.0.24
[Unknown]

10.5.0.25
[Unknown]

10.5.0.26
[Unknown]

10.5.0.27
[Unknown]

10.5.0.28
[Unknown]

10.5.0.29
[Unknown]

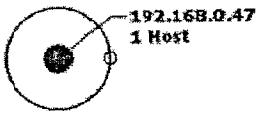
10.5.0.30
[Unknown]

10.5.0.31
[Unknown]

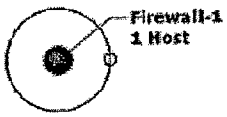
10.5.0.32
[Unknown]

10.5.0.33
[Unknown]

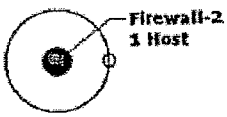
10.5.0.34 [Unknown]
10.5.0.35 [Unknown]
10.5.0.36 [Unknown]
10.5.0.37 [Unknown]
10.5.0.38 [Unknown]
10.5.0.39 [Unknown]
10.5.0.40 [Unknown]
10.5.0.41 [Unknown]
10.5.0.42 [Unknown]
10.5.0.43 [Unknown]
10.5.0.44 [Unknown]
10.5.0.45 [Unknown]
10.5.0.46 [Unknown]
10.5.0.47 [Unknown]
10.5.0.48 [Unknown]
10.5.0.49 [Unknown]
10.5.0.50 [Unknown]
10.5.0.51 [Unknown]
10.5.0.52 [Unknown]
10.5.0.53 [Unknown]
10.5.0.54 [Unknown]
10.5.0.55 [Unknown]
10.5.0.56 [Unknown]
10.5.0.57 [Unknown]
10.5.0.58 [Unknown]
10.5.0.59 [Unknown]
10.5.0.60 [Unknown]
10.5.0.61 [Unknown]
10.5.0.62 [Unknown]
10.5.0.63 [Unknown]
10.5.0.64 [Unknown]
10.5.0.65 [Unknown]
10.5.0.66 [Unknown]
10.5.0.67 [Unknown]
10.5.0.68 [Unknown]
10.5.0.69 [Unknown]
10.5.0.94 [Unknown]
10.5.0.109 [Unknown]
10.5.0.111 [Unknown]
10.5.0.123 [Unknown]
10.5.0.124 [Unknown]



10.11.0.2
fropen.acme.com



10.2.0.6
6.acme.net



10.4.0.3
[Unknown]

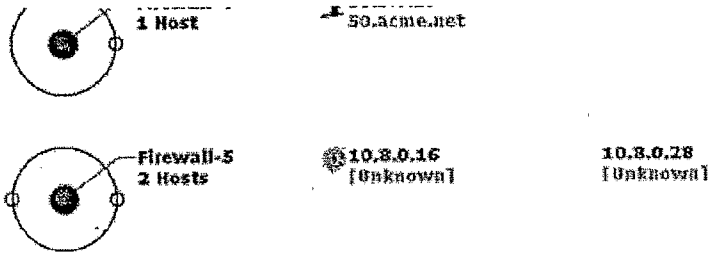


10.1.0.0
[Unknown]



10.3.0.17

FoundScan for Acme Company: Network Topology Report



WHAT IS CLAIMED IS:

1. A system for determining an operating system of a target computer operably connected to a network, the system comprising:

first and second data packets, said first and second data packets compliant with a protocol supported by said network, said first and second data packets transmitted via said network to said target computer;

first and second operating system fingerprints comprising data bits stored in a computer-readable medium, said first and second operating system fingerprints associated with a first operating system;

a first target computer fingerprint comprising data bits stored in a computer-readable medium, said first target computer fingerprint including a representation of at least a portion of data received in response to said transmission of said first data packet;

a second target computer fingerprint comprising data bits stored in a computer-readable medium, said second target computer fingerprint including a representation of at least a portion of data received in response to said transmission of said second data packet; and

fingerprint comparison instructions executable by a computer to compare said first operating system fingerprint and said first target computer fingerprint, to compare said second operating system fingerprint and said second target computer fingerprint, and to generate a result indicative of whether said first operating system was running on said target computer.

2. The system as described in Claim 1, wherein a first range of bits of said first data packet represents a first parameter value, and wherein said first range of bits of said second data packet represents a second parameter value different from said first parameter value.

3. The system as described in Claim 2, wherein said second parameter value is derived by changing one bit in said first range of bits of said first data packet.

4. The system as described in Claim 2, wherein said first and second operating system fingerprints differ.

5. The system as described in Claim 4, further comprising:

a third data packet, said third data packet compliant with said protocol, said first range of bits of said third data packet representing a third parameter value different from said first and second parameter values, said third data packet transmitted via said network to said target computer;

a third operating system fingerprint comprising data bits stored in a computer-readable medium, said third operating system fingerprint associated with said first operating system, said third operating system fingerprint differing from said first and second operating system fingerprints; and

a third target computer fingerprint comprising data bits stored in a computer-readable medium, said third target computer fingerprint including a representation of at least a portion of data received in response to said transmission of said first data packet, said comparison instructions executable by a computer to compare said third operating system fingerprint and said third target computer fingerprint before generating said result.

6. The system as described in Claim 5, further comprising:

fourth, fifth and sixth operating system fingerprints comprising data bits stored in a computer-readable medium, said fourth, fifth and sixth operating system fingerprints associated with a second operating system, at least one of said fourth, fifth and sixth operating system fingerprints differing from a respective one of said first, second and third operating system fingerprints; said comparison instructions executable by a computer to compare said fourth operating system fingerprint and said first target computer fingerprint, to compare said fifth operating system fingerprint and said second target computer fingerprint, to compare said sixth operating system fingerprint and said third target computer fingerprint, and to generate a second result indicative of whether said second operating system was running on said target computer.

7. The system as described in Claim 5, wherein said protocol is TCP/IP and wherein said first range of bits corresponds to a packet field representing a maximum segment size.

8. The system as described in Claim 5, wherein said first parameter value is obtained by setting no bits, said second parameter value is obtained by setting one bit, and said third parameter value is obtained by setting two bits.

9. The system as described in Claim 5, wherein said first parameter value is 0, said second parameter value is 128, and said third parameter value is 128 plus a multiple of 256.

10. The system as described in Claim 5, wherein said first range of bits represents at least two bytes, and wherein a value of said second parameter is obtained by setting the last bit in a byte, and a value for said third parameter is obtained by setting the last bit in a byte.

11. The system as described in Claim 10, wherein said third parameter is obtained by setting adjacent bits in said first range of bits.

12. The system as described in Claim 5, wherein said first, second and third data packets are transmitted in order of lowest parameter value first.

13. A system for determining an operating system of a target computer accessible via a network, the system comprising:

a plurality of data packets compliant with a protocol supported by said network, said plurality of data packets transmitted via said network to said target computer;

a first plurality of operating system fingerprints, each comprising data bits stored in a computer-readable medium, each associated with a first operating system;

a plurality of target computer fingerprints, each comprising data bits stored in a computer-readable medium, each including a representation of at least a portion of data received in response to said transmission of said plurality of data packets; and

fingerprint comparison instructions executable by a computer to compare said first plurality of said operating system fingerprint and said plurality of said target computer fingerprints, and to generate a result indicative of whether said first operating system was running on said target computer.

14. The system as described in Claim 13, wherein said protocol is TCP/IP and wherein each of said plurality of data packets has a different value represented in a respective packet field.

15. The system as described in Claim 14, wherein said packet field is a maximum segment size field.

16. The system as described in Claim 13, further comprising:

a second plurality of operating system fingerprints, each comprising data bits stored in a computer-readable medium, each associated with a second operating system, said fingerprint comparison instructions comparing said second plurality of said operating system fingerprints and said plurality of said target computer fingerprints to generate a second result indicative of whether said second operating system was running on said target computer.

17. A method for determining an operating system of a target computer accessible via a network, the method comprising the steps of:

transmitting to said target computer a plurality of data packets compliant with a protocol supported by said network;

generating a plurality of target computer fingerprints, each including at least a portion of data received via said network in response to said transmission of said plurality of data packets;

comparing said plurality of target computer fingerprints to a first set of predetermined operating system fingerprints, each of said first set of predetermined operating system fingerprints associated with a first operating system; and

generating a result indicative of whether said first operating system was running on said target computer.

18. The method as described in Claim 17, comprising the further steps of:

comparing said plurality of target computer fingerprints to a second set of predetermined operating system fingerprints, each of said second set of predetermined operating system fingerprints associated with a second operating system; and

generating a result indicative of whether said second operating system was running on said target computer.

19. The method as described in Claim 17, wherein said protocol is TCP/IP and wherein some of said plurality of data packets have different values in the same packet field.

20. The method as described in Claim 17, wherein said protocol is TCP/IP and wherein the value of the MSS option of two of said plurality of data packets is divisible by 128.

21. The method as described in Claim 17, wherein a first of said plurality of data packets has a maximum segment size option of 0, wherein a second of said plurality of data packets has a maximum segment size option of 128, and wherein a third of said plurality of data packets has a maximum segment size option of 384.

22. A method for identifying an operating system of a target computer via a network, the method comprising the steps of:

sending a first data packet to said target computer via said network, said first data packet complying with a protocol of said network and having a first pattern of bits in a first range of bits;

generating a first response value representing at least a portion of data received via said network in response to said sending of said first data packet;

sending a second data packet to said target computer via said network, said second data packet complying with said protocol and having a second pattern of bits in a first range of bits, said second pattern of bits different from said first pattern;

generating a second response value representing at least a portion of data received via said network in response to said sending of said second data packet;

sending a third data packet to said target computer via said network, said third data packet complying with said protocol and having a third pattern of bits in a first range of bits, said third pattern of bits different from said first or said second pattern;

generating a third response value representing at least a portion of data received via said network in response to said sending of said third data packet;

comparing said first response value to a first predetermined value associated with a first operating system;

comparing said second response value to a second predetermined value associated with said first operating system;

comparing said third response value to a third predetermined value associated with said first operating system; and

generating a value indicative of a relationship between said first operating system and said target computer.

23. The method as described in Claim 22, the method comprising the further steps of:

comparing said first response value to a fourth predetermined value associated with a second operating system;

comparing said second response value to a fifth predetermined value associated with said second operating system; and

comparing said third response value to a sixth predetermined value associated with said second operating system.

24. The method as described in Claim 22, wherein no bit is set in said first pattern of bits, wherein one bit is set in said second pattern of bits, and wherein two bits are set in said third pattern of bits.

25. The method as described in Claim 22, wherein the number of bytes in said second pattern of bits that have at least one bit set is greater than the number of bytes in said first pattern of bits that have at least one bit set, and wherein the number of bytes in said third pattern of bits that have at least one bit set is greater than the number of bytes in said second pattern of bits that have at least one bit set.

26. The method as described in Claim 22, wherein no byte in said first pattern of bits has a least significant bit or a most significant bit that is set, wherein at least one byte in said second pattern of bits has a most significant bit that is set, and wherein at least one byte in said third pattern of bits has a least significant bit that is set.

27. A system for determining whether a target computer is on a network, the system comprising:

a first set of port identifiers stored in a computer-readable medium, each of said first set of port identifiers representing a port used by computers to receive data packets compliant with a first protocol of said network, each of said first set of port identifiers representing a port associated with known network services;

a first set of data packets, each directed to a port represented by at least one of said first set of port identifiers, each of said first set of data packets compliant with said first protocol and transmitted to said target computer via said network;

a first set of acknowledgement packets received via said network in response to said transmission of said first set of data packets; and

a list of host identifiers, each host identifier representing a computer on said network that transmits data in response to a packet sent to said respective computer, a host identifier representing said target computer added to said list of host identifiers if said first set of acknowledgment packets indicates a responsiveness of said target computer.

28. The system as described in Claim 27, the system further comprising:

a second set of port identifiers stored in a computer-readable medium, each of said second set of port identifiers representing a port used by computers to receive data packets compliant with a second protocol of said network, each of said second set of port identifiers representing a port associated with known network services;

a second set of data packets, each directed to a port represented by at least one of said second set of port identifiers, each of said second set of data packets compliant with said second protocol and transmitted to said target computer via said network, at least one of said second set of data packets including data associated with said known network services;

a second set of acknowledgement packets received via said network in response to said transmission of said second set of data packets; and

a host identifier representing said target computer added to said list of host identifiers if said second set of acknowledgment packets indicates a responsiveness of said target computer.

29. The system as described in Claim 28, wherein said first protocol is TCP, wherein said second protocol is UDP, wherein said second set of acknowledgment packets is a nonzero set of UDP data response packets.

30. The system as described in Claim 27, the system further comprising:

a second set of port identifiers stored in a computer-readable medium, each of said second set of port identifiers representing a port used by computers to receive data packets compliant with a second protocol of said network, each of said second set of port identifiers representing a port associated with known network services;

a second set of data packets, each directed to a port represented by at least one of said second set of port identifiers, each of said second set of data packets compliant with said second protocol and transmitted to said target computer via said network, at least one of said second set of data packets including data associated with said known network services;

a second set of acknowledgement packets received via said network in response to said transmission of said second set of data packets; and

a host identifier representing said target computer added to a second list of host identifiers if said second set of acknowledgment packets does not indicate an unresponsiveness of said target computer, each of said second list of host identifiers representing a computer not known to be unresponsive.

31. The system as described in Claim 30, wherein said first protocol is TCP, wherein said second protocol is UDP, wherein said second set of acknowledgment packets is an empty set of ICMP error packets.

32. The system as described in Claim 30, the system further comprising:

a third set of data packets, each directed to a port represented by at least one of said second set of port identifiers, each compliant with said second

protocol, said third set of data packets transmitted to said target computer throughout a predetermined maximum latency period;

a first response received first in time in response to said transmission of said third set of data packets; and

a second response received second in time in response to said transmission of said third set of data packets, a time duration between said receipt of said first response and said receipt of said second response defining a target computer latency period.

33. The system as described in Claim 32, wherein each of said second set of data packets is transmitted continuously to said target computer for the duration of said target computer latency period.

34. The system as described in Claim 28, the system further comprising:

a third set of data packets, each directed to a port represented by at least one of said second set of port identifiers, each compliant with said second protocol, said third set of data packets transmitted to said target computer throughout a predetermined maximum latency period;

a first response received first in time in response to said transmission of said third set of data packets; and

a second response received second in time in response to said transmission of said third set of data packets, a time duration between said receipt of said first response and said receipt of said second response defining a target computer latency period.

35. The system as described in Claim 34, wherein each of said second set of data packets is transmitted continuously to said target computer for the duration of said target computer latency period.

36. A system for testing the accessibility of a target computer via a network, the system comprising:

a set of port identifiers stored in a computer-readable medium, each of said set of port identifiers representing a UDP-compliant port, at least one of said port identifiers representing a port associated with known network services;

a set of UDP-compliant data packets, each associated with a port represented by at least one of said set of port identifiers, each of said UDP-compliant data packets transmitted continuously to said target computer for a duration approximately the same as the latency period of said target computer, at least one of said UDP-compliant data packets including data associated with said known network services;

a first list representing computers accessible via said network, said first list including said target computer if a nonzero set of UDP data response packets is received in response to said transmission of said data packets; and

a second list representing computers not known to be inaccessible via said network, said second list including said target computer if an empty set of ICMP error packets is received in response to said transmission of said data packets.

37. A method for determining whether a target computer is accessible via a network, the method comprising the steps of:

identifying TCP ports;

sending first data packets to said TCP ports of said target computer, each of said first data packets compliant with TCP;

receiving first acknowledgment packets in response to said sending of said first data packets; and

adding a representation of said target computer to a list representing accessible computers if said first acknowledgment packets are nonzero.

38. The method as described in Claim 37, the method comprising the further steps of:

identifying UDP ports associated with network services;

sending second data packets to said UDP ports of said target computer, at least one of said second data packets sent continuously to said target computer throughout a latency period of said target computer;

receiving second acknowledgment packets in response to said sending of said second data packets; and

adding a representation of said target computer to a list representing accessible computers if said second acknowledgment packets are nonzero UDP data response packets.

39. The method as described in Claim 38, the method comprising the further step of:

determining said latency period of said target computer by measuring the time between responses received in response to packets transmitted to said target computer.

40. The method as described in Claim 38, the method comprising the further step of:

adding a representation of said target computer to a list representing computers not known to be inaccessible via said network, said adding performed if said second acknowledgment packets comprise an empty set of ICMP error packets.

41. A method for assessing the vulnerability of a target computer via a network, the method comprising the steps of:

discovering a set of responsive computers on a network by transmitting a set of ICMP packets, a set of TCP packets and a set of UDP packets to a group of computers on a network;

detecting services on each of said set of responsive computers by transmitting TCP packets to first ports of each of said set of responsive computers and by transmitting UDP packets to second ports of each of said set of responsive computers, said first and second ports commonly used by computers to receive data packets over a network, said TCP packets including data associated with at least one computer-based service known to use one of said first ports, said UDP packets including data associated with at least one computer-based service known to use one of said second ports; and

generating a list of responsive ports using responses received in response to said transmission of said TCP packets and said UDP packets.

42. The method described in Claim 41, the method comprising the further step of:

determining an operating system used by each of said set of responsive computers by comparing predetermined values with portions of responses received from each of said set of responsive computers in response to transmission of a plurality of TCP-compliant packets to each of said set of responsive computers.

43. The method described in Claim 42, the method comprising the further step of:

confirming the presence of vulnerabilities on said network by applying an automated vulnerability script to each responsive port represented in said list of responsive ports, each of said automated vulnerability scripts testing a vulnerability known to be associated with a computer configuration comprising a particular responsive port and a particular operating system.

44. The method described in Claim 43, the method comprising the further step of:

calculating an objective indicia of security of said network, said calculation based on a weighted summation of confirmed vulnerabilities.

45. The method described in Claim 44, the method comprising the further step of:

determining a topology of said network, said topology determination made by transmitting a set of ICMP packets with varying time to live (TTL) settings and by transmitting a set of TCP packets with varying TTL settings.

46. The method described in Claim 45, the method comprising the further step of:

producing a graphical representation of said network, said representation including a topological map of said network, a color-based representation of weighted confirmed vulnerabilities, and an association between said graphical representation and information descriptive of confirmed vulnerabilities and computers on said network.

47. A method of creating a graphical representation of a network, the method comprising the steps of:

obtaining IP addresses of nodes on a network;

obtaining node distance and connectivity relationships between said nodes;

identifying some nodes as routers;

identifying other nodes as leaf nodes connected to one of said routers;

generating graphical representations of router nodes;

for each router, generating graphical representations of directly connected leaf nodes by depicting graphical representations of said directly connected leaf nodes having a spatial relationship to said graphical representation of said respective router; and

depicting links between routers having no intervening routers.

48. The method described in Claim 47, comprising the further step of:

comparing, for each router, the IP address of said respective router and the IP address of each of said directly connected leaf nodes to resolve instances where a directly connected leaf node and said respective router represent two network connections of the same node.

49. A method for creating a topological representation of a network, said method comprising the steps of:

identifying responsive computers on said network;

obtaining a plurality of sequences of IP addresses by sending to each responsive computer a sequence of packets having increasing time to live (TTL) values, each sequence of IP addresses representing nodes in said network between a source computer and one of said responsive computers, adjacent IP addresses in each sequence representing connected nodes, each of said nodes comprising a computer or a router;

generating a list of node structures, each of said node structures including data representing a node and data indicative of other nodes to which it directly connects, said list representing all IP addresses in said plurality of sequences;

determining for each IP address a distance count, said distance count representing a number of nodes between a node having said IP address and a source node;

creating a router structure for each node structure that represents a node comprising a router;

associating with each of said router structures connection data representative of each connecting node that connects to no other node except the router represented by said respective router structure;

for each router structure, visually depicting a graphical shape spatially related to one or more graphical shapes corresponding to connecting nodes represented by said connection data of said respective router structure; and

for each router structure, visually depicting a connection between a graphical shape associated with the respective router structure and another graphical shape associated with a different router structure when distance counts associated with the IP addresses of routers represented by said respective router structure and said different router structure indicate a direct connection.

50. The method described in Claim 49, said method comprising the further step of:

testing whether a router represented by a router structure and a connecting node represented in connection data comprise two network connections of one node.

51. The method described in Claim 49, wherein the graphical shape representing a router is a sphere, and wherein each of said spatially related graphical shapes is a sphere orbiting said sphere representing said router.

52. A method for calculating an objective vulnerability score, said method comprising the steps of:

identifying known vulnerabilities of a network;

weighting said known vulnerabilities based on either ease of exploitation or level of access granted; and

determining a vulnerability value numerically representing a combination of weighted known vulnerabilities of a network.

53. A method for calculating an objective security score for a network, said method comprising the steps of:

determining a vulnerability value numerically representing a combination of known vulnerabilities of a network;

determining an exposure value numerically representing a combination of accessible ports of computers on said network; and

deriving a score by combining said vulnerability value and said exposure value.

54. The method described in Claim 53, wherein said combination of known vulnerabilities is a summation of weighted numeric expressions of particular vulnerabilities, the weighting based on an ease of exploitation ranking and on an access granted ranking for each vulnerability.

55. A method for conducting an automated network vulnerability attack, said method comprising the steps of:

selecting a set of vulnerability attacks for each responsive computer on a network, each selected vulnerability attack for each responsive computer designed to expose a vulnerability associated with ports of said respective computer known to be accessible and also associated with an operating system used by said respective computer;

encoding said set of vulnerability attacks such that each is represented in a database by a unique identifier;

representing each of said set of vulnerability attacks using instructions of an automated scripting language; and

executing said vulnerability attacks by processing said instructions with a computer.

56. A hierarchical network vulnerability report, comprising:

a first report level comprising:

an objective score representing the security of said network; and

a graphical representation of a network topology, including a graphical representation of computers accessible via said network and a color-based graphical representation of the vulnerability of at least some of said computers;

and

a second report level comprising:

a textual list describing said computers and their associated vulnerabilities; and

an exposure report describing accessible ports and services of said computers.

57. A vulnerability assessment language comprising:

a set of programming language statements used to create executable scripts, said scripts executed in a thread-safe execution architecture wherein all variables are stack variables and wherein a parse tree is treated as a read-only data structure;

a set of special scalar data types interchangeable with an integer data type in expressions, each of said set of special scalar data types having a set of constant values configured to support vulnerability assessment operations embodied in scripts;

a set of native objects declared in a metascope owning a script scope to make available said native objects to executable scripts, said native objects facilitating network communication, providing callable member functions for building lists of unique ports and directing script execution to certain hosts, and providing IP addresses for scripts; and

a vulnerability object behaving to copy itself into a global data area where other scripts may access its information to compromise another machine, facilitating the use by one script of vulnerability data discovered by a different script.

58. A method for automated application of a known vulnerability on a target computer, the method comprising the steps of:

providing a database of known vulnerabilities, the database including a data object;

providing an executable script, the executable script associated with the data object;

applying the executable script to the target computer, the script performing the known vulnerability on a port of the target computer; and

returning a value representing at least one of the success, failure or other outcome of the executable script.

59. A method for automated application of known vulnerabilities to target computers of a network, the method comprising the steps of:

providing a database of known vulnerabilities;

providing a set of executable scripts, each executable to apply a known vulnerability to a specified target computer;

executing first executable scripts to apply vulnerabilities on specified target computers;

monitoring return values representing a success, failure or other outcome of each of said first executable scripts; and

generating a report using said return values, said report representing a security level of said network.

60. The method described in Claim 59, comprising the further step of:

identifying execution time intervals wherein execution of said first executable scripts commences at the beginning of each of said time intervals and pauses at the end of each of said time intervals, until all of said first executable scripts have executed.

61. The method described in Claim 60, comprising the further step of:

automatically repeating said execution of said first executable scripts when said execution of said first executable scripts is completed.

62. The method described in Claim 61, comprising the further steps of:

generating a report upon each completed execution of said first executable scripts; and

calculating a security trend for said network by comparing a plurality of said reports.

63. The method described in Claim 59, comprising the further step of:

executing second executable scripts to apply vulnerabilities to a second network of computers during said execution of said first executable scripts.

64. The method described in Claim 63, wherein said second network is a subset of said network.

65. The method described in Claim 59, wherein said first executable scripts are configured to apply vulnerabilities to a first port of all of said target computers before applying vulnerabilities to a second port of all of said target computers.

66. The method described in Claim 57, the method comprising the further step of:

allocating a plurality of packet slots, each packet slot permitting asynchronous transmission of a packet by one of said executable scripts.

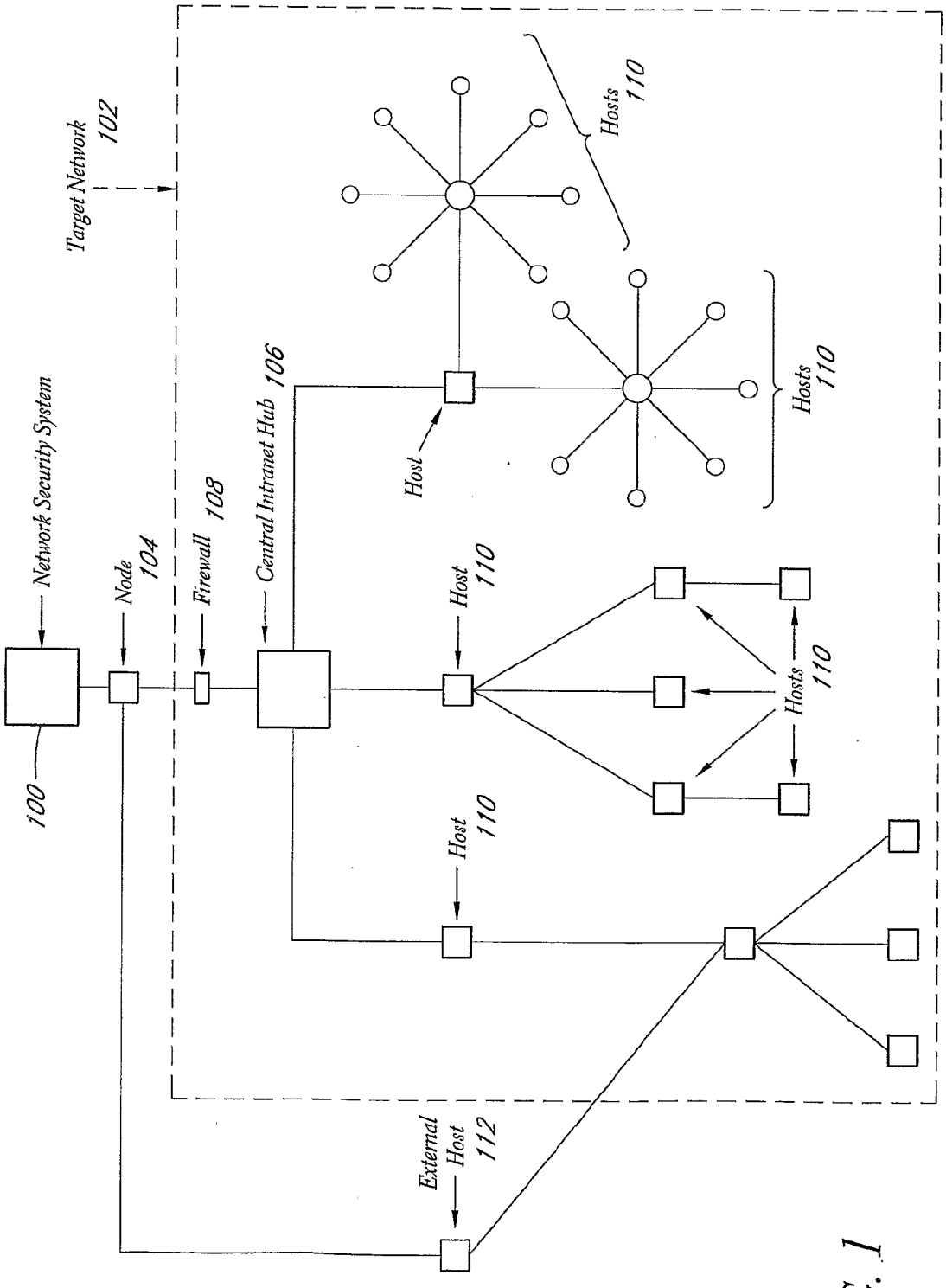

Target Network
102
Hosts
110
Network Security System
Node
104
Firewall
108
Central Intranet Hub
106
Host
110
Hosts
110
Host
110
Hosts
110
Host
110
100
External
Host
112

FIG. 1

FIG. 2

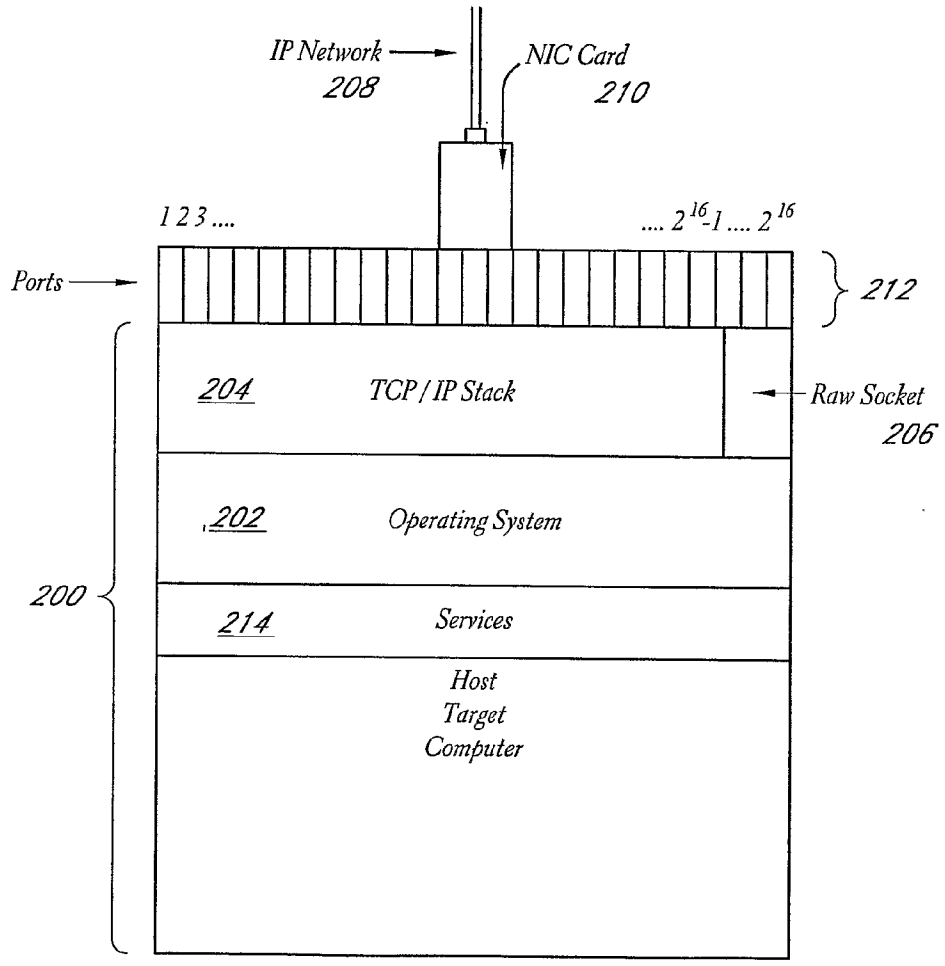
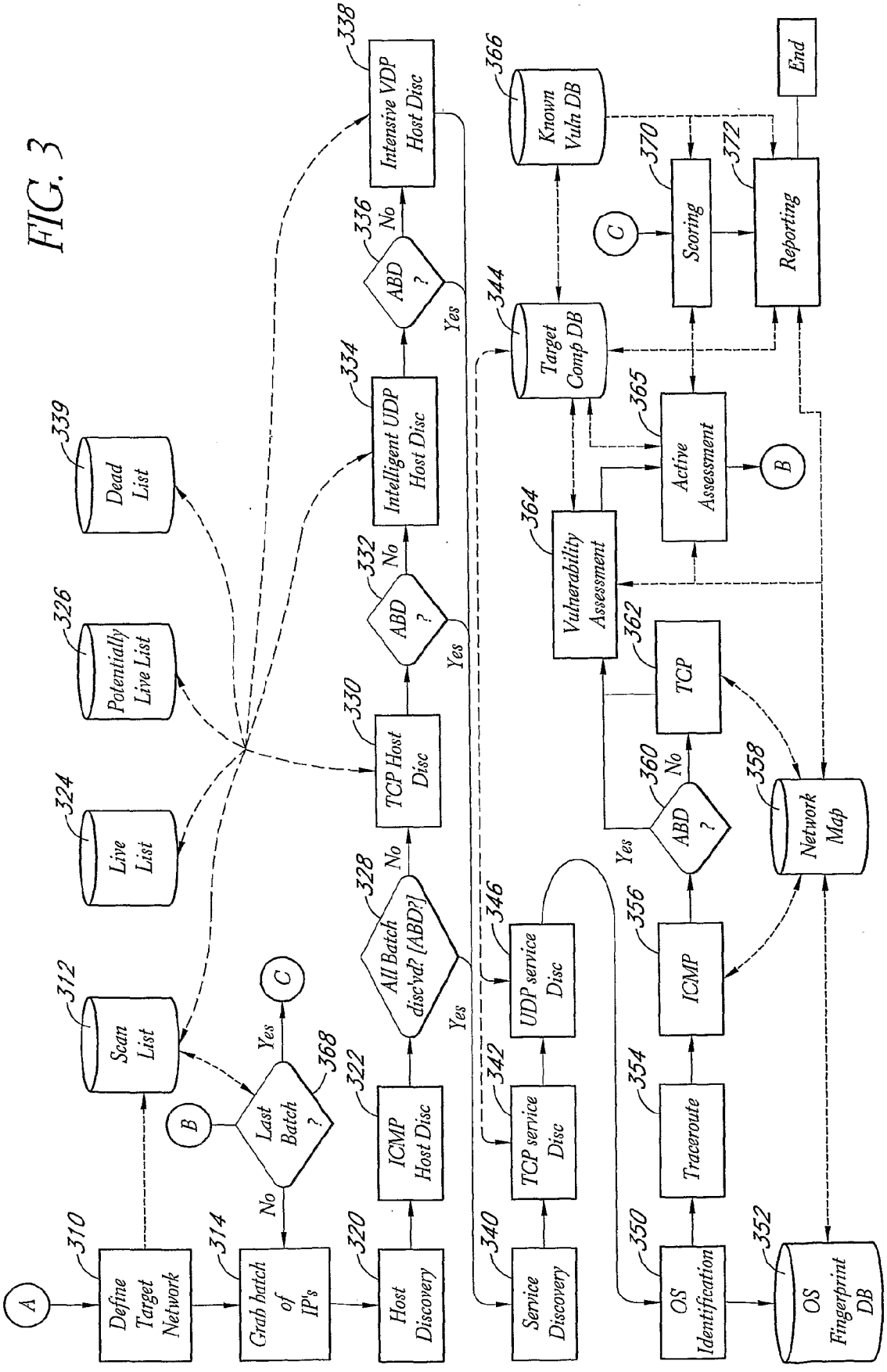

IP Network
208
NIC Card
210
1 2 3 ....
.... 2^16-1 .... 2^16
Ports
212
204
TCP / IP Stack
Raw Socket
206
202
Operating System
200
214
Services
Host
Target
Computer

FIG. 3


A
310
Define Target Network
312
Scan List
314
Grab batch of IP's
320
Host Discovery
322
ICMP Host Disc
324
Live List
326
Potentially Live List
328
All Batch disc'vd? [ABD?]
No
Yes
330
TCP Host Disc
332
ABD ?
No
Yes
334
Intelligent UDP Host Disc
336
ABD ?
No
Yes
338
Intensive VDP Host Disc
339
Dead List
340
Service Discovery
342
TCP service Disc
346
UDP service Disc
B
Last Batch ?
No
Yes
C
368
350
OS Identification
352
OS Fingerprint DB
354
Traceroute
356
ICMP
358
Network Map
360
ABD ?
No
Yes
362
TCP
364
Vulnerability Assessment
365
Active Assessment
B
344
Target Comp DB
366
Known Vuln DB
C
370
Scoring
372
Reporting
End

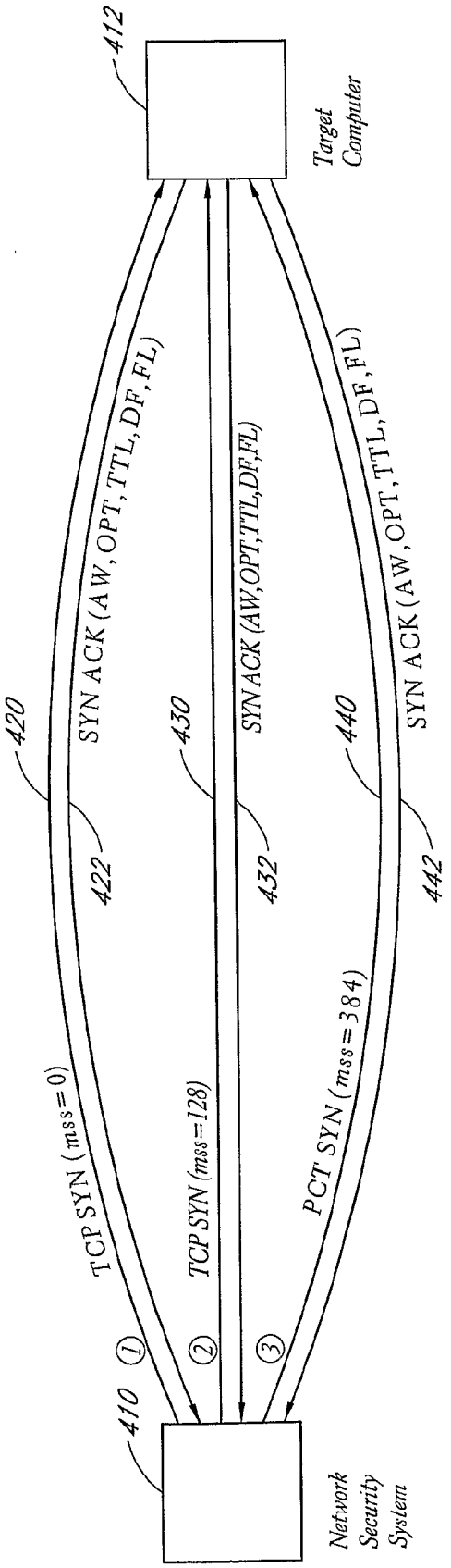
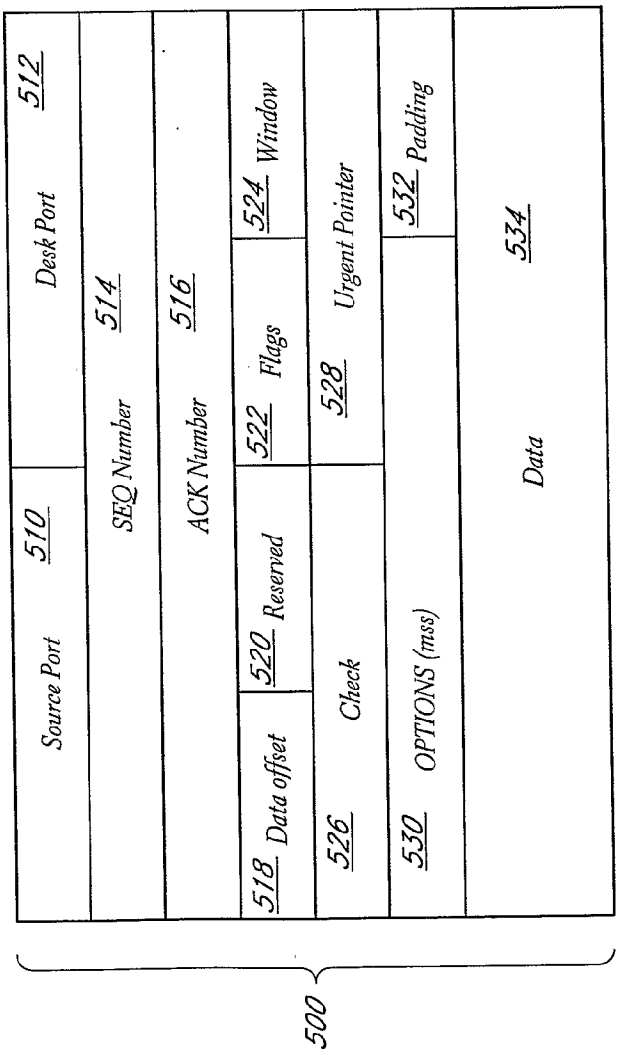

412
Target Computer
420
SYN ACK (AW, OPT, TTL, DF, FL)
422
430
SYN ACK (AW, OPT, TTL, DF, FL)
432
440
SYN ACK (AW, OPT, TTL, DF, FL)
442
TCP SYN (mss=0)
TCP SYN (mss=128)
PCT SYN (mss=384)
1
2
3
410
Network Security System

FIG. 4

FIG. 5


Source Port 510
Desk Port 512
SEQ Number 514
ACK Number 516
518 Data offset
520 Reserved
522 Flags
524 Window
526 Check
528 Urgent Pointer
530 OPTIONS (mss)
532 Padding
Data 534
500

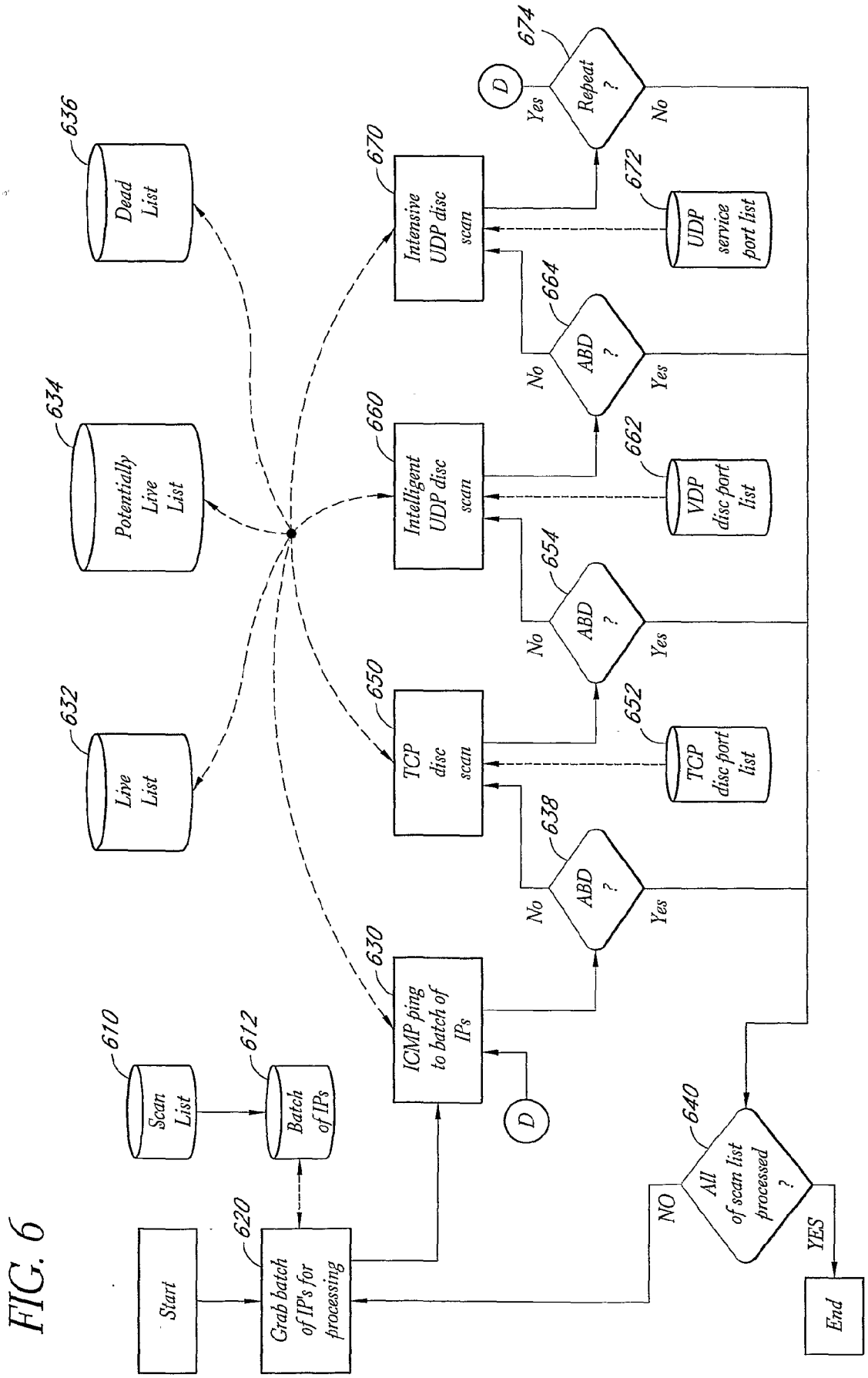

Start
620
Grab batch of IP's for processing
610
Scan List
612
Batch of IPs
630
ICMP ping to batch of IPs
D
638
ABD ?
No
Yes
650
TCP disc scan
652
TCP disc port list
654
ABD ?
No
Yes
660
Intelligent UDP disc scan
662
VDP disc port list
664
ABD ?
No
Yes
670
Intensive UDP disc scan
672
UDP service port list
D
674
Repeat ?
Yes
No
632
Live List
634
Potentially Live List
636
Dead List
640
All of scan list processed ?
NO
YES
End

FIG. 6

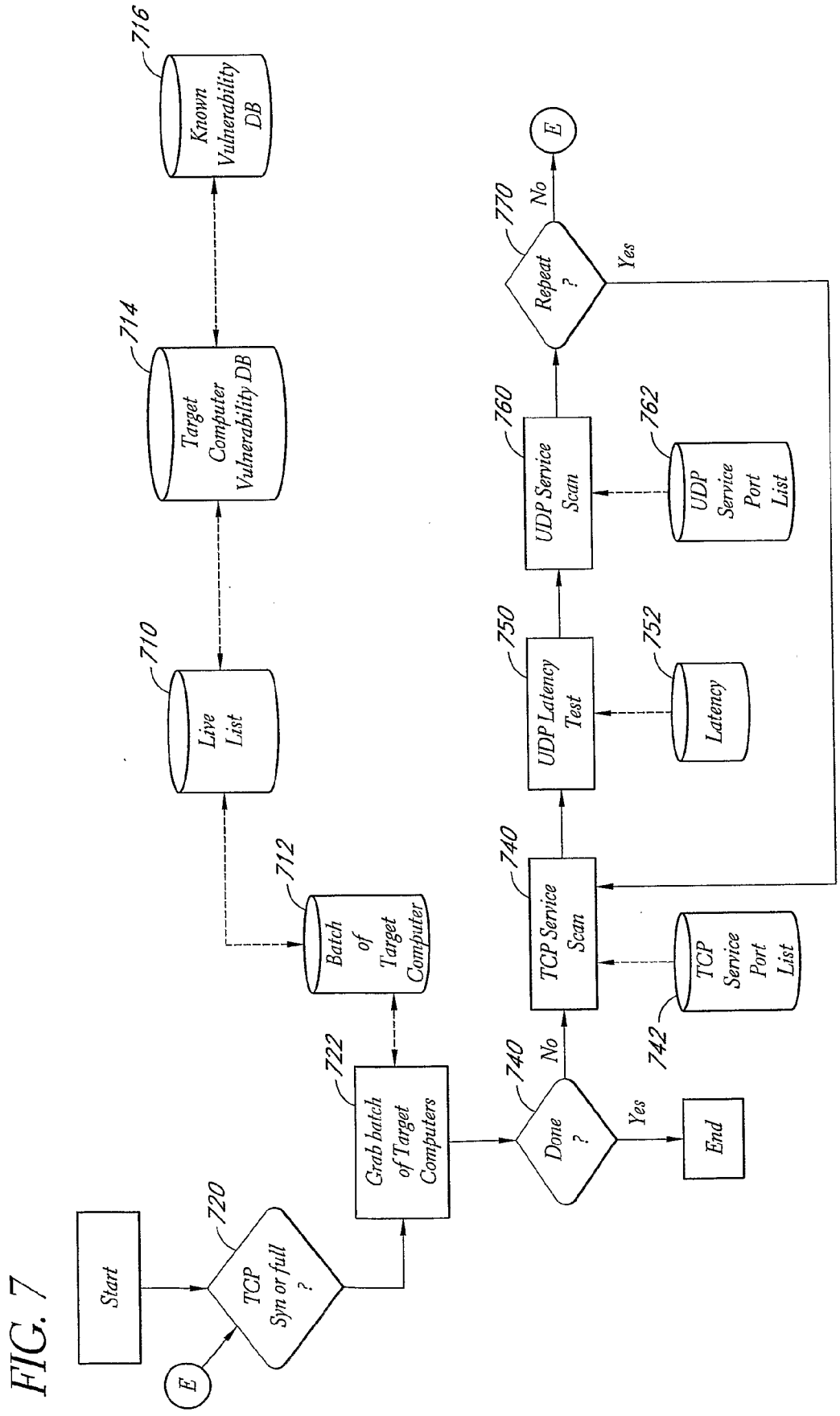

Start
E
720
TCP Syn or full ?
722
Grab batch of Target Computers
712
Batch of Target Computer
710
Live List
714
Target Computer Vulnerability DB
716
Known Vulnerability DB
740
Done ?
No
Yes
End
740
TCP Service Scan
742
TCP Service Port List
750
UDP Latency Test
752
Latency
760
UDP Service Scan
762
UDP Service Port List
770
Repeat ?
No
Yes
E

FIG. 7

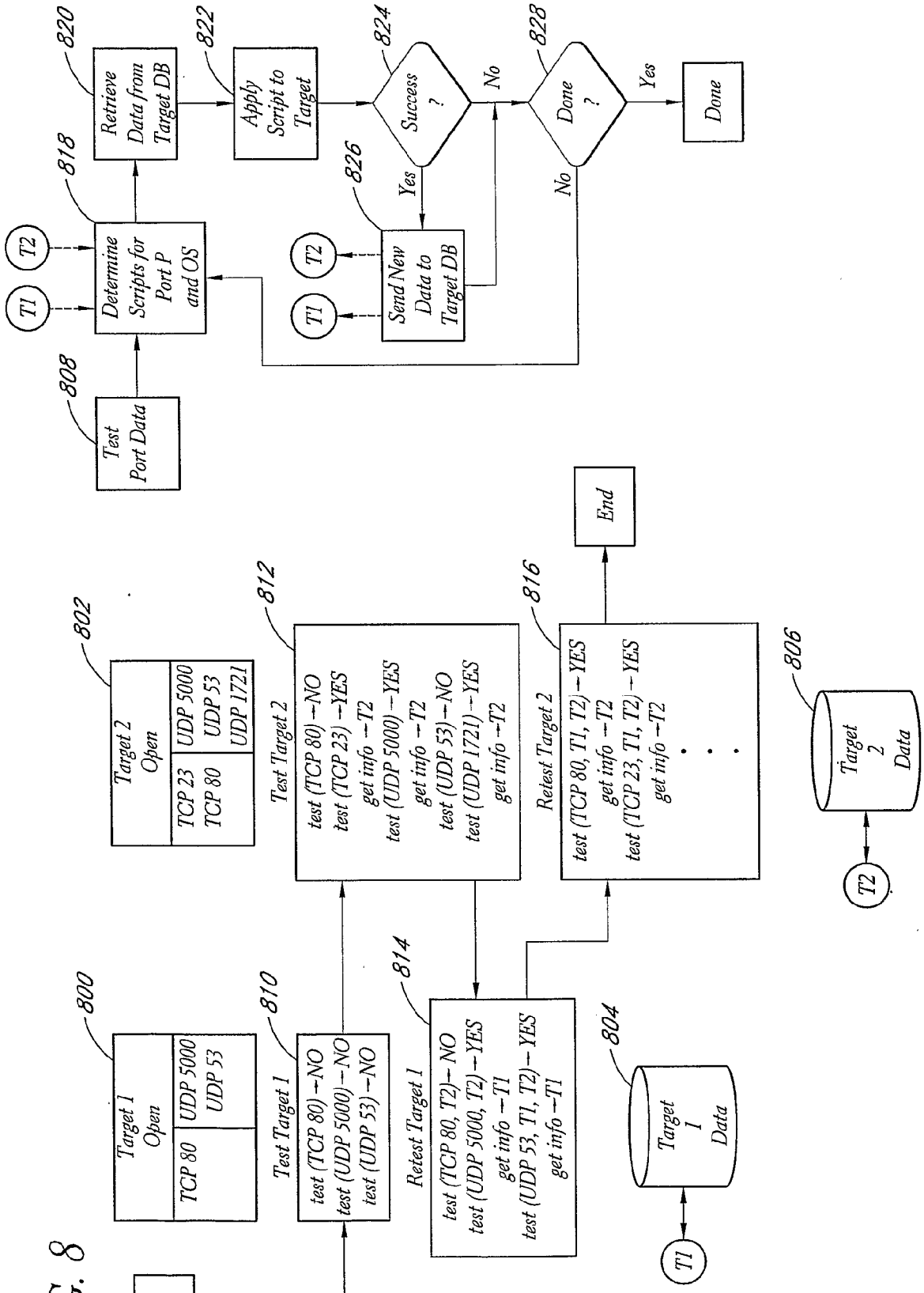

Start
Target 1 Open
TCP 80
UDP 5000
UDP 53
800
Test Target 1
test (TCP 80) – NO
test (UDP 5000) – NO
test (UDP 53) – NO
810
Retest Target 1
test (TCP 80, T2) – NO
test (UDP 5000, T2) – YES
get info – T1
test (UDP 53, T1, T2) – YES
get info – T1
814
T1
Target 1 Data
804
Target 2 Open
TCP 23
TCP 80
UDP 5000
UDP 53
UDP 1721
802
Test Target 2
test (TCP 80) – NO
test (TCP 23) – YES
get info – T2
test (UDP 5000) – YES
get info – T2
test (UDP 53) – NO
test (UDP 1721) – YES
get info – T2
812
Retest Target 2
test (TCP 80, T1, T2) – YES
get info – T2
test (TCP 23, T1, T2) – YES
get info – T2
816
End
T2
Target 2 Data
806
Test Port Data
808
Determine Scripts for Port P and OS
818
T1
T2
Retrieve Data from Target DB
820
Apply Script to Target
822
Success ?
824
Yes
No
Send New Data to Target DB
826
Done ?
828
Yes
No
Done

FIG. 8

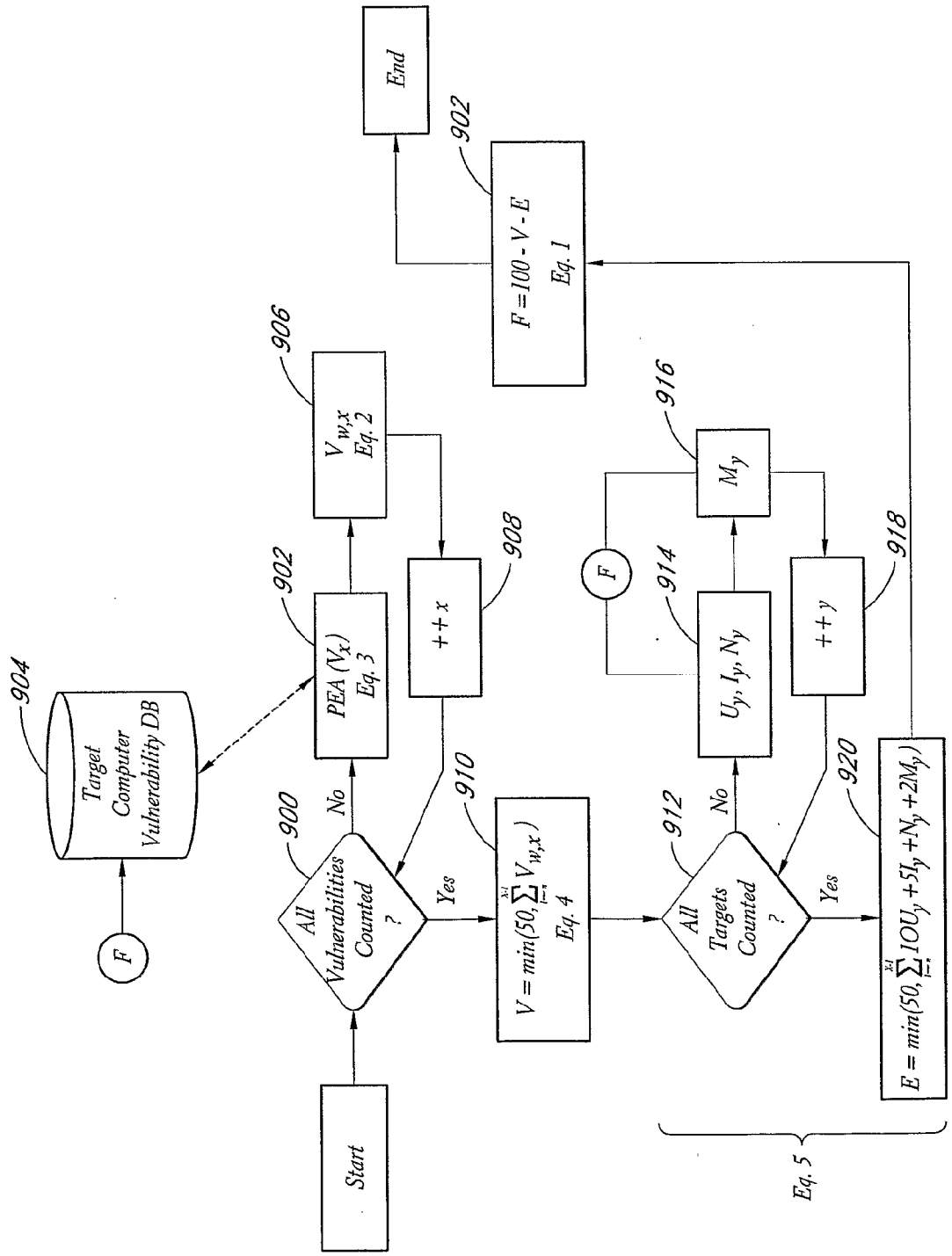

Start
All Vulnerabilities Counted ?
No
Yes
PEA (V_x) Eq. 3
$V_{w,x}$ Eq. 2
++ x
$V = min(50, \sum_{i=1}^{x,l} V_{w,x})$ Eq. 4
All Targets Counted ?
No
Yes
U_y, I_y, N_y
M_y
++ y
$E = min(50, \sum_{i=1}^{x,l} IOU_y + 5I_y + N_y + 2M_y)$
Eq. 5
$F = 100 - V - E$ Eq. 1
End
Target Computer Vulnerability DB
F
900
902
904
906
908
910
912
914
916
918
920

FIG. 9

FIG. 10

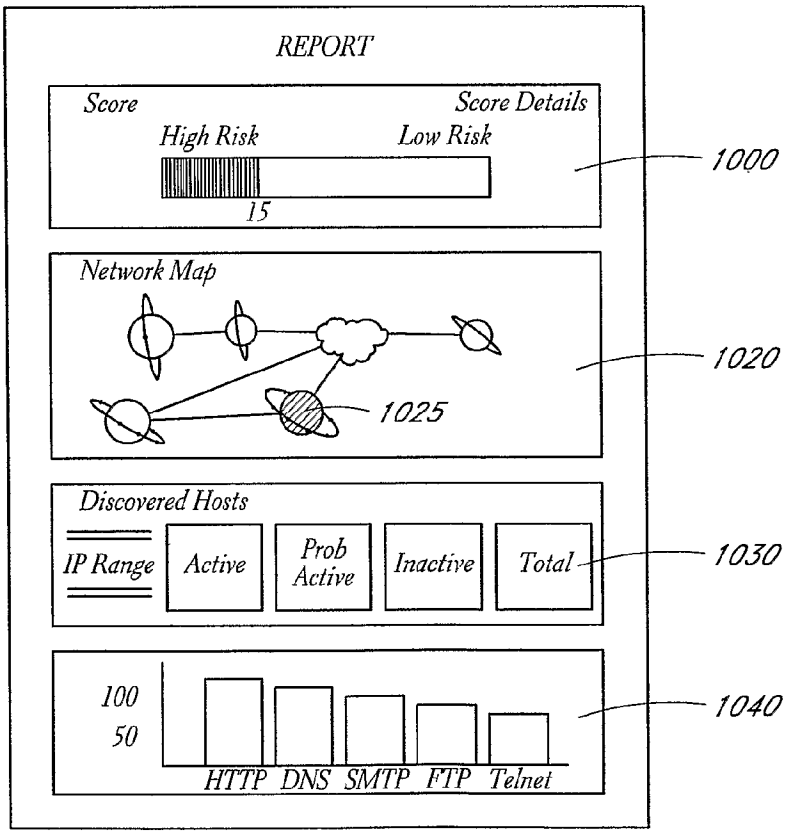
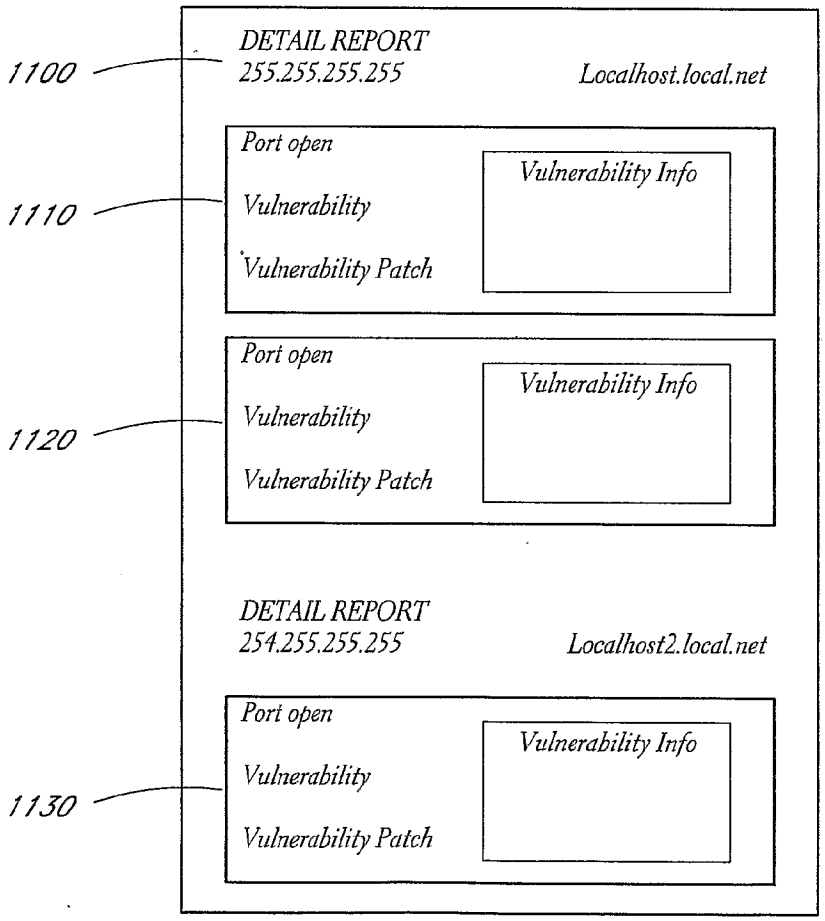

REPORT
Score
Score Details
High Risk
Low Risk
15
1000
Network Map
1025
1020
Discovered Hosts
IP Range
Active
Prob Active
Inactive
Total
1030
100
50
HTTP DNS SMTP FTP Telnet
1040

FIG. 11


1100
DETAIL REPORT
255.255.255.255
Localhost.local.net
1110
Port open
Vulnerability
Vulnerability Patch
Vulnerability Info
1120
Port open
Vulnerability
Vulnerability Patch
Vulnerability Info
DETAIL REPORT
254.255.255.255
Localhost2.local.net
1130
Port open
Vulnerability
Vulnerability Patch
Vulnerability Info

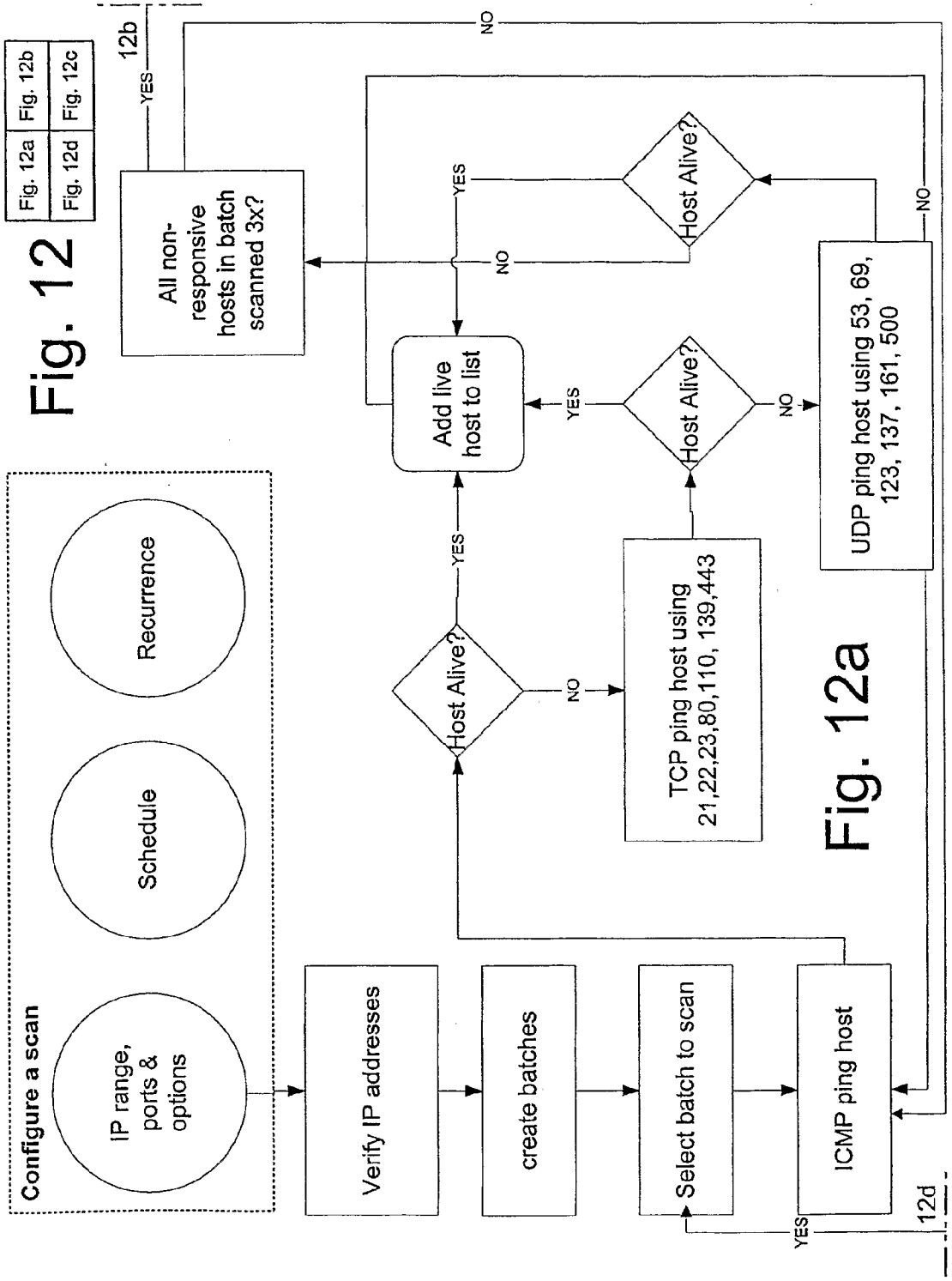


Fig. 12

Fig. 12a	Fig. 12b
Fig. 12d	Fig. 12c

Fig. 12a

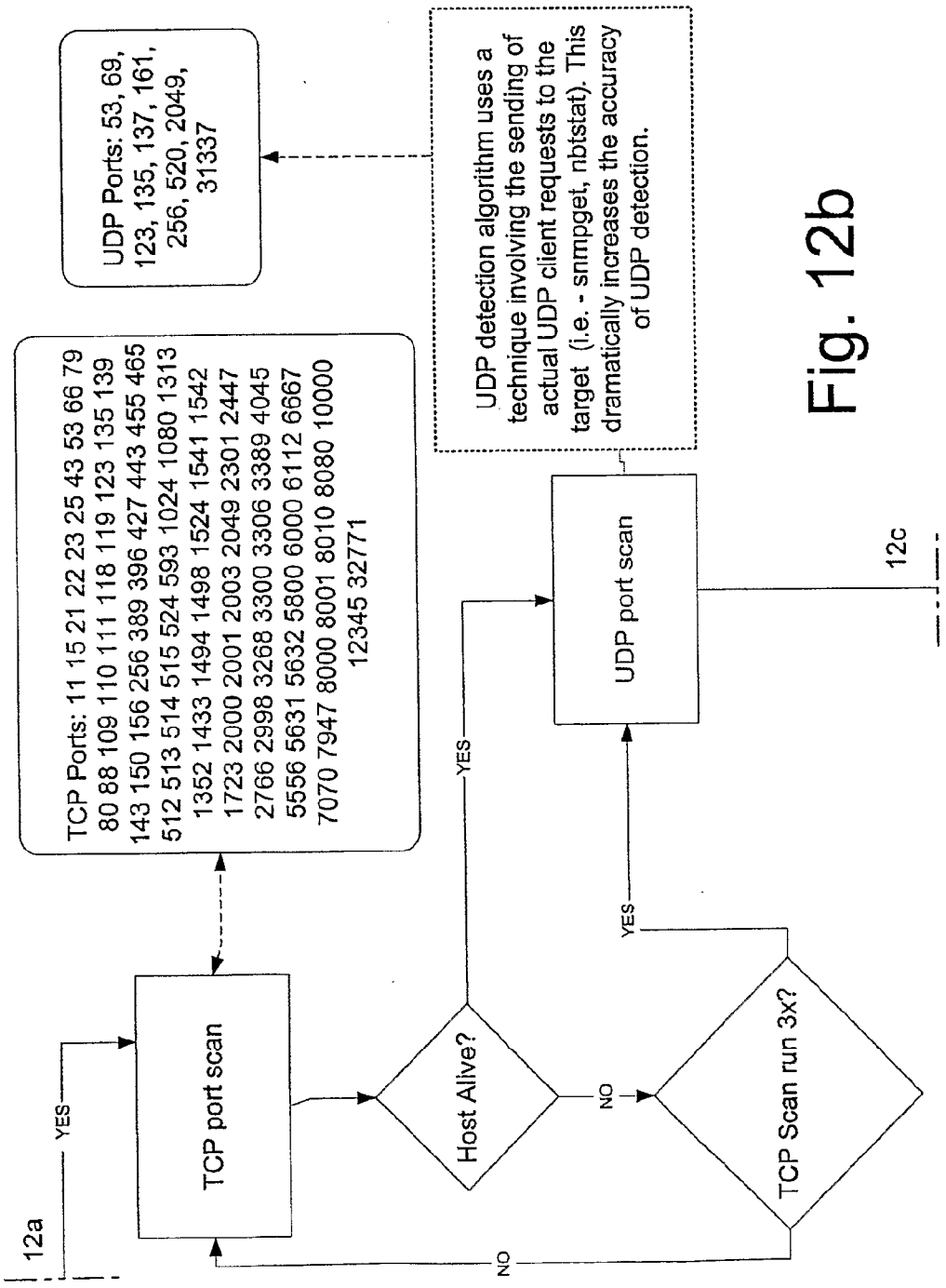

12a
YES
TCP port scan
TCP Ports: 11 15 21 22 23 25 43 53 66 79
80 88 109 110 111 118 119 123 135 139
143 150 156 256 389 396 427 443 455 465
512 513 514 515 524 593 1024 1080 1313
1352 1433 1494 1498 1524 1541 1542
1723 2000 2001 2003 2049 2301 2447
2766 2998 3268 3300 3306 3389 4045
5556 5631 5632 5800 6000 6112 6667
7070 7947 8000 8001 8010 8080 10000
12345 32771
Host Alive?
YES
NO
TCP Scan run 3x?
YES
NO
UDP port scan
UDP Ports: 53, 69,
123, 135, 137, 161,
256, 520, 2049,
31337
UDP detection algorithm uses a technique involving the sending of actual UDP client requests to the target (i.e. - snmpget, nbtstat). This dramatically increases the accuracy of UDP detection.
12c

Fig. 12b

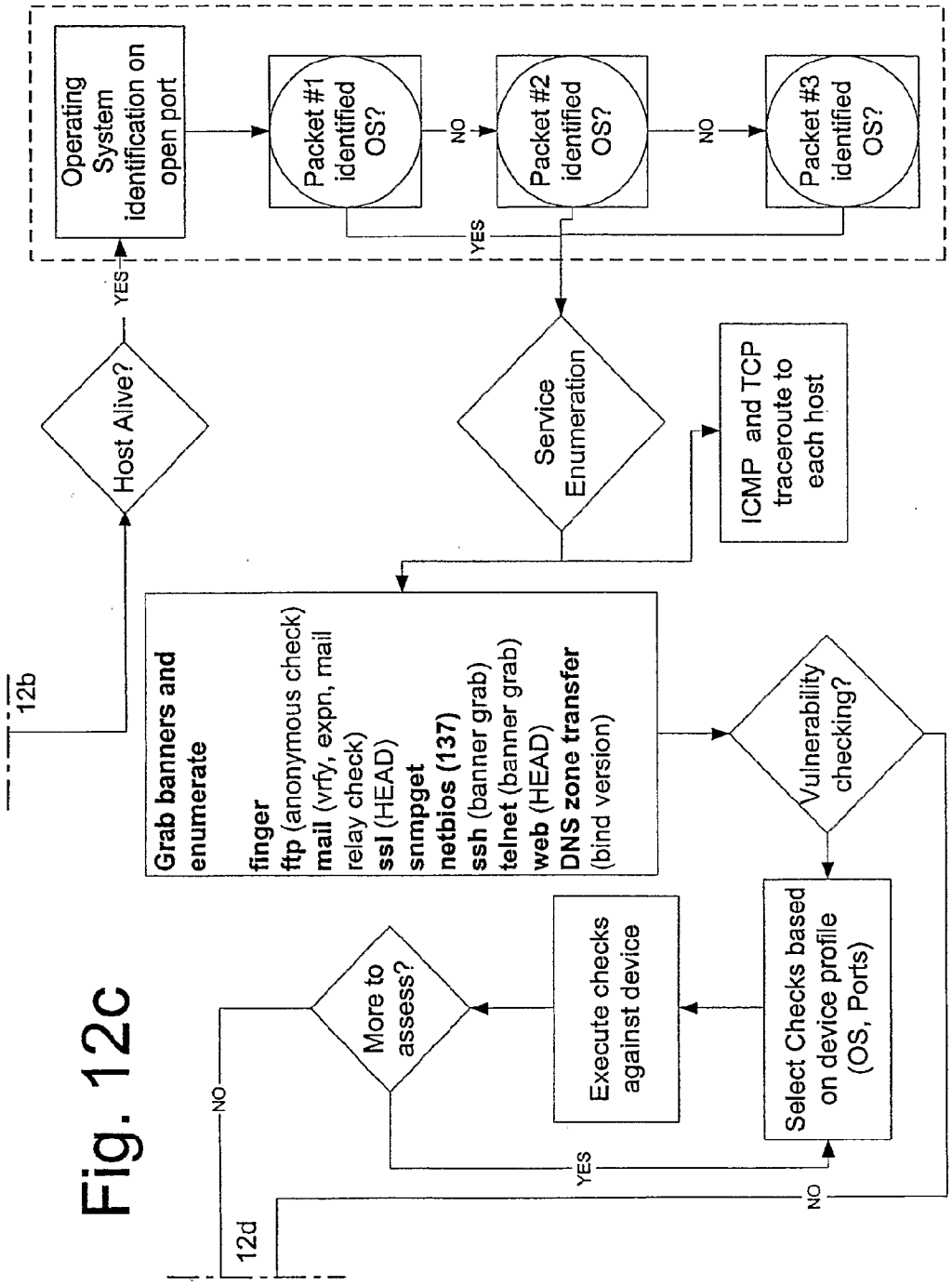
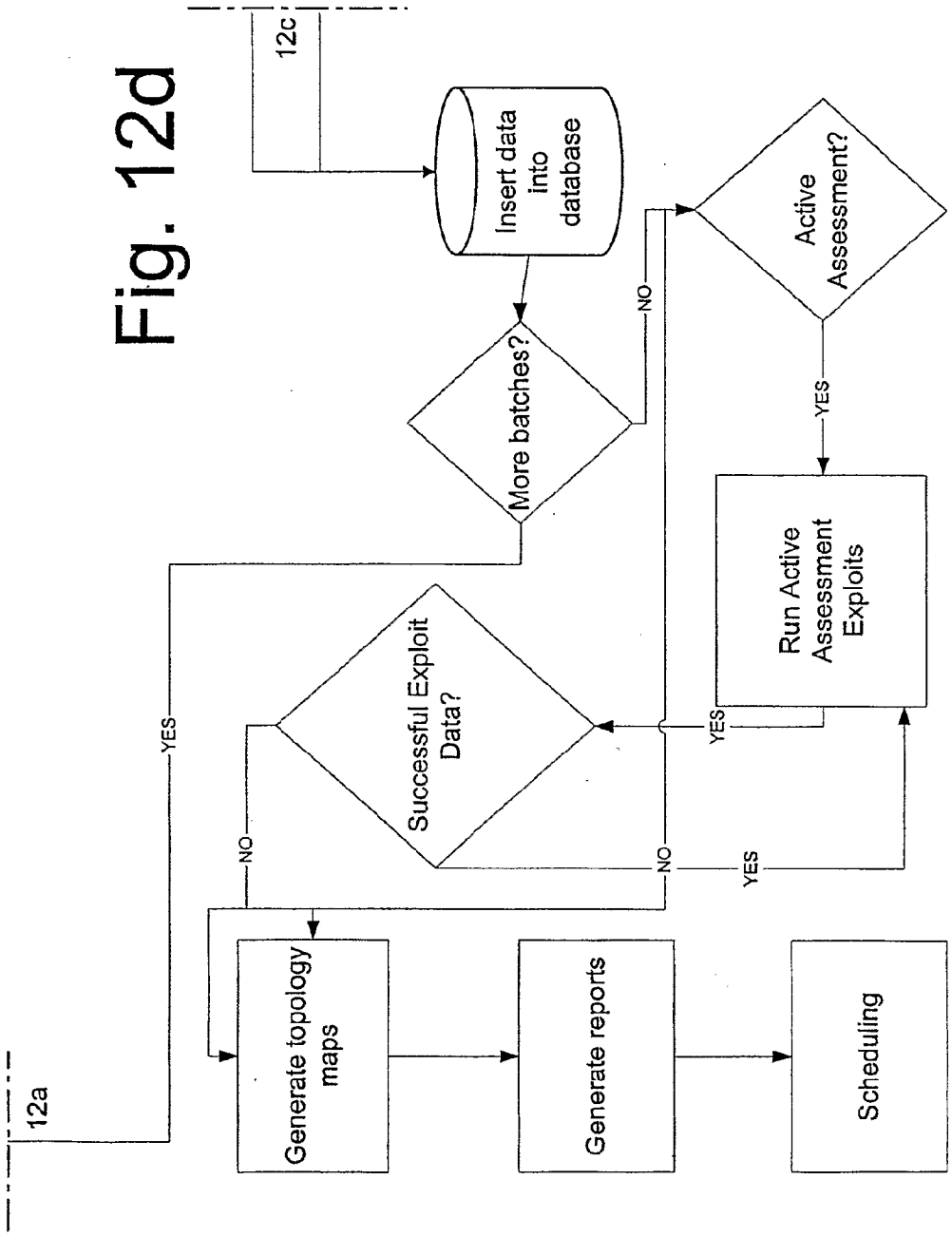

Operating System identification on open port
Packet #1 identified OS?
NO
Packet #2 identified OS?
NO
Packet #3 identified OS?
YES
YES
Host Alive?
Service Enumeration
ICMP and TCP traceroute to each host
12b
Grab banners and enumerate
finger
ftp (anonymous check)
mail (vrfy, expn, mail relay check)
ssl (HEAD)
snmpget
netbios (137)
ssh (banner grab)
telnet (banner grab)
web (HEAD)
DNS zone transfer (bind version)
Vulnerability checking?
Select Checks based on device profile (OS, Ports)
Execute checks against device
More to assess?
NO
YES
NO
12d

Fig. 12c

Fig. 12d


12c
Insert data into database
Active Assessment?
More batches?
NO
YES
Run Active Assessment Exploits
Successful Exploit Data?
YES
NO
YES
NO
YES
12a
Generate topology maps
Generate reports
Scheduling

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US02/01093

A. CLASSIFICATION OF SUBJECT MATTER

IPC(7) :G06F 11/30
US CL : 713/200,201

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

U.S. : 713/200,201

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

Please See Extra Sheet.

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	US 6,282,546 B1(GLEICHAUF et al.) 28 August 2001, col.4,lines 41-64, col.5,lines 22-36.	52-56,58-65
Y	US 6,298,445 B1(SHOSTACK et al.) 02 October 2001, col.6,lines 53-67, col.7,lines 1-30,col.12,lines 14-26.	52-56, 58-65
A	US 5,892,903 A(KLAUS) 06 April 1999, col.11,lines 15-31, col.12,lines 28-55	1-66
A	US 5,931,946 A(TERADA et al.) 03 August 1999, col.5,lines 39-67, col.6,lines 7-61	1-66

☐ Further documents are listed in the continuation of Box C. ☐ See patent family annex.

*	Special categories of cited documents:	"T"	later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"A"	document defining the general state of the art which is not considered to be of particular relevance	"X"	document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"E"	earlier document published on or after the international filing date	"Y"	document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art
"L"	document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)	"&"	document member of the same patent family
"O"	document referring to an oral disclosure, use, exhibition or other means		
"P"	document published prior to the international filing date but later than the priority date claimed		

Date of the actual completion of the international search 28 JUNE 2002	Date of mailing of the international search report 05 AUG 2002
Name and mailing address of the ISA/US Commissioner of Patents and Trademarks Box PCT Washington, D.C. 20231 Facsimile No. (703) 305-3230	Authorized officer GAIL HAYES Peggy Harrod Telephone No. (703) 305-0042

Form PCT/ISA/210 (second sheet) (July 1998)★

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US02/01093

B. FIELDS SEARCHED
Electronic data bases consulted (Name of data base and where practicable terms used):

STN,EAST
search terms: topology,intrusion,detectioon,vulnerability,weakness,firewall,attack,spoofing,expolitation, monitoring,access.

UNITED STATES
PATENT AND TRADEMARK OFFICE

P.O. Box 1450
Alexandria, VA 22313 - 1450
www.uspto.gov

ELECTRONIC ACKNOWLEDGEMENT RECEIPT

APPLICATION #	RECEIPT DATE / TIME	ATTORNEY DOCKET #
17/657,981	**02/08/2023 03:08:46 PM ET**	**15635.0008-00000**

Title of Invention

Forward and Rearward Facing Attack Vector Visualization

Application Information

APPLICATION TYPE	Utility - Nonprovisional Application under 35 USC 111(a)	PATENT #	-
CONFIRMATION #	2397	FILED BY	Michelle Daniels
PATENT CENTER #	61566251	FILING DATE	04/05/2022
CUSTOMER #	22852	FIRST NAMED INVENTOR	Avi SHUA
CORRESPONDENCE ADDRESS	-	AUTHORIZED BY	Christopher Johns

Documents

TOTAL DOCUMENTS: 5

DOCUMENT	PAGES	DESCRIPTION	SIZE (KB)
15635.0008-IDS after Allowance.pdf	3	Transmittal Letter	18 KB
15635.0008-00000-sb08.pdf	1	Information Disclosure Statement (IDS) Form (SB08)	25 KB
Warning: This is not a USPTO supplied IDS fillable form. Data in the form cannot be automatically loaded to other USPTO systems.			
FOR-WO-03060717 A1.pdf	138	Foreign Reference	10248 KB
FOR-WO-2021108569 A1.pdf	131	Foreign Reference	8267 KB
NPL-Notice of Allowance -	16	Non Patent Literature	1075 KB

02-01-23 - 17657972 - 15635.0005.pdf

Digest

DOCUMENT	MESSAGE DIGEST(SHA-512)
15635.0008-IDS after Allowance.pdf	22EFE662EA1E9BC04078AB3BA7ED6CA043E58714C5AE238727201FFF509F55EC5A6386EB02964791444CBDBB4BBEE3DE1CBCCE78DD5C504D0EECF4CDCDF50C40
15635.0008-00000-sb08.pdf	77FB408EF8D52C8EA60530E943FDB3C4929B7D96427BBAD8CD915623F11E5FE6D62802619898A0B9A369E01BD273FA7993B2B52A209700C732B4218EE8085515
FOR-WO-03060717 A1.pdf	C79943D5CCD7247A00F9ED76A078F5F429CA909579D75F687E71A69362AC99A29E93661BA1A734A8F3F006A5B13FCAD23025DA7F855561660D49BEB14298B37B
FOR-WO-2021108569 A1.pdf	E14DDF167A9A710755288F116A002BE863A16FFFCBD645E0D2B8FF5EDDE62695443AAD4529D6C8DADE923944276F0D8F75C6739AB2897A5F5933ECFAABE0242A
NPL-Notice of Allowance - 02-01-23 - 17657972 - 15635.0005.pdf	9EA1D46636F78072FA435B6FA67A319B1A00D0107E33204914A273E909991EBB673B956B250EB6708E30BB7FB24F8537790E683B4A2202A0FCD3111A77562A03

This Acknowledgement Receipt evidences receipt on the noted date by the USPTO of the indicated documents, characterized by the applicant, and including page counts, where applicable. It serves as evidence of receipt similar to a Post Card, as described in MPEP 503.

New Applications Under 35 U.S.C. 111

If a new application is being filed and the application includes the necessary components for filing date (see 37 CFR 1.53(b)-(d) and MPEP 506), a Filing Receipt (37 CFR 1.54) will be issued in due course and the date shown on this Acknowledgement Receipt will establish the filing date of the application

National Stage of an International Application under 35 U.S.C. 371

If a timely submission to enter the national stage of an international application is compliant with the conditions of 35 U.S.C. 371 and other applicable requirements a Form PCT/DO/EO/903 indicating acceptance of the application as a national stage submission under 35 U.S.C. 371 will be issued in addition to the Filing Receipt, in due course.

New International Application Filed with the USPTO as a Receiving Office

If a new international application is being filed and the international application includes the necessary components for an international filing date (see PCT Article 11 and MPEP 1810), a Notification of the International Application Number and of the International Filing Date (Form PCT/RO/105) will be issued in due course, subject to prescriptions concerning national security, and the date shown on this Acknowledgement Receipt will establish the international filing date of the application.

UNITED STATES
PATENT AND TRADEMARK OFFICE

P.O. Box 1450
Alexandria, VA 22313 - 1450
www.uspto.gov

ELECTRONIC PAYMENT RECEIPT

APPLICATION #	RECEIPT DATE / TIME	ATTORNEY DOCKET #
17/657,981	02/08/2023 03:08:46 PM ET	15635.0008-00000

Title of Invention

Forward and Rearward Facing Attack Vector Visualization

Application Information

APPLICATION TYPE	Utility - Nonprovisional Application under 35 USC 111(a)	PATENT #	-
CONFIRMATION #	2397	FILED BY	Michelle Daniels
PATENT CENTER #	61566251	AUTHORIZED BY	Christopher Johns
CUSTOMER #	22852	FILING DATE	04/05/2022
CORRESPONDENCE ADDRESS	-	FIRST NAMED INVENTOR	Avi SHUA

Payment Information

PAYMENT METHOD	PAYMENT TRANSACTION ID	PAYMENT AUTHORIZED BY
CARD / 1007	E202328F09383665	Michelle Daniels

FEE CODE	DESCRIPTION	ITEM PRICE($)	QUANTITY	ITEM TOTAL($)
2806	SUBMISSION OF AN INFORMATION DISCLOSURE STATEMENT	104.00	1	104.00
			TOTAL AMOUNT:	$104.00

This Acknowledgement Receipt evidences receipt on the noted date by the USPTO of the indicated documents, characterized by the applicant, and including page counts, where applicable. It serves as evidence of receipt similar to a Post Card, as described in MPEP 503.

New Applications Under 35 U.S.C. 111

If a new application is being filed and the application includes the necessary components for filing date (see 37 CFR 1.53(b)-(d) and MPEP 506), a Filing Receipt (37 CFR 1.54) will be issued in due course and the date shown on this Acknowledgement Receipt will establish the filing date of the application

National Stage of an International Application under 35 U.S.C. 371

If a timely submission to enter the national stage of an international application is compliant with the conditions of 35 U.S.C. 371 and other applicable requirements a Form PCT/DO/EO/903 indicating acceptance of the application as a national stage submission under 35 U.S.C. 371 will be issued in addition to the Filing Receipt, in due course.

New International Application Filed with the USPTO as a Receiving Office

If a new international application is being filed and the international application includes the necessary components for an international filing date (see PCT Article 11 and MPEP 1810), a Notification of the International Application Number and of the International Filing Date (Form PCT/RO/105) will be issued in due course, subject to prescriptions concerning national security, and the date shown on this Acknowledgement Receipt will establish the international filing date of the application.

PATENT
Attorney Docket No. 15635.0008-00000

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Application of:	)	
	)	
Avi SHUA	)	Group Art Unit: 2431
	)	
Application No.: 17/657,981	)	Examiner: SU, SARAH
	)	
Filed: April 5, 2022	)	Confirmation No.: 2397
	)	
For: FORWARD AND REARWARD FACING ATTACK VECTOR VISUALIZATION	)	

Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Commissioner:

SUPPLEMENTAL INFORMATION DISCLOSURE STATEMENT UNDER 37 C.F.R. § 1.97(d)

Pursuant to 37 C.F.R. §§ 1.56 and 1.97(d), Applicant brings to the attention of the Examiner the documents on the attached listing. This Information Disclosure Statement is being filed after a Notice of Allowance and is accompanied by a fee of $104.00 as specified under § 1.17(p) and a statement as specified under § 1.97(e).

Statement under 37 C.F.R. § 1.97(e)

The following constitutes the statement specified by 37 C.F.R. § 1.97(e)(2). No item of information contained in the IDS was cited in a communication from a foreign patent office in a counterpart foreign application, and, to the knowledge of the person signing the certification after making reasonable inquiry, no item of information contained in the IDS was known to any individual designated in 37 CFR 1.56(c) more

than three months prior to the filing of this Information Disclosure Statement. See 37 CFR 1.97(e)(2).

A copy of the non-patent literature documents is attached.

Applicant respectfully requests that the Examiner consider the listed documents on the attached Form PTO/SB/08 and indicate that they were considered by making appropriate notations on the attached form.

This submission does not represent that a search has been made or that no better art exists and does not constitute an admission that each or all of the listed documents are material or constitute "prior art." If the Examiner applies any of the documents as prior art against any claim in the application and Applicant determines that the cited documents do not constitute "prior art" under United States law, Applicant reserves the right to present to the U.S. Patent and Trademark Office the relevant facts and law regarding the appropriate status of such documents.

Applicant further reserves the right to take appropriate action to establish the patentability of the disclosed invention over the listed documents, should one or more of the documents be applied against the claims of the present application.

If there is any fee due in connection with the filing of this Statement, please charge the fee to deposit account 06-0916.

Respectfully submitted,

FINNEGAN, HENDERSON, FARABOW,
GARRETT & DUNNER, L.L.P.

Dated: February 8, 2023

By: /Christopher C. Johns/
Christopher C. Johns
Reg. No. 68,664
(202) 408-4000

(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
03 June 2021 (03.06.2021)

WIPO | PCT

(10) International Publication Number
WO 2021/108569 A1

(51) International Patent Classification:
H04L 29/06 (2006.01) *G06F 11/00* (2006.01)

(21) International Application Number:
PCT/US2020/062252

(22) International Filing Date:
25 November 2020 (25.11.2020)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
62/940,733 26 November 2019 (26.11.2019) US

(71) Applicant: **RELIAQUEST HOLDINGS, LLC** [US/US]; 777 S. Harbour Island Blvd., Suite 500, Tampa, FL 33602 (US).

(72) Inventors: **MURPHY, Brian, P.**; 777 S. Harbour Island Blvd., Suite 500, Tampa, FL 33602 (US). **PARTLOW, Joe**; 777 S. Harbour Island Blvd., Suite 500, Tampa, FL 33602 (US). **O'CONNOR, Colin**; 777 S. Harbour Island Blvd., Suite 500, Tampa, FL 33602 (US). **PFEIFFER, Jason**; 777 S. Harbour Island Blvd., Suite 500, Tampa, FL 33602 (US). **MURPHY, Brian, Philip**; 777 S. Harbour Island Blvd., Suite 500, Tampa, FL 33602 (US).

(74) Agent: **COLANDREO, Brian, J.** et al.; Holland & Knight LLP, 10 St. James Avenue, Boston, MA 02116 (US).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DJ, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IR, IS, IT, JO, JP, KE, KG, KH, KN, KP, KR, KW, KZ, LA, LC, LK, LR, LS, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA, SC, SD, SE, SG, SK, SL, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, WS, ZA, ZM, ZW.

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, ST, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM,

(54) Title: THREAT MITIGATION SYSTEM AND METHOD

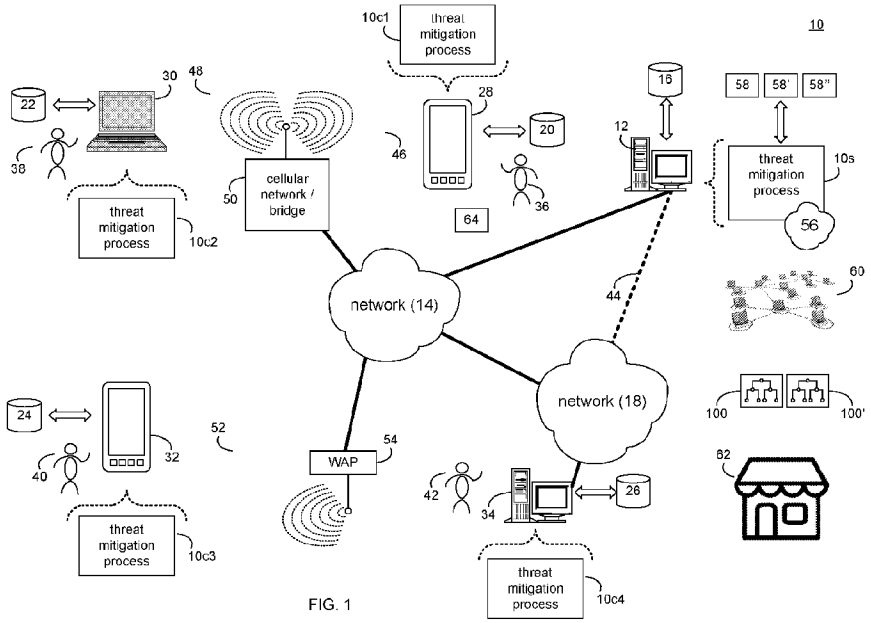


FIG. 1

(57) Abstract: A computer-implemented method, computer program product and computing system for: establishing connectivity with a plurality of security-relevant subsystems within a computing platform; and mapping one or more data fields of a unified platform to one or more data fields of each of the plurality of security-relevant subsystems.

WO 2021/108569 A1

[Continued on next page]

TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG).

Published:

— *with international search report (Art. 21(3))*

Threat Mitigation System and Method

Related Application(s)

[001] This application claims the benefit of U.S. Provisional Application No.: 62/940,733, filed on 26 November 2019, the entire contents of which are herein incorporated by reference

Technical Field

[002] This disclosure relates to threat mitigation systems and, more particularly, to threat mitigation systems that utilize a universal query language.

Background

[003] In the computer world, there is a constant battle occurring between bad actors that want to attack computing platforms and good actors who try to prevent the same. Unfortunately, the complexity of such computer attacks in constantly increasing, so technology needs to be employed that understands the complexity of these attacks and is capable of addressing the same.

[004] Threat mitigation systems may utilize and/or communicate with a plurality of security-relevant subsystems, wherein these security-relevant subsystems may gather information concerning such computer attacks. Unfortunately and in order to obtain such gathered information from these security-relevant subsystems, the user of the threat mitigation system would often be required to formulate a unique query for each security-relevant subsystem.

Summary of Disclosure

Concept 2

[005] In one implementation, a computer-implemented method is executed on a computing device and includes: establishing connectivity with a plurality of security-relevant subsystems within a computing platform; and mapping one or more data fields of a unified platform to one or more data fields of each of the plurality of security-relevant subsystems.

[006] One or more of the following features may be included. Mapping one or more data fields of a unified platform to one or more data fields of each of the plurality of security-relevant subsystems may include: mapping one or more data fields within a query structure of the unified platform to one or more data fields within a query structure of each of the plurality of security-relevant subsystems. Mapping one or more data fields of a unified platform to one or more data fields of each of the plurality of security-relevant subsystems may include: mapping one or more data fields within a result set structure of each of the plurality of security-relevant subsystems to one or more data fields within a result set structure of the unified platform. Mapping one or more data fields of a unified platform to one or more data fields of each of the plurality of security-relevant subsystems may include: mapping one or more data fields of the unified platform to one or more data fields of each of the plurality of security-relevant subsystems at a defined periodicity. Mapping one or more data fields of a unified platform to one or more data fields of each of the plurality of security-relevant subsystems may include: proactively mapping one or more data fields of the unified platform to one or more data fields of each of the plurality of security-relevant subsystems. Mapping one or more data fields of a unified platform to one or more data

fields of each of the plurality of security-relevant subsystems may include: reactively mapping one or more data fields of the unified platform to one or more data fields of each of the plurality of security-relevant subsystems. A unified query may be defined on a unified platform concerning the plurality of security-relevant subsystems. The unified query may be denormalized to define a subsystem-specific query for each of the plurality of security-relevant subsystems, thus defining a plurality of subsystem-specific queries. The plurality of subsystem-specific queries may be provided to the plurality of security-relevant subsystems. A plurality of subsystem-specific results sets may be received from the plurality of security-relevant subsystems that were generated in response to the plurality of subsystem-specific queries. The plurality of subsystem-specific results sets received from the plurality of security-relevant subsystems may be normalized to define a unified result set. The unified result may be provided set to a third-party.

[007] In another implementation, a computer program product resides on a computer readable medium and has a plurality of instructions stored on it. When executed by a processor, the instructions cause the processor to perform operations including: establishing connectivity with a plurality of security-relevant subsystems within a computing platform; and mapping one or more data fields of a unified platform to one or more data fields of each of the plurality of security-relevant subsystems.

[008] One or more of the following features may be included. Mapping one or more data fields of a unified platform to one or more data fields of each of the plurality of security-relevant subsystems may include: mapping one or more data fields within a query structure of the unified platform to one or more data fields within a query structure of each of the plurality of security-relevant subsystems. Mapping one or more data fields of a unified platform to one or more data fields of each of the plurality of security-relevant subsystems may include: mapping one or more data fields within a result set structure of each of the plurality of security-relevant subsystems to one or more data fields within a result set structure of the unified platform. Mapping one or more data

fields of a unified platform to one or more data fields of each of the plurality of security-relevant subsystems may include: mapping one or more data fields of the unified platform to one or more data fields of each of the plurality of security-relevant subsystems at a defined periodicity. Mapping one or more data fields of a unified platform to one or more data fields of each of the plurality of security-relevant subsystems may include: proactively mapping one or more data fields of the unified platform to one or more data fields of each of the plurality of security-relevant subsystems. Mapping one or more data fields of a unified platform to one or more data fields of each of the plurality of security-relevant subsystems may include: reactively mapping one or more data fields of the unified platform to one or more data fields of each of the plurality of security-relevant subsystems. A unified query may be defined on a unified platform concerning the plurality of security-relevant subsystems. The unified query may be denormalized to define a subsystem-specific query for each of the plurality of security-relevant subsystems, thus defining a plurality of subsystem-specific queries. The plurality of subsystem-specific queries may be provided to the plurality of security-relevant subsystems. A plurality of subsystem-specific results sets may be received from the plurality of security-relevant subsystems that were generated in response to the plurality of subsystem-specific queries. The plurality of subsystem-specific results sets received from the plurality of security-relevant subsystems may be normalized to define a unified result set. The unified result may be provided set to a third-party.

[009] In another implementation, a computing system includes a processor and memory is configured to perform operations including: establishing connectivity with a plurality of security-relevant subsystems within a computing platform; and mapping one or more data fields of a unified platform to one or more data fields of each of the plurality of security-relevant subsystems.

[0010] One or more of the following features may be included. Mapping one or more data fields of a unified platform to one or more data fields of each of the plurality of

security-relevant subsystems may include: mapping one or more data fields within a query structure of the unified platform to one or more data fields within a query structure of each of the plurality of security-relevant subsystems. Mapping one or more data fields of a unified platform to one or more data fields of each of the plurality of security-relevant subsystems may include: mapping one or more data fields within a result set structure of each of the plurality of security-relevant subsystems to one or more data fields within a result set structure of the unified platform. Mapping one or more data fields of a unified platform to one or more data fields of each of the plurality of security-relevant subsystems may include: mapping one or more data fields of the unified platform to one or more data fields of each of the plurality of security-relevant subsystems at a defined periodicity. Mapping one or more data fields of a unified platform to one or more data fields of each of the plurality of security-relevant subsystems may include: proactively mapping one or more data fields of the unified platform to one or more data fields of each of the plurality of security-relevant subsystems. Mapping one or more data fields of a unified platform to one or more data fields of each of the plurality of security-relevant subsystems may include: reactively mapping one or more data fields of the unified platform to one or more data fields of each of the plurality of security-relevant subsystems. A unified query may be defined on a unified platform concerning the plurality of security-relevant subsystems. The unified query may be denormalized to define a subsystem-specific query for each of the plurality of security-relevant subsystems, thus defining a plurality of subsystem-specific queries. The plurality of subsystem-specific queries may be provided to the plurality of security-relevant subsystems. A plurality of subsystem-specific results sets may be received from the plurality of security-relevant subsystems that were generated in response to the plurality of subsystem-specific queries. The plurality of subsystem-specific results sets received from the plurality of security-relevant subsystems may be normalized to define a unified result set. The unified result may be provided set to a third-party.

[0011] The details of one or more implementations are set forth in the accompanying drawings and the description below. Other features and advantages will become apparent from the description, the drawings, and the claims.

Brief Description of the Drawings

[0012] FIG. 1 is a diagrammatic view of a distributed computing network including a computing device that executes a threat mitigation process according to an embodiment of the present disclosure;

[0013] FIG. 2 is a diagrammatic view of an exemplary probabilistic model rendered by a probabilistic process of the threat mitigation process of FIG. 1 according to an embodiment of the present disclosure;

[0014] FIG. 3 is a diagrammatic view of the computing platform of FIG. 1 according to an embodiment of the present disclosure;

[0015] FIG. 4 is a flowchart of an implementation of the threat mitigation process of FIG. 1 according to an embodiment of the present disclosure;

[0016] FIGS. 5-6 are diagrammatic views of screens rendered by the threat mitigation process of FIG. 1 according to an embodiment of the present disclosure;

[0017] FIGS. 7-9 are flowcharts of other implementations of the threat mitigation process of FIG. 1 according to an embodiment of the present disclosure;

[0018] FIG. 10 is a diagrammatic view of a screen rendered by the threat mitigation process of FIG. 1 according to an embodiment of the present disclosure;

[0019] FIG. 11 is a flowchart of another implementation of the threat mitigation process of FIG. 1 according to an embodiment of the present disclosure;

[0020] FIG. 12 is a diagrammatic view of a screen rendered by the threat mitigation process of FIG. 1 according to an embodiment of the present disclosure;

[0021] FIG. 13 is a flowchart of another implementation of the threat mitigation process of FIG. 1 according to an embodiment of the present disclosure;

[0022] FIG. 14 is a diagrammatic view of a screen rendered by the threat mitigation process of FIG. 1 according to an embodiment of the present disclosure;

[0023] FIG. 15 is a flowchart of another implementation of the threat mitigation process of FIG. 1 according to an embodiment of the present disclosure;

[0024] FIG. 16 is a diagrammatic view of screens rendered by the threat mitigation process of FIG. 1 according to an embodiment of the present disclosure;

[0025] FIGS. 17-23 are flowcharts of other implementations of the threat mitigation process of FIG. 1 according to an embodiment of the present disclosure;

[0026] FIG. 24 is a diagrammatic view of a screen rendered by the threat mitigation process of FIG. 1 according to an embodiment of the present disclosure;

[0027] FIGS. 25-31 are flowcharts of other implementations of the threat mitigation process of FIG. 1 according to an embodiment of the present disclosure; and

[0028] FIG. 32 is a diagrammatic view of data field mapping according to an embodiment of the present disclosure.

[0029] Like reference symbols in the various drawings indicate like elements.

Detailed Description of the Preferred Embodiments

System Overview

[0030] Referring to FIG. 1, there is shown threat mitigation process 10. Threat mitigation process 10 may be implemented as a server-side process, a client-side process, or a hybrid server-side / client-side process. For example, threat mitigation process 10 may be implemented as a purely server-side process via threat mitigation process 10s. Alternatively, threat mitigation process 10 may be implemented as a purely client-side process via one or more of threat mitigation process 10c1, threat mitigation process 10c2, threat mitigation process 10c3, and threat mitigation process 10c4. Alternatively still, threat mitigation process 10 may be implemented as a hybrid server-side / client-side process via threat mitigation process 10s in combination with one or more of threat

mitigation process 10c1, threat mitigation process 10c2, threat mitigation process 10c3, and threat mitigation process 10c4. Accordingly, threat mitigation process 10 as used in this disclosure may include any combination of threat mitigation process 10s, threat mitigation process 10c1, threat mitigation process 10c2, threat mitigation process, and threat mitigation process 10c4.

[0031] Threat mitigation process 10s may be a server application and may reside on and may be executed by computing device 12, which may be connected to network 14 (e.g., the Internet or a local area network). Examples of computing device 12 may include, but are not limited to: a personal computer, a laptop computer, a personal digital assistant, a data-enabled cellular telephone, a notebook computer, a television with one or more processors embedded therein or coupled thereto, a cable / satellite receiver with one or more processors embedded therein or coupled thereto, a server computer, a series of server computers, a mini computer, a mainframe computer, or a cloud-based computing network.

[0032] The instruction sets and subroutines of threat mitigation process 10s, which may be stored on storage device 16 coupled to computing device 12, may be executed by one or more processors (not shown) and one or more memory architectures (not shown) included within computing device 12. Examples of storage device 16 may include but are not limited to: a hard disk drive; a RAID device; a random-access memory (RAM); a read-only memory (ROM); and all forms of flash memory storage devices.

[0033] Network 14 may be connected to one or more secondary networks (e.g., network 18), examples of which may include but are not limited to: a local area network; a wide area network; or an intranet, for example.

[0034] Examples of threat mitigation processes 10c1, 10c2, 10c3, 10c4 may include but are not limited to a client application, a web browser, a game console user interface, or a specialized application (e.g., an application running on e.g., the Android ^{tm} platform or the iOS ^{tm} platform). The instruction sets and subroutines of threat mitigation

processes 10c1, 10c2, 10c3, 10c4, which may be stored on storage devices 20, 22, 24, 26 (respectively) coupled to client electronic devices 28, 30, 32, 34 (respectively), may be executed by one or more processors (not shown) and one or more memory architectures (not shown) incorporated into client electronic devices 28, 30, 32, 34 (respectively). Examples of storage device 16 may include but are not limited to: a hard disk drive; a RAID device; a random-access memory (RAM); a read-only memory (ROM); and all forms of flash memory storage devices.

[0035] Examples of client electronic devices 28, 30, 32, 34 may include, but are not limited to, data-enabled, cellular telephone 28, laptop computer 30, personal digital assistant 32, personal computer 34, a notebook computer (not shown), a server computer (not shown), a gaming console (not shown), a smart television (not shown), and a dedicated network device (not shown). Client electronic devices 28, 30, 32, 34 may each execute an operating system, examples of which may include but are not limited to Microsoft Windows ™, Android ™, WebOS ™, iOS ™, Redhat Linux ™, or a custom operating system.

[0036] Users 36, 38, 40, 42 may access threat mitigation process 10 directly through network 14 or through secondary network 18. Further, threat mitigation process 10 may be connected to network 14 through secondary network 18, as illustrated with link line 44.

[0037] The various client electronic devices (e.g., client electronic devices 28, 30, 32, 34) may be directly or indirectly coupled to network 14 (or network 18). For example, data-enabled, cellular telephone 28 and laptop computer 30 are shown wirelessly coupled to network 14 via wireless communication channels 46, 48 (respectively) established between data-enabled, cellular telephone 28, laptop computer 30 (respectively) and cellular network / bridge 50, which is shown directly coupled to network 14. Further, personal digital assistant 32 is shown wirelessly coupled to network 14 via wireless communication channel 52 established between personal digital assistant 32 and wireless

access point (i.e., WAP) 54, which is shown directly coupled to network 14. Additionally, personal computer 34 is shown directly coupled to network 18 via a hardwired network connection.

[0038] WAP 54 may be, for example, an IEEE 802.11a, 802.11b, 802.11g, 802.11n, Wi-Fi, and/or Bluetooth device that is capable of establishing wireless communication channel 52 between personal digital assistant 32 and WAP 54. As is known in the art, IEEE 802.11x specifications may use Ethernet protocol and carrier sense multiple access with collision avoidance (i.e., CSMA/CA) for path sharing. The various 802.11x specifications may use phase-shift keying (i.e., PSK) modulation or complementary code keying (i.e., CCK) modulation, for example. As is known in the art, Bluetooth is a telecommunications industry specification that allows e.g., mobile phones, computers, and personal digital assistants to be interconnected using a short-range wireless connection.

Artificial Intelligence / Machines Learning Overview:

[0039] Assume for illustrative purposes that threat mitigation process 10 includes probabilistic process 56 (e.g., an artificial intelligence / machine learning process) that is configured to process information (e.g., information 58). As will be discussed below in greater detail, examples of information 58 may include but are not limited to platform information (e.g., structured or unstructured content) being scanned to detect security events (e.g., access auditing; anomalies; authentication; denial of services; exploitation; malware; phishing; spamming; reconnaissance; and/or web attack) within a monitored computing platform (e.g., computing platform 60).

[0040] As is known in the art, structured content may be content that is separated into independent portions (e.g., fields, columns, features) and, therefore, may have a pre-defined data model and/or is organized in a pre-defined manner. For example, if the structured content concerns an employee list: a first field, column or feature may define the first name of the employee; a second field, column or feature may define the last

name of the employee; a third field, column or feature may define the home address of the employee; and a fourth field, column or feature may define the hire date of the employee.

[0041] Further and as is known in the art, unstructured content may be content that is not separated into independent portions (e.g., fields, columns, features) and, therefore, may not have a pre-defined data model and/or is not organized in a pre-defined manner. For example, if the unstructured content concerns the same employee list: the first name of the employee, the last name of the employee, the home address of the employee, and the hire date of the employee may all be combined into one field, column or feature.

[0042] For the following illustrative example, assume that information 58 is unstructured content, an example of which may include but is not limited to unstructured user feedback received by a company (e.g., text-based feedback such as text-messages, social media posts, and email messages; and transcribed voice-based feedback such as transcribed voice mail, and transcribed voice messages).

[0043] When processing information 58, probabilistic process 56 may use probabilistic modeling to accomplish such processing, wherein examples of such probabilistic modeling may include but are not limited to discriminative modeling, generative modeling, or combinations thereof.

[0044] As is known in the art, probabilistic modeling may be used within modern artificial intelligence systems (e.g., probabilistic process 56), in that these probabilistic models may provide artificial intelligence systems with the tools required to autonomously analyze vast quantities of data (e.g., information 58).

[0045] Examples of the tasks for which probabilistic modeling may be utilized may include but are not limited to:

- predicting media (music, movies, books) that a user may like or enjoy based upon media that the user has liked or enjoyed in the past;
- transcribing words spoken by a user into editable text;

- grouping genes into gene clusters;
- identifying recurring patterns within vast data sets;
- filtering email that is believed to be spam from a user's inbox;
- generating clean (i.e., non-noisy) data from a noisy data set;
- analyzing (voice-based or text-based) customer feedback; and
- diagnosing various medical conditions and diseases.

[0046] For each of the above-described applications of probabilistic modeling, an initial probabilistic model may be defined, wherein this initial probabilistic model may be subsequently (e.g., iteratively or continuously) modified and revised, thus allowing the probabilistic models and the artificial intelligence systems (e.g., probabilistic process 56) to "learn" so that future probabilistic models may be more precise and may explain more complex data sets.

[0047] Accordingly, probabilistic process 56 may define an initial probabilistic model for accomplishing a defined task (e.g., the analyzing of information 58). For the illustrative example, assume that this defined task is analyzing customer feedback (e.g., information 58) that is received from customers of e.g., store 62 via an automated feedback phone line. For this example, assume that information 58 is initially voice-based content that is processed via e.g., a speech-to-text process that results in unstructured text-based customer feedback (e.g., information 58).

[0048] With respect to probabilistic process 56, a probabilistic model may be utilized to go from initial observations about information 58 (e.g., as represented by the initial branches of a probabilistic model) to conclusions about information 58 (e.g., as represented by the leaves of a probabilistic model).

[0049] As used in this disclosure, the term "branch" may refer to the existence (or non-existence) of a component (e.g., a sub-model) of (or included within) a model. Examples of such a branch may include but are not limited to: an execution branch of a probabilistic program or other generative model, a part (or parts) of a probabilistic

graphical model, and/or a component neural network that may (or may not) have been previously trained.

[0050] While the following discussion provides a detailed example of a probabilistic model, this is for illustrative purposes only and is not intended to be a limitation of this disclosure, as other configurations are possible and are considered to be within the scope of this disclosure. For example, the following discussion may concern any type of model (e.g., be it probabilistic or other) and, therefore, the below-described probabilistic model is merely intended to be one illustrative example of a type of model and is not intended to limit this disclosure to probabilistic models.

[0051] Additionally, while the following discussion concerns word-based routing of messages through a probabilistic model, this is for illustrative purposes only and is not intended to be a limitation of this disclosure, as other configurations are possible and are considered to be within the scope of this disclosure. Examples of other types of information that may be used to route messages through a probabilistic model may include: the order of the words within a message; and the punctuation interspersed throughout the message.

[0052] For example and referring also to FIG. 2, there is shown one simplified example of a probabilistic model (e.g., probabilistic model 100) that may be utilized to analyze information 58 (e.g., unstructured text-based customer feedback) concerning store 62. The manner in which probabilistic model 100 may be automatically-generated by probabilistic process 56 will be discussed below in detail. In this particular example, probabilistic model 100 may receive information 58 (e.g., unstructured text-based customer feedback) at branching node 102 for processing. Assume that probabilistic model 100 includes four branches off of branching node 102, namely: service branch 104; selection branch 106; location branch 108; and value branch 110 that respectively lead to service node 112, selection node 114, location node 116, and value node 118.

[0053] As stated above, service branch 104 may lead to service node 112, which may be configured to process the portion of information 58 (e.g., unstructured text-based customer feedback) that concerns (in whole or in part) feedback concerning the customer service of store 62. For example, service node 112 may define service word list 120 that may include e.g., the word service, as well as synonyms of (and words related to) the word service (e.g., cashier, employee, greeter and manager). Accordingly and in the event that a portion of information 58 (e.g., a text-based customer feedback message) includes the word cashier, employee, greeter and/or manager, that portion of information 58 may be considered to be text-based customer feedback concerning the service received at store 62 and (therefore) may be routed to service node 112 of probabilistic model 100 for further processing. Assume for this illustrative example that probabilistic model 100 includes two branches off of service node 112, namely: good service branch 122 and bad service branch 124.

[0054] Good service branch 122 may lead to good service node 126, which may be configured to process the portion of information 58 (e.g., unstructured text-based customer feedback) that concerns (in whole or in part) good feedback concerning the customer service of store 62. For example, good service node 126 may define good service word list 128 that may include e.g., the word good, as well as synonyms of (and words related to) the word good (e.g., courteous, friendly, lovely, happy, and smiling). Accordingly and in the event that a portion of information 58 (e.g., a text-based customer feedback message) that was routed to service node 112 includes the word good, courteous, friendly, lovely, happy, and/or smiling, that portion of information 58 may be considered to be text-based customer feedback indicative of good service received at store 62 (and, therefore, may be routed to good service node 126).

[0055] Bad service branch 124 may lead to bad service node 130, which may be configured to process the portion of information 58 (e.g., unstructured text-based customer feedback) that concerns (in whole or in part) bad feedback concerning the

customer service of store 62. For example, bad service node 130 may define bad service word list 132 that may include e.g., the word bad, as well as synonyms of (and words related to) the word bad (e.g., rude, mean, jerk, miserable, and scowling). Accordingly and in the event that a portion of information 58 (e.g., a text-based customer feedback message) that was routed to service node 112 includes the word bad, rude, mean, jerk, miserable, and/or scowling, that portion of information 58 may be considered to be text-based customer feedback indicative of bad service received at store 62 (and, therefore, may be routed to bad service node 130).

[0056] As stated above, selection branch 106 may lead to selection node 114, which may be configured to process the portion of information 58 (e.g., unstructured text-based customer feedback) that concerns (in whole or in part) feedback concerning the selection available at store 62. For example, selection node 114 may define selection word list 134 that may include e.g., words indicative of the selection available at store 62. Accordingly and in the event that a portion of information 58 (e.g., a text-based customer feedback message) includes any of the words defined within selection word list 134, that portion of information 58 may be considered to be text-based customer feedback concerning the selection available at store 62 and (therefore) may be routed to selection node 114 of probabilistic model 100 for further processing. Assume for this illustrative example that probabilistic model 100 includes two branches off of selection node 114, namely: good selection branch 136 and bad selection branch 138.

[0057] Good selection branch 136 may lead to good selection node 140, which may be configured to process the portion of information 58 (e.g., unstructured text-based customer feedback) that concerns (in whole or in part) good feedback concerning the selection available at store 62. For example, good selection node 140 may define good selection word list 142 that may include words indicative of a good selection at store 62. Accordingly and in the event that a portion of information 58 (e.g., a text-based customer feedback message) that was routed to selection node 114 includes any of the words

defined within good selection word list 142, that portion of information 58 may be considered to be text-based customer feedback indicative of a good selection available at store 62 (and, therefore, may be routed to good selection node 140).

[0058] Bad selection branch 138 may lead to bad selection node 144, which may be configured to process the portion of information 58 (e.g., unstructured text-based customer feedback) that concerns (in whole or in part) bad feedback concerning the selection available at store 62. For example, bad selection node 144 may define bad selection word list 146 that may include words indicative of a bad selection at store 62. Accordingly and in the event that a portion of information 58 (e.g., a text-based customer feedback message) that was routed to selection node 114 includes any of the words defined within bad selection word list 146, that portion of information 58 may be considered to be text-based customer feedback indicative of a bad selection being available at store 62 (and, therefore, may be routed to bad selection node 144).

[0059] As stated above, location branch 108 may lead to location node 116, which may be configured to process the portion of information 58 (e.g., unstructured text-based customer feedback) that concerns (in whole or in part) feedback concerning the location of store 62. For example, location node 116 may define location word list 148 that may include e.g., words indicative of the location of store 62. Accordingly and in the event that a portion of information 58 (e.g., a text-based customer feedback message) includes any of the words defined within location word list 148, that portion of information 58 may be considered to be text-based customer feedback concerning the location of store 62 and (therefore) may be routed to location node 116 of probabilistic model 100 for further processing. Assume for this illustrative example that probabilistic model 100 includes two branches off of location node 116, namely: good location branch 150 and bad location branch 152.

[0060] Good location branch 150 may lead to good location node 154, which may be configured to process the portion of information 58 (e.g., unstructured text-based

customer feedback) that concerns (in whole or in part) good feedback concerning the location of store 62. For example, good location node 154 may define good location word list 156 that may include words indicative of store 62 being in a good location. Accordingly and in the event that a portion of information 58 (e.g., a text-based customer feedback message) that was routed to location node 116 includes any of the words defined within good location word list 156, that portion of information 58 may be considered to be text-based customer feedback indicative of store 62 being in a good location (and, therefore, may be routed to good location node 154).

[0061] Bad location branch 152 may lead to bad location node 158, which may be configured to process the portion of information 58 (e.g., unstructured text-based customer feedback) that concerns (in whole or in part) bad feedback concerning the location of store 62. For example, bad location node 158 may define bad location word list 160 that may include words indicative of store 62 being in a bad location. Accordingly and in the event that a portion of information 58 (e.g., a text-based customer feedback message) that was routed to location node 116 includes any of the words defined within bad location word list 160, that portion of information 58 may be considered to be text-based customer feedback indicative of store 62 being in a bad location (and, therefore, may be routed to bad location node 158).

[0062] As stated above, value branch 110 may lead to value node 118, which may be configured to process the portion of information 58 (e.g., unstructured text-based customer feedback) that concerns (in whole or in part) feedback concerning the value received at store 62. For example, value node 118 may define value word list 162 that may include e.g., words indicative of the value received at store 62. Accordingly and in the event that a portion of information 58 (e.g., a text-based customer feedback message) includes any of the words defined within value word list 162, that portion of information 58 may be considered to be text-based customer feedback concerning the value received at store 62 and (therefore) may be routed to value node 118 of probabilistic model 100 for

further processing. Assume for this illustrative example that probabilistic model 100 includes two branches off of value node 118, namely: good value branch 164 and bad value branch 166.

[0063] Good value branch 164 may lead to good value node 168, which may be configured to process the portion of information 58 (e.g., unstructured text-based customer feedback) that concerns (in whole or in part) good value being received at store 62. For example, good value node 168 may define good value word list 170 that may include words indicative of receiving good value at store 62. Accordingly and in the event that a portion of information 58 (e.g., a text-based customer feedback message) that was routed to value node 118 includes any of the words defined within good value word list 170, that portion of information 58 may be considered to be text-based customer feedback indicative of good value being received at store 62 (and, therefore, may be routed to good value node 168).

[0064] Bad value branch 166 may lead to bad value node 172, which may be configured to process the portion of information 58 (e.g., unstructured text-based customer feedback) that concerns (in whole or in part) bad value being received at store 62. For example, bad value node 172 may define bad value word list 174 that may include words indicative of receiving bad value at store 62. Accordingly and in the event that a portion of information 58 (e.g., a text-based customer feedback message) that was routed to value node 118 includes any of the words defined within bad value word list 174, that portion of information 58 may be considered to be text-based customer feedback indicative of bad value being received at store 62 (and, therefore, may be routed to bad value node 172).

[0065] Once it is established that good or bad customer feedback was received concerning store 62 (i.e., with respect to the service, the selection, the location or the value), representatives and/or agents of store 62 may address the provider of such good or bad feedback via e.g., social media postings, text-messages and/or personal contact.

[0066] Assume for illustrative purposes that user 36 uses data-enabled, cellular telephone 28 to provide feedback 64 (e.g., a portion of information 58) to an automated feedback phone line concerning store 62. Upon receiving feedback 64 for analysis, probabilistic process 56 may identify any pertinent content that is included within feedback 64.

[0067] For illustrative purposes, assume that user 36 was not happy with their experience at store 62 and that feedback 64 provided by user 36 was "my cashier was rude and the weather was rainy". Accordingly and for this example, probabilistic process 56 may identify the pertinent content (included within feedback 64) as the phrase "my cashier was rude" and may ignore / remove the irrelevant content "the weather was rainy". As (in this example) feedback 64 includes the word "cashier", probabilistic process 56 may route feedback 64 to service node 112 via service branch 104. Further, as feedback 64 also includes the word "rude", probabilistic process 56 may route feedback 64 to bad service node 130 via bad service branch 124 and may consider feedback 64 to be text-based customer feedback indicative of bad service being received at store 62.

[0068] For further illustrative purposes, assume that user 36 was happy with their experience at store 62 and that feedback 64 provided by user 36 was "the clothing I purchased was classy but my cab got stuck in traffic". Accordingly and for this example, probabilistic process 56 may identify the pertinent content (included within feedback 64) as the phrase "the clothing I purchased was classy" and may ignore / remove the irrelevant content "my cab got stuck in traffic". As (in this example) feedback 64 includes the word "clothing", probabilistic process 56 may route feedback 64 to selection node 114 via selection branch 106. Further, as feedback 64 also includes the word "classy", probabilistic process 56 may route feedback 64 to good selection node 140 via good selection branch 136 and may consider feedback 64 to be text-based customer feedback indicative of a good selection being available at store 62.

Model Generation Overview:

[0069] While the following discussion concerns the automated generation of a probabilistic model, this is for illustrative purposes only and is not intended to be a limitation of this disclosure, as other configurations are possible and are considered to be within the scope of this disclosure. For example, the following discussion of automated generation may be utilized on any type of model. For example, the following discussion may be applicable to any other form of probabilistic model or any form of generic model (such as Dempster Shaffer theory or fuzzy logic).

[0070] As discussed above, probabilistic model 100 may be utilized to categorize information 58, thus allowing the various messages included within information 58 to be routed to (in this simplified example) one of eight nodes (e.g., good service node 126, bad service node 130, good selection node 140, bad selection node 144, good location node 154, bad location node 158, good value node 168, and bad value node 172). For the following example, assume that store 62 is a long-standing and well-established shopping establishment. Further, assume that information 58 is a very large quantity of voice mail messages (>10,000 messages) that were left by customers of store 62 on a voice-based customer feedback line. Additionally, assume that this very large quantity of voice mail messages (>10,000) have been transcribed into a very large quantity of text-based messages (>10,000).

[0071] Probabilistic process 56 may be configured to automatically define probabilistic model 100 based upon information 58. Accordingly, probabilistic process 56 may receive content (e.g., a very large quantity of text-based messages) and may be configured to define one or more probabilistic model variables for probabilistic model 100. For example, probabilistic process 56 may be configured to allow a user to specify such probabilistic model variables. Another example of such variables may include but is not limited to values and/or ranges of values for a data flow variable. For the following

discussion and for this disclosure, examples of a "variable" may include but are not limited to variables, parameters, ranges, branches and nodes.

[0072] Specifically and for this example, assume that probabilistic process 56 defines the initial number of branches (i.e., the number of branches off of branching node 102) within probabilistic model 100 as four (i.e., service branch 104, selection branch 106, location branch 108 and value branch 110). The defining of the initial number of branches (i.e., the number of branches off of branching node 102) within probabilistic model 100 as four may be effectuated in various ways (e.g., manually or algorithmically). Further and when defining probabilistic model 100 based, at least in part, upon information 58 and the one or more model variables (i.e., defining the number of branches off of branching node 102 as four), probabilistic process 56 may process information 58 to identify the pertinent content included within information 58. As discussed above, probabilistic process 56 may identify the pertinent content (included within information 58) and may ignore / remove the irrelevant content.

[0073] This type of processing of information 58 may continue for all of the very large quantity of text-based messages (>10,000) included within information 58. And using the probabilistic modeling technique described above, probabilistic process 56 may define a first version of the probabilistic model (e.g., probabilistic model 100) based, at least in part, upon pertinent content found within information 58. Accordingly, a first text-based message included within information 58 may be processed to extract pertinent information from that first message, wherein this pertinent information may be grouped in a manner to correspond (at least temporarily) with the requirement that four branches originate from branching node 102 (as defined above).

[0074] As probabilistic process 56 continues to process information 58 to identify pertinent content included within information 58, probabilistic process 56 may identify patterns within these text-based message included within information 58. For example, the messages may all concern one or more of the service, the selection, the location

and/or the value of store 62. Further and e.g., using the probabilistic modeling technique described above, probabilistic process 56 may process information 58 to e.g.: a) sort text-based messages concerning the service into positive or negative service messages; b) sort text-based messages concerning the selection into positive or negative selection messages; c) sort text-based messages concerning the location into positive or negative location messages; and/or d) sort text-based messages concerning the value into positive or negative service messages. For example, probabilistic process 56 may define various lists (e.g., lists 128, 132, 142, 146, 156, 160, 170, 174) by starting with a root word (e.g., good or bad) and may then determine synonyms for these words and use those words and synonyms to populate lists 128, 132, 142, 146, 156, 160, 170, 174.

[0075] Continuing with the above-stated example, once information 58 (or a portion thereof) is processed by probabilistic process 56, probabilistic process 56 may define a first version of the probabilistic model (e.g., probabilistic model 100) based, at least in part, upon pertinent content found within information 58. Probabilistic process 56 may compare the first version of the probabilistic model (e.g., probabilistic model 100) to information 58 to determine if the first version of the probabilistic model (e.g., probabilistic model 100) is a good explanation of the content.

[0076] When determining if the first version of the probabilistic model (e.g., probabilistic model 100) is a good explanation of the content, probabilistic process 56 may use an ML algorithm to fit the first version of the probabilistic model (e.g., probabilistic model 100) to the content, wherein examples of such an ML algorithm may include but are not limited to one or more of: an inferencing algorithm, a learning algorithm, an optimization algorithm, and a statistical algorithm.

[0077] For example and as is known in the art, probabilistic model 100 may be used to generate messages (in addition to analyzing them). For example and when defining a first version of the probabilistic model (e.g., probabilistic model 100) based, at least in part, upon pertinent content found within information 58, probabilistic process 56 may

define a weight for each branch within probabilistic model 100 based upon information 58. For example, threat mitigation process 10 may equally weight each of branches 104, 106, 108, 110 at 25%. Alternatively, if e.g., a larger percentage of information 58 concerned the service received at store 62, threat mitigation process 10 may equally weight each of branches 106, 108, 110 at 20%, while more heavily weighting branch 104 at 40%.

[0078] Accordingly and when probabilistic process 56 compares the first version of the probabilistic model (e.g., probabilistic model 100) to information 58 to determine if the first version of the probabilistic model (e.g., probabilistic model 100) is a good explanation of the content, probabilistic process 56 may generate a very large quantity of messages e.g., by auto-generating messages using the above-described probabilities, the above-described nodes & node types, and the words defined in the above-described lists (e.g., lists 128, 132, 142, 146, 156, 160, 170, 174), thus resulting in generated information 58'. Generated information 58' may then be compared to information 58 to determine if the first version of the probabilistic model (e.g., probabilistic model 100) is a good explanation of the content. For example, if generated information 58' exceeds a threshold level of similarity to information 58, the first version of the probabilistic model (e.g., probabilistic model 100) may be deemed a good explanation of the content. Conversely, if generated information 58' does not exceed a threshold level of similarity to information 58, the first version of the probabilistic model (e.g., probabilistic model 100) may be deemed not a good explanation of the content.

[0079] If the first version of the probabilistic model (e.g., probabilistic model 100) is not a good explanation of the content, probabilistic process 56 may define a revised version of the probabilistic model (e.g., revised probabilistic model 100'). When defining revised probabilistic model 100', probabilistic process 56 may e.g., adjust weighting, adjust probabilities, adjust node counts, adjust node types, and/or adjust branch counts to define the revised version of the probabilistic model (e.g., revised

probabilistic model 100'). Once defined, the above-described process of auto-generating messages (this time using revised probabilistic model 100') may be repeated and this newly-generated content (e.g., generated information 58") may be compared to information 58 to determine if e.g., revised probabilistic model 100' is a good explanation of the content. If revised probabilistic model 100' is not a good explanation of the content, the above-described process may be repeated until a proper probabilistic model is defined.

The Threat Mitigation Process

[0080] As discussed above, threat mitigation process 10 may include probabilistic process 56 (e.g., an artificial intelligence / machine learning process) that may be configured to process information (e.g., information 58), wherein examples of information 58 may include but are not limited to platform information (e.g., structured or unstructured content) that may be scanned to detect security events (e.g., access auditing; anomalies; authentication; denial of services; exploitation; malware; phishing; spamming; reconnaissance; and/or web attack) within a monitored computing platform (e.g., computing platform 60).

[0081] Referring also to FIG. 3, the monitored computing platform (e.g., computing platform 60) utilized by business today may be a highly complex, multi-location computing system / network that may span multiple buildings / locations / countries. For this illustrative example, the monitored computing platform (e.g., computing platform 60) is shown to include many discrete computing devices, examples of which may include but are not limited to: server computers (e.g., server computers 200, 202), desktop computers (e.g., desktop computer 204), and laptop computers (e.g., laptop computer 206), all of which may be coupled together via a network (e.g., network 208), such as an Ethernet network. Computing platform 60 may be coupled to an external network (e.g., Internet 210) through WAF (i.e., Web Application Firewall) 212. A wireless access point (e.g., WAP 214) may be configured to allow wireless devices (e.g., smartphone 216) to

access computing platform 60. Computing platform 60 may include various connectivity devices that enable the coupling of devices within computing platform 60, examples of which may include but are not limited to: switch 216, router 218 and gateway 220. Computing platform 60 may also include various storage devices (e.g., NAS 222), as well as functionality (e.g., API Gateway 224) that allows software applications to gain access to one or more resources within computing platform 60.

[0082] In addition to the devices and functionality discussed above, other technology (e.g., security-relevant subsystems 226) may be deployed within computing platform 60 to monitor the operation of (and the activity within) computing platform 60. Examples of security-relevant subsystems 226 may include but are not limited to: CDN (i.e., Content Delivery Network) systems; DAM (i.e., Database Activity Monitoring) systems; UBA (i.e., User Behavior Analytics) systems; MDM (i.e., Mobile Device Management) systems; IAM (i.e., Identity and Access Management) systems; DNS (i.e., Domain Name Server) systems, antivirus systems, operating systems, data lakes; data logs; security-relevant software applications; security-relevant hardware systems; and resources external to the computing platform.

[0083] Each of security-relevant subsystems 226 may monitor and log their activity with respect to computing platform 60, resulting in the generation of platform information 228. For example, platform information 228 associated with a client-defined MDM (i.e., Mobile Device Management) system may monitor and log the mobile devices that were allowed access to computing platform 60.

[0084] Further, SEIM (i.e., Security Information and Event Management) system 230 may be deployed within computing platform 60. As is known in the art, SIEM system 230 is an approach to security management that combines SIM (security information management) functionality and SEM (security event management) functionality into one security management system. The underlying principles of a SIEM system is to aggregate relevant data from multiple sources, identify deviations from the norm and take

appropriate action. For example, when a security event is detected, SIEM system 230 might log additional information, generate an alert and instruct other security controls to mitigate the security event. Accordingly, SIEM system 230 may be configured to monitor and log the activity of security-relevant subsystems 226 (e.g., CDN (i.e., Content Delivery Network) systems; DAM (i.e., Database Activity Monitoring) systems; UBA (i.e., User Behavior Analytics) systems; MDM (i.e., Mobile Device Management) systems; IAM (i.e., Identity and Access Management) systems; DNS (i.e., Domain Name Server) systems, antivirus systems, operating systems, data lakes; data logs; security-relevant software applications; security-relevant hardware systems; and resources external to the computing platform).

Computing Platform Analysis & Reporting

[0085] As will be discussed below in greater detail, threat mitigation process 10 may be configured to e.g., analyze computing platform 60 and provide reports to third-parties concerning the same.

[0086] Referring also to FIGS. 4-6, threat mitigation process 10 may be configured to obtain and combine information from multiple security-relevant subsystem to generate a security profile for computing platform 60. For example, threat mitigation process 10 may obtain 300 first system-defined platform information (e.g., system-defined platform information 232) concerning a first security-relevant subsystem (e.g., the number of operating systems deployed) within computing platform 60 and may obtain 302 at least a second system-defined platform information (e.g., system-defined platform information 234) concerning at least a second security-relevant subsystem (e.g., the number of antivirus systems deployed) within computing platform 60.

[0087] The first system-defined platform information (e.g., system-defined platform information 232) and the at least a second system-defined platform information (e.g., system-defined platform information 234) may be obtained from one or more log files defined for computing platform 60.

[0088] Specifically, system-defined platform information 232 and/or system-defined platform information 234 may be obtained from SIEM system 230, wherein (and as discussed above) SIEM system 230 may be configured to monitor and log the activity of security-relevant subsystems 226 (e.g., CDN (i.e., Content Delivery Network) systems; DAM (i.e., Database Activity Monitoring) systems; UBA (i.e., User Behavior Analytics) systems; MDM (i.e., Mobile Device Management) systems; IAM (i.e., Identity and Access Management) systems; DNS (i.e., Domain Name Server) systems, antivirus systems, operating systems, data lakes; data logs; security-relevant software applications; security-relevant hardware systems; and resources external to the computing platform).

[0089] Alternatively, the first system-defined platform information (e.g., system-defined platform information 232) and the at least a second system-defined platform information (e.g., system-defined platform information 234) may be obtained from the first security-relevant subsystem (e.g., the operating systems themselves) and the at least a second security-relevant subsystem (e.g., the antivirus systems themselves). Specifically, system-defined platform information 232 and/or system-defined platform information 234 may be obtained directly from the security-relevant subsystems (e.g., the operating systems and/or the antivirus systems), which (as discussed above) may be configured to self-document their activity.

[0090] Threat mitigation process 10 may combine 308 the first system-defined platform information (e.g., system-defined platform information 232) and the at least a second system-defined platform information (e.g., system-defined platform information 234) to form system-defined consolidated platform information 236. Accordingly and in this example, system-defined consolidated platform information 236 may independently define the security-relevant subsystems (e.g., security-relevant subsystems 226) present on computing platform 60.

[0091] Threat mitigation process 10 may generate 310 a security profile (e.g., security profile 350) based, at least in part, upon system-defined consolidated platform

information 236. Through the use of security profile (e.g., security profile 350), the user / owner / operator of computing platform 60 may be able to see that e.g., they have a security score of 605 out of a possible score of 1,000, wherein the average customer has a security score of 237. While security profile 350 in shown in the example to include several indicators that may enable a user to compare (in this example) computing platform 60 to other computing platforms, this is for illustrative purposes only and is not intended to be a limitation of this disclosure, as it is understood that other configurations are possible and are considered to be within the scope of this disclosure.

[0092] Naturally, the format, appearance and content of security profile 350 may be varied greatly depending upon the design criteria and anticipated performance / use of threat mitigation process 10. Accordingly, the appearance, format, completeness and content of security profile 350 is for illustrative purposes only and is not intended to be a limitation of this disclosure, as other configurations are possible and are considered to be within the scope of this disclosure. For example, content may be added to security profile 350, removed from security profile 350, and/or reformatted within security profile 350.

[0093] Additionally, threat mitigation process 10 may obtain 312 client-defined consolidated platform information 238 for computing platform 60 from a client information source, examples of which may include but are not limited to one or more client-completed questionnaires (e.g., questionnaires 240) and/or one or more client-deployed platform monitors (e.g., client-deployed platform monitor 242, which may be configured to effectuate SIEM functionality). Accordingly and in this example, client-defined consolidated platform information 238 may define the security-relevant subsystems (e.g., security-relevant subsystems 226) that the client believes are present on computing platform 60.

[0094] When generating 310 a security profile (e.g., security profile 350) based, at least in part, upon system-defined consolidated platform information 236, threat

mitigation process 10 may compare 314 the system-defined consolidated platform information (e.g., system-defined consolidated platform information 236) to the client-defined consolidated platform information (e.g., client-defined consolidated platform information 238) to define differential consolidated platform information 352 for computing platform 60.

[0095] Differential consolidated platform information 352 may include comparison table 354 that e.g., compares computing platform 60 to other computing platforms. For example and in this particular implementation of differential consolidated platform information 352, comparison table 354 is shown to include three columns, namely: security-relevant subsystem column 356 (that identifies the security-relevant subsystems in question); system-defined consolidated platform information column 358 (that is based upon system-defined consolidated platform information 236 and independently defines what security-relevant subsystems are present on computing platform 60); and client-defined consolidated platform column 360 (that is based upon client-defined platform information 238 and defines what security-relevant subsystems the client believes are present on computing platform 60). As shown within comparison table 354, there are considerable differences between that is actually present on computing platform 60 and what is believed to be present on computing platform 60 (e.g., 1 IAM system vs. 10 IAM systems; 4,000 operating systems vs. 10,000 operating systems, 6 DNS systems vs. 10 DNS systems; 0 antivirus systems vs. 1 antivirus system, and 90 firewalls vs. 150 firewalls).

[0096] Naturally, the format, appearance and content of differential consolidated platform information 352 may be varied greatly depending upon the design criteria and anticipated performance / use of threat mitigation process 10. Accordingly, the appearance, format, completeness and content of differential consolidated platform information 352 is for illustrative purposes only and is not intended to be a limitation of this disclosure, as other configurations are possible and are considered to be within the

scope of this disclosure. For example, content may be added to differential consolidated platform information 352, removed from differential consolidated platform information 352, and/or reformatted within differential consolidated platform information 352.

[0097] Referring also to FIG. 7, threat mitigation process 10 may be configured to compare what security relevant subsystems are actually included within computing platform 60 versus what security relevant subsystems were believed to be included within computing platform 60. As discussed above, threat mitigation process 10 may combine 308 the first system-defined platform information (e.g., system-defined platform information 232) and the at least a second system-defined platform information (e.g., system-defined platform information 234) to form system-defined consolidated platform information 236.

[0098] Threat mitigation process 10 may obtain 400 system-defined consolidated platform information 236 for computing platform 60 from an independent information source, examples of which may include but are not limited to: one or more log files defined for computing platform 60 (e.g., such as those maintained by SIEM system 230); and two or more security-relevant subsystems (e.g., directly from the operating system security-relevant subsystem and the antivirus security-relevant subsystem) deployed within computing platform 60.

[0099] Further and as discussed above, threat mitigation process 10 may obtain 312 client-defined consolidated platform information 238 for computing platform 60 from a client information source, examples of which may include but are not limited to one or more client-completed questionnaires (e.g., questionnaires 240) and/or one or more client-deployed platform monitors (e.g., client-deployed platform monitor 242, which may be configured to effectuate SIEM functionality).

[00100] Additionally and as discussed above, threat mitigation process 10 may compare 402 system-defined consolidated platform information 236 to client-defined consolidated platform information 238 to define differential consolidated platform

information 352 for computing platform 60, wherein differential consolidated platform information 352 may include comparison table 354 that e.g., compares computing platform 60 to other computing platforms.

[00101] Threat mitigation process 10 may process 404 system-defined consolidated platform information 236 prior to comparing 402 system-defined consolidated platform information 236 to client-defined consolidated platform information 238 to define differential consolidated platform information 352 for computing platform 60. Specifically, threat mitigation process 10 may process 404 system-defined consolidated platform information 236 so that it is comparable to client-defined consolidated platform information 238.

[00102] For example and when processing 404 system-defined consolidated platform information 236, threat mitigation process 10 may homogenize 406 system-defined consolidated platform information 236 prior to comparing 402 system-defined consolidated platform information 236 to client-defined consolidated platform information 238 to define differential consolidated platform information 352 for computing platform 60. Such homogenization 406 may result in system-defined consolidated platform information 236 and client-defined consolidated platform information 238 being comparable to each other (e.g., to accommodate for differing data nomenclatures / headers).

[00103] Further and when processing 404 system-defined consolidated platform information 236, threat mitigation process 10 may normalize 408 system-defined consolidated platform information 236 prior to comparing 402 system-defined consolidated platform information 236 to client-defined consolidated platform information 238 to define differential consolidated platform information 352 for computing platform 60 (e.g., to accommodate for data differing scales / ranges).

[00104] Referring also to FIG. 8, threat mitigation process 10 may be configured to compare what security relevant subsystems are actually included within computing

platform 60 versus what security relevant subsystems were believed to be included within computing platform 60.

[00105] As discussed above, threat mitigation process 10 may obtain 400 system-defined consolidated platform information 236 for computing platform 60 from an independent information source, examples of which may include but are not limited to: one or more log files defined for computing platform 60 (e.g., such as those maintained by SIEM system 230); and two or more security-relevant subsystems (e.g., directly from the operating system security-relevant subsystem and the antivirus security-relevant subsystem) deployed within computing platform 60

[00106] Further and as discussed above, threat mitigation process 10 may obtain 312 client-defined consolidated platform information 238 for computing platform 60 from a client information source, examples of which may include but are not limited to one or more client-completed questionnaires (e.g., questionnaires 240) and/or one or more client-deployed platform monitors (e.g., client-deployed platform monitor 242, which may be configured to effectuate SIEM functionality).

[00107] Threat mitigation process 10 may present 450 differential consolidated platform information 352 for computing platform 60 to a third-party, examples of which may include but are not limited to the user / owner / operator of computing platform 60.

[00108] Additionally and as discussed above, threat mitigation process 10 may compare 402 system-defined consolidated platform information 236 to client-defined consolidated platform information 238 to define differential consolidated platform information 352 for computing platform 60, wherein differential consolidated platform information 352 may include comparison table 354 that e.g., compares computing platform 60 to other computing platforms, wherein (and as discussed above) threat mitigation process 10 may process 404 (e.g., via homogenizing 406 and/or normalizing 408) system-defined consolidated platform information 236 prior to comparing 402 system-defined consolidated platform information 236 to client-defined consolidated

platform information 236 to define differential consolidated platform information 352 for computing platform 60.

Computing Platform Analysis & Recommendation

[00109] As will be discussed below in greater detail, threat mitigation process 10 may be configured to e.g., analyze & display the vulnerabilities of computing platform 60.

[00110] Referring also to FIG. 9, threat mitigation process 10 may be configured to make recommendations concerning security relevant subsystems that are missing from computing platform 60. As discussed above, threat mitigation process 10 may obtain 500 consolidated platform information for computing platform 60 to identify one or more deployed security-relevant subsystems 226 (e.g., CDN (i.e., Content Delivery Network) systems; DAM (i.e., Database Activity Monitoring) systems; UBA (i.e., User Behavior Analytics) systems; MDM (i.e., Mobile Device Management) systems; IAM (i.e., Identity and Access Management) systems; DNS (i.e., Domain Name Server) systems, antivirus systems, operating systems, data lakes; data logs; security-relevant software applications; security-relevant hardware systems; and resources external to the computing platform). This consolidated platform information may be obtained from an independent information source (e.g., such as SIEM system 230 that may provide system-defined consolidated platform information 236) and/or may be obtained from a client information source (e.g., such as questionnaires 240 that may provide client-defined consolidated platform information 238).

[00111] Referring also to FIG. 10, threat mitigation process 10 may process 506 the consolidated platform information (e.g., system-defined consolidated platform information 236 and/or client-defined consolidated platform information 238) to identify one or more non-deployed security-relevant subsystems (within computing platform 60) and may then generate 508 a list of ranked & recommended security-relevant subsystems

(e.g., non-deployed security-relevant subsystem list 550) that ranks the one or more non-deployed security-relevant subsystems.

[00112] For this particular illustrative example, non-deployed security-relevant subsystem list 550 is shown to include column 552 that identifies six non-deployed security-relevant subsystems, namely: a CDN subsystem, a WAF subsystem, a DAM subsystem; a UBA subsystem; an API subsystem, and an MDM subsystem.

[00113] When generating 508 a list of ranked & recommended security-relevant subsystems (e.g., non-deployed security-relevant subsystem list 550) that ranks the one or more non-deployed security-relevant subsystems, threat mitigation process 10 may rank 510 the one or more non-deployed security-relevant subsystems (e.g., a CDN subsystem, a WAF subsystem, a DAM subsystem; a UBA subsystem; a API subsystem, and an MDM subsystem) based upon the anticipated use of the one or more non-deployed security-relevant subsystems within computing platform 60. This ranking 510 of the non-deployed security-relevant subsystems (e.g., a CDN subsystem, a WAF subsystem, a DAM subsystem; a UBA subsystem; a API subsystem, and an MDM subsystem) may be agnostic in nature and may be based on the functionality / effectiveness of the non-deployed security-relevant subsystems and the anticipated manner in which their implementation may impact the functionality / security of computing platform 60.

[00114] Threat mitigation process 10 may provide 512 the list of ranked & recommended security-relevant subsystems (e.g., non-deployed security-relevant subsystem list 550) to a third-party, examples of which may include but are not limited to a user / owner / operator of computing platform 60.

[00115] Additionally, threat mitigation process 10 may identify 514 a comparative for at least one of the non-deployed security-relevant subsystems (e.g., a CDN subsystem, a WAF subsystem, a DAM subsystem; a UBA subsystem; an API subsystem, and an MDM subsystem) defined within the list of ranked & recommended security-relevant subsystems (e.g., non-deployed security-relevant subsystem list 550). This comparative

may include vendor customers in a specific industry comparative and/or vendor customers in any industry comparative.

[00116] For example and in addition to column 552, non-deployed security-relevant subsystem list 550 may include columns 554, 556 for defining the comparatives for the six non-deployed security-relevant subsystems, namely: a CDN subsystem, a WAF subsystem, a DAM subsystem; a UBA subsystem; an API subsystem, and an MDM subsystem. Specifically, column 554 is shown to define comparatives concerning vendor customers that own the non-deployed security-relevant subsystems in a specific industry (i.e., the same industry as the user / owner / operator of computing platform 60). Additionally, column 556 is shown to define comparatives concerning vendor customers that own the non-deployed security-relevant subsystems in any industry (i.e., not necessarily the same industry as the user / owner / operator of computing platform 60). For example and concerning the comparatives of the WAF subsystem: 33% of the vendor customers in the same industry as the user / owner / operator of computing platform 60 deploy a WAF subsystem; while 71% of the vendor customers in any industry deploy a WAF subsystem.

[00117] Naturally, the format, appearance and content of non-deployed security-relevant subsystem list 550 may be varied greatly depending upon the design criteria and anticipated performance / use of threat mitigation process 10. Accordingly, the appearance, format, completeness and content of non-deployed security-relevant subsystem list 550 is for illustrative purposes only and is not intended to be a limitation of this disclosure, as other configurations are possible and are considered to be within the scope of this disclosure. For example, content may be added to non-deployed security-relevant subsystem list 550, removed from non-deployed security-relevant subsystem list 550, and/or reformatted within non-deployed security-relevant subsystem list 550.

[00118] Referring also to FIG. 11, threat mitigation process 10 may be configured to compare the current capabilities to the possible capabilities of computing platform 60.

As discussed above, threat mitigation process 10 may obtain 600 consolidated platform information to identify current security-relevant capabilities for computing platform 60. This consolidated platform information may be obtained from an independent information source (e.g., such as SIEM system 230 that may provide system-defined consolidated platform information 236) and/or may be obtained from a client information source (e.g., such as questionnaires 240 that may provide client-defined consolidated platform information 238. Threat mitigation process 10 may then determine 606 possible security-relevant capabilities for computing platform 60 (i.e., the difference between the current security-relevant capabilities of computing platform 60 and the possible security-relevant capabilities of computing platform 60. For example, the possible security-relevant capabilities may concern the possible security-relevant capabilities of computing platform 60 using the currently-deployed security-relevant subsystems. Additionally / alternatively, the possible security-relevant capabilities may concern the possible security-relevant capabilities of computing platform 60 using one or more supplemental security-relevant subsystems.

[00119] Referring also to FIG. 12 and as will be explained below, threat mitigation process 10 may generate 608 comparison information 650 that compares the current security-relevant capabilities of computing platform 60 to the possible security-relevant capabilities of computing platform 60 to identify security-relevant deficiencies. Comparison information 650 may include graphical comparison information, such as multi-axial graphical comparison information that simultaneously illustrates a plurality of security-relevant deficiencies.

[00120] For example, comparison information 650 may define (in this particular illustrative example) graphical comparison information that include five axes (e.g. axes 652, 654, 656, 658, 660) that correspond to five particular types of computer threats. Comparison information 650 includes origin 662, the point at which computing platform 60 has no protection with respect to any of the five types of computer threats that correspond to axes 652, 654, 656, 658, 660. Accordingly, as the capabilities of computing platform 60 are increased to counter a particular type of computer threat, the data point along the corresponding axis is proportionately displaced from origin 652.

[00121] As discussed above, threat mitigation process 10 may obtain 600 consolidated platform information to identify current security-relevant capabilities for computing platform 60. Concerning such current security-relevant capabilities for computing platform 60, these current security-relevant capabilities are defined by data points 664, 666, 668, 670, 672, the combination of which define bounded area 674. Bounded area 674 (in this example) defines the current security-relevant capabilities of computing platform 60.

[00122] Further and as discussed above, threat mitigation process 10 may determine 606 possible security-relevant capabilities for computing platform 60 (i.e., the difference between the current security-relevant capabilities of computing platform 60 and the possible security-relevant capabilities of computing platform 60.

[00123] As discussed above, the possible security-relevant capabilities may concern the possible security-relevant capabilities of computing platform 60 using the currently-deployed security-relevant subsystems. For example, assume that the currently-deployed security relevant subsystems are not currently being utilized to their full potential. Accordingly, certain currently-deployed security relevant subsystems may have certain features that are available but are not utilized and/or disabled. Further, certain currently-deployed security relevant subsystems may have expanded features available if additional licensing fees are paid. Therefore and concerning such possible

security-relevant capabilities of computing platform 60 using the currently-deployed security-relevant subsystems, data points 676, 678, 680, 682, 684 may define bounded area 686 (which represents the full capabilities of the currently-deployed security-relevant subsystems within computing platform 60).

[00124] Further and as discussed above, the possible security-relevant capabilities may concern the possible security-relevant capabilities of computing platform 60 using one or more supplemental security-relevant subsystems. For example, assume that supplemental security-relevant subsystems are available for the deployment within computing platform 60. Therefore and concerning such possible security-relevant capabilities of computing platform 60 using such supplemental security-relevant subsystems, data points 688, 690, 692, 694, 696 may define bounded area 698 (which represents the total capabilities of computing platform 60 when utilizing the full capabilities of the currently-deployed security-relevant subsystems and any supplemental security-relevant subsystems).

[00125] Naturally, the format, appearance and content of comparison information 650 may be varied greatly depending upon the design criteria and anticipated performance / use of threat mitigation process 10. Accordingly, the appearance, format, completeness and content of comparison information 650 is for illustrative purposes only and is not intended to be a limitation of this disclosure, as other configurations are possible and are considered to be within the scope of this disclosure. For example, content may be added to comparison information 650, removed from comparison information 650, and/or reformatted within comparison information 650.

[00126] Referring also to FIG. 13, threat mitigation process 10 may be configured to generate a threat context score for computing platform 60. As discussed above, threat mitigation process 10 may obtain 600 consolidated platform information to identify current security-relevant capabilities for computing platform 60. This consolidated platform information may be obtained from an independent information source (e.g., such

as SIEM system 230 that may provide system-defined consolidated platform information 236) and/or may be obtained from a client information source (e.g., such as questionnaires 240 that may provide client-defined consolidated platform information 238. As will be discussed below in greater detail, threat mitigation process 10 may determine 700 comparative platform information that identifies security-relevant capabilities for a comparative platform, wherein this comparative platform information may concern vendor customers in a specific industry (i.e., the same industry as the user / owner / operator of computing platform 60) and/or vendor customers in any industry (i.e., not necessarily the same industry as the user / owner / operator of computing platform 60).

[00127] Referring also to FIG. 14 and as will be discussed below, threat mitigation process 10 may generate 702 comparison information 750 that compares the current security-relevant capabilities of computing platform 60 to the comparative platform information determined 700 for the comparative platform to identify a threat context indicator for computing platform 60, wherein comparison information 750 may include graphical comparison information 752.

[00128] Graphical comparison information 752 (which in this particular example is a bar chart) may identify one or more of: a current threat context score 754 for a client (e.g., the user / owner / operator of computing platform 60); a maximum possible threat context score 756 for the client (e.g., the user / owner / operator of computing platform 60); a threat context score 758 for one or more vendor customers in a specific industry (i.e., the same industry as the user / owner / operator of computing platform 60); and a threat context score 760 for one or more vendor customers in any industry (i.e., not necessarily the same industry as the user / owner / operator of computing platform 60).

[00129] Naturally, the format, appearance and content of comparison information 750 may be varied greatly depending upon the design criteria and anticipated performance / use of threat mitigation process 10. Accordingly, the appearance, format,

completeness and content of comparison information 750 is for illustrative purposes only and is not intended to be a limitation of this disclosure, as other configurations are possible and are considered to be within the scope of this disclosure. For example, content may be added to comparison information 750, removed from comparison information 750, and/or reformatted within comparison information 750.

Computing Platform Monitoring & Mitigation

[00130] As will be discussed below in greater detail, threat mitigation process 10 may be configured to e.g., monitor the operation and performance of computing platform 60.

[00131] Referring also to FIG. 15, threat mitigation process 10 may be configured to monitor the health of computing platform 60 and provide feedback to a third-party concerning the same. Threat mitigation process 10 may obtain 800 hardware performance information 244 concerning hardware (e.g., server computers, desktop computers, laptop computers, switches, firewalls, routers, gateways, WAPs, and NASs), deployed within computing platform 60. Hardware performance information 244 may concern the operation and/or functionality of one or more hardware systems (e.g., server computers, desktop computers, laptop computers, switches, firewalls, routers, gateways, WAPs, and NASs) deployed within computing platform 60.

[00132] Threat mitigation process 10 may obtain 802 platform performance information 246 concerning the operation of computing platform 60. Platform performance information 246 may concern the operation and/or functionality of computing platform 60.

[00133] When obtaining 802 platform performance information concerning the operation of computing platform 60, threat mitigation process 10 may (as discussed above): obtain 400 system-defined consolidated platform information 236 for computing platform 60 from an independent information source (e.g., SIEM system 230); obtain 312 client-defined consolidated platform information 238 for computing platform 60 from a

client information (e.g., questionnaires 240); and present 450 differential consolidated platform information 352 for computing platform 60 to a third- party, examples of which may include but are not limited to the user / owner / operator of computing platform 60.

[00134] When obtaining 802 platform performance information concerning the operation of computing platform 60, threat mitigation process 10 may (as discussed above): obtain 500 consolidated platform information for computing platform 60 to identify one or more deployed security-relevant subsystems 226 (e.g., CDN (i.e., Content Delivery Network) systems; DAM (i.e., Database Activity Monitoring) systems; UBA (i.e., User Behavior Analytics) systems; MDM (i.e., Mobile Device Management) systems; IAM (i.e., Identity and Access Management) systems; DNS (i.e., Domain Name Server) systems, antivirus systems, operating systems, data lakes; data logs; security-relevant software applications; security-relevant hardware systems; and resources external to the computing platform); process 506 the consolidated platform information (e.g., system-defined consolidated platform information 236 and/or client-defined consolidated platform information 238) to identify one or more non-deployed security-relevant subsystems (within computing platform 60); generate 508 a list of ranked & recommended security-relevant subsystems (e.g., non-deployed security-relevant subsystem list 550) that ranks the one or more non-deployed security-relevant subsystems; and provide 514 the list of ranked & recommended security-relevant subsystems (e.g., non-deployed security-relevant subsystem list 550) to a third- party, examples of which may include but are not limited to a user / owner / operator of computing platform 60.

[00135] When obtaining 802 platform performance information concerning the operation of computing platform 60, threat mitigation process 10 may (as discussed above): obtain 600 consolidated platform information to identify current security-relevant capabilities for the computing platform; determine 606 possible security-relevant capabilities for computing platform 60; and generate 608 comparison information 650

that compares the current security-relevant capabilities of computing platform 60 to the possible security-relevant capabilities of computing platform 60 to identify security-relevant deficiencies.

[00136] When obtaining 802 platform performance information concerning the operation of computing platform 60, threat mitigation process 10 may (as discussed above): obtain 600 consolidated platform information to identify current security-relevant capabilities for computing platform 60; determine 700 comparative platform information that identifies security-relevant capabilities for a comparative platform; and generate 702 comparison information 750 that compares the current security-relevant capabilities of computing platform 60 to the comparative platform information determined 700 for the comparative platform to identify a threat context indicator for computing platform 60.

[00137] Threat mitigation process 10 may obtain 804 application performance information 248 concerning one or more applications (e.g., operating systems, user applications, security application, and utility application) deployed within computing platform 60. Application performance information 248 may concern the operation and/or functionality of one or more software applications (e.g., operating systems, user applications, security application, and utility application) deployed within computing platform 60.

[00138] Referring also to FIG. 16, threat mitigation process 10 may generate 806 holistic platform report (e.g., holistic platform reports 850, 852) concerning computing platform 60 based, at least in part, upon hardware performance information 244, platform performance information 246 and application performance information 248. Threat mitigation process 10 may be configured to receive e.g., hardware performance information 244, platform performance information 246 and application performance information 248 at regular intervals (e.g., continuously, every minute, every ten minutes, etc.).

[00139] As illustrated, holistic platform reports 850, 852 may include various pieces of content such as e.g., thought clouds that identity topics / issues with respect to computing platform 60, system logs that memorialize identified issues within computing platform 60, data sources providing information to computing system 60, and so on. The holistic platform report (e.g., holistic platform reports 850, 852) may identify one or more known conditions concerning the computing platform; and threat mitigation process 10 may effectuate 808 one or more remedial operations concerning the one or more known conditions.

[00140] For example, assume that the holistic platform report (e.g., holistic platform reports 850, 852) identifies that computing platform 60 is under a DoS (i.e., Denial of Services) attack. In computing, a denial-of-service attack (DoS attack) is a cyber-attack in which the perpetrator seeks to make a machine or network resource unavailable to its intended users by temporarily or indefinitely disrupting services of a host connected to the Internet. Denial of service is typically accomplished by flooding the targeted machine or resource with superfluous requests in an attempt to overload systems and prevent some or all legitimate requests from being fulfilled.

[00141] In response to detecting such a DoS attack, threat mitigation process 10 may effectuate 808 one or more remedial operations. For example and with respect to such a DoS attack, threat mitigation process 10 may effectuate 808 e.g., a remedial operation that instructs WAF (i.e., Web Application Firewall) 212 to deny all incoming traffic from the identified attacker based upon e.g., protocols, ports or the originating IP addresses.

[00142] Threat mitigation process 10 may also provide 810 the holistic report (e.g., holistic platform reports 850, 852) to a third-party, examples of which may include but are not limited to a user / owner / operator of computing platform 60.

[00143] Naturally, the format, appearance and content of the holistic platform report (e.g., holistic platform reports 850, 852) may be varied greatly depending upon the

design criteria and anticipated performance / use of threat mitigation process 10. Accordingly, the appearance, format, completeness and content of the holistic platform report (e.g., holistic platform reports 850, 852) is for illustrative purposes only and is not intended to be a limitation of this disclosure, as other configurations are possible and are considered to be within the scope of this disclosure. For example, content may be added to the holistic platform report (e.g., holistic platform reports 850, 852), removed from the holistic platform report (e.g., holistic platform reports 850, 852), and/or reformatted within the holistic platform report (e.g., holistic platform reports 850, 852).

[00144] Referring also to FIG. 17, threat mitigation process 10 may be configured to monitor computing platform 60 for the occurrence of a security event and (in the event of such an occurrence) gather artifacts concerning the same. For example, threat mitigation process 10 may detect 900 a security event within computing platform 60 based upon identified suspect activity. Examples of such security events may include but are not limited to: DDoS events, DoS events, phishing events, spamming events, malware events, web attacks, and exploitation events.

[00145] When detecting 900 a security event (e.g., DDoS events, DoS events, phishing events, spamming events, malware events, web attacks, and exploitation events) within computing platform 60 based upon identified suspect activity, threat mitigation process 10 may monitor 902 a plurality of sources to identify suspect activity within computing platform 60.

[00146] For example, assume that threat mitigation process 10 detects 900 a security event within computing platform 60. Specifically, assume that threat mitigation process 10 is monitoring 902 a plurality of sources (e.g., the various log files maintained by SIEM system 230). And by monitoring 902 such sources, assume that threat mitigation process 10 detects 900 the receipt of inbound content (via an API) from a device having an IP address located in Uzbekistan; the subsequent opening of a port within WAF (i.e., Web Application Firewall) 212; and the streaming of content from a

computing device within computing platform 60 through that recently-opened port in WAF (i.e., Web Application Firewall) 212 and to a device having an IP address located in Moldova.

[00147] Upon detecting 900 such a security event within computing platform 60, threat mitigation process 10 may gather 904 artifacts (e.g., artifacts 250) concerning the above-described security event. When gathering 904 artifacts (e.g., artifacts 250) concerning the above-described security event, threat mitigation process 10 may gather 906 artifacts concerning the security event from a plurality of sources associated with the computing platform, wherein examples of such plurality of sources may include but are not limited to the various log files maintained by SIEM system 230, and the various log files directly maintained by the security-relevant subsystems.

[00148] Once the appropriate artifacts (e.g., artifacts 250) are gathered 904, threat mitigation process 10 may assign 908 a threat level to the above-described security event based, at least in part, upon the artifacts (e.g., artifacts 250) gathered 904.

[00149] When assigning 908 a threat level to the above-described security event, threat mitigation process 10 may assign 910 a threat level using artificial intelligence / machine learning. As discussed above and with respect to artificial intelligence / machine learning being utilized to process data sets, an initial probabilistic model may be defined, wherein this initial probabilistic model may be subsequently (e.g., iteratively or continuously) modified and revised, thus allowing the probabilistic models and the artificial intelligence systems (e.g., probabilistic process 56) to "learn" so that future probabilistic models may be more precise and may explain more complex data sets. As further discussed above, probabilistic process 56 may define an initial probabilistic model for accomplishing a defined task (e.g., the analyzing of information 58), wherein the probabilistic model may be utilized to go from initial observations about information 58 (e.g., as represented by the initial branches of a probabilistic model) to conclusions about information 58 (e.g., as represented by the leaves of a probabilistic model). Accordingly

and through the use of probabilistic process 56, massive data sets concerning security events may be processed so that a probabilistic model may be defined (and subsequently revised) to assign 910 a threat level to the above-described security event.

[00150] Once assigned 910 a threat level, threat mitigation process 10 may execute 912 a remedial action plan (e.., remedial action plan 252) based, at least in part, upon the assigned threat level.

[00151] For example and when executing 912 a remedial action plan, threat mitigation process 10 may allow 914 the above-described suspect activity to continue when e.g., threat mitigation process 10 assigns 908 a "low" threat level to the above-described security event (e.g., assuming that it is determined that the user of the local computing device is streaming video of his daughter's graduation to his parents in Moldova).

[00152] Further and when executing 912 a remedial action plan, threat mitigation process 10 may generate 916 a security event report (e.g., security event report 254) based, at least in part, upon the artifacts (e.g., artifacts 250) gathered 904; and provide 918 the security event report (e.g., security event report 254) to an analyst (e.g., analyst 256) for further review when e.g., threat mitigation process 10 assigns 908 a "moderate" threat level to the above-described security event (e.g., assuming that it is determined that while the streaming of the content is concerning, the content is low value and the recipient is not a known bad actor).

[00153] Further and when executing 912 a remedial action plan, threat mitigation process 10 may autonomously execute 920 a threat mitigation plan (shutting down the stream and closing the port) when e.g., threat mitigation process 10 assigns 908 a "severe" threat level to the above-described security event (e.g., assuming that it is determined that the streaming of the content is very concerning, as the content is high value and the recipient is a known bad actor).

[00154] Additionally, threat mitigation process 10 may allow 922 a third-party (e.g., the user / owner / operator of computing platform 60) to manually search for artifacts within computing platform 60. For example, the third-party (e.g., the user / owner / operator of computing platform 60) may be able to search the various information resources include within computing platform 60, examples of which may include but are not limited to the various log files maintained by SIEM system 230, and the various log files directly maintained by the security-relevant subsystems within computing platform 60.

Computing Platform Aggregation & Searching

[00155] As will be discussed below in greater detail, threat mitigation process 10 may be configured to e.g., aggregate data sets and allow for unified search of those data sets.

[00156] Referring also to FIG. 18, threat mitigation process 10 may be configured to consolidate multiple separate and discrete data sets to form a single, aggregated data set. For example, threat mitigation process 10 may establish 950 connectivity with a plurality of security-relevant subsystems (e.g., security-relevant subsystems 226) within computing platform 60. As discussed above, examples of security-relevant subsystems 226 may include but are not limited to: CDN (i.e., Content Delivery Network) systems; DAM (i.e., Database Activity Monitoring) systems; UBA (i.e., User Behavior Analytics) systems; MDM (i.e., Mobile Device Management) systems; IAM (i.e., Identity and Access Management) systems; DNS (i.e., Domain Name Server) systems, Antivirus systems, operating systems, data lakes; data logs; security-relevant software applications; security-relevant hardware systems; and resources external to the computing platform.

[00157] When establishing 950 connectivity with a plurality of security-relevant subsystems, threat mitigation process 10 may utilize 952 at least one application program interface (e.g., API Gateway 224) to access at least one of the plurality of security-relevant subsystems. For example, a 1st API gateway may be utilized to access CDN (i.e.,

Content Delivery Network) system; a 2nd API gateway may be utilized to access DAM (i.e., Database Activity Monitoring) system; a 3rd API gateway may be utilized to access UBA (i.e., User Behavior Analytics) system; a 4th API gateway may be utilized to access MDM (i.e., Mobile Device Management) system; a 5th API gateway may be utilized to access IAM (i.e., Identity and Access Management) system; and a 6th API gateway may be utilized to access DNS (i.e., Domain Name Server) system.

[00158] Threat mitigation process 10 may obtain 954 at least one security-relevant information set (e.g., a log file) from each of the plurality of security-relevant subsystems (e.g., CDN system; DAM system; UBA system; MDM system; IAM system; and DNS system), thus defining plurality of security-relevant information sets 258. As would be expected, plurality of security-relevant information sets 258 may utilize a plurality of different formats and/or a plurality of different nomenclatures. Accordingly, threat mitigation process 10 may combine 956 plurality of security-relevant information sets 258 to form an aggregated security-relevant information set 260 for computing platform 60.

[00159] When combining 956 plurality of security-relevant information sets 258 to form aggregated security-relevant information set 260, threat mitigation process 10 may homogenize 958 plurality of security-relevant information sets 258 to form aggregated security-relevant information set 260. For example, threat mitigation process 10 may process one or more of security-relevant information sets 258 so that they all have a common format, a common nomenclature, and/or a common structure.

[00160] Once threat mitigation process 10 combines 956 plurality of security-relevant information sets 258 to form an aggregated security-relevant information set 260 for computing platform 60, threat mitigation process 10 may enable 960 a third-party (e.g., the user / owner / operator of computing platform 60) to access aggregated security-relevant information set 260 and/or enable 962 a third-party (e.g., the user / owner /

operator of computing platform 60) to search aggregated security-relevant information set 260.

[00161] Referring also to FIG. 19, threat mitigation process 10 may be configured to enable the searching of multiple separate and discrete data sets using a single search operation. For example and as discussed above, threat mitigation process 10 may establish 950 connectivity with a plurality of security-relevant subsystems (e.., security-relevant subsystems 226) within computing platform 60. As discussed above, examples of security-relevant subsystems 226 may include but are not limited to: CDN (i.e., Content Delivery Network) systems; DAM (i.e., Database Activity Monitoring) systems; UBA (i.e., User Behavior Analytics) systems; MDM (i.e., Mobile Device Management) systems; IAM (i.e., Identity and Access Management) systems; DNS (i.e., Domain Name Server) systems, Antivirus systems, operating systems, data lakes; data logs; security-relevant software applications; security-relevant hardware systems; and resources external to the computing platform.

[00162] When establishing 950 connectivity with a plurality of security-relevant subsystems, threat mitigation process 10 may utilize 952 at least one application program interface (e.g., API Gateway 224) to access at least one of the plurality of security-relevant subsystems. For example, a 1st API gateway may be utilized to access CDN (i.e., Content Delivery Network) system; a 2nd API gateway may be utilized to access DAM (i.e., Database Activity Monitoring) system; a 3rd API gateway may be utilized to access UBA (i.e., User Behavior Analytics) system; a 4th API gateway may be utilized to access MDM (i.e., Mobile Device Management) system; a 5th API gateway may be utilized to access IAM (i.e., Identity and Access Management) system; and a 6th API gateway may be utilized to access DNS (i.e., Domain Name Server) system.

[00163] Threat mitigation process 10 may receive 1000 unified query 262 from a third-party (e.g., the user / owner / operator of computing platform 60) concerning the plurality of security-relevant subsystems. As discussed above, examples of security-

relevant subsystems 226 may include but are not limited to: CDN (i.e., Content Delivery Network) systems; DAM (i.e., Database Activity Monitoring) systems; UBA (i.e., User Behavior Analytics) systems; MDM (i.e., Mobile Device Management) systems; IAM (i.e., Identity and Access Management) systems; DNS (i.e., Domain Name Server) systems, Antivirus systems, operating systems, data lakes; data logs; security-relevant software applications; security-relevant hardware systems; and resources external to the computing platform.

[00164] Threat mitigation process 10 may distribute 1002 at least a portion of unified query 262 to the plurality of security-relevant subsystems, resulting in the distribution of plurality of queries 264 to the plurality of security-relevant subsystems. For example, assume that a third-party (e.g., the user / owner / operator of computing platform 60) wishes to execute a search concerning the activity of a specific employee. Accordingly, the third-party (e.g., the user / owner / operator of computing platform 60) may formulate the appropriate unified query (e.g., unified query 262) that defines the employee name, the computing device(s) of the employee, and the date range of interest. Unified query 262 may then be parsed to form plurality of queries 264, wherein a specific query (within plurality of queries 264) may be defined for each of the plurality of security-relevant subsystems and provided to the appropriate security-relevant subsystems. For example, a 1st query may be included within plurality of queries 264 and provided to CDN (i.e., Content Delivery Network) system; a 2nd query may be included within plurality of queries 264 and provided to DAM (i.e., Database Activity Monitoring) system; a 3rd query may be included within plurality of queries 264 and provided to UBA (i.e., User Behavior Analytics) system; a 4th query may be included within plurality of queries 264 and provided to MDM (i.e., Mobile Device Management) system; a 5th query may be included within plurality of queries 264 and provided to IAM (i.e., Identity and Access Management) system; and a 6th query may be included within plurality of queries 264 and provided to DNS (i.e., Domain Name Server) system.

[00165] Threat mitigation process 10 may effectuate 1004 at least a portion of unified query 262 on each of the plurality of security-relevant subsystems to generate plurality of result sets 266. For example, the 1st query may be executed on CDN (i.e., Content Delivery Network) system to produce a 1st result set; the 2nd query may be executed on DAM (i.e., Database Activity Monitoring) system to produce a 2nd result set; the 3rd query may be executed on UBA (i.e., User Behavior Analytics) system to produce a 3rd result set; the 4th query may be executed on MDM (i.e., Mobile Device Management) system to produce a 4th result set; the 5th query may be executed on IAM (i.e., Identity and Access Management) system to produce a 5th result set; and the 6th query may executed on DNS (i.e., Domain Name Server) system to produce a 6th result set.

[00166] Threat mitigation process 10 may receive 1006 plurality of result sets 266 from the plurality of security-relevant subsystems. Threat mitigation process 10 may then combine 1008 plurality of result sets 266 to form unified query result 268. When combining 1008 plurality of result sets 266 to form unified query result 268, threat mitigation process 10 may homogenize 1010 plurality of result sets 266 to form unified query result 268. For example, threat mitigation process 10 may process one or more discrete result sets included within plurality of result sets 266 so that the discrete result sets within plurality of result sets 266 all have a common format, a common nomenclature, and/or a common structure. Threat mitigation process 10 may then provide 1012 unified query result 268 to the third-party (e.g., the user / owner / operator of computing platform 60).

[00167] Referring also to FIG. 20, threat mitigation process 10 may be configured to utilize artificial intelligence / machine learning to automatically consolidate multiple separate and discrete data sets to form a single, aggregated data set. For example and as discussed above, threat mitigation process 10 may establish 950 connectivity with a plurality of security-relevant subsystems (e.g., security-relevant subsystems 226) within

computing platform 60. As discussed above, examples of security-relevant subsystems 226 may include but are not limited to: CDN (i.e., Content Delivery Network) systems; DAM (i.e., Database Activity Monitoring) systems; UBA (i.e., User Behavior Analytics) systems; MDM (i.e., Mobile Device Management) systems; IAM (i.e., Identity and Access Management) systems; DNS (i.e., Domain Name Server) systems, Antivirus systems, operating systems, data lakes; data logs; security-relevant software applications; security-relevant hardware systems; and resources external to the computing platform.

[00168] As discussed above and when establishing 950 connectivity with a plurality of security-relevant subsystems, threat mitigation process 10 may utilize 952 at least one application program interface (e.g., API Gateway 224) to access at least one of the plurality of security-relevant subsystems. For example, a 1st API gateway may be utilized to access CDN (i.e., Content Delivery Network) system; a 2nd API gateway may be utilized to access DAM (i.e., Database Activity Monitoring) system; a 3rd API gateway may be utilized to access UBA (i.e., User Behavior Analytics) system; a 4th API gateway may be utilized to access MDM (i.e., Mobile Device Management) system; a 5th API gateway may be utilized to access IAM (i.e., Identity and Access Management) system; and a 6th API gateway may be utilized to access DNS (i.e., Domain Name Server) system.

[00169] As discussed above, threat mitigation process 10 may obtain 954 at least one security-relevant information set (e.g., a log file) from each of the plurality of security-relevant subsystems (e.g., CDN system; DAM system; UBA system; MDM system; IAM system; and DNS system), thus defining plurality of security-relevant information sets 258. As would be expected, plurality of security-relevant information sets 258 may utilize a plurality of different formats and/or a plurality of different nomenclatures.

[00170] Threat mitigation process 10 may process 1050 plurality of security-relevant information sets 258 using artificial learning / machine learning to identify one or more commonalities amongst plurality of security-relevant information sets 258. As

discussed above and with respect to artificial intelligence / machine learning being utilized to process data sets, an initial probabilistic model may be defined, wherein this initial probabilistic model may be subsequently (e.g., iteratively or continuously) modified and revised, thus allowing the probabilistic models and the artificial intelligence systems (e.g., probabilistic process 56) to "learn" so that future probabilistic models may be more precise and may explain more complex data sets. As further discussed above, probabilistic process 56 may define an initial probabilistic model for accomplishing a defined task (e.g., the analyzing of information 58), wherein the probabilistic model may be utilized to go from initial observations about information 58 (e.g., as represented by the initial branches of a probabilistic model) to conclusions about information 58 (e.g., as represented by the leaves of a probabilistic model). Accordingly and through the use of probabilistic process 56, plurality of security-relevant information sets 258 may be processed so that a probabilistic model may be defined (and subsequently revised) to identify one or more commonalities (e.g., common headers, common nomenclatures, common data ranges, common data types, common formats, etc.) amongst plurality of security-relevant information sets 258. When processing 1050 plurality of security-relevant information sets 258 using artificial learning / machine learning to identify one or more commonalities amongst plurality of security-relevant information sets 258, threat mitigation process 10 may utilize 1052 a decision tree (e.g., probabilistic model 100) based, at least in part, upon one or more previously-acquired security-relevant information sets.

[00171] Threat mitigation process 10 may combine 1054 plurality of security-relevant information sets 258 to form aggregated security-relevant information set 260 for computing platform 60 based, at least in part, upon the one or more commonalities identified.

[00172] When combining 1054 plurality of security-relevant information sets 258 to form aggregated security-relevant information set 260 for computing platform 60

based, at least in part, upon the one or more commonalities identified, threat mitigation process 10 may homogenize 1056 plurality of security-relevant information sets 258 to form aggregated security-relevant information set 260. For example, threat mitigation process 10 may process one or more of security-relevant information sets 258 so that they all have a common format, a common nomenclature, and/or a common structure.

[00173] Once threat mitigation process 10 combines 1054 plurality of security-relevant information sets 258 to form an aggregated security-relevant information set 260 for computing platform 60, threat mitigation process 10 may enable 1058 a third-party (e.g., the user / owner / operator of computing platform 60) to access aggregated security-relevant information set 260 and/or enable 1060 a third-party (e.g., the user / owner / operator of computing platform 60) to search aggregated security-relevant information set 260.

Threat Event Information Updating

[00174] As will be discussed below in greater detail, threat mitigation process 10 may be configured to be updated concerning threat event information.

[00175] Referring also to FIG. 21, threat mitigation process 10 may be configured to receive updated threat event information for security-relevant subsystems 226. For example, threat mitigation process 10 may receive 1100 updated threat event information 270 concerning computing platform 60, wherein updated threat event information 270 may define one or more of: updated threat listings; updated threat definitions; updated threat methodologies; updated threat sources; and updated threat strategies. Threat mitigation process 10 may enable 1102 updated threat event information 270 for use with one or more security-relevant subsystems 226 within computing platform 60. As discussed above, examples of security-relevant subsystems 226 may include but are not limited to: CDN (i.e., Content Delivery Network) systems; DAM (i.e., Database Activity Monitoring) systems; UBA (i.e., User Behavior Analytics) systems; MDM (i.e., Mobile Device Management) systems; IAM (i.e., Identity and Access Management) systems;

DNS (i.e., Domain Name Server) systems, Antivirus systems, operating systems, data lakes; data logs; security-relevant software applications; security-relevant hardware systems; and resources external to the computing platform.

[00176] When enabling 1102 updated threat event information 270 for use with one or more security-relevant subsystems 226 within computing platform 60, threat mitigation process 10 may install 1104 updated threat event information 270 on one or more security-relevant subsystems 226 within computing platform 60.

[00177] Threat mitigation process 10 may retroactively apply 1106 updated threat event information 270 to previously-generated information associated with one or more security-relevant subsystems 226.

[00178] When retroactively apply 1106 updated threat event information 270 to previously-generated information associated with one or more security-relevant subsystems 226, threat mitigation process 10 may: apply 1108 updated threat event information 270 to one or more previously-generated log files (not shown) associated with one or more security-relevant subsystems 226; apply 1110 updated threat event information 270 to one or more previously-generated data files (not shown) associated with one or more security-relevant subsystems 226; and apply 1112 updated threat event information 270 to one or more previously-generated application files (not shown) associated with one or more security-relevant subsystems 226.

[00179] Additionally, / alternatively, threat mitigation process 10 may proactively apply 1114 updated threat event information 270 to newly-generated information associated with one or more security-relevant subsystems 226.

[00180] When proactively applying 1114 updated threat event information 270 to newly-generated information associated with one or more security-relevant subsystems 226, threat mitigation process 10 may: apply 1116 updated threat event information 270 to one or more newly-generated log files (not shown) associated with one or more security-relevant subsystems 226; apply 1118 updated threat event information 270 to

one or more newly-generated data files (not shown) associated with one or more security-relevant subsystems 226; and apply 1120 updated threat event information 270 to one or more newly-generated application files (not shown) associated with one or more security-relevant subsystems 226.

[00181] Referring also to FIG. 22, threat mitigation process 10 may be configured to receive updated threat event information 270 for security-relevant subsystems 226. For example and as discussed above, threat mitigation process 10 may receive 1100 updated threat event information 270 concerning computing platform 60, wherein updated threat event information 270 may define one or more of: updated threat listings; updated threat definitions; updated threat methodologies; updated threat sources; and updated threat strategies. Further and as discussed above, threat mitigation process 10 may enable 1102 updated threat event information 270 for use with one or more security-relevant subsystems 226 within computing platform 60. As discussed above, examples of security-relevant subsystems 226 may include but are not limited to: CDN (i.e., Content Delivery Network) systems; DAM (i.e., Database Activity Monitoring) systems; UBA (i.e., User Behavior Analytics) systems; MDM (i.e., Mobile Device Management) systems; IAM (i.e., Identity and Access Management) systems; DNS (i.e., Domain Name Server) systems, Antivirus systems, operating systems, data lakes; data logs; security-relevant software applications; security-relevant hardware systems; and resources external to the computing platform.

[00182] As discussed above and when enabling 1102 updated threat event information 270 for use with one or more security-relevant subsystems 226 within computing platform 60, threat mitigation process 10 may install 1104 updated threat event information 270 on one or more security-relevant subsystems 226 within computing platform 60.

[00183] Sometimes, it may not be convenient and/or efficient to immediately apply updated threat event information 270 to security-relevant subsystems 226. Accordingly,

threat mitigation process 10 may schedule 1150 the application of updated threat event information 270 to previously-generated information associated with one or more security-relevant subsystems 226.

[00184] When scheduling 1150 the application of updated threat event information 270 to previously-generated information associated with one or more security-relevant subsystems 226, threat mitigation process 10 may: schedule 1152 the application of updated threat event information 270 to one or more previously-generated log files (not shown) associated with one or more security-relevant subsystems 226; schedule 1154 the application of updated threat event information 270 to one or more previously-generated data files (not shown) associated with one or more security-relevant subsystems 226; and schedule 1156 the application of updated threat event information 270 to one or more previously-generated application files (not shown) associated with one or more security-relevant subsystems 226.

[00185] Additionally, / alternatively, threat mitigation process 10 may schedule 1158 the application of the updated threat event information to newly-generated information associated with the one or more security-relevant subsystems.

[00186] When scheduling 1158 the application of updated threat event information 270 to newly-generated information associated with one or more security-relevant subsystems 226, threat mitigation process 10 may: schedule 1160 the application of updated threat event information 270 to one or more newly-generated log files (not shown) associated with one or more security-relevant subsystems 226; schedule 1162 the application of updated threat event information 270 to one or more newly-generated data files (not shown) associated with one or more security-relevant subsystems 226; and schedule 1164 the application of updated threat event information 270 to one or more newly-generated application files (not shown) associated with one or more security-relevant subsystems 226.

[00187] Referring also to FIGS. 23-24, threat mitigation process 10 may be configured to initially display analytical data, which may then be manipulated / updated to include automation data. For example, threat mitigation process 10 may display 1200 initial security-relevant information 1250 that includes analytical information (e.g., thought cloud 1252). Examples of such analytical information may include but is not limited to one or more of: investigative information; and hunting information.

[00188] **Investigative Information** (a portion of analytical information): Unified searching and/or automated searching, such as e.g., a security event occurring and searches being performed to gather artifacts concerning that security event.

[00189] **Hunt Information** (a portion of analytical information): Targeted searching / investigations, such as the monitoring and cataloging of the videos that an employee has watched or downloaded over the past 30 days.

[00190] Threat mitigation process 10 may allow 1202 a third-party (e.g., the user / owner / operator of computing platform 60) to manipulate initial security-relevant information 1250 with automation information.

[00191] **Automate Information** (a portion of automation): The execution of a single (and possibly simple) action one time, such as the blocking an IP address from accessing computing platform 60 whenever such an attempt is made.

[00192] **Orchestrate Information** (a portion of automation): The execution of a more complex batch (or series) of tasks, such as sensing an unauthorized download via an API and a) shutting down the API, adding the requesting IP address to a blacklist, and closing any ports opened for the requestor.

[00193] When allowing 1202 a third-party (e.g., the user / owner / operator of computing platform 60) to manipulate initial security-relevant information 1250 with automation information, threat mitigation process 10 may allow 1204 a third-party (e.g., the user / owner / operator of computing platform 60) to select the automation information to add to initial security-relevant information 1250 to generate revised

security-relevant information 1250'. For example and when allowing 1204 a third-party (e.g., the user / owner / operator of computing platform 60) to select the automation information to add to initial security-relevant information 1250 to generate revised security-relevant information 1250', threat mitigation process 10 may allow 1206 the third-party (e.g., the user / owner / operator of computing platform 60) to choose a specific type of automation information from a plurality of automation information types.

[00194] For example, the third-party (e.g., the user / owner / operator of computing platform 60) may choose to add / initiate the automation information to generate revised security-relevant information 1250'. Accordingly, threat mitigation process 10 may render selectable options (e.g., selectable buttons 1254, 1256) that the third-party (e.g., the user / owner / operator of computing platform 60) may select to manipulate initial security-relevant information 1250 with automation information to generate revised security-relevant information 1250'. For this particular example, the third-party (e.g., the user / owner / operator of computing platform 60) may choose two different options to manipulate initial security-relevant information 1250, namely: "block ip" or "search", both of which will result in threat mitigation process 10 generating 1208 revised security-relevant information 1250' (that includes the above-described automation information).

[00195] When generating 1208 revised security-relevant information 1250' (that includes the above-described automation information), threat mitigation process 10 may combine 1210 the automation information (that results from selecting "block IP" or "search") and initial security-relevant information 1250 to generate and render 1212 revised security-relevant information 1250'.

[00196] When rendering 1212 revised security-relevant information 1250', threat mitigation process 10 may render 1214 revised security-relevant information 1250' within interactive report 1258.

Training Routine Generation and Execution

[00197] As will be discussed below in greater detail, threat mitigation process 10 may be configured to allow for the manual or automatic generation of training routines, as well as the execution of the same.

[00198] Referring also to FIG. 25, threat mitigation process 10 may be configured to allow for the manual generation of testing routine 272. For example, threat mitigation process 10 may define 1300 training routine 272 for a specific attack (e.g., a Denial of Services attack) of computing platform 60. Specifically, threat mitigation process 10 may generate 1302 a simulation of the specific attack (e.g., a Denial of Services attack) by executing training routine 272 within a controlled test environment, an example of which may include but is not limited to virtual machine 274 executed on a computing device (e.g., computing device 12).

[00199] When generating 1302 a simulation of the specific attack (e.g., a Denial of Services attack) by executing training routine 272 within the controlled test environment (e.g., virtual machine 274), threat mitigation process 10 may render 1304 the simulation of the specific attack (e.g., a Denial of Services attack) on the controlled test environment (e.g., virtual machine 274).

[00200] Threat mitigation process 10 may allow 1306 a trainee (e.g., trainee 276) to view the simulation of the specific attack (e.g., a Denial of Services attack) and may allow 1308 the trainee (e.g., trainee 276) to provide a trainee response (e.g., trainee response 278) to the simulation of the specific attack (e.g., a Denial of Services attack). For example, threat mitigation process 10 may execute training routine 272, which trainee 276 may "watch" and provide trainee response 278.

[00201] Threat mitigation process 10 may then determine 1310 the effectiveness of trainee response 278, wherein determining 1310 the effectiveness of the trainee response may include threat mitigation process 10 assigning 1312 a grade (e.g., a letter grade or a number grade) to trainee response 278.

[00202] Referring also to FIG. 26, threat mitigation process 10 may be configured to allow for the automatic generation of testing routine 272. For example, threat mitigation process 10 may utilize 1350 artificial intelligence / machine learning to define training routine 272 for a specific attack (e.g., a Denial of Services attack) of computing platform 60.

[00203] As discussed above and with respect to artificial intelligence / machine learning being utilized to process data sets, an initial probabilistic model may be defined, wherein this initial probabilistic model may be subsequently (e.g., iteratively or continuously) modified and revised, thus allowing the probabilistic models and the artificial intelligence systems (e.g., probabilistic process 56) to "learn" so that future probabilistic models may be more precise and may explain more complex data sets. As further discussed above, probabilistic process 56 may define an initial probabilistic model for accomplishing a defined task (e.g., the analyzing of information 58), wherein the probabilistic model may be utilized to go from initial observations about information 58 (e.g., as represented by the initial branches of a probabilistic model) to conclusions about information 58 (e.g., as represented by the leaves of a probabilistic model). Accordingly and through the use of probabilistic process 56, information may be processed so that a probabilistic model may be defined (and subsequently revised) to define training routine 272 for a specific attack (e.g., a Denial of Services attack) of computing platform 60.

[00204] When using 1350 artificial intelligence / machine learning to define training routine 272 for a specific attack (e.g., a Denial of Services attack) of computing platform 60, threat mitigation process 10 may process 1352 security-relevant information to define training routine 272 for specific attack (e.g., a Denial of Services attack) of computing platform 60. Further and when using 1350 artificial intelligence / machine learning to define training routine 272 for a specific attack (e.g., a Denial of Services attack) of computing platform 60, threat mitigation process 10 may utilize 1354 security-relevant rules to define training routine 272 for a specific attack (e.g., a Denial of

Services attack) of computing platform 60. Accordingly, security-relevant information that e.g., defines the symptoms of e.g., a Denial of Services attack and security-relevant rules that define the behavior of e.g., a Denial of Services attack may be utilized by threat mitigation process 10 when defining training routine 272.

[00205] As discussed above, threat mitigation process 10 may generate 1302 a simulation of the specific attack (e.g., a Denial of Services attack) by executing training routine 272 within a controlled test environment, an example of which may include but is not limited to virtual machine 274 executed on a computing device (e.g., computing device 12.

[00206] Further and as discussed above, when generating 1302 a simulation of the specific attack (e.g., a Denial of Services attack) by executing training routine 272 within the controlled test environment (e.g., virtual machine 274), threat mitigation process 10 may render 1304 the simulation of the specific attack (e.g., a Denial of Services attack) on the controlled test environment (e.g., virtual machine 274).

[00207] Threat mitigation process 10 may allow 1306 a trainee (e.g., trainee 276) to view the simulation of the specific attack (e.g., a Denial of Services attack) and may allow 1308 the trainee (e.g., trainee 276) to provide a trainee response (e.g., trainee response 278) to the simulation of the specific attack (e.g., a Denial of Services attack). For example, threat mitigation process 10 may execute training routine 272, which trainee 276 may "watch" and provide trainee response 278.

[00208] Threat mitigation process 10 may utilize 1356 artificial intelligence / machine learning to revise training routine 272 for the specific attack (e.g., a Denial of Services attack) of computing platform 60 based, at least in part, upon trainee response 278.

[00209] As discussed above, threat mitigation process 10 may then determine 1310 the effectiveness of trainee response 278, wherein determining 1310 the effectiveness of

the trainee response may include threat mitigation process 10 assigning 1312 a grade (e.g., a letter grade or a number grade) to trainee response 278.

[00210] Referring also to FIG. 27, threat mitigation process 10 may be configured to allow a trainee to choose their training routine. For example mitigation process 10 may allow 1400 a third-party (e.g., the user / owner / operator of computing platform 60) to select a training routine for a specific attack (e.g., a Denial of Services attack) of computing platform 60, thus defining a selected training routine. When allowing 1400 a third-party (e.g., the user / owner / operator of computing platform 60) to select a training routine for a specific attack (e.g., a Denial of Services attack) of computing platform 60, threat mitigation process 10 may allow 1402 the third-party (e.g., the user / owner / operator of computing platform 60) to choose a specific training routine from a plurality of available training routines. For example, the third-party (e.g., the user / owner / operator of computing platform 60) may be able to select a specific type of attack (e.g., DDoS events, DoS events, phishing events, spamming events, malware events, web attacks, and exploitation events) and/or select a specific training routine (that may or may not disclose the specific type of attack).

[00211] Once selected, threat mitigation process 10 may analyze 1404 the requirements of the selected training routine (e.g., training routine 272) to determine a quantity of entities required to effectuate the selected training routine (e.g., training routine 272), thus defining one or more required entities. For example, assume that training routine 272 has three required entities (e.g., an attacked device and two attacking devices). According, threat mitigation process 10 may generate 1406 one or more virtual machines (e.g., such as virtual machine 274) to emulate the one or more required entities. In this particular example, threat mitigation process 10 may generate 1406 three virtual machines, a first VM for the attacked device, a second VM for the first attacking device and a third VM for the second attacking device. As is known in the art, a virtual machine (VM) is a virtual emulation of a physical computing system. Virtual machines may be

based on computer architectures and may provide the functionality of a physical computer, wherein their implementations may involve specialized hardware, software, or a combination thereof.

[00212] Threat mitigation process 10 may generate 1408 a simulation of the specific attack (e.g., a Denial of Services attack) by executing the selected training routine (e.g., training routine 272). When generating 1408 the simulation of the specific attack (e.g., a Denial of Services attack) by executing the selected training routine (e.g., training routine 272), threat mitigation process 10 may render 1410 the simulation of the specific attack (e.g., a Denial of Services attack) by executing the selected training routine (e.g., training routine 272) within a controlled test environment (e.g., such as virtual machine 274).

[00213] As discussed above, threat mitigation process 10 may allow 1306 a trainee (e.g., trainee 276) to view the simulation of the specific attack (e.g., a Denial of Services attack) and may allow 1308 the trainee (e.g., trainee 276) to provide a trainee response (e.g., trainee response 278) to the simulation of the specific attack (e.g., a Denial of Services attack). For example, threat mitigation process 10 may execute training routine 272, which trainee 276 may "watch" and provide trainee response 278.

[00214] Further and as discussed above, threat mitigation process 10 may then determine 1310 the effectiveness of trainee response 278, wherein determining 1310 the effectiveness of the trainee response may include threat mitigation process 10 assigning 1312 a grade (e.g., a letter grade or a number grade) to trainee response 278.

[00215] When training is complete, threat mitigation process 10 may cease 1412 the simulation of the specific attack (e.g., a Denial of Services attack), wherein ceasing 1412 the simulation of the specific attack (e.g., a Denial of Services attack) may include threat mitigation process 10 shutting down 1414 the one or more virtual machines (e.g., the first VM for the attacked device, the second VM for the first attacking device and the third VM for the second attacking device).

Information Routing

[00216] As will be discussed below in greater detail, threat mitigation process 10 may be configured to route information based upon whether the information is more threat-pertinent or less threat-pertinent.

[00217] Referring also to FIG. 28, threat mitigation process 10 may be configured to route more threat-pertinent content in a specific manner. For example, threat mitigation process 10 may receive 1450 platform information (e.g., log files) from a plurality of security-relevant subsystems (e.g., security-relevant subsystems 226). As discussed above, examples of security-relevant subsystems 226 may include but are not limited to: CDN (i.e., Content Delivery Network) systems; DAM (i.e., Database Activity Monitoring) systems; UBA (i.e., User Behavior Analytics) systems; MDM (i.e., Mobile Device Management) systems; IAM (i.e., Identity and Access Management) systems; DNS (i.e., Domain Name Server) systems, Antivirus systems, operating systems, data lakes; data logs; security-relevant software applications; security-relevant hardware systems; and resources external to the computing platform.

[00218] Threat mitigation process 10 may process 1452 this platform information (e.g., log files) to generate processed platform information. And when processing 1452 this platform information (e.g., log files) to generate processed platform information, threat mitigation process 10 may: parse 1454 the platform information (e.g., log files) into a plurality of subcomponents (e.g., columns, rows, etc.) to allow for compensation of varying formats and/or nomenclature; enrich 1456 the platform information (e.g., log files) by including supplemental information from external information resources; and/or utilize 1458 artificial intelligence / machine learning (in the manner described above) to identify one or more patterns / trends within the platform information (e.g., log files).

[00219] Threat mitigation process 10 may identify 1460 more threat-pertinent content 280 included within the processed content, wherein identifying 1460 more threat-pertinent content 280 included within the processed content may include processing 1462

the processed content to identify actionable processed content that may be used by a threat analysis engine (e.g., SIEM system 230) for correlation purposes. Threat mitigation process 10 may route 1464 more threat-pertinent content 280 to this threat analysis engine (e.g., SIEM system 230).

[00220] Referring also to FIG. 29, threat mitigation process 10 may be configured to route less threat-pertinent content in a specific manner. For example and as discussed above, threat mitigation process 10 may receive 1450 platform information (e.g., log files) from a plurality of security-relevant subsystems (e.g., security-relevant subsystems 226). As discussed above, examples of security-relevant subsystems 226 may include but are not limited to: CDN (i.e., Content Delivery Network) systems; DAM (i.e., Database Activity Monitoring) systems; UBA (i.e., User Behavior Analytics) systems; MDM (i.e., Mobile Device Management) systems; IAM (i.e., Identity and Access Management) systems; DNS (i.e., Domain Name Server) systems, Antivirus systems, operating systems, data lakes; data logs; security-relevant software applications; security-relevant hardware systems; and resources external to the computing platform

[00221] Further and as discussed above, threat mitigation process 10 may process 1452 this platform information (e.g., log files) to generate processed platform information. And when processing 1452 this platform information (e.g., log files) to generate processed platform information, threat mitigation process 10 may: parse 1454 the platform information (e.g., log files) into a plurality of subcomponents (e.g., columns, rows, etc.) to allow for compensation of varying formats and/or nomenclature; enrich 1456 the platform information (e.g., log files) by including supplemental information from external information resources; and/or utilize 1458 artificial intelligence / machine learning (in the manner described above) to identify one or more patterns / trends within the platform information (e.g., log files).

[00222] Threat mitigation process 10 may identify 1500 less threat-pertinent content 282 included within the processed content, wherein identifying 1500 less threat-

pertinent content 282 included within the processed content may include processing 1502 the processed content to identify non-actionable processed content that is not usable by a threat analysis engine (e.g., SIEM system 230) for correlation purposes. Threat mitigation process 10 may route 1504 less threat-pertinent content 282 to a long-term storage system (e.g., long term storage system 284). Further, threat mitigation process 10 may be configured to allow 1506 a third-party (e.g., the user / owner / operator of computing platform 60) to access and search long term storage system 284.

Automated Analysis

[00223] As will be discussed below in greater detail, threat mitigation process 10 may be configured to automatically analyze a detected security event.

[00224] Referring also to FIG. 30, threat mitigation process 10 may be configured to automatically classify and investigate a detected security event. As discussed above and in response to a security event being detected, threat mitigation process 10 may obtain 1550 one or more artifacts (e.g., artifacts 250) concerning the detected security event. Examples of such a detected security event may include but are not limited to one or more of: access auditing; anomalies; authentication; denial of services; exploitation; malware; phishing; spamming; reconnaissance; and web attack. These artifacts (e.g., artifacts 250) may be obtained 1550 from a plurality of sources associated with the computing platform, wherein examples of such plurality of sources may include but are not limited to the various log files maintained by SIEM system 230, and the various log files directly maintained by the security-relevant subsystems

[00225] Threat mitigation process 10 may obtain 1552 artifact information (e.g., artifact information 286) concerning the one or more artifacts (e.g., artifacts 250), wherein artifact information 286 may be obtained from information resources include within (or external to) computing platform 60.

[00226] For example and when obtaining 1552 artifact information 286 concerning the one or more artifacts (e.g., artifacts 250), threat mitigation process 10 may obtain

1554 artifact information 286 concerning the one or more artifacts (e.g., artifacts 250) from one or more investigation resources (such as third-party resources that may e.g., provide information on known bad actors).

[00227] Once the investigation is complete, threat mitigation process 10 may generate 1556 a conclusion (e.g., conclusion 288) concerning the detected security event (e.g., a Denial of Services attack) based, at least in part, upon the detected security event (e.g., a Denial of Services attack), the one or more artifacts (e.g., artifacts 250), and artifact information 286. Threat mitigation process 10 may document 1558 the conclusion (e.g., conclusion 288), report 1560 the conclusion (e.g., conclusion 288) to a third-party (e.g., the user / owner / operator of computing platform 60). Further, threat mitigation process 10 may obtain 1562 supplemental artifacts and artifact information (if needed to further the investigation).

[00228] While the system is described above as being computer-implemented, this is for illustrative purposes only and is not intended to be a limitation of this disclosure, as other configurations are possible and are considered to be within the scope of this disclosure. For example, some or all of the above-described system may be implemented by a human being.

Unified Searching

[00229] As discussed above, threat mitigation process 10 may be configured to e.g., analyze a monitored computing platform (e.g., computing platform 60) and provide information to third-parties concerning the same. Further and as discussed above, such a monitored computing platform (e.g., computing platform 60) may be a highly complex, multi-location computing system / network that may span multiple buildings / locations / countries.

[00230] For this illustrative example, the monitored computing platform (e.g., computing platform 60) is shown to include many discrete computing devices, examples of which may include but are not limited to: server computers (e.g., server computers

200, 202), desktop computers (e.g., desktop computer 204), and laptop computers (e.g., laptop computer 206), all of which may be coupled together via a network (e.g., network 208), such as an Ethernet network. Computing platform 60 may be coupled to an external network (e.g., Internet 210) through WAF (i.e., Web Application Firewall) 212. A wireless access point (e.g., WAP 214) may be configured to allow wireless devices (e.g., smartphone 216) to access computing platform 60. Computing platform 60 may include various connectivity devices that enable the coupling of devices within computing platform 60, examples of which may include but are not limited to: switch 216, router 218 and gateway 220. Computing platform 60 may also include various storage devices (e.g., NAS 222), as well as functionality (e.g., API Gateway 224) that allows software applications to gain access to one or more resources within computing platform 60.

[00231] In addition to the devices and functionality discussed above, other technology (e.g., security-relevant subsystems 226) may be deployed within computing platform 60 to monitor the operation of (and the activity within) computing platform 60. Examples of security-relevant subsystems 226 may include but are not limited to: CDN (i.e., Content Delivery Network) systems; DAM (i.e., Database Activity Monitoring) systems; UBA (i.e., User Behavior Analytics) systems; MDM (i.e., Mobile Device Management) systems; IAM (i.e., Identity and Access Management) systems; DNS (i.e., Domain Name Server) systems, antivirus systems, operating systems, data lakes; data logs; security-relevant software applications; security-relevant hardware systems; and resources external to the computing platform. Each of security-relevant subsystems 226 may monitor and log their activity with respect to computing platform 60, resulting in the generation of platform information 228. For example, platform information 228 associated with a client-defined MDM (i.e., Mobile Device Management) system may monitor and log the mobile devices that were allowed access to computing platform 60.

[00232] Further, SEIM (i.e., Security Information and Event Management) system 230 may be deployed within computing platform 60. As is known in the art, SIEM system 230 is an approach to security management that combines SIM (security information management) functionality and SEM (security event management) functionality into one security management system. The underlying principles of a SIEM system is to aggregate relevant data from multiple sources, identify deviations from the norm and take appropriate action. For example, when a security event is detected, SIEM system 230 might log additional information, generate an alert and instruct other security controls to mitigate the security event. Accordingly, SIEM system 230 may be configured to monitor and log the activity of security-relevant subsystems 226 (e.g., CDN (i.e., Content Delivery Network) systems; DAM (i.e., Database Activity Monitoring) systems; UBA (i.e., User Behavior Analytics) systems; MDM (i.e., Mobile Device Management) systems; IAM (i.e., Identity and Access Management) systems; DNS (i.e., Domain Name Server) systems, antivirus systems, operating systems, data lakes; data logs; security-relevant software applications; security-relevant hardware systems; and resources external to the computing platform).

[00233] Referring also to FIGS. 31-32, threat mitigation process 10 may be configured to enable the querying of multiple separate and discrete subsystems (e.g., security-relevant subsystems 226) using a single query operation. For example, threat mitigation process 10 may establish 1600 connectivity with a plurality of security-relevant subsystems (e.g., security-relevant subsystems 226) within computing platform 60.

[00234] As discussed above, examples of security-relevant subsystems 226 may include but are not limited to: CDN (i.e., Content Delivery Network) systems; DAM (i.e., Database Activity Monitoring) systems; UBA (i.e., User Behavior Analytics) systems; MDM (i.e., Mobile Device Management) systems; IAM (i.e., Identity and Access Management) systems; DNS (i.e., Domain Name Server) systems, Antivirus systems,

operating systems, data lakes; data logs; security-relevant software applications; security-relevant hardware systems; and resources external to the computing platform.

[00235] When establishing 1600 connectivity with a plurality of security-relevant subsystems (e.g., security-relevant subsystems 226), threat mitigation process 10 may utilize at least one application program interface (e.g., API Gateway 224) to access at least one of the plurality of security-relevant subsystems. For example, a 1st API gateway may be utilized to access CDN (i.e., Content Delivery Network) system; a 2nd API gateway may be utilized to access DAM (i.e., Database Activity Monitoring) system; a 3rd API gateway may be utilized to access UBA (i.e., User Behavior Analytics) system; a 4th API gateway may be utilized to access MDM (i.e., Mobile Device Management) system; a 5th API gateway may be utilized to access IAM (i.e., Identity and Access Management) system; and a 6th API gateway may be utilized to access DNS (i.e., Domain Name Server) system.

[00236] In order to enable the querying of multiple separate and discrete subsystems (e.g., security-relevant subsystems 226) using a single query operation, threat mitigation process 10 may map 1602 one or more data fields of unified platform 290 (e.g., a platform effectuated by threat mitigation process 10) to one or more data fields of each of the plurality of security-relevant subsystems (e.g., security-relevant subsystems 226).

[00237] For example, unified platform 290 may be a platform that enables a third-party (e.g., the user / owner / operator of computing platform 60) to query multiple security-relevant subsystems (within security-relevant subsystems 226), such as security-relevant subsystem 1650, security-relevant subsystem 1652 and security-relevant subsystem 1654. As discussed above, examples of such security-relevant subsystem (e.g., security-relevant subsystem 1650, security-relevant subsystem 1652 and security-relevant subsystem 1654) may include but are not limited to: CDN (i.e., Content Delivery Network) systems; DAM (i.e., Database Activity Monitoring) systems; UBA (i.e., User

Behavior Analytics) systems; MDM (i.e., Mobile Device Management) systems; IAM (i.e., Identity and Access Management) systems; DNS (i.e., Domain Name Server) systems, antivirus systems, operating systems, data lakes; data logs; security-relevant software applications; security-relevant hardware systems; and resources external to the computing platform.

[00238] Each of these security-relevant subsystem (e.g., security-relevant subsystem 1650, security-relevant subsystem 1652 and security-relevant subsystem 1654) may include a plurality of data fields that enable the third-party (e.g., the user / owner / operator of computing platform 60) to search for and obtain information from these security-relevant subsystems (e.g., security-relevant subsystem 1650, security-relevant subsystem 1652 and security-relevant subsystem 1654). For example: security-relevant subsystem 1650 is shown to include data fields 1656, 1658, 1660, 1662; security-relevant subsystem 1652 is shown to include data fields 1664, 1666, 1668, 1670; and security-relevant subsystem 1654 is shown to include data fields 1672, 1674, 1676, 1678.

[00239] These data fields (e.g., data fields 1656, 1658, 1660, 1662, 1664, 1666, 1668, 1670, 1672, 1674, 1676, 1678) may be populatable by the third-party (e.g., the user / owner / operator of computing platform 60) to enable such searching. For example, the third-party (e.g., the user / owner / operator of computing platform 60) may populate these data fields by typing information into some of these data fields (e.g., data fields 1656, 1658, 1660, 1666, 1668, 1670, 1672, 1674, 1676). Additionally / alternatively, the third-party (e.g., the user / owner / operator of computing platform 60) may populate these data fields via a drop-down menu available within some of these data fields (e.g., data fields 1662, 1664, 1678). For example, data field 1662 is shown to be populatable via drop down menu 1680, data field 1664 is shown to be populatable via drop down menu 1682, and data field 1678 is shown to be populatable via drop down menu 1684.

[00240] Through the use of such data fields, the third-party (e.g., the user / owner / operator of computing platform 60) may populate one of more of these data fields to

define a query that may be effectuated on the information contained / available within these security-relevant subsystems (e.g., security-relevant subsystem 1650, security-relevant subsystem 1652 and security-relevant subsystem 1654) so that the pertinent information may be obtained.

[00241] Naturally, the subject matter of these individual data fields may vary depending upon the type of information available via these security-relevant subsystems (e.g., security-relevant subsystem 1650, security-relevant subsystem 1652 and security-relevant subsystem 1654). As (in this example) these are security-relevant subsystems, the information available from these security-relevant subsystems concerns the security of computing platform 60 and/or any security events (e.g., access auditing; anomalies; authentication; denial of services; exploitation; malware; phishing; spamming; reconnaissance; and/or web attack) occurring therein. For example, some of these data fields may concern e.g., user names, user IDs, device locations, device types, device IP addresses, source IP addresses, destination IP addresses, port addresses, deployed operating systems, utilized bandwidth, etc.

[00242] As discussed above, in order to enable the querying of multiple separate and discrete subsystems (e.g., security-relevant subsystem 1650, security-relevant subsystem 1652 and security-relevant subsystem 1654) using a single query operation, threat mitigation process 10 may map 1602 one or more data fields of unified platform 290 (e.g., a platform effectuated by threat mitigation process 10) to one or more data fields of each of the plurality of security-relevant subsystems (e.g., security-relevant subsystem 1650, security-relevant subsystem 1652 and security-relevant subsystem 1654).

[00243] In this particular example, unified platform 290 (e.g., a platform effectuated by threat mitigation process 10) is shown to include four data fields (e.g., data fields 1686, 1688, 1690, 1692), wherein:

- data field 1686 within unified platform 290 concerns a user ID (and is entitled USER_ID);
- data field 1688 within unified platform 290 concerns a device IP address (and is entitled DEVICE_IP);
- data field 1690 within unified platform 290 concerns a destination IP address (and is entitled DESTINATION_IP); and
- data field 1692 within unified platform 290 concerns a query result set (and is entitled QUERY_RESULT).

[00244] When mapping 1602 data fields within unified platform 290 (e.g., a platform effectuated by threat mitigation process 10) to data fields within each of the plurality of security-relevant subsystems (e.g., security-relevant subsystem 1650, security-relevant subsystem 1652 and security-relevant subsystem 1654), threat mitigation process 10 may only map 1602 data fields that are related with respect to subject matter.

[00245] As discussed above, data field 1686 within unified platform 290 (e.g., a platform effectuated by threat mitigation process 10) concerns a user ID (and is entitled USER_ID). For this example, assume that:

- data field 1656 within security-relevant subsystem 1650 also concerns a user ID and is entitled USER;
- data field 1666 within security-relevant subsystem 1652 also concerns a user ID and is entitled ID; and
- data field 1676 within security-relevant subsystem 1654 also concerns a user ID and is entitled USR_ID.

[00246] Accordingly, threat mitigation process 10 may map 1602 data field 1686 of unified platform 290 (e.g., a platform effectuated by threat mitigation process 10) to:

- data field 1656 of security-relevant subsystem 1650;
- data field 1666 of security-relevant subsystem 1652; and

- data field 1676 of security-relevant subsystem 1654.

[00247] As discussed above, data field 1688 within unified platform 290 (e.g., a platform effectuated by threat mitigation process 10) concerns a device IP address (and is entitled DEVICE_IP). For this example, assume that:

- data field 1660 within security-relevant subsystem 1650 also concerns a device IP address and is entitled DEV_IP;
- data field 1670 within security-relevant subsystem 1652 also concerns a device IP address and is entitled IP_DEVICE; and
- data field 1674 within security-relevant subsystem 1654 also concerns a device IP address and is entitled IP_DEV.

[00248] Accordingly, threat mitigation process 10 may map 1602 data field 1688 of unified platform 290 (e.g., a platform effectuated by threat mitigation process 10) to:

- data field 1660 of security-relevant subsystem 1650;
- data field 1670 of security-relevant subsystem 1652; and
- data field 1674 of security-relevant subsystem 1654.

[00249] As discussed above, data field 1690 within unified platform 290 (e.g., a platform effectuated by threat mitigation process 10) concerns a destination IP address (and is entitled DESTINATION_IP). For this example, assume that:

- data field 1658 within security-relevant subsystem 1650 also concerns a destination IP address and is entitled DEST_IP;
- data field 1668 within security-relevant subsystem 1652 also concerns a destination IP address and is entitled IP_DEST; and
- data field 1672 within security-relevant subsystem 1654 also concerns a destination IP address and is entitled IP_DES.

[00250] Accordingly, threat mitigation process 10 may map 1602 data field 1690 of unified platform 290 (e.g., a platform effectuated by threat mitigation process 10) to:

- data field 1658 of security-relevant subsystem 1650;

- data field 1668 of security-relevant subsystem 1652; and
- data field 1672 of security-relevant subsystem 1654.

[00251] As discussed above, data field 1692 within unified platform 290 (e.g., a platform effectuated by threat mitigation process 10) concerns a query result (and is entitled QUERY_RESULT). For this example, assume that:

- data field 1662 within security-relevant subsystem 1650 also concerns a query result and is entitled RESULT;
- data field 1664 within security-relevant subsystem 1652 also concerns a query result and is entitled Q_RESULT; and
- data field 1678 within security-relevant subsystem 1654 also concerns a query result and is entitled RESULT_Q.

[00252] Accordingly, threat mitigation process 10 may map 1602 data field 1692 of unified platform 290 (e.g., a platform effectuated by threat mitigation process 10) to:

- data field 1662 of security-relevant subsystem 1650;
- data field 1664 of security-relevant subsystem 1652; and
- data field 1678 of security-relevant subsystem 1654.

[00253] Through the use of threat mitigation process 10, a query (e.g., query 1694) may be defined within one or more of data fields 1686, 1688, 1690 of unified platform 290 (e.g., a platform effectuated by threat mitigation process 10), wherein this query (e.g., query 1694) may be provided (via the above-described mappings) to the appropriate data fields within the security-relevant subsystems (e.g., security-relevant subsystem 1650, security-relevant subsystem 1652 and security-relevant subsystem 1654).

[00254] Accordingly and when mapping 1602 one or more data fields of the unified platform (e.g., unified platform 290) to one or more data fields of each of the plurality of security-relevant subsystems (e.g., security-relevant subsystem 1650, security-relevant subsystem 1652 and security-relevant subsystem 1654), threat mitigation process 10 may map 1604 one or more data fields within a query structure of

the unified platform (e.g., unified platform 290) to one or more data fields within a query structure of each of the plurality of security-relevant subsystems (e.g., security-relevant subsystem 1650, security-relevant subsystem 1652 and security-relevant subsystem 1654).

[00255] Therefore, if a query (e.g., query 1694) was defined on unified platform 290 (e.g., a platform effectuated by threat mitigation process 10) that specified a user ID within data field 1686, a device IP address within data field 1688, and a destination IP address within data field 1690; by mapping 1604 one or more data fields of the unified platform (e.g., unified platform 290) to one or more data fields of each of the plurality of security-relevant subsystems (e.g., security-relevant subsystem 1650, security-relevant subsystem 1652 and security-relevant subsystem 1654), this structured query (e.g., query 1694) may be provided to the plurality of security-relevant subsystems (e.g., security-relevant subsystem 1650, security-relevant subsystem 1652 and security-relevant subsystem 1654) in a fashion that enables the plurality of security-relevant subsystems (e.g., security-relevant subsystem 1650, security-relevant subsystem 1652 and security-relevant subsystem 1654) to effectuate the structured query (e.g., query 1694).

[00256] Upon effectuating such a structured query (e.g., query 1694), the plurality of security-relevant subsystems (e.g., security-relevant subsystem 1650, security-relevant subsystem 1652 and security-relevant subsystem 1654) may each generate a subsystem-specific result set. For example, security-relevant subsystem 1650 may generate subsystem-specific result set 1696, security-relevant subsystem 1652 may generate subsystem-specific result set 1698, and security-relevant subsystem 1654 may generate subsystem-specific result set 1700.

[00257] Through the use of threat mitigation process 10, subsystem-specific result sets (e.g., subsystem-specific result sets 1696, 1698, 1700) may be defined within one or more of data fields (e.g., data fields 1662, 1664, 1678) of the plurality of security-relevant subsystems (e.g., security-relevant subsystem 1650, security-relevant subsystem

1652 and security-relevant subsystem 1654), wherein these subsystem-specific result sets (e.g., subsystem-specific result sets 1696, 1698, 1700) may be provided (via the above-described mappings) to the appropriate data fields within the unified platform (e.g., unified platform 290).

[00258] Accordingly and when mapping 1602 one or more data fields of the unified platform (e.g., unified platform 290) to one or more data fields of each of the plurality of security-relevant subsystems (e.g., security-relevant subsystem 1650, security-relevant subsystem 1652 and security-relevant subsystem 1654), threat mitigation process 10 may map 1606 one or more data fields within a result set structure of each of the plurality of security-relevant subsystems (e.g., security-relevant subsystem 1650, security-relevant subsystem 1652 and security-relevant subsystem 1654) to one or more data fields within a result set structure of the unified platform (e.g., unified platform 290).

[00259] Therefore, by mapping 1606 one or more data fields within a result set structure of each of the plurality of security-relevant subsystems (e.g., security-relevant subsystem 1650, security-relevant subsystem 1652 and security-relevant subsystem 1654) to one or more data fields within a result set structure of the unified platform (e.g., unified platform 290), these subsystem-specific result sets (e.g., subsystem-specific result sets 1696, 1698, 1700) may be provided to the unified platform (e.g., unified platform 290) in a fashion that enables the unified platform (e.g., unified platform 290) to properly process these subsystem-specific result sets (e.g., subsystem-specific result sets 1696, 1698, 1700).

[00260] It is foreseeable that over time, the data fields within the plurality of security-relevant subsystems (e.g., security-relevant subsystem 1650, security-relevant subsystem 1652 and security-relevant subsystem 1654) may change. For example, additional data fields may be added to and/or certain data fields may be deleted from the plurality of security-relevant subsystems. Accordingly and in order to ensure that the

above-described mapping remain current and accurate, such mappings may be periodically refreshed.

[00261] Accordingly and when mapping 1602 one or more data fields of the unified platform (e.g., unified platform 290) to one or more data fields of each of the plurality of security-relevant subsystems (e.g., security-relevant subsystem 1650, security-relevant subsystem 1652 and security-relevant subsystem 1654), threat mitigation process 10 may map 1608 one or more data fields of the unified platform (e.g., unified platform 290) to one or more data fields of each of the plurality of security-relevant subsystems (e.g., security-relevant subsystem 1650, security-relevant subsystem 1652 and security-relevant subsystem 1654) at a defined periodicity.

[00262] Therefore, at a certain frequency (e.g., every few minutes, every few hours, every few days, every few weeks or every few months), the above-describe mapping process may be reperformed to ensure that the above-described mappings are up to date.

[00263] Further and when mapping 1602 one or more data fields of the unified platform (e.g., unified platform 290) to one or more data fields of each of the plurality of security-relevant subsystems (e.g., security-relevant subsystem 1650, security-relevant subsystem 1652 and security-relevant subsystem 1654), threat mitigation process 10 may proactively map 1610 one or more data fields of the unified platform (e.g., unified platform 290) to one or more data fields of each of the plurality of security-relevant subsystems (e.g., security-relevant subsystem 1650, security-relevant subsystem 1652 and security-relevant subsystem 1654).

[00264] For example, the above-described mapping process may be proactively done, wherein threat mitigation process 10 actively monitors the security-relevant subsystems within computing platform 60 so that the data fields within these security-relevant subsystems may be proactively mapped 1610 prior to a third-party (e.g., the user

/ owner / operator of computing platform 60) defining a query within unified platform 290.

[00265] Additionally and when mapping 1602 one or more data fields of the unified platform (e.g., unified platform 290) to one or more data fields of each of the plurality of security-relevant subsystems (e.g., security-relevant subsystem 1650, security-relevant subsystem 1652 and security-relevant subsystem 1654), threat mitigation process 10 may reactively map 1612 one or more data fields of the unified platform (e.g., unified platform 290) to one or more data fields of each of the plurality of security-relevant subsystems (e.g., security-relevant subsystem 1650, security-relevant subsystem 1652 and security-relevant subsystem 1654).

[00266] For example, the above-described mapping process may be reactively performed, wherein threat mitigation process 10 may not actively monitor the security-relevant subsystems within computing platform 60 and the data fields within these security-relevant subsystems may be reactively mapped 1612 after a third-party (e.g., the user / owner / operator of computing platform 60) defines a query within unified platform 290.

[00267] As discussed above, threat mitigation process 10 may allow a third-party (e.g., the user / owner / operator of computing platform 60) to define 1614 a unified query (e.g., query 1694) on a unified platform (e.g., unified platform 290) concerning security-relevant subsystems 226 (e.g., security-relevant subsystem 1650, security-relevant subsystem 1652 and security-relevant subsystem 1654).

[00268] As discussed above, examples of security-relevant subsystems 226 may include but are not limited to: CDN (i.e., Content Delivery Network) systems; DAM (i.e., Database Activity Monitoring) systems; UBA (i.e., User Behavior Analytics) systems; MDM (i.e., Mobile Device Management) systems; IAM (i.e., Identity and Access Management) systems; DNS (i.e., Domain Name Server) systems, Antivirus systems,

operating systems, data lakes; data logs; security-relevant software applications; security-relevant hardware systems; and resources external to the computing platform.

[00269] Threat mitigation process 10 may denormalize 1616 the unified query (e.g., query 1694) to define a subsystem-specific query for each of the plurality of security-relevant subsystems (e.g., security-relevant subsystem 1650, security-relevant subsystem 1652 and security-relevant subsystem 1654), thus defining a plurality of subsystem-specific queries (e.g., subsystem-specific queries 1702, 1704, 1706).

[00270] As discussed above, unified platform 290 (e.g., a platform effectuated by threat mitigation process 10) is shown to include four data fields (e.g., data fields 1686, 1688, 1690, 1692), wherein a third-party (e.g., the user / owner / operator of computing platform 60) may utilize these data fields to define the unified query (e.g., query 1694). As this unified query (e.g., query 1694) may be used as the basis to search for pertinent information on (in this example) three entirely separate and discrete subsystems (e.g., security-relevant subsystem 1650, security-relevant subsystem 1652 and security-relevant subsystem 1654), it is foreseeable that these subsystems (e.g., security-relevant subsystem 1650, security-relevant subsystem 1652 and security-relevant subsystem 1654) may require queries to be structured differently.

[00271] Accordingly and when denormalizing 1616 the unified query (e.g., query 1694) to define a subsystem-specific query for each of the plurality of security-relevant subsystems (e.g., security-relevant subsystem 1650, security-relevant subsystem 1652 and security-relevant subsystem 1654), thus defining a plurality of subsystem-specific queries (e.g., subsystem-specific queries 1702, 1704, 1706), threat mitigation process 10 may translate 1618 a syntax of the unified query (e.g., query 1694) to a syntax of each of the plurality of subsystem-specific queries (e.g., subsystem-specific queries 1702, 1704, 1706). For example:

- security-relevant subsystem 1650 may only be capable of processing queries having a first structure and/or utilizing a first nomenclature;

- security-relevant subsystem 1652 may only be capable of processing queries having a second structure and/or utilizing a second nomenclature; and
- security-relevant subsystem 1654 may only be capable of processing queries having a third structure and/or utilizing a third nomenclature.

[00272] Accordingly and when denormalizing 1616 the unified query (e.g., query 1694) to define a plurality of subsystem-specific queries (e.g., subsystem-specific queries 1702, 1704, 1706), threat mitigation process 10 may translate 1618 the syntax of the unified query (e.g., query 1694) so that:

- subsystem-specific query 1702 has a first structure and/or utilizes a first nomenclature;
- subsystem-specific query 1704 has a second structure and/or utilizes a second nomenclature;
- subsystem-specific query 1706 has a third structure and/or utilizes a third nomenclature.

[00273] Threat mitigation process 10 may provide 1620 the plurality of subsystem-specific queries (e.g., subsystem-specific queries 1702, 1704, 1706) to the plurality of security-relevant subsystems (e.g., security-relevant subsystem 1650, security-relevant subsystem 1652 and security-relevant subsystem 1654).

[00274] The plurality of subsystem-specific queries (e.g., subsystem-specific queries 1702, 1704, 1706) may be effectuated on the appropriate security-relevant subsystem. For example, subsystem-specific query 1702 may be effectuated on security-relevant subsystem 1650, subsystem-specific query 1704 may be effectuated on security-relevant subsystem 1652, and subsystem-specific query 1706 may be effectuated on security-relevant subsystem 1654; resulting in the generation of subsystem-specific result sets. For example, security-relevant subsystem 1650 may generate subsystem-specific result set 1696, security-relevant subsystem 1652 may generate subsystem-specific result

set 1698, and security-relevant subsystem 1654 may generate subsystem-specific result set 1700.

[00275] Threat mitigation process 10 may receive 1622 a plurality of subsystem-specific results sets (e.g., subsystem-specific result sets 1696, 1698, 1700) from the plurality of security-relevant subsystems (e.g., security-relevant subsystem 1650, security-relevant subsystem 1652 and security-relevant subsystem 1654, respectively) that were generated in response to the plurality of subsystem-specific queries (e.g., subsystem-specific queries 1702, 1704, 1706).

[00276] Threat mitigation process 10 may normalize 1624 the plurality of subsystem-specific results sets (e.g., subsystem-specific result sets 1696, 1698, 1700) received from the plurality of security-relevant subsystems (e.g., security-relevant subsystem 1650, security-relevant subsystem 1652 and security-relevant subsystem 1654, respectively) to define a unified result set (e.g., unified result set 1708). For example, threat mitigation process 10 may process the plurality of subsystem-specific results sets (e.g., subsystem-specific result sets 1696, 1698, 1700) so that the subsystem-specific results sets all have a common format, a common nomenclature, and/or a common structure.

[00277] Accordingly and when normalizing 1624 the plurality of subsystem-specific results sets (e.g., subsystem-specific result sets 1696, 1698, 1700) received from the plurality of security-relevant subsystems (e.g., security-relevant subsystem 1650, security-relevant subsystem 1652 and security-relevant subsystem 1654, respectively) to define a unified result set (e.g., unified result set 1708), threat mitigation process 10 may translate 1626 a syntax of each of the plurality of subsystem-specific results sets (e.g., subsystem-specific result sets 1696, 1698, 1700) to a syntax of the unified result set (e.g., unified result set 1708).

[00278] As discussed above:

- security-relevant subsystem 1650 may only be capable of processing queries having a first structure and/or utilizing a first nomenclature;
- security-relevant subsystem 1652 may only be capable of processing queries having a second structure and/or utilizing a second nomenclature; and
- security-relevant subsystem 1654 may only be capable of processing queries having a third structure and/or utilizing a third nomenclature.

[00279] Accordingly and when producing a result set:

- security-relevant subsystem 1650 may only be capable producing a result set (e.g., subsystem-specific result set 1696) having a first structure and/or utilizing a first nomenclature;
- security-relevant subsystem 1652 may only be capable producing a result set (e.g., subsystem-specific result set 1698) having a second structure and/or utilizing a second nomenclature; and
- security-relevant subsystem 1654 may only be capable producing a result set (e.g., subsystem-specific result set 1700) having a third structure and/or utilizing a third nomenclature.

[00280] Accordingly and when normalizing 1624 the plurality of subsystem-specific results sets (e.g., subsystem-specific result sets 1696, 1698, 1700) received from the plurality of security-relevant subsystems (e.g., security-relevant subsystem 1650, security-relevant subsystem 1652 and security-relevant subsystem 1654, respectively) to define a unified result set (e.g., unified result set 1708), threat mitigation process 10 may translate 1626 the syntax of:

- subsystem-specific result set 1696 from a first structure / first nomenclature to a unified syntax of the unified result set (e.g., unified result set 1708);

- subsystem-specific result set 1698 from a second structure / second nomenclature to the unified syntax of the unified result set (e.g., unified result set 1708);
- subsystem-specific result set 1700 from a third structure / third nomenclature to a unified syntax of the unified result set (e.g., unified result set 1708).

[00281] Once normalized 1624, 1626, threat mitigation process 10 may combine the subsystem-specific results sets (e.g., subsystem-specific result sets 1696, 1698, 1700) to form the unified result set (e.g., unified result set 1708), wherein threat mitigation process 10 may then provide 1628 the unified result set (e.g., unified result set 1708) to a third-party (e.g., the user / owner / operator of computing platform 60).

General

[00282] As will be appreciated by one skilled in the art, the present disclosure may be embodied as a method, a system, or a computer program product. Accordingly, the present disclosure may take the form of an entirely hardware embodiment, an entirely software embodiment (including firmware, resident software, micro-code, etc.) or an embodiment combining software and hardware aspects that may all generally be referred to herein as a "circuit," "module" or "system." Furthermore, the present disclosure may take the form of a computer program product on a computer-usable storage medium having computer-usable program code embodied in the medium.

[00283] Any suitable computer usable or computer readable medium may be utilized. The computer-usable or computer-readable medium may be, for example but not limited to, an electronic, magnetic, optical, electromagnetic, infrared, or semiconductor system, apparatus, device, or propagation medium. More specific examples (a non-exhaustive list) of the computer-readable medium may include the following: an electrical connection having one or more wires, a portable computer diskette, a hard disk, a random access memory (RAM), a read-only memory (ROM), an

erasable programmable read-only memory (EPROM or Flash memory), an optical fiber, a portable compact disc read-only memory (CD-ROM), an optical storage device, a transmission media such as those supporting the Internet or an intranet, or a magnetic storage device. The computer-usable or computer-readable medium may also be paper or another suitable medium upon which the program is printed, as the program can be electronically captured, via, for instance, optical scanning of the paper or other medium, then compiled, interpreted, or otherwise processed in a suitable manner, if necessary, and then stored in a computer memory. In the context of this document, a computer-usable or computer-readable medium may be any medium that can contain, store, communicate, propagate, or transport the program for use by or in connection with the instruction execution system, apparatus, or device. The computer-usable medium may include a propagated data signal with the computer-usable program code embodied therewith, either in baseband or as part of a carrier wave. The computer usable program code may be transmitted using any appropriate medium, including but not limited to the Internet, wireline, optical fiber cable, RF, etc.

[00284] Computer program code for carrying out operations of the present disclosure may be written in an object-oriented programming language such as Java, Smalltalk, C++ or the like. However, the computer program code for carrying out operations of the present disclosure may also be written in conventional procedural programming languages, such as the "C" programming language or similar programming languages. The program code may execute entirely on the user's computer, partly on the user's computer, as a stand-alone software package, partly on the user's computer and partly on a remote computer or entirely on the remote computer or server. In the latter scenario, the remote computer may be connected to the user's computer through a local area network / a wide area network / the Internet (e.g., network 14).

[00285] The present disclosure is described with reference to flowchart illustrations and/or block diagrams of methods, apparatus (systems) and computer

program products according to embodiments of the disclosure. It will be understood that each block of the flowchart illustrations and/or block diagrams, and combinations of blocks in the flowchart illustrations and/or block diagrams, may be implemented by computer program instructions. These computer program instructions may be provided to a processor of a general-purpose computer / special purpose computer / other programmable data processing apparatus, such that the instructions, which execute via the processor of the computer or other programmable data processing apparatus, create means for implementing the functions/acts specified in the flowchart and/or block diagram block or blocks.

[00286] These computer program instructions may also be stored in a computer-readable memory that may direct a computer or other programmable data processing apparatus to function in a particular manner, such that the instructions stored in the computer-readable memory produce an article of manufacture including instruction means which implement the function/act specified in the flowchart and/or block diagram block or blocks.

[00287] The computer program instructions may also be loaded onto a computer or other programmable data processing apparatus to cause a series of operational steps to be performed on the computer or other programmable apparatus to produce a computer-implemented process such that the instructions which execute on the computer or other programmable apparatus provide steps for implementing the functions/acts specified in the flowchart and/or block diagram block or blocks.

[00288] The flowcharts and block diagrams in the figures may illustrate the architecture, functionality, and operation of possible implementations of systems, methods and computer program products according to various embodiments of the present disclosure. In this regard, each block in the flowchart or block diagrams may represent a module, segment, or portion of code, which comprises one or more executable instructions for implementing the specified logical function(s). It should also be noted

that, in some alternative implementations, the functions noted in the block may occur out of the order noted in the figures. For example, two blocks shown in succession may, in fact, be executed substantially concurrently, or the blocks may sometimes be executed in the reverse order, depending upon the functionality involved. It will also be noted that each block of the block diagrams and/or flowchart illustrations, and combinations of blocks in the block diagrams and/or flowchart illustrations, may be implemented by special purpose hardware-based systems that perform the specified functions or acts, or combinations of special purpose hardware and computer instructions.

[00289] The terminology used herein is for the purpose of describing particular embodiments only and is not intended to be limiting of the disclosure. As used herein, the singular forms "a", "an" and "the" are intended to include the plural forms as well, unless the context clearly indicates otherwise. It will be further understood that the terms "comprises" and/or "comprising," when used in this specification, specify the presence of stated features, integers, steps, operations, elements, and/or components, but do not preclude the presence or addition of one or more other features, integers, steps, operations, elements, components, and/or groups thereof.

[00290] The corresponding structures, materials, acts, and equivalents of all means or step plus function elements in the claims below are intended to include any structure, material, or act for performing the function in combination with other claimed elements as specifically claimed. The description of the present disclosure has been presented for purposes of illustration and description, but is not intended to be exhaustive or limited to the disclosure in the form disclosed. Many modifications and variations will be apparent to those of ordinary skill in the art without departing from the scope and spirit of the disclosure. The embodiment was chosen and described in order to best explain the principles of the disclosure and the practical application, and to enable others of ordinary skill in the art to understand the disclosure for various embodiments with various modifications as are suited to the particular use contemplated.

[00291] A number of implementations have been described. Having thus described the disclosure of the present application in detail and by reference to embodiments thereof, it will be apparent that modifications and variations are possible without departing from the scope of the disclosure defined in the appended claims.

What Is Claimed Is:

Concept 2

1. A computer-implemented method, executed on a computing device, comprising:

establishing connectivity with a plurality of security-relevant subsystems within a computing platform; and

mapping one or more data fields of a unified platform to one or more data fields of each of the plurality of security-relevant subsystems.

2. The computer-implemented method of claim 1 wherein mapping one or more data fields of a unified platform to one or more data fields of each of the plurality of security-relevant subsystems includes:

mapping one or more data fields within a query structure of the unified platform to one or more data fields within a query structure of each of the plurality of security-relevant subsystems.

3. The computer-implemented method of claim 1 wherein mapping one or more data fields of a unified platform to one or more data fields of each of the plurality of security-relevant subsystems includes:

mapping one or more data fields within a result set structure of each of the plurality of security-relevant subsystems to one or more data fields within a result set structure of the unified platform.

4. The computer-implemented method of claim 1 wherein mapping one or more data fields of a unified platform to one or more data fields of each of the plurality of security-relevant subsystems includes:

mapping one or more data fields of the unified platform to one or more data fields of each of the plurality of security-relevant subsystems at a defined periodicity.

5. The computer-implemented method of claim 1 wherein mapping one or more data fields of a unified platform to one or more data fields of each of the plurality of security-relevant subsystems includes:

proactively mapping one or more data fields of the unified platform to one or more data fields of each of the plurality of security-relevant subsystems.

6. The computer-implemented method of claim 1 wherein mapping one or more data fields of a unified platform to one or more data fields of each of the plurality of security-relevant subsystems includes:

reactively mapping one or more data fields of the unified platform to one or more data fields of each of the plurality of security-relevant subsystems.

7. The computer-implemented method of claim 1 further comprising:

defining a unified query on a unified platform concerning the plurality of security-relevant subsystems;

denormalizing the unified query to define a subsystem-specific query for each of the plurality of security-relevant subsystems, thus defining a plurality of subsystem-specific queries; and

providing the plurality of subsystem-specific queries to the plurality of security-relevant subsystems.

8. The computer-implemented method of claim 7 further comprising:

receiving a plurality of subsystem-specific results sets from the plurality of security-relevant subsystems that were generated in response to the plurality of subsystem-specific queries.

9. The computer-implemented method of claim 8 further comprising:

normalizing the plurality of subsystem-specific results sets received from the plurality of security-relevant subsystems to define a unified result set.

10. The computer-implemented method of claim 9 further comprising:

providing the unified result set to a third-party.

11. A computer program product residing on a computer readable medium having a plurality of instructions stored thereon which, when executed by a processor, cause the processor to perform operations comprising:

establishing connectivity with a plurality of security-relevant subsystems within a computing platform; and

mapping one or more data fields of a unified platform to one or more data fields of each of the plurality of security-relevant subsystems.

12. The computer program product of claim 11 wherein mapping one or more data fields of a unified platform to one or more data fields of each of the plurality of security-relevant subsystems includes:

mapping one or more data fields within a query structure of the unified platform to one or more data fields within a query structure of each of the plurality of security-relevant subsystems.

13. The computer program product of claim 11 wherein mapping one or more data fields of a unified platform to one or more data fields of each of the plurality of security-relevant subsystems includes:

mapping one or more data fields within a result set structure of each of the plurality of security-relevant subsystems to one or more data fields within a result set structure of the unified platform.

14. The computer program product of claim 11 wherein mapping one or more data fields of a unified platform to one or more data fields of each of the plurality of security-relevant subsystems includes:

mapping one or more data fields of the unified platform to one or more data fields of each of the plurality of security-relevant subsystems at a defined periodicity.

15. The computer program product of claim 11 wherein mapping one or more data fields of a unified platform to one or more data fields of each of the plurality of security-relevant subsystems includes:

proactively mapping one or more data fields of the unified platform to one or more data fields of each of the plurality of security-relevant subsystems.

16. The computer program product of claim 11 wherein mapping one or more data fields of a unified platform to one or more data fields of each of the plurality of security-relevant subsystems includes:

reactively mapping one or more data fields of the unified platform to one or more data fields of each of the plurality of security-relevant subsystems.

17. The computer program product of claim 11 further comprising:

defining a unified query on a unified platform concerning the plurality of security-relevant subsystems;

denormalizing the unified query to define a subsystem-specific query for each of the plurality of security-relevant subsystems, thus defining a plurality of subsystem-specific queries; and

providing the plurality of subsystem-specific queries to the plurality of security-relevant subsystems.

18. The computer program product of claim 17 further comprising:

receiving a plurality of subsystem-specific results sets from the plurality of security-relevant subsystems that were generated in response to the plurality of subsystem-specific queries.

19. The computer program product of claim 18 further comprising:

normalizing the plurality of subsystem-specific results sets received from the plurality of security-relevant subsystems to define a unified result set.

20. The computer program product of claim 19 further comprising:

providing the unified result set to a third-party.

21. A computing system including a processor and memory configured to perform operations comprising:

establishing connectivity with a plurality of security-relevant subsystems within a computing platform; and

mapping one or more data fields of a unified platform to one or more data fields of each of the plurality of security-relevant subsystems.

22. The computing system of claim 21 wherein mapping one or more data fields of a unified platform to one or more data fields of each of the plurality of security-relevant subsystems includes:

mapping one or more data fields within a query structure of the unified platform to one or more data fields within a query structure of each of the plurality of security-relevant subsystems.

23. The computing system of claim 21 wherein mapping one or more data fields of a unified platform to one or more data fields of each of the plurality of security-relevant subsystems includes:

mapping one or more data fields within a result set structure of each of the plurality of security-relevant subsystems to one or more data fields within a result set structure of the unified platform.

24. The computing system of claim 21 wherein mapping one or more data fields of a unified platform to one or more data fields of each of the plurality of security-relevant subsystems includes:

mapping one or more data fields of the unified platform to one or more data fields of each of the plurality of security-relevant subsystems at a defined periodicity.

25. The computing system of claim 21 wherein mapping one or more data fields of a unified platform to one or more data fields of each of the plurality of security-relevant subsystems includes:

proactively mapping one or more data fields of the unified platform to one or more data fields of each of the plurality of security-relevant subsystems.

26. The computing system of claim 21 wherein mapping one or more data fields of a unified platform to one or more data fields of each of the plurality of security-relevant subsystems includes:

reactively mapping one or more data fields of the unified platform to one or more data fields of each of the plurality of security-relevant subsystems.

27. The computing system of claim 21 further comprising:

defining a unified query on a unified platform concerning the plurality of security-relevant subsystems;

denormalizing the unified query to define a subsystem-specific query for each of the plurality of security-relevant subsystems, thus defining a plurality of subsystem-specific queries; and

providing the plurality of subsystem-specific queries to the plurality of security-relevant subsystems.

28. The computing system of claim 27 further comprising:

receiving a plurality of subsystem-specific results sets from the plurality of security-relevant subsystems that were generated in response to the plurality of subsystem-specific queries.

29. The computing system of claim 28 further comprising:

normalizing the plurality of subsystem-specific results sets received from the plurality of security-relevant subsystems to define a unified result set.

30. The computing system of claim 29 further comprising:

providing the unified result set to a third-party.

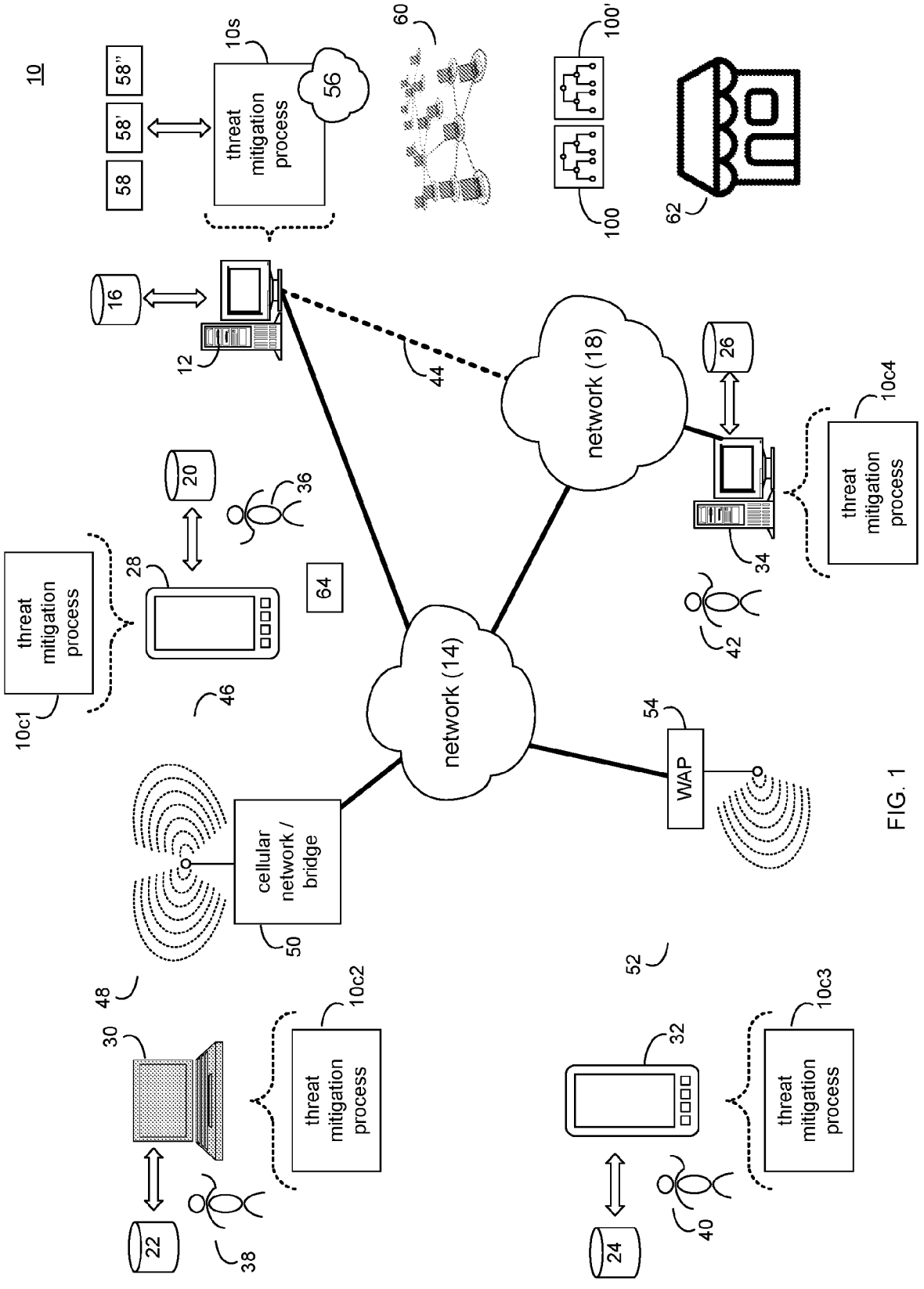

10
58
58'
58"
threat mitigation process
10s
56
60
100'
100
62
16
12
44
network (18)
26
10c4
threat mitigation process
20
36
28
64
threat mitigation process
10c1
46
34
42
network (14)
54
WAP
cellular network / bridge
50
48
30
10c2
threat mitigation process
52
32
10c3
threat mitigation process
22
38
24
40

FIG. 1

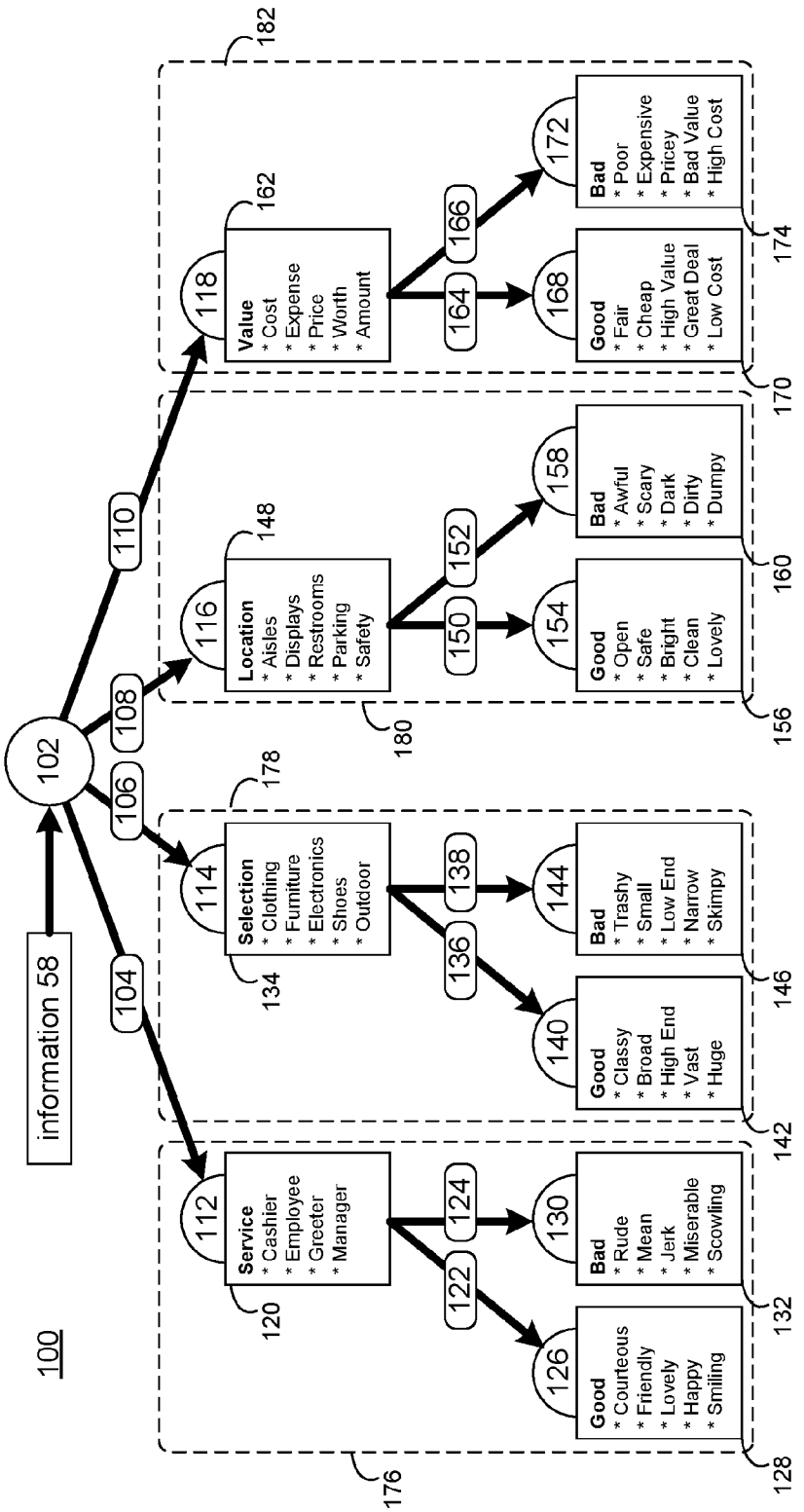

100
information 58
102
104
106
108
110
112
Service
* Cashier
* Employee
* Greeter
* Manager
120
114
Selection
* Clothing
* Furniture
* Electronics
* Shoes
* Outdoor
134
116
Location
* Aisles
* Displays
* Restrooms
* Parking
* Safety
148
118
Value
* Cost
* Expense
* Price
* Worth
* Amount
162
122
124
126
Good
* Courteous
* Friendly
* Lovely
* Happy
* Smiling
128
130
Bad
* Rude
* Mean
* Jerk
* Miserable
* Scowling
132
136
138
140
Good
* Classy
* Broad
* High End
* Vast
* Huge
142
144
Bad
* Trashy
* Small
* Low End
* Narrow
* Skimpy
146
150
152
154
Good
* Open
* Safe
* Bright
* Clean
* Lovely
156
158
Bad
* Awful
* Scary
* Dark
* Dirty
* Dumpy
160
164
166
168
Good
* Fair
* Cheap
* High Value
* Great Deal
* Low Cost
170
172
Bad
* Poor
* Expensive
* Pricey
* Bad Value
* High Cost
174
176
178
180
182

FIG. 2

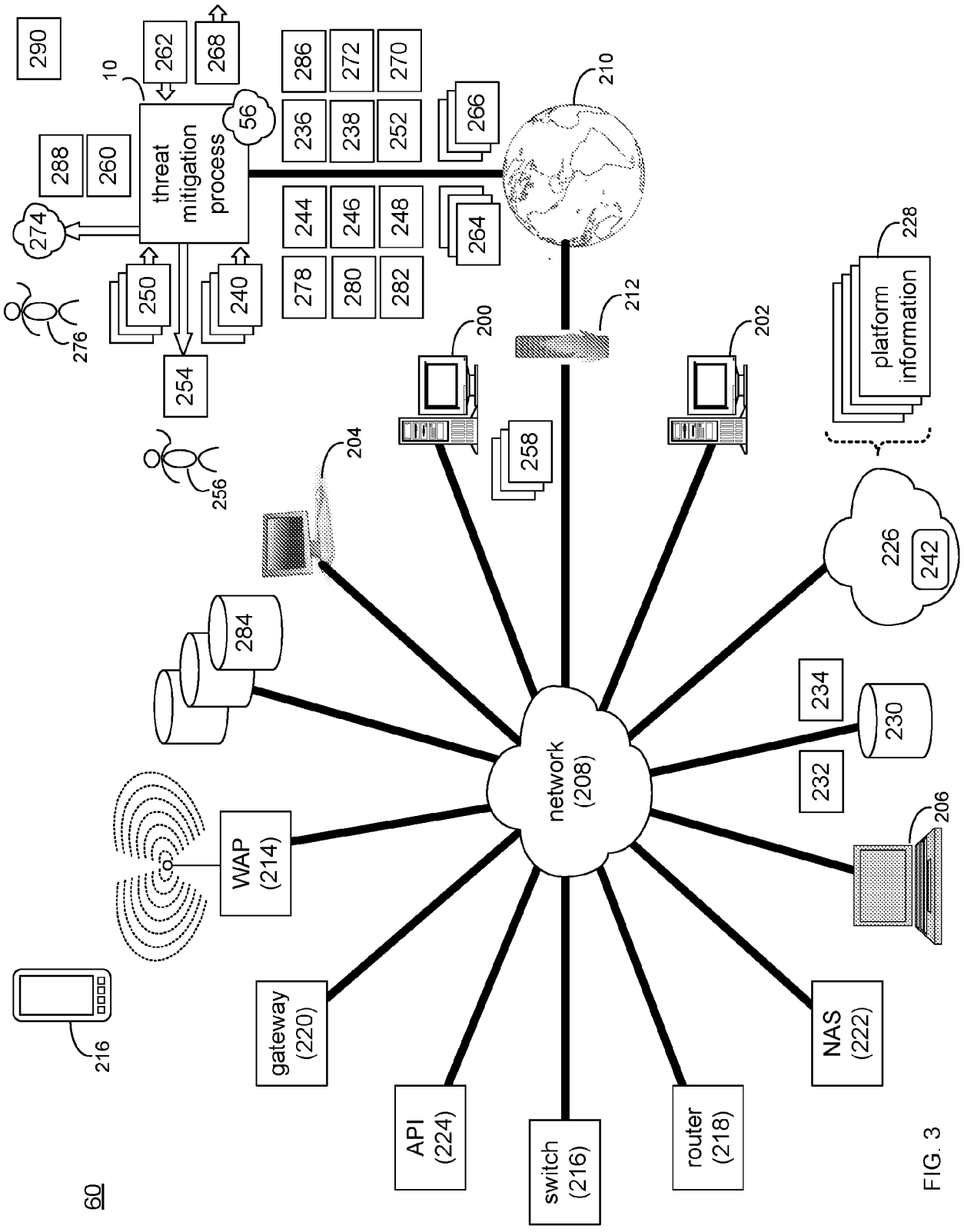

60
216
WAP (214)
gateway (220)
API (224)
switch (216)
router (218)
NAS (222)
network (208)
206
232
234
230
226
242
228
platform information
202
212
210
200
204
284
258
254
256
276
250
240
274
288
260
290
10
262
268
threat mitigation process
56
278
280
282
244
246
248
264
236
238
252
266
286
272
270

FIG. 3

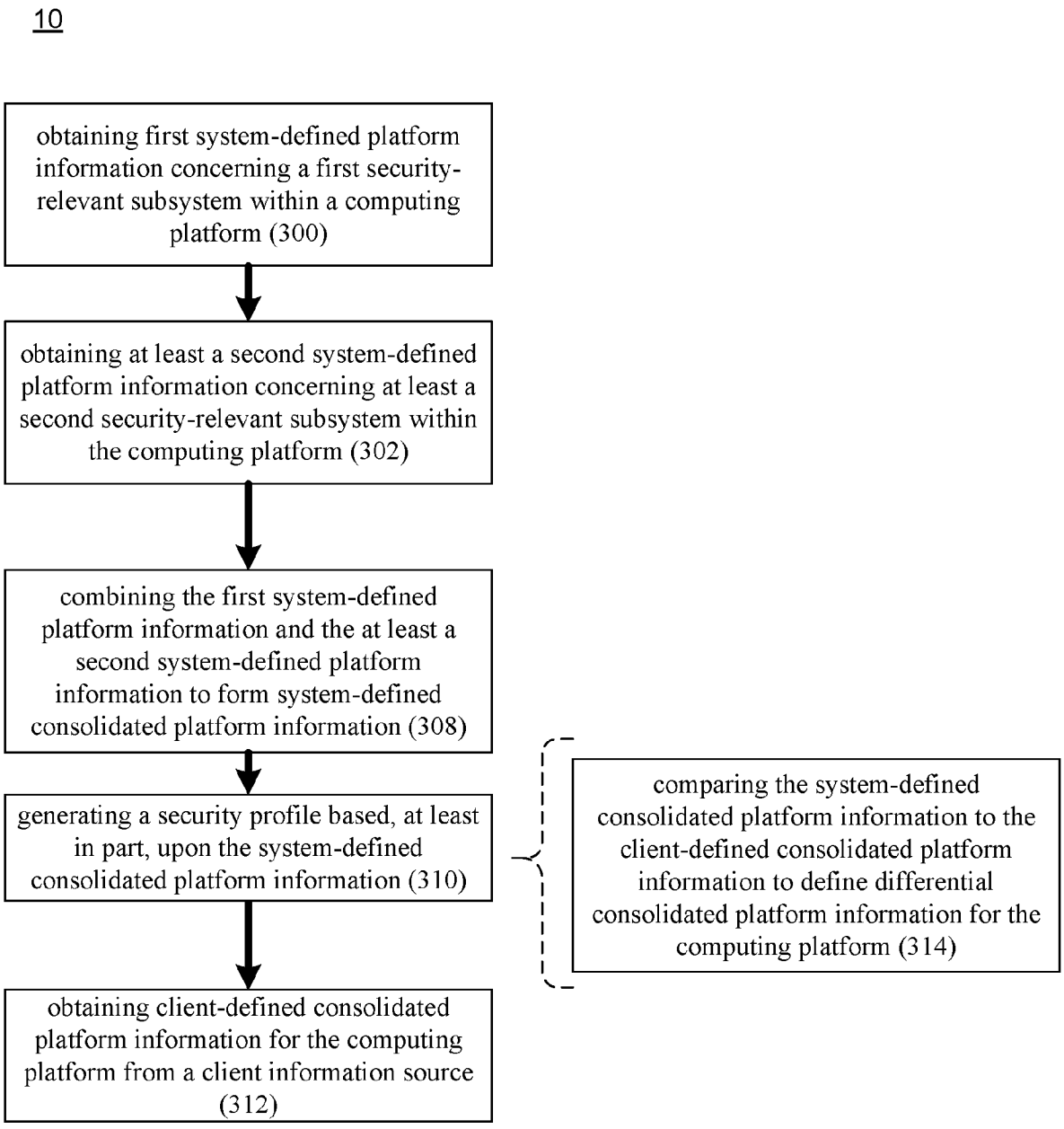

10
obtaining first system-defined platform information concerning a first security-relevant subsystem within a computing platform (300)
obtaining at least a second system-defined platform information concerning at least a second security-relevant subsystem within the computing platform (302)
combining the first system-defined platform information and the at least a second system-defined platform information to form system-defined consolidated platform information (308)
generating a security profile based, at least in part, upon the system-defined consolidated platform information (310)
obtaining client-defined consolidated platform information for the computing platform from a client information source (312)
comparing the system-defined consolidated platform information to the client-defined consolidated platform information to define differential consolidated platform information for the computing platform (314)

FIG. 4


350
Summary Overview
605
1000
Visibility
237
400
Log Source Coverage
Log Source Diversity
Kill Chain Coverage
Threat Context
Tool Efficacy
149
200
SIEM Health
SIEM Maturity
Team Performance
219
400
False Positive Rate
Anomalous Safe Rate
No Response Rate
Mean Time to Resolve (MTTR)
RELIAQUEST
MODEL INDEX

FIG. 5

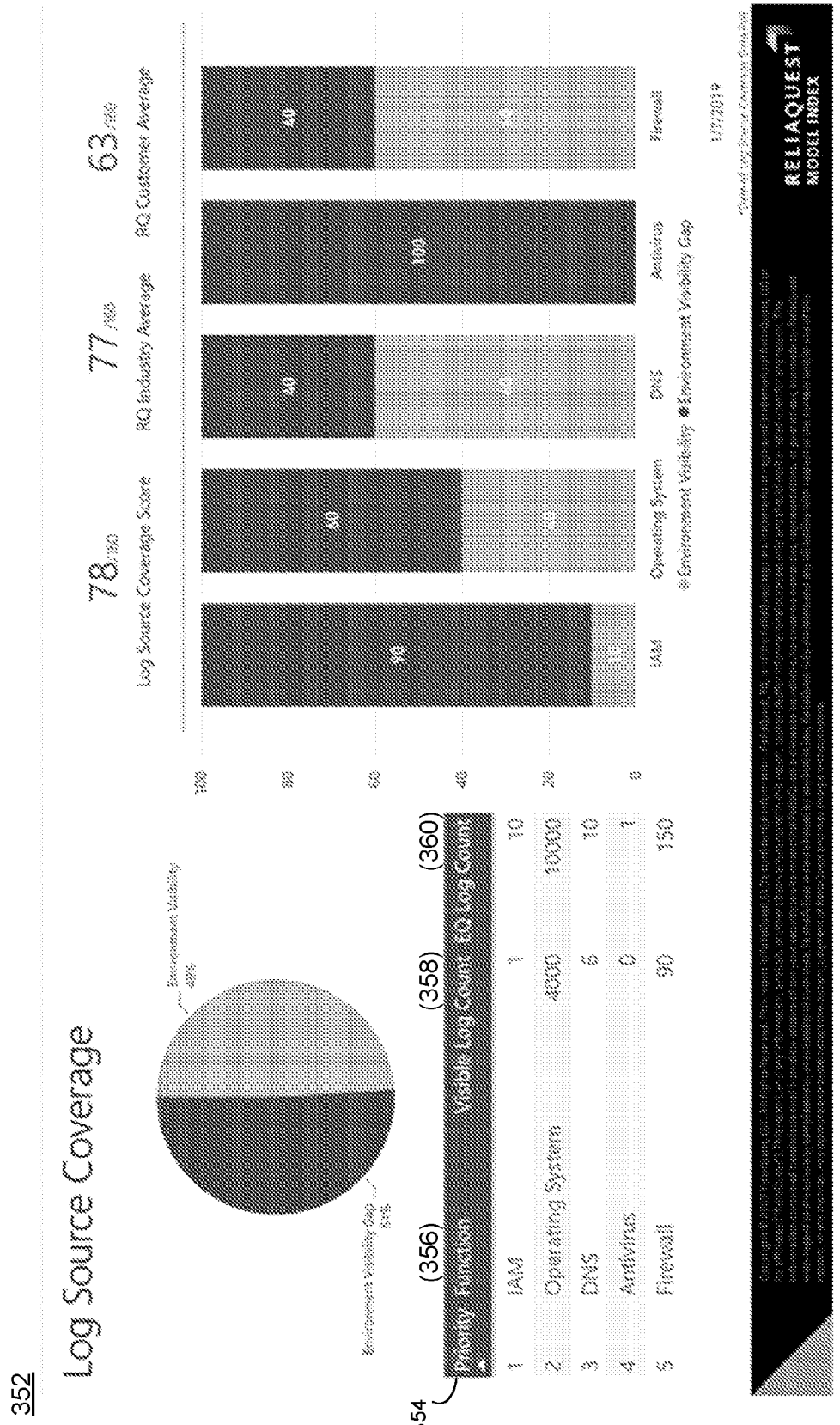

352
Log Source Coverage
Environment Visibility
48%
Environment Visibility Gap
52%
354
(356) Priority Function
(358) Visibility
(360) Log Count
1 IAM 1 10
2 Operating System 4000 10000
3 DNS 6 10
4 Antivirus 0 1
5 Firewall 90 150
78/100
Log Source Coverage Score
77/100
RQ Industry Average
63/100
RQ Customer Average
50
50
50
50
50
50
100
50
50
100
IAM
Operating System
DNS
Antivirus
Firewall
Environment Visibility
Environment Visibility Gap
RELIAQUEST
MODEL INDEX

FIG. 6

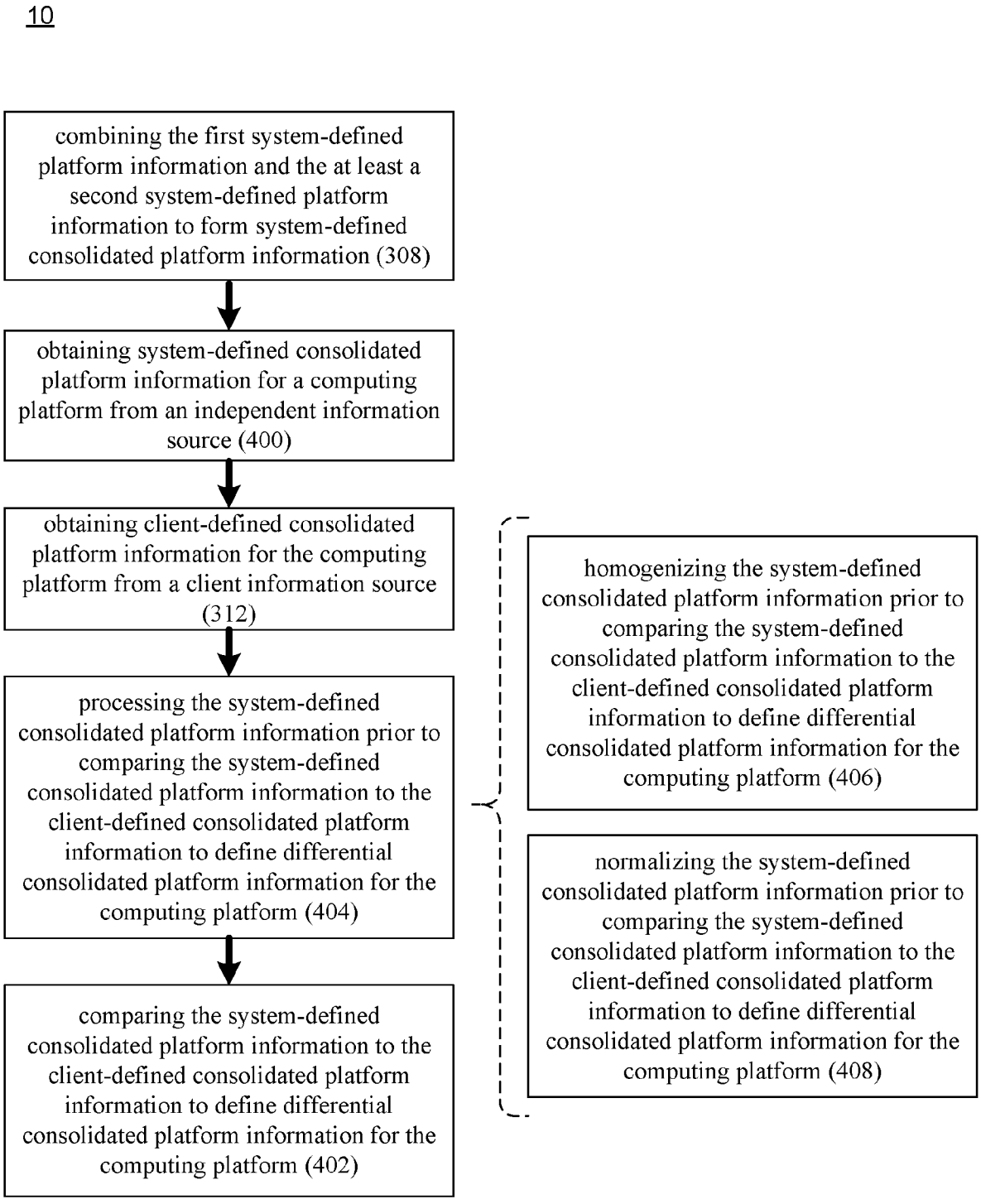

10
combining the first system-defined platform information and the at least a second system-defined platform information to form system-defined consolidated platform information (308)
obtaining system-defined consolidated platform information for a computing platform from an independent information source (400)
obtaining client-defined consolidated platform information for the computing platform from a client information source (312)
processing the system-defined consolidated platform information prior to comparing the system-defined consolidated platform information to the client-defined consolidated platform information to define differential consolidated platform information for the computing platform (404)
comparing the system-defined consolidated platform information to the client-defined consolidated platform information to define differential consolidated platform information for the computing platform (402)
homogenizing the system-defined consolidated platform information prior to comparing the system-defined consolidated platform information to the client-defined consolidated platform information to define differential consolidated platform information for the computing platform (406)
normalizing the system-defined consolidated platform information prior to comparing the system-defined consolidated platform information to the client-defined consolidated platform information to define differential consolidated platform information for the computing platform (408)

FIG. 7

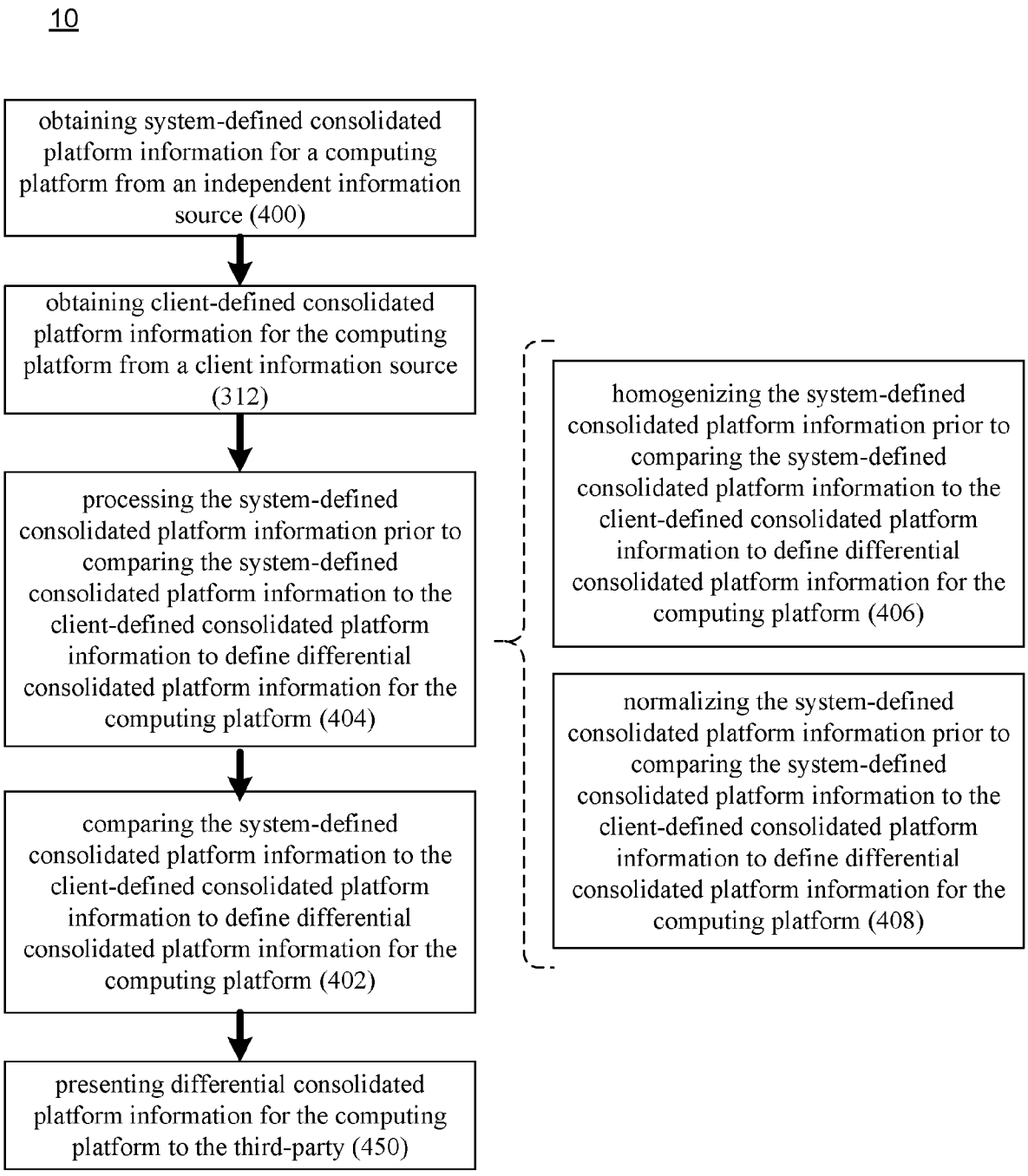

10
obtaining system-defined consolidated platform information for a computing platform from an independent information source (400)
obtaining client-defined consolidated platform information for the computing platform from a client information source (312)
processing the system-defined consolidated platform information prior to comparing the system-defined consolidated platform information to the client-defined consolidated platform information to define differential consolidated platform information for the computing platform (404)
comparing the system-defined consolidated platform information to the client-defined consolidated platform information to define differential consolidated platform information for the computing platform (402)
presenting differential consolidated platform information for the computing platform to the third-party (450)
homogenizing the system-defined consolidated platform information prior to comparing the system-defined consolidated platform information to the client-defined consolidated platform information to define differential consolidated platform information for the computing platform (406)
normalizing the system-defined consolidated platform information prior to comparing the system-defined consolidated platform information to the client-defined consolidated platform information to define differential consolidated platform information for the computing platform (408)

FIG. 8

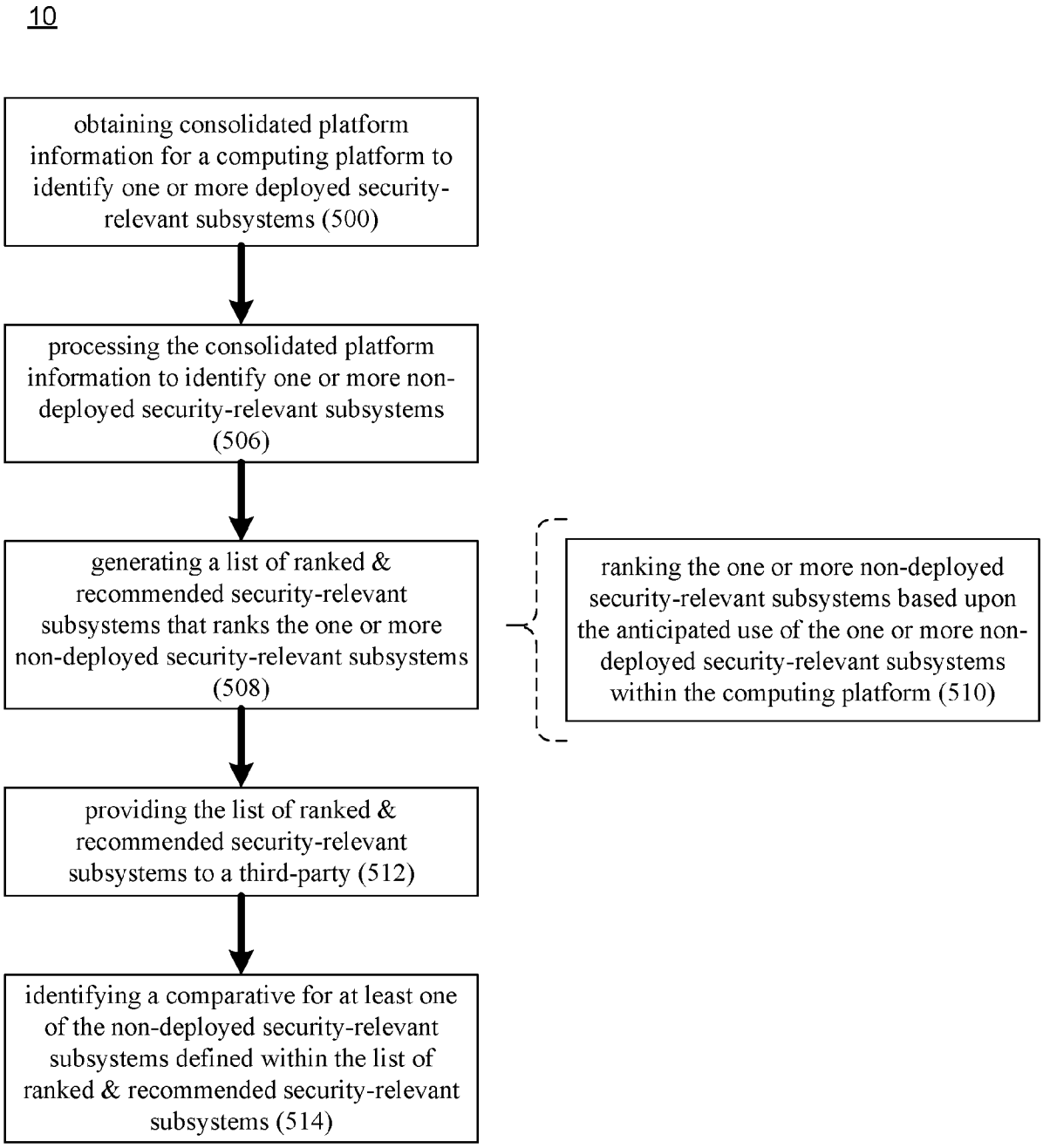

10
obtaining consolidated platform information for a computing platform to identify one or more deployed security-relevant subsystems (500)
processing the consolidated platform information to identify one or more non-deployed security-relevant subsystems (506)
generating a list of ranked & recommended security-relevant subsystems that ranks the one or more non-deployed security-relevant subsystems (508)
ranking the one or more non-deployed security-relevant subsystems based upon the anticipated use of the one or more non-deployed security-relevant subsystems within the computing platform (510)
providing the list of ranked & recommended security-relevant subsystems to a third-party (512)
identifying a comparative for at least one of the non-deployed security-relevant subsystems defined within the list of ranked & recommended security-relevant subsystems (514)

FIG. 9

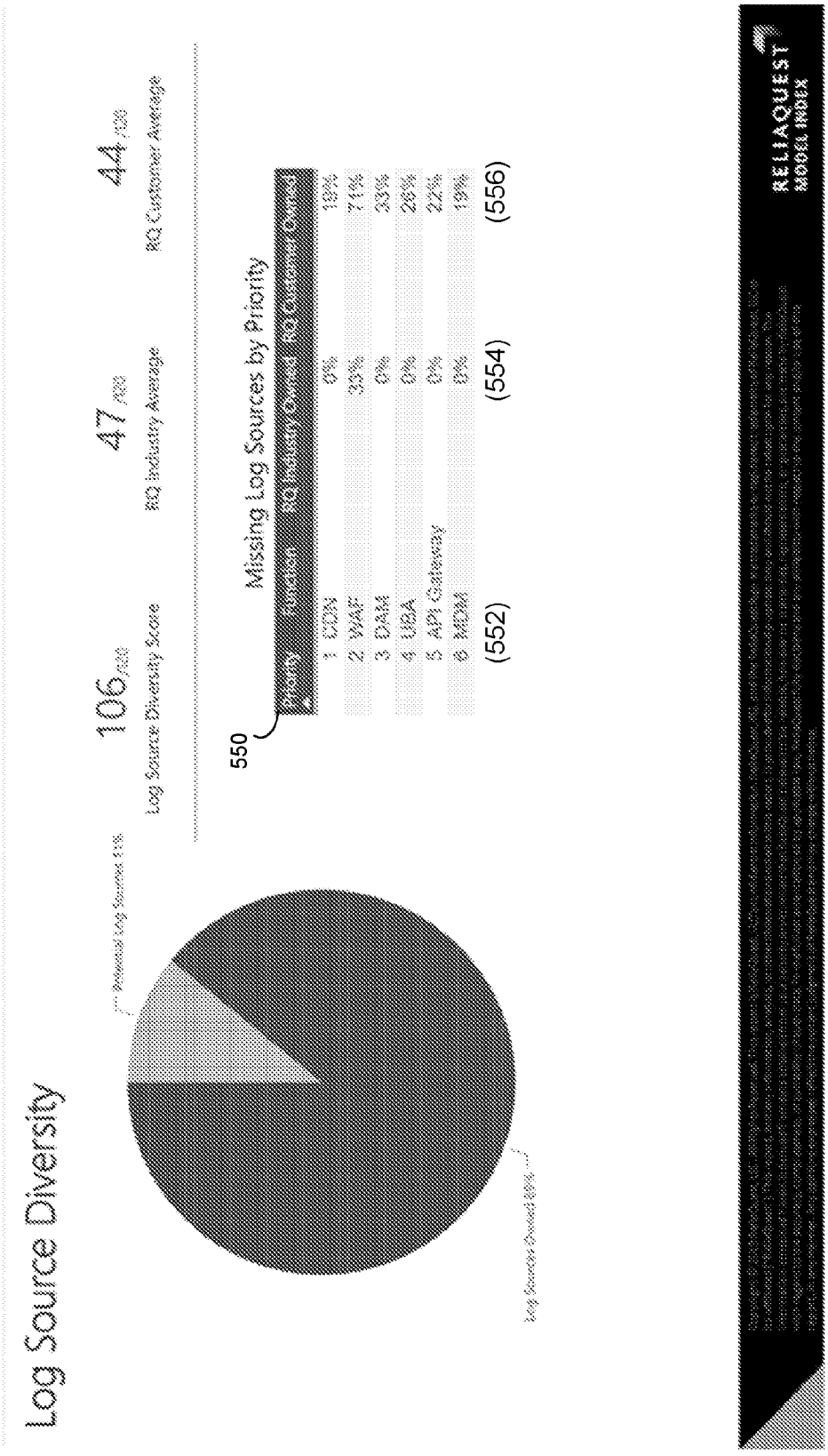

Log Source Diversity
Potential Log Sources 11%
Log Sources Owned 89%
106/100
Log Source Diversity Score
47/100
RQ Industry Average
44/100
RQ Customer Average
550
Missing Log Sources by Priority
Priority
Function
RQ Industry Owned
RQ Customer Owned
1 EDR 0% 18%
2 WAF 33% 71%
3 DAM 0% 33%
4 UBA 0% 26%
5 API Gateway 0% 22%
6 MDM 0% 19%
(552)
(554)
(556)
RELIAQUEST
MODEL INDEX

FIG. 10

10

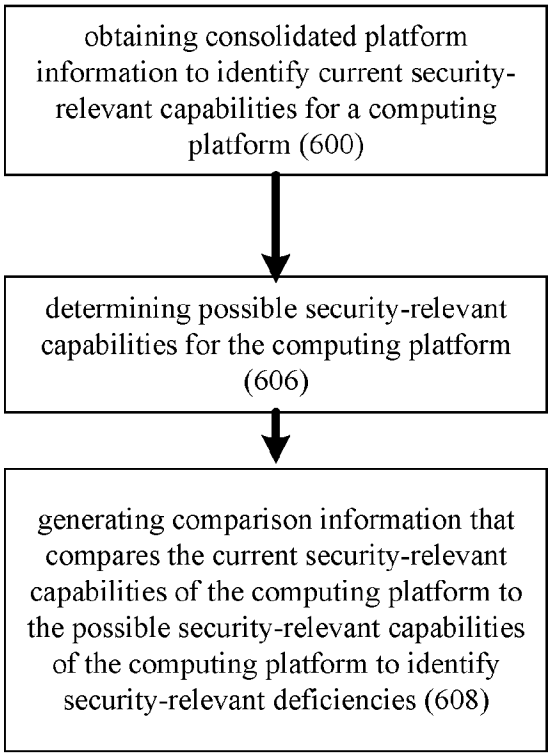

obtaining consolidated platform information to identify current security-relevant capabilities for a computing platform (600)
determining possible security-relevant capabilities for the computing platform (606)
generating comparison information that compares the current security-relevant capabilities of the computing platform to the possible security-relevant capabilities of the computing platform to identify security-relevant deficiencies (608)

FIG. 11

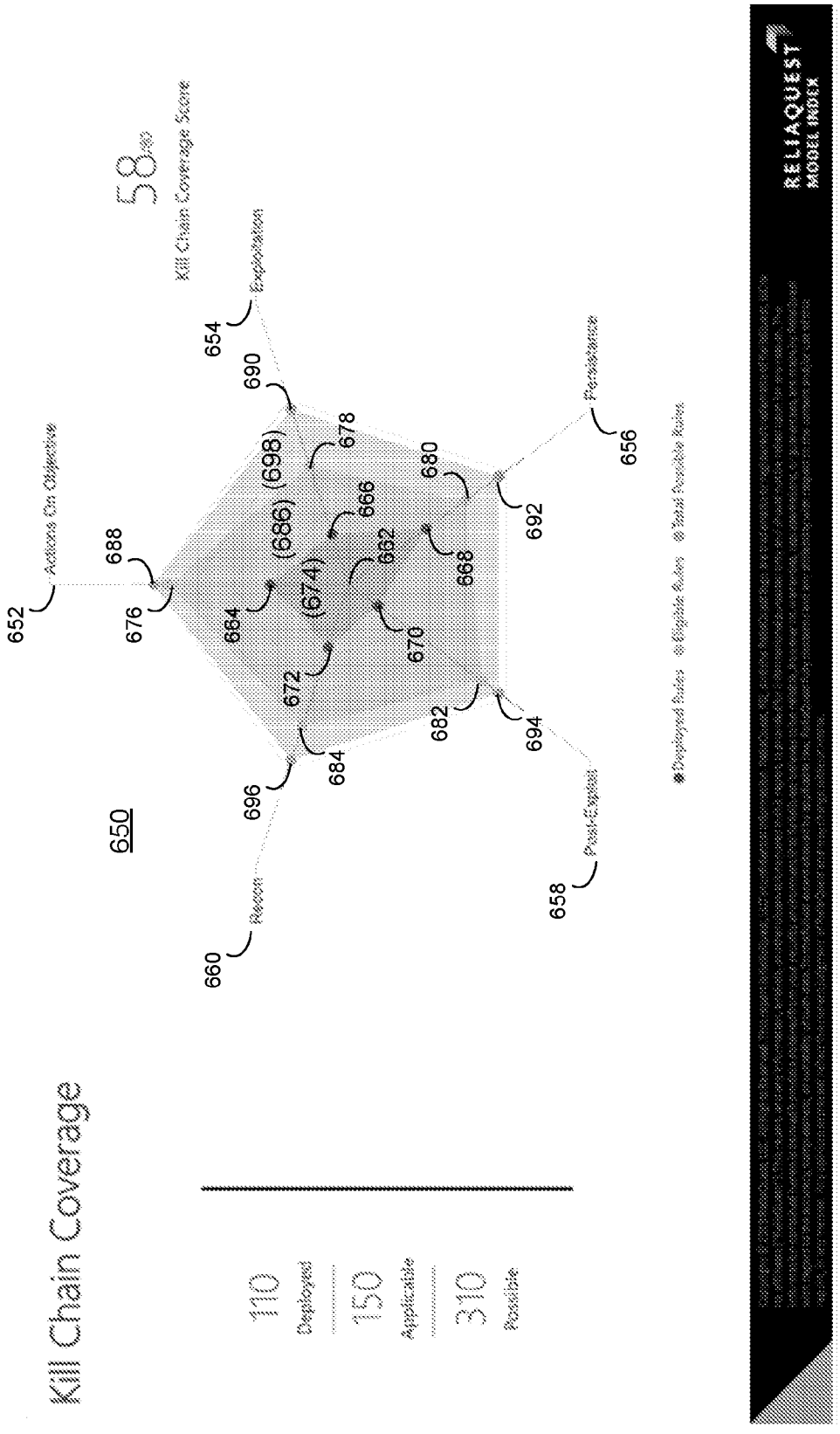

Kill Chain Coverage
110
Deployed
150
Applicable
310
Possible
650
58%
Kill Chain Coverage Score
652
Actions On Objective
654
Exploitation
656
Persistence
658
Post-Exploitation
660
Recon
676
688
690
678
680
692
694
696
684
682
672
664
(674)
(686)
(698)
666
662
668
670
RELIAQUEST
MODEL INDEX

FIG. 12

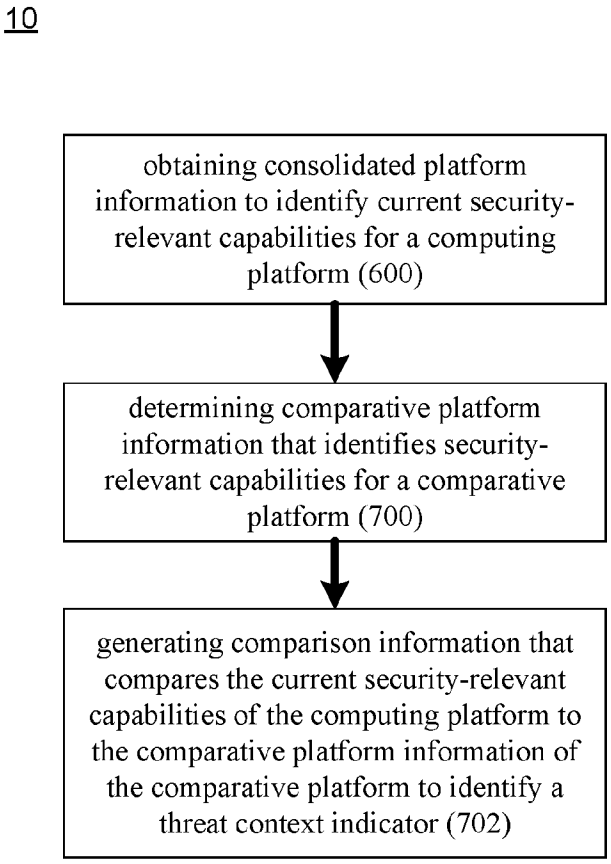

10
obtaining consolidated platform information to identify current security-relevant capabilities for a computing platform (600)
determining comparative platform information that identifies security-relevant capabilities for a comparative platform (700)
generating comparison information that compares the current security-relevant capabilities of the computing platform to the comparative platform information of the comparative platform to identify a threat context indicator (702)

FIG. 13

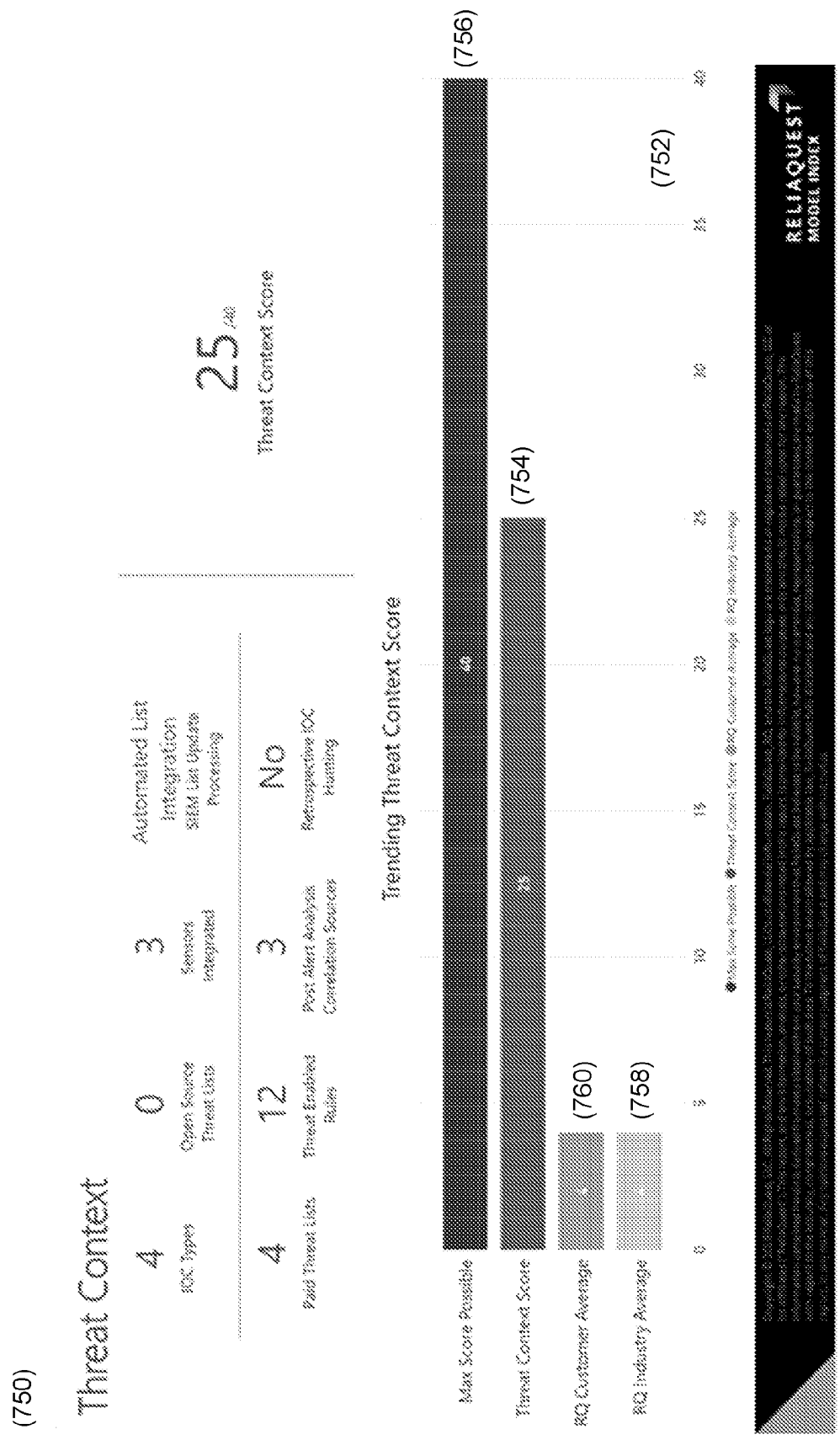

(750)
Threat Context
4
IOC Types
0
Open Source Threat Lists
3
Sensors Integrated
Automated List Integration SIEM List Update Processing
4
Paid Threat Lists
12
Threat Enabled Rules
3
Post Alert Analysis Correlation Sources
No
Retrospective IOC Hunting
25 /40
Threat Context Score
Trending Threat Context Score
Max Score Possible
Threat Context Score
RQ Customer Average
RQ Industry Average
(760)
(758)
(754)
(756)
(752)
RELIAQUEST MODEL INDEX

FIG. 14

10

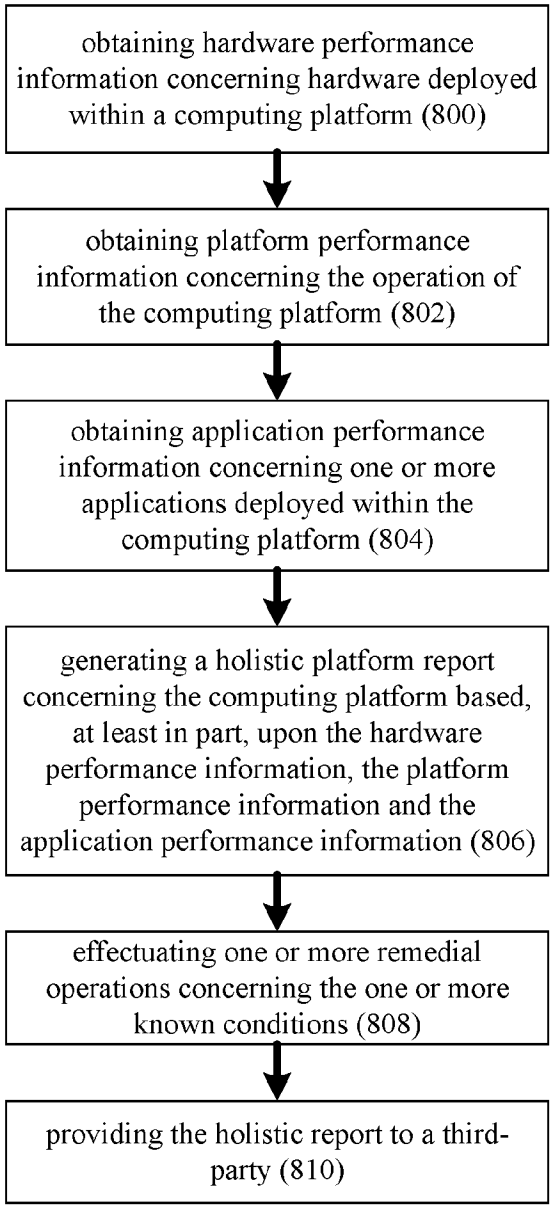

obtaining hardware performance information concerning hardware deployed within a computing platform (800)
obtaining platform performance information concerning the operation of the computing platform (802)
obtaining application performance information concerning one or more applications deployed within the computing platform (804)
generating a holistic platform report concerning the computing platform based, at least in part, upon the hardware performance information, the platform performance information and the application performance information (806)
effectuating one or more remedial operations concerning the one or more known conditions (808)
providing the holistic report to a third-party (810)

FIG. 15

352



352

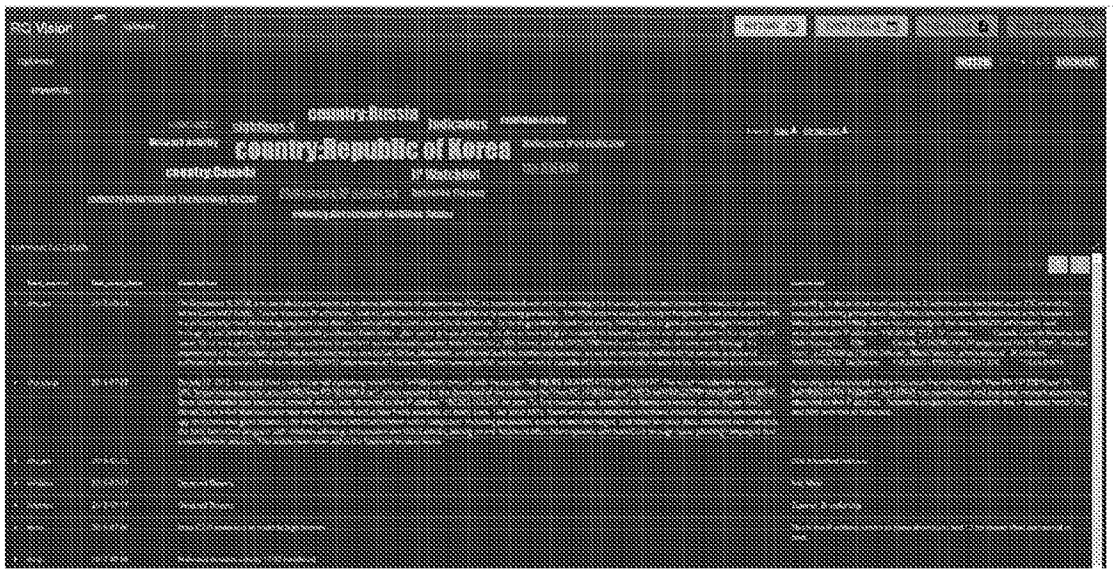


FIG. 16

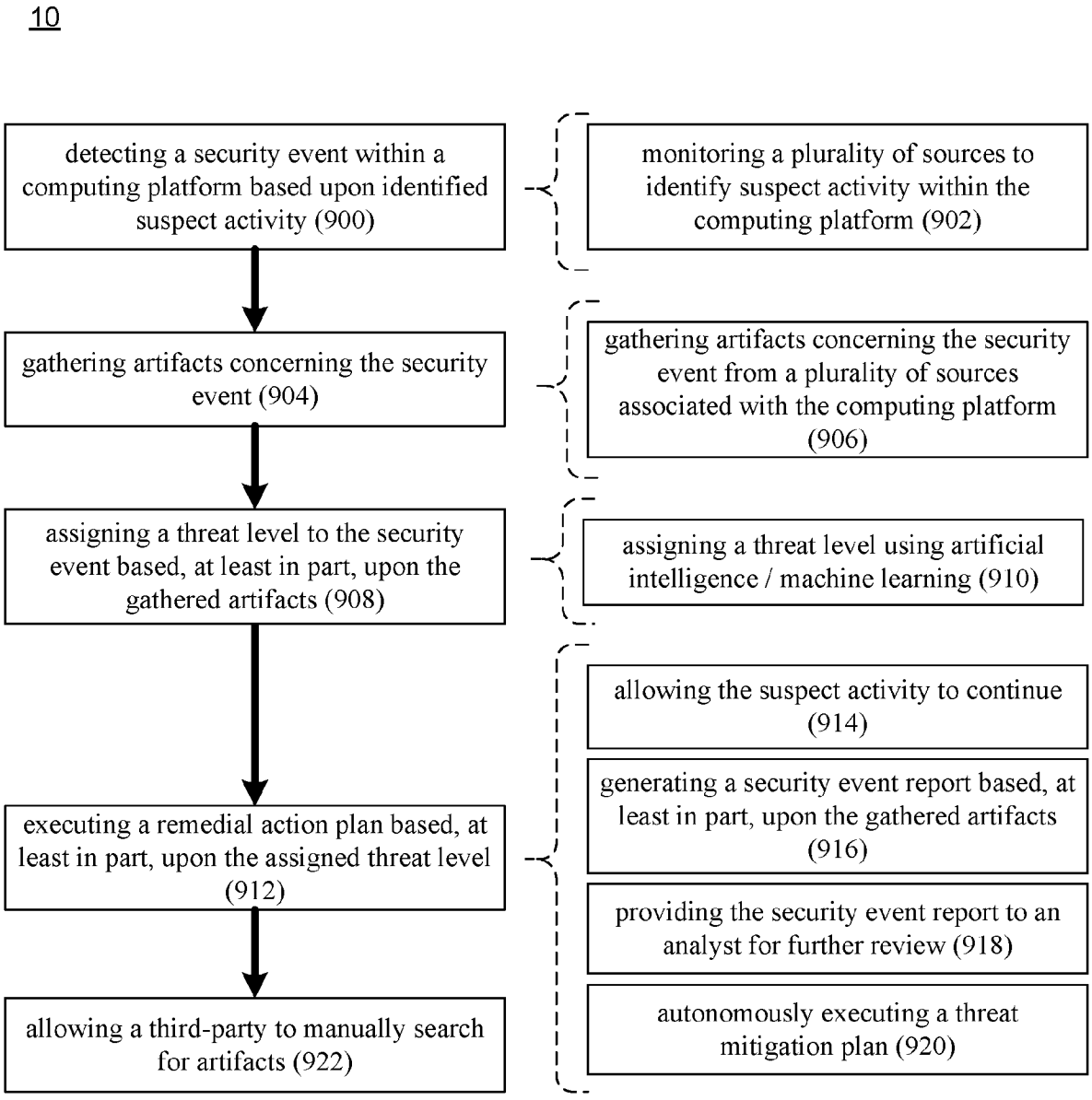

10
detecting a security event within a computing platform based upon identified suspect activity (900)
monitoring a plurality of sources to identify suspect activity within the computing platform (902)
gathering artifacts concerning the security event (904)
gathering artifacts concerning the security event from a plurality of sources associated with the computing platform (906)
assigning a threat level to the security event based, at least in part, upon the gathered artifacts (908)
assigning a threat level using artificial intelligence / machine learning (910)
executing a remedial action plan based, at least in part, upon the assigned threat level (912)
allowing the suspect activity to continue (914)
generating a security event report based, at least in part, upon the gathered artifacts (916)
providing the security event report to an analyst for further review (918)
autonomously executing a threat mitigation plan (920)
allowing a third-party to manually search for artifacts (922)

FIG. 17

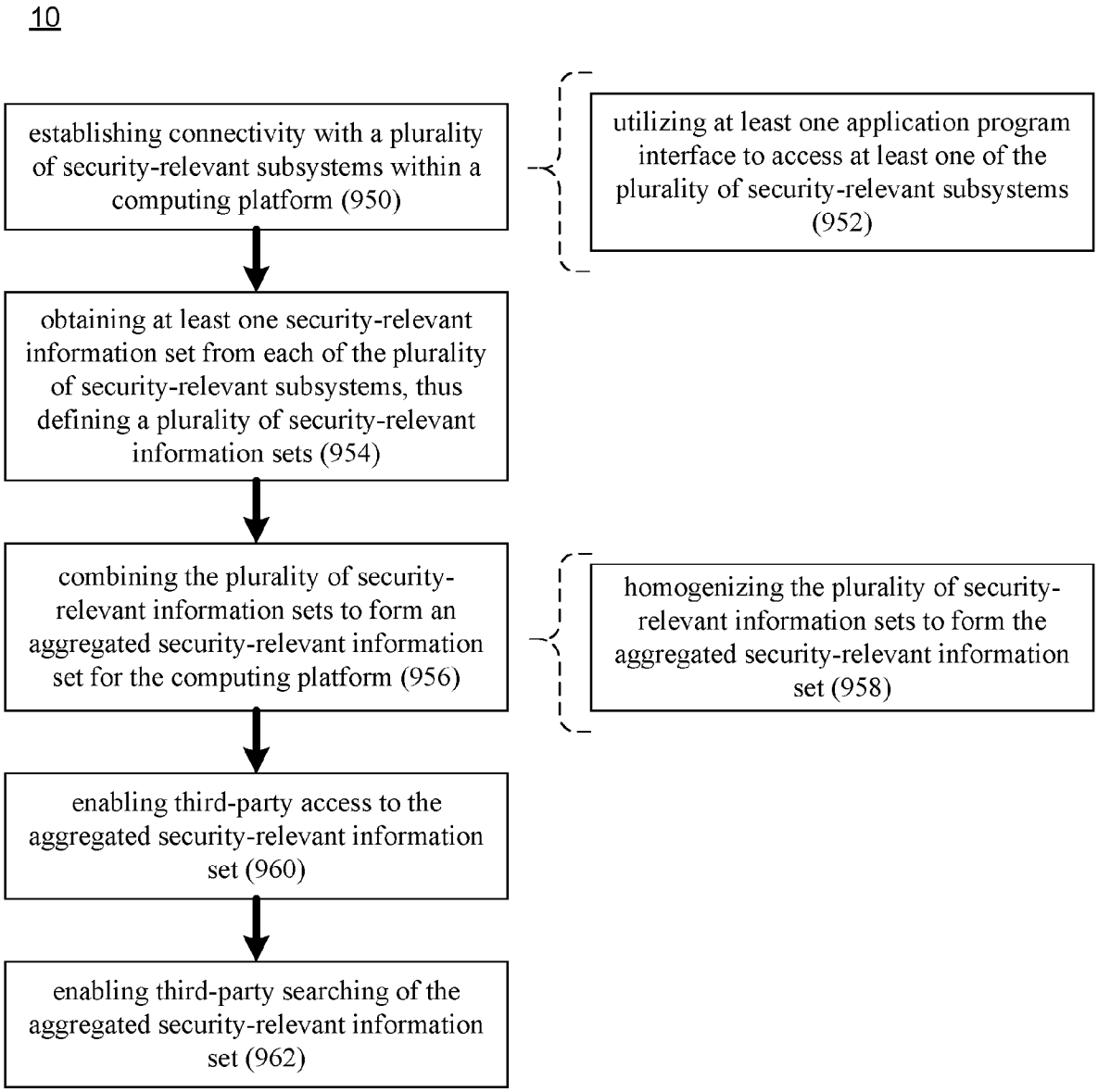

10
establishing connectivity with a plurality of security-relevant subsystems within a computing platform (950)
utilizing at least one application program interface to access at least one of the plurality of security-relevant subsystems (952)
obtaining at least one security-relevant information set from each of the plurality of security-relevant subsystems, thus defining a plurality of security-relevant information sets (954)
combining the plurality of security-relevant information sets to form an aggregated security-relevant information set for the computing platform (956)
homogenizing the plurality of security-relevant information sets to form the aggregated security-relevant information set (958)
enabling third-party access to the aggregated security-relevant information set (960)
enabling third-party searching of the aggregated security-relevant information set (962)

FIG. 18

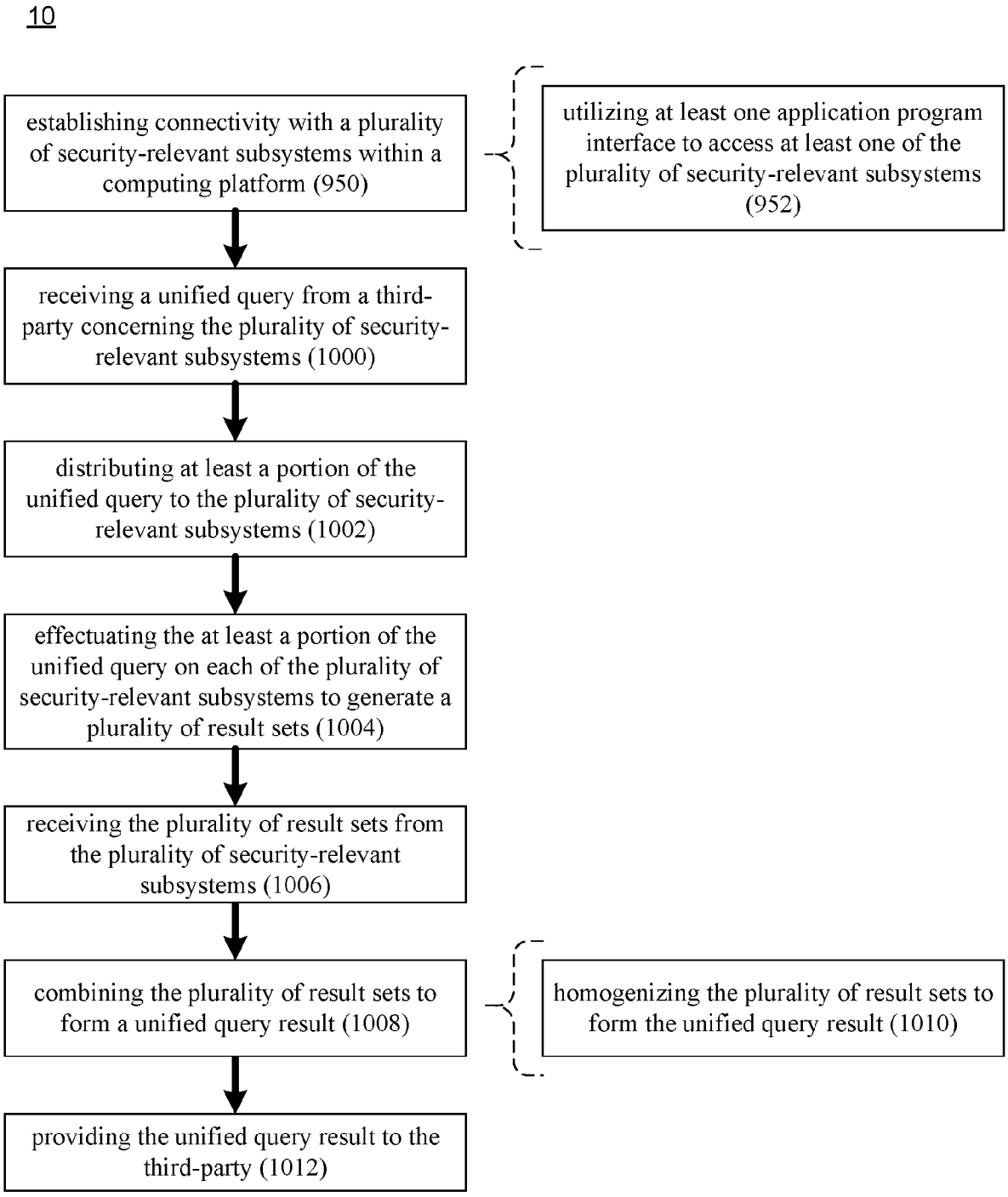

10
establishing connectivity with a plurality of security-relevant subsystems within a computing platform (950)
utilizing at least one application program interface to access at least one of the plurality of security-relevant subsystems (952)
receiving a unified query from a third-party concerning the plurality of security-relevant subsystems (1000)
distributing at least a portion of the unified query to the plurality of security-relevant subsystems (1002)
effectuating the at least a portion of the unified query on each of the plurality of security-relevant subsystems to generate a plurality of result sets (1004)
receiving the plurality of result sets from the plurality of security-relevant subsystems (1006)
combining the plurality of result sets to form a unified query result (1008)
homogenizing the plurality of result sets to form the unified query result (1010)
providing the unified query result to the third-party (1012)

FIG. 19

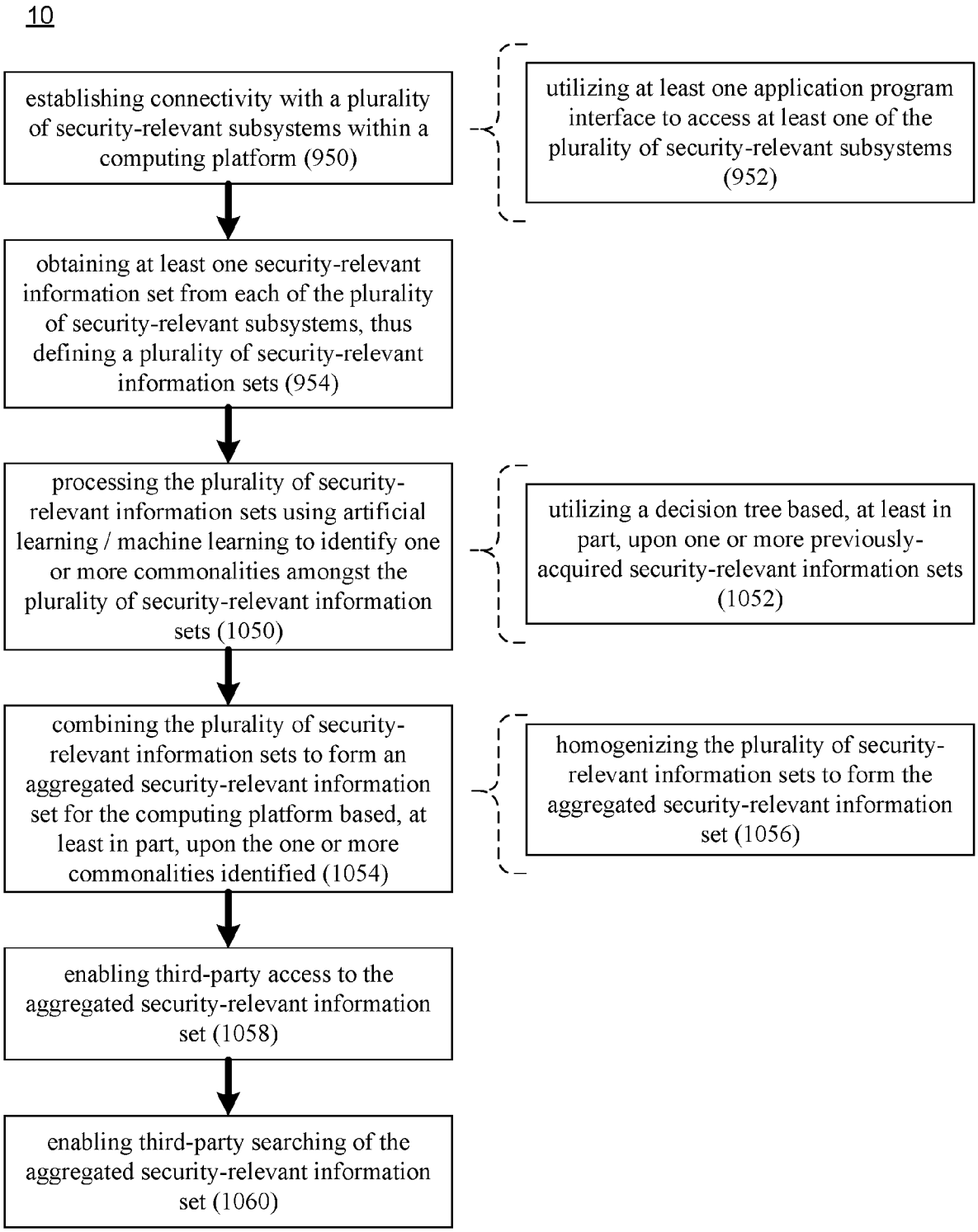

10
establishing connectivity with a plurality of security-relevant subsystems within a computing platform (950)
utilizing at least one application program interface to access at least one of the plurality of security-relevant subsystems (952)
obtaining at least one security-relevant information set from each of the plurality of security-relevant subsystems, thus defining a plurality of security-relevant information sets (954)
processing the plurality of security-relevant information sets using artificial learning / machine learning to identify one or more commonalities amongst the plurality of security-relevant information sets (1050)
utilizing a decision tree based, at least in part, upon one or more previously-acquired security-relevant information sets (1052)
combining the plurality of security-relevant information sets to form an aggregated security-relevant information set for the computing platform based, at least in part, upon the one or more commonalities identified (1054)
homogenizing the plurality of security-relevant information sets to form the aggregated security-relevant information set (1056)
enabling third-party access to the aggregated security-relevant information set (1058)
enabling third-party searching of the aggregated security-relevant information set (1060)

FIG. 20

10

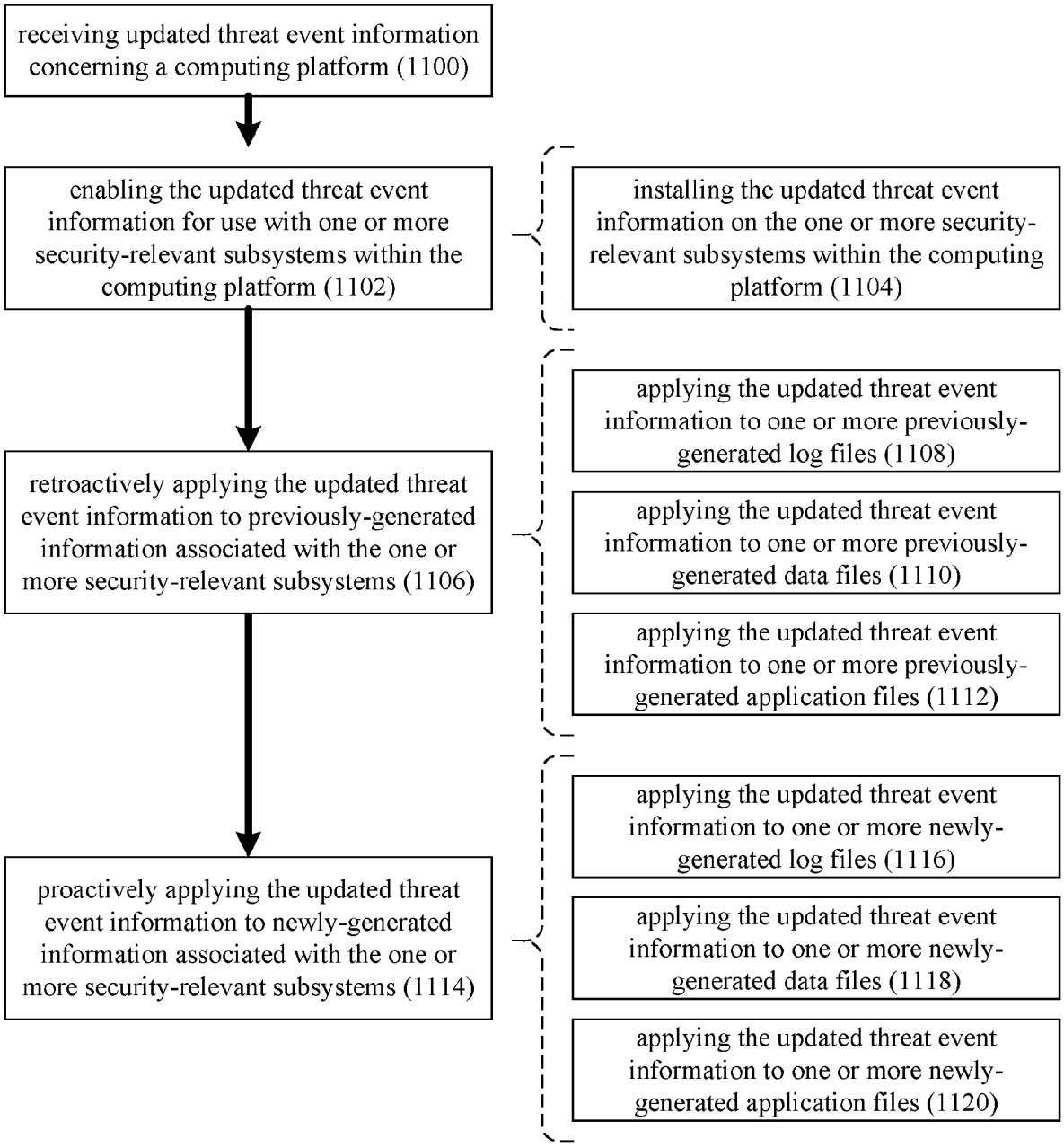

receiving updated threat event information concerning a computing platform (1100)
enabling the updated threat event information for use with one or more security-relevant subsystems within the computing platform (1102)
installing the updated threat event information on the one or more security-relevant subsystems within the computing platform (1104)
retroactively applying the updated threat event information to previously-generated information associated with the one or more security-relevant subsystems (1106)
applying the updated threat event information to one or more previously-generated log files (1108)
applying the updated threat event information to one or more previously-generated data files (1110)
applying the updated threat event information to one or more previously-generated application files (1112)
proactively applying the updated threat event information to newly-generated information associated with the one or more security-relevant subsystems (1114)
applying the updated threat event information to one or more newly-generated log files (1116)
applying the updated threat event information to one or more newly-generated data files (1118)
applying the updated threat event information to one or more newly-generated application files (1120)

FIG. 21

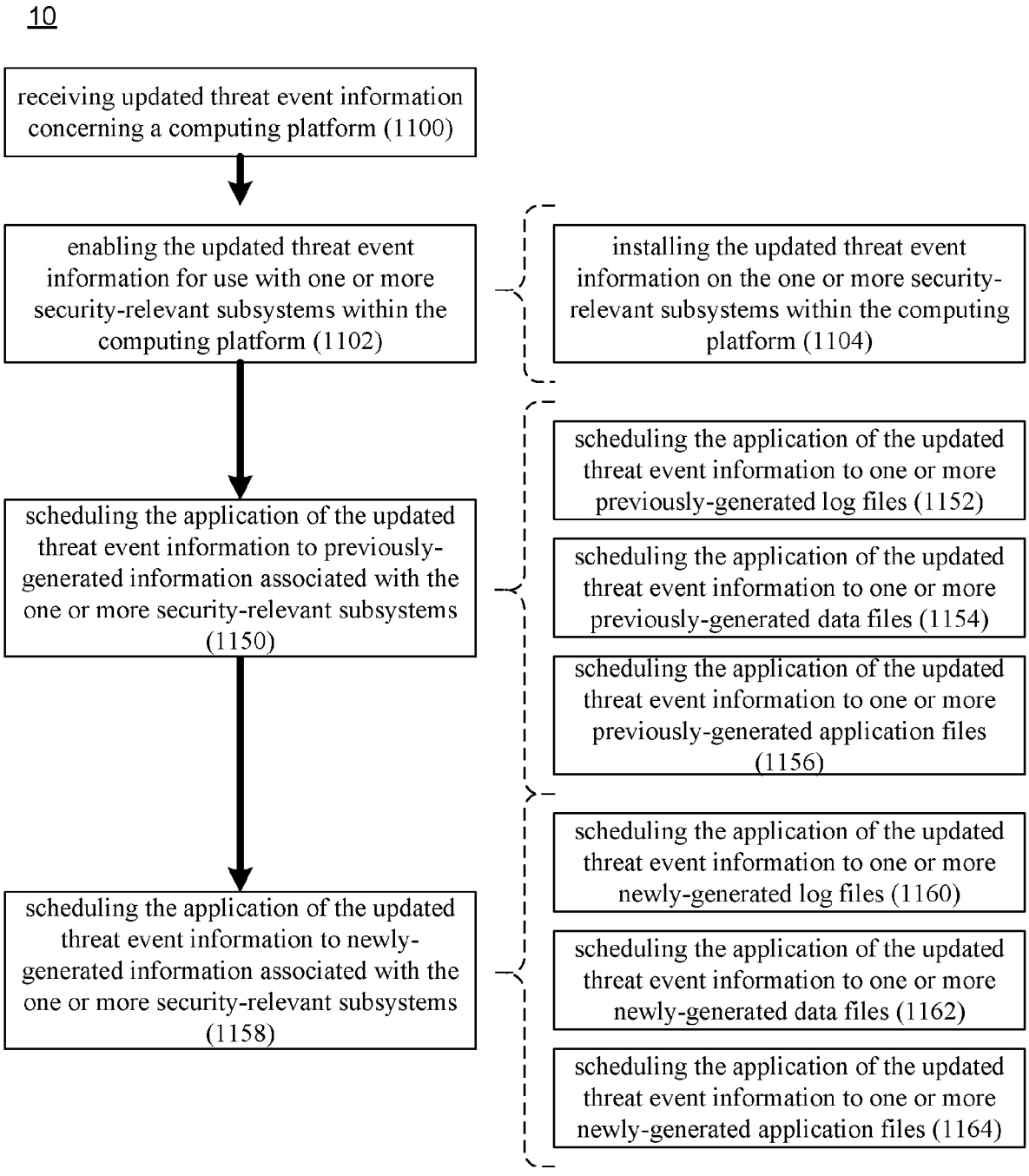

10
receiving updated threat event information concerning a computing platform (1100)
enabling the updated threat event information for use with one or more security-relevant subsystems within the computing platform (1102)
installing the updated threat event information on the one or more security-relevant subsystems within the computing platform (1104)
scheduling the application of the updated threat event information to previously-generated information associated with the one or more security-relevant subsystems (1150)
scheduling the application of the updated threat event information to one or more previously-generated log files (1152)
scheduling the application of the updated threat event information to one or more previously-generated data files (1154)
scheduling the application of the updated threat event information to one or more previously-generated application files (1156)
scheduling the application of the updated threat event information to newly-generated information associated with the one or more security-relevant subsystems (1158)
scheduling the application of the updated threat event information to one or more newly-generated log files (1160)
scheduling the application of the updated threat event information to one or more newly-generated data files (1162)
scheduling the application of the updated threat event information to one or more newly-generated application files (1164)

FIG. 22

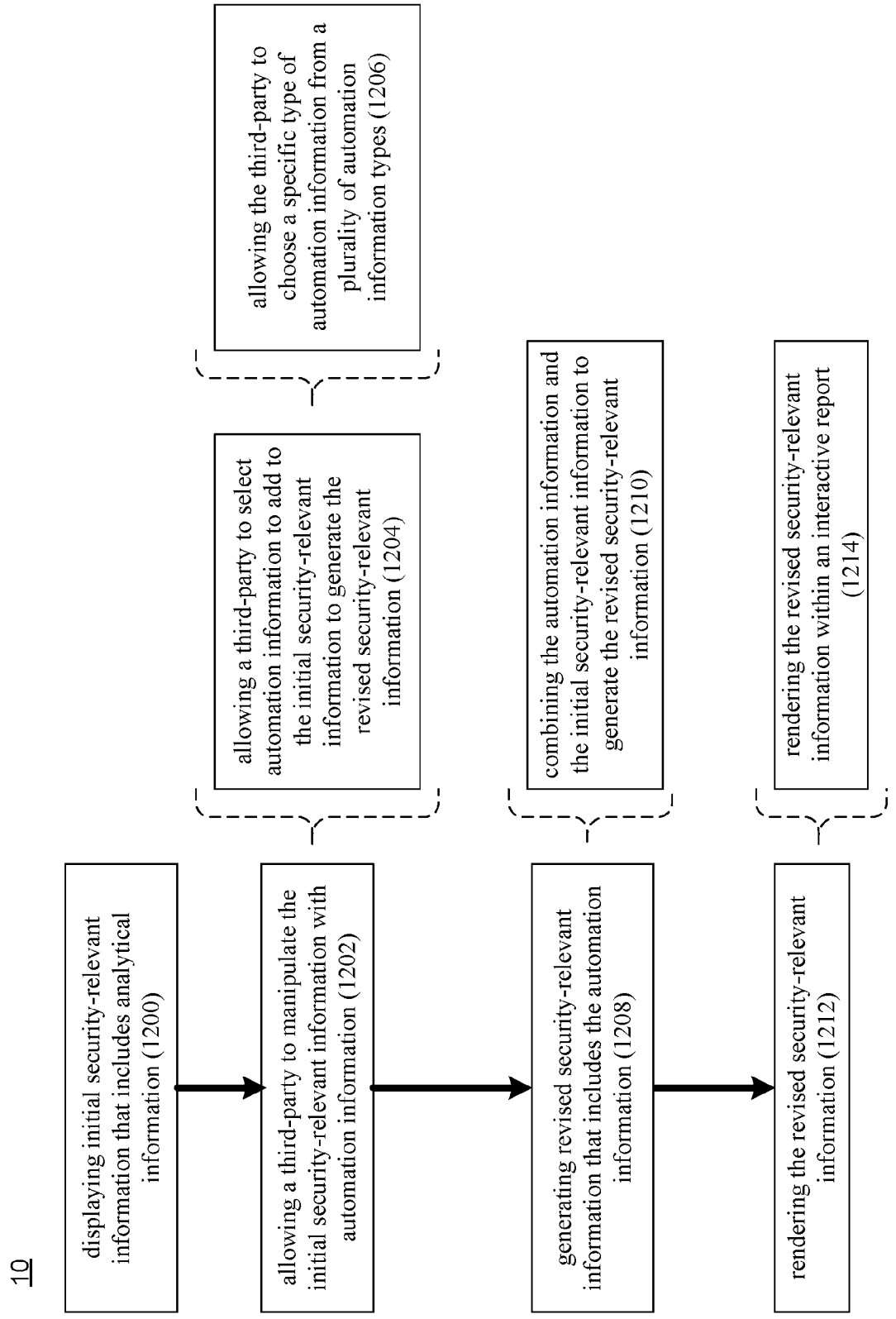

10
displaying initial security-relevant information that includes analytical information (1200)
allowing a third-party to manipulate the initial security-relevant information with automation information (1202)
allowing a third-party to select automation information to add to the initial security-relevant information to generate the revised security-relevant information (1204)
allowing the third-party to choose a specific type of automation information from a plurality of automation information types (1206)
generating revised security-relevant information that includes the automation information (1208)
combining the automation information and the initial security-relevant information to generate the revised security-relevant information (1210)
rendering the revised security-relevant information (1212)
rendering the revised security-relevant information within an interactive report (1214)

FIG. 23

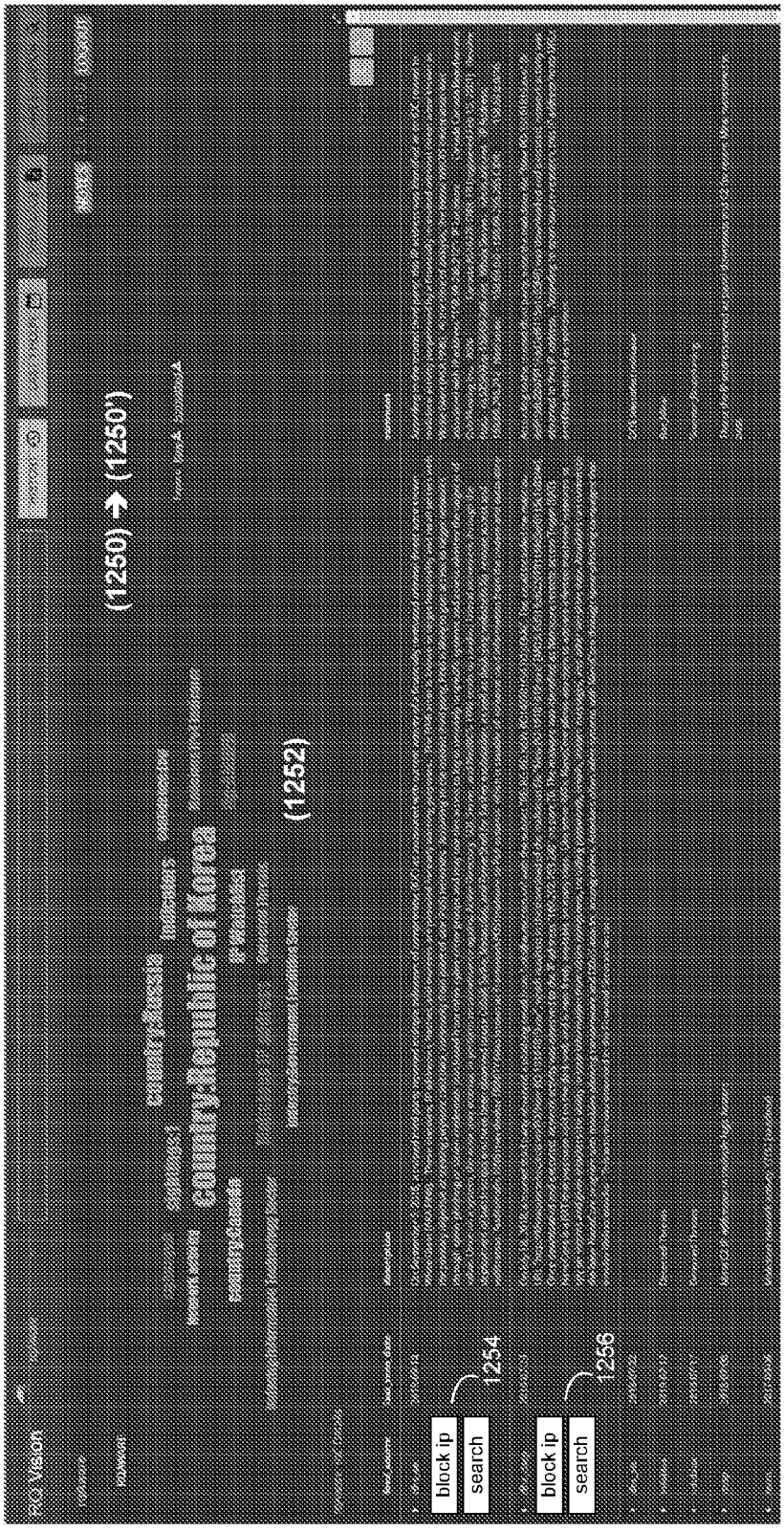

1258
(1250) → (1250')
(1252)
1254
1256
block ip
search
block ip
search

FIG. 24

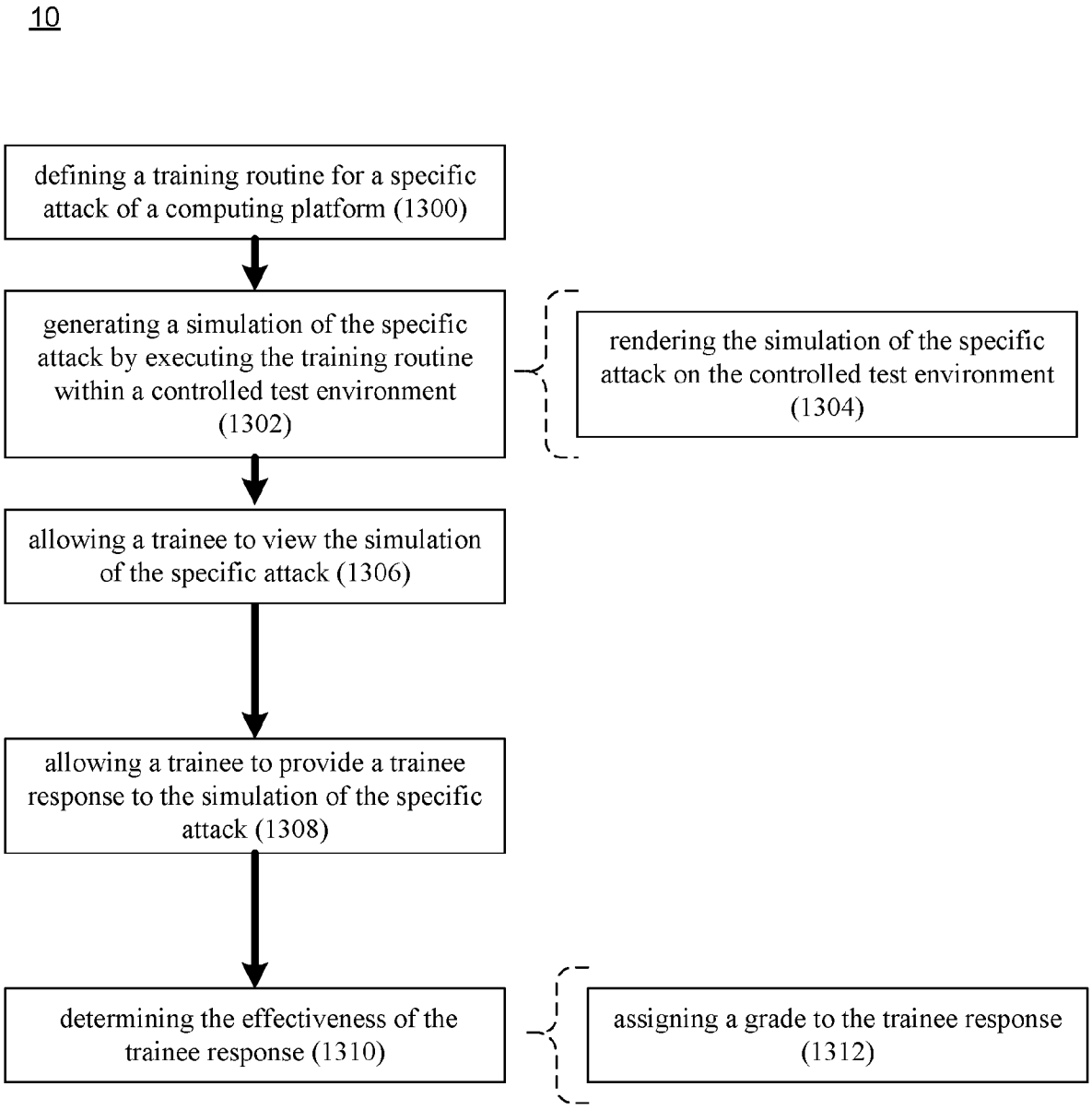

10
defining a training routine for a specific attack of a computing platform (1300)
generating a simulation of the specific attack by executing the training routine within a controlled test environment (1302)
rendering the simulation of the specific attack on the controlled test environment (1304)
allowing a trainee to view the simulation of the specific attack (1306)
allowing a trainee to provide a trainee response to the simulation of the specific attack (1308)
determining the effectiveness of the trainee response (1310)
assigning a grade to the trainee response (1312)

FIG. 25

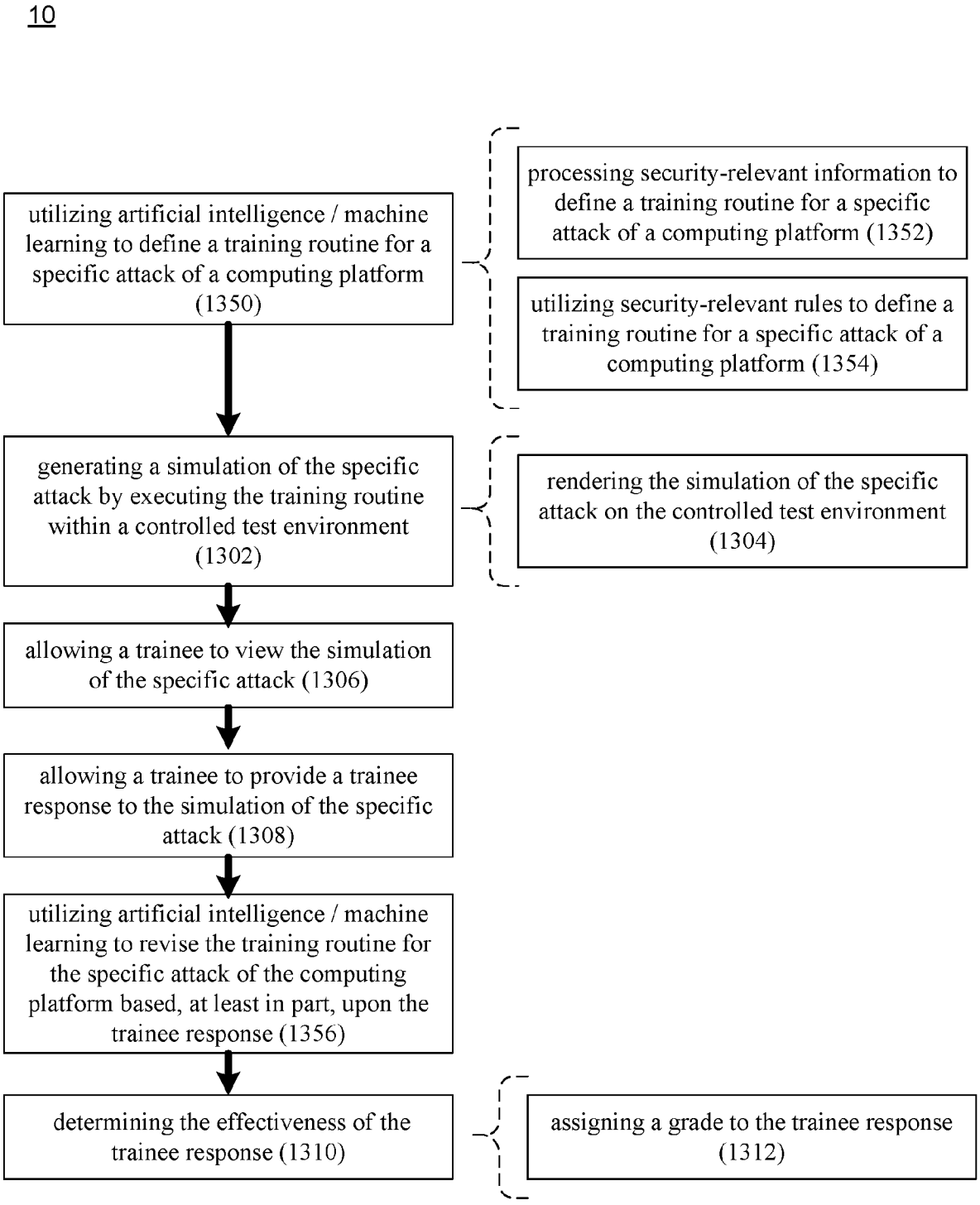

10
utilizing artificial intelligence / machine learning to define a training routine for a specific attack of a computing platform (1350)
processing security-relevant information to define a training routine for a specific attack of a computing platform (1352)
utilizing security-relevant rules to define a training routine for a specific attack of a computing platform (1354)
generating a simulation of the specific attack by executing the training routine within a controlled test environment (1302)
rendering the simulation of the specific attack on the controlled test environment (1304)
allowing a trainee to view the simulation of the specific attack (1306)
allowing a trainee to provide a trainee response to the simulation of the specific attack (1308)
utilizing artificial intelligence / machine learning to revise the training routine for the specific attack of the computing platform based, at least in part, upon the trainee response (1356)
determining the effectiveness of the trainee response (1310)
assigning a grade to the trainee response (1312)

FIG. 26

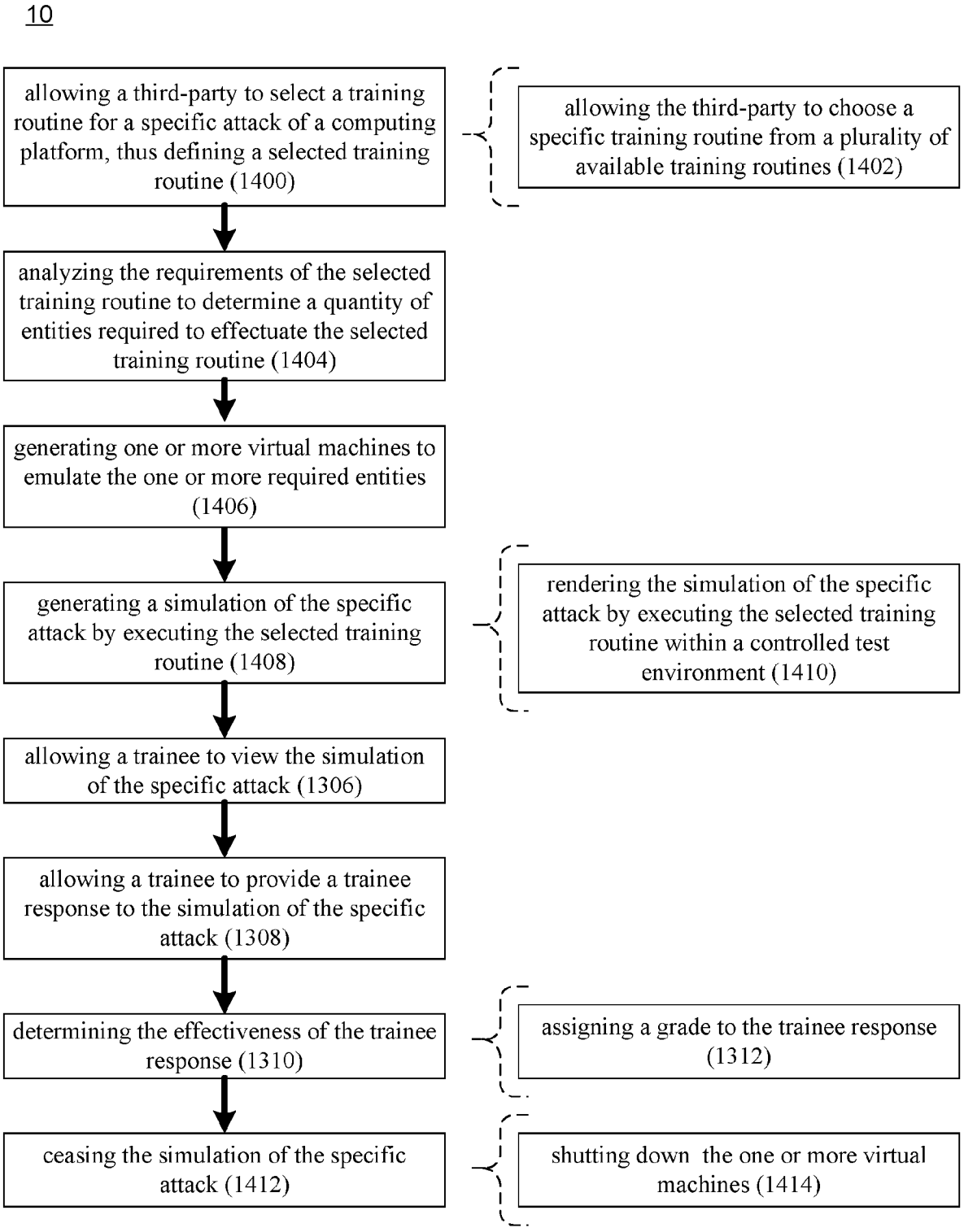

10
allowing a third-party to select a training routine for a specific attack of a computing platform, thus defining a selected training routine (1400)
allowing the third-party to choose a specific training routine from a plurality of available training routines (1402)
analyzing the requirements of the selected training routine to determine a quantity of entities required to effectuate the selected training routine (1404)
generating one or more virtual machines to emulate the one or more required entities (1406)
generating a simulation of the specific attack by executing the selected training routine (1408)
rendering the simulation of the specific attack by executing the selected training routine within a controlled test environment (1410)
allowing a trainee to view the simulation of the specific attack (1306)
allowing a trainee to provide a trainee response to the simulation of the specific attack (1308)
determining the effectiveness of the trainee response (1310)
assigning a grade to the trainee response (1312)
ceasing the simulation of the specific attack (1412)
shutting down the one or more virtual machines (1414)

FIG. 27

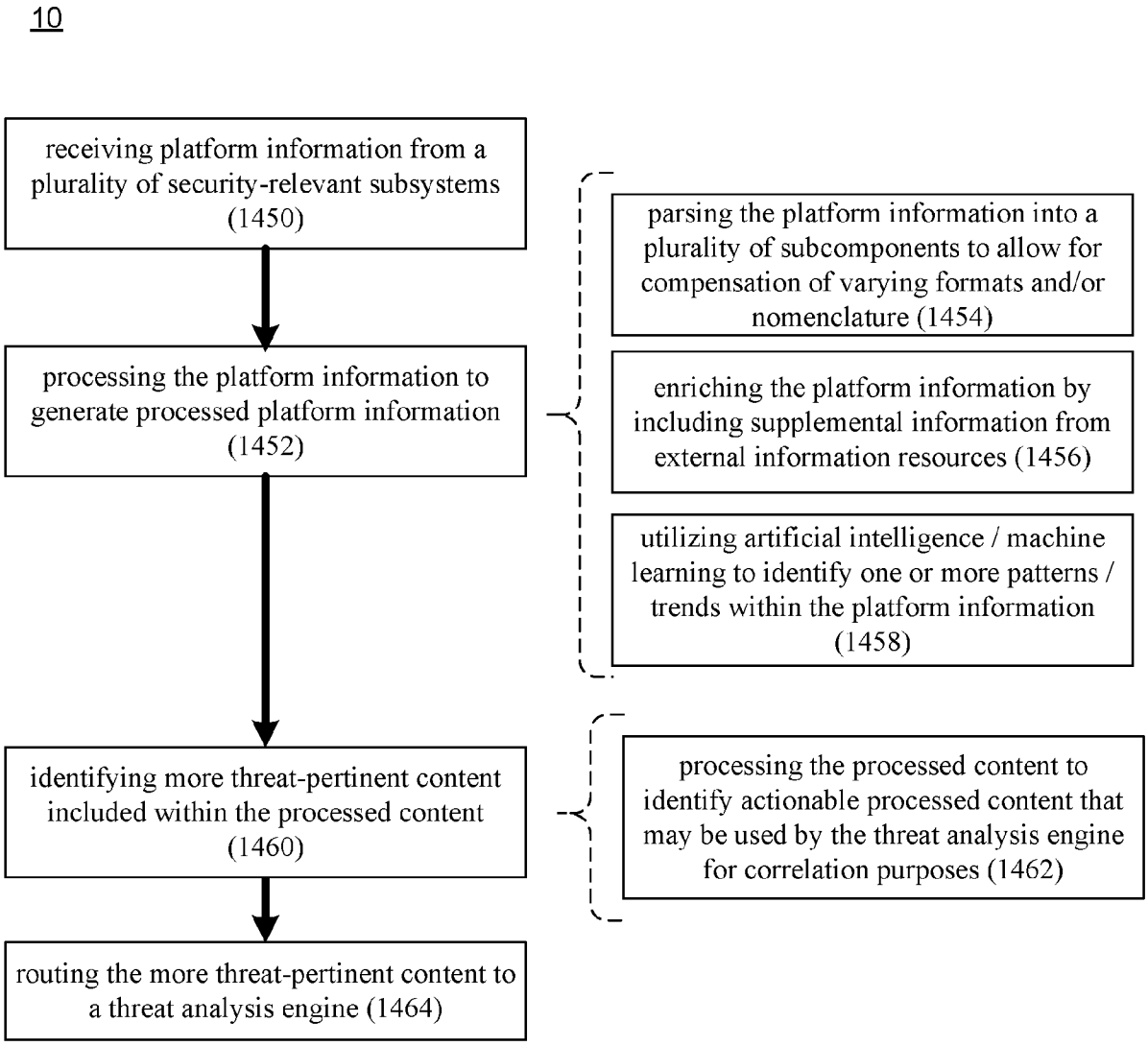

10
receiving platform information from a plurality of security-relevant subsystems (1450)
processing the platform information to generate processed platform information (1452)
identifying more threat-pertinent content included within the processed content (1460)
routing the more threat-pertinent content to a threat analysis engine (1464)
parsing the platform information into a plurality of subcomponents to allow for compensation of varying formats and/or nomenclature (1454)
enriching the platform information by including supplemental information from external information resources (1456)
utilizing artificial intelligence / machine learning to identify one or more patterns / trends within the platform information (1458)
processing the processed content to identify actionable processed content that may be used by the threat analysis engine for correlation purposes (1462)

FIG. 28

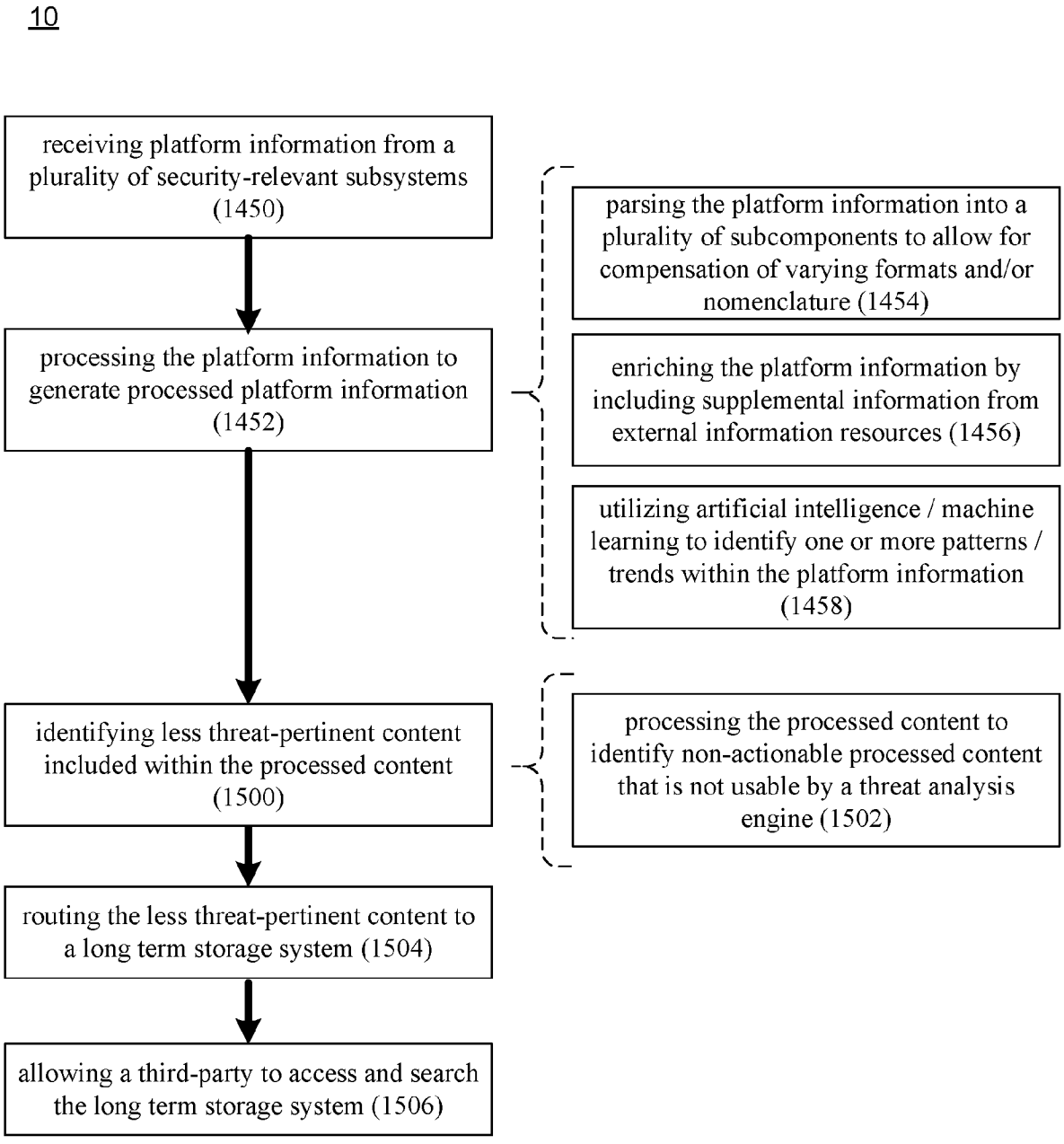

10
receiving platform information from a plurality of security-relevant subsystems (1450)
processing the platform information to generate processed platform information (1452)
identifying less threat-pertinent content included within the processed content (1500)
routing the less threat-pertinent content to a long term storage system (1504)
allowing a third-party to access and search the long term storage system (1506)
parsing the platform information into a plurality of subcomponents to allow for compensation of varying formats and/or nomenclature (1454)
enriching the platform information by including supplemental information from external information resources (1456)
utilizing artificial intelligence / machine learning to identify one or more patterns / trends within the platform information (1458)
processing the processed content to identify non-actionable processed content that is not usable by a threat analysis engine (1502)

FIG. 29

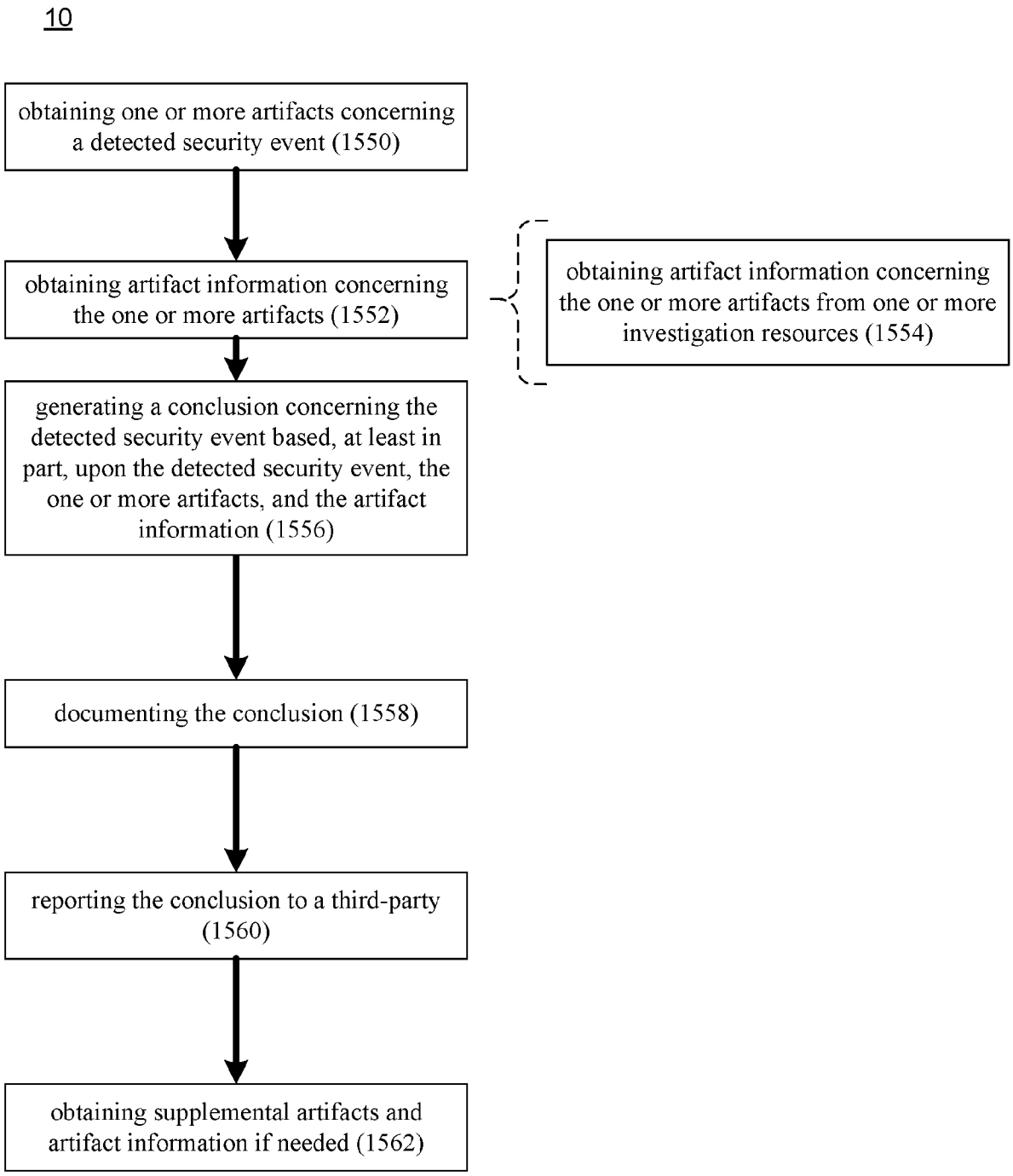

10
obtaining one or more artifacts concerning a detected security event (1550)
obtaining artifact information concerning the one or more artifacts (1552)
obtaining artifact information concerning the one or more artifacts from one or more investigation resources (1554)
generating a conclusion concerning the detected security event based, at least in part, upon the detected security event, the one or more artifacts, and the artifact information (1556)
documenting the conclusion (1558)
reporting the conclusion to a third-party (1560)
obtaining supplemental artifacts and artifact information if needed (1562)

FIG. 30

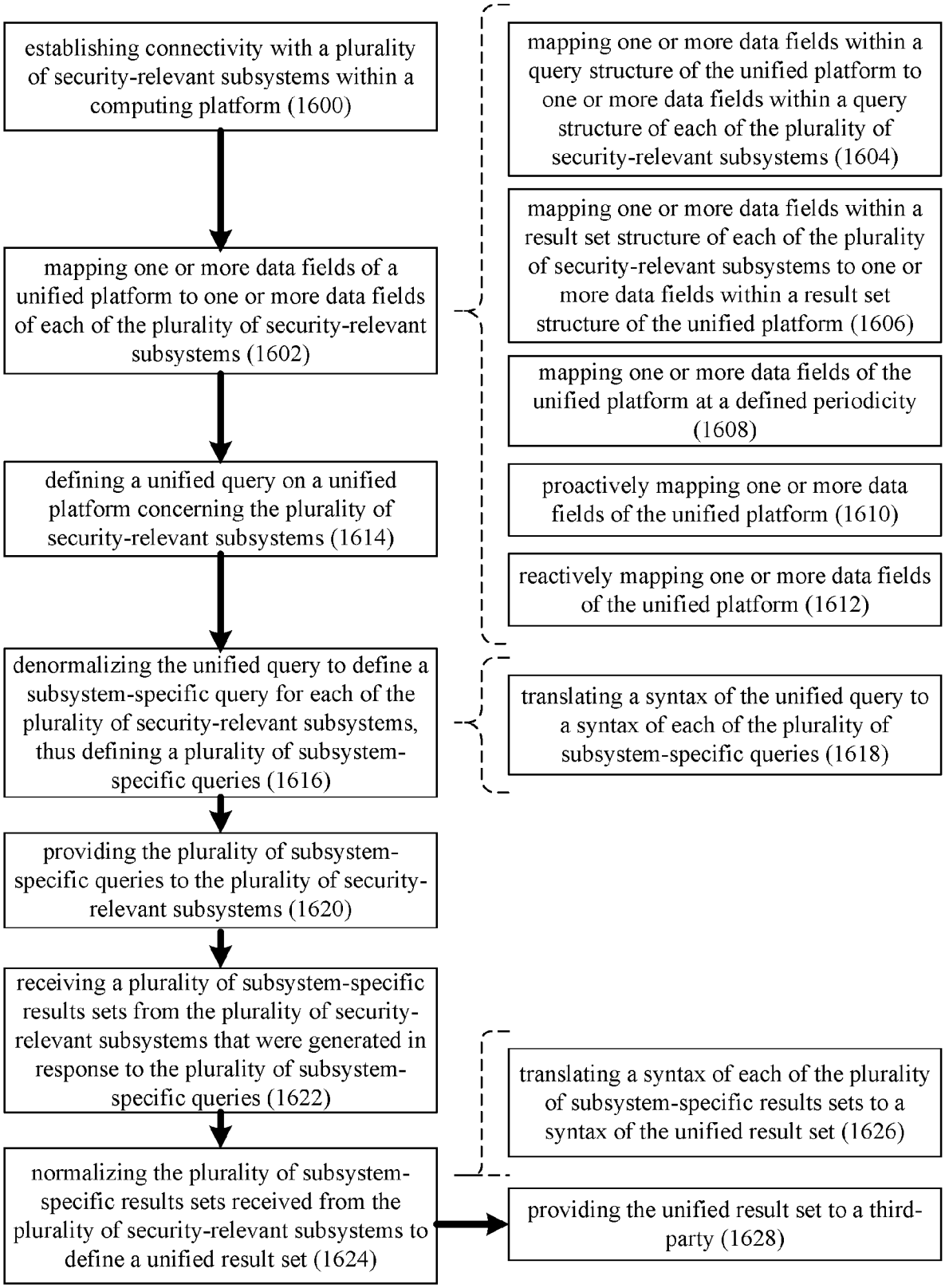

establishing connectivity with a plurality of security-relevant subsystems within a computing platform (1600)
mapping one or more data fields of a unified platform to one or more data fields of each of the plurality of security-relevant subsystems (1602)
defining a unified query on a unified platform concerning the plurality of security-relevant subsystems (1614)
denormalizing the unified query to define a subsystem-specific query for each of the plurality of security-relevant subsystems, thus defining a plurality of subsystem-specific queries (1616)
providing the plurality of subsystem-specific queries to the plurality of security-relevant subsystems (1620)
receiving a plurality of subsystem-specific results sets from the plurality of security-relevant subsystems that were generated in response to the plurality of subsystem-specific queries (1622)
normalizing the plurality of subsystem-specific results sets received from the plurality of security-relevant subsystems to define a unified result set (1624)
mapping one or more data fields within a query structure of the unified platform to one or more data fields within a query structure of each of the plurality of security-relevant subsystems (1604)
mapping one or more data fields within a result set structure of each of the plurality of security-relevant subsystems to one or more data fields within a result set structure of the unified platform (1606)
mapping one or more data fields of the unified platform at a defined periodicity (1608)
proactively mapping one or more data fields of the unified platform (1610)
reactively mapping one or more data fields of the unified platform (1612)
translating a syntax of the unified query to a syntax of each of the plurality of subsystem-specific queries (1618)
translating a syntax of each of the plurality of subsystem-specific results sets to a syntax of the unified result set (1626)
providing the unified result set to a third-party (1628)

FIG. 31

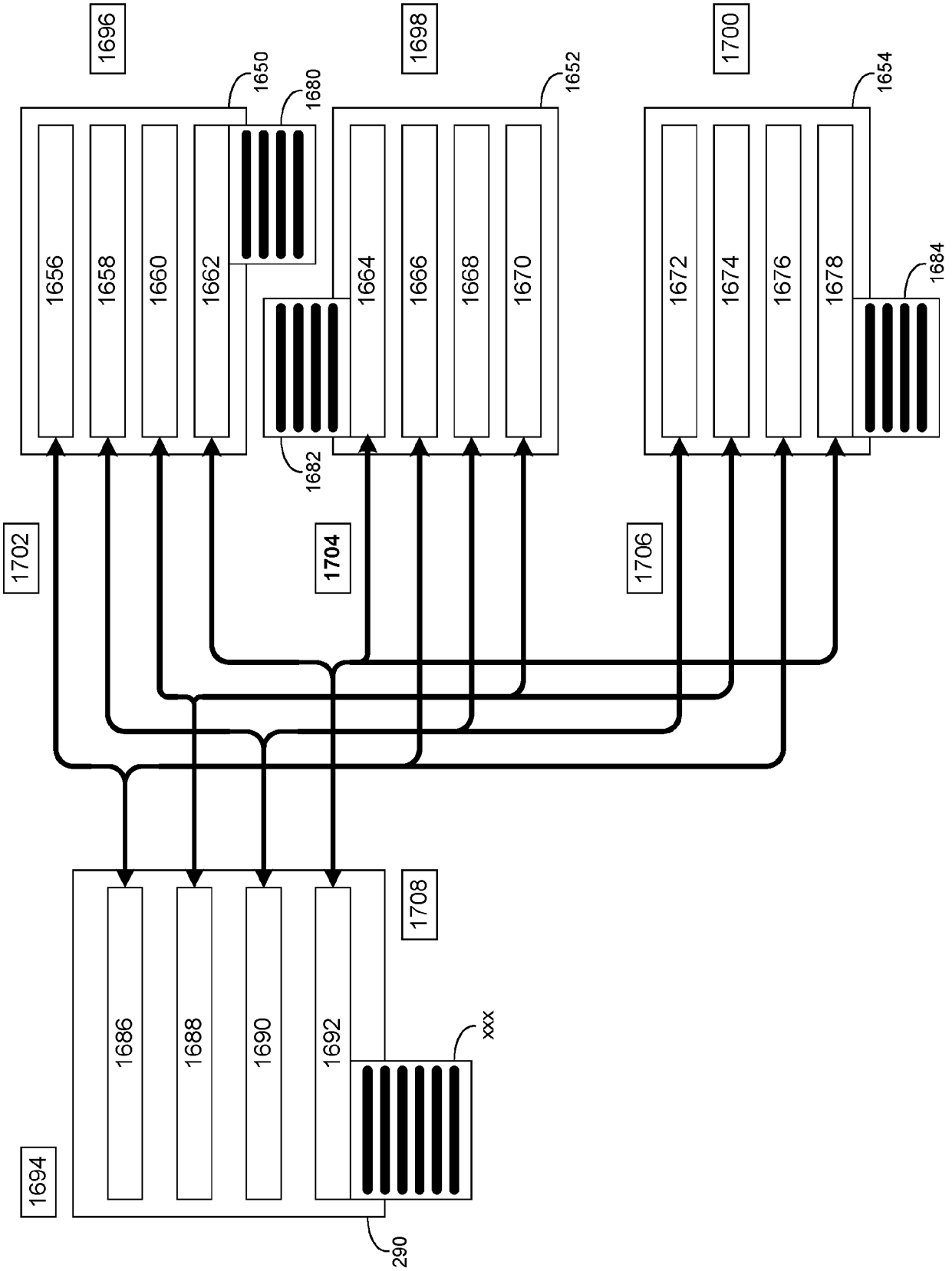

1696
1650
1680
1698
1652
1700
1654
1656
1658
1660
1662
1664
1666
1668
1670
1672
1674
1676
1678
1684
1682
1702
1704
1706
1708
1686
1688
1690
1692
xxx
1694
290

FIG. 32

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US 20/62252

A. CLASSIFICATION OF SUBJECT MATTER
IPC - H04L 29/06; G06F 11/00 (2021.01)
CPC - H04L 63/1433; H04L 63/1441

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
See Search History document

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched
See Search History document

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)
See Search History document

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	US 2008/0016043 A1 (Bakalash et al.) 17 January 2008 (17.01.2008), entire document especially paras [0054], [0171], [0173], [0230], [0231], [0234], [0238], [0239], [0241], [0247], [0248]	1-30
A	US 2004/0122814 A1 (Zhang et al.) 24 June 2004 (24.06.2004), entire document	1-30
A	US 2008/0086716 A1 (Devore, JR. et al.) 10 April 2008 (10.04.2008), entire document	1-30
A	US 2007/0136323 A1 (Zurek et al.) 14 June 2007 (14.06.2007), entire document	1-30
A	US 2011/0022627 A1 (Mohan) 27 January 2011 (27.01.2011), entire document	1-30

☐ Further documents are listed in the continuation of Box C. ☐ See patent family annex.

* Special categories of cited documents:
"A" document defining the general state of the art which is not considered to be of particular relevance
"D" document cited by the applicant in the international application
"E" earlier application or patent but published on or after the international filing date
"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)
"O" document referring to an oral disclosure, use, exhibition or other means
"P" document published prior to the international filing date but later than the priority date claimed
"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art
"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
22 January 2021 (22.01.2021)	11 FEB 2021
Name and mailing address of the ISA/US Mail Stop PCT, Attn: ISA/US, Commissioner for Patents P.O. Box 1450, Alexandria, Virginia 22313-1450 Facsimile No. 571-273-8300	Authorized officer Lee Young Telephone No. PCT Helpdesk: 571-272-4300

Form PCT/ISA/210 (second sheet) (July 2019)

UNITED STATES
PATENT AND TRADEMARK OFFICE

P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

ACKNOWLEDGEMENT OF CHANGE TO SMALL ENTITY STATUS

APPLICATION #	FILING OR 371(C) DATE	FIRST NAMED APPLICANT	ATTORNEY DOCKET #	REQUEST ID
17/657,981	04/05/2022	Avi SHUA	15635.0008-00000	62499

The entity status change request below filed through Patent Center on 02/03/2023 has been accepted.

Certifications

APPLICANT ASSERTING SMALL ENTITY STATUS. See 37CFR 1.27.

Signature

I certify, in accordance with 37 CFR 1.4(d)(4), that I am one of the signatories making the entity status change.

Signature	Name	Registration #
/Christopher C. Johns/	Christopher Johns	68664

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
17/657,981	04/05/2022	Avi SHUA	15635.0008-00000	2397

22852 7590 01/30/2023

FINNEGAN, HENDERSON, FARABOW, GARRETT & DUNNER LLP
901 NEW YORK AVENUE, NW
WASHINGTON, DC 20001-4413

EXAMINER
SU, SARAH

ART UNIT	PAPER NUMBER
2431	

NOTIFICATION DATE	DELIVERY MODE
01/30/2023	ELECTRONIC

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Notice of the Office communication was sent electronically on above-indicated "Notification Date" to the following e-mail address(es):

regional-desk@finnegan.com

Corrected ***Notice of Allowability***	**Application No.** 17/657,981	**Applicant(s)** SHUA, Avi	
	Examiner SARAH SU	**Art Unit** 2431	**AIA (FITF) Status** Yes

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address--

All claims being allowable, PROSECUTION ON THE MERITS IS (OR REMAINS) CLOSED in this application. If not included herewith (or previously mailed), a Notice of Allowance (PTOL-85) or other appropriate communication will be mailed in due course. **THIS NOTICE OF ALLOWABILITY IS NOT A GRANT OF PATENT RIGHTS.** This application is subject to withdrawal from issue at the initiative of the Office or upon petition by the applicant. See 37 CFR 1.313 and MPEP 1308.

1. ☑ This communication is responsive to IDS filed 24 January 2023.
   ☐ A declaration(s)/affidavit(s) under **37 CFR 1.130(b)** was/were filed on _____.

2. ☐ An election was made by the applicant in response to a restriction requirement set forth during the interview on _____; the restriction requirement and election have been incorporated into this action.

3. ☑ The allowed claim(s) is/are 81-100 . As a result of the allowed claim(s), you may be eligible to benefit from the **Patent Prosecution Highway** program at a participating intellectual property office for the corresponding application. For more information, please see **http://www.uspto.gov/patents/init_events/pph/index.jsp** or send an inquiry to **PPHfeedback@uspto.gov.**

4. ☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

   Certified copies:

   a) ☐ All b) ☐ Some* c) ☐ None of the:

   1. ☐ Certified copies of the priority documents have been received.
   2. ☐ Certified copies of the priority documents have been received in Application No. _____ .
   3. ☐ Copies of the certified copies of the priority documents have been received in this national stage application from the International Bureau (PCT Rule 17.2(a)).

   * Certified copies not received: _____ .

Applicant has THREE MONTHS FROM THE "MAILING DATE" of this communication to file a reply complying with the requirements noted below. Failure to timely comply will result in ABANDONMENT of this application.
THIS THREE-MONTH PERIOD IS NOT EXTENDABLE.

5. ☐ CORRECTED DRAWINGS (as "replacement sheets") must be submitted.
   ☐ including changes required by the attached Examiner's Amendment / Comment or in the Office action of Paper No./Mail Date _____ .

   Identifying indicia such as the application number (see 37 CFR 1.84(c)) should be written on the drawings in the front (not the back) of each sheet. Replacement sheet(s) should be labeled as such in the header according to 37 CFR 1.121(d).

6. ☐ DEPOSIT OF and/or INFORMATION about the deposit of BIOLOGICAL MATERIAL must be submitted. Note the attached Examiner's comment regarding REQUIREMENT FOR THE DEPOSIT OF BIOLOGICAL MATERIAL.

Attachment(s)

1. ☐ Notice of References Cited (PTO-892)
2. ☑ Information Disclosure Statements (PTO/SB/08), Paper No./Mail Date _____.
3. ☐ Examiner's Comment Regarding Requirement for Deposit of Biological Material _____.
4. ☐ Interview Summary (PTO-413), Paper No./Mail Date. _____.
5. ☐ Examiner's Amendment/Comment
6. ☐ Examiner's Statement of Reasons for Allowance
7. ☐ Other _____.

/SARAH SU/
Primary Examiner, Art Unit 2431

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Use as many sheets as necessary)				Complete if Known	
				Application Number	17/657,981
				Filing Date	April 5, 2022
				First Named Inventor	Avi Shua
				Art Unit	2431
				Examiner Name	SU, SARAH
Sheet	1	of	2	Attorney Docket Number	15635.0008-00000

U.S. PATENTS

Examiner Initials*	Cite No.	Document Number Number-Kind Code (if known)	Issue Date MM-DD-YYYY	Name of Patentee or Applicant of Cited Document	Pages, Columns, Lines, Where Relevant Passages or Relevant Figures Appear
		US-11,120,124 B2	09-14-2021	Fusenig et al.	
		US-10,402,560 B2	09-03-2019	Gilbert	
		US-10,079,842 B1	09-18-2018	Brandwine et al.	
		US-9,229,758 B2	01-05-2016	Ammons et al.	
		US-9,734,325 B1	08-15-2017	Neumann et al.	
		US-11,216,563 B1	01-04-2022	Veselov et al.	
		US-9,756,070 B1	09-05-2017	Crowell et al.	
		US-9,858,105 B1	01-02-2018	Upadhyay et al.	
		US-10,782,952 B1	09-22-2020	Doring et al.	
		US-9,069,983 B1	06-30-2015	Nijjar	
		US-9,268,689 B1	02-23-2016	Chen et al.	
		US-10,534,915 B2	01-14-2020	Cherny et al.	
		US-8,479,294 B1	07-02-2013	Li et al.	
		US-10,956,484 B1	03-23-2021	Kulkarni et al.	

U.S. PUBLISHED PATENT APPLICATIONS

Examiner Initials*	Cite No.	Document Number Number-Kind Code (if known)	Publication Date MM-DD-YYYY	Name of Patentee or Applicant of Cited Document	Pages, Columns, Lines, Where Relevant Passages or Relevant Figures Appear
		US-2020/0042707 A1	02-06-2020	Kucherov et al.	
		US-2012/0323853 A1	12-20-2012	Fries et al.	
		US-2014/0096135 A1	04-03-2014	Kundu et al.	
		US-2009/0007100 A1	01-01-2009	Field et al.	
		US-2008/0263658 A1	10-23-2008	Michael et al.	
		US-2010/0017512 A1	01-21-2010	Ciano et al.	
		US-2021/0019404 A1	01-21-2021	Strogov et al.	
		US-2016/0063251 A1	03-03-2016	Spernow et al.	
		US-2015/0020203 A1	01-15-2015	Xie et al.	
		US-2014/0259168 A1	09-11-2014	McNamee et al.	
		US-2005/0055558 A1	03-10-2005	Carmona	
		US-2014/0007239 A1	01-02-2014	Sharpe et al.	
		US-2018/0198610 A1	07-12-2018	Mullins et al.	
		US-2022/0294817 A1	09-15-2022	Parekh et al.	

Examiner Signature	/SARAH SU/	Date Considered	01/26/2023

*EXAMINER: Initial if reference considered, whether or not citation is in conformance with MPEP 609. Draw line through citation if not in conformance and not considered. Include copy of this form with next communication to applicant.

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.S/

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Use as many sheets as necessary)				Complete if Known	
				Application Number	17/657,981
				Filing Date	April 5, 2022
				First Named Inventor	Avi Shua
				Art Unit	2431
				Examiner Name	SU, SARAH
Sheet	2	of	2	Attorney Docket Number	15635.0008-00000

Note: Submission of copies of U.S. Patents and published U.S. Patent Applications is not required.

NON PATENT LITERATURE DOCUMENTS			
Examiner Initials*	Cite No.	Include name of the author (in CAPITAL LETTERS), title of the article (when appropriate), title of the item (book, magazine, journal, serial, symposium, catalog, etc.), date, page(s), volume-issue number(s), publisher, city and/or country where published.	Translation
		Rani et al.; "An Efficient Approach to Forensic Investigation in Cloud using VM Snapshots", 2015 International Conference on Pervasive Computing (ICPC), 5 pages, (2015).	N/A
		Wei et al., "Managing Security of Virtual Machine Images in a Cloud Environment", CCSW'09, pp. 91-96, November 13, 2009	N/A
		Almulla et al., "Digital Forensic of a Cloud Based Snapshot", The Sixth International Conference on Innovative Computing Technology (INTECH 2016), pp. 724-729, (2016)	N/A
		Pandey et al., "An Approach for Virtual Machine Image Security", Computer Science and Engineering MNNIT, Allahabad, 2014 International Conference on Signal Propagation and Computer Technology (ICSPCT), pp. 616-623, (2014)	N/A
		Rajasekaran et al., "Scalable Cloud Security via Asynchronous Virtual Machine Introspection", 8th USENIX Workshop on Hot Topics in Cloud Computing, Cover page and pages 1-6, (2016)	N/A
		Kaur et al., "Secure VM Backup and Vulnerability Removal in Infrastructure Clouds", 2014 International Conference on Advances in Computing, Communications and Informatics (ICACCI), pp. 1217-1226, (2014)	N/A
		Fernandez et al., "Two patterns for cloud computing: Secure Virtual Machine Image Repository and Cloud Policy Management Point", PLoP '13: Proceedings of the 20th Conference on Pattern Languages of Programs October 2013 Article No.: 15, Association for Computing Machinery (ACM), Cover sheet and pages 1-11, (2013)	N/A
		Fernandez et al. "Building a security reference architecture for cloud systems", Springer, Requirements Eng, vol. 21, pp. 225-249, (2016)	N/A
		Ammons et al., "Virtual machine images as structured data: the Mirage image library", IBM Research, Cover sheet 2 pages and pages 1-6, (2011)	N/A
		"IBM Point of View Security and Cloud Computing", IBM SmartCloud Enterprise, Cloud Computing White Paper, 13 sheets of cover pages and pages 1-20, (2009)	N/A
		Cui et al., "A Less Resource-Consumed Security Architecture on Cloud Platform", Wuhan University Journal of Natural Sciences, vol. 21, no. 5, pp. 407-414, (2016)	N/A
		Bugiel et al., "AmazonIA: When Elasticity Snaps Back", CCS' 11, ACM, pp. 389-400, (2011)	N/A
		Non-Final Office Action Application No. 17/658,149 dated December 30, 2022, in the United States Patent and Trademark Office. (15635.0007-00000)	N/A
		Non-Final Office Action Application No. 17/658,156 dated January 3, 2023, in the United States Patent and Trademark Office. (15635.0010-00000)	N/A
		Final Office Action Application No. 17/657,986 dated November 23, 2022, in the United States Patent and Trademark Office. (15635.0009-00000)	N/A

Examiner Signature	/SARAH SU/	Date Considered	01/26/2023

*EXAMINER: Initial if reference considered, whether or not citation is in conformance with MPEP 609. Draw line through citation if not in conformance and not considered. Include copy of this form with next communication to applicant.

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.S/

INFORMATION DISCLOSURE STATEMENT BY APPLICANT *(Use as many sheets as necessary)*				***Complete if Known***	
				Application Number	17/657,981
				Filing Date	April 5, 2022
				First Named Inventor	Avi Shua
				Art Unit	2431
				Examiner Name	SU, SARAH
Sheet	1	of	2	*Attorney Docket Number*	15635.0008-00000

U.S. PATENTS

Examiner Initials*	Cite No.	Document Number Number-Kind Code *(if known)*	Issue Date MM-DD-YYYY	Name of Patentee or Applicant of Cited Document	Pages, Columns, Lines, Where Relevant Passages or Relevant Figures Appear
		US-11,120,124 B2	09-14-2021	Fusenig et al.	
		US-10,402,560 B2	09-03-2019	Gilbert	
		US-10,079,842 B1	09-18-2018	Brandwine et al.	
		US-9,229,758 B2	01-05-2016	Ammons et al.	
		US-9,734,325 B1	08-15-2017	Neumann et al.	
		US-11,216,563 B1	01-04-2022	Veselov et al.	
		US-9,756,070 B1	09-05-2017	Crowell et al.	
		US-9,858,105 B1	01-02-2018	Upadhyay et al.	
		US-10,782,952 B1	09-22-2020	Doring et al.	
		US-9,069,983 B1	06-30-2015	Nijjar	
		US-9,268,689 B1	02-23-2016	Chen et al.	
		US-10,534,915 B2	01-14-2020	Cherny et al.	
		US-8,479,294 B1	07-02-2013	Li et al.	
		US-10,956,484 B1	03-23-2021	Kulkarni et al.	

U.S. PUBLISHED PATENT APPLICATIONS

Examiner Initials*	Cite No.	Document Number Number-Kind Code *(if known)*	Publication Date MM-DD-YYYY	Name of Patentee or Applicant of Cited Document	Pages, Columns, Lines, Where Relevant Passages or Relevant Figures Appear
		US-2020/0042707 A1	02-06-2020	Kucherov et al.	
		US-2012/0323853 A1	12-20-2012	Fries et al.	
		US-2014/0096135 A1	04-03-2014	Kundu et al.	
		US-2009/0007100 A1	01-01-2009	Field et al.	
		US-2008/0263658 A1	10-23-2008	Michael et al.	
		US-2010/0017512 A1	01-21-2010	Ciano et al.	
		US-2021/0019404 A1	01-21-2021	Strogov et al.	
		US-2016/0063251 A1	03-03-2016	Spernow et al.	
		US-2015/0020203 A1	01-15-2015	Xie et al.	
		US-2014/0259168 A1	09-11-2014	McNamee et al.	
		US-2005/0055558 A1	03-10-2005	Carmona	
		US-2014/0007239 A1	01-02-2014	Sharpe et al.	
		US-2018/0198610 A1	07-12-2018	Mullins et al.	
		US-2022/0294817 A1	09-15-2022	Parekh et al.	

Examiner Signature		Date Considered	

*EXAMINER: Initial if reference considered, whether or not citation is in conformance with MPEP 609. Draw line through citation if not in conformance and not considered. Include copy of this form with next communication to applicant.

INFORMATION DISCLOSURE STATEMENT BY APPLICANT *(Use as many sheets as necessary)*				*Complete if Known*	
				Application Number	17/657,981
				Filing Date	April 5, 2022
				First Named Inventor	Avi Shua
				Art Unit	2431
				Examiner Name	SU, SARAH
Sheet	2	of	2	*Attorney Docket Number*	15635.0008-00000

Note: Submission of copies of U.S. Patents and published U.S. Patent Applications is not required.

NON PATENT LITERATURE DOCUMENTS			
Examiner Initials*	Cite No.	Include name of the author (in CAPITAL LETTERS), title of the article (when appropriate), title of the item (book, magazine, journal, serial, symposium, catalog, etc.), date, page(s), volume-issue number(s), publisher, city and/or country where published.	Translation
		Rani et al.; "An Efficient Approach to Forensic Investigation in Cloud using VM Snapshots", 2015 International Conference on Pervasive Computing (ICPC), 5 pages, (2015).	N/A
		Wei et al., "Managing Security of Virtual Machine Images in a Cloud Environment", CCSW'09, pp. 91-96, November 13, 2009	N/A
		Almulla et al., "Digital Forensic of a Cloud Based Snapshot", The Sixth International Conference on Innovative Computing Technology (INTECH 2016), pp. 724-729, (2016)	N/A
		Pandey et al., "An Approach for Virtual Machine Image Security", Computer Science and Engineering MNNIT, Allahabad, 2014 International Conference on Signal Propagation and Computer Technology (ICSPCT), pp. 616-623, (2014)	N/A
		Rajasekaran et al., "Scalable Cloud Security via Asynchronous Virtual Machine Introspection", 8th USENIX Workshop on Hot Topics in Cloud Computing, Cover page and pages 1-6, (2016)	N/A
		Kaur et al., "Secure VM Backup and Vulnerability Removal in Infrastructure Clouds", 2014 International Conference on Advances in Computing, Communications and Informatics (ICACCI), pp. 1217-1226, (2014)	N/A
		Fernandez et al., "Two patterns for cloud computing: Secure Virtual Machine Image Repository and Cloud Policy Management Point", PLoP '13: Proceedings of the 20th Conference on Pattern Languages of Programs October 2013 Article No.: 15, Association for Computing Machinery (ACM), Cover sheet and pages 1-11, (2013)	N/A
		Fernandez et al. "Building a security reference architecture for cloud systems", Springer, Requirements Eng, vol. 21, pp. 225-249, (2016)	N/A
		Ammons et al., "Virtual machine images as structured data: the Mirage image library", IBM Research, Cover sheet 2 pages and pages 1-6, (2011)	N/A
		"IBM Point of View Security and Cloud Computing", IBM SmartCloud Enterprise, Cloud Computing White Paper, 13 sheets of cover pages and pages 1-20, (2009)	N/A
		Cui et al., "A Less Resource-Consumed Security Architecture on Cloud Platform", Wuhan University Journal of Natural Sciences, vol. 21, no. 5, pp. 407-414, (2016)	N/A
		Bugiel et al., "AmazonIA: When Elasticity Snaps Back", CCS' 11, ACM, pp. 389-400, (2011)	N/A
		Non-Final Office Action Application No. 17/658,149 dated December 30, 2022, in the United States Patent and Trademark Office. (15635.0007-00000)	N/A
		Non-Final Office Action Application No. 17/658,156 dated January 3, 2023, in the United States Patent and Trademark Office. (15635.0010-00000)	N/A
		Final Office Action Application No. 17/657,986 dated November 23, 2022, in the United States Patent and Trademark Office. (15635.0009-00000)	N/A

Examiner Signature		Date Considered	

*EXAMINER: Initial if reference considered, whether or not citation is in conformance with MPEP 609. Draw line through citation if not in conformance and not considered. Include copy of this form with next communication to applicant.

PATENT
Attorney Docket No. 15635.0008-00000

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Application of:	)	
	)	
Avi SHUA	)	Group Art Unit: 2431
	)	
Application No.: 17/657,981	)	Examiner: SU, SARAH
	)	
Filed: April 5, 2022	)	Confirmation No.: 2397
	)	
For: FORWARD AND REARWARD FACING ATTACK VECTOR VISUALIZATION	)	

Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Commissioner:

SUPPLEMENTAL INFORMATION DISCLOSURE STATEMENT UNDER 37 C.F.R. § 1.97(d)

Pursuant to 37 C.F.R. §§ 1.56 and 1.97(d), Applicant brings to the attention of the Examiner the documents on the attached listing. This Information Disclosure Statement is being filed after a Notice of Allowance and is accompanied by a fee of $260.00 as specified under § 1.17(p) and a statement as specified under § 1.97(e).

Statement under 37 C.F.R. § 1.97(e)

The following constitutes the statement specified by 37 C.F.R. § 1.97(e)(2). No item of information contained in the IDS was cited in a communication from a foreign patent office in a counterpart foreign application, and, to the knowledge of the person signing the certification after making reasonable inquiry, no item of information contained in the IDS was known to any individual designated in 37 CFR 1.56(c) more

than three months prior to the filing of this Information Disclosure Statement. See 37 CFR 1.97(e)(2).

A copy of the non-patent literature documents is attached.

Applicant respectfully requests that the Examiner consider the listed documents on the attached Form PTO/SB/08 and indicate that they were considered by making appropriate notations on the attached form.

This submission does not represent that a search has been made or that no better art exists and does not constitute an admission that each or all of the listed documents are material or constitute "prior art." If the Examiner applies any of the documents as prior art against any claim in the application and Applicant determines that the cited documents do not constitute "prior art" under United States law, Applicant reserves the right to present to the U.S. Patent and Trademark Office the relevant facts and law regarding the appropriate status of such documents.

Applicant further reserves the right to take appropriate action to establish the patentability of the disclosed invention over the listed documents, should one or more of the documents be applied against the claims of the present application.

If there is any fee due in connection with the filing of this Statement, please charge the fee to deposit account 06-0916.

Respectfully submitted,

FINNEGAN, HENDERSON, FARABOW,
GARRETT & DUNNER, L.L.P.

Dated: January 24, 2023

By: /Christopher C. Johns/
Christopher C. Johns
Reg. No. 68,664
(202) 408-4000

UNITED STATES
PATENT AND TRADEMARK OFFICE

P.O. Box 1450
Alexandria, VA 22313 - 1450
www.uspto.gov

ELECTRONIC PAYMENT RECEIPT

APPLICATION #	RECEIPT DATE / TIME	ATTORNEY DOCKET #
17/657,981	01/24/2023 12:16:42 PM ET	15635.0008-00000

Title of Invention

Forward and Rearward Facing Attack Vector Visualization

Application Information

APPLICATION TYPE	Utility - Nonprovisional Application under 35 USC 111(a)	PATENT #	-
CONFIRMATION #	2397	FILED BY	Michelle Daniels
PATENT CENTER #	61482282	AUTHORIZED BY	Christopher Johns
CUSTOMER #	22852	FILING DATE	04/05/2022
CORRESPONDENCE ADDRESS	-	FIRST NAMED INVENTOR	Avi SHUA

Payment Information

PAYMENT METHOD	PAYMENT TRANSACTION ID	PAYMENT AUTHORIZED BY
CARD / 1007	E20231NC17482837	Michelle Daniels

FEE CODE	DESCRIPTION	ITEM PRICE($)	QUANTITY	ITEM TOTAL($)
1806	SUBMISSION OF AN INFORMATION DISCLOSURE STATEMENT	260.00	1	260.00
			TOTAL AMOUNT:	$260.00

This Acknowledgement Receipt evidences receipt on the noted date by the USPTO of the indicated documents, characterized by the applicant, and including page counts, where applicable. It serves as evidence of receipt similar to a Post Card, as described in MPEP 503.

New Applications Under 35 U.S.C. 111

If a new application is being filed and the application includes the necessary components for filing date (see 37 CFR 1.53(b)-(d) and MPEP 506), a Filing Receipt (37 CFR 1.54) will be issued in due course and the date shown on this Acknowledgement Receipt will establish the filing date of the application

National Stage of an International Application under 35 U.S.C. 371

If a timely submission to enter the national stage of an international application is compliant with the conditions of 35 U.S.C. 371 and other applicable requirements a Form PCT/DO/EO/903 indicating acceptance of the application as a national stage submission under 35 U.S.C. 371 will be issued in addition to the Filing Receipt, in due course.

New International Application Filed with the USPTO as a Receiving Office

If a new international application is being filed and the international application includes the necessary components for an international filing date (see PCT Article 11 and MPEP 1810), a Notification of the International Application Number and of the International Filing Date (Form PCT/RO/105) will be issued in due course, subject to prescriptions concerning national security, and the date shown on this Acknowledgement Receipt will establish the international filing date of the application.

UNITED STATES
PATENT AND TRADEMARK OFFICE

P.O. Box 1450
Alexandria, VA 22313 - 1450
www.uspto.gov

ELECTRONIC ACKNOWLEDGEMENT RECEIPT

APPLICATION #	RECEIPT DATE / TIME	ATTORNEY DOCKET #
17/657,981	**01/24/2023 12:16:42 PM ET**	**15635.0008-00000**

Title of Invention

Forward and Rearward Facing Attack Vector Visualization

Application Information

APPLICATION TYPE	Utility - Nonprovisional Application under 35 USC 111(a)	PATENT #	-
CONFIRMATION #	2397	FILED BY	Michelle Daniels
PATENT CENTER #	61482282	FILING DATE	04/05/2022
CUSTOMER #	22852	FIRST NAMED INVENTOR	Avi SHUA
CORRESPONDENCE ADDRESS	-	AUTHORIZED BY	Christopher Johns

Documents

TOTAL DOCUMENTS: 17

DOCUMENT	PAGES	DESCRIPTION	SIZE (KB)
15635.0008-IDS after Allowance.pdf	3	Transmittal Letter	18 KB
15635.0008-00000-sb08.pdf	2	Information Disclosure Statement (IDS) Form (SB08)	34 KB
Warning: This is not a USPTO supplied IDS fillable form. Data in the form cannot be automatically loaded to other USPTO systems.			
NPL-Non-Final Rejection-12-30-2022-15635.0007-00000.pdf	19	Non Patent Literature	549 KB
NPL-Non-Final Rejection-1-3-	22	Non Patent Literature	562 KB

2023-15635.0010-00000.pdf			
NPL-Rani et al..pdf	5	Non Patent Literature	3087 KB
NPL-Wei et al..pdf	6	Non Patent Literature	6285 KB
NPL-Rajasekaran et al..pdf	7	Non Patent Literature	6077 KB
NPL-Almulla et al..pdf	6	Non Patent Literature	5700 KB
NPL-Final Rejection-11-23-2022-15635.0009-00000.pdf	14	Non Patent Literature	432 KB
NPL-WEB-ARCHIVE - IBM - SmartCloud.pdf	33	Non Patent Literature	12233 KB
NPL-Ammons et al..pdf	8	Non Patent Literature	7102 KB
NPL-Pandey et al..pdf	8	Non Patent Literature	13829 KB
NPL-Cui et al..pdf	8	Non Patent Literature	5180 KB
NPL-Fernandez et al. Two Patterns for Cloud Computing.pdf	12	Non Patent Literature	5569 KB
NPL-Kaur et al..pdf	10	Non Patent Literature	10148 KB
NPL-Bugiel et al..pdf	12	Non Patent Literature	12358 KB
NPL-Fernandez - Building a Security Reference Architecture.pdf	25	Non Patent Literature	22917 KB

Digest

DOCUMENT	MESSAGE DIGEST(SHA-512)

15635.0008-IDS after Allowance.pdf	D271C486F402856960163BF98FF3517727CDCB12D709466886210926AD82A83AD3FEB16E7D0D5614512BDACF3AC84979C0383B0BC21A1DB92C91B8F48F5679C7
15635.0008-00000-sb08.pdf	3E7E3EF99F295BC41363E18D906C3AF1F25F1FF5492027B8EC82E20208F7BB9ADB92BE06FAE582DF88F19F5E99A6DF4F098D8F808DC3BECC8AE4B169728F10A1
NPL-Non-Final Rejection-12-30-2022-15635.0007-00000.pdf	9A6CD73D75D08A5CA6F6BCF63BED25C83A43542CB8E47394FBBAD91B78B6097304357D6065BC9398DCEE4BF931393783E263EAED33E94ACE9D153FF6C13C3A08
NPL-Non-Final Rejection-1-3-2023-15635.0010-00000.pdf	3265609DA106CF454EA95F608E5F9025C08DA3681825AE5ED635A908BE661EFA457301E21BB78F6118A47CBB28D7FF111495EA2C7DB85B458FB7B2E53F8D232E
NPL-Rani et al..pdf	F165B7EA73ED18B1BF9C17E9ACD2F2FA33FE2E0F7EEBDE244246DA0559D60F27AA66E681D26858F34B085DE83FB65F53C53D2994B7AD04B9EDE12E80E4A75A87
NPL-Wei et al..pdf	1FFA3361F189451192C4A43B664793936DC9B8BF1ABA89CD8A9039FD0DE9DCE5534D6450F26A618E39F0F9BC027DDDC7B5F581AC48312049AE93AA1C73EBF0B0
NPL-Rajasekaran et al..pdf	97856A5770786D970FB46D606BDA5374C9386B25BE690C85B8CD37E81C63A80650E055EA5D77CBBA7C485CAA82BCD7C855FA4A63036E5297D58D27A7164BF4D8
NPL-Almulla et al..pdf	6F090A77C7C05DA6CF6503DFF5D07BB2EFE06C5DD5268AC3945C984F2C695DB6CB92F718AECB17400CEB6C57CDA1EA503CB2AD762FD4D5FE11566BE4872593CD
NPL-Final Rejection-11-23-2022-15635.0009-00000.pdf	4D18C45D52FB20D4C8CAB9E02B20B4739E6D1EADC260DB86E0EB1D350337A75800EEC72B94EABF72D0953F299693BF711B5599999EA865B12661A22750467A4E
NPL-WEB-ARCHIVE - IBM - SmartCloud.pdf	7FA0D1BBE9E11351BB64A9BA65809AD0A1A3A5E426EE581A715E8775EE5E4D1C3FCC9084A4A5D2709C3EE0DD09F0DE1465DE039F7BEEE196A12A3BF12E3737B6

NPL-Ammons et al..pdf	747103FEF5246626D8D179C457238BAF76C36D62AD2A0514D6190FF22BA536D2572405A42326282628490A389826B53DE623DCAB8019E232707604B8BB822415
NPL-Pandey et al..pdf	49EB3A155A60396F74B3540E1A474FFDC5D8005C2CFD2B5FA1D8C8807B748C817F906CAB6E3A5604D8827D194161CC180A51B6A26ED730FAB1EF007A463D013F
NPL-Cui et al..pdf	2B17CE2E6324A4871FCBF55D89B45E3274BCBA16DA1932A78ECF1247867416EF29C70C8CF0457EF8F04AA13792B7BB1AA59EB9D55B859B352856C0D1ED6CE4B4
NPL-Fernandez et al. Two Patterns for Cloud Computing.pdf	15BACFCB6E27481DD20EA4F6D2151DA3575C9EC290F1BFBB0BF64BA331DD5879754276BEFCC62C3DB4E5860674DA9CFBB077B9067D240262330AD97738370467
NPL-Kaur et al..pdf	D3C013C12B359601247549A613F32188D7B3C06B18252DA9A86640769DB19748DB40DB133193E99444284A6BBF7C6A37DF9BAE0A85E84303F642FD164FA83A4B
NPL-Bugiel et al..pdf	AA12649F8EFE06E31721B40EC057B2A2F579D3DA3DDDBFE0C4562D7FD00B2E28223A9DDA1E50ACFE4156D524C2B4BE4ED7E4D470A0C8F4C335043E038E3A649D
NPL-Fernandez - Building a Security Reference Architecture.pdf	1E2F12564C89D0182932AB3D9816B8047B93F7A20CDBD69478F6C7B9583C9E9AE551AB8F113358B80A27010A13D2C7B5707CD1E3793A776D5FE19DA4962A3BC3

This Acknowledgement Receipt evidences receipt on the noted date by the USPTO of the indicated documents, characterized by the applicant, and including page counts, where applicable. It serves as evidence of receipt similar to a Post Card, as described in MPEP 503.

New Applications Under 35 U.S.C. 111

If a new application is being filed and the application includes the necessary components for filing date (see 37 CFR 1.53(b)-(d) and MPEP 506), a Filing Receipt (37 CFR 1.54) will be issued in due course and the date shown on this Acknowledgement Receipt will establish the filing date of the application

National Stage of an International Application under 35 U.S.C. 371

If a timely submission to enter the national stage of an international application is compliant with the conditions of 35 U.S.C. 371 and other applicable requirements a Form PCT/DO/EO/903 indicating acceptance of the application as a national stage submission under 35 U.S.C. 371 will be issued in addition to the Filing Receipt, in due course.

New International Application Filed with the USPTO as a Receiving Office

If a new international application is being filed and the international application includes the necessary components for an

international filing date (see PCT Article 11 and MPEP 1810), a Notification of the International Application Number and of the International Filing Date (Form PCT/RO/105) will be issued in due course, subject to prescriptions concerning national security, and the date shown on this Acknowledgement Receipt will establish the international filing date of the application.

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
17/657,981	04/05/2022	Avi SHUA	15635.0008-00000	2397

22852 7590 01/05/2023

FINNEGAN, HENDERSON, FARABOW, GARRETT & DUNNER LLP
901 NEW YORK AVENUE, NW
WASHINGTON, DC 20001-4413

EXAMINER	
SU, SARAH	

ART UNIT	PAPER NUMBER
2431	

NOTIFICATION DATE	DELIVERY MODE
01/05/2023	ELECTRONIC

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Notice of the Office communication was sent electronically on above-indicated "Notification Date" to the following e-mail address(es):

regional-desk@finnegan.com

Corrected ***Notice of Allowability***	**Application No.** 17/657,981	**Applicant(s)** SHUA, Avi	
	Examiner SARAH SU	**Art Unit** 2431	**AIA (FITF) Status** Yes

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address--

All claims being allowable, PROSECUTION ON THE MERITS IS (OR REMAINS) CLOSED in this application. If not included herewith (or previously mailed), a Notice of Allowance (PTOL-85) or other appropriate communication will be mailed in due course. **THIS NOTICE OF ALLOWABILITY IS NOT A GRANT OF PATENT RIGHTS.** This application is subject to withdrawal from issue at the initiative of the Office or upon petition by the applicant. See 37 CFR 1.313 and MPEP 1308.

1. ☑ This communication is responsive to IDS f iled 12/22/22.
   ☐ A declaration(s)/affidavit(s) under **37 CFR 1.130(b)** was/were filed on _____.

2. ☐ An election was made by the applicant in response to a restriction requirement set forth during the interview on _____; the restriction requirement and election have been incorporated into this action.

3. ☑ The allowed claim(s) is/are 81-100 . As a result of the allowed claim(s), you may be eligible to benefit from the **Patent Prosecution Highway** program at a participating intellectual property office for the corresponding application. For more information, please see **http://www.uspto.gov/patents/init_events/pph/index.jsp** or send an inquiry to **PPHfeedback@uspto.gov.**

4. ☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

   Certified copies:

   a) ☐ All b) ☐ Some* c) ☐ None of the:

   1. ☐ Certified copies of the priority documents have been received.
   2. ☐ Certified copies of the priority documents have been received in Application No. _____ .
   3. ☐ Copies of the certified copies of the priority documents have been received in this national stage application from the International Bureau (PCT Rule 17.2(a)).

   * Certified copies not received: _____ .

Applicant has THREE MONTHS FROM THE "MAILING DATE" of this communication to file a reply complying with the requirements noted below. Failure to timely comply will result in ABANDONMENT of this application.
THIS THREE-MONTH PERIOD IS NOT EXTENDABLE.

5. ☐ CORRECTED DRAWINGS (as "replacement sheets") must be submitted.
   ☐ including changes required by the attached Examiner's Amendment / Comment or in the Office action of Paper No./Mail Date _____ .

   Identifying indicia such as the application number (see 37 CFR 1.84(c)) should be written on the drawings in the front (not the back) of each sheet. Replacement sheet(s) should be labeled as such in the header according to 37 CFR 1.121(d).

6. ☐ DEPOSIT OF and/or INFORMATION about the deposit of BIOLOGICAL MATERIAL must be submitted. Note the attached Examiner's comment regarding REQUIREMENT FOR THE DEPOSIT OF BIOLOGICAL MATERIAL.

Attachment(s)

1. ☐ Notice of References Cited (PTO-892)
2. ☑ Information Disclosure Statements (PTO/SB/08), Paper No./Mail Date _____.
3. ☐ Examiner's Comment Regarding Requirement for Deposit of Biological Material _____.
4. ☐ Interview Summary (PTO-413), Paper No./Mail Date. _____.
5. ☐ Examiner's Amendment/Comment
6. ☐ Examiner's Statement of Reasons for Allowance
7. ☐ Other _____.

/SARAH SU/
Primary Examiner, Art Unit 2431

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Use as many sheets as necessary)				Complete if Known	
				Application Number	17/657,981
				Filing Date	April 5, 2022
				First Named Inventor	Avi SHUA
				Art Unit	2431
				Examiner Name	2397
Sheet	1	of	1	Attorney Docket Number	**15635.0008-00000**

U.S. PATENTS

Examiner Initials*	Cite No.	Document Number / Number-Kind Code (if known)	Issue Date MM-DD-YYYY	Name of Patentee or Applicant of Cited Document	Pages, Columns, Lines, Where Relevant Passages or Relevant Figures Appear

U.S. PUBLISHED PATENT APPLICATIONS

Examiner Initials*	Cite No.	Document Number / Number-Kind Code (if known)	Publication Date MM-DD-YYYY	Name of Patentee or Applicant of Cited Document	Pages, Columns, Lines, Where Relevant Passages or Relevant Figures Appear

Note: Submission of copies of U.S. Patents and published U.S. Patent Applications is not required.

FOREIGN PATENT DOCUMENTS

Examiner Initials*	Cite No.	Foreign Patent Document: Country Code Number Kind Code (if known)	Publication Date MM-DD-YYYY	Name of Patentee or Applicant of Cited Document	Pages, Columns, Lines, Where Relevant Passages or Relevant Figures Appear	Translation

NON PATENT LITERATURE DOCUMENTS

Examiner Initials*	Cite No.	Include name of the author (in CAPITAL LETTERS), title of the article (when appropriate), title of the item (book, magazine, journal, serial, symposium, catalog, etc.), date, page(s), volume-issue number(s), publisher, city and/or country where published.	Translation
		Notice of Allowance, Application No. 17/716583 dated December 8, 2022, in the United States Patent and Trademark Office. (15635.0011-00000)	N/A

Examiner Signature	/SARAH SU/	Date Considered	12/30/2022

*EXAMINER: Initial if reference considered, whether or not citation is in conformance with MPEP 609. Draw line through citation if not in conformance and not considered. Include copy of this form with next communication to applicant.

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.S/

INFORMATION DISCLOSURE STATEMENT BY APPLICANT *(Use as many sheets as necessary)*				***Complete if Known***	
				Application Number	17/657,981
				Filing Date	April 5, 2022
				First Named Inventor	Avi SHUA
				Art Unit	2431
				Examiner Name	2397
Sheet	1	of	1	*Attorney Docket Number*	**15635.0008-00000**

U.S. PATENTS

Examiner Initials*	Cite No.	Document Number Number-Kind Code *(if known)*	Issue Date MM-DD-YYYY	Name of Patentee or Applicant of Cited Document	Pages, Columns, Lines, Where Relevant Passages or Relevant Figures Appear

U.S. PUBLISHED PATENT APPLICATIONS

Examiner Initials*	Cite No.	Document Number Number-Kind Code *(if known)*	Publication Date MM-DD-YYYY	Name of Patentee or Applicant of Cited Document	Pages, Columns, Lines, Where Relevant Passages or Relevant Figures Appear

Note: Submission of copies of U.S. Patents and published U.S. Patent Applications is not required.

FOREIGN PATENT DOCUMENTS

Examiner Initials*	Cite No.	Foreign Patent Document Country Code Number Kind Code *(if known)*	Publication Date MM-DD-YYYY	Name of Patentee or Applicant of Cited Document	Pages, Columns, Lines, Where Relevant Passages or Relevant Figures Appear	Translation

NON PATENT LITERATURE DOCUMENTS

Examiner Initials*	Cite No.	Include name of the author (in CAPITAL LETTERS), title of the article (when appropriate), title of the item (book, magazine, journal, serial, symposium, catalog, etc.), date, page(s), volume-issue number(s), publisher, city and/or country where published.	Translation
		Notice of Allowance, Application No. 17/716583 dated December 8, 2022, in the United States Patent and Trademark Office. (15635.0011-00000)	N/A

Examiner Signature		Date Considered	

*EXAMINER: Initial if reference considered, whether or not citation is in conformance with MPEP 609. Draw line through citation if not in conformance and not considered. Include copy of this form with next communication to applicant.

UNITED STATES
PATENT AND TRADEMARK OFFICE

P.O. Box 1450
Alexandria, VA 22313 - 1450
www.uspto.gov

ELECTRONIC ACKNOWLEDGEMENT RECEIPT

APPLICATION #	RECEIPT DATE / TIME	ATTORNEY DOCKET #
17/657,981	**12/22/2022 01:35:54 PM ET**	**15635.0008-00000**

Title of Invention

Forward and Rearward Facing Attack Vector Visualization

Application Information

APPLICATION TYPE	Utility - Nonprovisional Application under 35 USC 111(a)	PATENT #	-
CONFIRMATION #	2397	FILED BY	Michelle Daniels
PATENT CENTER #	61342518	FILING DATE	04/05/2022
CUSTOMER #	22852	FIRST NAMED INVENTOR	Avi SHUA
CORRESPONDENCE ADDRESS	-	AUTHORIZED BY	Christopher Johns

Documents

TOTAL DOCUMENTS: 3

DOCUMENT	PAGES	DESCRIPTION	SIZE (KB)
15635.0008-IDS after Allowance.pdf	3	Transmittal Letter	19 KB
sb08.pdf	1	Information Disclosure Statement (IDS) Form (SB08)	30 KB
Notice of Allowance-12-8-2022-15635.0011-00000.pdf	13	Non Patent Literature	511 KB

Digest

DOCUMENT	MESSAGE DIGEST(SHA-512)

15635.0008-IDS after Allowance.pdf	E73F4125369941 9C76C0B67F1768DB8E9BCAD096DC3C78C92 3C3128FDDD47421816F898A847CC1302CDAB0F690652FAACD 5E05A1247675181EBBEA9FD71B5BF8
sb08.pdf	6FF62834EADBACC4C04A836940B80EDCAC220880A92C2781 D149E56C0A9AEACA56566F2AA0FA26E2F88E44CBDD219260 A315DC6F69FF02A69D949ECAB9A80F0F
Notice of Allowance-12-8-2022-15635.0011-00000.pdf	CC7213A7E65B267DBEBFD349AAE4F20680DFD4F78253EF028 5BD2E2A18C2F06641E27FF9960061BF247FC370E2FEE341B40 ED3AF98B6C73346F29678AAE5F425

This Acknowledgement Receipt evidences receipt on the noted date by the USPTO of the indicated documents, characterized by the applicant, and including page counts, where applicable. It serves as evidence of receipt similar to a Post Card, as described in MPEP 503.

New Applications Under 35 U.S.C. 111

If a new application is being filed and the application includes the necessary components for filing date (see 37 CFR 1.53(b)-(d) and MPEP 506), a Filing Receipt (37 CFR 1.54) will be issued in due course and the date shown on this Acknowledgement Receipt will establish the filing date of the application

National Stage of an International Application under 35 U.S.C. 371

If a timely submission to enter the national stage of an international application is compliant with the conditions of 35 U.S.C. 371 and other applicable requirements a Form PCT/DO/EO/903 indicating acceptance of the application as a national stage submission under 35 U.S.C. 371 will be issued in addition to the Filing Receipt, in due course.

New International Application Filed with the USPTO as a Receiving Office

If a new international application is being filed and the international application includes the necessary components for an international filing date (see PCT Article 11 and MPEP 1810), a Notification of the International Application Number and of the International Filing Date (Form PCT/RO/105) will be issued in due course, subject to prescriptions concerning national security, and the date shown on this Acknowledgement Receipt will establish the international filing date of the application.

PATENT
Attorney Docket No. 15635.0008-00000

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Application of:	)	
	)	
Avi SHUA	)	Group Art Unit: 2431
	)	
Application No.: 17/657,981	)	Examiner: SU, SARAH
	)	
Filed: April 5, 2022	)	Confirmation No.: 2397
	)	
For: FORWARD AND REARWARD FACING ATTACK VECTOR VISUALIZATION	)	

Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Commissioner:

SUPPLEMENTAL INFORMATION DISCLOSURE STATEMENT UNDER 37 C.F.R. § 1.97(d)

Pursuant to 37 C.F.R. §§ 1.56 and 1.97(d), Applicant brings to the attention of the Examiner the documents on the attached listing. This Information Disclosure Statement is being filed after a Notice of Allowance and is accompanied by a fee of $260.00 as specified under § 1.17(p) and a statement as specified under § 1.97(e).

Statement under 37 C.F.R. § 1.97(e)

The following constitutes the statement specified by 37 C.F.R. § 1.97(e)(2). No item of information contained in the IDS was cited in a communication from a foreign patent office in a counterpart foreign application, and, to the knowledge of the person signing the certification after making reasonable inquiry, no item of information contained in the IDS was known to any individual designated in 37 CFR 1.56(c) more

than three months prior to the filing of this Information Disclosure Statement. See 37 CFR 1.97(e)(2).

A copy of the non-patent literature document is attached.

Applicant respectfully requests that the Examiner consider the listed documents on the attached Form PTO/SB/08 and indicate that they were considered by making appropriate notations on the attached form.

This submission does not represent that a search has been made or that no better art exists and does not constitute an admission that each or all of the listed documents are material or constitute "prior art." If the Examiner applies any of the documents as prior art against any claim in the application and Applicant determines that the cited documents do not constitute "prior art" under United States law, Applicant reserves the right to present to the U.S. Patent and Trademark Office the relevant facts and law regarding the appropriate status of such documents.

Applicant further reserves the right to take appropriate action to establish the patentability of the disclosed invention over the listed documents, should one or more of the documents be applied against the claims of the present application.

If there is any fee due in connection with the filing of this Statement, please charge the fee to deposit account 06-0916.

Respectfully submitted,

FINNEGAN, HENDERSON, FARABOW,
GARRETT & DUNNER, L.L.P.

Dated: December 22, 2022

By: /Christopher C. Johns/
Christopher C. Johns
Reg. No. 68,664
(202) 408-4000

UNITED STATES
PATENT AND TRADEMARK OFFICE

P.O. Box 1450
Alexandria, VA 22313 - 1450
www.uspto.gov

ELECTRONIC PAYMENT RECEIPT

APPLICATION #	RECEIPT DATE / TIME	ATTORNEY DOCKET #
17/657,981	12/22/2022 01:35:54 PM ET	15635.0008-00000

Title of Invention

Forward and Rearward Facing Attack Vector Visualization

Application Information

APPLICATION TYPE	Utility - Nonprovisional Application under 35 USC 111(a)	PATENT #	-
CONFIRMATION #	2397	FILED BY	Michelle Daniels
PATENT CENTER #	61342518	AUTHORIZED BY	Christopher Johns
CUSTOMER #	22852	FILING DATE	04/05/2022
CORRESPONDENCE ADDRESS	-	FIRST NAMED INVENTOR	Avi SHUA

Payment Information

PAYMENT METHOD	PAYMENT TRANSACTION ID	PAYMENT AUTHORIZED BY
CARD / 1007	E2022BLD36394707	Michelle Daniels

FEE CODE	DESCRIPTION	ITEM PRICE($)	QUANTITY	ITEM TOTAL($)
1806	SUBMISSION OF AN INFORMATION DISCLOSURE STATEMENT	260.00	1	260.00
			TOTAL AMOUNT:	$260.00

This Acknowledgement Receipt evidences receipt on the noted date by the USPTO of the indicated documents, characterized by the applicant, and including page counts, where applicable. It serves as evidence of receipt similar to a Post Card, as described in MPEP 503.

New Applications Under 35 U.S.C. 111

If a new application is being filed and the application includes the necessary components for filing date (see 37 CFR 1.53(b)-(d) and MPEP 506), a Filing Receipt (37 CFR 1.54) will be issued in due course and the date shown on this Acknowledgement Receipt will establish the filing date of the application

National Stage of an International Application under 35 U.S.C. 371

If a timely submission to enter the national stage of an international application is compliant with the conditions of 35 U.S.C. 371 and other applicable requirements a Form PCT/DO/EO/903 indicating acceptance of the application as a national stage submission under 35 U.S.C. 371 will be issued in addition to the Filing Receipt, in due course.

New International Application Filed with the USPTO as a Receiving Office

If a new international application is being filed and the international application includes the necessary components for an international filing date (see PCT Article 11 and MPEP 1810), a Notification of the International Application Number and of the International Filing Date (Form PCT/RO/105) will be issued in due course, subject to prescriptions concerning national security, and the date shown on this Acknowledgement Receipt will establish the international filing date of the application.

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

NOTICE OF ALLOWANCE AND FEE(S) DUE

22852 7590 12/07/2022

FINNEGAN, HENDERSON, FARABOW, GARRETT & DUNNER LLP
901 NEW YORK AVENUE, NW
WASHINGTON, DC 20001-4413

EXAMINER
SU, SARAH

ART UNIT	PAPER NUMBER
2431	

DATE MAILED: 12/07/2022

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
17/657,981	04/05/2022	Avi SHUA	15635.0008-00000	2397

TITLE OF INVENTION: Forward and Rearward Facing Attack Vector Visualization

APPLN. TYPE	ENTITY STATUS	ISSUE FEE DUE	PUBLICATION FEE DUE	PREV. PAID ISSUE FEE	TOTAL FEE(S) DUE	DATE DUE
nonprovisional	UNDISCOUNTED	$1200	$0.00	$0.00	$1200	03/07/2023

THE APPLICATION IDENTIFIED ABOVE HAS BEEN EXAMINED AND IS ALLOWED FOR ISSUANCE AS A PATENT. PROSECUTION ON THE MERITS IS CLOSED. THIS NOTICE OF ALLOWANCE IS NOT A GRANT OF PATENT RIGHTS. THIS APPLICATION IS SUBJECT TO WITHDRAWAL FROM ISSUE AT THE INITIATIVE OF THE OFFICE OR UPON PETITION BY THE APPLICANT. SEE 37 CFR 1.313 AND MPEP 1308.

THE ISSUE FEE AND PUBLICATION FEE (IF REQUIRED) MUST BE PAID WITHIN THREE MONTHS FROM THE MAILING DATE OF THIS NOTICE OR THIS APPLICATION SHALL BE REGARDED AS ABANDONED. THIS STATUTORY PERIOD CANNOT BE EXTENDED. SEE 35 U.S.C. 151. THE ISSUE FEE DUE INDICATED ABOVE DOES NOT REFLECT A CREDIT FOR ANY PREVIOUSLY PAID ISSUE FEE IN THIS APPLICATION. IF AN ISSUE FEE HAS PREVIOUSLY BEEN PAID IN THIS APPLICATION (AS SHOWN ABOVE), THE RETURN OF PART B OF THIS FORM WILL BE CONSIDERED A REQUEST TO REAPPLY THE PREVIOUSLY PAID ISSUE FEE TOWARD THE ISSUE FEE NOW DUE.

HOW TO REPLY TO THIS NOTICE:

I. Review the ENTITY STATUS shown above. If the ENTITY STATUS is shown as SMALL or MICRO, verify whether entitlement to that entity status still applies.

If the ENTITY STATUS is the same as shown above, pay the TOTAL FEE(S) DUE shown above.

If the ENTITY STATUS is changed from that shown above, on PART B - FEE(S) TRANSMITTAL, complete section number 5 titled "Change in Entity Status (from status indicated above)".

For purposes of this notice, small entity fees are 1/2 the amount of undiscounted fees, and micro entity fees are 1/2 the amount of small entity fees.

II. PART B - FEE(S) TRANSMITTAL, or its equivalent, must be completed and returned to the United States Patent and Trademark Office (USPTO) with your ISSUE FEE and PUBLICATION FEE (if required). If you are charging the fee(s) to your deposit account, section "4b" of Part B - Fee(s) Transmittal should be completed. If an equivalent of Part B is filed, a request to reapply a previously paid issue fee must be clearly made, and delays in processing may occur due to the difficulty in recognizing the paper as an equivalent of Part B.

III. All communications regarding this application must give the application number. Please direct all communications prior to issuance to Mail Stop ISSUE FEE unless advised to the contrary.

IMPORTANT REMINDER: Maintenance fees are due in utility patents issuing on applications filed on or after Dec. 12, 1980. It is patentee's responsibility to ensure timely payment of maintenance fees when due. More information is available at www.uspto.gov/PatentMaintenanceFees.

PART B - FEE(S) TRANSMITTAL

Complete and send this form, together with applicable fee(s), by mail or fax, or via EFS-Web.

By mail, send to: Mail Stop ISSUE FEE
Commissioner for Patents
P.O. Box 1450
Alexandria, Virginia 22313-1450

By fax, send to: (571)-273-2885

INSTRUCTIONS: This form should be used for transmitting the ISSUE FEE and PUBLICATION FEE (if required). Blocks 1 through 5 should be completed where appropriate. All further correspondence including the Patent, advance orders and notification of maintenance fees will be mailed to the current correspondence address as indicated unless corrected below or directed otherwise in Block 1, by (a) specifying a new correspondence address; and/or (b) indicating a separate "FEE ADDRESS" for maintenance fee notifications.

CURRENT CORRESPONDENCE ADDRESS (Note: Use Block 1 for any change of address)

22852 7590 12/07/2022

FINNEGAN, HENDERSON, FARABOW, GARRETT & DUNNER LLP
901 NEW YORK AVENUE, NW
WASHINGTON, DC 20001-4413

Note: A certificate of mailing can only be used for domestic mailings of the Fee(s) Transmittal. This certificate cannot be used for any other accompanying papers. Each additional paper, such as an assignment or formal drawing, must have its own certificate of mailing or transmission.

Certificate of Mailing or Transmission

I hereby certify that this Fee(s) Transmittal is being deposited with the United States Postal Service with sufficient postage for first class mail in an envelope addressed to the Mail Stop ISSUE FEE address above, or being transmitted to the USPTO via EFS-Web or by facsimile to (571) 273-2885, on the date below.

(Typed or printed name)
(Signature)
(Date)

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
17/657,981	04/05/2022	Avi SHUA	15635.0008-00000	2397

TITLE OF INVENTION: Forward and Rearward Facing Attack Vector Visualization

APPLN. TYPE	ENTITY STATUS	ISSUE FEE DUE	PUBLICATION FEE DUE	PREV. PAID ISSUE FEE	TOTAL FEE(S) DUE	DATE DUE
nonprovisional	UNDISCOUNTED	$1200	$0.00	$0.00	$1200	03/07/2023

EXAMINER	ART UNIT	CLASS-SUBCLASS
SU, SARAH	2431	726-025000

1. Change of correspondence address or indication of "Fee Address" (37 CFR 1.363).

❑ Change of correspondence address (or Change of Correspondence Address form PTO/AIA/122 or PTO/SB/122) attached.

❑ "Fee Address" indication (or "Fee Address" Indication form PTO/AIA/47 or PTO/SB/47; Rev 03-02 or more recent) attached. **Use of a Customer Number is required.**

2. For printing on the patent front page, list
(1) The names of up to 3 registered patent attorneys or agents OR, alternatively,
(2) The name of a single firm (having as a member a registered attorney or agent) and the names of up to 2 registered patent attorneys or agents. If no name is listed, no name will be printed.

1 _____
2 _____
3 _____

3. ASSIGNEE NAME AND RESIDENCE DATA TO BE PRINTED ON THE PATENT (print or type)

PLEASE NOTE: Unless an assignee is identified below, no assignee data will appear on the patent. If an assignee is identified below, the document must have been previously recorded, or filed for recordation, as set forth in 37 CFR 3.11 and 37 CFR 3.81(a). Completion of this form is NOT a substitute for filing an assignment.

(A) NAME OF ASSIGNEE (B) RESIDENCE: (CITY and STATE OR COUNTRY)

Please check the appropriate assignee category or categories (will not be printed on the patent) : ❑ Individual ❑ Corporation or other private group entity ❑ Government

4a. Fees submitted: ❑ Issue Fee ❑ Publication Fee (if required) ❑ Advance Order - # of Copies _____

4b. Method of Payment: *(Please first reapply any previously paid fee shown above)*

❑ Electronic Payment via EFS-Web ❑ Enclosed check ❑ Non-electronic payment by credit card (Attach form PTO-2038)

❑ The Director is hereby authorized to charge the required fee(s), any deficiency, or credit any overpayment to Deposit Account No. _____

5. **Change in Entity Status** (from status indicated above)

❑ Applicant certifying micro entity status. See 37 CFR 1.29

❑ Applicant asserting small entity status. See 37 CFR 1.27

❑ Applicant changing to regular undiscounted fee status.

NOTE: Absent a valid certification of Micro Entity Status (see forms PTO/SB/15A and 15B), issue fee payment in the micro entity amount will not be accepted at the risk of application abandonment.
NOTE: If the application was previously under micro entity status, checking this box will be taken to be a notification of loss of entitlement to micro entity status.
NOTE: Checking this box will be taken to be a notification of loss of entitlement to small or micro entity status, as applicable.

NOTE: This form must be signed in accordance with 37 CFR 1.31 and 1.33. See 37 CFR 1.4 for signature requirements and certifications.

Authorized Signature _____ Date _____

Typed or printed name _____ Registration No. _____

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
17/657,981	04/05/2022	Avi SHUA	15635.0008-00000	2397

22852 7590 12/07/2022

FINNEGAN, HENDERSON, FARABOW, GARRETT & DUNNER LLP
901 NEW YORK AVENUE, NW
WASHINGTON, DC 20001-4413

EXAMINER	
SU, SARAH	

ART UNIT	PAPER NUMBER
2431	

DATE MAILED: 12/07/2022

Determination of Patent Term Adjustment under 35 U.S.C. 154 (b)
(Applications filed on or after May 29, 2000)

The Office has discontinued providing a Patent Term Adjustment (PTA) calculation with the Notice of Allowance.

Section 1(h)(2) of the AIA Technical Corrections Act amended 35 U.S.C. 154(b)(3)(B)(i) to eliminate the requirement that the Office provide a patent term adjustment determination with the notice of allowance. See Revisions to Patent Term Adjustment, 78 Fed. Reg. 19416, 19417 (Apr. 1, 2013). Therefore, the Office is no longer providing an initial patent term adjustment determination with the notice of allowance. The Office will continue to provide a patent term adjustment determination with the Issue Notification Letter that is mailed to applicant approximately three weeks prior to the issue date of the patent, and will include the patent term adjustment on the patent. Any request for reconsideration of the patent term adjustment determination (or reinstatement of patent term adjustment) should follow the process outlined in 37 CFR 1.705.

Any questions regarding the Patent Term Extension or Adjustment determination should be directed to the Office of Patent Legal Administration at (571)-272-7702. Questions relating to issue and publication fee payments should be directed to the Customer Service Center of the Office of Patent Publication at 1-(888)-786-0101 or (571)-272-4200.

OMB Clearance and PRA Burden Statement for PTOL-85 Part B

The Paperwork Reduction Act (PRA) of 1995 requires Federal agencies to obtain Office of Management and Budget approval before requesting most types of information from the public. When OMB approves an agency request to collect information from the public, OMB (i) provides a valid OMB Control Number and expiration date for the agency to display on the instrument that will be used to collect the information and (ii) requires the agency to inform the public about the OMB Control Number's legal significance in accordance with 5 CFR 1320.5(b).

The information collected by PTOL-85 Part B is required by 37 CFR 1.311. The information is required to obtain or retain a benefit by the public which is to file (and by the USPTO to process) an application. Confidentiality is governed by 35 U.S.C. 122 and 37 CFR 1.14. This collection is estimated to take 30 minutes to complete, including gathering, preparing, and submitting the completed application form to the USPTO. Time will vary depending upon the individual case. Any comments on the amount of time you require to complete this form and/or suggestions for reducing this burden, should be sent to the Chief Information Officer, U.S. Patent and Trademark Office, U.S. Department of Commerce, P.O. Box 1450, Alexandria, Virginia 22313-1450. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. SEND TO: Commissioner for Patents, P.O. Box 1450, Alexandria, Virginia 22313-1450. Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number.

Privacy Act Statement

The Privacy Act of 1974 (P.L. 93-579) requires that you be given certain information in connection with your submission of the attached form related to a patent application or patent. Accordingly, pursuant to the requirements of the Act, please be advised that: (1) the general authority for the collection of this information is 35 U.S.C. 2(b)(2); (2) furnishing of the information solicited is voluntary; and (3) the principal purpose for which the information is used by the U.S. Patent and Trademark Office is to process and/or examine your submission related to a patent application or patent. If you do not furnish the requested information, the U.S. Patent and Trademark Office may not be able to process and/or examine your submission, which may result in termination of proceedings or abandonment of the application or expiration of the patent.

The information provided by you in this form will be subject to the following routine uses:

1. The information on this form will be treated confidentially to the extent allowed under the Freedom of Information Act (5 U.S.C. 552) and the Privacy Act (5 U.S.C 552a). Records from this system of records may be disclosed to the Department of Justice to determine whether disclosure of these records is required by the Freedom of Information Act.
2. A record from this system of records may be disclosed, as a routine use, in the course of presenting evidence to a court, magistrate, or administrative tribunal, including disclosures to opposing counsel in the course of settlement negotiations.
3. A record in this system of records may be disclosed, as a routine use, to a Member of Congress submitting a request involving an individual, to whom the record pertains, when the individual has requested assistance from the Member with respect to the subject matter of the record.
4. A record in this system of records may be disclosed, as a routine use, to a contractor of the Agency having need for the information in order to perform a contract. Recipients of information shall be required to comply with the requirements of the Privacy Act of 1974, as amended, pursuant to 5 U.S.C. 552a(m).
5. A record related to an International Application filed under the Patent Cooperation Treaty in this system of records may be disclosed, as a routine use, to the International Bureau of the World Intellectual Property Organization, pursuant to the Patent Cooperation Treaty.
6. A record in this system of records may be disclosed, as a routine use, to another federal agency for purposes of National Security review (35 U.S.C. 181) and for review pursuant to the Atomic Energy Act (42 U.S.C. 218(c)).
7. A record from this system of records may be disclosed, as a routine use, to the Administrator, General Services, or his/her designee, during an inspection of records conducted by GSA as part of that agency's responsibility to recommend improvements in records management practices and programs, under authority of 44 U.S.C. 2904 and 2906. Such disclosure shall be made in accordance with the GSA regulations governing inspection of records for this purpose, and any other relevant (i.e., GSA or Commerce) directive. Such disclosure shall not be used to make determinations about individuals.
8. A record from this system of records may be disclosed, as a routine use, to the public after either publication of the application pursuant to 35 U.S.C. 122(b) or issuance of a patent pursuant to 35 U.S.C. 151. Further, a record may be disclosed, subject to the limitations of 37 CFR 1.14, as a routine use, to the public if the record was filed in an application which became abandoned or in which the proceedings were terminated and which application is referenced by either a published application, an application open to public inspection or an issued patent.
9. A record from this system of records may be disclosed, as a routine use, to a Federal, State, or local law enforcement agency, if the USPTO becomes aware of a violation or potential violation of law or regulation.

Notice of Allowability	Application No. 17/657,981	Applicant(s) SHUA, Avi	
	Examiner SARAH SU	Art Unit 2431	AIA (FITF) Status Yes

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address--

All claims being allowable, PROSECUTION ON THE MERITS IS (OR REMAINS) CLOSED in this application. If not included herewith (or previously mailed), a Notice of Allowance (PTOL-85) or other appropriate communication will be mailed in due course. **THIS NOTICE OF ALLOWABILITY IS NOT A GRANT OF PATENT RIGHTS.** This application is subject to withdrawal from issue at the initiative of the Office or upon petition by the applicant. See 37 CFR 1.313 and MPEP 1308.

1. ☑ This communication is responsive to amendment filed 17 November 2022.
   ☐ A declaration(s)/affidavit(s) under **37 CFR 1.130(b)** was/were filed on _____.

2. ☐ An election was made by the applicant in response to a restriction requirement set forth during the interview on _____; the restriction requirement and election have been incorporated into this action.

3. ☑ The allowed claim(s) is/are 81-100 . As a result of the allowed claim(s), you may be eligible to benefit from the **Patent Prosecution Highway** program at a participating intellectual property office for the corresponding application. For more information, please see **http://www.uspto.gov/patents/init_events/pph/index.jsp** or send an inquiry to **PPHfeedback@uspto.gov.**

4. ☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

   Certified copies:

   a) ☐ All b) ☐ Some* c) ☐ None of the:

   1. ☐ Certified copies of the priority documents have been received.
   2. ☐ Certified copies of the priority documents have been received in Application No. _____ .
   3. ☐ Copies of the certified copies of the priority documents have been received in this national stage application from the International Bureau (PCT Rule 17.2(a)).

   * Certified copies not received: _____ .

Applicant has THREE MONTHS FROM THE "MAILING DATE" of this communication to file a reply complying with the requirements noted below. Failure to timely comply will result in ABANDONMENT of this application.
THIS THREE-MONTH PERIOD IS NOT EXTENDABLE.

5. ☐ CORRECTED DRAWINGS (as "replacement sheets") must be submitted.
   ☐ including changes required by the attached Examiner's Amendment / Comment or in the Office action of Paper No./Mail Date _____ .

   Identifying indicia such as the application number (see 37 CFR 1.84(c)) should be written on the drawings in the front (not the back) of each sheet. Replacement sheet(s) should be labeled as such in the header according to 37 CFR 1.121(d).

6. ☐ DEPOSIT OF and/or INFORMATION about the deposit of BIOLOGICAL MATERIAL must be submitted. Note the attached Examiner's comment regarding REQUIREMENT FOR THE DEPOSIT OF BIOLOGICAL MATERIAL.

Attachment(s)

1. ☑ Notice of References Cited (PTO-892)
2. ☑ Information Disclosure Statements (PTO/SB/08), Paper No./Mail Date _____.
3. ☐ Examiner's Comment Regarding Requirement for Deposit of Biological Material _____.
4. ☐ Interview Summary (PTO-413), Paper No./Mail Date. _____.
5. ☐ Examiner's Amendment/Comment
6. ☑ Examiner's Statement of Reasons for Allowance
7. ☐ Other _____.

/SARAH SU/
Primary Examiner, Art Unit 2431

NOTICE OF ALLOWANCE

Notice of Pre-AIA or AIA Status

1. The present application, filed on or after March 16, 2013, is being examined under the first inventor to file provisions of the AIA.

2. Amendment A, received on 17 November 2022, has been entered into record.

3. Claims 81-100 are presented for examination.

Response to Arguments

4. With regards to the objections to the specification and drawings, the applicant has submitted amendments, and the examiner hereby withdraws the objections.

Allowable Subject Matter

5. Claims 81-100 are allowed.

6. This communication warrants No Examiner's Reason for Allowance, previous Office Action mailed 31 October 2022 make evident the reasons for allowance, satisfying the “record as a whole” proviso of the rule 37 CFR 1.104(e). As such the reasons for allowance are in all probability evident from the record and no statement is deemed necessary (see MPEP 1302.14).

Any comments considered necessary by applicant must be submitted no later than the payment of the issue fee and, to avoid processing delays, should preferably accompany the issue fee. Such submissions should be clearly labeled “Comments on Statement of Reasons for Allowance.”

Prior Art Made of Record

7. The prior art made of record and not relied upon is considered pertinent to applicant's disclosure.

a. Fang et al. (CN 114584359 A) discloses a system and method for safety trapping.

b. Shua (US 2022/0345483 A1) discloses a system and method for prioritizing internet-accessible workloads for cyber security.

c. Xu et al. (CN 109995722 A) discloses a system and method for mass detection data analysis for APT protection.

Conclusion

8. Any inquiry concerning this communication or earlier communications from the examiner should be directed to SARAH SU whose telephone number is (571)270-3835. The examiner can normally be reached 7:30 AM - 4:00 PM.

Examiner interviews are available via telephone, in-person, and video conferencing using a USPTO supplied web-based collaboration tool. To schedule an interview, applicant is encouraged to use the USPTO Automated Interview Request (AIR) at http://www.uspto.gov/interviewpractice.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Lynn Feild can be reached on 571-272-2092. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of published or unpublished applications may be obtained from Patent Center. Unpublished application information in Patent Center is available to registered users. To file and manage patent submissions in Patent Center, visit: https://patentcenter.uspto.gov. Visit https://www.uspto.gov/patents/apply/patent-center for more information about Patent Center and

https://www.uspto.gov/patents/docx for information about filing in DOCX format. For additional questions, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

/SARAH SU/
Primary Examiner, Art Unit 2431

Notice of References Cited	Application/Control No. 17/657,981	Applicant(s)/Patent Under Reexamination SHUA, Avi	
	Examiner SARAH SU	Art Unit 2431	Page 1 of 1

U.S. PATENT DOCUMENTS

*		Document Number Country Code-Number-Kind Code	Date MM-YYYY	Name	CPC Classification	US Classification
*	A	US-20220345483-A1	10-2022	Shua; Avi	G06F21/554	1/1
	B					
	C					
	D					
	E					
	F					
	G					
	H					
	I					
	J					
	K					
	L					
	M					

FOREIGN PATENT DOCUMENTS

*		Document Number Country Code-Number-Kind Code	Date MM-YYYY	Country	Name	CPC Classification
	N	CN-114584359-A	06-2022	CN	FANG Y	
	O	CN-109995722-A	07-2019	CN	CHEN D	
	P					
	Q					
	R					
	S					
	T					

NON-PATENT DOCUMENTS

*		Include as applicable: Author, Title Date, Publisher, Edition or Volume, Pertinent Pages)
	U	
	V	
	W	
	X	

*A copy of this reference is not being furnished with this Office action. (See MPEP § 707.05(a).)
Dates in MM-YYYY format are publication dates. Classifications may be US or foreign.

Search Notes	Application/Control No.	Applicant(s)/Patent Under Reexamination
	17/657,981	SHUA, Avi
	Examiner	Art Unit
	SARAH SU	2431

CPC - Searched*		
Symbol	**Date**	**Examiner**
H04L 63/1433, 14$2	10/25/2022	SS
H04L 9/45558, 5072	10/25/2022	SS
G06F 2009/4557, 45595	10/25/2022	SS

CPC Combination Sets - Searched*		
Symbol	**Date**	**Examiner**

US Classification - Searched*			
Class	**Subclass**	**Date**	**Examiner**

* See search history printout included with this form or the SEARCH NOTES box below to determine the scope of the search.

Search Notes		
Search Notes	**Date**	**Examiner**
Inventor Search	10/25/2022	SS
IP.com Search	10/25/2022	SS
PE2E Search (see attached)	10/25/2022	SS
H04L 63/1433, 14$2 (see search history notes to determine scope)	10/25/2022	SS
H04L 9/45558, 5072 (see search history notes to determine scope)	10/25/2022	SS
G06F 2009/4557, 45595 (see search history notes to determine scope)	10/25/2022	SS
PE2E search (see attached)	11/23/2022	SS
Interference Search	11/23/2022	SS
NPL Search (See attached)	11/23/2022	SS

Search Notes	Application/Control No.	Applicant(s)/Patent Under Reexamination
	17/657,981	SHUA, Avi
	Examiner	Art Unit
	SARAH SU	2431

Interference Search			
US Class/CPC Symbol	US Subclass/CPC Group	Date	Examiner
H04L	63/1433, 14$2	11/23/2022	SS
H04L	9/45558, 5072	11/23/2022	SS
G06F	2009/4557, 45595	11/23/2022	SS

Issue Classification	Application/Control No.	Applicant(s)/Patent Under Reexamination
	17/657,981	SHUA, Avi
	Examiner	**Art Unit**
	SARAH SU	2431

CPC						
Symbol					**Type**	**Version**
H04L	/	63	/	1433	F	2013-01-01
H04L	/	63	/	10	I	2013-01-01
H04L	/	63	/	145	I	2013-01-01
H04L	/	67	/	1008	I	2013-01-01
H04L	/	67	/	101	I	2013-01-01
G06F	/	9	/	45558	I	2013-01-01
G06F	/	9	/	5072	I	2013-01-01
H04L	/	63	/	1441	I	2013-01-01
G06F	/	21	/	54	I	2013-01-01
G06F	/	21	/	554	I	2013-01-01
G06F	/	21	/	568	I	2013-01-01
G06F	/	21	/	78	I	2013-01-01
G06F	/	2009	/	4557	A	2013-01-01
G06F	/	2009	/	45595	A	2013-01-01

CPC Combination Sets							
Symbol					**Type**	**Set**	**Ranking**	**Version**
	/		/					

NONE		**Total Claims Allowed:**	
(Assistant Examiner)	(Date)	20	
/SARAH SU/ Primary Examiner, Art Unit 2431	23 November 2022	O.G. Print Claim(s)	O.G. Print Figure
(Primary Examiner)	(Date)	81	2d

U.S. Patent and Trademark Office

Part of Paper No.: 20221123

Issue Classification	Application/Control No.	Applicant(s)/Patent Under Reexamination
	17/657,981	SHUA, Avi
	Examiner	**Art Unit**
	SARAH SU	2431

INTERNATIONAL CLASSIFICATION				
CLAIMED				
H04L	/	9	/	40

NON-CLAIMED				
H04L67/1008	/	67	/	1008

US ORIGINAL CLASSIFICATION	
CLASS	**SUBCLASS**

CROSS REFERENCES(S)						
CLASS	**SUBCLASS (ONE SUBCLASS PER BLOCK)**					

NONE (Assistant Examiner)	(Date)	**Total Claims Allowed:** 20	
/SARAH SU/ Primary Examiner, Art Unit 2431 (Primary Examiner)	23 November 2022 (Date)	O.G. Print Claim(s) 81	O.G. Print Figure 2d

U.S. Patent and Trademark Office

Part of Paper No.: 20221123

Issue Classification	Application/Control No.	Applicant(s)/Patent Under Reexamination
	17/657,981	SHUA, Avi
	Examiner	**Art Unit**
	SARAH SU	2431

☐ Claims renumbered in the same order as presented by applicant ☐ CPA ☐ T.D. ☐ R.1.47

CLAIMS

Final	Original	Final	Original	Final	Original	Final	Original	Final	Original	Final	Original	Final	Original	Final	Original
	1		14		27		40		53		66		79	12	92
	2		15		28		41		54		67		80	13	93
	3		16		29		42		55		68	1	81	14	94
	4		17		30		43		56		69	2	82	15	95
	5		18		31		44		57		70	3	83	16	96
	6		19		32		45		58		71	4	84	17	97
	7		20		33		46		59		72	5	85	18	98
	8		21		34		47		60		73	6	86	19	99
	9		22		35		48		61		74	8	87	20	100
	10		23		36		49		62		75	9	88		
	11		24		37		50		63		76	10	89		
	12		25		38		51		64		77	7	90		
	13		26		39		52		65		78	11	91		

NONE (Assistant Examiner)	(Date)	**Total Claims Allowed:** 20	
/SARAH SU/ Primary Examiner, Art Unit 2431 (Primary Examiner)	23 November 2022 (Date)	O.G. Print Claim(s) 81	O.G. Print Figure 2d

Index of Claims	Application/Control No.	Applicant(s)/Patent Under Reexamination
	17/657,981	SHUA, Avi
	Examiner	Art Unit
	SARAH SU	2431

✓	Rejected	-	Cancelled	N	Non-Elected	A	Appeal
=	Allowed	÷	Restricted	I	Interference	O	Objected

CLAIMS

☐ Claims renumbered in the same order as presented by applicant ☐ CPA ☐ T.D. ☐ R.1.47

CLAIM		DATE								
Final	Original	10/25/2022	11/23/2022							
	1	-	-							
	2	-	-							
	3	-	-							
	4	-	-							
	5	-	-							
	6	-	-							
	7	-	-							
	8	-	-							
	9	-	-							
	10	-	-							
	11	-	-							
	12	-	-							
	13	-	-							
	14	-	-							
	15	-	-							
	16	-	-							
	17	-	-							
	18	-	-							
	19	-	-							
	20	-	-							
	21	-	-							
	22	-	-							
	23	-	-							
	24	-	-							
	25	-	-							
	26	-	-							
	27	-	-							
	28	-	-							
	29	-	-							
	30	-	-							
	31	-	-							
	32	-	-							
	33	-	-							
	34	-	-							
	35	-	-							
	36	-	-							
	37	-	-							
	38	-	-							
	39	-	-							
	40	-	-							
	41	-	-							
	42	-	-							

Index of Claims	Application/Control No.	Applicant(s)/Patent Under Reexamination
	17/657,981	SHUA, Avi
	Examiner	Art Unit
	SARAH SU	2431

CLAIM		DATE								
Final	Original	10/25/2022	11/23/2022							
	43	-	-							
	44	-	-							
	45	-	-							
	46	-	-							
	47	-	-							
	48	-	-							
	49	-	-							
	50	-	-							
	51	-	-							
	52	-	-							
	53	-	-							
	54	-	-							
	55	-	-							
	56	-	-							
	57	-	-							
	58	-	-							
	59	-	-							
	60	-	-							
	61	-	-							
	62	-	-							
	63	-	-							
	64	-	-							
	65	-	-							
	66	-	-							
	67	-	-							
	68	-	-							
	69	-	-							
	70	-	-							
	71	-	-							
	72	-	-							
	73	-	-							
	74	-	-							
	75	-	-							
	76	-	-							
	77	-	-							
	78	-	-							
	79	-	-							
	80	-	-							
1	81	=	=							
2	82	=	=							
3	83	=	=							
4	84	=	=							
5	85	=	=							
6	86	=	=							
8	87	=	=							
9	88	=	=							
10	89	=	=							
7	90	=	=							
11	91	=	=							
12	92	=	=							
13	93	=	=							
14	94	=	=							

Index of Claims	Application/Control No.	Applicant(s)/Patent Under Reexamination
	17/657,981	SHUA, Avi
	Examiner	**Art Unit**
	SARAH SU	2431

CLAIM		DATE								
Final	Original	10/25/2022	11/23/2022							
15	95	=	=							
16	96	=	=							
17	97	=	=							
18	98	=	=							
19	99	=	=							
20	100	=	=							

cloud × / Everything ADVANCED SEA

My Favorites

SAVED RECORDS | SEARCH HISTORY

5 search queries

1. ↪ Any field contains cloud
   AND Any field contains risk or threat or vulnerability
   AND Any field contains attack vector AND Any field contains analysis
   AND Any field contains path or route
   SCOPE: Everything / Search everything
   11/23/2022

2. ↪ Any field contains cloud
   AND Any field contains risk or threat or vulnerability
   AND Any field contains attack vector AND Any field contains analysis
   SCOPE: Everything / Search everything
   11/23/2022

3. ↪ Any field contains cloud
   AND Any field contains risk or threat or vulnerability
   AND Any field contains attack vector
   SCOPE: Everything / Search everything
   11/23/2022

4. ↪ Any field contains cloud
   AND Any field contains risk or threat or vulnerability
   SCOPE: Everything / Search everything
   11/23/2022

5. cloud
   SCOPE: label / Search everything
   11/23/2022

safety trapping method, device and computer device

DOCUMENT ID	DATE PUBLISHED
CN-114584359-A	2022-06-03

INVENTOR INFORMATION

NAME	CITY	STATE	ZIP CODE	COUNTRY
FANG, Yong-cheng	N/A	N/A	N/A	N/A
ZHAO, Chong-hao	N/A	N/A	N/A	N/A
LIU, MAO-LIN	N/A	N/A	N/A	N/A
GONG, Liang-hua	N/A	N/A	N/A	N/A

DATE FILED
2022-02-24

CPC CURRENT

TYPE	CPC	DATE
CPCI	H04L49/70	2013-01-01
CPCI	H04L63/1491	2013-01-01
CPCI	H04L63/1466	2013-01-01

Abstract

The invention is suitable for the technical field of network safety, providing a safety for trapping, comprising: generating a plurality of categories of the first simulation device and the first virtual service, the first simulation device and the first virtual service is according to the device parameter of the entity device and the application scene of the entity device, generated by the simulator; generating a shadow device of the entity device, each shadow device corresponding to a plurality of virtual communication address; when the virtual communication address receives the flow data packet from the attack device, if the shadow device corresponding to the virtual communication address is high interaction, the flow data packet is forwarded to the corresponding first simulation device and/or the first virtual service through the shadow device. In the method, the simulation device is generated by the simulator corresponding to the entity device, so as to realize the deployment of the industrial internet spoofing trapping system. Because there is no need to generate a plurality of containers, so it can save a lot of server resource, so as to reduce the cost of deploying industrial internet spoofing trapping system.

Description

INVENTION-TITLE
Technical Field
The invention belongs to the technical field of network safety, especially relates to a safety and device for trapping and trapping.
Background

With the upgrading of industrial transformation, the industrial Internet is widely used. However, the industrial Internet has the problem of network information safety application, such information leakage, network attack, there is a safety of using the loophole and so on. At present, the active defence can be carried out by the industrial internet spoofing and trapping system, and the threat intelligence is obtained.
However, the existing industrial internet spoofing trapping system needs to occupy a large amount of server resource deployment, deployment cost is high.
Contents of the Invention
The embodiment of the invention claims safety trapping method and device, which can solve the problem that the industrial internet spoofing trapping system needs to occupy a large amount of server resource when deployed, and the deployment cost is high.
In a first aspect, the embodiment of the invention claims a safety for capturing, comprising: generating a plurality of categories of the first simulation device and the first virtual service, the first simulation device and the first virtual service is according to the device parameter of the entity device and the application scene of the entity device, generated by the simulator corresponding to the entity device; generating a shadow device of at least one entity device, the shadow device is a mapping of the entity device, each shadow device corresponding to a plurality of virtual communication address; when the virtual communication address receives the flow data packet from the attack device, if the shadow device corresponding to the virtual communication address is high interaction, the flow data packet is forwarded to the corresponding first simulation device and/or the first virtual service through the shadow device.
In one possible implementation of the first aspect, generating shadow device of at least one entity device, comprising: in the idle IP address space, batch generating virtual IP address of a plurality of different physical addresses; generating the corresponding relation between the shadow device and the first simulation device and the first simulation service according to the mapping relationship between the shadow device and the entity device; according to the corresponding relation, associating a plurality of virtual IP address associated with each first simulation device and the first simulation service.
In one possible implementation of the first aspect, the traffic data packet comprises an application layer message, the method further comprises: when the virtual communication address receives the attack application layer message from the attack device, the shadow device corresponding to the virtual communication address forwarding the attack application layer message to the corresponding first simulation device and the first virtual service; when the first simulation device and the first virtual service receive the first message sent by the attack device, responding to the corresponding second message; correspondingly storing the contents of the first message and the second message in the data table one by one; when the first simulation device and the first virtual service receives the target application layer message of the target attack device, matching the content of the target message with the content of the first message in the data table, if the matching is successful, then sending the second message corresponding to the successfully matched first message to the attack device.
In one possible implementation of the first aspect, the method further comprises: generating the attack image of the target attack device according to the history attack flow data log; collecting the attack behaviour of the attack image of the target attack device, generating the alarm behaviour log corresponding to the target attack device.
In one possible implementation of the first aspect, the method further comprises: according to history attack flow data log generating attack image of the target attack device, comprising: obtaining the historical flow data record generated by each attacking device; obtaining key field from the attack flow data record, wherein the key field comprises attack device communication address, attack device fingerprint and the user login of the specific software if at least one of the key field in the flow data sent by any attack device and the target attack device is the same, then determining the attack device and the target attack device are the same attack device.
In one possible implementation of the first aspect, the method further comprises: sending the alarm behaviour log corresponding to the attack device to the target server, so that the target server sends the alarm behaviour log to the client.
In one possible implementation of the first aspect, the method further comprises: orderly counting the attack behaviour total number in the attack flow data in each preset time period and the total number of the alarm attack behaviour; generating an alarm attack behaviour tendency chart according to the total number of

attack behaviour and the total number of alarm attack behaviour in each preset time period, and displaying the alarm attack behaviour tendency chart in the monitoring large screen.
In a second aspect, the embodiment of the invention claims a device safety trapping, comprising: service simulation module, used for generating a plurality of categories of the first simulation device and the first virtual service, the first simulation device and the first virtual service is according to the device parameter of the entity device and the application scene of the entity device, generated by the simulator corresponding to the entity device; a device generating module, a shadow device generating at least one entity device, the shadow device is a mapping of the entity device, each shadow device corresponding to a plurality of virtual communication address; a flow forwarding module, used for when the virtual communication address receives the flow data packet from the attack device, if the virtual communication address corresponding to the shadow device is high interaction, the flow data packet is forwarded to the corresponding first simulation device and/or the first virtual service through the shadow device.
the third aspect, the embodiment of the invention claims a terminal device, the terminal device comprises a memory and a processor, the memory stores a computer program, the processor executes the computer program to realize the following steps: Including: generating a plurality of categories of the first simulation device and the first virtual service, the first simulation device and the first virtual service is according to the device parameter of the entity device and the application scene of the entity device, generated by the simulator corresponding to the entity device; generating a shadow device of at least one entity device, the shadow device is a mapping of the entity device, each shadow device corresponding to a plurality of virtual communication address; when the virtual communication address receives the flow data packet from the attack device, if the shadow device corresponding to the virtual communication address is high interaction, the flow data packet is forwarded to the corresponding first simulation device and/or the first virtual service through the shadow device.
the fourth aspect, the embodiment of the invention claims a computer readable storage medium computer, readable storage medium is stored with a computer program, the computer program is executed by a processor to implement the following steps: Including: generating a plurality of categories of the first simulation device and the first virtual service, the first simulation device and the first virtual service is according to the device parameter of the entity device and the application scene of the entity device, generated by the simulator corresponding to the entity device; generating a shadow device of at least one entity device, the shadow device is a mapping of the entity device, each shadow device corresponding to a plurality of virtual communication address; when the virtual communication address receives the flow data packet from the attack device, if the virtual communication address corresponding to the flow data packet is forwarded to the corresponding first simulation device.
the fifth aspect, the embodiment of the invention claims a computer program product, when the computer program product is run on the terminal device, the terminal device executes any one safety the first aspect of the trapping method.
It can be understood that the second aspect to the fifth aspect of the beneficial effect can be described in the first aspect of the related description, which will not be repeated here.
The embodiment of the invention compared with the existing technology has the beneficial effects as follows: The safety, device, computer device, storage medium computer program product, in the server, generating the first simulation device and the first virtual service by the simulator corresponding to the entity device, and configuring the virtual communication address for each first simulation device, when the virtual communication address receives the flow data packet from the attack device, and forwarding it to the corresponding first simulation device and/or the first virtual service. In the method, through the simulator corresponding to the entity device to generate the simulation device, so as to realize the deployment of industrial internet spoofing trapping system. Because there is no need to generate a plurality of containers, so it can save a lot of server resource, so as to reduce the cost of deploying industrial internet spoofing trapping system.
Description of pictures
In order to more clearly explain the technical solution of the embodiment of the present application, the embodiments or the prior art described in the description of the accompanying drawings is briefly described, it is obvious, the following description of the accompanying drawings is only some embodiments of the invention, for the common technician in the field, under the premise of not paying creative labour, it can

obtain other drawings according to the drawings.
FIG. 1 is an embodiment of the invention provides safety environment diagram of the trapping method;
FIG. 2 is a flow diagram of the device generating step provided by one embodiment of the present application;
FIG. 3 is a flow diagram of the safety for trapping according to one embodiment of the invention;
FIG. 4 is another embodiment of the invention claims safety capture method of the flow diagram;
FIG. 5 is a schematic diagram of an application scenario provided by an embodiment of the present application;
FIG. 6 is a structure schematic safety of the trapping device provided by the embodiment of the invention;
FIG. 7 is a structure schematic diagram of the terminal device provided by the embodiment of the invention.
Specific implementation examples
In order to make the purpose of the present application, technical solution and advantages are more clearly understood, the following combine the accompanying drawings and examples, for further detailed explanation of this application. It should be understood that the specific embodiment described herein is only for explaining the application, and not for defining the application.
FIG. 1 is an embodiment of the invention claims safety environment diagram of the trapping method.
safety of the invention claims a trapping method, which can be applied to the application environment shown in FIG. 1. wherein, the attack device terminal 102 communicates with the server 104 through the network. The data storage system may store the data to be processed by the server 104. the data storage system can be integrated on the server 104, also can be placed on the cloud or other network server. server 104 generates a plurality of categories of the first simulation device and the first virtual service, the first simulation device and the first virtual service is according to the device parameter of the entity device and the application scene of the entity device, generated by the simulator corresponding to the entity device; generating a shadow device of at least one entity device, the shadow device is a mapping of the entity device, each shadow device corresponding to a plurality of virtual communication address; when the virtual communication address received from the attack device 102 sent by the flow data packet, if the virtual communication address corresponding to the shadow device is high interaction, then the flow data packet is forwarded to the corresponding first simulation device and/or the first virtual service through the shadow device. wherein the attack device terminal can be but not limited to mobile phone, the tablet computer can also be internet of things device and portable wearable device, the internet of things device can be a intelligent, a intelligent TV, a intelligent air conditioner, intelligent vehicle-mounted device and so on. The portable wearable device can be intelligent watch, a smart wristband, a head-mounted device and so on. The server 104 may be implemented using an independent server or a server cluster consisting of a plurality of servers.
FIG. 2 is a flow diagram of the device generating step provided by one embodiment of the present application.
In one embodiment, as shown in FIG. 2, there is provided a safety for capturing, using the method applied to FIG. 1 in the server 104 as an example for explanation, comprising the following steps:
S202, generating a plurality of categories of the first simulation device and the first virtual service, wherein the first simulation device and the first virtual service is the device parameter of the entity device and the application scene of the entity device, generated by the simulator corresponding to the entity device.
Specifically, the server can realize the simulation of the entity industrial control device by entity industrial control device issued by the simulator corresponding to the entity industrial control device, and then the corresponding firmware is loaded into the industrial control device simulation model, so as to realize the simulation of the entity industrial control device to obtain the first simulation device. For example, when the server needs to simulate the industrial control device is Siemens river graph industrial control device, the Siemens manufacturer issued by the Siemens simulation software simulated with the firmware, so as to obtain the first simulation device corresponding to the Siemens river graph industrial control device.
the first virtual service can be defined according to the protocol, through the protocol simulation of the code level, so as to realize the network service simulation. For example, when the simulation ssh (Secure Shell safety shell protocol), the service end receives the request sent by the client end, handshake for three times, establishing tcp (Transmission ControlProtocol, transmission control protocol). then the two parties send the version number and algorithm supported by itself, negotiating to determine the version and algorithm. then

the client generates a pair of keys according to the data sent by the service end, and sends the public key to the server. when the client receives the identity of the first verification server, then storing it in the local folder, then using the temporary public key sent by the server to generate the shared key, and then encrypting the signature, comparing the result with the calculated hash value, when the data is verification problem, so as to finish the simulation of ssh protocol.

when the simulation simulation of the entity industrial control device and simulation simulation of the network protocol, according to the production process of each industry, office network, the scene of the production network and the simulation parameter of each entity device are used for configuring each simulation device and the virtual service combination into different types of industrial internet spoofing trapping system. It should be noted that the industrial internet spoofing trapping system also can be called honey net system, and the honey net system is composed of honey field and honey net, wherein the honey field is composed of a plurality of simulation device and a plurality of virtual service, honey net is covered on the network of the honey field.

S204, generating at least one shadow device of the entity device, the shadow device is a mapping of the entity device, each shadow device corresponding to a plurality of virtual communication address.

wherein the shadow device is a virtual image on the existing operating system, it has the same function system is completely the same. after entering the shadow system, all operations are virtual, will not generate any influence to the real system. When the first simulation device is operated on the existing operation system, the shadow device is the mapping of the first simulation device on the virtual operation system. Because the flow rate is introduced into the honey field, actually the attacker attacks the virtual device in the honey field, service, but for the attacker, the attack is the shadow device through the communication address virtual virtual technology, service, device.

In other embodiments, a shadow device may correspond to one entity device, also can correspond to a plurality of entity device. when a shadow device corresponding to a plurality of entity device, shadow device obtains a plurality of virtual communication address, and the plurality of virtual communication address are distributed to each entity device.

S206, when the virtual communication address receives the flow data packet from the attack device, if the virtual communication address corresponding to the shadow device is high interaction, then the flow data packet is forwarded to the corresponding first simulation device and/or the first virtual service through the shadow device.

Specifically, when the virtual communication address set by the shadow device in the virtual operation system receives the data flow packet from the attack device, if the virtual communication address corresponding to the shadow device is high interaction, the shadow device introduces the received flow data packet into the first simulation device or the first simulation service corresponding to the virtual communication address by means of the net address conversion. wherein the shadow device can be configured to high interaction, also can be configured to low interaction. The shadow device with low interaction only can imitate a small amount of internet protocol and network service, and the shadow device with high interaction can simulate the industrial control device, also can simulate most internet protocol and network service.

It should be noted that when the shadow device is low interaction, it only needs to transmit the data flow sent by the attack device to the first simulation device or the first simulation service through specific port on the shadow device.

The safety for capturing, generating a first simulation device and a first virtual service by the simulator corresponding to the entity device in the server, and configuring a virtual address for each first simulation device, when the virtual communication address receives the flow data packet from the attack device, and forwarding it to the corresponding first simulation device and/or the first virtual service, the method generates the simulation device by the simulator corresponding to the entity device, it can save a lot of server resource, so as to reduce the cost of deploying the honey net.

In one embodiment, the server generates a shadow device of at least one entity device, comprising: in the idle IP address space, batch generating virtual IP address of a plurality of different physical addresses; generating the corresponding relation between the shadow device and the first simulation device and the first simulation service according to the mapping relationship between the shadow device and the entity device; according to the corresponding relation, associating a plurality of virtual IP address associated with

each first simulation device and the first simulation service.
Specifically, the virtual communication address of the idle virtual communication address in the virtual operation system selects virtual communication address of different physical addresses, because the shadow device is a mapping of the entity device, the mapping relation of the shadow device and the entity device can be one-to-one mapping relation, also can be multiple-to-one mapping relationship. the first simulation device is generated by the simulator corresponding to the entity device, so the first simulation device and the entity device are one-to-one corresponding relation, according to the corresponding relation between the first simulation device and the entity device and the mapping relation of the shadow device and the entity device to obtain the corresponding relation between the shadow device and the first simulation device, then the plurality of virtual addresses according to the respective first simulation device associated with each other.
In this embodiment, the virtual communication address is configured for the first simulation device and the first simulation service, so that the attacker sends attack flow packet to the virtual communication address, and the attack flow packet is forwarded to the first simulation device and the first simulation service through the shadow device, so as to collect the attack information passive discovery attack behaviour, resisting external network attack.
FIG. 3 is a flow diagram of the safety for trapping according to one embodiment of the invention.
In one embodiment, the traffic data packet comprises an application layer message, referring to FIG. 3, the trapping safety further comprises:
S302, when the virtual communication address receives the first message from the attack device, the shadow device corresponding to the virtual communication address forwarding the first message to the corresponding first simulation device and the first virtual service.
wherein the first message refers to the application layer message in the flow data packet sent by the attack device to the virtual communication address. Specifically, the virtual communication address in the virtual operation system receives the first message from the attack device, the shadow device can through the network address conversion technology to the public address of the first message into local area network address, transmitting the first message to the first simulation device or the first simulation service according to the local area network address.
S304, when the first simulation device and the first virtual service receives the first message sent by the attack device response corresponding to the second message.
Because the first virtual service comprises a simulation network protocol, the first simulation device and/or the first virtual service receives the first message sent by the attack device, according to the simulation network protocol response to the first message, the response message is sent to the attack device.
S306, the content of the first message and the second message are stored in the data table one by one.
Specifically, for part using the unencrypted protocol of the industrial control device, capturing the flow of the client and the service end, extracting the interactive application layer message, to " key name: The form of the key value is in the data table in the database.
Illustratively, the application layer message sent by the attack device is AAAA, the application layer message replied by the service end is BBBB, the database is stored as AAAA: BBBB; after using the flow learning result, if the server detects the application layer message sent by the attack device is AAAA, the service end returns the application layer message BBBB corresponding to the AAAA.
S308, when the first simulation device and the first virtual service receives the target application layer message of the target attack device, the content of the target message is matched with the content of the first message in the data table, if the matching is successful, then the second message corresponding to the successfully matched first message is sent to the attack device.
Illustratively, when the first simulation device and the first virtual service when using flow learning, function the server detects the application layer message sent by the attack device is AAAA, searching the same content with the AAAA message from the data table stored in the server, if the content is the first message of AAAA in the data table, then obtaining the second message corresponding to the first message, and forwarding the second message to the attack device through the shadow device corresponding to the virtual communication address.
In this embodiment, according to the flow data packet between the attack device and the server, extracting the application layer message in the interactive data packet, when detecting the same application layer

message of the first message stored in the database, the server can directly return the second message corresponding to the first message in the database, The invention improves the response efficiency of the flow data packet sent by the attack device.
FIG. 4 is another embodiment of the invention claims safety capture method of the flow diagram.
In one embodiment, the trapping safety further comprises:
S402, according to the history attack flow data log generating attack image of the target attack device. wherein the attack portrait refers to the intrusion attack device for identity authentication.
Specifically, the server obtains the historical flow data record generated by each attack device; then obtaining the key field from the history attack flow data record, wherein the key field comprises attack device address, attack device fingerprint and the user name of the specific software of the attack device login Illustratively, specific software Ai-cui, QQ music, China's professional IT community, and so on. if at least one of the key field in the flow data sent by any attack device and the target attack device is the same, then determining the attack device and the target attack device are the same attack device.
S404, collecting the attack behaviour of the attack image of the target attack device, generating the alarm behaviour log corresponding to the target attack device.
Specifically, the server analyzes the attack behaviour belonging to the same device from the historical flow data packet, analyzing all the attack behaviour, and extracting the alarm behaviour, combining the alarm behaviour of the execution time, the executor and generating the alarm behaviour log corresponding to the attack device.
In this embodiment, the server to attack the communication address of the device, device fingerprint, social account number three latitude associated with the external attacker image, and recording the change condition of the communication address of the attack device, solving the manual modification of the communication address, dynamic host configuration protocol lease expiration of communication address change condition caused by subsequent attack device is difficult to trace.
In one embodiment, the method further comprises: sending the alarm behaviour log corresponding to the attack device to the target server, so that the target server sends the alarm behaviour log to the client. wherein the target server is used for storing and analyzing various alarm behaviour log generated in the interaction process of the server and the attack device, interactive flow log and other data.
honey pot generating server sends the alarm action log generated by interaction with the attack device to the target server, the target server sends the alarm action log to the client, so as to alarm the manager to timely manually intervene the attack device.
In one embodiment, the method further comprises: orderly counting the attack behaviour total number in the attack flow data in each preset time period and the total number of the alarm attack behaviour; generating an alarm attack behaviour tendency chart according to the total number of attack behaviour and the total number of alarm attack behaviour in each preset time period, and displaying the alarm attack behaviour tendency chart in the monitoring large screen.
Specifically, the server the attack behaviour total number in each preset time period and alarm attack behaviour total number, and counting the ratio of alarm attack behaviour number and attack behaviour number of each preset time period, and the alarm attack behaviour number and number ratio of each preset time period to generate trend graph, and displaying the trend graph in the monitoring large screen; In addition, the large screen can also display early warning level, total number of assets (total number of internal attackers), asset high risk number (internal attacker high risk), shadow device number, attack network topology, high risk behaviour rolling playing, attack shadow device type statistics, attack behaviour type statistics, high risk alarm trend/total alarm trend
In this embodiment, the high risk behaviour rolling play, attack shadow device type statistics, attack behaviour type statistics, high risk alarm trend/total alarm trend and real-time display on the monitoring large screen, can make the manager timely monitors the attack device high risk behaviour and alarm behaviour.
FIG. 5 is an application scene schematic diagram provided by one embodiment of the present application.
In one embodiment, as shown in FIG. 5, the invention claims an industrial internet fraud trapping system applied to industrial internet platform.
The system comprises a data display layer, a data storage layer, a data processing layer and a data collecting layer. wherein the data collecting layer comprises an analogue service module and a shadow device module, the simulation service comprises industrial control service, traditional IT service, network

device and each industrial control device; The shadow device comprises a virtual communication address response module and a port response module. the data processing layer comprises the original flow log, original attack behaviour log, original attacker information data merging, data analysis, collecting external threat intelligence and multiple reverse tracing method. the data storage layer is used for the alarm behaviour log, alarm event log, request flow log, interactive flow log and other data to store. the data display layer comprises a real-time monitoring module and a platform display module.

in the trapping safety, firstly simulating the corresponding first simulation device in the data collecting layer through the industrial control device simulator, then simulating the protocol of the code level to obtain the first virtual service, the server through the trapping management module to the first simulation device and the first virtual service according to field deployment condition to generate honey field . For example: if the safety is deployed under the working condition, the server generates the host computer, printer and switch and other analog network device, then generating various simulation network protocol needed by the office network through the server.

the server sends the shadow device management strategy to the shadow device module, the shadow device module in the virtual system idle IP address space, batch generating a plurality of virtual IP address of different physical address; generating the corresponding relation between the shadow device and the first simulation device and the first simulation service according to the mapping relationship between the shadow device and the entity device; then according to the corresponding relation, associating a plurality of virtual IP address associated with each first simulation device and the first simulation service, so that the first simulation device and the first simulation service are covered by the network. forming a low interaction service simulation for the shadow device without high interaction according to the policy response specific port and the service open state, introducing the attack flow into the simulation service module by means of network address conversion for the shadow device configured with high interaction, forming a high interaction service simulation. when the attack device attacks the target network in the process of touching the virtual shadow device, the shadow device the attack flow data packet is forwarded to the corresponding first simulation device and/or the first virtual service.

at the same time, the data collecting and analyzing module will collect and record the attack flow data packet of the recording attack device interaction with the shadow device, and analyzing and analyzing the flow, generating the original flow log and sending to the data processing module. when the attack device starts the network scan and network attack, the data analysis module of the data collecting and analyzing module will detect the corresponding attack, such as scanning, password blasting, vulnerability utilization, industrial control attack, and other network attack behaviour, and generating the original alarm behaviour log to the data processing module. when the attacker finishes the attack and obtains the simulation service authority, the data collecting and analyzing module will collect the host action of the attacker to the simulation service, comprising a sample transmitted to the simulation service, generating the original alarm behaviour log to send to the data processing module.

after the data processing module receives the original flow log, generating the attack image of the target attack device according to the history attack flow data log. Specifically, the data processing module takes the physical communication address as root base, the external network uses communication address, device fingerprint, social account number three latitude extracting the attacker information and inquiring by domain name, domain name reverse analysis, domain name location, external threat intelligence, port scanning and other reverse traceability path to the attacker to trace the source, completing the attacker information then the data processing module divides the attack flow into interaction flow and request flow according to the type of the shadow device, then supplementing shadow device according to type of attack flow, source communication address geographic location and session address and so information, and storing all the information into the database. the other data processing module is further used for collecting the attack behaviour of the attack image of the target attack device, generating the alarm behaviour log corresponding to the target attack device. then taking the attacker and the attacked shadow device as the condition, merging the alarm behaviour log, extracting the label, attack times and so on information generate alarm event log. the data processing module is used for analyzing the sample extracted from the original alarm behaviour log in the sandbox, and calling the threat information to inquire the related information and finally the sample data.

at last, collecting the industrial internet spoofing trapping system in the data display layer, the analyzed data

is visually displayed. the real-time monitoring module mainly displays the real-time situation of the industrial internet spoofing trapping system deployment area, comprising a pre-warning level, the total number of attacks in the first month, the number of the high-risk behavior in the first month, the total number of the assets (total number of the attackers), the high risk number of the assets (high risk number of the inner network attackers), the number of the shadow devices, and the attack network topology, high risk behaviour rolling playing, attack shadow device type statistics, attack behaviour type statistics, high risk alarm trend/total alarm trend statistics data. and the data display layer also can send the alarm behaviour log timing to the target server, so as to prompt the manager to pay attention to the high-risk behaviour of the attack device.

The system is a safety trapping product suitable for industrial internet. different from the traditional honey net system is only used for collecting the attack information passive discovery attack behaviour, resisting external network attack, industrial internet fraud trapping system has active defense capability, can by simulating a plurality of industrial control device to construct virtual shadow device, building virtual honey net, trapping the attack behaviour of the industrial control network, and realizing the detection of each kind of attack behaviour, recognition and early warning ability. At the same time, through reverse engineering and large data analysis technology, for the attack of the industrial control network, quickly locating the suspicious behaviour, recognition network attack, timely implementing the safety protection, omni-directional threat early warning.

It should be understood that, although each of the steps involved in the above embodiments is displayed in turn according to the indication of the arrows, these steps are not necessarily performed sequentially in the order indicated by the arrows. Unless otherwise specified herein, the execution of these steps is not strictly limited in order, and these steps can be performed in other orders. Furthermore, at least a portion of the steps involved in the above embodiments may include a plurality of steps or stages, these steps or stages are not necessarily performed at the same time, but can be executed at different times, the execution order of these steps or stages is not necessarily performed in turn, and it can be executed alternately or alternately with other steps or at least a part of the step or stage in other steps.

FIG. 6 is a structure schematic safety of the trapping device provided by the embodiment of the invention. In one embodiment, as shown in FIG. 6, providing a safety device 600, comprising: The service simulation module 602, the device generation module 604 and the traffic forwarding module 606, wherein:

service simulation module 602, for generating a plurality of categories of the first simulation device and the first virtual service, the first simulation device and the first virtual service is the device parameter of the entity device and the application scene of the entity device, generated by the simulator corresponding to the entity device.

The device generating module 604 generates at least one shadow device of the entity device, the shadow device is a mapping of the entity device, and each shadow device corresponds to a plurality of virtual communication addresses.

flow forwarding module 606, for when the virtual communication address received from the attack device of the flow data packet, if the virtual communication address corresponding to the shadow device is high interaction, then the flow data packet is forwarded to the corresponding first simulation device and/or the first virtual service through the shadow device.

In one embodiment, the device generating module 604 comprises: an address generating sub-module, for in the idle IP address space, batch generating a plurality of virtual IP address of different physical address; corresponding relation sub-module, for according to the mapping relationship between the shadow device and the entity device to generate the corresponding relation between the shadow device and the first simulation device and the first simulation service; associated sub-module, for according to the corresponding relation, the plurality of virtual IP address associated with each first simulation device and the first simulation service.

In one embodiment, the device further comprises: a forwarding sub-module, for when the virtual communication address receives the attack application layer message from the attack device, the shadow device corresponding to the virtual communication address of the attack application layer message is forwarded to the corresponding first simulation device and the first virtual service; a second message generating sub-module, for when the first simulation device and the first virtual service receiving the first message sent by the attack device response corresponding to the second message; a storage sub-module,

for correspondingly storing the content of the first message and the second message in the data table one by one; a flow learning sub-module, for when the first simulation device and the first virtual service receives the target application layer message of the target attack device, the content of the target message is matched with the content of the first message in the data table, if the matching is successful, then sending the second message corresponding to the successfully matched first message to the attack device.
In one embodiment, the device further comprises: attack picture sub-module, used for according to the history attack flow data log generating attack picture of the target attack device; a behaviour sun sub-module, used for collecting the attack behaviour of the attack image of the target attack device, generating the alarm behaviour log corresponding to the target attack device.
In one embodiment, the attack image sub-module is further used for obtaining the historical flow data record generated by each attack device; obtaining key field from the attack flow data record, wherein the key field comprises attack device communication address, attack device fingerprint and the user login of the specific software if at least one of the key field in the flow data sent by any attack device and the target attack device is the same, then determining the attack device and the target attack device are the same attack device.
In one embodiment, the device is further used for sending the alarm behaviour log corresponding to the attack device to the target server, so that the target server sends the alarm behaviour log to the client.
In one embodiment, the device is further used for orderly counting the attack behaviour total number in the attack flow data in each preset time period and the total number of alarm attack behaviour; generating an alarm attack behaviour tendency chart according to the total number of attack behaviour and the total number of alarm attack behaviour in each preset time period, and displaying the alarm attack behaviour tendency chart in the monitoring large screen.
Each module in the robot teaching device can be all or partially implemented by software, hardware and combinations thereof. the each module can be embedded in the hardware form or independent from the processor in the computer device, also can be stored in the memory of the computer device in software form, so as to facilitate the processor to perform the operation corresponding to each module.
FIG. 7 is a structure schematic diagram of the terminal device provided by the embodiment of the invention.
In one embodiment, there is provided a computer device, the computer device can be a server, the internal structure diagram can be shown in FIG. 7. The computer device comprises a processor connected by a system bus, a memory and a network interface. wherein the processor of the computer device is used for providing calculation and control capability. The memory of the computer device includes a non-volatile storage medium and an internal memory. The non-volatile storage medium is stored with an operating system, a computer program and a database. The internal memory provides an environment for operating systems and computer programs in a non-volatile storage medium The database of the computer device is used for storing path track data. The network interface of the computer device is used for communicating with the external terminal through the network connection The computer program is executed by a processor to implement a robot teaching method.
It will be understood by those skilled in the art, the structure shown in FIG. 7, is only a block diagram of the partial structure associated with the present application scheme, and does not form the application scheme applied to the computer device of the definition, the specific computer device may include more or less than shown in the figure of the component, or a combination of certain components, or a different component arrangement.
In one embodiment, there is provided a computer device, comprising a memory and a processor, the memory is stored with a computer program, the processor executes the computer program to realize the safety one of the embodiments provided in the trapping method.
In one embodiment, there is provided a computer readable storage medium which is stored with a computer program, a computer program when executed by a processor to realize the safety provided in any one embodiment of the trapping method.
In one embodiment, there is provided a computer program product, comprising a computer program, when the computer program is executed by a processor to implement the method provided in any one safety of the capture method.
It should be noted that, the present application relates to the user (information but not limited to user equipment information user personal information and so on) and data (including but not limited to data for analysis, stored data, data display), are authorized by the user or fully authorized by each information of the

data.
Those skilled in the art can understand all or part of the flow in the above embodiment method, which can be finished by computer program instruction related hardware to finish, the computer program can be stored in a non-volatile computer storage medium, the computer program when executing, The flow of the embodiments of the above methods may be included. wherein any reference of the memory, database or other medium used in each embodiment provided by the present application, may include at least one of non-volatile and volatile memory. The non-volatile memory may include a Read Only Memory (Read-OnlyMemory, ROM), a tape, a floppy disk, a flash memory, an optical memory, a high density embedded non-volatile memory, a resistive random access memory (ReRAM), Magnetoresistive Random Access Memory (MRAM), Ferroelectric Random Access Memory (FRAM), phase change memory (Phase Change Memory, PCM), graphene memory. The volatile memory may include a Random Access Memory (Random Access Memory, RAM) or an external cache. As explanation and not limitation, RAM can be a plurality of forms, such as static random access memory (Static Random Access Memory, SRAM) or dynamic random access memory (Dynamic RandomAccess Memory, DRAM) and so on. The database involved in each embodiment of the present application may include at least one of a type database and a non-type database. The non-type database may include a blockchain-based distributed database, and the like, not limited to this. The processor involved in the embodiments provided by the present application may be a general purpose processor, a central processor, a graphics processor, a digital signal processor, a programmable logic device, a data processing logic based on quantum computation, and so on, not limited to this.
each technical feature of the above embodiments can be any combination, to make the description is simple, not all possible combinations of all technical features in the above embodiments are described, however, as long as the combination of these technical characteristics does not exist contradiction, should be considered as the scope of the specification.
The above embodiments only express several embodiments of the present application, the description is more specific and detailed, but not to be understood as the limit of the patent application range of the invention. It should be noted that, for the common technician in the field, without departing from the conception of the invention, it also can make several deformation and improvement, these belong to the protection range of the invention. Therefore, the protection scope of the present application should be based on the appended claims.

Claims

1. A safety for trapping and trapping, wherein the safety comprises the following steps: generating a plurality of categories of the first simulation device and the first virtual service, the first simulation device and the first virtual service is according to the device parameter of the entity device and the application scene of the entity device, generated by the simulator corresponding to the entity device; generating at least one shadow device of the entity device, the shadow device is a mapping of the entity device, each of the shadow device corresponding to a plurality of virtual communication address; when the virtual communication address receives the flow data packet from the attack device, if the shadow device corresponding to the virtual communication address is high interaction, then forwarding the flow data packet to the corresponding first simulation device and/or the first virtual service through the shadow device.
2. The safety according to claim 1, wherein the step of generating at least one shadow device of the entity device comprises: generating at least one shadow device of the entity device according to the image information of the object device, in the idle IP address space, batch generating virtual IP address of a plurality of different physical addresses; generating the corresponding relation between the shadow device and the first simulation device and the first simulation service according to the mapping relationship between the shadow device and the entity device; according to the corresponding relation, associating a plurality of the virtual IP address associated with each of the first simulation device and the first simulation service.
3. The safety for trapping according to claim 1, wherein the flow data packet comprises an application layer message, the method further comprises the following steps: when the virtual communication address receives the attack application layer message from the attack device, the shadow device corresponding to

the virtual communication address transfers the attack application layer message to the corresponding first simulation device and the first virtual service; when the first analogue device and the first virtual service receive the first message sent by the attack device, responding to the corresponding second message; correspondingly storing the contents of the first message and the second message in the data table one by one; when the first simulation device and the first virtual service receives the target application layer message of the target attack device, matching the content of the target message with the content of the first message in the data table, if the matching is successful, then sending the second message corresponding to the first message matched successfully to the attack device.

4. The safety for trapping according to claim 1, wherein the method further comprises: generating the attack image of the target attack device according to the history attack flow data log; collecting the attack behaviour of the attack image of the target attack device, generating the alarm behaviour log corresponding to the target attack device.

5. The safety according to claim 4, wherein the step of generating the attack image of the target attack device according to the history attack flow data log comprises: obtaining the historical flow data record generated by each said attacking device; obtaining key field from the attack flow data record, wherein the key field comprises attack device communication address, attack device fingerprint and user login of the specific software if at least one of the key field in the flow data sent by any attack device and the target attack device is the same, then determining that the attack device and the target attack device are the same attack device.

6. The safety for trapping according to claim 1, wherein the method further comprises: sending the alarm behaviour log corresponding to the attack device to the target server, so that the target server sends the alarm behaviour log to the client.

7. The safety for trapping according to claim 1, wherein the method further comprises: orderly counting the attack behaviour total number in the attack flow data in each preset time period and the total number of the alarm attack behaviour; generating an alarm attack behaviour tendency chart according to the total number of attack behaviour and the total number of alarm attack behaviour in each preset time period, and displaying the alarm attack behaviour trend graph in the monitoring large screen.

8. An safety device, wherein the device comprises: service simulation module, used for generating a plurality of categories of the first simulation device and the first virtual service, the first simulation device and the first virtual service according to the device parameter of the entity device and the application scene of the entity device, generated by the simulator corresponding to the entity device; device generating module, generating at least one shadow device of the entity device, the shadow device is a mapping of the entity device, each of the shadow device corresponding to a plurality of virtual communication address; a flow forwarding module, for when the virtual communication address receives the flow data packet from the attack device, if the shadow device corresponding to the virtual communication address is high interaction, then forwarding the flow data packet to the corresponding first simulation device and/or the first virtual service through the shadow device.

9. A terminal device, comprising a memory, a processor and a computer program stored in the memory and capable of running on the processor, wherein the processor executes the method according to any one of claims 1 to 1 to 7 when executing the computer program.

10. A computer readable storage medium wherein the computer readable storage medium stored with a computer program, wherein the computer program is executed by a processor to implement the method according to any one of claims 1 to 7.

Copyright © 2022 Clarivate Analytics. All rights reserved. Republication or redistribution of Clarivate Analytics content, including by framing or similar means, is prohibited without the prior written consent of Clarivate Analytics. Clarivate and its logo are trademarks of the Clarivate Analytics group.

(19)国家知识产权局

(12)发明专利申请

(10)申请公布号 CN 114584359 A
(43)申请公布日 2022.06.03

(21)申请号 202210178181.1

(22)申请日 2022.02.24

(71)申请人 烽台科技(北京)有限公司
地址 100000 北京市海淀区北坞村路甲25号静心园I座一层101室

(72)发明人 方永成 赵重浩 刘茂林 龚亮华

(74)专利代理机构 深圳中一联合知识产权代理有限公司 44414
专利代理师 张延薇

(51)Int.Cl.
H04L 9/40(2022.01)
H04L 49/00(2022.01)

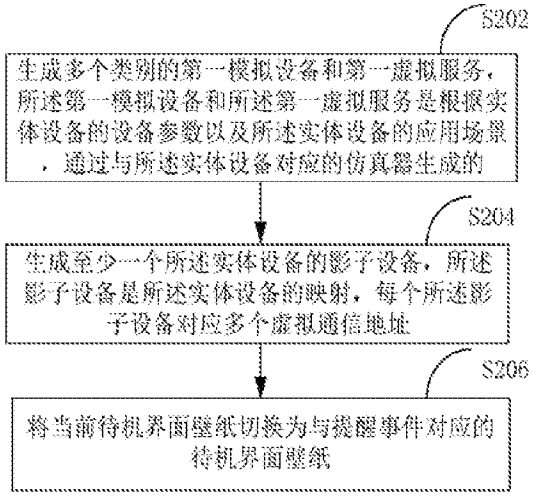
权利要求书2页 说明书10页 附图5页

(54)发明名称

安全诱捕方法、装置和计算机设备

(57)摘要

本申请适用于网络安全技术领域，提供了一种安全诱捕方法，包括：生成多个类别的第一模拟设备和第一虚拟服务，第一模拟设备和第一虚拟服务是根据实体设备的设备参数以及实体设备的应用场景，通过与实体设备对应的仿真器生成的；生成实体设备的影子设备，每个影子设备对应多个虚拟通信地址；当所述虚拟通信地址接收到来自攻击设备的流量数据包时，若虚拟通信地址对应的影子设备为高交互，则通过影子设备将流量数据包转发至对应的第一模拟设备和/或第一虚拟服务。该方法中，通过与实体设备对应的仿真器来生成模拟设备，进而实现工业互联网欺骗诱捕系统的部署。由于无需生成多个容器，所以可以节省大量的服务器资源，从而减少部署工业互联网欺骗诱捕系统的成本。



CN 114584359 A

1.一种安全诱捕方法,其特征在于,所述安全诱捕方法,包括:

生成多个类别的第一模拟设备和第一虚拟服务,所述第一模拟设备和所述第一虚拟服务是根据实体设备的设备参数以及所述实体设备的应用场景,通过与所述实体设备对应的仿真器生成的;

生成至少一个所述实体设备的影子设备,所述影子设备是所述实体设备的映射,每个所述影子设备对应多个虚拟通信地址;

当所述虚拟通信地址接收到来自攻击设备的流量数据包时,若所述虚拟通信地址对应的影子设备为高交互,则通过所述影子设备将所述流量数据包转发至对应的第一模拟设备和/或第一虚拟服务。

2.如权利要求1所述的安全诱捕方法,其特征在于,所述生成至少一个所述实体设备的影子设备,包括:

在空闲的IP地址空间中,批量生成多个不同物理地址的虚拟IP地址;

根据影子设备与所述实体设备之间的映射关系生成影子设备与所述第一模拟设备以及第一模拟服务之间的对应关系;

根据所述对应关系,将多个所述虚拟IP地址关联上各个所述第一模拟设备以及第一模拟服务。

3.如权利要求1所述的安全诱捕方法,其特征在于,所述流量数据包中包含应用层报文,所述方法还包括:

当所述虚拟通信地址接收到来自攻击设备的攻击应用层报文,与所述虚拟通信地址相对应的影子设备将所述攻击应用层报文转发至对应的第一模拟设备以及第一虚拟服务;

当所述第一模拟设备以及第一虚拟服务接收到所述攻击设备发送的第一报文后响应相对应的第二报文;

将所述第一报文与所述第二报文的内容一一对应地存储在数据表中;

当所述第一模拟设备以及第一虚拟服务接收到目标攻击设备的目标应用层报文时,将所述目标报文的内容与所述数据表中的第一报文的内容相匹配,若匹配成功,则将匹配成功的第一报文相对应的第二报文发送至所述攻击设备。

4.如权利要求1所述的安全诱捕方法,其特征在于,所述方法还包括:

根据所述历史攻击流量数据日志生成目标攻击设备的攻击画像;

收集所述目标攻击设备的攻击画像的攻击行为,生成与所述目标攻击设备对应的告警行为日志。

5.如权利要求4所述的安全诱捕方法,其特征在于,所述根据所述历史攻击流量数据日志生成目标攻击设备的攻击画像,包括:

获取各个所述攻击设备产生的历史流量数据记录;

从所述攻击流量数据记录中获取关键字段,其中所述关键字段包括攻击设备地通信地址、攻击设备指纹以及攻击设备上登录的特定软件的用户名;

若任一攻击设备与目标攻击设备发送的流量数据中的关键字段中至少一项相同,则认定所述攻击设备与所述目标攻击设备为同一攻击设备。

6.如权利要求1所述的安全诱捕方法,其特征在于,所述方法还包括:

将与所述攻击设备对应的告警行为日志发送目标服务器,以使目标服务器将所述告警

行为日志发送至客户端。

7.如权利要求1所述的安全诱捕方法，其特征在于，所述方法还包括：

依次统计每一预设时间段内的攻击流量数据中的攻击行为总数以及告警攻击行为总数；

根据每一预设时间段内的攻击行为总数以及告警攻击行为总数生成告警攻击行为趋势图，并将所述告警攻击行为趋势图显示在监视大屏中。

8.一种安全诱捕装置，其特征在于，所述装置包括：

服务模拟模块，用于生成多个类别的第一模拟设备和第一虚拟服务，所述第一模拟设备和所述第一虚拟服务是根据实体设备的设备参数以及所述实体设备的应用场景，通过与所述实体设备对应的仿真器生成的；

设备生成模块，生成至少一个所述实体设备的影子设备，所述影子设备是所述实体设备的映射，每个所述影子设备对应多个虚拟通信地址；

流量转发模块，用于当所述虚拟通信地址接收到来自攻击设备的流量数据包时，若所述虚拟通信地址对应的影子设备为高交互，则通过所述影子设备将所述流量数据包转发至对应的第一模拟设备和/或第一虚拟服务。

9.一种终端设备，包括存储器、处理器以及存储在所述存储器中并可在所述处理器上运行的计算机程序，其特征在于，所述处理器执行所述计算机程序时实现如权利要求1至7任一项所述的方法。

10.一种计算机可读存储介质，所述计算机可读存储介质存储有计算机程序，其特征在于，所述计算机程序被处理器执行时实现如权利要求1至7任一项所述的方法。

安全诱捕方法、装置和计算机设备

技术领域

[0001] 本申请属于网络安全技术领域，尤其涉及一种安全诱捕方法及装置。

背景技术

[0002] 随着工业转型升级，工业互联网被广泛应用。但是，工业互联网在应用时存在网络信息安全方面的问题，如信息泄露、遭受网络攻击、存在可被利用的安全漏洞等问题。目前可以通过工业互联网欺骗诱捕系统进行主动防御，获取威胁情报。

[0003] 但是，现有的工业互联网欺骗诱捕系统在部署时需要占用大量的服务器资源，部署成本较高。

发明内容

[0004] 本申请实施例提供了安全诱捕方法及装置，可以解决工业互联网欺骗诱捕系统在部署时需要占用大量的服务器资源，部署成本较高的问题。

[0005] 第一方面，本申请实施例提供了一种安全诱捕方法，包括：生成多个类别的第一模拟设备和第一虚拟服务，第一模拟设备和第一虚拟服务是根据实体设备的设备参数以及实体设备的应用场景，通过与实体设备对应的仿真器生成的；生成至少一个实体设备的影子设备，影子设备是实体设备的映射，每个影子设备对应多个虚拟通信地址；当虚拟通信地址接收到来自攻击设备的流量数据包时，若虚拟通信地址对应的影子设备为高交互，则通过影子设备将流量数据包转发至对应的第一模拟设备和/或第一虚拟服务。

[0006] 在第一方面的一种可能的实现方式中，生成至少一个实体设备的影子设备，包括：在空闲的IP地址空间中，批量生成多个不同物理地址的虚拟IP地址；根据影子设备与实体设备之间的映射关系生成影子设备与第一模拟设备以及第一模拟服务之间的对应关系；根据对应关系，将多个虚拟IP地址关联上各个第一模拟设备以及第一模拟服务。

[0007] 在第一方面的一种可能的实现方式中，流量数据包中包含应用层报文，方法还包括：当虚拟通信地址接收到来自攻击设备的攻击应用层报文，与虚拟通信地址相对应的影子设备将攻击应用层报文转发至对应的第一模拟设备以及第一虚拟服务；当第一模拟设备以及第一虚拟服务接收到攻击设备发送的第一报文后响应相对应的第二报文；将第一报文与第二报文的内容一一对应地存储在数据表中；当第一模拟设备以及第一虚拟服务接收到目标攻击设备的目标应用层报文时，将目标报文的内容与数据表中的第一报文的内容相匹配，若匹配成功，则将匹配成功的第一报文相对应的第二报文发送至攻击设备。

[0008] 在第一方面的一种可能的实现方式中，该方法还包括：根据历史攻击流量数据日志生成目标攻击设备的攻击画像；收集目标攻击设备的攻击画像的攻击行为，生成与目标攻击设备对应的告警行为日志。

[0009] 在第一方面的一种可能的实现方式中，该方法还包括：根据历史攻击流量数据日志生成目标攻击设备的攻击画像，包括：获取各个攻击设备产生的历史流量数据记录；从攻击流量数据记录中获取关键字段，其中关键字段包括攻击设备地通信地址、攻击设备指纹

以及攻击设备上登录的特定软件的用户名；若任一攻击设备与目标攻击设备发送的流量数据中的关键字段中至少一项相同，则认定攻击设备与目标攻击设备为同一攻击设备。

[0010] 在第一方面的一种可能的实现方式中，该方法还包括：将与攻击设备对应的告警行为日志发送目标服务器，以使目标服务器将告警行为日志发送至客户端。

[0011] 在第一方面的一种可能的实现方式中，该方法还包括：依次统计每一预设时间段内的攻击流量数据中的攻击行为总数以及告警攻击行为总数；根据每一预设时间段内的攻击行为总数以及告警攻击行为总数生成告警攻击行为趋势图，并将告警攻击行为趋势图显示在监视大屏中。

[0012] 第二方面，本申请实施例提供了一种安全诱捕装置，包括：服务模拟模块，用于生成多个类别的第一模拟设备和第一虚拟服务，第一模拟设备和第一虚拟服务是根据实体设备的设备参数以及实体设备的应用场景，通过与实体设备对应的仿真器生成的；设备生成模块，生成至少一个实体设备的影子设备，影子设备是实体设备的映射，每个影子设备对应多个虚拟通信地址；流量转发模块，用于当虚拟通信地址接收到来自攻击设备的流量数据包时，若虚拟通信地址对应的影子设备为高交互，则通过影子设备将流量数据包转发至对应的第一模拟设备和/或第一虚拟服务。

[0013] 第三方面，本申请实施例提供了一种终端设备，终端设备包括存储器和处理器，存储器存储有计算机程序，处理器执行计算机程序时实现以下步骤：包括：生成多个类别的第一模拟设备和第一虚拟服务，第一模拟设备和第一虚拟服务是根据实体设备的设备参数以及实体设备的应用场景，通过与实体设备对应的仿真器生成的；生成至少一个实体设备的影子设备，影子设备是实体设备的映射，每个影子设备对应多个虚拟通信地址；当虚拟通信地址接收到来自攻击设备的流量数据包时，若虚拟通信地址对应的影子设备为高交互，则通过影子设备将流量数据包转发至对应的第一模拟设备和/或第一虚拟服务。

[0014] 第四方面，本申请实施例提供了一种计算机可读存储介质，计算机可读存储介质，其上存储有计算机程序，计算机程序被处理器执行时实现以下步骤：包括：生成多个类别的第一模拟设备和第一虚拟服务，第一模拟设备和第一虚拟服务是根据实体设备的设备参数以及实体设备的应用场景，通过与实体设备对应的仿真器生成的；生成至少一个实体设备的影子设备，影子设备是实体设备的映射，每个影子设备对应多个虚拟通信地址；当虚拟通信地址接收到来自攻击设备的流量数据包时，若虚拟通信地址对应的流量数据包转发至相对应的第一模拟设备。

[0015] 第五方面，本申请实施例提供了一种计算机程序产品，当计算机程序产品在终端设备上运行时，使得终端设备执行上述第一方面中任一项安全诱捕方法。

[0016] 可以理解的是，上述第二方面至第五方面的有益效果可以参见上述第一方面中的相关描述，在此不再赘述。

[0017] 本申请实施例与现有技术相比存在的有益效果是：上述安全诱捕方法、装置、计算机设备、存储介质和计算机程序产品，在服务器中通过与实体设备对应的仿真器生成第一模拟设备以及第一虚拟服务，并为各个第一模拟设备配置虚拟通信地址，当该虚拟通信地址接收到来自攻击设备的流量数据包时，并将其转发至发生对应的第一模拟设备和/或第一虚拟服务。在本方法中，通过与实体设备对应的仿真器来生成模拟设备，进而实现工业互联网欺骗诱捕系统的部署。由于无需生成多个容器，所以可以节省大量的服务器资源，从而

减少部署工业互联网欺骗诱捕系统的成本。

附图说明

[0018] 为了更清楚地说明本申请实施例中的技术方案，下面将对实施例或现有技术描述中所需要使用的附图作简单地介绍，显而易见地，下面描述中的附图仅仅是本申请的一些实施例，对于本领域普通技术人员来讲，在不付出创造性劳动的前提下，还可以根据这些附图获得其他的附图。

[0019] 图1是本申请一实施例提供的安全诱捕方法的应用环境图；

[0020] 图2是本申请一实施例提供的设备生成步骤的流程示意图；

[0021] 图3是本申请一实施例提供的安全诱捕方法的流程示意图；

[0022] 图4是本申请另一实施例提供的安全诱捕方法的流程示意图；

[0023] 图5是本申请一实施例提供的应用场景示意图；

[0024] 图6是本申请实施例提供的安全诱捕装置的结构示意图；

[0025] 图7是本申请实施例提供的终端设备的结构示意图。

具体实施方式

[0026] 为了使本申请的目的、技术方案及优点更加清楚明白，以下结合附图及实施例，对本申请进行进一步详细说明。应当理解，此处描述的具体实施例仅仅用以解释本申请，并不用于限定本申请。

[0027] 图1是本申请一实施例提供的安全诱捕方法的应用环境图。

[0028] 本申请实施例提供的安全诱捕方法，可以应用于如图1所示的应用环境中。其中，攻击设备终端102通过网络与服务器104进行通信。数据存储系统可以存储服务器104需要处理的数据。数据存储系统可以集成在服务器104上，也可以放在云上或其他网络服务器上。服务器104生成多个类别的第一模拟设备和第一虚拟服务，第一模拟设备和第一虚拟服务是根据实体设备的设备参数以及实体设备的应用场景，通过与实体设备对应的仿真器生成的；生成至少一个实体设备的影子设备，影子设备是实体设备的映射，每个影子设备对应多个虚拟通信地址；当虚拟通信地址接收到来自攻击设备102发送的流量数据包时，若虚拟通信地址对应的影子设备为高交互，则通过影子设备将流量数据包转发至对应的第一模拟设备和/或第一虚拟服务。其中，攻击设备终端可以但不限于手机、平板电脑还可以是物联网设备和便携式可穿戴设备，物联网设备可为智能音箱、智能电视、智能空调、智能车载设备等。便携式可穿戴设备可为智能手表、智能手环、头戴设备等。服务器104可以用独立的服务器或者是多个服务器组成的服务器集群来实现。

[0029] 图2是本申请一实施例提供的设备生成步骤的流程示意图。

[0030] 在一个实施例中，如图2所示，提供了一种安全诱捕方法，以该方法应用于图1中的服务器104为例进行说明，包括以下步骤：

[0031] S202，生成多个类别的第一模拟设备和第一虚拟服务，其中，第一模拟设备和第一虚拟服务是根据实体设备的设备参数以及实体设备的应用场景，通过与实体设备对应的仿真器生成的。

[0032] 具体地，服务器可以通过实体工控设备厂家发布的与实体工控设备相对应的仿真

器来实现对实体工控设备的仿真，再将对应的固件加载进入工控设备仿真模型中，从而实现对实体工控设备的仿真得到第一模拟设备。例如，当服务器需要模拟的工控设备为西门子河图工控设备时，将西门子厂家发布的西门子仿真软件与固件一起进行仿真，从而得到与西门子河图工控设备相对应的第一模拟设备。

[0033] 第一虚拟服务则可以是根据协议的定义，通过代码层级的协议仿真，进而实现对网络服务仿真得到的。例如，当仿真ssh(Secure Shell，安全外壳协议)时，服务端接收到来自客户端发送过来的请求后，先进行三次握手，建立tcp(Transmission Control Protocol，传输控制协议)连接。然后双方发送自己所支持的版本号以及算法，协商确定版本及算法。接着客户端根据服务端发送过来的数据生成一对密钥，将公钥发送给服务器。当客户端接收到后会通过先验证服务器的身份，然后将其保存到本地文件夹中，再使用服务器发送过来的临时公钥生成共享密钥，再加密签名，将其结果与自己所计算的哈希值进行比较，验证数据时没有问题的，从而完成对ssh协议的仿真。

[0034] 当完成对实体工控设备的仿真模拟以及网络协议的仿真模拟后，根据各个行业的生产工艺、办公网、生产网等的场景以及各个实体设备的仿真参数来将各个模拟设备以及虚拟服务组合配置成各种不同类型的工业互联网欺骗诱捕系统。需要说明的是，工业互联网欺骗诱捕系统也可以叫蜜网系统，而蜜网系统又是由蜜场与蜜网组成，其中蜜场由多个模拟设备以及多种虚拟服务组成，蜜网是指覆盖在蜜场上的网络。

[0035] S204，生成至少一个实体设备的影子设备，影子设备是实体设备的映射，每个影子设备对应多个虚拟通信地址。

[0036] 其中，影子设备是可以现有操作系统上的虚拟影像，它具有真实系统完全一样的功能。进入影子系统后，所有操作都是虚拟的，不会对真正的系统产生任何影响。当在现有的操作系统上运行出第一模拟设备时，影子设备是在虚拟操作系统上第一模拟设备的映射。由于流量引入蜜场，实际上攻击者攻击的是蜜场里面的虚拟设备、服务，但是对于攻击者而言，他攻击的是影子设备通过ip虚拟技术虚拟出的通信地址、服务、设备。

[0037] 在其他实施例中，一个影子设备可以对应一个实体设备，也可以对应多个实体设备。当一个影子设备对应多个实体设备时，影子设备获取多个虚拟通信地址，并将多个虚拟通信地址依次分配给各个实体设备。

[0038] S206，当虚拟通信地址接收到来自攻击设备的流量数据包时，若虚拟通信地址对应的影子设备为高交互，则通过影子设备将流量数据包转发至对应的第一模拟设备和/或第一虚拟服务。

[0039] 具体地，当虚拟操作系统中影子设备设置的虚拟通信地址接收到来自攻击设备的数据流量包时，若此时虚拟通信地址对应的影子设备为高交互时，影子设备将接收到的流量数据包通过net地址转换的方式把攻击流量数据包引入与该虚拟通信地址对应的第一模拟设备或第一模拟服务。其中，影子设备可以配置成高交互的，也可以配置成低交互的。低交互的影子设备只能模仿少量的互联网协议和网络服务，而高交互的影子设备既可以模仿工控设备、也可以模拟大多数的互联网协议以及网络服务。

[0040] 需要说明的是，当该影子设备为低交互时，则只需通过影子设备上特定的端口将攻击设备发来的数据流量转发至第一模拟设备或者第一模拟服务。

[0041] 上述安全诱捕方法，在服务器中通过与实体设备对应的仿真器生成第一模拟设备

以及第一虚拟服务，并为各个第一模拟设备配置虚拟地址，当该虚拟通信地址接收到来自攻击设备的流量数据包时，并将其转发至发生对应的第一模拟设备和/或第一虚拟服务，该方法中通过与实体设备对应的仿真器来生成模拟设备，可以节省大量的服务器资源，从而减少部署蜜网的成本。

[0042] 在一个实施例中，服务器生成至少一个实体设备的影子设备，包括：在空闲的IP地址空间中，批量生成多个不同物理地址的虚拟IP地址；根据影子设备与实体设备之间的映射关系生成影子设备与第一模拟设备以及第一模拟服务之间的对应关系；根据对应关系，将多个虚拟IP地址关联上各个第一模拟设备以及第一模拟服务。

[0043] 具体地，从虚拟操作系统中的空闲虚拟通信地址批量选择出不同物理地址的虚拟通信地址，由于影子设备是实体设备的映射，影子设备与实体设备的映射关系可以是一对一的映射关系，也可以是多对一的映射关系。第一模拟设备是由与实体设备对应的仿真器生成的，因此第一模拟设备与实体设备之间是一对一的对应关系，根据第一模拟设备与实体设备之间的对应关系以及影子设备与实体设备的映射关系得出影子设备与第一模拟设备之间的对应关系，然后将多个虚拟地址根据该对应关联上各个第一模拟设备。

[0044] 本实施例中，通过为第一模拟设备以及第一模拟服务配置虚拟通信地址，从而引诱攻击者向虚拟通信地址发送攻击流量包，并通过影子设备将攻击流量包转发至第一模拟设备以及第一模拟服务，从而可以收集攻击信息、被动发现攻击行为、抵御外部网络攻击。

[0045] 图3是本申请一实施例提供的安全诱捕方法的流程示意图。

[0046] 在一实施例中，流量数据包中包含应用层报文，参考图3，该安全诱捕方法还包括：

[0047] S302，当虚拟通信地址接收到来自攻击设备的第一报文，与虚拟通信地址相对应的影子设备将第一报文转发至对应的第一模拟设备以及第一虚拟服务。

[0048] 其中，第一报文是指攻击设备发送给虚拟通信地址的流量数据包中的应用层报文。具体地，当虚拟操作系统中的虚拟通信地址接收到来自攻击设备的第一报文，影子设备可以通过网络地址转换技术将第一报文的公网地址转换成局域网地址，然后根据该局域网地址将第一报文转发至第一模拟设备或者第一模拟服务。

[0049] S304，当第一模拟设备以及第一虚拟服务接收到攻击设备发送的第一报文后响应相对应的第二报文。

[0050] 由于第一虚拟服务中包含有仿真网络协议，第一模拟设备和/或第一虚拟服务接收到攻击设备发送的第一报文后，会根据仿真网络协议响应第一报文，将响应得到报文发送给攻击设备。

[0051] S306，将第一报文与第二报文的内容一一对应地存储在数据表中。

[0052] 具体地，针对部分使用未加密协议的工控设备，抓取客户端和服务端的流量，提取出交互的应用层报文，以“键名:键值”的形式存在数据库中数据表中。

[0053] 示例性地，攻击设备发送的应用层报文为AAAA，服务端回复的应用层报文为BBBB，数据库中保存为AAAA:BBBB；当使用流量学习的结果后，如果服务器检测到攻击设备发送的应用层报文为AAAA，服务端就返回AAAA对应的应用层报文BBBB。

[0054] S308，当第一模拟设备以及第一虚拟服务接收到目标攻击设备的目标应用层报文时，将目标报文的内容与数据表中的第一报文的内容相匹配，若匹配成功，则将匹配成功的第一报文相对应的第二报文发送至攻击设备。

[0055] 示例性地，当第一模拟设备以及第一虚拟服务当使用流量学习功能后，如果服务器检测到攻击设备发送的应用层报文为AAAA，从服务器中的存储的数据表中查找与AAAA相同内容的报文，若在数据表中存在内容为AAAA的第一报文，则获取与该第一报文相对应的第二报文，并将第二报文通过与该虚拟通信地址相对应的影子设备转发至攻击设备。

[0056] 本实施例中，根据攻击设备与服务器之间的流量数据包，提取出交互数据包中的应用层报文，当检测到与数据库中存储的第一报文相同的应用层报文后，服务器可以直接返回数据库中与该第一报文对应的第二报文，提高了对对攻击设备发送的流量数据包的响应效率。

[0057] 图4是本申请另一实施例提供的安全诱捕方法的流程示意图。

[0058] 在一个实施例中，该安全诱捕方法还包括：

[0059] S402，根据历史攻击流量数据日志生成目标攻击设备的攻击画像。

[0060] 其中，攻击画像是指对入侵攻击设备进行身份认证。

[0061] 具体地，首先服务器获取各个攻击设备产生的历史流量数据记录；然后从历史攻击流量数据记录中获取关键字段，其中关键字段包括攻击设备地通信地址、攻击设备指纹以及攻击设备上登录的特定软件的用户名等。示例性的，特定软件包括爱奇艺、QQ音乐、中国专业IT社区等。若任一攻击设备与目标攻击设备发送的流量数据中的关键字段中至少一项相同，则认定该攻击设备与目标攻击设备为同一攻击设备。

[0062] S404，收集目标攻击设备的攻击画像的攻击行为，生成与目标攻击设备对应的告警行为日志。

[0063] 具体地，服务器从历史流量数据包中解析出属于同一设备的攻击行为，对所有攻击行为进行解析，并从中提取出告警行为，结合告警行为的实行时间、执行者等生成与该攻击设备对应的告警行为日志。

[0064] 本实施例中，服务器以攻击设备的通信地址、设备指纹、社交账号三个纬度对外网攻击者进行关联画像，并记录下攻击设备的通信地址的变化情况，解决针对手动修改通信地址、动态主机配置协议租约到期等通信地址变化的情况导致后续攻击设备追溯困难的问题。

[0065] 在一个实施例中，该方法还包括：将与攻击设备对应的告警行为日志发送目标服务器，以使目标服务器将告警行为日志发送至客户端。

[0066] 其中，目标服务器是用于存储以及解析服务器与攻击设备交互过程中产生的各种告警行为日志、交互流量日志等数据。

[0067] 蜜罐生成服务器将与攻击设备交互产生的告警行为日志发送个目标服务器，目标服务器将该告警行为日志发送给至客户端，以警示管理者能够及时对攻击设备进行人为干预。

[0068] 在一个实施例中，该方法还包括：依次统计每一预设时间段内的攻击流量数据中的攻击行为总数以及告警攻击行为总数；根据每一预设时间段内的攻击行为总数以及告警攻击行为总数生成告警攻击行为趋势图，并将告警攻击行为趋势图显示在监视大屏中。

[0069] 具体地，服务器将每一预设时间段内的攻击行为总数以及告警攻击行为总数，并统计出每一预设时间段的告警攻击行为数量与攻击行为数量之比，并将每一预设时间段的告警攻击行为数量以及数量比生成趋势图，并将该趋势图显示在监视大屏中；另外，大屏上

还可以展示预警等级，资产总数(内网攻击者总数)，资产高危数(内网攻击者高危数)，影子设备数，攻击网络拓扑、高危行为滚动播放、被攻击影子设备类型统计、攻击行为类型统计、高危告警趋势/总体告警趋势等统计数据。

[0070] 本实施例中，将高危行为滚动播放、被攻击影子设备类型统计、攻击行为类型统计、高危告警趋势/总体告警趋势等实时显示在监控大屏上，可以使管理者及时地监察到攻击设备地高危行为和告警行为。

[0071] 图5是本申请一实施例提供的应用场景示意图。

[0072] 在一实施例中，如图5所示，提出了一种应用于工业互联网平台的工业互联网欺骗诱捕系统。

[0073] 该系统中包括数据展示层、数据存储层、数据处理层和数据采集层。其中，数据采集层包括模拟服务模块和影子设备模块，模拟服务中包含工控服务、传统IT服务、网络设备以及各工控设备；影子设备中包括虚拟通信地址响应模块以及端口响应模块。数据处理层包括对原始流量日志、原始攻击行为日志、原始攻击者信息日志进行数据归并、数据分析、收集外网威胁情报等多项反向溯源方法。数据存储层用于对告警行为日志、告警事件日志、请求流量日志、交互流量日志等数据进行存储。数据展示层包括实时监控模块和平台展示模块。

[0074] 在该安全诱捕方法中首先在数据采集层中通过工控设备仿真器仿真出对应的第一模拟设备，然后对代码层级的协议进行仿真得到第一虚拟服务，服务器通过诱捕管理模块将第一模拟设备和第一虚拟服务根据现场部署情况生成蜜场。例如：若该安全诱捕方法是部署在办公工况下，则通过服务器生成主机、打印机以及交换机等模拟网络设备，再通过服务器生成办公网所需的各种仿真网络协议。

[0075] 服务器再将影子设备管理策略下发到影子设备模块，影子设备模块在虚拟系统中空闲的IP地址空间中，批量生成多个不同物理地址的虚拟IP地址；根据影子设备与实体设备之间的映射关系生成影子设备与第一模拟设备以及第一模拟服务之间的对应关系；然后根据该对应关系，将多个虚拟IP地址关联上各个第一模拟设备以及第一模拟服务，使得第一模拟设备以及第一模拟服务被网络所覆盖。对于没有配置高交互的影子设备根据策略响应特定的端口与服务开放状态形成低交互服务模拟，对于配置高交互的影子设备通过网络地址转换的方式把攻击流量引入模拟服务模块，形成高交互服务模拟。当攻击设备攻击目标网络的过程中触碰到虚拟的影子设备时，该影子设备将攻击流量数据包转发至对应的第一模拟设备和/或第一虚拟服务。

[0076] 同时，数据采集分析模块会全程采集记录攻击设备与影子设备交互的攻击流量数据包，并对流量进行解析和分析，生成原始流量日志发送给数据处理模块。当攻击设备发动网络扫描和网络攻击时，数据采集分析模块的数据分析模块会通过检测到对应的攻击，比如扫描、密码爆破、漏洞利用、工控攻击、等网络攻击行为，并生成原始告警行为日志发送给数据处理模块。当攻击者在攻击完成并取得模拟服务权限后，数据采集分析模块会采集攻击者对模拟服务做的主机行为，包括传输到模拟服务中样本，生成原始告警行为日志发送给数据处理模块。

[0077] 数据处理模块接收到原始流量日志后，会根据历史攻击流量数据日志生成目标攻击设备的攻击画像。具体地，数据处理模块以物理通信地址为根基，外网以通信地址、设备

指纹、社交账号三个纬度提取出攻击者信息，并且通过域名查询、域名反向解析、域名定位、外网威胁情报、端口扫描等多种反向溯源途径对攻击者进行溯源追踪，补全攻击者信息。然后数据处理模块根据影子设备的类型将攻击流量分为交互流量和请求流量，然后根据攻击流量的类型补全影子设备、源通信地址地理位置以及会话地址等信息，并将所有信息存入数据库。另外数据处理模块还用于收集目标攻击设备的攻击画像的攻击行为，生成与目标攻击设备对应的告警行为日志。再以攻击者和被攻击影子设备为条件，对告警行为日志进行归并，提取出标签，攻击次数等信息生成告警事件日志。数据处理模块对原始告警行为日志中提取出的样本放入沙箱进行分析，并调用威胁情报查询相关信息，最后样本数据。

[0078] 最后在数据展示层中对工业互联网欺骗诱捕系统收集、分析的数据进行可视化展示。实时监控模块主要展示工业互联网欺骗诱捕系统部署区域的实时态势，包括预警等级、近1月攻击行为总数，近1月高危行为数，资产总数(内网攻击者总数)，资产高危数(内网攻击者高危数)，影子设备数，攻击网络拓扑、高危行为滚动播放、被攻击影子设备类型统计、攻击行为类型统计、高危告警趋势/总体告警趋势等统计数据。并且数据展示层还可以将告警行为日志定时发送给目标服务器，以提示管理人员及时关注攻击设备的高危行为。

[0079] 本系统是一款适用于工业互联网的安全诱捕产品。不同于传统的蜜网系统仅用于收集攻击信息、被动发现攻击行为、抵御外部网络攻击，工业互联网欺骗诱捕系统具备主动防御能力，可以通过模拟多种工控设备来构建虚拟影子设备、搭建虚拟蜜网，诱捕对工控网络的攻击行为，并实现对各种攻击行为的检测、识别和预警能力。同时，通过逆向工程和大数据分析技术解构，针对工控网络的攻击，做到迅速定位可疑行为，精确识别网络攻击，及时实施安全防护，全方位的完成威胁预警。

[0080] 应该理解的是，虽然如上的各实施例所涉及的流程图中的各个步骤按照箭头的指示依次显示，但是这些步骤并不是必然按照箭头指示的顺序依次执行。除非本文中有明确的说明，这些步骤的执行并没有严格的顺序限制，这些步骤可以以其它的顺序执行。而且，如上的各实施例所涉及的流程图中的至少一部分步骤可以包括多个步骤或者多个阶段，这些步骤或者阶段并不必然是在同一时刻执行完成，而是可以在不同的时刻执行，这些步骤或者阶段的执行顺序也不必然是依次进行，而是可以与其它步骤或者其它步骤中的步骤或者阶段的至少一部分轮流或者交替地执行。

[0081] 图6是本申请实施例提供的安全诱捕装置的结构示意图。

[0082] 在一个实施例中，如图6所示，提供了一种安全诱捕装置600，包括：服务模拟模块602、设备生成模块604和流量转发模块606，其中：

[0083] 服务模拟模块602，用于生成多个类别的第一模拟设备和第一虚拟服务，第一模拟设备和第一虚拟服务是根据实体设备的设备参数以及实体设备的应用场景，通过与实体设备对应的仿真器生成的。

[0084] 设备生成模块604，生成至少一个实体设备的影子设备，影子设备是实体设备的映射，每个影子设备对应多个虚拟通信地址。

[0085] 流量转发模块606，用于当虚拟通信地址接收到来自攻击设备的流量数据包时，若虚拟通信地址对应的影子设备为高交互，则通过影子设备将流量数据包转发至对应的第一模拟设备和/或第一虚拟服务。

[0086] 在一个实施例中，设备生成模块604包括：地址生成子模块，用于在空闲的IP地址

空间中，批量生成多个不同物理地址的虚拟IP地址；对应关系子模块，用于根据影子设备与实体设备之间的映射关系生成影子设备与第一模拟设备以及第一模拟服务之间的对应关系；关联子模块，用于根据对应关系，将多个虚拟IP地址关联上各个第一模拟设备以及第一模拟服务。

[0087] 在一个实施例中，该装置还包括：转发子模块，用于当虚拟通信地址接收到来自攻击设备的攻击应用层报文，与虚拟通信地址相对应的影子设备将攻击应用层报文转发至对应的第一模拟设备以及第一虚拟服务；第二报文生成子模块，用于当第一模拟设备以及第一虚拟服务接收到攻击设备发送的第一报文后响应相对应的第二报文；存储子模块，用于将第一报文与第二报文的内容一一对应地存储在数据表中；流量学习子模块，用于当第一模拟设备以及第一虚拟服务接收到目标攻击设备的目标应用层报文时，将目标报文的内容与数据表中的第一报文的内容相匹配，若匹配成功，则将匹配成功的第一报文相对应的第二报文发送至攻击设备。

[0088] 在一个实施例中，该装置还包括：攻击画像子模块，用于根据历史攻击流量数据日志生成目标攻击设备的攻击画像；行为日志子模块，用于收集目标攻击设备的攻击画像的攻击行为，生成与目标攻击设备对应的告警行为日志。

[0089] 在一个实施例中，攻击画像子模块还用于获取各个攻击设备产生的历史流量数据记录；从攻击流量数据记录中获取关键字段，其中关键字段包括攻击设备地通信地址、攻击设备指纹以及攻击设备上登录的特定软件的用户名；若任一攻击设备与目标攻击设备发送的流量数据中的关键字段中至少一项相同，则认定攻击设备与目标攻击设备为同一攻击设备。

[0090] 在一个实施例中，装置还用于将与攻击设备对应的告警行为日志发送目标服务器，以使目标服务器将告警行为日志发送至客户端。

[0091] 在一个实施例中，装置还用于依次统计每一预设时间段内的攻击流量数据中的攻击行为总数以及告警攻击行为总数；根据每一预设时间段内的攻击行为总数以及告警攻击行为总数生成告警攻击行为趋势图，并将告警攻击行为趋势图显示在监视大屏中。

[0092] 上述机器人示教装置中的各个模块可全部或部分通过软件、硬件及其组合来实现。上述各模块可以硬件形式内嵌于或独立于计算机设备中的处理器中，也可以以软件形式存储于计算机设备中的存储器中，以便于处理器调用执行以上各个模块对应的操作。

[0093] 图7是本申请实施例提供的终端设备的结构示意图。

[0094] 在一个实施例中，提供了一种计算机设备，该计算机设备可以是服务器，其内部结构图可以如图7所示。该计算机设备包括通过系统总线连接的处理器、存储器和网络接口。其中，该计算机设备的处理器用于提供计算和控制能力。该计算机设备的存储器包括非易失性存储介质和内存储器。该非易失性存储介质存储有操作系统、计算机程序和数据库。该内存储器为非易失性存储介质中的操作系统和计算机程序的运行提供环境。该计算机设备的数据库用于存储路径轨迹数据。该计算机设备的网络接口用于与外部的终端通过网络连接通信。该计算机程序被处理器执行时以实现一种机器人示教方法。

[0095] 本领域技术人员可以理解，图7中示出的结构，仅仅是与本申请方案相关的部分结构的框图，并不构成对本申请方案所应用于其上的计算机设备的限定，具体的计算机设备可以包括比图中所示更多或更少的部件，或者组合某些部件，或者具有不同的部件布置。

[0096] 在一个实施例中，提供了一种计算机设备，包括存储器和处理器，存储器中存储有计算机程序，该处理器执行计算机程序时实现上述任一实施例中提供的安全诱捕方法。

[0097] 在一个实施例中，提供了一种计算机可读存储介质，其上存储有计算机程序，计算机程序被处理器执行时实现上述任一实施例中提供的安全诱捕方法。

[0098] 在一个实施例中，提供了一种计算机程序产品，包括计算机程序，该计算机程序被处理器执行时实现上述任一实施例中提供的安全诱捕方法。

[0099] 需要说明的是，本申请所涉及的用户信息(包括但不限于用户设备信息、用户个人信息等)和数据(包括但不限于用于分析的数据、存储的数据、展示的数据等)，均为经用户授权或者经过各方充分授权的信息和数据。

[0100] 本领域普通技术人员可以理解实现上述实施例方法中的全部或部分流程，是可以通过计算机程序来指令相关的硬件来完成，的计算机程序可存储于一非易失性计算机可读取存储介质中，该计算机程序在执行时，可包括如上述各方法的实施例的流程。其中，本申请所提供的各实施例中所使用的对存储器、数据库或其它介质的任何引用，均可包括非易失性和易失性存储器中的至少一种。非易失性存储器可包括只读存储器(Read-Only Memory，ROM)、磁带、软盘、闪存、光存储器、高密度嵌入式非易失性存储器、阻变存储器(ReRAM)、磁变存储器(Magnetoresistive Random Access Memory，MRAM)、铁电存储器(Ferroelectric Random Access Memory，FRAM)、相变存储器(Phase Change Memory，PCM)、石墨烯存储器等。易失性存储器可包括随机存取存储器(Random Access Memory，RAM)或外部高速缓冲存储器等。作为说明而非局限，RAM可以是多种形式，比如静态随机存取存储器(Static Random Access Memory，SRAM)或动态随机存取存储器(Dynamic Random Access Memory，DRAM)等。本申请所提供的各实施例中所涉及的数据库可包括关系型数据库和非关系型数据库中至少一种。非关系型数据库可包括基于区块链的分布式数据库等，不限于此。本申请所提供的各实施例中所涉及的处理器可为通用处理器、中央处理器、图形处理器、数字信号处理器、可编程逻辑器、基于量子计算的数据处理逻辑器等，不限于此。

[0101] 以上实施例的各技术特征可以进行任意的组合，为使描述简洁，未对上述实施例中的各个技术特征所有可能的组合都进行描述，然而，只要这些技术特征的组合不存在矛盾，都应当认为是本说明书记载的范围。

[0102] 以上实施例仅表达了本申请的几种实施方式，其描述较为具体和详细，但并不能因此而理解为对本申请专利范围的限制。应当指出的是，对于本领域的普通技术人员来说，在不脱离本申请构思的前提下，还可以做出若干变形和改进，这些都属于本申请的保护范围。因此，本申请的保护范围应以所附权利要求为准。

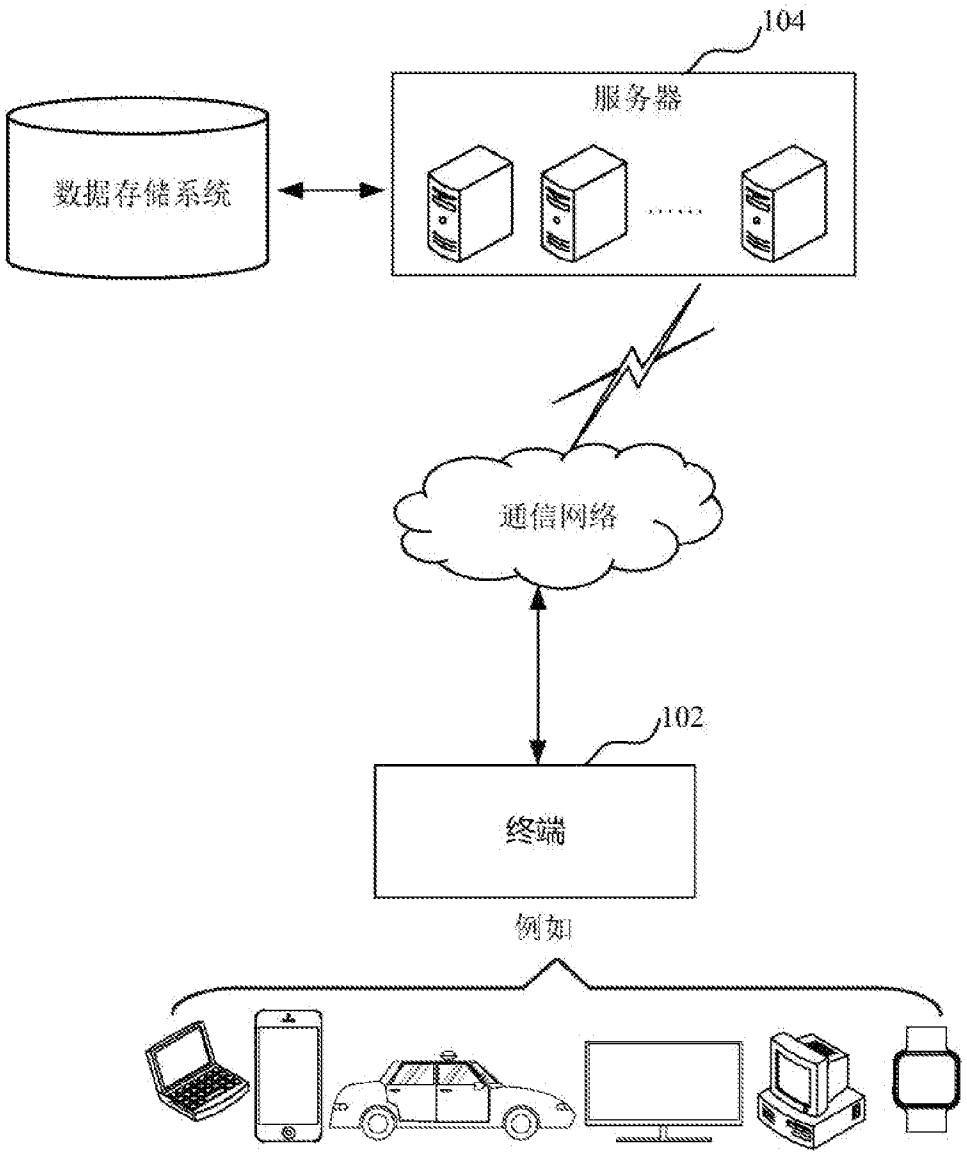


图1

360

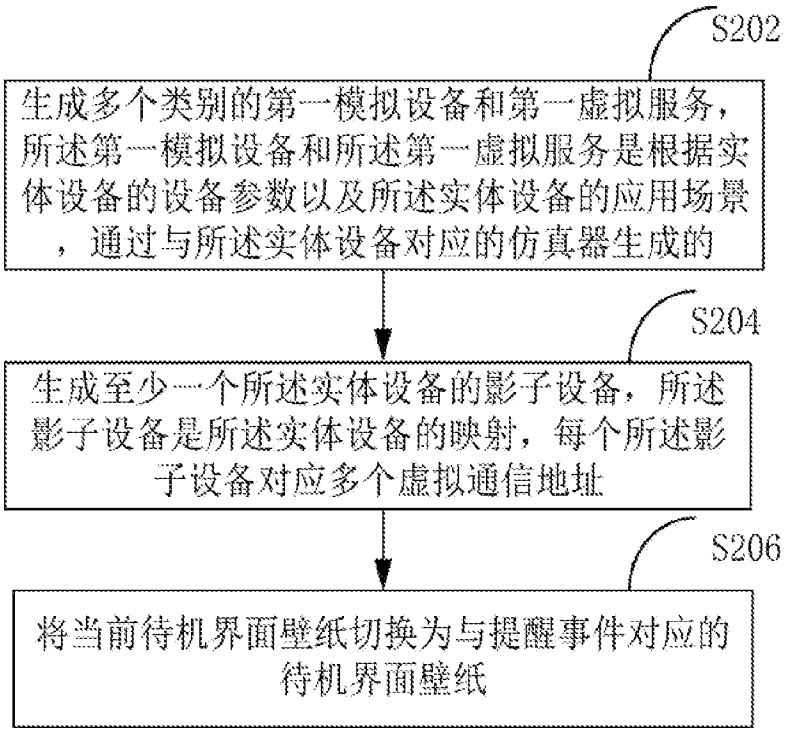

S202
生成多个类别的第一模拟设备和第一虚拟服务，所述第一模拟设备和所述第一虚拟服务是根据实体设备的设备参数以及所述实体设备的应用场景，通过与所述实体设备对应的仿真器生成的
S204
生成至少一个所述实体设备的影子设备，所述影子设备是所述实体设备的映射，每个所述影子设备对应多个虚拟通信地址
S206
将当前待机界面壁纸切换为与提醒事件对应的待机界面壁纸

图2

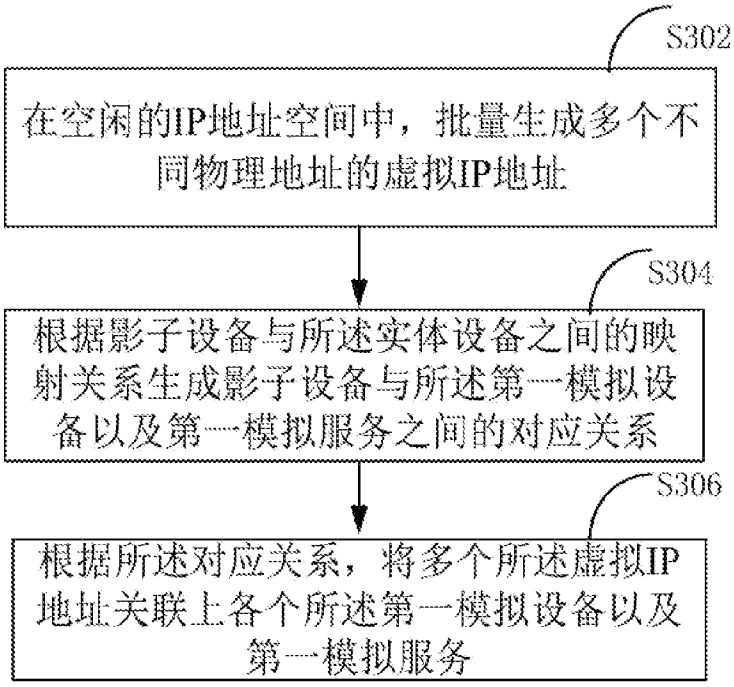

S302
在空闲的IP地址空间中，批量生成多个不同物理地址的虚拟IP地址
S304
根据影子设备与所述实体设备之间的映射关系生成影子设备与所述第一模拟设备以及第一模拟服务之间的对应关系
S306
根据所述对应关系，将多个所述虚拟IP地址关联上各个所述第一模拟设备以及第一模拟服务

图3

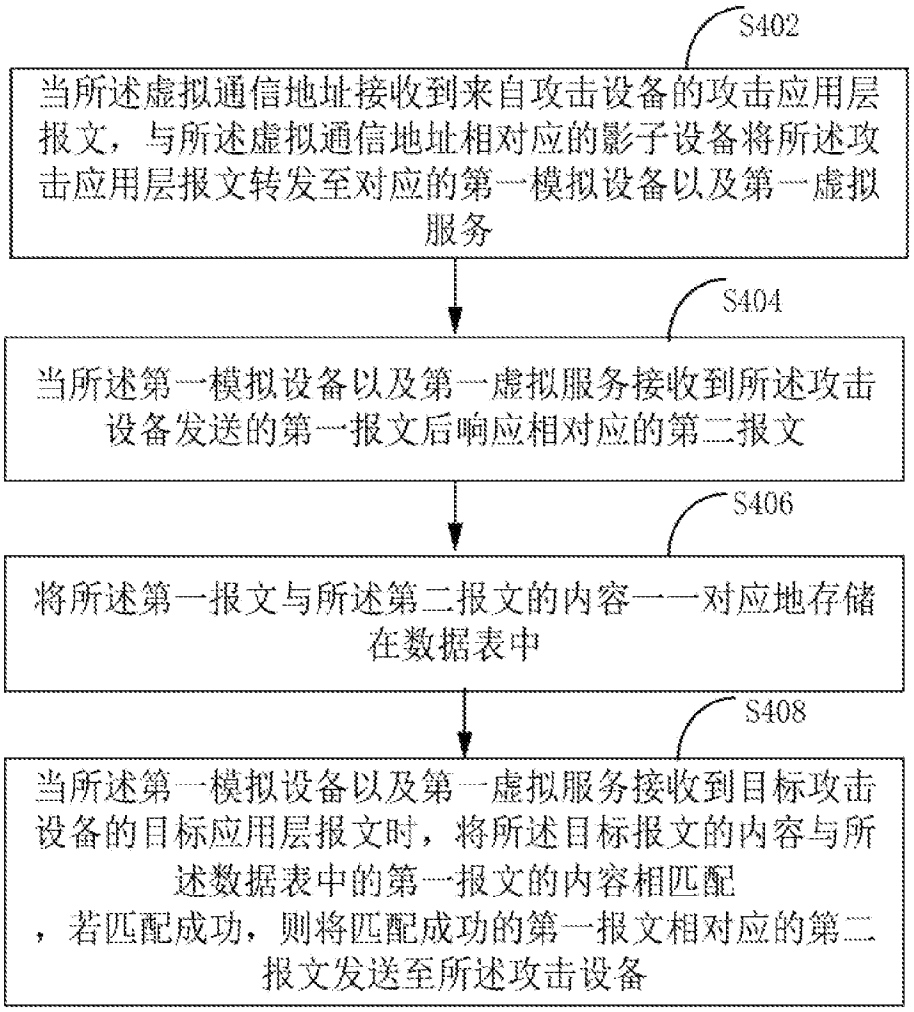

S402
当所述虚拟通信地址接收到来自攻击设备的攻击应用层报文，与所述虚拟通信地址相对应的影子设备将所述攻击应用层报文转发至对应的第一模拟设备以及第一虚拟服务
S404
当所述第一模拟设备以及第一虚拟服务接收到所述攻击设备发送的第一报文后响应相对应的第二报文
S406
将所述第一报文与所述第二报文的内容一一对应地存储在数据表中
S408
当所述第一模拟设备以及第一虚拟服务接收到目标攻击设备的目标应用层报文时，将所述目标报文的内容与所述数据表中的第一报文的内容相匹配
，若匹配成功，则将匹配成功的第一报文相对应的第二报文发送至所述攻击设备

图4

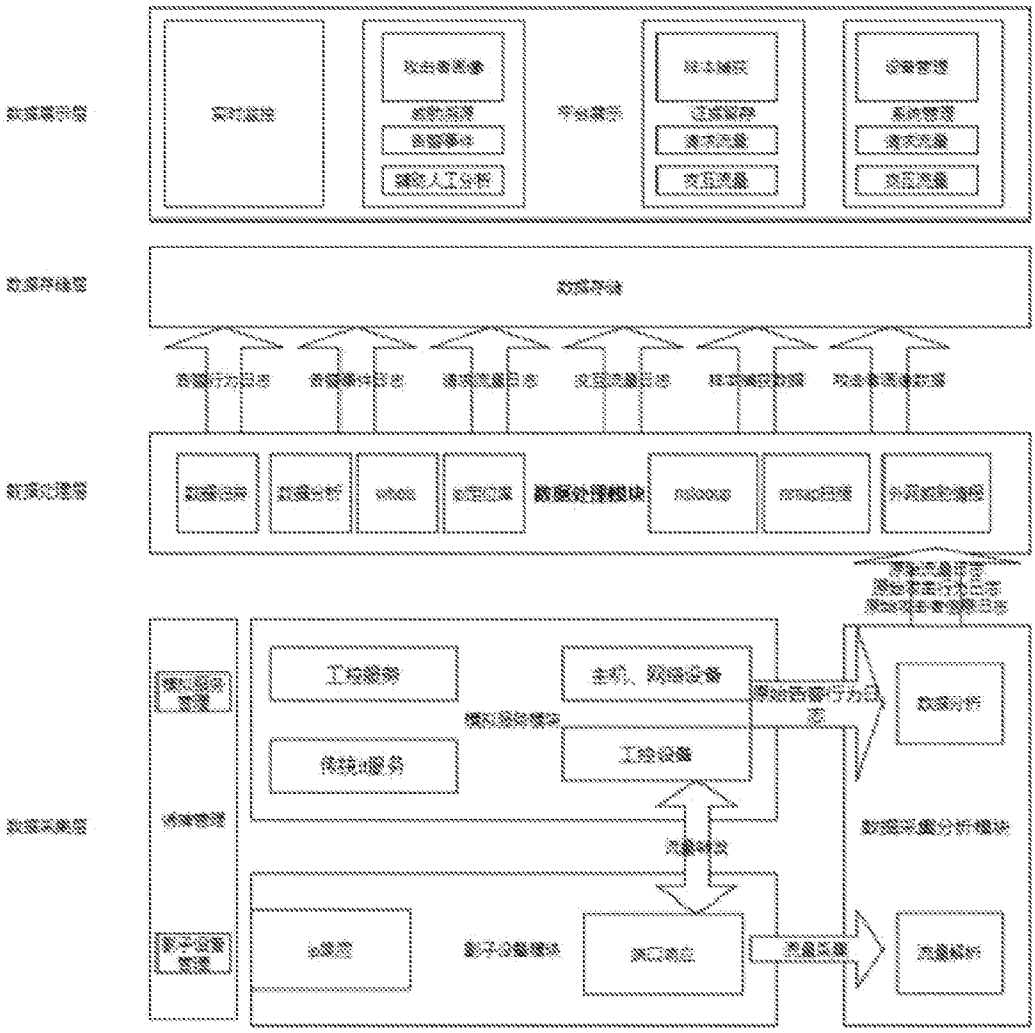


图5

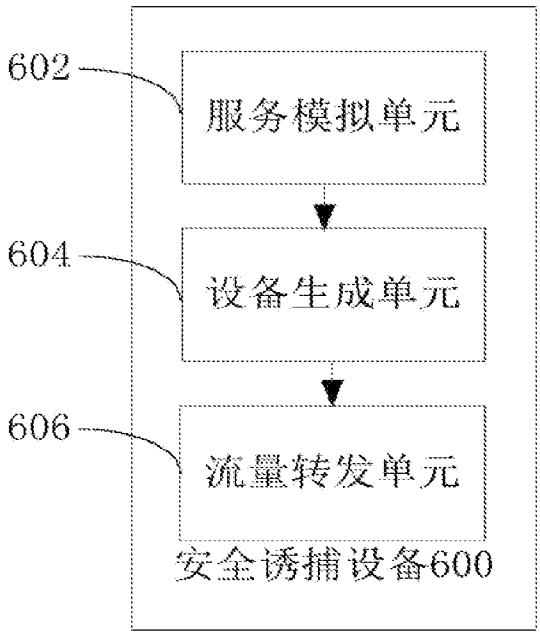

602
服务模拟单元
604
设备生成单元
606
流量转发单元
安全诱捕设备600

图6

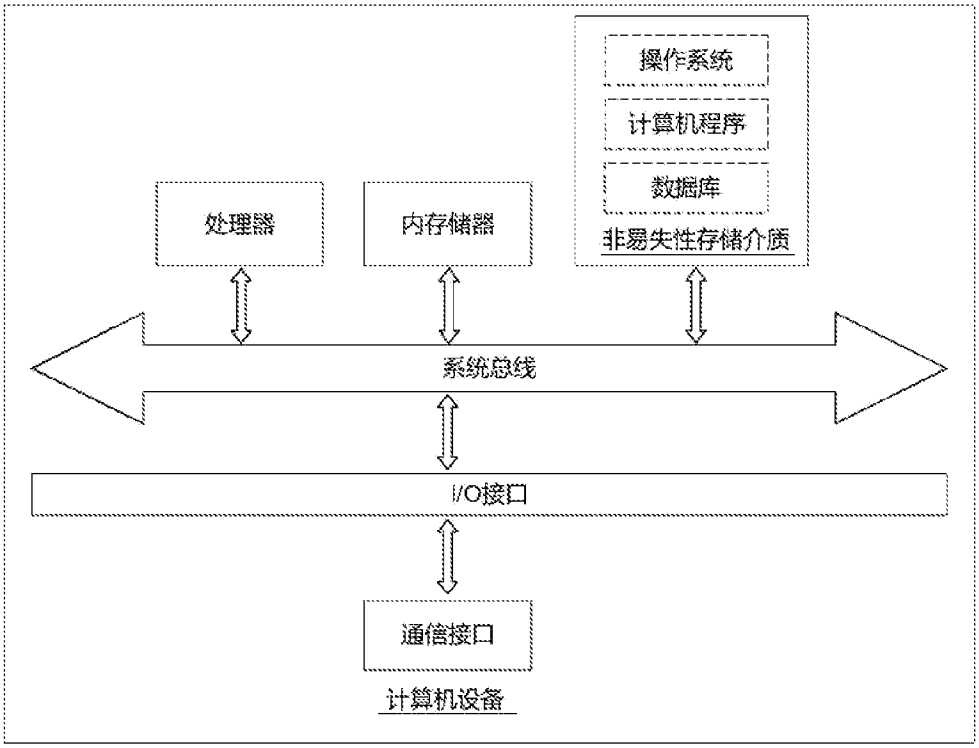

操作系统
计算机程序
数据库
非易失性存储介质
处理器
内存储器
系统总线
I/O接口
通信接口
计算机设备

图7

mass detection data analysis system for APT protection

DOCUMENT ID	DATE PUBLISHED
CN-109995722-A	2019-07-09

INVENTOR INFORMATION

NAME	CITY	STATE	ZIP CODE	COUNTRY
XU, Ji-feng	N/A	N/A	N/A	N/A
QI, Jian-ming	N/A	N/A	N/A	N/A
ZHOU, Jun-song	N/A	N/A	N/A	N/A
CHEN, Dun-jin	N/A	N/A	N/A	N/A

DATE FILED
2017-12-30

CPC CURRENT

TYPE	CPC	DATE
CPCI	H04L63/1466	2013-01-01
CPCI	H04L63/20	2013-01-01
CPCI	H04L63/1416	2013-01-01

Abstract

The invention claims a APT protection of mass detection data analysis system, the system comprising: a data collecting layer and a data processing layer, several mass heterogeneous data sources, wherein the data collecting layer by physical location adjacent or non-adjacent, responsible for providing security data to the data processing layer and scene multi-angle; said data processing layer is responsible for data correlation analysis rule to subscribe the data collecting layer so as to perform analysis and processing. The solution of the invention using large data technology for monitoring and detecting data for depth correlation analysis, not only can realize beforehand alarm, also can realize the blocking in the event, also can realize traceability of post-audit.

Description

INVENTION-TITLE
technical field
The invention belongs to the big data analysis technology field, relating to a detection data analysis system for mass facing APT protection.
BACKGROUND
APT (advanced persistent threat) advanced persistent threat attack, which is different from the traditional network man, APT emphasizes penetrant and get information to steal the enterprises and the government core information asset as the main purpose.
the characteristics of the APT on A and P2 properties: A represents Advanced, mainly expressed on

hypersonic attack means, character information of hacker attack behavior is difficult to extract, attack single point strong hidden performance, diversified attack and attack space is not determined. P represents a Persistent, mainly showing the attack duration and shielding time length, it makes the network threat from random attack of the type become objects, organized, with a population of type attack, which makes the traditional in real time detection, defense mode real-time blocking body is hard to exert the effect. Currently, the APT attack event has been explosive growth. Although already caused by the height the attention of government and enterprise, but the existing protection scheme has certain limitations, cannot cover the global, to possibly caused by missing, and many product can only detect APT attacks, but can not provide real-time protection function.

summary of the invention

The purpose of the invention is to provide a mass detection data analysis system facing APT protection, such a population for APT type attack to the traditional real time monitoring, real-time blocking huge impact is caused by main body of the detecting technology, the monitoring detection data for depth correlation analysis using large data technology, not only can the comprehensive analysis target system whether there is risk being attacked so as to realize beforehand alarm, it can find the attack is carried out, more accurate understanding of the intent and backward tracing. realize the comprehensive determination of the threat so as to timely take related policy to prevent attack to realize blocking event, also for big data analysis the security audit information, reproduction history state and evolution process of the data according to the tracing path to realize post-audit tracing.

In order to solve the above technical problem, the present invention adopts the following technical solution: a detection data analysis system for mass facing APT protection, the system comprising: a data collecting layer and a data processing layer, wherein several mass heterogeneous data sources of the data collecting layer by physical location adjacent or not adjacent, is responsible for the data processing layer provides security data and scene multi-angle; said data processing layer is responsible for data association analyzing rules subscribing to the data collecting layer; so as to carry out analysis processing.

Further data source, the data acquisition layer comprises a flow mirroring, firewall, the UDS and IPS.

Further, the data processing layer by the safety detecting module, safety audit module and large data correlation analysis module.

Further, the safety detection module comprises detection and high detection function.

Further, the correlation analysis module of larger data containing transverse association analysis and longitudinal association analysis function.

Further, the security audit module comprises analysis of vulnerabilities and attack source function.

Compared with the existing technology, the invention has the following beneficial effects:

The solution of the invention for the APT attack to traditional in this population in real time monitoring, real-time blocking huge impact is caused by main body of the detecting technology, using large data technology for monitoring and detecting data for depth correlation analysis so as to realize beforehand alarm and comprehensive determination of the threat, to take the relevant policy block the attack, realize event in blocking and post-audit tracing.

Description

Picture 1 the whole frame image data analysis of mass detection system facing APT protection.

Specific implementation methods

With reference to the accompanying drawings and specific embodiment of this invention will be described in a more detailed and complete. It should be understood that the specific embodiments described here are only used to explain the present invention, but not to limit the scope of the invention.

Referring to FIG. 1, a facing APT protection of mass detection data analysis system of the present invention, the system comprises a data collecting layer and a data processing layer, wherein said data processing layer is responsible for data association analyzing rules subscribing to the data collecting layer; so as to analyze and process, multiple mass heterogeneous data source data source of the data collecting layer comprises physical location flow mirroring, firewall, the UDS, IPS and a series of adjacent or not adjacent, is responsible for providing security data to the data processing layer and multi-angle scene.

a data processing layer with a safety detecting module, safety audit module and large data correlation analysis module, wherein the security detection module further comprises a low level detecting module and position detecting module, a data correlation analysis module is divided into transverse relation analyzing

module and longitudinal associated analysis module, a safety audit module is divided into the vulnerability analysis module and attack source modules, each module is implemented by a variety of techniques.
1) low detection module
Because APT generally uses Oday vulnerability by unknown Trojan obtained authority, remote control, and the detecting device of the traditional based on feature matching always to firstly capture the malicious code sample, then extracting the characteristic and the attack based on feature identification, the presence of hysteresis. low detection is emergency subsystem response platform facing the malfunction detection early warning terminal. system and the traditional safety protection software has intrinsic technical difference. safety enhancing module using the malfunction detection theory model, which not only can effectively monitor the known threat, and can find the unknown threat.
malfunction detection analysis and warning subsystem comprises an application malfunction detection module, system malfunction detection module, a communication malfunction detection module, a transmission malfunction detection module, abnormal data discovery module and other abnormal discovery module. the inner main logic is associated abnormal discovery, decision judging and early warning. associated malfunction detection is the hazard level abnormal set according to the strategy base information, i.e., risk, the potential threat for the 2 time estimation, the policy library and to update the warning through the I/O interface with high monitoring communication, for the 3 inference, and can through I/O interface uploads the abnormal data set, searching for the 4 threat more accurate inference of threat according to user attribute database information. the result set of the data of the user attribute database can be collected through the self-defined by the user or by associating each module of user operation habit for slowly adjusting to generate. constructing the target user attribute database is associated user social attribute, to sense the customizability of threat penetrant. low detection not only can support high detection, also can through the output interface of the warning I/O, the warning information is sent to other safety protection system, such as antivirus and firewall software, honey.
2) high detection module
position detection is facing the core response platform malfunction detection early warning emergency source and pathway. the advantage of high detection is that it can capture discovery abnormality of a backbone network, find a lot of resource attack implementation, detecting and finding that a large source a malicious application. associating low-level detection data, can make high detection has large "view" capability, can effectively analyze and detect unknown threat, tracing the attacker identity, responding to the APT attack.
position detection technology comprises processing fast identification and classification technology for network service based on streaming mode, for cognitive recognition and feature extraction technology to flow based on information theory, network monitoring technology based on content analysis based on detecting technology of behaviour characteristic; to the network data stream and network behaviour as the centre, a multi-angle detection analyzing and calculating abnormal discovery technology, for detecting and analyzing abnormal network technology obtained, uses multi-level network anomaly associated with data fusion technology, multi-layered network abnormal association rule library building technology and so on. position detection by statistical characteristic of the protocol layer of the IP layer, transport layer, application layer, establishing the normal user network access model, detecting the abnormal behavior possible, including abnormal network behaviour of the abnormal flow transmission, DNS of the speed change behavior and other Trojan attack. different behaviour characteristics of user research service network and a gateway on the network, respectively establishing abnormal model to service the network behaviour feature and boundary network behaviour characteristic, discovering unknown threat. the core of the high-position detection is abnormal decision engine, data low detecting malfunction detection decision to provide support for high through I/O interface. In addition, in low detection transmits the abnormal data packet in the malfunction detection module also can incoming high detection system for depth analysis via an I/O interface corresponding to the network traffic. the result of the abnormality decision engine is composed of the corresponding I/O interface output, for low detection reference. result set of malfunction detection also can provide technical support is other safety protective platform through the I/O interface.
3) security audit module
security audit module collecting a system log, resource usage, user behavior and network traffic log and data record, and the vulnerability analysis sub-module and attack tracing sub-module initially analyzes the

collected data, besides using the traditional blacklist and white list analysis method, has to be combined big data correlation analysis module to perform the comprehensive determination, from a lot of log record and data record of mixed in digging the potential threatening problem.

4) large data correlation analysis module

monitoring data of the low level detection and high detection, security audit information for association analysis and data mining to find correlation between the recording, discovery of new features. for the sample data digging out of the manual classification or clustering method uses machine learning for classification result building the feature base training set, thus improving the sorting accuracy of feature database.

log record using large collected data technology to the security audit module, detecting module and monitoring data of the high low level detecting module for big data correlation analysis to comprehensively analyze the target system whether there is likely to be attack to realize beforehand alarm. the attack is being performed for big data analysis, can more accurately understand the intent and backward tracing, realize the comprehensive determination of the threat so as to timely take related policy to prevent attack to realize blocking in the event. and the security audit information of the whole attack process for big data analysis, reproduction history state and evolution process of the data according to the tracing path, realizing traceability of post-audit.

detecting data generated by all kinds of APT attack detection tool, key, context and information flow, log information of the external to form the mass data with multiple different structures. data of the multi-source multi-structural, long time, low density property, defense method to the big data analysis brings the protective challenge. is the key problem of how to efficiently process data analysis by data technology.

1) data management technology

in today's large data age, complete difficult development of relational data management system (RDBMS) encountered a single data type, insufficient expandability and so on. along with the increase of the data amount, data analysis of basic policy is pushed to the computing data, rather than data calculation. In recent years, the non-relational data management technology rise suddenly, the plurality of types of data for effective management, processing and analyzing, obtaining good system performance through parallel processing technology, and satisfies the processing requirement of the constantly increasing data amount to its high expansibility.

the data engine relational database as the core, the data from various sources by the ETL tool leads the relational database system. client end tool realized by SQL language routine report generation. for complex analysis, the SQL expression ability to expose its limitations. If the data extracted from the database, the leading front end analysis tool (SAS.SPSS) for subsequent analysis, will result in a large amount of data movement. meanwhile, the front end analysis data processing ability of the tool the limited memory size, data analysis efficiency will deteriorate. In addition, because data tend to types, large scale relational database organization and processing ability cannot deal with high-dimensional large scale when the challenge of null data. Therefore, in the data analysis process is necessarily parallel calculation, the selection of the high performance data management.

relative to the RDBMS, MapReduce technology is represented by the non-relational data management technology support a higher fault tolerance from the storage model and the calculation model, stronger expansibility, analysis task is big data analysis provides a good operation platform is ensured, at the same time, cannot be used for more easily by using SQL to express the MapReduce computing function expression. As a aiming at solving the large-scale non-structured data fast and batch processing of parallel technique frame, MapReduce to obtain the cyrtotrachelus buqueti development. low performance problem caused by too simple to overcome the scheduling algorithm, for multi-core CPU, GPU, heterogeneous system and the cloud platform of different environment such as scheduling policy can be optimized. through the logic model of the defined data processing flow, converting the logical model into a physical model using model conversion algorithm and code generation arithmetic to realize the multi-data source for processing MapReduce data support. flow and localized by an intermediate result cache, improves the MapReduce data stream processing ability, and claims a method for high speed sensing data stream real-time MapReduce method.

To detect APT attack behaviour of each attack stage from the multi-dimensional data, the data must be deeply analyzed, rather than merely generating a simple report. This complex analysis must rely on complicated analysis model, it is difficult to perform expression using SQL, belonging to the depth analysis.

Therefore, the challenge facing large data depth analysis, in MapReduce technology is represented by the non-relational data management technology has obvious advantages.

2) the depth analysis technology

Techtarget defined the depth analysis is to use complex data processing technology to be integrated obtaining knowledge from the structured, non-structured and semi-structured multi-source data. it starts from the requirement of the user, the way to effectively, the user can accept a large amount of complicated distributed data for analysis, extraction and summarization, so as to provide the corresponding guidance to the user decision.

Because the depth analysis usually needs to finish the PB level even level EB data for accurately locating and complex query, and normally used for environment real-time and near real-time response, a typical data analysis operation (e.g., aggregation, rotation, slicing and summary) analysis technique cannot meet the requirement, requirement people is more complex, for example, time series analysis, path analysis, What-if analysis, social network analysis and complex statistical analysis model and did not attempt due to hardware/software limitation. following the time sequence analysis and pattern analysis is described: analyzing the time sequence). time sequence analyzing method of mathematical statistics according to the time order of data sequence is processed, and thus predicting the future development trend. it not only admit the continuity of business development but also considers the randomness of object development. Because the time sequence analysis comprises predicting future data according to the history data, so the commonly used in market potential, hydrology forecast prediction, weather forecast, national economic macro-control and enterprise operation and management and so on.

(social network analysis and graph analysis. mode of social network analysis focuses on relationships and relationship, on the concept, the manner and method differs from the traditional statistical analysis and data processing method. In the figure, 1 independent of entity 1 node represents a social network contact 1, in edge represents the entity between the 2 points. using the social network analysis, can obtain some useful knowledge, such as found in the network key entity and so on (each group key entity in the network connected together). Such information can be used for sentiment analysis, product sales and potential threat prediction and so on.

APT attack data based on complex and multi-source characteristics, introduction depth analysis can effectively extract knowledge from the history data, and to predict the attack action has not occurred so as to nip in the bud.

3) large data mining technology

safety detection based on large data mining is a class of important safety detecting technique. the traditional data mining algorithm based on serial calculation, processing data size limited, once the migration to large data environment, the executing efficiency will greatly decrease even cannot run and cannot meet the present requirement of large data.

Cloud computing is a big data analysis and mining technology can support of the computing platform, mining and analyzing large data point is the mass data, it can not only finish operation using a single computer, but must adopts the distributed computing architecture. Therefore, relying on distributed processing, distributed database of cloud computing, cloud storage and virtualization technology is an effective means for improving the large data mining efficiency, dispersing the computing task while running on a different computer, the various application systems can be adjusted according to needs to acquire storage resource, computing resource and other service resource. MapReduce, Hadoop, Spark and other high performance, high expandability of the programming model of parallel computing, distributed processing large data frame and continuously emerge and development of related key technology, makes the data storage and distributed computing of large data. based on distributed computing, large data mining technology can be better to provide new theoretical and technical support based on security detection problem of the large data. current security detection technology based on large data mining main research various data mining parallelization strategy based on distributed computing algorithm, new large data mining method and realizing abnormity analysis and detection of large scale on MapReduce, Spark frame and so on.

The above description is only the preferred embodiments of the present invention, which are not used to restrict the present invention to those skilled in the art, the present invention may have various modifications and changes. Any modifications, within the spirit and principle of the invention the any modification, equivalent replacing, improved and so on, are contained in the protection range of the invention.

Claims

1. for mass of APT protection detection data analysis system, wherein the system comprises a data collecting layer and a data processing layer, several mass heterogeneous data sources, wherein the data collecting layer by physical location adjacent or non-adjacent, responsible for providing security data to the data processing layer and scene multi-angle; said data processing layer is responsible for data correlation analysis rule to subscribe the data collecting layer so as to perform analysis and processing.
2. The facing APT protection according to claim 1, the mass detection data analysis system, wherein the data source of the data collecting layer comprises a flow mirroring, firewall, the UDS and IPS.
3. The APT protection-oriented data analysis of mass detection system according to claim 1, wherein, the data processing layer formed by the safety detecting module, safety audit module and large data correlation analysis module.
4. The APT protection-oriented data analysis of mass detection system according to claim 3, wherein, said safety detection module comprises detection and high detection function.
5. The APT-oriented protection according to claim 3 mass detection data analysis system, wherein the large data correlation analysis module comprising transverse association analysis and longitudinal association analysis function.
6. The APT protection-oriented data analysis of mass detection system according to claim 3, wherein, said safe audit module comprises vulnerability analysis and attack source function.

Copyright © 2022 Clarivate Analytics. All rights reserved. Republication or redistribution of Clarivate Analytics content, including by framing or similar means, is prohibited without the prior written consent of Clarivate Analytics. Clarivate and its logo are trademarks of the Clarivate Analytics group.

(19)中华人民共和国国家知识产权局

(12)发明专利申请

(10)申请公布号 CN 109995722 A
(43)申请公布日 2019.07.09

(21)申请号 201711489965.1

(22)申请日 2017.12.30

(71)申请人 广州明领基因科技有限公司
地址 510610 广东省广州市天河区林和东路281号天伦控股大厦17层自编05单元

(72)发明人 徐继峰 祁建明 周峻松 陈墩金

(51)Int.Cl.
H04L 29/06(2006.01)

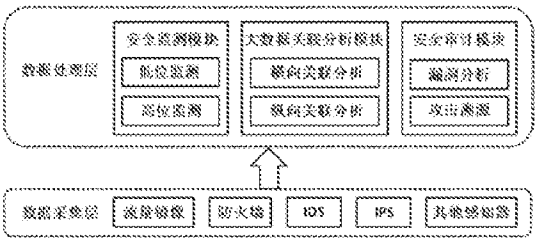
权利要求书1页 说明书5页 附图1页

(54)发明名称

面向APT防护的海量检测数据分析系统

(57)摘要

本发明公开了一种面向APT防护的海量检测数据分析系统，该系统包括：数据采集层以及数据处理层；其中，所述数据采集层由物理位置相邻或不相邻的若干海量异构数据源组成，负责为所述数据处理层提供多角度的安全数据及情景；所述数据处理层负责根据关联分析规则订阅所述数据采集层的数据，从而进行分析处理。本发明方案利用大数据技术对监控检测数据进行深度关联分析，不仅能够实现事前预警，也可实现事中阻断，还可实现事后审计溯源。



CN 109995722 A

1.面向APT防护的海量检测数据分析系统，其特征在于，所述系统包括：数据采集层以及数据处理层；其中，所述数据采集层由物理位置相邻或不相邻的若干海量异构数据源组成，负责为所述数据处理层提供多角度的安全数据及情景；所述数据处理层负责根据关联分析规则订阅所述数据采集层的数据，从而进行分析处理。

2.根据权利要求1所述的面向APT防护的海量检测数据分析系统，其特征在于，所述数据采集层的数据来源包括流量镜像、防火墙、UDS及IPS等。

3.根据权利要求1所述的面向APT防护的海量检测数据分析系统，其特征在于，所述数据处理层由安全检测模块、安全审计模块以及大数据关联分析模块组成。

4.根据权利要求3所述的面向APT防护的海量检测数据分析系统，其特征在于，所述安全检测模块包含低位检测及高位检测功能。

5.根据权利要求3所述的面向APT防护的海量检测数据分析系统，其特征在于，所述大数据关联分析模块含有横向关联分析及纵向关联分析功能。

6.根据权利要求3所述的面向APT防护的海量检测数据分析系统，其特征在于，所述安全审计模块包含漏洞分析及攻击溯源功能。

面向APT防护的海量检测数据分析系统

技术领域

[0001] 本发明属于大数据分析技术领域，涉及一种面向APT防护的海量检测数据分析系统。

背景技术

[0002] APT(advanced persistent threat)即高级持续性威胁攻击，其不同于传统的网络人侵，APT侧重渗透和获取信息，以窃取企业和政府核心信息资产为主要目的。

[0003] APT的特点表现在A和P2个特性上：A代表Advanced，主要表现在攻击手段高超，攻击行为特征难以提取，攻击单点隐蔽性强，攻击渠道多样化以及攻击空间不确定。P代表Persistent，主要表现在攻击持续和隐蔽时间长，它使网络威胁从散兵游勇式的随机攻击变成有目的、有组织、有预谋的群体式攻击，使传统的以实时检测、实时阻断为主体的防御方式难以再发挥作用。

[0004] 目前，APT攻击事件已趋于爆发式增长。虽说已引起了政府和企业的高度重视，但现有防御方案大都具有一定的局限性，无法覆盖全局，从而可能导致漏报，而且很多产品只能检测APT攻击，却无法提供实时防护功能。

发明内容

[0005] 本发明目的在于提供一种面向APT防护的海量检测数据分析系统，针对APT这种群体式攻击给传统的以实时监测、实时阻断为主体的检测技术带来的巨大冲击，利用大数据技术对监控检测数据进行深度关联分析，不仅能够综合分析目标系统是否存在被攻击的风险，进而实现事前预警，也可发现正在进行的攻击，更加准确地理解意图和反向追踪，实现对威胁的综合研判，从而及时采取相关的策略阻止攻击，实现事中阻断，还可同时对安全审计信息进行大数据分析，根据追踪路径重现数据的历史状态和演变过程，实现事后审计溯源。

[0006] 为解决上述技术问题，本发明采用如下的技术方案：一种面向APT防护的海量检测数据分析系统，该系统包括：数据采集层以及数据处理层；其中，所述数据采集层由物理位置相邻或不相邻的若干海量异构数据源组成，负责为所述数据处理层提供多角度的安全数据及情景；所述数据处理层负责根据关联分析规则订阅所述数据采集层的数据，从而进行分析处理。

[0007] 进一步地，所述数据采集层的数据来源包括流量镜像、防火墙、UDS及IPS等。

[0008] 进一步地，所述数据处理层由安全检测模块、安全审计模块以及大数据关联分析模块组成。

[0009] 进一步地，所述安全检测模块包含低位检测及高位检测功能。

[0010] 进一步地，所述大数据关联分析模块含有横向关联分析及纵向关联分析功能。

[0011] 进一步地，所述安全审计模块包含漏洞分析及攻击溯源功能。

[0012] 本发明与现有技术相比具有以下的有益效果：

[0013] 本发明方案针对APT这种群体式攻击给传统的以实时监测、实时阻断为主体的检测技术带来的巨大冲击，利用大数据技术对监控检测数据进行深度关联分析，进而实现事前预警及对威胁的综合研判，及时采取相关的策略阻止攻击，实现事中阻断以及事后审计溯源。

附图说明

[0014] 图1是面向APT防护的海量检测数据分析系统的整体框架图。

具体实施方式

[0015] 下面结合附图及具体实施例对本发明进行更加详细与完整的说明。可以理解的是，此处所描述的具体实施例仅用于解释本发明，而非对本发明的限定。

[0016] 参照图1，本发明的一种面向APT防护的海量检测数据分析系统，该系统包括：数据采集层以及数据处理层；其中，所述数据处理层负责根据关联分析规则订阅所述数据采集层的数据，从而进行分析处理；所述数据采集层的数据来源包括流量镜像、防火墙、UDS、IPS等一系列物理位置相邻或不相邻的若干海量异构数据源组成，负责为所述数据处理层提供多角度的安全数据及情景。

[0017] 数据处理层有安全检测模块、安全审计模块及大数据关联分析模块，其中安全检测模块又包括低位检测模块和高位检测模块，大数据关联分析模块分为横向关联分析模块和纵向关联分析模块，安全审计模块分为漏洞分析模块及攻击溯源模块，每个模块由多种技术实现。

[0018] 1)低位检测模块

[0019] 由于APT普遍采用0day漏洞获取权限，通过未知木马进行远程控制，而传统基于特征匹配的检测设备总是要先捕获恶意代码样本，才能提取特征并基于特征进行攻击识别，这就存在先天的滞后性。低位检测是面向终端的异常发现预警应急响应平台的子系统。系统与传统的安全防护软件有本质技术区别。安全增强模块采用异常发现理论模型，不仅能够对已知威胁进行有效监测，同时能够对未知威胁及时发现。

[0020] 异常发现分析与预警子系统包含应用异常发现模块、系统异常发现模块、通信异常发现模块、传输异常发现模块、数据异常发现模块和其他异常发现模块。内部主要逻辑是关联异常发现、决策判决和预警。关联异常发现是根据策略库信息判断异常集的危害等级，即风险，对潜在威胁进行2次推测；其策略库可通过I/O接口与高位监测通信并进行更新、预警，根据用户属性库信息对威胁进行第3次推断，并可通过I/O接口将异常数据集上传，以寻求更为精确的第4次威胁推断。用户属性库的数据可通过用户自主定义或者通过关联各模块收集的用户操作习惯的结果集进行缓慢调整而生成。构建用户属性库的目的是关联用户社会属性，以感知定制性的威胁渗透。低位检测不仅可以支持高位检测，还可以通过预警I/O的输出接口，将预警信息发送给其他安全防护系统，如杀毒软件、蜜网和防火墙等。

[0021] 2)高位检测模块

[0022] 高位检测是面对源头和途径的异常发现预警应急响应平台的核心。高位检测的优势在于可以捕获发现大量骨干网络的异常，挖掘发现大量攻击实施者的资源，检测发现大量源头恶意应用。关联低位检测数据，可使高位检测具备“大视野”的能力，可以有效地分析

发现未知威胁，追溯攻击者身份，应对APT攻击。

[0023] 高位检测技术包括基于流模式对网络服务进行快速识别和分类技术；基于信息论对流量进行认知识别和特征提取技术；基于内容分析的网络监测技术；基于行为特征的检测技术；以网络数据流和网络行为为中心，多角度检测分析计算异常发现技术；针对检测分析获取的网络异常技术；采用多层次网络异常关联与数据融合技术；多层次网络异常关联规则库构建技术等。

[0024] 高位检测通过统计IP层、传输层、应用层等各协议层的特征，建立正常用户网络访问模型，检测可能的异常行为，包括流量传输异常、DNS的速变攻击行为和其他木马的异常网络行为。研究业务网络和出入口网络上用户的不同行为特征，分别对业务网络行为特征和边界网络行为特征建立异常模型，发现未知威胁。高位检测的核心是异常决策引擎，低位检测数据通过I/O接口为高位异常发现决策提供支持。此外低位检测中传输异常发现模块中的异常数据包也可通过与网络信息流对应的I/O接口传入高位检测体系进行深度分析。异常决策引擎的结果可由对应的I/O接口输出，供低位检测参考。异常发现的结果集亦可通过I/O接口为其他安全防护平台提供技术支持。

[0025] 3)安全审计模块

[0026] 安全审计模块收集包括系统日志、资源使用、用户行为以及网络流量等日志和数据记录，并利用漏洞分析子模块和攻击溯源子模块初步分析所收集的数据，除使用传统的黑名单、白名单等分析方法外，更要结合大数据关联分析模块进行综合研判，以从大量的、混杂的日志记录和数据记录中挖掘出潜在威胁问题。

[0027] 4)大数据关联分析模块

[0028] 对低位检测与高位检测的监测数据、安全审计信息进行关联分析和数据挖掘，找出记录之间的相关性，发现新的特征。对于挖掘出来的样本数据进行人工的分类或聚类，以机器学习的方法用分类结果构建特征库训练集，不断提高特征库的分类准确率。

[0029] 利用大数据技术对安全审计模块收集的日志记录、高位检测模块以及低位检测模块的监控数据进行大数据关联分析，以综合分析目标系统是否存在被攻击的可能，实现事前预警。对正在进行的攻击进行大数据分析，可以更准确地理解意图和反向追踪，实现对威胁的综合研判，从而及时采取相关的策略阻止攻击，实现事中阻断。同时对攻击全过程的安全审计信息进行大数据分析，根据追踪路径重现数据的历史状态和演变过程，实现事后审计溯源。

[0030] 各类APT攻击检测工具产生的检测数据、关键流量、日志、上下文以及外部的情报信息等形成了具有多种不同结构的海量数据。数据的多来源性、多结构性、长时序性、低密度性，给大数据分析的防御方法带来了防护挑战。如何有效地利用大数据技术处理分析数据是关键问题。

[0031] 1)数据管理技术

[0032] 在当今大数据时代，曾发展完善的关系数据管理系统(RDBMS)遇到了数据类型单一、扩展性不足等困难。随着数据量的增长，数据分析的基本策略是把计算推向数据，而不是把数据推向计算。近年来，非关系数据管理技术异军突起，对多种类型的数据进行有效的管理、处理和分析；通过并行处理技术获得良好的系统性能；并以其高度的扩展性满足不断增长的数据量的处理要求。

[0033] 关系数据库作为核心的数据引擎，各种来源的数据通过ETL工具导入关系数据库系统。客户端工具通过SQL语言实现例行性的报表生成。针对复杂的分析，SQL的表达能力就暴露出其局限性。如果把数据从数据库中提取出来，导入前端分析工具(SAS.SPSS)进行后续分析，将导致大量的数据移动。同时前端分析工具的数据处理能力将受限于内存大小，数据分析效率将大打折扣。另外，由于当今数据趋向类型多样、规模巨大，关系数据库的组织和处理能力已无法应对大规模的高维时空数据的挑战。因此，在数据分析过程中引入并行计算，将是实现高性能数据管理的必然选择。

[0034] 相对于RDBMS，以MapReduce技术为代表的非关系数据管理技术从存储模型和计算模型上支持更高的容错性、更强的扩展性，为大数据分析提供了很好的运行平台保障，同时，难以用SQL进行表达的分析任务更容易用MapReduce计算函数表达。作为一种旨在解决大规模非结构化数据快速批量处理的并行技术框架，MapReduce获得了长足的发展。为了克服调度算法过于简单导致的性能低下问题，面向多核CPU、GPU、异构系统以及云平台等不同环境的调度策略得以优化。通过定义数据处理流程的逻辑模型，使用模型转换算法和代码生成算法将逻辑模型转化为物理模型，实现了MapReduce对多数据源的数据处理的支持。通过中间结果缓存、流水化和本地化改进了MapReduce的数据流处理能力，进而提出一种针对高速传感数据流的实时MapReduce方法。

[0035] 为了从多维度的数据中检测APT攻击的各个攻击阶段的行为，必须对数据进行深入的分析，而不是仅仅生成简单的报表。这种复杂的分析必须依赖于复杂的分析模型，很难用SQL来进行表达，属于深度分析。因此，面对大数据深度分析的挑战，以MapReduce技术为代表的非关系数据管理技术具有明显的优势。

[0036] 2)深度分析技术

[0037] Techtarget对深度分析的定义是利用复杂数据处理技术从包含结构化、非结构化及半结构化的多源数据集中获取知识。其从用户的需求出发，以有效、用户可接受的方式对大量复杂分布式数据进行分析、提取与汇总，从而为用户决策提供相应的指导。

[0038] 由于深度分析通常需要完成PB级甚至EB级数据的精确定位及复杂查询，并常用于需要实时和近实时响应的环境，典型的数据分析操作(例如：聚集、旋转、切片和汇总)已无法满足其要求，需要引入更复杂的分析技术，例如：时间序列分析、路径分析、What-if分析、社会网络分析以及由于硬件/软件限制而未曾尝试过的复杂统计分析模型。以下对时间序列分析和图分析进行简单介绍：

[0039] ①时间序列分析。时间序列分析利用数理统计的方法对按时间顺序的数据序列加以处理，并以此预测未来事物发展趋势。其既承认事物发展的延续性又考虑到事物发展的随机性。由于时间序列分析具备根据历史数据预测未来数据的特点，故其常应用于市场潜量预测、气象预报、水文预报、国民经济宏观控制以及企业经营管理等领域。

[0040] ②社会网络分析和大规模图分析。社会网络分析侧重于关系和关系的模式，在概念上，采用的方式和方法有别于传统的统计分析和数据处理方法。在图中，1个节点代表社会网络中1个独立的实体，2点之间的1条边代表实体之间的联系。利用社会网络分析，可以获取一些有用的知识，比如发现网络中的关键实体等(关键实体把网络中的各个小组连接在一起)。这些信息可以用于舆情分析、产品销售以及潜在威胁预测等领域。

[0041] 基于APT攻击数据的复杂和多源性特征，引入深度分析可以有效地从历史数据中

提取知识，并以此对未发生的攻击行为进行预测，防患于未然。

[0042] 3)大数据挖掘技术

[0043] 基于大数据挖掘的安全检测是一类重要的安全检测技术。传统的数据挖掘算法基于串行计算，处理数据规模有限，一旦迁移到大数据环境，其执行效率会大幅下降甚至无法运行，无法满足当今大数据时代的需求。

[0044] 云计算是大数据分析与挖掘技术可依托的计算平台，大数据的重点在于对海量数据的挖掘与分析，无法仅使用单台计算机完成运算，而必须采用分布式计算架构。因此，依托云计算的分布式处理、分布式数据库、云存储和虚拟化技术是提高大数据挖掘效率的有效手段，将计算任务分散在不同的计算机上同时运行，使各种应用系统可以根据需要获取存储资源、计算资源和其他服务资源。MapReduce，Hadoop，Spark等高性能、高扩展性的并行计算编程模型、分布式大数据处理框架以及相关关键技术的不断涌现和发展，使得大数据的数据存储和分布式计算成为现实。以分布式计算为基础，大数据挖掘技术可以更好地为基于大数据的安全检测问题提供新的理论与技术支持。目前基于大数据挖掘的安全检测技术主要研究各种数据挖掘算法的并行化策略，基于分布式计算的新型大数据挖掘方法以及在MapReduce，Spark框架上实现大规模的异常分析和检测等。

[0045] 以上所述仅为本发明的优选实施例，并不用于限制本发明，对于本领域技术人员而言，本发明可以有各种改动和变化。凡在本发明的精神和原理之内所作的任何修改、等同替换、改进等，均应包含在本发明的保护范围之内。

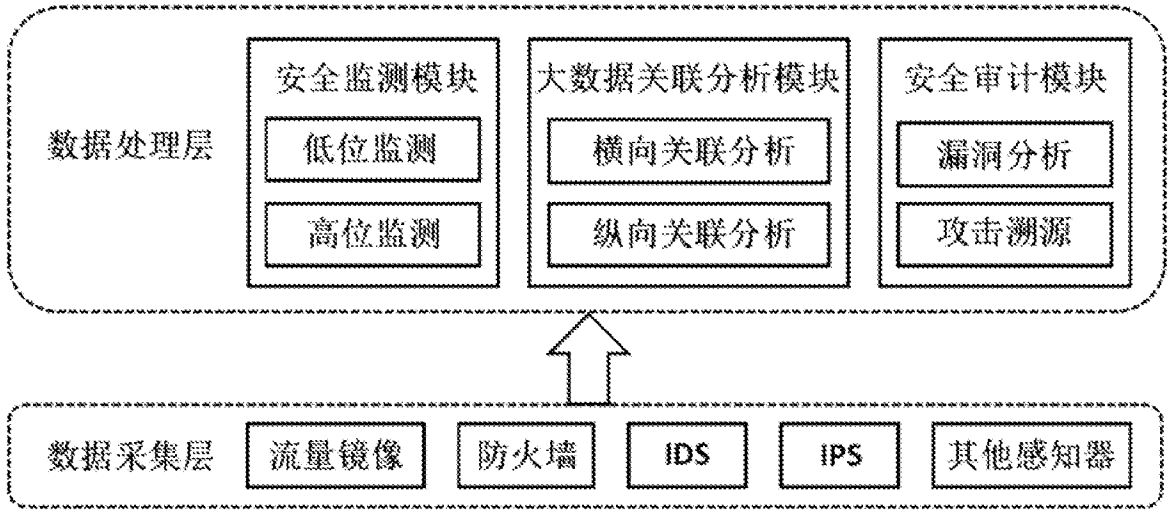

数据处理层
安全监测模块
低位监测
高位监测
大数据关联分析模块
横向关联分析
纵向关联分析
安全审计模块
漏洞分析
攻击溯源
数据采集层
流量镜像
防火墙
IDS
IPS
其他感知器

图1

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Use as many sheets as necessary)				Complete if Known	
				Application Number	17/657,981
				Filing Date	April 5, 2022
				First Named Inventor	Avi SHUA
				Art Unit	2431
				Examiner Name	SU, SARAH
Sheet	1	of	1	Attorney Docket Number	**15635.0008-00000**

U.S. PATENTS

Examiner Initials*	Cite No.	Document Number Number-Kind Code (if known)	Issue Date MM-DD-YYYY	Name of Patentee or Applicant of Cited Document	Pages, Columns, Lines, Where Relevant Passages or Relevant Figures Appear
		US-8,621,073-B2	12-31-2013	McClure et al.	

U.S. PUBLISHED PATENT APPLICATIONS

Examiner Initials*	Cite No.	Document Number Number-Kind Code (if known)	Publication Date MM-DD-YYYY	Name of Patentee or Applicant of Cited Document	Pages, Columns, Lines, Where Relevant Passages or Relevant Figures Appear
		US-2018/0288087-A1	10-04-2018	HITTEL et al.	
		US-2017/0012998-A1	01-12-2017	Thornbury et al.	
		US-2021/0377287-A1	12-02-2021	SHUA	

Note: Submission of copies of U.S. Patents and published U.S. Patent Applications is not required.

FOREIGN PATENT DOCUMENTS

Examiner Initials*	Cite No.	Foreign Patent Document Country Code Number Kind Code (if known)	Publication Date MM-DD-YYYY	Name of Patentee or Applicant of Cited Document	Pages, Columns, Lines, Where Relevant Passages or Relevant Figures Appear	Translation

NON PATENT LITERATURE DOCUMENTS

Examiner Initials*	Cite No.	Include name of the author (in CAPITAL LETTERS), title of the article (when appropriate), title of the item (book, magazine, journal, serial, symposium, catalog, etc.), date, page(s), volume-issue number(s), publisher, city and/or country where published.	Translation
		Final Office Action Application No. 17/658,156 dated September 13, 2022, in the United States Patent and Trademark Office. (15635.0010-00000)	N/A
		Notice of Allowance No. 17/716,491 dated October 11, 2022, in the United States Patent and Trademark Office. (15635.0006-00000)	N/A
		Notice of Allowance No. 17/716,231 dated September 6, 2022, in the United States Patent and Trademark Office. (15635.0004-00000)	N/A

Examiner Signature	/SARAH SU/	Date Considered	11/23/2022

*EXAMINER: Initial if reference considered, whether or not citation is in conformance with MPEP 609. Draw line through citation if not in conformance and not considered. Include copy of this form with next communication to applicant.

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.S/

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Use as many sheets as necessary)				Complete if Known	
				Application Number	17/657,981
				Filing Date	April 5, 2022
				First Named Inventor	Avi SHUA
				Art Unit	2431
				Examiner Name	2397
Sheet	1	of	1	Attorney Docket Number	**15635.0008-00000**

U.S. PATENTS

Examiner Initials*	Cite No.	Document Number / Number-Kind Code (if known)	Issue Date MM-DD-YYYY	Name of Patentee or Applicant of Cited Document	Pages, Columns, Lines, Where Relevant Passages or Relevant Figures Appear

U.S. PUBLISHED PATENT APPLICATIONS

Examiner Initials*	Cite No.	Document Number / Number-Kind Code (if known)	Publication Date MM-DD-YYYY	Name of Patentee or Applicant of Cited Document	Pages, Columns, Lines, Where Relevant Passages or Relevant Figures Appear
		US-2019/0116199-A1	04-18-2019	Stopel et al.	

Note: Submission of copies of U.S. Patents and published U.S. Patent Applications is not required.

FOREIGN PATENT DOCUMENTS

Examiner Initials*	Cite No.	Foreign Patent Document — Country Code Number Kind Code (if known)	Publication Date MM-DD-YYYY	Name of Patentee or Applicant of Cited Document	Pages, Columns, Lines, Where Relevant Passages or Relevant Figures Appear	Translation

NON PATENT LITERATURE DOCUMENTS

Examiner Initials*	Cite No.	Include name of the author (in CAPITAL LETTERS), title of the article (when appropriate), title of the item (book, magazine, journal, serial, symposium, catalog, etc.), date, page(s), volume-issue number(s), publisher, city and/or country where published.	Translation
		Advisory Action No. 17/657,972 dated October 31, 2022, in the United States Patent and Trademark Office. (15635.0005-00000)	N/A

Examiner Signature	/SARAH SU/	Date Considered	11/23/2022

*EXAMINER: Initial if reference considered, whether or not citation is in conformance with MPEP 609. Draw line through citation if not in conformance and not considered. Include copy of this form with next communication to applicant.

ALL REFERENCES CONSIDERED EXCEPT WHERE LINED THROUGH. /S.S/

PE2E SEARCH - Search History (Prior Art)

Ref #	Hits	Search Query	DBs	Default Operator	Plurals	British Equivalents	Time Stamp
L1	10	((("SHUA") near3 ("Avi"))).INV.	(US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT)	ADJ	ON	ON	2022/10/13 10:09 AM
L2	6	L1 AND (analy$3 OR examin$3 OR examination OR inspect$3) WITH (asset OR resource) WITH (risk OR threat OR vulnerab$5 OR attack)	(US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT)	ADJ	ON	ON	2022/10/13 10:11 AM
L3	0	L1 AND (analy$3 OR examin$3 OR examination OR inspect$3) WITH (asset OR resource) WITH (risk OR threat OR vulnerab$5 OR attack) WITH back$1ward	(US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT)	ADJ	ON	ON	2022/10/13 10:12 AM
L4	1	((("Orca") near3 ("Security"))).AS,AANM.	(USPAT)	ADJ	ON	ON	2022/10/13 10:13 AM
L5	38	("0002014" OR "10152487" OR "10868825" OR "11036856" OR "11075923" OR "20130262801" OR "20160330237" OR "20180205755" OR "20180367515" OR "20190250857" OR "20190327272" OR "20200097662" OR "20200220887" OR "20200228560" OR "20200244692" OR "20200344256" OR "20200389483" OR "20200396254" OR "20210099478" OR "20210342297" OR "20210409438" OR "20220000021" OR "20220000191" OR "20220001241" OR "20220100869" OR "20220141247" OR "20220182294" OR "20220239681" OR "7013395" OR "7143143" OR "8272061" OR "9294498" OR	(US-PGPUB; USPAT)	ADJ	ON	ON	2022/10/13 10:14 AM

		"9356942" OR "9413774" OR "9569201" OR "9686296" OR "9749349" OR "9794289").pn.					
L6	2	("20190281115" OR "20200137097").pn.	(US-PGPUB; USPAT)	ADJ	ON	ON	2022/10/13 10:22 AM
L7	311485	(G06F9/45558 OR G06F9/5072 OR H04L63/1433 OR G06F2009/4557 OR G06F2009/45595 OR H04L63/14$2).cpc.	(US-PGPUB; USPAT; USOCR; FIT (AU, AP, AT, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, OA, RU, SU, WO, MC, MD, MY, NL, NO, NZ, PH, PL, PT, RO, RS, SE, SG, SI, SK, TH, TN, TR, TW, UA, VN); FPRS; EPO; JPO; DERWENT; IBM_TDB)	ADJ	ON	ON	2022/10/13 10:45 AM
L8	1707	L7 AND (asset OR resource) WITH (risk OR threat OR vulnerab$5) WITH cloud	(US-PGPUB; USPAT; USOCR; FIT (AU, AP, AT, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, OA, RU, SU, WO, MC, MD, MY, NL, NO, NZ, PH, PL, PT, RO, RS, SE, SG, SI, SK, TH, TN, TR, TW, UA, VN); FPRS; EPO; JPO; DERWENT; IBM_TDB)	ADJ	ON	ON	2022/10/13 10:48 AM
L9	254	L7 AND (analy$3 OR examin$3 OR examination OR inspect$3) WITH (asset OR resource) WITH (risk OR threat OR vulnerab$5) WITH cloud	(US-PGPUB; USPAT; USOCR; FIT (AU, AP, AT, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, OA, RU, SU, WO, MC, MD, MY, NL, NO, NZ, PH, PL, PT, RO, RS, SE, SG, SI, SK, TH, TN, TR, TW, UA, VN); FPRS; EPO; JPO; DERWENT; IBM_TDB)	ADJ	ON	ON	2022/10/13 10:48 AM

L10	45	(Artificial Intelligence) Similar to: Publication Status Unavailable from Similarity Search Text: (CN-106663022-A OR US-11099872-B2 OR US-20190073237-A1 OR US-10120706-B2 OR US-8621030-B2 OR US-20080080514-A1 OR US-9088590-B2 OR US-20140207906-A1 OR US-9342346-B2 OR EP-3177994-A1 OR WO-2016016695-A1 OR US-20160026489-A1 OR CN-104035868-A OR CN-104035868-B OR WO-2015188504-A1 OR CN-106802823-A OR US-20170083478-A1 OR US-10235341-B2 OR CN-106802823-B OR US-20150254095-A1 OR US-20160267051-A1 OR CN-107329801-A OR US-9417903-B2 OR CN-107329801-B OR CN-111538561-A OR US-20160239323-A1 OR EP-4002139-A3 OR CN-111857960-A OR CN-109416661-A OR US-9367370-B2 OR US-20160055041-A1 OR WO-2018087906-A1 OR US-10705879-B2 OR US-20200192654-A1 OR US-10740088-B2 OR CN-109416661-B OR CA-2753228-A1 OR WO-2010111149-A2 OR US-20100250868-A1 OR EP-2411915-A2 OR CA-2753228-C OR US-9529636-B2 OR RU-2011139151-A OR RU-2569805-C2 OR US-20180365138-A1 OR KR-101661783-B1 OR EP-2411915-B1 OR US-20190129700-A1 OR US-10628145-B2 OR US-20180121183-	(US-PGPUB; USPAT; USOCR; FIT (AU, AP, AT, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, OA, RU, SU, WO, MC, MD, MY, NL, NO, NZ, PH, PL, PT, RO, RS, SE, SG, SI, SK, TH, TN, TR, TW, UA, VN))	ADJ	ON	ON	2022/10/13 10:49 AM

		A1).did.					
L11	46	(Artificial Intelligence) Similar to: Publication Status Unavailable from Similarity Search Text: (US-20180270130-A1 OR US-20180365138-A1 OR US-9367370-B2 OR US-20160055041-A1 OR US-20190079896-A1 OR US-10657095-B2 OR US-20140207906-A1 OR US-9088590-B2 OR US-10120706-B2 OR US-11099872-B2 OR CN-111538561-A OR US-20150254095-A1 OR US-9495212-B2 OR US-20150339161-A1 OR US-20150026287-A1 OR US-9348649-B2 OR US-8856337-B2 OR WO-2015188504-A1 OR US-20130046892-A1 OR CN-106688208-A OR CN-106688208-B OR JP-2013041569-A OR US-20080080514-A1 OR US-8621030-B2 OR US-20190073237-A1 OR JP-5860320-B2 OR US-20160267051-A1 OR CN-112000606-A OR US-10621138-B2 OR CN-106663022-A OR KR-20100027936-A OR US-20160205215-A1 OR EP-3198806-B1 OR EP-3198806-A1 OR WO-2016045055-A1 OR US-20180225254-A1 OR WO-2010057775-A3 OR US-20180121183-A1 OR US-20190129700-A1 OR US-10628145-B2 OR US-9342346-B2 OR US-20170083478-A1 OR CN-101739113-A OR US-20160239323-A1 OR US-20140237607-A1 OR US-10235341-B2 OR EP-3177994-A1 OR	(US-PGPUB; USPAT; USOCR; FIT (AU, AP, AT, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, OA, RU, SU, WO, MC, MD, MY, NL, NO, NZ, PH, PL, PT, RO, RS, SE, SG, SI, SK, TH, TN, TR, TW, UA, VN))	ADJ	ON	ON	2022/10/13 10:49 AM

		WO-2016016695-A1 OR US-20160026489-A1 OR US-9043624-B2).did.					
L12	0	17/657981.app.	(US-PGPUB; USPAT; USOCR; FIT (AU, AP, AT, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, OA, RU, SU, WO, MC, MD, MY, NL, NO, NZ, PH, PL, PT, RO, RS, SE, SG, SI, SK, TH, TN, TR, TW, UA, VN); FPRS; EPO; JPO; DERWENT; IBM_TDB)	ADJ	ON	ON	2022/10/24 08:25 AM
L13	315067	(G06F9/45558 OR G06F9/5072 OR H04L63/1433 OR G06F2009/4557 OR G06F2009/45595 OR H04L63/14$2).cpc.	(US-PGPUB; USPAT; USOCR; FIT (AU, AP, AT, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, OA, RU, SU, WO, MC, MD, MY, NL, NO, NZ, PH, PL, PT, RO, RS, SE, SG, SI, SK, TH, TN, TR, TW, UA, VN); FPRS; EPO; JPO; DERWENT; IBM_TDB)	ADJ	ON	ON	2022/10/24 08:54 AM
L14	1977	L13 AND (risk OR threat OR vulnerab$5) WITH (identify$3 OR identification OR determin$3 OR determination) WITH cloud	(US-PGPUB; USPAT; USOCR; FIT (AU, AP, AT, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, OA, RU, SU, WO, MC, MD, MY, NL, NO, NZ, PH, PL, PT, RO, RS, SE, SG, SI, SK, TH, TN, TR, TW, UA, VN); FPRS; EPO; JPO; DERWENT; IBM_TDB)	ADJ	ON	ON	2022/10/24 08:55 AM
L15	383	L13 AND (risk OR threat OR vulnerab$5) WITH (identify$3 OR identification OR determin$3 OR	(US-PGPUB; USPAT; USOCR; FIT (AU, AP, AT, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA,	ADJ	ON	ON	2022/10/24 08:56 AM

		determination) WITH cloud WITH (resource OR asset)	EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, OA, RU, SU, WO, MC, MD, MY, NL, NO, NZ, PH, PL, PT, RO, RS, SE, SG, SI, SK, TH, TN, TR, TW, UA, VN); FPRS; EPO; JPO; DERWENT; IBM_TDB)				
L16	163	L13 AND (risk OR threat OR vulnerab$5) WITH (identify$3 OR identification OR determin$3 OR determination) WITH cloud WITH (resource OR asset) WITH network	(US-PGPUB; USPAT; USOCR; FIT (AU, AP, AT, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, OA, RU, SU, WO, MC, MD, MY, NL, NO, NZ, PH, PL, PT, RO, RS, SE, SG, SI, SK, TH, TN, TR, TW, UA, VN); FPRS; EPO; JPO; DERWENT; IBM_TDB)	ADJ	ON	ON	2022/10/24 08:59 AM
L17	848	L13 AND (forward OR backward) WITH (analysis OR analyz$3 OR examin$3 OR examination OR evaluat$3 OR test$3) WITH (risk OR threat OR vulnerab$5 OR attack OR mal$1ware OR spy$1ware OR ad$1ware OR trojan OR virus OR worm)	(US-PGPUB; USPAT; USOCR; FIT (AU, AP, AT, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, OA, RU, SU, WO, MC, MD, MY, NL, NO, NZ, PH, PL, PT, RO, RS, SE, SG, SI, SK, TH, TN, TR, TW, UA, VN); FPRS; EPO; JPO; DERWENT; IBM_TDB)	ADJ	ON	ON	2022/10/25 12:23 PM
L18	100	L13 AND (forward OR backward) near2 (analysis OR analyz$3 OR examin$3 OR examination OR evaluat$3 OR test$3) WITH (risk OR threat OR vulnerab$5 OR attack OR mal$1ware OR spy$1ware OR ad$1ware OR trojan OR virus OR worm)	(US-PGPUB; USPAT; USOCR; FIT (AU, AP, AT, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, OA, RU, SU, WO, MC, MD, MY, NL, NO, NZ, PH, PL, PT, RO, RS, SE, SG, SI, SK, TH, TN, TR, TW, UA, VN); FPRS; EPO;	ADJ	ON	ON	2022/10/25 12:23 PM

			JPO; DERWENT; IBM_TDB)				
L19	0	L13 AND (forward OR backward) near2 (analysis OR analyz$3 OR examin$3 OR examination OR evaluat$3 OR test$3) WITH (risk OR threat OR vulnerab$5 OR attack OR mal$1ware OR spy$1ware OR ad$1ware OR trojan OR virus OR worm) WITH vector	(US-PGPUB; USPAT; USOCR; FIT (AU, AP, AT, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, OA, RU, SU, WO, MC, MD, MY, NL, NO, NZ, PH, PL, PT, RO, RS, SE, SG, SI, SK, TH, TN, TR, TW, UA, VN); FPRS; EPO; JPO; DERWENT; IBM_TDB)	ADJ	ON	ON	2022/10/25 12:25 PM
L20	31	L13 AND (forward OR backward) WITH (analysis OR analyz$3 OR examin$3 OR examination OR evaluat$3 OR test$3) WITH (risk OR threat OR vulnerab$5 OR attack OR mal$1ware OR spy$1ware OR ad$1ware OR trojan OR virus OR worm) WITH vector	(US-PGPUB; USPAT; USOCR; FIT (AU, AP, AT, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, OA, RU, SU, WO, MC, MD, MY, NL, NO, NZ, PH, PL, PT, RO, RS, SE, SG, SI, SK, TH, TN, TR, TW, UA, VN); FPRS; EPO; JPO; DERWENT; IBM_TDB)	ADJ	ON	ON	2022/10/25 12:26 PM
L21	5	L13 AND (forward OR backward) near3 (analysis OR analyz$3 OR examin$3 OR examination OR evaluat$3 OR test$3) WITH (risk OR threat OR vulnerab$5 OR attack OR mal$1ware OR spy$1ware OR ad$1ware OR trojan OR virus OR worm) WITH vector	(US-PGPUB; USPAT; USOCR; FIT (AP, AT, AU, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, MC, MD, MY, NL, NO, NZ, OA, PH, PL, PT, RO, RS, RU, SE, SG, SI, SK, SU, TH, TN, TR, TW, UA, VN, WO); FPRS; EPO; JPO; DERWENT; IBM_TDB)	ADJ	ON	ON	2022/10/25 12:46 PM
L22	5	L13 AND (forward OR backward) near3 (analysis OR analyz$3 OR examin$3 OR examination OR evaluat$3 OR test$3) WITH (risk OR threat	(US-PGPUB; USPAT; USOCR; FIT (AP, AT, AU, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS,	ADJ	ON	ON	2022/10/25 12:47 PM

		OR vulnerab$5 OR attack OR mal$1ware OR spy$1ware OR ad$1ware OR trojan OR virus OR worm) SAME vector	IT, JP, KR, LT, LU, LV, MA, MC, MD, MY, NL, NO, NZ, OA, PH, PL, PT, RO, RS, RU, SE, SG, SI, SK, SU, TH, TN, TR, TW, UA, VN, WO); FPRS; EPO; JPO; DERWENT; IBM_TDB)				
L23	5	L13 AND (forward OR backward) near3 (analysis OR analyz$3 OR examin$3 OR examination OR evaluat$3 OR test$3) SAME (risk OR threat OR vulnerab$5 OR attack OR mal$1ware OR spy$1ware OR ad$1ware OR trojan OR virus OR worm) WITH vector	(US-PGPUB; USPAT; USOCR; FIT (AP, AT, AU, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, MC, MD, MY, NL, NO, NZ, OA, PH, PL, PT, RO, RS, RU, SE, SG, SI, SK, SU, TH, TN, TR, TW, UA, VN, WO); FPRS; EPO; JPO; DERWENT; IBM_TDB)	ADJ	ON	ON	2022/10/25 12:47 PM
L24	71	(forward OR backward) near3 (analysis OR analyz$3 OR examin$3 OR examination OR evaluat$3 OR test$3) SAME (risk OR threat OR vulnerab$5 OR attack OR mal$1ware OR spy$1ware OR ad$1ware OR trojan OR virus OR worm) WITH vector	(US-PGPUB; USPAT; USOCR; FIT (AP, AT, AU, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, MC, MD, MY, NL, NO, NZ, OA, PH, PL, PT, RO, RS, RU, SE, SG, SI, SK, SU, TH, TN, TR, TW, UA, VN, WO); FPRS; EPO; JPO; DERWENT; IBM_TDB)	ADJ	ON	ON	2022/10/25 12:48 PM
L25	0	L13 AND (up$1stream OR down$1stream) near3 (analysis OR analyz$3 OR examin$3 OR examination OR evaluat$3 OR test$3) SAME (risk OR threat OR vulnerab$5 OR attack OR mal$1ware OR spy$1ware OR ad$1ware OR trojan OR virus OR worm) WITH vector	(US-PGPUB; USPAT; USOCR; FIT (AP, AT, AU, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, MC, MD, MY, NL, NO, NZ, OA, PH, PL, PT, RO, RS, RU, SE, SG, SI, SK, SU, TH, TN, TR, TW, UA, VN, WO); FPRS; EPO; JPO; DERWENT; IBM_TDB)	ADJ	ON	ON	2022/10/25 12:51 PM

L26	116	L13 AND (up$1stream OR down$1stream) near3 (analysis OR analyz$3 OR examin$3 OR examination OR evaluat$3 OR test$3) SAME (risk OR threat OR vulnerab$5 OR attack OR mal$1ware OR spy$1ware OR ad$1ware OR trojan OR virus OR worm)	(US-PGPUB; USPAT; USOCR; FIT (AP, AT, AU, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, MC, MD, MY, NL, NO, NZ, OA, PH, PL, PT, RO, RS, RU, SE, SG, SI, SK, SU, TH, TN, TR, TW, UA, VN, WO); FPRS; EPO; JPO; DERWENT; IBM_TDB)	ADJ	ON	ON	2022/10/25 12:51 PM
L27	83	L13 AND (up$1stream OR down$1stream) near3 (analysis OR analyz$3 OR examin$3 OR examination OR evaluat$3 OR test$3) WITH (risk OR threat OR vulnerab$5 OR attack OR mal$1ware OR spy$1ware OR ad$1ware OR trojan OR virus OR worm)	(US-PGPUB; USPAT; USOCR; FIT (AP, AT, AU, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, MC, MD, MY, NL, NO, NZ, OA, PH, PL, PT, RO, RS, RU, SE, SG, SI, SK, SU, TH, TN, TR, TW, UA, VN, WO); FPRS; EPO; JPO; DERWENT; IBM_TDB)	ADJ	ON	ON	2022/10/25 12:51 PM
L28	0	L13 AND (up$1stream) near3 (analysis OR analyz$3 OR examin$3 OR examination OR evaluat$3 OR test$3) WITH (risk OR threat OR vulnerab$5 OR attack OR mal$1ware OR spy$1ware OR ad$1ware OR trojan OR virus OR worm) WITH vector	(US-PGPUB; USPAT; USOCR; FIT (AP, AT, AU, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, MC, MD, MY, NL, NO, NZ, OA, PH, PL, PT, RO, RS, RU, SE, SG, SI, SK, SU, TH, TN, TR, TW, UA, VN, WO); FPRS; EPO; JPO; DERWENT; IBM_TDB)	ADJ	ON	ON	2022/10/25 12:56 PM
L29	0	L13 AND (down$1stream) near3 (analysis OR analyz$3 OR examin$3 OR examination OR evaluat$3 OR test$3) WITH (risk OR threat OR vulnerab$5 OR attack OR mal$1ware	(US-PGPUB; USPAT; USOCR; FIT (AP, AT, AU, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, MC, MD, MY, NL,	ADJ	ON	ON	2022/10/25 12:58 PM

		OR spy$1ware OR ad$1ware OR trojan OR virus OR worm) WITH lateral	NO, NZ, OA, PH, PL, PT, RO, RS, RU, SE, SG, SI, SK, SU, TH, TN, TR, TW, UA, VN, WO); FPRS; EPO; JPO; DERWENT; IBM_TDB)				
L30	0	L13 AND (down$1stream) near3 (analysis OR analyz$3 OR examin$3 OR examination OR evaluat$3 OR test$3) WITH (risk OR threat OR vulnerab$5 OR attack OR mal$1ware OR spy$1ware OR ad$1ware OR trojan OR virus OR worm) SAME lateral	(US-PGPUB; USPAT; USOCR; FIT (AP, AT, AU, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, MC, MD, MY, NL, NO, NZ, OA, PH, PL, PT, RO, RS, RU, SE, SG, SI, SK, SU, TH, TN, TR, TW, UA, VN, WO); FPRS; EPO; JPO; DERWENT; IBM_TDB)	ADJ	ON	ON	2022/10/25 12:58 PM
L31	0	L13 AND (down$1stream) near3 (analysis OR analyz$3 OR examin$3 OR examination OR evaluat$3 OR test$3) SAME (risk OR threat OR vulnerab$5 OR attack OR mal$1ware OR spy$1ware OR ad$1ware OR trojan OR virus OR worm) SAME lateral	(US-PGPUB; USPAT; USOCR; FIT (AP, AT, AU, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, MC, MD, MY, NL, NO, NZ, OA, PH, PL, PT, RO, RS, RU, SE, SG, SI, SK, SU, TH, TN, TR, TW, UA, VN, WO); FPRS; EPO; JPO; DERWENT; IBM_TDB)	ADJ	ON	ON	2022/10/25 12:58 PM
L32	3	L13 AND (up$1stream OR down$1stream OR forward OR backward) WITH path WITH entry WITH (display$3 OR visual$7)	(US-PGPUB; USPAT; USOCR; FIT (AP, AT, AU, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, MC, MD, MY, NL, NO, NZ, OA, PH, PL, PT, RO, RS, RU, SE, SG, SI, SK, SU, TH, TN, TR, TW, UA, VN, WO); FPRS; EPO; JPO; DERWENT; IBM_TDB)	ADJ	ON	ON	2022/10/25 01:01 PM
L33	45	L13 AND (up$1stream OR down$1stream OR	(US-PGPUB; USPAT; USOCR; FIT (AP, AT,	ADJ	ON	ON	2022/10/25 01:20 PM

		forward OR backward) WITH path WITH (display$3 OR visual$7)	AU, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, MC, MD, MY, NL, NO, NZ, OA, PH, PL, PT, RO, RS, RU, SE, SG, SI, SK, SU, TH, TN, TR, TW, UA, VN, WO); FPRS; EPO; JPO; DERWENT; IBM_TDB)				
L35	1571	L13 AND (up$1stream OR down$1stream OR forward OR backward) WITH path AND cloud	(US-PGPUB; USPAT; USOCR; FIT (AP, AT, AU, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, MC, MD, MY, NL, NO, NZ, OA, PH, PL, PT, RO, RS, RU, SE, SG, SI, SK, SU, TH, TN, TR, TW, UA, VN, WO); FPRS; EPO; JPO; DERWENT; IBM_TDB)	ADJ	ON	ON	2022/10/25 01:24 PM
L36	130	L13 AND (up$1stream OR down$1stream OR forward OR backward) WITH path WITH (risk OR threat OR vulnerab$5 OR attack OR mal$1ware OR spy$1ware OR ad$1ware OR trojan OR virus OR worm) AND cloud	(US-PGPUB; USPAT; USOCR; FIT (AP, AT, AU, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, MC, MD, MY, NL, NO, NZ, OA, PH, PL, PT, RO, RS, RU, SE, SG, SI, SK, SU, TH, TN, TR, TW, UA, VN, WO); FPRS; EPO; JPO; DERWENT; IBM_TDB)	ADJ	ON	ON	2022/10/25 01:24 PM
L37	17	((("SHUA") near3 ("Avi"))).INV.	(US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT)	ADJ	ON	ON	2022/11/23 07:54 AM
L38	13	L1 AND (analy$3 OR examin$3 OR examination OR inspect$3) WITH (asset OR resource) WITH (risk OR threat OR vulnerab$5 OR attack)	(US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT)	ADJ	ON	ON	2022/11/23 07:54 AM

L39	7	L1 AND (analy$3 OR examin$3 OR examination OR inspect$3) WITH (asset OR resource) WITH (risk OR threat OR vulnerab$5 OR attack) WITH back$1ward	(US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT)	ADJ	ON	ON	2022/11/23 07:54 AM
L40	2	((("Orca") near3 ("Security"))).AS,AANM.	(USPAT)	ADJ	ON	ON	2022/11/23 07:54 AM
L41	38	("0002014" OR "10152487" OR "10868825" OR "11036856" OR "11075923" OR "20130262801" OR "20160330237" OR "20180205755" OR "20180367515" OR "20190250857" OR "20190327272" OR "20200097662" OR "20200220887" OR "20200228560" OR "20200244692" OR "20200344256" OR "20200389483" OR "20200396254" OR "20210099478" OR "20210342297" OR "20210409438" OR "20220000021" OR "20220000191" OR "20220001241" OR "20220100869" OR "20220141247" OR "20220182294" OR "20220239681" OR "7013395" OR "7143143" OR "8272061" OR "9294498" OR "9356942" OR "9413774" OR "9569201" OR "9686296" OR "9749349" OR "9794289").pn.	(US-PGPUB; USPAT)	ADJ	ON	ON	2022/11/23 07:54 AM
L42	2	("20190281115" OR "20200137097").pn.	(US-PGPUB; USPAT)	ADJ	ON	ON	2022/11/23 07:54 AM
L43	320386	(G06F9/45558 OR G06F9/5072 OR H04L63/1433 OR G06F2009/4557 OR G06F2009/45595 OR H04L63/14$2).cpc.	(US-PGPUB; USPAT; USOCR; FIT (AU, AP, AT, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS,	ADJ	ON	ON	2022/11/23 07:54 AM

			IT, JP, KR, LT, LU, LV, MA, OA, RU, SU, WO, MC, MD, MY, NL, NO, NZ, PH, PL, PT, RO, RS, SE, SG, SI, SK, TH, TN, TR, TW, UA, VN); FPRS; EPO; JPO; DERWENT; IBM_TDB)				
L44	1757	L7 AND (asset OR resource) WITH (risk OR threat OR vulnerab$5) WITH cloud	(US-PGPUB; USPAT; USOCR; FIT (AU, AP, AT, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, OA, RU, SU, WO, MC, MD, MY, NL, NO, NZ, PH, PL, PT, RO, RS, SE, SG, SI, SK, TH, TN, TR, TW, UA, VN); FPRS; EPO; JPO; DERWENT; IBM_TDB)	ADJ	ON	ON	2022/11/23 07:54 AM
L45	266	L7 AND (analy$3 OR examin$3 OR examination OR inspect$3) WITH (asset OR resource) WITH (risk OR threat OR vulnerab$5) WITH cloud	(US-PGPUB; USPAT; USOCR; FIT (AU, AP, AT, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, OA, RU, SU, WO, MC, MD, MY, NL, NO, NZ, PH, PL, PT, RO, RS, SE, SG, SI, SK, TH, TN, TR, TW, UA, VN); FPRS; EPO; JPO; DERWENT; IBM_TDB)	ADJ	ON	ON	2022/11/23 07:54 AM
L46	45	(Artificial Intelligence) Similar to: 17/657,981 with 0 CPC Selections and 0 Text Selections Text: (CN-106663022-A OR US-11099872-B2 OR US-20190073237-A1 OR US-10120706-B2 OR US-8621030-B2 OR US-20080080514-A1 OR US-9088590-B2 OR US-20140207906-A1 OR US-9342346-B2 OR EP-3177994-A1 OR	(US-PGPUB; USPAT; USOCR; FIT (AU, AP, AT, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, OA, RU, SU, WO, MC, MD, MY, NL, NO, NZ, PH, PL, PT, RO, RS, SE, SG, SI, SK, TH, TN, TR, TW, UA, VN))	ADJ	ON	ON	2022/11/23 07:54 AM

		WO-2016016695-A1 OR US-20160026489-A1 OR CN-104035868-A OR CN-104035868-B OR WO-2015188504-A1 OR CN-106802823-A OR US-20170083478-A1 OR US-10235341-B2 OR CN-106802823-B OR US-20150254095-A1 OR US-20160267051-A1 OR CN-107329801-A OR US-9417903-B2 OR CN-107329801-B OR CN-111538561-A OR US-20160239323-A1 OR EP-4002139-A3 OR CN-111857960-A OR CN-109416661-A OR US-9367370-B2 OR US-20160055041-A1 OR WO-2018087906-A1 OR US-10705879-B2 OR US-20200192654-A1 OR US-10740088-B2 OR CN-109416661-B OR CA-2753228-A1 OR WO-2010111149-A2 OR US-20100250868-A1 OR EP-2411915-A2 OR CA-2753228-C OR US-9529636-B2 OR RU-2011139151-A OR RU-2569805-C2 OR US-20180365138-A1 OR KR-101661783-B1 OR EP-2411915-B1 OR US-20190129700-A1 OR US-10628145-B2 OR US-20180121183-A1).did.					
L47	46	(Artificial Intelligence) Similar to: 17/657,981 with 2 CPC Selections and 2 Text Selections Text: (US-20180270130-A1 OR US-20180365138-A1 OR US-9367370-B2 OR US-20160055041-A1 OR US-20190079896-A1 OR US-10657095-B2 OR US-20140207906-A1	(US-PGPUB; USPAT; USOCR; FIT (AU, AP, AT, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, OA, RU, SU, WO, MC, MD, MY, NL, NO, NZ, PH, PL, PT, RO, RS, SE, SG, SI, SK, TH, TN, TR, TW, UA, VN))	ADJ	ON	ON	2022/11/23 07:54 AM

		OR US-9088590-B2 OR US-10120706-B2 OR US-11099872-B2 OR CN-111538561-A OR US-20150254095-A1 OR US-9495212-B2 OR US-20150339161-A1 OR US-20150026287-A1 OR US-9348649-B2 OR US-8856337-B2 OR WO-2015188504-A1 OR US-20130046892-A1 OR CN-106688208-A OR CN-106688208-B OR JP-2013041569-A OR US-20080080514-A1 OR US-8621030-B2 OR US-20190073237-A1 OR JP-5860320-B2 OR US-20160267051-A1 OR CN-112000606-A OR US-10621138-B2 OR CN-106663022-A OR KR-20100027936-A OR US-20160205215-A1 OR EP-3198806-B1 OR EP-3198806-A1 OR WO-2016045055-A1 OR US-20180225254-A1 OR WO-2010057775-A3 OR US-20180121183-A1 OR US-20190129700-A1 OR US-10628145-B2 OR US-9342346-B2 OR US-20170083478-A1 OR CN-101739113-A OR US-20160239323-A1 OR US-20140237607-A1 OR US-10235341-B2 OR EP-3177994-A1 OR WO-2016016695-A1 OR US-20160026489-A1 OR US-9043624-B2).did.				
L48	1	17/657981.app.	(US-PGPUB; USPAT; USOCR; FIT (AU, AP, AT, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, OA, RU, SU, WO, MC, MD, MY, NL, NO, NZ, PH, PL, PT, RO, RS, SE, SG, SI, SK,	ADJ	ON	ON	2022/11/23 07:54 AM

			TH, TN, TR, TW, UA, VN); FPRS; EPO; JPO; DERWENT; IBM_TDB)				
L49	320386	(G06F9/45558 OR G06F9/5072 OR H04L63/1433 OR G06F2009/4557 OR G06F2009/45595 OR H04L63/14$2).cpc.	(US-PGPUB; USPAT; USOCR; FIT (AU, AP, AT, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, OA, RU, SU, WO, MC, MD, MY, NL, NO, NZ, PH, PL, PT, RO, RS, SE, SG, SI, SK, TH, TN, TR, TW, UA, VN); FPRS; EPO; JPO; DERWENT; IBM_TDB)	ADJ	ON	ON	2022/11/23 07:54 AM
L50	2022	L13 AND (risk OR threat OR vulnerab$5) WITH (identify$3 OR identification OR determin$3 OR determination) WITH cloud	(US-PGPUB; USPAT; USOCR; FIT (AU, AP, AT, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, OA, RU, SU, WO, MC, MD, MY, NL, NO, NZ, PH, PL, PT, RO, RS, SE, SG, SI, SK, TH, TN, TR, TW, UA, VN); FPRS; EPO; JPO; DERWENT; IBM_TDB)	ADJ	ON	ON	2022/11/23 07:54 AM
L51	395	L13 AND (risk OR threat OR vulnerab$5) WITH (identify$3 OR identification OR determin$3 OR determination) WITH cloud WITH (resource OR asset)	(US-PGPUB; USPAT; USOCR; FIT (AU, AP, AT, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, OA, RU, SU, WO, MC, MD, MY, NL, NO, NZ, PH, PL, PT, RO, RS, SE, SG, SI, SK, TH, TN, TR, TW, UA, VN); FPRS; EPO; JPO; DERWENT; IBM_TDB)	ADJ	ON	ON	2022/11/23 07:54 AM
L52	173	L13 AND (risk OR threat OR vulnerab$5) WITH (identify$3 OR identification OR determin$3 OR	(US-PGPUB; USPAT; USOCR; FIT (AU, AP, AT, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA,	ADJ	ON	ON	2022/11/23 07:54 AM

		determination) WITH cloud WITH (resource OR asset) WITH network	EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, OA, RU, SU, WO, MC, MD, MY, NL, NO, NZ, PH, PL, PT, RO, RS, SE, SG, SI, SK, TH, TN, TR, TW, UA, VN); FPRS; EPO; JPO; DERWENT; IBM_TDB)				
L53	867	L13 AND (forward OR backward) WITH (analysis OR analyz$3 OR examin$3 OR examination OR evaluat$3 OR test$3) WITH (risk OR threat OR vulnerab$5 OR attack OR mal$1ware OR spy$1ware OR ad$1ware OR trojan OR virus OR worm)	(US-PGPUB; USPAT; USOCR; FIT (AU, AP, AT, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, OA, RU, SU, WO, MC, MD, MY, NL, NO, NZ, PH, PL, PT, RO, RS, SE, SG, SI, SK, TH, TN, TR, TW, UA, VN); FPRS; EPO; JPO; DERWENT; IBM_TDB)	ADJ	ON	ON	2022/11/23 07:54 AM
L54	109	L13 AND (forward OR backward) near2 (analysis OR analyz$3 OR examin$3 OR examination OR evaluat$3 OR test$3) WITH (risk OR threat OR vulnerab$5 OR attack OR mal$1ware OR spy$1ware OR ad$1ware OR trojan OR virus OR worm)	(US-PGPUB; USPAT; USOCR; FIT (AU, AP, AT, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, OA, RU, SU, WO, MC, MD, MY, NL, NO, NZ, PH, PL, PT, RO, RS, SE, SG, SI, SK, TH, TN, TR, TW, UA, VN); FPRS; EPO; JPO; DERWENT; IBM_TDB)	ADJ	ON	ON	2022/11/23 07:54 AM
L55	8	L13 AND (forward OR backward) near2 (analysis OR analyz$3 OR examin$3 OR examination OR evaluat$3 OR test$3) WITH (risk OR threat OR vulnerab$5 OR attack OR mal$1ware OR spy$1ware OR ad$1ware OR trojan OR virus OR worm) WITH vector	(US-PGPUB; USPAT; USOCR; FIT (AU, AP, AT, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, OA, RU, SU, WO, MC, MD, MY, NL, NO, NZ, PH, PL, PT, RO, RS, SE, SG, SI, SK, TH, TN, TR, TW, UA, VN); FPRS; EPO;	ADJ	ON	ON	2022/11/23 07:54 AM

			JPO; DERWENT; IBM_TDB)				
L56	39	L13 AND (forward OR backward) WITH (analysis OR analyz$3 OR examin$3 OR examination OR evaluat$3 OR test$3) WITH (risk OR threat OR vulnerab$5 OR attack OR mal$1ware OR spy$1ware OR ad$1ware OR trojan OR virus OR worm) WITH vector	(US-PGPUB; USPAT; USOCR; FIT (AU, AP, AT, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, OA, RU, SU, WO, MC, MD, MY, NL, NO, NZ, PH, PL, PT, RO, RS, SE, SG, SI, SK, TH, TN, TR, TW, UA, VN); FPRS; EPO; JPO; DERWENT; IBM_TDB)	ADJ	ON	ON	2022/11/23 07:54 AM
L57	13	L13 AND (forward OR backward) near3 (analysis OR analyz$3 OR examin$3 OR examination OR evaluat$3 OR test$3) WITH (risk OR threat OR vulnerab$5 OR attack OR mal$1ware OR spy$1ware OR ad$1ware OR trojan OR virus OR worm) WITH vector	(US-PGPUB; USPAT; USOCR; FIT (AP, AT, AU, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, MC, MD, MY, NL, NO, NZ, OA, PH, PL, PT, RO, RS, RU, SE, SG, SI, SK, SU, TH, TN, TR, TW, UA, VN, WO); FPRS; EPO; JPO; DERWENT; IBM_TDB)	ADJ	ON	ON	2022/11/23 07:54 AM
L58	13	L13 AND (forward OR backward) near3 (analysis OR analyz$3 OR examin$3 OR examination OR evaluat$3 OR test$3) WITH (risk OR threat OR vulnerab$5 OR attack OR mal$1ware OR spy$1ware OR ad$1ware OR trojan OR virus OR worm) SAME vector	(US-PGPUB; USPAT; USOCR; FIT (AP, AT, AU, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, MC, MD, MY, NL, NO, NZ, OA, PH, PL, PT, RO, RS, RU, SE, SG, SI, SK, SU, TH, TN, TR, TW, UA, VN, WO); FPRS; EPO; JPO; DERWENT; IBM_TDB)	ADJ	ON	ON	2022/11/23 07:54 AM
L59	13	L13 AND (forward OR backward) near3 (analysis OR analyz$3 OR examin$3 OR examination OR evaluat$3 OR test$3) SAME (risk OR threat	(US-PGPUB; USPAT; USOCR; FIT (AP, AT, AU, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS,	ADJ	ON	ON	2022/11/23 07:54 AM

		OR vulnerab$5 OR attack OR mal$1ware OR spy$1ware OR ad$1ware OR trojan OR virus OR worm) WITH vector	IT, JP, KR, LT, LU, LV, MA, MC, MD, MY, NL, NO, NZ, OA, PH, PL, PT, RO, RS, RU, SE, SG, SI, SK, SU, TH, TN, TR, TW, UA, VN, WO); FPRS; EPO; JPO; DERWENT; IBM_TDB)				
L60	83	(forward OR backward) near3 (analysis OR analyz$3 OR examin$3 OR examination OR evaluat$3 OR test$3) SAME (risk OR threat OR vulnerab$5 OR attack OR mal$1ware OR spy$1ware OR ad$1ware OR trojan OR virus OR worm) WITH vector	(US-PGPUB; USPAT; USOCR; FIT (AP, AT, AU, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, MC, MD, MY, NL, NO, NZ, OA, PH, PL, PT, RO, RS, RU, SE, SG, SI, SK, SU, TH, TN, TR, TW, UA, VN, WO); FPRS; EPO; JPO; DERWENT; IBM_TDB)	ADJ	ON	ON	2022/11/23 07:54 AM
L61	0	L13 AND (up$1stream OR down$1stream) near3 (analysis OR analyz$3 OR examin$3 OR examination OR evaluat$3 OR test$3) SAME (risk OR threat OR vulnerab$5 OR attack OR mal$1ware OR spy$1ware OR ad$1ware OR trojan OR virus OR worm) WITH vector	(US-PGPUB; USPAT; USOCR; FIT (AP, AT, AU, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, MC, MD, MY, NL, NO, NZ, OA, PH, PL, PT, RO, RS, RU, SE, SG, SI, SK, SU, TH, TN, TR, TW, UA, VN, WO); FPRS; EPO; JPO; DERWENT; IBM_TDB)	ADJ	ON	ON	2022/11/23 07:54 AM
L62	117	L13 AND (up$1stream OR down$1stream) near3 (analysis OR analyz$3 OR examin$3 OR examination OR evaluat$3 OR test$3) SAME (risk OR threat OR vulnerab$5 OR attack OR mal$1ware OR spy$1ware OR ad$1ware OR trojan OR virus OR worm)	(US-PGPUB; USPAT; USOCR; FIT (AP, AT, AU, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, MC, MD, MY, NL, NO, NZ, OA, PH, PL, PT, RO, RS, RU, SE, SG, SI, SK, SU, TH, TN, TR, TW, UA, VN, WO); FPRS; EPO; JPO; DERWENT; IBM_TDB)	ADJ	ON	ON	2022/11/23 07:54 AM

L63	84	L13 AND (up$1stream OR down$1stream) near3 (analysis OR analyz$3 OR examin$3 OR examination OR evaluat$3 OR test$3) WITH (risk OR threat OR vulnerab$5 OR attack OR mal$1ware OR spy$1ware OR ad$1ware OR trojan OR virus OR worm)	(US-PGPUB; USPAT; USOCR; FIT (AP, AT, AU, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, MC, MD, MY, NL, NO, NZ, OA, PH, PL, PT, RO, RS, RU, SE, SG, SI, SK, SU, TH, TN, TR, TW, UA, VN, WO); FPRS; EPO; JPO; DERWENT; IBM_TDB)	ADJ	ON	ON	2022/11/23 07:54 AM
L64	0	L13 AND (up$1stream) near3 (analysis OR analyz$3 OR examin$3 OR examination OR evaluat$3 OR test$3) WITH (risk OR threat OR vulnerab$5 OR attack OR mal$1ware OR spy$1ware OR ad$1ware OR trojan OR virus OR worm) WITH vector	(US-PGPUB; USPAT; USOCR; FIT (AP, AT, AU, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, MC, MD, MY, NL, NO, NZ, OA, PH, PL, PT, RO, RS, RU, SE, SG, SI, SK, SU, TH, TN, TR, TW, UA, VN, WO); FPRS; EPO; JPO; DERWENT; IBM_TDB)	ADJ	ON	ON	2022/11/23 07:54 AM
L65	0	L13 AND (down$1stream) near3 (analysis OR analyz$3 OR examin$3 OR examination OR evaluat$3 OR test$3) WITH (risk OR threat OR vulnerab$5 OR attack OR mal$1ware OR spy$1ware OR ad$1ware OR trojan OR virus OR worm) WITH lateral	(US-PGPUB; USPAT; USOCR; FIT (AP, AT, AU, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, MC, MD, MY, NL, NO, NZ, OA, PH, PL, PT, RO, RS, RU, SE, SG, SI, SK, SU, TH, TN, TR, TW, UA, VN, WO); FPRS; EPO; JPO; DERWENT; IBM_TDB)	ADJ	ON	ON	2022/11/23 07:54 AM
L66	0	L13 AND (down$1stream) near3 (analysis OR analyz$3 OR examin$3 OR examination OR evaluat$3 OR test$3) WITH (risk OR threat OR vulnerab$5 OR attack OR mal$1ware	(US-PGPUB; USPAT; USOCR; FIT (AP, AT, AU, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, MC, MD, MY, NL,	ADJ	ON	ON	2022/11/23 07:54 AM

		OR spy$1ware OR ad$1ware OR trojan OR virus OR worm) SAME lateral	NO, NZ, OA, PH, PL, PT, RO, RS, RU, SE, SG, SI, SK, SU, TH, TN, TR, TW, UA, VN, WO); FPRS; EPO; JPO; DERWENT; IBM_TDB)				
L67	0	L13 AND (down$1stream) near3 (analysis OR analyz$3 OR examin$3 OR examination OR evaluat$3 OR test$3) SAME (risk OR threat OR vulnerab$5 OR attack OR mal$1ware OR spy$1ware OR ad$1ware OR trojan OR virus OR worm) SAME lateral	(US-PGPUB; USPAT; USOCR; FIT (AP, AT, AU, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, MC, MD, MY, NL, NO, NZ, OA, PH, PL, PT, RO, RS, RU, SE, SG, SI, SK, SU, TH, TN, TR, TW, UA, VN, WO); FPRS; EPO; JPO; DERWENT; IBM_TDB)	ADJ	ON	ON	2022/11/23 07:54 AM
L68	11	L13 AND (up$1stream OR down$1stream OR forward OR backward) WITH path WITH entry WITH (display$3 OR visual$7)	(US-PGPUB; USPAT; USOCR; FIT (AP, AT, AU, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, MC, MD, MY, NL, NO, NZ, OA, PH, PL, PT, RO, RS, RU, SE, SG, SI, SK, SU, TH, TN, TR, TW, UA, VN, WO); FPRS; EPO; JPO; DERWENT; IBM_TDB)	ADJ	ON	ON	2022/11/23 07:54 AM
L69	53	L13 AND (up$1stream OR down$1stream OR forward OR backward) WITH path WITH (display$3 OR visual$7)	(US-PGPUB; USPAT; USOCR; FIT (AP, AT, AU, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, MC, MD, MY, NL, NO, NZ, OA, PH, PL, PT, RO, RS, RU, SE, SG, SI, SK, SU, TH, TN, TR, TW, UA, VN, WO); FPRS; EPO; JPO; DERWENT; IBM_TDB)	ADJ	ON	ON	2022/11/23 07:54 AM
L70	1593	L13 AND (up$1stream OR down$1stream OR	(US-PGPUB; USPAT; USOCR; FIT (AP, AT,	ADJ	ON	ON	2022/11/23 07:54 AM

		forward OR backward) WITH path AND cloud	AU, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, MC, MD, MY, NL, NO, NZ, OA, PH, PL, PT, RO, RS, RU, SE, SG, SI, SK, SU, TH, TN, TR, TW, UA, VN, WO); FPRS; EPO; JPO; DERWENT; IBM_TDB)				
L71	138	L13 AND (up$1stream OR down$1stream OR forward OR backward) WITH path WITH (risk OR threat OR vulnerab$5 OR attack OR mal$1ware OR spy$1ware OR ad$1ware OR trojan OR virus OR worm) AND cloud	(US-PGPUB; USPAT; USOCR; FIT (AP, AT, AU, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, MC, MD, MY, NL, NO, NZ, OA, PH, PL, PT, RO, RS, RU, SE, SG, SI, SK, SU, TH, TN, TR, TW, UA, VN, WO); FPRS; EPO; JPO; DERWENT; IBM_TDB)	ADJ	ON	ON	2022/11/23 07:54 AM
L72	4	("20170012998" OR "20180288087" OR "20210377287" OR "8621073").pn.	(US-PGPUB; USPAT)	ADJ	ON	ON	2022/11/23 07:55 AM
L73	1	("20190116199").pn.	(US-PGPUB; USPAT)	ADJ	ON	ON	2022/11/23 08:30 AM
L75	405	L13 AND (risk OR threat OR vulnerab$5) WITH (analyz$3 OR analysis OR examin$3 OR examination) WITH (back$ward OR reverse)	(US-PGPUB; USPAT; USOCR; FIT (AP, AT, AU, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, MC, MD, MY, NL, NO, NZ, OA, PH, PL, PT, RO, RS, RU, SE, SG, SI, SK, SU, TH, TN, TR, TW, UA, VN, WO); FPRS; EPO; JPO; DERWENT; IBM_TDB)	ADJ	ON	ON	2022/11/23 08:36 AM
L76	29801	L13 AND (risk OR threat OR vulnerab$5) WITH (analyz$3 OR analysis OR examin$3	(US-PGPUB; USPAT; USOCR; FIT (AP, AT, AU, BE, BG, BR, BY, CA, CH, CN, CS, CU,	ADJ	ON	ON	2022/11/23 08:37 AM

		OR examination)	CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, MC, MD, MY, NL, NO, NZ, OA, PH, PL, PT, RO, RS, RU, SE, SG, SI, SK, SU, TH, TN, TR, TW, UA, VN, WO); FPRS; EPO; JPO; DERWENT; IBM_TDB)				
L77	7	L13 AND (risk OR threat OR vulnerab$5) WITH lateral WITH (analyz$3 OR analysis OR examin$3 OR examination) WITH (back$ward OR reverse)	(US-PGPUB; USPAT; USOCR; FIT (AP, AT, AU, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, MC, MD, MY, NL, NO, NZ, OA, PH, PL, PT, RO, RS, RU, SE, SG, SI, SK, SU, TH, TN, TR, TW, UA, VN, WO); FPRS; EPO; JPO; DERWENT; IBM_TDB)	ADJ	ON	ON	2022/11/23 08:37 AM
L78	713	L13 AND (risk OR threat OR vulnerab$5) WITH (analyz$3 OR analysis OR examin$3 OR examination) WITH (back$ward OR reverse OR forward)	(US-PGPUB; USPAT; USOCR; FIT (AP, AT, AU, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, MC, MD, MY, NL, NO, NZ, OA, PH, PL, PT, RO, RS, RU, SE, SG, SI, SK, SU, TH, TN, TR, TW, UA, VN, WO); FPRS; EPO; JPO; DERWENT; IBM_TDB)	ADJ	ON	ON	2022/11/23 09:33 AM
L79	313	L13 AND (risk OR threat OR vulnerab$5) WITH (analyz$3 OR analysis OR examin$3 OR examination) WITH (back$ward OR reverse OR forward) AND cloud	(US-PGPUB; USPAT; USOCR; FIT (AP, AT, AU, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, MC, MD, MY, NL, NO, NZ, OA, PH, PL, PT, RO, RS, RU, SE, SG, SI, SK, SU, TH, TN, TR, TW, UA, VN,	ADJ	ON	ON	2022/11/23 09:34 AM

			WO); FPRS; EPO; JPO; DERWENT; IBM_TDB)				
L80	325	L13 AND (risk OR threat OR vulnerab$5 OR (attack WITH vector)) WITH (analyz$3 OR analysis OR examin$3 OR examination) WITH (back$ward OR reverse OR forward) AND cloud	(US-PGPUB; USPAT; USOCR; FIT (AP, AT, AU, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, MC, MD, MY, NL, NO, NZ, OA, PH, PL, PT, RO, RS, RU, SE, SG, SI, SK, SU, TH, TN, TR, TW, UA, VN, WO); FPRS; EPO; JPO; DERWENT; IBM_TDB)	ADJ	ON	ON	2022/11/23 09:34 AM
L81	29	L13 AND (risk OR threat OR vulnerab$5 OR (attack WITH vector)) WITH (analyz$3 OR analysis OR examin$3 OR examination) WITH (back$ward OR reverse OR forward) WITH (route OR path) AND cloud	(US-PGPUB; USPAT; USOCR; FIT (AP, AT, AU, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, MC, MD, MY, NL, NO, NZ, OA, PH, PL, PT, RO, RS, RU, SE, SG, SI, SK, SU, TH, TN, TR, TW, UA, VN, WO); FPRS; EPO; JPO; DERWENT; IBM_TDB)	ADJ	ON	ON	2022/11/23 09:34 AM
L82	38	(Artificial Intelligence) Similar to: 17/657,981 with 0 CPC Selections and 0 Text Selections Text: (CN-107688745-A OR CN-107688745-B OR US-20160026791-A1 OR EP-3195178-A1 OR WO-2016014567-A1 OR EP-3474148-A1 OR US-20200104510-A1 OR WO-2019212757-A1 OR EP-3765965-A1 OR US-10990685-B2 OR US-9990492-B2 OR EP-3195178-B1 OR US-9530007-B1 OR US-20170076102-A1 OR GB-2345996-A OR GB-2459629-A OR	(US-PGPUB; USPAT; USOCR; FIT (AP, AT, AU, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, MC, MD, MY, NL, NO, NZ, OA, PH, PL, PT, RO, RS, RU, SE, SG, SI, SK, SU, TH, TN, TR, TW, UA, VN, WO))	ADJ	ON	ON	2022/11/23 09:48 AM

		WO-2008103286-A3 OR WO-2018224948-A3 OR US-9449172-B2 OR US-20150242624-A1 OR EP-3474148-B1 OR DE-102014102623-A1 OR EP-3072079-A1 OR WO-2015077331-A1 OR CA-2930833-A1 OR US-20150143524-A1 OR US-20160275292-A1 OR EP-3072079-B1 OR US-20180247057-A1 OR US-10990667-B2 OR WO-2021034695-A1 OR EP-3991073-A1 OR EP-3788529-A1 OR US-20190340366-A1 OR JP-2022537079-A OR TW-202125224-A OR US-10878103-B2 OR EP-3635562-B1 OR CN-114258527-A OR DE-102014111022-A1 OR JP-7144642-B2 OR US-20210049031-A1 OR ES-2767049-T3 OR US-20170322828-A1 OR US-9419996-B2 OR CN-113094700-A OR US-20220269236-A1 OR BR-0215388-A OR US-11176011-B2 OR KR-20210099724-A).did.					
L83	38	(Artificial Intelligence) Similar to: 17/657,981 with 15 CPC Selections and 2 Text Selections Text: (CN-107688745-A OR CN-107688745-B OR US-20160026791-A1 OR EP-3195178-A1 OR WO-2016014567-A1 OR US-20200104510-A1 OR EP-3765965-A1 OR WO-2019212757-A1 OR US-10990685-B2 OR EP-3474148-A1 OR US-9449172-B2 OR US-20150242624-A1 OR US-9990492-B2 OR DE-102014102623-A1 OR EP-3195178-B1 OR	(US-PGPUB; USPAT; USOCR; FIT (AP, AT, AU, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, MC, MD, MY, NL, NO, NZ, OA, PH, PL, PT, RO, RS, RU, SE, SG, SI, SK, SU, TH, TN, TR, TW, UA, VN, WO))	ADJ	ON	ON	2022/11/23 09:49 AM

		GB-2345996-A OR WO-2018224948-A3 OR US-20170076102-A1 OR US-9530007-B1 OR GB-2459629-A OR WO-2008103286-A3 OR EP-3474148-B1 OR WO-2021034695-A1 OR EP-3991073-A1 OR JP-2022537079-A OR CN-114258527-A OR JP-7144642-B2 OR US-20210049031-A1 OR US-20180247057-A1 OR US-10990667-B2 OR EP-3072079-A1 OR WO-2015077331-A1 OR CA-2930833-A1 OR US-20150143524-A1 OR US-20160275292-A1 OR US-9419996-B2 OR EP-3788529-A1 OR US-20190340366-A1 OR US-20220269236-A1 OR EP-3072079-B1 OR ES-2767049-T3 OR TW-202125224-A OR US-20090049548-A1 OR DE-102014111022-A1 OR WO-2007049181-A1 OR EP-1943604-A1 OR EP-3635562-B1 OR CN-106295326-A OR US-20170322828-A1 OR KR-20210099724-A).did.					
L84	961115	(G06F9/45558 OR G06F9/5072 OR H04L63/1433 OR G06F2009/4557 OR G06F2009/45595 OR H04L63/14$2 OR G06F21/5$2 OR G06F21/78 OR H04L63/10 OR H04L67/10$2).cpc.	(US-PGPUB; USPAT; USOCR; FIT (AP, AT, AU, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, MC, MD, MY, NL, NO, NZ, OA, PH, PL, PT, RO, RS, RU, SE, SG, SI, SK, SU, TH, TN, TR, TW, UA, VN, WO); FPRS; EPO; JPO; DERWENT; IBM_TDB)	ADJ	ON	ON	2022/11/23 10:38 AM
L85	736	L84 AND (risk OR threat OR vulnerab$5) WITH (analyz$3 OR analysis OR examin$3	(US-PGPUB; USPAT; USOCR; FIT (AP, AT, AU, BE, BG, BR, BY, CA, CH, CN, CS, CU,	ADJ	ON	ON	2022/11/23 10:38 AM

		OR examination) WITH ((ack$ward OR reverse))	CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, MC, MD, MY, NL, NO, NZ, OA, PH, PL, PT, RO, RS, RU, SE, SG, SI, SK, SU, TH, TN, TR, TW, UA, VN, WO); FPRS; EPO; JPO; DERWENT; IBM_TDB)				
L86	1170	L84 AND (risk OR threat OR vulnerab$5) WITH (analyz$3 OR analysis OR examin$3 OR examination) WITH (back$ward OR reverse OR forward)	(US-PGPUB; USPAT; USOCR; FIT (AP, AT, AU, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, MC, MD, MY, NL, NO, NZ, OA, PH, PL, PT, RO, RS, RU, SE, SG, SI, SK, SU, TH, TN, TR, TW, UA, VN, WO); FPRS; EPO; JPO; DERWENT; IBM_TDB)	ADJ	ON	ON	2022/11/23 10:39 AM
L87	436	L84 AND (risk OR threat OR vulnerab$5 OR (attack WITH vector)) WITH (analyz$3 OR analysis OR examin$3 OR examination) WITH (back$ward OR reverse OR forward) AND cloud	(US-PGPUB; USPAT; USOCR; FIT (AP, AT, AU, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, MC, MD, MY, NL, NO, NZ, OA, PH, PL, PT, RO, RS, RU, SE, SG, SI, SK, SU, TH, TN, TR, TW, UA, VN, WO); FPRS; EPO; JPO; DERWENT; IBM_TDB)	ADJ	ON	ON	2022/11/23 10:39 AM
L88	50	L84 AND (risk OR threat OR vulnerab$5 OR (attack WITH vector)) WITH (analyz$3 OR analysis OR examin$3 OR examination) WITH (back$ward OR reverse OR forward) WITH (route OR path) AND cloud	(US-PGPUB; USPAT; USOCR; FIT (AP, AT, AU, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, MC, MD, MY, NL, NO, NZ, OA, PH, PL, PT, RO, RS, RU, SE, SG, SI, SK, SU, TH, TN, TR, TW, UA, VN,	ADJ	ON	ON	2022/11/23 10:39 AM

			WO); FPRS; EPO; JPO; DERWENT; IBM_TDB)				
L89	13	L84 AND (up$1stream OR down$1stream OR forward OR backward) WITH path WITH entry WITH (display$3 OR visual$7)	(US-PGPUB; USPAT; USOCR; FIT (AP, AT, AU, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, MC, MD, MY, NL, NO, NZ, OA, PH, PL, PT, RO, RS, RU, SE, SG, SI, SK, SU, TH, TN, TR, TW, UA, VN, WO); FPRS; EPO; JPO; DERWENT; IBM_TDB)	ADJ	ON	ON	2022/11/23 10:39 AM
L90	0	L84 AND (up$1stream OR down$1stream) near3 (analysis OR analyz$3 OR examin$3 OR examination OR evaluat$3 OR test$3) SAME (risk OR threat OR vulnerab$5 OR attack OR mal$1ware OR spy$1ware OR ad$1ware OR trojan OR virus OR worm) WITH vector	(US-PGPUB; USPAT; USOCR; FIT (AP, AT, AU, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, MC, MD, MY, NL, NO, NZ, OA, PH, PL, PT, RO, RS, RU, SE, SG, SI, SK, SU, TH, TN, TR, TW, UA, VN, WO); FPRS; EPO; JPO; DERWENT; IBM_TDB)	ADJ	ON	ON	2022/11/23 10:40 AM
L91	7	L84 AND (risk OR threat OR vulnerab$5) WITH lateral WITH (analyz$3 OR analysis OR examin$3 OR examination) WITH (back$ward OR reverse)	(US-PGPUB; USPAT; USOCR; FIT (AP, AT, AU, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA, EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, MC, MD, MY, NL, NO, NZ, OA, PH, PL, PT, RO, RS, RU, SE, SG, SI, SK, SU, TH, TN, TR, TW, UA, VN, WO); FPRS; EPO; JPO; DERWENT; IBM_TDB)	ADJ	ON	ON	2022/11/23 10:40 AM
L92	95	L84 AND (up$1stream OR down$1stream OR forward OR backward) WITH path WITH (display$3 OR visual$7)	(US-PGPUB; USPAT; USOCR; FIT (AP, AT, AU, BE, BG, BR, BY, CA, CH, CN, CS, CU, CZ, DD, DE, DK, EA,	ADJ	ON	ON	2022/11/23 10:40 AM

			EE, EP, ES, FI, FR, GB, HR, HU, ID, IE, IL, IS, IT, JP, KR, LT, LU, LV, MA, MC, MD, MY, NL, NO, NZ, OA, PH, PL, PT, RO, RS, RU, SE, SG, SI, SK, SU, TH, TN, TR, TW, UA, VN, WO); FPRS; EPO; JPO; DERWENT; IBM_TDB)				

PE2E SEARCH - Search History (Interference)

Ref #	Hits	Search Query	DBs	Default Operator	Plurals	British Equivalents	Time Stamp
N1	83366	(G06F9/45558 OR G06F9/5072 OR H04L63/1433 OR G06F2009/4557 OR G06F2009/45595 OR H04L63/14$2).cpc.	(US-PGPUB; USPAT)	ADJ	ON	ON	2022/11/23 09:36 AM
N2	2728	N1 AND vector.clm.	(US-PGPUB; USPAT)	ADJ	ON	ON	2022/11/23 09:36 AM
N3	294	N1 AND vector.clm. AND risk.clm.	(US-PGPUB; USPAT)	ADJ	ON	ON	2022/11/23 09:36 AM
N4	25	N1 AND vector.clm. AND risk.clm. AND path.clm.	(US-PGPUB; USPAT)	ADJ	ON	ON	2022/11/23 09:37 AM
N5	4	N1 AND vector.clm. AND risk.clm. AND path.clm. AND cloud.clm.	(US-PGPUB; USPAT)	ADJ	ON	ON	2022/11/23 09:37 AM
N6	0	N1 AND risk.clm AND analysis.clm.	(US-PGPUB; USPAT)	ADJ	ON	ON	2022/11/23 09:44 AM
N7	232410	L84	(US-PGPUB; USPAT)	ADJ	ON	ON	2022/11/23 10:41 AM
N8	5167	N7 AND vector.clm.	(US-PGPUB; USPAT)	ADJ	ON	ON	2022/11/23 10:41 AM
N9	30	N7 AND vector.clm. AND risk.clm. AND path.clm.	(US-PGPUB; USPAT)	ADJ	ON	ON	2022/11/23 10:41 AM
N10	5	N7 AND vector.clm. AND risk.clm. AND path.clm. AND cloud.clm.	(US-PGPUB; USPAT)	ADJ	ON	ON	2022/11/23 10:41 AM
N11	0	N7 AND risk.clm AND analysis.clm.	(US-PGPUB; USPAT)	ADJ	ON	ON	2022/11/23 10:41 AM

PE2E SEARCH - Search History (Similarity Search)

Ref #	Application Number	Text Selections	CPC Selections
L46	17/657,981	No selections	No selections

L47	17/657,981	(Specification) This summary is provided for the convenience of the reader to present a basic understanding of a few exemplary embodiments of this disclosure, and does not wholly define the breadth of the disclosure. This summary is not an extensive overview of all contemplated embodiments and is intended to neither identify key or critical elements of all embodiments nor to delineate the scope of any or all aspects. Its sole purpose is to present some features of one or more embodiments in a simplified form as a prelude to the more detailed description presented later. For convenience, the term "embodiments" may be used herein to refer to one or more features or combinations of features described herein. A description associated with an embodiment does not necessarily apply to all embodiments. [0014] Embodiments of the present disclosure may deliver instant-on, workload level visibility across cloud infrastructure assets without running opcodes on the target scanned workloads, helping organizations to: (1) Detect risks such as vulnerabilities, malware, misconfigurations, lateral movement risk, and unsecured sensitive data; (2) See cloud inventory at every layer infrastructure, operating system, applications, and data; (3) Discover and see previously missed assets. [0015] Embodiments of the present disclosure may further use an out-of-band process to reach cloud workloads through the runtime storage layer, combining this with information gathered from cloud provider APIs, including but not limited to K8S APIs. (Claims) A graphical user interface system for providing forward and backward facing attack vector visualizations, the system comprising:	(H04L63/1433) Vulnerability analysis (G06F2009/45595) Network integration; Enabling network access in virtual machine instances

		at least one processor configured to: identify assets in a cloud environment; identify risks associated with each of the identified asset; identify relationships between at least some of the identified assets, the relationships including at least one of a trust, a network connectivity, or a mechanism of network proxying; receive an identification of a specific asset under investigation; perform a forward analysis of the specific asset under investigation to identify at least one possible attack vector reaching the specific asset via a network outside the cloud environment; perform a backward analysis of the specific asset to identify at least one exposure risk to one or more assets that is in a downstream of the specific asset, wherein the at least one exposure risk includes an identification of an exposed asset, an entry point to the exposed asset, and a lateral movement risk associated with the exposed asset; and output a signal to cause on a display to present a presentation of forward and backward paths associated with the specific asset, thereby enabling visualization of a plurality of entry points and lateral movement risks associated with the plurality of entry points.	
L82	17/657,981	No selections	No selections
L83	17/657,981	(Abstract) Systems, methods, and computer-readable media are provided for securing cloud infrastructure, including a method comprising: establishing a trusted relationship between a source account in a cloud environment and a scanner account, using the established trust relationship, utilizing at least one cloud provider API to identify workloads in the source account, using the at least one cloud provider API to query a geographical location of at least one of the identified workloads, receiving an identification	(G06F21/577) Assessing vulnerabilities and evaluating computer system security (H04L63/1433) Vulnerability analysis (G06F9/45558) Hypervisor-specific management and integration aspects (G06F21/78) Protecting specific internal or peripheral components, in which the protection of a component leads to protection of the entire computer to assure secure storage of data (G06F21/568) Computer malware detection or handling, e.g. anti-virus arrangements eliminating virus, restoring damaged files

		of the geographic location, using the cloud provider APIs to access block storage volumes of the at least one workload, determining a file- system of the at least one workload, mounting the block storage volumes on a scanner based on the determined file-system, activating a scanner at the geographic location, reconstructing from the block storage volumes a state of the workload, and assessing the reconstructed state of the workload to extract insights. (Claims) graphical user interface system for providing forward and backward facing attack vector visualizations, the system comprising: at least one processor configured to: identify assets in a cloud environment; identify risks associated with each of the identified asset; identify relationships between at least some of the identified assets, the relationships including at least one of a trust, a network connectivity, or a mechanism of network proxying; receive an identification of a specific asset under investigation; perform a forward analysis of the specific asset under investigation to identify at least one possible attack vector reaching the specific asset via a network outside the cloud environment; perform a backward analysis of the specific asset to identify at least one exposure risk to one or more assets that is in a downstream of the specific asset, wherein the at least one exposure risk includes an identification of an exposed asset, an entry point to the exposed asset, and a lateral movement risk associated with the exposed asset; and output a signal to cause on a display to present a presentation of forward and backward paths associated with the specific asset, thereby enabling visualization of a plurality of entry points and lateral movement	(G06F21/554) Detecting local intrusion or implementing counter-measures involving event detection and direct action (G06F21/54) Monitoring users, programs or devices to maintain the integrity of platforms, e.g. of processors, firmware or operating systems during program execution, e.g. stack integrity ; Preventing unwanted data erasure; Buffer overflow by adding security routines or objects to programs (G06F21/57) Certifying or maintaining trusted computer platforms, e.g. secure boots or power-downs, version controls, system software checks, secure updates or assessing vulnerabilities (G06F2009/4557) Distribution of virtual machine instances; Migration and load balancing (G06F2009/45595) Network integration; Enabling network access in virtual machine instances (H04L63/10) Network architectures or network communication protocols for network security for controlling access to network resources (H04L63/1441) Countermeasures against malicious traffic (H04L63/145) Countermeasures against malicious traffic the attack involving the propagation of malware through the network, e.g. viruses, trojans or worms (H04L67/1008) Server selection in load balancing based on parameters of servers, e.g. available memory or workload (H04L67/101) Server selection in load balancing based on network conditions

		risks associated with the plurality of entry points.	

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Application of:	)	
	)	Group Art Unit: 2431
Avi SHUA	)	
	)	Examiner: Sarah SU
Application No.: 17/657,981	)	
	)	
Filed: April 5, 2022	)	Confirmation No.: 2397
	)	
For: FORWARD AND REARWARD FACING ATTACK VECTOR VISUALIZATION	)	***VIA EFS-WEB***
	)	

Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Commissioner:

REPLY TO *EX PARTE QUAYLE* ACTION

In reply to the *Ex Parte Quayle* Action mailed October 31, 2022, please amend the above-identified application as follows:

Amendments to the Specification are reflected in the substitute specification attached.

Remarks follow the amendments section of this paper.

REMARKS

I. CLAIM STATUS

Claims 81-100 are pending and under examination. Claims 1-80 and 101-159 were previously canceled without prejudice or disclaimer. No new matter is added.

II. ALLOWABLE SUBJECT MATTER

Applicant thanks Examiner Su for noting claims 81-100 are free of prior art and are allowable pending resolution of the objections in the present action.

III. RESPONSE TO *EX PARTE QUAYLE* ACTION

The Examiner contends that alleged informalities are present in the specification and in the present drawings. In response, Applicant has amended and attached a substitute specification to remedy these informalities. For example, Applicant has amended paragraphs [008], [00114], [00145], and [00456].

IV. CONCLUSION

In view of the foregoing Amendments and Remarks, Applicant respectfully requests the timely allowance of the pending claims.

Please charge any additional required fees to Deposit Account No. 06-0916.

Respectfully submitted,

FINNEGAN, HENDERSON, FARABOW,
GARRETT & DUNNER, L.L.P.

Dated: November 17, 2022

By: /Christopher C. Johns/
Christopher C. Johns
Reg. No. 68,664
(202) 408-4000

UNITED STATES
PATENT AND TRADEMARK OFFICE

P.O. Box 1450
Alexandria, VA 22313 - 1450
www.uspto.gov

ELECTRONIC ACKNOWLEDGEMENT RECEIPT

APPLICATION #	RECEIPT DATE / TIME	ATTORNEY DOCKET #
17/657,981	**11/17/2022 04:33:07 PM ET**	**15635.0008-00000**

Title of Invention

Forward and Rearward Facing Attack Vector Visualization

Application Information

APPLICATION TYPE	Utility - Nonprovisional Application under 35 USC 111(a)	PATENT #	-
CONFIRMATION #	2397	FILED BY	Wanda Whittington
PATENT CENTER #	61194385	FILING DATE	04/05/2022
CUSTOMER #	22852	FIRST NAMED INVENTOR	Avi SHUA
CORRESPONDENCE ADDRESS	-	AUTHORIZED BY	Christopher Johns

Documents

TOTAL DOCUMENTS: 4

DOCUMENT		PAGES	DESCRIPTION	SIZE (KB)
15635-0008-00000-Clean Substitute Specification.pdf		101	Specification	621 KB
15635-0008-0000-As-Filed Marked Up Substitute Specification.pdf		101	Specification	627 KB
15635-0008-00000Final Response to Ex Parte Quayle Action - SIGNED.pdf		2	-	21 KB
15635-0008-00000Final Response to Ex Parte Quayle Action - SIGNED-A.QU.pdf	(1-1)	1	Response after Ex Parte Quayle Action	16 KB

15635-0008-00000Final Response to Ex Parte Quayle Action - SIGNED-REM.pdf	(2-2)	1	Applicant Arguments/Remarks Made in an Amendment	15 KB

Digest

DOCUMENT	MESSAGE DIGEST(SHA-512)
15635-0008-00000-Clean Substitute Specification.pdf	A808630EACE09267EA97F301DABBFEEF40B97CBA778B1BA2852DC6DE4B2D9121D04C7DB9752121A7EC86D556E0D49B09FF6C6C54BD6164EC1EE2B65E523DCA5E
15635-0008-0000-As-Filed Marked Up Substitute Specification.pdf	8FD86EE1DD1229C7308EE21404FEDA96ECAD768D3D153A3864C3A4ED052F9EE9CA4FFDB6091FC3B7F72D8EDE11F0A92EA77AD2B7AE5F7BDAEA8CEC3A482516D7
15635-0008-00000Final Response to Ex Parte Quayle Action - SIGNED.pdf	437338C72DE40FEC31E8C63ECB9FF99E75D97B92D763BEB826AFFB929D64C8FC6597541AD532AFEB8ACA90B03AB8D79DF71472BE9A8049A5C6949FA698F73A74
15635-0008-00000Final Response to Ex Parte Quayle Action - SIGNED-A.QU.pdf	D25E78D0C2C58409C8F651593339B71DD955379455318E58D6A6828061CB235103F987E0409E5042A7DFDC40ED5CBF1A4172C5A4EFF2517DE6ADE0030914C63A
15635-0008-00000Final Response to Ex Parte Quayle Action - SIGNED-REM.pdf	D264F1BDC7F7D7472CDBFD4D582CADB627C4129569280EAB79F38CE7390B6A4E06C204C7294F77B77F5B900FAD418B20FFE1C73CD38B91FFDCCE85D43DE232E7

This Acknowledgement Receipt evidences receipt on the noted date by the USPTO of the indicated documents, characterized by the applicant, and including page counts, where applicable. It serves as evidence of receipt similar to a Post Card, as described in MPEP 503.

New Applications Under 35 U.S.C. 111

If a new application is being filed and the application includes the necessary components for filing date (see 37 CFR 1.53(b)-(d) and MPEP 506), a Filing Receipt (37 CFR 1.54) will be issued in due course and the date shown on this Acknowledgement Receipt will establish the filing date of the application

National Stage of an International Application under 35 U.S.C. 371

If a timely submission to enter the national stage of an international application is compliant with the conditions of 35 U.S.C. 371 and other applicable requirements a Form PCT/DO/EO/903 indicating acceptance of the application as a national stage

submission under 35 U.S.C. 371 will be issued in addition to the Filing Receipt, in due course.

New International Application Filed with the USPTO as a Receiving Office

If a new international application is being filed and the international application includes the necessary components for an international filing date (see PCT Article 11 and MPEP 1810), a Notification of the International Application Number and of the International Filing Date (Form PCT/RO/105) will be issued in due course, subject to prescriptions concerning national security, and the date shown on this Acknowledgement Receipt will establish the international filing date of the application.

Docket Number: 15635.0004-00304

SYSTEMS AND METHODS FOR SIDE SCANNING

PRIORITY

[001] This application claims the benefit of U.S. Provisional Application No. 63/180,048 filed on April 26, 2021, the contents of which are hereby incorporated by reference.

TECHNICAL FIELD

[002] Disclosed embodiments include new systems and methods for securing cloud infrastructure to help meet compliance mandates, without disrupting business operations in live environments.

BACKGROUND

[003] Modern organizations typically depend on cloud infrastructure for data storage and processing. Data storage and processing nodes, among other things, power today's electronic infrastructure. In contrast to early network-centric days, an organization may have little in the way of "on-prem" (on-premises) systems and may run its entire business through systems having shared resources. Such an infrastructure may be the entire inventory of cloud assets for an organization, including running and stopped workloads of all types: virtual machines, containers, storage objects, load balancers, IAM (Identity and Access Management) configurations, and more. Organizations are searching for effective ways to scan their entire cloud estate to look for risks stemming from vulnerabilities, misconfigurations, malware, lateral movement risk, weak and leaked passwords, and improperly secured PII.

[004] The infrastructure of these organizations are plainly different than '90s-style physical servers running on bare metal. Existing agents and scanners from the on-prem days have been simply reused in cloud environments without reconfiguration. Before the cloud, it was required to secure physical hosts. That meant spending time installing multiple security agents—one for each server. IP addresses were assigned to physical assets and they seldom changed. Even then, however, agent integration was tedious and coverage rarely reached 100% of assets. In a cloud environment, scaling up and down is a frequent occurrence—possibly thousands of times per hour across multiple clouds—and all within a CI/CD pipeline that builds an organization's infrastructure. There are containers and virtual machines to contend with, and agents carry high operational costs.

[005] This causes problems of inefficiency and overuse of resources, including excess processor cycles on regular scans. In turn, such problems and inefficiencies can lead to a higher monetary cost for hosted resources as some cloud providers charge per processing cycle.

[006] Relying on agents for security visibility in the cloud also has drawbacks. Visibility is limited to only those assets that are already known and accessible to an agent, requiring that such assets are capable of having the agent installed and maintained in the first place. The assets also must have ongoing network connectivity to the backend, something that makes the

development cycle inefficient as developers tend not to be inclined to deploy agents on virtual machines, in containers, and in serverless configurations—let alone dealing with their never-ending maintenance—thus leading to insecure implementations.

[007] An authenticated scan allows for direct host access using remote protocols such as SSH or RDP. The scanner uses a privileged account to log in and determine how secure each host is from an inside vantage point. While authenticated scans can successfully discover potential vulnerabilities, they are limited as they require a privileged account on each scanned host. Furthermore, scans use significant system resources during the test procedures and require opening ports that by themselves pose a security risk.

[008] An unauthenticated scan can only examine publicly visible information and is not able to provide detailed information about assets. It essentially acts as a friendly attacker. An unauthenticated scan can easily miss identifying some assets and vulnerabilities, making it much less effective. For example, a website at ~~https://mydomain.com/email_campaign~~ mydomain.com/email_campaign not linked from a main website will not be scanned unless the scanner is manually configured. This can leave organizations exposed to vulnerabilities in areas where the scanner cannot reach.

[009] While unauthenticated scanners act like an attacker, they often get stuck where a real attacker would not. For example, a CAPTCHA can easily prevent automatic mechanisms (such as scanners) from registering. But such techniques would not have any effect on a human attacker.

[0010] Cloud security posture managers (CSPMs) are scanning tools developed specifically for the cloud. Rather than going inside the machine, a CSPM analyzes the cloud configuration itself for weaknesses. CSPMs are used to discover, assess, and solve cloud misconfigurations but provide shallow coverage where cloud security is concerned because CSPMs have difficulty detecting critical risks such as vulnerabilities, malware, and misconfigurations within the workloads themselves. Organizations choosing to combine agent-based solutions with a CSPM end up getting flooded with separate alerts that lack context which results in alert fatigue on behalf of security analysts.

[0011] Therefore, there is a need for improved security scanning systems that solve the problem with existing systems and are more adapted for use in cloud infrastructures.

[0012] The foregoing background is for illustrative purposes and is not intended as a discussion of the scope of the prior art or an admission of prior art.

SUMMARY

[0013] This summary is provided for the convenience of the reader to present a basic understanding of a few exemplary embodiments of this disclosure, and does not wholly define the breadth of the disclosure. This summary is not an extensive overview of all contemplated embodiments and is intended to neither identify key or critical elements of all embodiments nor

to delineate the scope of any or all aspects. Its sole purpose is to present some features of one or more embodiments in a simplified form as a prelude to the more detailed description presented later. For convenience, the term "embodiments" may be used herein to refer to one or more features or combinations of features described herein. A description associated with an embodiment does not necessarily apply to all embodiments.

[0014] Embodiments of the present disclosure may deliver instant-on, workload level visibility across cloud infrastructure assets without running opcodes on the target scanned workloads, helping organizations to: (1) Detect risks such as vulnerabilities, malware, misconfigurations, lateral movement risk, and unsecured sensitive data; (2) See cloud inventory at every layer: infrastructure, operating system, applications, and data; (3) Discover and see previously missed assets.

[0015] Embodiments of the present disclosure may further use an out-of-band process to reach cloud workloads through the runtime storage layer, combining this with information gathered from cloud provider APIs, including but not limited to K8S APIs.

DRAWINGS

[0016] The drawings and figures contained herein are a part of this specification, illustrate several embodiments consistent with the disclosure and, together with the description, serve to explain principles of the disclosure.

[0017] **Fig. 1** is a schematic block diagram illustrating an exemplary embodiment of a network including computerized systems, consistent with disclosed embodiments.

[0018] **Fig. 2A** is a schematic block diagram illustrating an exemplary embodiment of a process for integration, scanning, assessment, analyzing/reporting, and change reviewing, consistent with disclosed embodiments.

[0019] **Fig. 2B** is a schematic block diagram illustrating an exemplary embodiment of a process for integration, consistent with disclosed embodiments.

[0020] **Fig. 2C** is a schematic block diagram illustrating an exemplary embodiment of a process for scanning/mapping, consistent with disclosed embodiments.

[0021] **Fig. 2D** is a schematic block diagram illustrating an exemplary embodiment of a process for assessment, consistent with disclosed embodiments.

[0022] **Fig. 3** is a block diagram of method for insight extraction, consistent with disclosed embodiments.

[0023] **Fig. 4** depicts a cybersecurity system performing a side scanning function to protect against potential vulnerabilities, consistent with disclosed embodiments.

[0024] **Fig. 5** of the disclosed embodiments describes a method of operating a cybersecurity system performing a side scanning function to protect against potential vulnerabilities, consistent with disclosed embodiments.

[0025] **Fig. 6** is a block diagram of method for risk prioritization, consistent with disclosed embodiments.

[0026] **Fig. 7** is a block diagram illustrating an exemplary process of cyber security scanning for a cloud environment, consistent with the disclosed embodiments.

[0027] **Fig. 8** is a schematic block diagram illustrating an exemplary embodiment of a system for performing visualization of forward and backward facing threats, consistent with the disclosed embodiments.

[0028] **Fig. 9** is a block diagram illustrating an exemplary process of matching keys with compute resources in a cloud environment, consistent with the disclosed embodiments.

[0029] **Fig. 10** is a block diagram of a method for deployment of ephemeral scanners, consistent with disclosed embodiments.

[0030] **Fig. 11** represents a schematic block diagram illustrating an exemplary embodiment of a method for providing a dashboard aggregating risk information, consistent with disclosed embodiments.

[0031] **Fig. 12** is a schematic block diagram illustrating an exemplary embodiment of a visual representation of displaying aggregated cybersecurity risk information, consistent with disclosed embodiments.

[0032] **Fig. 13** is a schematic block diagram illustrating an exemplary embodiment of a visual representation of a flow path for aggregated risk information, consistent with disclosed embodiments.

DETAILED DESCRIPTION

[0033] Exemplary embodiments are described with reference to the accompanying drawings. The figures are not necessarily drawn to scale. While examples and features of disclosed principles are described herein, modifications, adaptations, and other implementations are possible without departing from the spirit and scope of the disclosed embodiments. For example, with this detailed description provides a few examples, these implementations are provided as examples only and are not restrictive of the claim concepts that follow or any of the descriptions herein. Also, the words “comprising,” “having,” “containing,” and “including,” and other similar forms are intended to be equivalent in meaning and be open ended in that an item or items following any one of these words is not meant to be an exhaustive listing of such item or items or meant to be limited to only the listed item or items. It should also be noted that as used herein and in the appended claims, the singular forms “a,” “an,” and “the” include plural references unless the context clearly dictates otherwise.

[0034] In the following description, various working examples are provided for illustrative purposes. However, it is to be understood that the present disclosure may be practiced without one or more of these details.

[0035] It is intended that one or more aspects of any mechanism may be combined with one or more aspect of any other mechanisms, and such combinations are within the scope of this disclosure.

[0036] Various embodiments are described herein with reference to a system, method, device, or computer readable medium. It is intended that the disclosure of one is a disclosure of all. For example, it is to be understood that disclosure of a computer readable medium described herein also constitutes a disclosure of methods implemented by the computer readable medium, and systems and devices for implementing those methods, via for example, at least one processor. It is to be understood that this form of disclosure is for ease of discussion only, and one or more aspects of one embodiment herein may be combined with one or more aspects of other embodiments herein, within the intended scope of this disclosure.

[0037] Embodiments described herein may refer to a non-transitory computer readable medium containing instructions that when executed by at least one processor, cause the at least one processor to perform a method. Non-transitory computer readable medium may include any medium capable of storing data in any memory in a way that may be read by any computing device with a processor to carry out methods or any other instructions stored in the memory. The non-transitory computer readable medium may be implemented as hardware, firmware, software, or any combination thereof. Moreover, the software may preferably be implemented as an application program tangibly embodied on a program storage unit or computer readable medium consisting of parts, or of certain devices and/or a combination of devices. The application program may be uploaded to, and executed by, a machine having any suitable architecture. Preferably, the machine may be implemented on a computer platform having hardware such as one or more central processing units ("CPUs"), a memory, and input/output interfaces. The computer platform may also include an operating system and microinstruction code. The various processes and functions described in this disclosure may be either part of the microinstruction code or part of the application program, or any combination thereof, which may be executed by a CPU, whether or not such a computer or processor is explicitly shown. In addition, various other peripheral units may be connected to the computer platform such as an additional data storage unit and a printing unit. Furthermore, a non-transitory computer readable medium may be any computer readable medium except for a transitory propagating signal.

[0038] Memory employed herein may include a Random Access Memory (RAM), a Read-Only Memory (ROM), a hard disk, an optical disk, a magnetic medium, a flash memory, other permanent, fixed, volatile or non-volatile memory, or any other mechanism capable of storing instructions. The memory may include one or more separate storage devices collocated or disbursed, capable of storing data structures, instructions, or any other data. The memory may further include a memory portion containing instructions for the processor to execute. The

memory may also be used as a working scratch pad for the processors or as a temporary storage.

[0039] Some embodiments may involve at least one processor. A processor may be any physical device or group of devices having electric circuitry that performs a logic operation on input or inputs. For example, the at least one processor may include one or more integrated circuits (IC), including application-specific integrated circuit (ASIC), microchips, microcontrollers, microprocessors, all or part of a central processing unit (CPU), graphics processing unit (GPU), digital signal processor (DSP), field-programmable gate array (FPGA), server, virtual server, or other circuits suitable for executing instructions or performing logic operations. The instructions executed by at least one processor may, for example, be pre-loaded into a memory integrated with or embedded into the controller or may be stored in a separate memory.

[0040] In some embodiments, the at least one processor may include more than one processor. Each processor may have a similar construction, or the processors may be of differing constructions that are electrically connected or disconnected from each other. For example, the processors may be separate circuits or integrated in a single circuit. When more than one processor is used, the processors may be configured to operate independently or collaboratively. The processors may be coupled electrically, magnetically, optically, acoustically, mechanically or by other means that permit them to interact.

[0041] Consistent with the present disclosure, disclosed embodiments may involve a network. A network may constitute any type of physical or wireless computer networking arrangement used to exchange data. For example, a network may be the Internet, a private data network, a virtual private network using a public network, a Wi-Fi network, a LAN or WAN network, and/or other suitable connections that may enable information exchange among various components of the system. In some embodiments, a network may include one or more physical links used to exchange data, such as Ethernet, coaxial cables, twisted pair cables, fiber optics, or any other suitable physical medium for exchanging data. A network may also include a public switched telephone network ("PSTN") and/or a wireless cellular network. A network may be a secured network or unsecured network. In other embodiments, one or more components of the system may communicate directly through a dedicated communication network. Direct communications may use any suitable technologies, including, for example, BLUETOOTH™, BLUETOOTH LE™ (BLE), Wi-Fi, near field communications (NFC), or other suitable communication methods that provide a medium for exchanging data and/or information between separate entities.

[0042] Certain embodiments disclosed herein may also include a computing device for cloud cybersecurity, the computing device may include processing circuitry communicatively connected to a network interface and to a memory, wherein the memory contains instructions to be executed. The computing devices may be devices such as mobile devices, desktops,

laptops, tablets, or any other devices capable of processing data. Such computing devices may include a display such as an LED display, augmented reality (AR), virtual reality (VR) display.

[0043] "Software" as used herein refers broadly to any type of instructions, whether referred to as software, firmware, middleware, microcode, hardware description language, or otherwise. Instructions may include code (e.g., in source code format, binary code format, executable code format, or any other suitable format of code). The instructions, when executed by the one or more processors, may cause the processing system to perform the various functions described in further detail herein.

[0044] The one or more processors may be implemented with any combination of general-purpose microprocessors, microcontrollers, digital signal processors (DSPs), field programmable gate array (FPGAs), programmable logic devices (PLDs), controllers, state machines, gated logic, discrete hardware components, dedicated hardware finite state machines, or any other suitable entities that can perform calculations or other manipulations of information.

[0045] Aspects of this disclosure may provide technical solutions to challenges associated with cloud cybersecurity. Disclosed embodiments include methods, systems, devices, and computer-readable media. For ease of discussion, a system is described below with the understanding that the disclosed details may equally apply to methods, devices, and computer-readable media.

[0046] Embodiments of the present disclosure include technology referred to as "SideScanning." In contrast to some existing systems and techniques, these embodiments may provide a distinct advantage because the technology does not necessarily require entering into each workload to inspect data. Rather, some embodiments use an out-of-band process to reach cloud workloads through a runtime storage layer, combining this with metadata gathered from API provided through a cloud service provider's system, thus providing visibility of cloud environments both at a low level and with context, without the requirement for an agent or network scanner.

[0047] **Fig. 1** is a schematic block diagram 100 illustrating an exemplary embodiment of a network including computerized systems, consistent with the disclosed embodiments. Diagram 100 includes user device 102, network 105, and cloud infrastructure 106.

[0048] Cloud infrastructure 106 includes scanning system 101, databases 103A-103D, virtual machines 107A-107D, databases 109A-109D, storage 111A-111D, keystores 113A-113D, and load balancer 115. While particular numbers and arrangements of devices, systems, and connections, are depicted in exemplary Fig. 1, in some embodiments, each of the devices, systems, or connections may be omitted, duplicated, or modified. For example, in some embodiments, databases 109A-109D may exist as only a single database; in other embodiments, cloud infrastructure 106 may exist as one or more distinct or combined

infrastructures (e.g., operated by the same or different cloud services). In some embodiments, scanning system 101 and/or databases 103A-103D may be part of cloud infrastructure 106 (and may be connected to the various other systems and devices in cloud infrastructure 106); in other embodiments, scanning system 101 and/or databases 103A-103D may be separate from cloud infrastructure 106 (e.g., connected to the systems and devices in cloud infrastructure 106 through network 105).

[0049] Scanning system 101, in some embodiments, may include one or more computer systems. Each of the one or more computer systems may include memory storing instructions and at least one CPU configured to execute those instructions to perform operations as discussed herein. In some embodiments, the instructions cause the CPU to perform scanning operations. In some embodiments, scanning system 101 may perform a scanning operation on one or more workloads (e.g., systems, devices, resources, etc.) in cloud infrastructure 106.

[0050] User device 102, in some embodiments, may include a mechanism operated by a user to control scanning system 101. For example, in some embodiments, user device 102 may be any of a personal computer, a server, a thin client, a tablet, a personal digital assistant, a smartphone, a kiosk, or any other mechanism enabling data input. User device 102 may be operated to instantiate functionality, access data, or otherwise interact with scanning system 101 via network 105, as described herein.

[0051] Databases 103A-103D include data stores for use by scanning system 101. In some embodiments, one or more of databases 103A-103D may be implemented as a NoSQL database, a relational database, a cloud database, a columnar database, a wide column database, a key-value database, an object-oriented database, a hierarchical database, or any other kind of database. In some embodiments, one or more of databases 103A-103D may be implemented as flat file stores, data stores, or other non-database storage systems. In some embodiments, databases 103A-103D may be implemented using one or more of ElasticCache, ElasticSearch, DocumentDb, DynamoDB, Neptune, RDS, Aurora, Redshift clusters, Kafka clusters, or EC2 instances.

[0052] Network 105 may be implemented as one or more interconnected data networks. For example, network 105 may include one or more of any type of network (including infrastructure) that provides communications, exchanges information, and/or facilitates the exchange of information, such as the Internet, a Local Area Network, a near field communication (NFC) network, or other suitable connection(s) that enables the sending and receiving of information between the components of system 100. Network 105 may be implemented using wireless connections, wired connections, or both. In some embodiments, one or more components of system 100 can communicate through network 105. In some embodiments, one or more components of system 100 may communicate directly through one or more dedicated communication links. While particular devices and systems are shown as connected to network

105, in some embodiments, more or fewer devices and systems may be connected to network 105.

[0053] Cloud infrastructure 106 may be implemented as a set of devices and systems offered by a single cloud service provider. For example, cloud infrastructure 106 may comprise devices and systems that are part of Amazon Web Services, Microsoft Azure, Google Cloud Platform, IBM Cloud, Alibaba Cloud, or any other cloud platform provider. In some embodiments, one or more of the devices and systems in cloud infrastructure may require authentication or other identity validation for access. For example, to access virtual machine 107A, a user may be required to enter a password or provide a key. Systems (e.g., scanning system 101 or user device 102) may administer or interact with cloud infrastructure 106 using a cloud service provider's system (not pictured).

[0054] Virtual machines 107A-107D may include one or more devices and systems that implement a virtualized/emulated version of a computer. A virtual machine may be implemented as an emulated version of a computer - including an operating system, memory, storage, graphics processing - such that it can be indistinguishable from a standard (non-virtual) machine to a running program. A computer system, referred to as a "host," may operate virtual machines 107A-107D, referred to as "guests," by dividing the resources of the host between the virtual machines such that each virtual machine is isolated from one another. This means that in some embodiments, one virtual machine, and the operating system(s) and application(s) running thereon, is only able to access the resources that are allocated to that virtual machine and cannot access resources allocated to other virtual machines. For example, if a host has 32 gigabytes of random access memory (RAM), and is hosting three virtual machines, the host may segment 8 gigabytes of RAM to each virtual machine such that each virtual machine may only access data in that 8 gigabytes of RAM and not any of the other 24 gigabytes. Examples of commercial virtual machine software and services include VMWare Workstation, VMWare Server, VMWare ESXi, VirtualBox, Parallels Desktop, Parallels RAS, Amazon Machine Image, Amazon ECS, Kubernetes, Microsoft Hyper-V, and Xen.

[0055] Databases 109A-109D may include data stores for use by devices and systems in cloud infrastructure 106. In some embodiments, one or more of databases 109A-109D may be implemented as a NoSQL database, a relational database, a cloud database, a columnar database, a wide column database, a key-value database, an object-oriented database, a hierarchical database, or any other kind of database. In some embodiments, one or more of databases 109A-109D may be implemented as flat file stores, data stores, or other non-database storage systems. In some embodiments, databases 109A-109D may be implemented using one or more of ElasticCache, ElasticSearch, DocumentDb, DynamoDB, Neptune, RDS, Aurora, Redshift clusters, Kafka clusters, or EC2 instances. Databases 109A-109D may store data usable by devices or systems in cloud infrastructure 106. The data, in some embodiments,

may include e-commerce data (e.g., shipments, orders, inventory), media data (e.g., pictures, movies, streaming data), financial data (e.g., banking data, investment data), or other data.

[0056] Storage 111A-111D may include storage systems for use by devices and systems in cloud infrastructure 106. In some embodiments, one or more of storage 111A-111D may be implemented as a hard drive, a RAID array, flash memory, optical storage, or any other kind of storage. Each of 111A-111D may include one or more filesystems (e.g., Amazon Elastic File System, GlusterFS, Google File System, Hadoop Distributed File System, OpenZFS, S3, Elastic Block Storage).

[0057] In some embodiments, systems and devices of cloud infrastructure 106 may use databases 109A-109D to store data that is accessed frequently (where, for example, access is required within a few milliseconds), and may use storage 111A-111D to store data that is accessed less frequently (where, for example, access is required within a few minutes or hours).

[0058] Keystores 113A-113D may include systems storing keys for accessing data and functionality. For example, to access certain data or systems, a system may require the use of passwords or keys in keystores 113A-113D for authentication. The data and functionality that the keys grant access to may be part of cloud infrastructure 106 or may be separate from cloud infrastructure 106. For example, keystores 113A-113D may include systems that store public and private keys (e.g., for use via SSH), may store passwords (e.g., login information for websites or programs), may store tokens (e.g., one-time passcodes), or the like. In some embodiments, keystores 113A-113D may be implemented as one or more of Amazon Web Services KMS, Azure Key Vault, or Google KMS.

[0059] Load balancer 115 may include one or more systems that balance incoming requests between the different systems and devices of cloud infrastructure 106. For example, load balancer 115 may be configured to determine usage (e.g., processor load, used storage capacity) of systems or devices in cloud infrastructure 106 to assist in determining where to route an incoming request from network 105 to store data, perform processing, or retrieve data.

[0060] Load balancer 115 may be configured to receive an incoming request from user device 102. Upon receipt of the request, load balancer 115 may consult a data store (part of or separate from load balancer 115; not pictured) to determine usage or forecasted usage of various systems or devices in cloud infrastructure 106, and may forward the request to the systems or devices having the lowest usage or forecasted usage.

[0061] **Fig. 2A** is a schematic block diagram illustrating an exemplary embodiment of a process 200 for integration, scanning, assessment, and review, consistent with the disclosed embodiments. In some embodiments, the steps in Fig. 2A are executed using software and hardware of scanning system 101. In some embodiments, the steps in Fig. 2A may be performed in an order other than those depicted in Fig. 2A. In some embodiments, steps may also be omitted, repeated, or modified. In some embodiments, information gathered in one step

of Fig. 2A may be used to provide context or other information for use in another step of Fig. 2A. In other embodiments, the steps in Fig. 2A may be executed by other devices.

[0062] Process 200 begins with step 201. In step 201, scanning system 101 may execute a process of integration. The integration process may be performed by scanning system 101 with cloud infrastructure 106. In some embodiments, the integration process includes creating a connection between an account on scanning system 101 and an account on cloud infrastructure 106. The process of integration in step 201 may, in some embodiments, be implemented as described below with respect to Fig. 2B.

[0063] In step 203, scanning system 101 may execute a process of scanning/mapping. The scanning/mapping process may be performed by scanning system 101 with cloud infrastructure 106. In some embodiments, the scanning and mapping process may include analyzing data relating to cloud infrastructure 106 using scanning system 101, by reading cloud infrastructure 106 through the connection made in step 201 and generate a "map" (e.g., a data structure or data collection) representing systems and devices in cloud infrastructure 106. The process of scanning and mapping in step 203 may, in some embodiments, be implemented as described below with respect to Fig. 2C.

[0064] In step 205, scanning system 101 may execute a process of assessing. The assessing process may be performed by scanning system 101. In some embodiments, the assessing process may include reviewing vulnerabilities, infrastructure, interconnections, data, and other information, using scanning system 101. The process of assessing in step 205 may, in some embodiments, be implemented as described below with respect to Fig. 2D.

[0065] In step 207, scanning system 101 may execute a process of analyzing/reporting. The analyzing/reporting may be performed by scanning system 101. In some embodiments, the change reviewing process comprises scanning system 101 scanning cloud infrastructure 106 again to determine the differences between an earlier observed snapshot of cloud infrastructure 106 and the current state thereof. In some embodiments, scanning system 101 may analyze information (e.g., steps 203 and/or 205) to generate reports. The reports may list what each vulnerability is, where it is located, and its priority. In this way, security engineers and DevOps teams may be able to easily assess how to best allocate their time and attention.

[0066] In some embodiments, analyzing/reporting in step 207 may include scanning system 101 combining conclusions from different environmental perspectives (e.g., metadata) into a single model. For example, scanning system 101 may map the running services on cloud infrastructure 106 and consider collected vulnerability data. Scanning system 101, in some embodiments, may generate a visualization of the map for review, listing where the vulnerabilities are on a two-dimensional graphical representation of cloud infrastructure 106.

[0067] In some embodiments, scanning system 101 may indicate in the map whether an asset (e.g., a system or device) is Internet-facing and easily accessible to attackers, exposed only to

internal assets or if it is private altogether (and may be less critical). For example, based on the contextual map, scanning system 101 may perform a "forward" analysis of the specific asset under identification to identify at least one possible Internet-originating attack vector to the asset. Alternatively or additionally, scanning system 101 may perform a "backward" analysis of the specific asset to identify exposure risk to assets downstream of the specific asset, wherein the downstream exposure risk includes an identification of an exposed asset, an entry point to the exposed asset, and lateral movement risks associated with the exposed asset. Both backward and forward analysis can be recursively, for one or more hops, Scanning system 101 may present these analyses to a user via graphical user interface or expose them via an API.

[0068] As one example, scanning system 101 may determine a vulnerability in a web service. In some embodiments, scanning system 101 may score the vulnerability as: high risk if the web service is connected to the Internet (e.g., has at least one public port forwarded through a firewall to the web service, is able to be accessed through a load balancer, is able to be accessed via reverse proxy); medium risk if the web service is only accessible internally (e.g., because of firewall configuration); and low risk if access to the web server is blocked by a configuration of cloud infrastructure 106. As another example, if a machine is stopped (e.g., turned off, not running, not connected), a vulnerability, but one that is less likely to be exploited because the machine isn't running. This affects its risk score and other mitigating factors.

[0069] In step 209, scanning system 101 may execute a process of change reviewing. The change reviewing may be performed by scanning system 101. In some embodiments, the change reviewing process may include scanning system 101 scanning cloud infrastructure 106 again to determine the differences between an earlier observed snapshot of cloud infrastructure 106 and the current state thereof.

[0070] In some embodiments, scanning system 101 may monitor cloud change, or "delta," logs (e.g., Cloud event logs such as AWS CloudTrail, monitoring a network-related change in state, a trust-related change in state, or a disk configuration-related change in state in at least one of the primary asset group or a secondary asset group) and contextualize associated risks discovered within a customer cloud account. In some embodiments, scanning system 101 may generate a new map of assets based on a review of all systems or devices in cloud infrastructure 106 (e.g., as described above with respect to step 203), and compare the new map to the map generated earlier in step 203 to determine which assets have changed. In some embodiments, this comparison may require comparing data, metadata, network connections, software configurations, firewall configurations, or any other aspects related to cloud infrastructure 106, including, for example, determining new assets that are created or assets for which their configuration has changed.

[0071] After step 209, as depicted in Fig. 2A, process 200 may return to step 205. Scanning system 101 may then perform step 205 again. In some embodiments, steps 205, and 207 may

be performed only on the devices or systems that have experienced a change that was detected by step 209. In some embodiments, process 200 may return to step 203 after step 209.

[0072] **Fig. 2B** is a schematic block diagram illustrating an exemplary embodiment of a process 210 for integration, consistent with the disclosed embodiments. In some embodiments, the steps in Fig. 2B are executed using software and hardware of scanning system 101.Integration, in some embodiments, is a process for creating a trust relationship between scanning system 101 and cloud infrastructure 106.

[0073] In some embodiments, the steps in Fig. 2B may be performed in an order other than those depicted in Fig. 2B. In some embodiments, steps may also be omitted, repeated, or modified. In some embodiments, information gathered in one step of Fig. 2B may be used to provide context or other information for use in another step of Fig. 2B. In other embodiments, the steps in Fig. 2B may be executed by other devices.

[0074] Process 210 may begin with step 211 to initiate a connection to cloud infrastructure 106. In step 211, scanning system 101 may send a message to user device 102 instructing a user to authenticate, or log in, to a cloud service provider's system operating cloud infrastructure 106. For example, the user may use a username, password, one-time password, two-factor authentication, or any other authentication mechanism to gain access to a cloud service provider's system.

[0075] Concurrently with or after the first message, scanning system 101 may send a second message to user device 102, instructing the user to generate a role. The second message may include instructions for the user to follow to generate the role. In step 213, a user may provide (e.g., via a keyboard at user device 102) a role definition to the cloud service provider's system. In some embodiments, the role definition includes read-only permissions and permissions to read a block storage layer (containing block storage volumes). In some embodiments, scanning system 101 provides a role formation template (e.g., an Amazon Web Services CloudFormation Template) for use with cloud infrastructure 106 to create the necessary role. In step 213, the user may utilize user device 102, for example, by copying and pasting a URL of the template, downloading and uploading the template to the cloud service provider's system, or selecting the template from a list of templates.

[0076] In step 215, the cloud service provider's system may generate a string (e.g., a "key" or "resource name") for use by scanning system 101. In some embodiments, this string may be used to enable access by scanning system 101 to the workload of cloud infrastructure as permitted by the generated role. A user using user device 102 may copy the string and paste it into a user interface presented by scanning system 101 on user device 102. Other aspects of transmitting this string to scanning system 101 are possible as well.

[0077] In step 217, scanning system 101 determines that it is able to access cloud infrastructure 106. For example, scanning system 101 may attempt to perform a command such as authenticating to cloud infrastructure 106 using the string received in step 215 or attempting to perform a command using cloud infrastructure (e.g., requesting a listing of files stored on storage 111A). Once scanning system 101 determines that it is able to access cloud infrastructure 106, the process may return to Fig. 2A and step 203.

[0078] **Fig. 2C** is a schematic block diagram illustrating an exemplary embodiment of a process 220 for scanning/mapping, consistent with the disclosed embodiments. In some embodiments, the steps in Fig. 2C are executed using software and hardware of scanning system 101. In some embodiments, the steps in Fig. 2C may be performed in an order other than those depicted in Fig. 2C. In some embodiments, steps may also be omitted, repeated, or modified. In some embodiments, information gathered in one step of Fig. 2C may be used to provide context or other information for use in another step of Fig. 2C. In other embodiments, the steps in Fig. 2C may be executed by other devices.

[0079] Process 220 may begin with step 221 where scanning system 101 initiates a process to access keys, such as those keys stored in keystores 113A-113D. Scanning system 101 may utilize the string (e.g., a key or resource name) received in process 210 to authenticate in order to retrieve keys stored in keystores 113A-113D. Once authenticated, scanning system 101 may access one, some, or all keys stored in keystores 113A-113D. In some embodiments, step 221 may comprise utilizing pre-established trust relationships, such as AWS trust policies, instead of creating a new relationship.

[0080] In step 223, scanning system 101 may generate a “snapshot” of devices and systems in cloud infrastructure 106. In some embodiments, generating a snapshot may include reading devices and systems in cloud infrastructure, such as storage 111A-111D, databases 109A-109D, and virtual machines 107A-107D, and copying the information read from those devices and systems to storage at scanning system 101. Generating a snapshot, in some embodiments, may include recording a reference count to data blocks in one or more of storage 111A-111D, databases 109A-109D, and virtual machines 107A-107D, and copying each block them like a copy and write operation. In some embodiments, scanning system 101 may generate an Elastic Block Storage snapshot in step 223.

[0081] In some embodiments, scanning system 101 uses one or more keys retrieved from keystores 113A-113D to encrypt “snapshots” of cloud infrastructure.

[0082] In step 225, scanning system 101 may apply “tags” to the snapshot. This process may include, in some embodiments, adding information to the snapshot to identify the snapshot as being associated with scanning system 101. Scanning system 101, in some embodiments, may be configured to delete only snapshots with the associated tags.

[0083] In step 227, scanning system 101 may generate a map of cloud infrastructure 106. In some embodiments, this map may be in a form of graph: a plurality of interconnected vectors connecting the plurality of systems and devices, based on the networking configuration. Scanning system 101, or other devices, may traverse the map to identify vectors originating in the Internet and reaching the devices and systems in cloud infrastructure 106. In some embodiments, generating the map may include enumerating Internet-accessible services that are capable of serving as an Internet proxy. In some embodiments, a user may control scanning system 101 (e.g., via user device 102) to display the map on a graphical user interface.

[0084] In some embodiments, generating the map may include enumerating properties of all assets, including: virtual machines 107A-107D, databases 109A-109D, storage 111A-111D, keystores 113A-113D, load balancer 115, log files or databases, API gateway resources, API gateway REST APIs, Autoscaling groups, CloudTrail logs, CloudFront services, volumes, snapshots, VPCs, subnets, route tables, network ACLs, VPC endpoints, NAT gateways, ELB and ALB, ECR repositories, ECS clusters, services, and tasks, EKS, S3 bucket and Glacier storage, SNS topics, IAM roles, policies, groups, users, KMS keys, and Lambda functions. In some embodiments, generating the map in step 227 may further include analyzing devices or systems for a subset of risk related situations, including determinations of compromise situations (e.g., where an attacker has already gained access), imminent compromise situations (e.g., a known attack vector exists and can be used, such as a data store that is exposed to the public Internet without authentication), hazardous situations (e.g., a serious security implication, but no full attack vector exists), or informational situations (e.g., when storage 111A-111D has a limited amount of free space, or an unexploitable vulnerability exists).

[0085] Generating the map in step 227 may include recording information such a region identifier, site identifier, datacenter identifier, physical address, network address, workload name, or any other identifier which may be acquired via an API provided through a cloud service provider's system.

[0086] In step 229, scanning system 101 may provide alerts, e.g., to user device 102, indicating any situations found during the process of generating the map. For example, scanning system 101 may send one or more of an email, a popup alert, a text message, or other notification to user device 102.

[0087] After performing step 229, the process may return to Fig. 2A and step 205.

[0088] **Fig. 2D** is a schematic block diagram illustrating an exemplary embodiment of a process 230 for assessing, consistent with the disclosed embodiments. In some embodiments, the steps in Fig. 2D are executed using software and hardware of scanning system 101. In other embodiments, the steps in Fig. 2D may be executed by other devices. In some embodiments, scanning system 101 may execute the steps in Fig. 2D against a snapshot created in step 223 of Fig. 2C and stored at scanning system 101. In some embodiments, the steps in Fig. 2D may

be performed in an order other than those depicted in Fig. 2D. In some embodiments, steps may also be omitted, repeated, or modified. In some embodiments, information gathered in one step of Fig. 2D may be used to provide context or other information for use in another step of Fig. 2D.

[0089] Process 230 begins with step 231. In step 231, scanning system 101 may perform a step of vulnerability scanning. In some embodiments, step 231 comprises extracting everything in the snapshot, including operating system packages, installed software applications, libraries, and program language libraries such as Java archives, Python packages, Go modules, Ruby gems, PHP packages, and Node.js modules, or other software applications.

[0090] In some embodiments, step 231 may determine library versions, software versions, and other identifying characteristics of software and operating systems in the snapshot. Scanning system 101 may then try to match them to known vulnerabilities stored in a vulnerability database (e.g., one of databases 103A-103D). The vulnerability database, in some embodiments, may include vulnerability data from: NVD, WPVulnDB, US-CERT, Node.js Security Working Group, OVAL – Red Hat, Oracle Linux, Debian, Ubuntu, SUSE, Ruby Advisory Database, JVN, Safety DB(Python), Alpine secdb, PHP Security Advisories Database, Amazon ALAS, RustSec Advisory Database, Red Hat Security Advisories, Microsoft MSRC, KB, Debian Security Bug Tracker, Kubernetes security announcements, Exploit Database, Drupal security advisories, JPCERT. The vulnerability database may also include other vulnerability data (including, e.g., manually-added vulnerability data or vulnerability sources not listed above).

[0091] In step 233, scanning system 101 may perform a step of configuration scanning. In step 233, scanning system 101 may gather configuration information—such as a list of users of each system or device (e.g., virtual machines 107A-D), each system's or device's services, password hashes, and application-specific configurations for software/services such as Apache, Nginx, SSH, and other services. In some embodiments, scanning system 101 may perform a first analysis on all information collected in step 233 to remove sensitive information (e.g., social security numbers, passwords, birthdates) before proceeding to review the configuration information. In some embodiments, step 233 may comprise verifying adherence of the systems or devices in cloud infrastructure 106 to standards or benchmarks established by an external entity, such as the Center for Internet Security.

[0092] In some embodiments, scanning system 101 may perform a benchmarking process to detect misconfigurations of any services based on the information gathered in step 233. For example, scanning system 101 may determine the software version of each service and examine it against known vulnerabilities (e.g., stored in database 103A).

[0093] In some embodiments, scanning system 101 may determine bugs or other configuration risks that might only be exploitable from internal machines, because such bugs and risks can facilitate an attacker's lateral movement.

[0094] In some embodiments, scanning system 101 may evaluate network misconfigurations and their implications. Scanning system 101 may query devices and systems capable of routing traffic (e.g., load balancer 115, routers, switches, firewalls, and proxies) using an API provided through a cloud service provider's system to determine network configurations, and may evaluate them against known problematic configurations or other configurations.

[0095] In step 235, scanning system 101 may perform a step of malware scanning. In some embodiments, scanning system 101 may perform malware scanning across all filesystems in the snapshot (e.g., gathered from virtual machines 107A-107D or storage 111A-111D). Scanning system 101 may use multiple malware scanning software solutions to perform a malware scan against the filesystems, including one sourced from another vendor, such as bucketAV, Trend Micro Cloud One, Sophos Cloud Optix, Crowdstrike Falcon CWP, or others. In some embodiments, malware scanning in step 235 comprises utilizing signatures, heuristics, or sandboxing capabilities to deduce whether there is an infection on the machine.

[0096] In step 237, scanning system 101 may perform a step of lateral movement scanning. An attacker who establishes a network foothold usually attempts to move laterally from one resource to another in search of rich targets such as valuable data. Stolen passwords and keys unlock access to servers, files, and privileged accounts. In some embodiments, scanning system 101 may gather keys from each scanned system or device (e.g., virtual machines 107A-107D or storage 111A-111D). In some embodiments, scanning system 101 searches for passwords, scripts, shell history, repositories, or other data that may contain passwords, cloud access keys, SSH keys, or other key/password/access information that provide unchecked access to important resources. In some embodiments, scanning system 101 searches for such keys/passwords/access information and calculates a "hash" (a mathematical fingerprint) of each string. Scanning system 101 then attempts to match the hashed strings to hashes of strings that that are stored on different systems or devices. This will be used to detect the potential lateral movement between assets.

[0097] In step 239, scanning system 101 may perform a step of key/password scanning. As one example situation, suppose there is a weak or unprotected password stored (in plain text) in storage 111A. For example, if a personal email account has been compromised, the passwords may be known about in advance. Scanning system 101 may search the snapshot for similar usernames or login names, and, either using known dictionaries or the account owner's previously leaked passwords (stored in, e.g., database 103A), may attempt to login to one or more systems or devices in cloud infrastructure 106, and may record the result thereof.

[0098] In some embodiments, scanning system 101 may perform a "fuzzy search" on any usernames found in a password database (e.g., database 103A) to determine existing targets for password testing. In some embodiments, the fuzzy search uses the Damerau-Levenshtein edit distance algorithm.

[0099] As one example, scanning system 101 may determine that a leaked username includes the email address johndoe@acme.com. Scanning system 101 may try to match passwords from john.doe@acme.net, johndoe@gmail.com, and other variations. Scanning system 101 may perform a similar process against leaked passwords (e.g., if a leaked password is "Victory@19," scanning system may attempt to log in using "Victory@20."

[00100] In step 241, scanning system 101 may perform a step of sensitive information scanning. In some embodiments, scanning system 101 may search the snapshot for sensitive information, such as personally identifiable information (PII), Social Security numbers, healthcare information, or credit card numbers. In some embodiments, scanning system 101 may search data repository history as well. This is because it is not uncommon for an entire production environment repository to be cloned, with no one remembering the copy contains sensitive information. In some situations, detecting sensitive data not secured is critical in adherence to data privacy regulations.

[00101] To be certain that such alerts do not constitute false positives, in some embodiments, scanning system 101 may perform statistical scans against the data. For example, it is possible for a random number to resemble a Social Security number, yet it is extremely unlikely for the majority of a file with thousands of numbers to be valid Social Security Number by pure chance.

[00102] In step 243, scanning system 101 may perform a step of container scanning. In some embodiments, scanning system 101 may apply one or more of the preceding steps of Fig. 2D against containerized environments. In some embodiments, in order to do so, scanning system 101 reconstructs a container runtime layered file system (LFS) before recursively running one or more of steps 231-241 on the reconstructed file system.

[00103] Scanning system 101 may read network information from the snapshot in order to determine which services within which containers are exposed externally and within which ports they are accessible. For example, scanning system 101 may identify a port on which a vulnerable application is accessible based on known software vulnerabilities for a versions of software application. Scanning system 101 may query network accessibility information via an API provided through a cloud service provider's system and may use it to identify specific vulnerabilities susceptible to attack.

[00104] After performing step 243 on one or more containers, the process may return to Fig. 2A and step 207 (as discussed above).

[00105] **Foundational Techniques**

[00106] Aspects of this disclosure may include establishing a trusted relationship between a source account in a cloud environment and a scanner account. A trusted relationship, as used herein, may refer to a secure communication channel between at least two accounts (e.g., domains). The secure communications channel may include, for example, an administration link, communication link, a connection or communication network configured with security protocols, or any other secure relationship between two networked entities.. A trusted relationship between two accounts may enable user accounts and global groups to be used in a domain other than the domain where the accounts are defined. A source account, as used herein, may refer to a domain, a location on a network server, or anything used to access system resources. A cloud environment, as used herein, may refer to a platform implemented on, hosted on, and/or accessing servers that may be reached via the Internet or other shared network. A scanner account, as used herein, may refer to any type of account associated with a scanner. A scanner, as used herein, may refer to a device for examining, reading, or monitoring electronically accessible information. Fig. 1 illustrates one example of a scanner, which is depicted in an exemplary manner as scanning system 101.

[00107] By way of example, establishing a trusted relationship between a source account in a cloud environment and a scanner account may include an integration process (creating a connection between an account on scanning system 101 and an account on cloud infrastructure 106) discussed with reference to Fig. 2A. By way of one example, scanning system 101 of Fig. 1 may create a trusted relationship between a source account in a cloud environment and a scanner account.

[00108] Aspects of this disclosure may include using the established trust relationship, utilizing at least one cloud provider API to identify workloads in the source account. A cloud provider API, as used herein, may refer to any application program interface that allows an end user to interact with a cloud provider's service. A cloud provider, may include any entity through whom information is accessible over the Internet or other shared network. Workloads, as used herein, may refer to systems, devices, or resources in a network (or available via a network) such as a cloud infrastructure. By way of example, Fig. 1 illustrates examples of a workload: virtual machines 107A-107D, databases 109A-109D, storage 111A-111D, keystores 113A-113D, and load balancer 115. In some embodiments, scanning system 101 of Fig. 1 may perform a scanning operation on workload (e.g., systems, devices, resources, etc.) in cloud infrastructure 106.

[00109] In some embodiments, scanning system 101 of Fig. 1 may use an API to detect workloads such as virtual machines 107A-107D, databases 109A-109D, storage 111A-111D, keystores 113A-113D, and load balancer 115 in the source account.

[00110] Aspects of this disclosure may involve using the at least one cloud provider API to query a geographical location of at least one of the identified workloads. Querying, as used

herein, may refer to a request for data or information from a resource. A geographical location, as used herein, may refer to the data center where this workload is served, a position on Earth, a location in a network, an address of a server or other hardware, or any other information capable of identifying a locus, region, or position. Internet geolocation involves software capable of deducing a geographic position of a device connected to the Internet. For example, the device's IP address may be used to determine the country, city, or ZIP code, determining its geographical location. By way of example, scanning system 101 of Fig. 1 may query any of workloads virtual machines 107A-107D, databases 109A-109D, storage 111A-111D, keystores 113A-113D of cloud infrastructure 106 for their geographical location using a cloud provider API.

[00111] In some embodiments, the geographic location may include an identifier of a physical site. An identifier, as used herein, may refer to anything that allows for recognition of an object, location, or resource. A physical site, as used herein, may refer to an actual location. By way of example, scanning system 101 of Fig. 1 may query any of workloads virtual machines 107A-107D, databases 109A-109D, storage 111A-111D, keystores 113A-113D of cloud infrastructure 106 for an identifier of their physical site (e.g., address, postal code, or any other physical descriptor) using a cloud provider API.

[00112] In some embodiments, scanning system 101 of Fig. 1 may use an out-of-band process to reach cloud workloads through a runtime storage layer, in combination with information gathered from cloud provider APIs.

[00113] Aspects of this disclosure may include receive an identification of the geographic location. An identification of the geographic location, as used herein, may refer to a recognition, association, or classification of the geographic location. By way of example, scanning system 101 of Fig. 1 may receive a geographical location, after querying for it, by interrogating one or more workloads, such as virtual machines 107A-107D, databases 109A-109D, storage 111A-111D, or keystores 113A-113D of cloud infrastructure 106. For example, the scanning system 101 of Fig. 1 may receive address, postal code, or any other physical descriptor of any workloads in cloud infrastructure 106. In some embodiments, scanning system 101 of Fig. 1 may receive a listing locating the workloads of cloud infrastructure 106 on a two-dimensional graphical representation.

[00114] In some embodiments, the identification of the geographic location may include an identification of a data center, at least one of a data center name, an Internet Protocol (IP) address, a name of the cloud provider, or ~~a unique identity.~~ a unique identity. A data center, as used herein, may refer to a facility that centralizes an organization's shared IT operations and equipment for the purposes of storing, processing, and/or disseminating data and applications. A data center, for example, may include a dedicated space within a building, or a group of buildings used to house computer systems and associated components, such as

telecommunications and storage systems. An identification of a data center, as used herein, may refer to a recognition, association, or classification of a data center. A data center name, as used herein, may refer to a series of letters or numbers, a word, or set of words or numbers, an address, or a code identifying the data center. An Internet Protocol (IP) address, as used herein, may refer to a unique string of characters that identifies a computer using the Internet Protocol to communicate over a network. A cloud provider, as used herein, may refer to a company that offers any components of cloud computing (e.g., infrastructure as a service (IaaS), software as a service (SaaS) or platform as a service (PaaS)). A unique identity, as used herein, may refer to a distinctive numeric or alphanumeric string that is associated with a single entity within a given system.

[00115] By way of example, scanning system 101 of Fig. 1 may query for and receive a geographical location such as a data center name/city and state, data center address, IP address / specific cloud identifier of the workload, name of cloud provider (e.g., Amazon Web Services), or a specific identifier of information of any of workloads such as virtual machines 107A-107D, databases 109A-109D, storage 111A-111D, keystores 113A-113D of cloud infrastructure 106.

[00116] In some embodiments, scanning system 101 of Fig. 1 may receive a listing indicating locations of the workloads of cloud infrastructure 106 on a two-dimensional graphical representation.

[00117] In some other embodiments, scanning system 101 may receive additional information (in addition to geographic location information) such as whether the workload is Internet-facing and easily accessible to attackers, or if it is private (and may be less critical).

[00118] Aspects of this disclosure may include using the cloud provider APIs to access block storage volumes of the at least one workload. Block storage volumes, as used herein, may refer to data files maintained on Storage Area Networks (SANs) or in cloud-based storage environments. Each storage volume may act as an individual hard drive configured by a storage administrator. Accessing block storage volumes, as used herein, may refer to obtaining, examining, or retrieving data.

[00119] As described with relation to process 210 of Fig. 2B (beginning with step 211 to initiate a connection to cloud infrastructure 106), scanning system 101 may send a second message to user device 102, instructing the user to generate a role. A role may refer to a type associated with a node that may be assigned or defined. The second message may include instructions for the user to follow to generate the role. In step 213, a user may provide (e.g., via a keyboard at user device 102) a role definition to the cloud service provider's system. In some embodiments, the role definition includes read-only permissions and permissions to read a block storage layer (containing block storage volumes). In some embodiments, scanning system 101 provides a role formation template (e.g., an Amazon Web Services CloudFormation

Template) for use with cloud infrastructure 106 to create the necessary role. In step 213, the user may utilize user device 102, for example, by copying and pasting a URL of the template, downloading and uploading the template to the cloud service provider's system, or selecting the template from a list of templates.

[00120] By way of example, scanning system 101 of Fig. 1 may access the block storage volume of any of workloads virtual machines 107A-107D, databases 109A-109D, storage 111A-111D, keystores 113A-113D of cloud infrastructure 106.

[00121] Aspects of this disclosure may include determining a file-system of the at least one workload. A file-system, as used herein, may refer to a method and/or a data structure that the operating system uses to control how data is stored and retrieved.

[00122] By way of example, scanning system 101 of Fig. 1 may identify the type of file-system of any of workloads virtual machines 107A-107D, databases 109A-109D, storage 111A-111D, keystores 113A-113D of cloud infrastructure 106. Some file systems may include NTFS, FAT32, ext3, exFAT, HFS, HFS+, HPFC, UFS, ZFS.

[00123] Aspects of this disclosure may include mounting the block storage volumes on a scanner based on the determined file-system. Mounting the block storage volumes, as used herein, may refer to a process by which the operating system makes files and directories (such as block storage volumes) on a storage device available for users to access via the computer's file system. A scanner, as used herein, may refer to a device for examining, reading, or monitoring electronically accessible information. By way of example, Fig. 1 illustrates one example of a scanner, scanning system 101 of Fig. 1. In some embodiments, mounting may include selecting a driver corresponding to the determined file system. In some embodiments, to mount the block storage volumes on the scanner, may include creating a snapshot of the block storage volumes; and mounting the snapshot of the block storage volumes on the scanner. A snapshot of the block storage volumes, as used herein, may refer to a current state or earlier state of the block storage volumes.

[00124] In some embodiments, the at least one processor may be further configured to encrypt the snapshot of the block storage volumes and mount the encrypted snapshot of the block storage volumes on the scanner. Encrypting a snapshot, as used herein, may refer to a process of encoding information (such as a snapshot). This process may convert the original representation of the information, known as plaintext, into an alternative form known as ciphertext.

[00125] In some embodiments, the scanner may use a privileged account to log in and determine how secure each host is from an inside vantage point. While authenticated scans can successfully discover potential vulnerabilities, they may be limited to the extent they require a privileged account on each scanned host. Furthermore, scans use significant system resources during the test procedures and require opening ports that by themselves pose a security risk.

[00126] Aspects of this disclosure may include activating a scanner at the geographic location. Activating a scanner, as used herein, may refer to starting up or initiating a device for examining, reading, or monitoring data. By way of example, scanning system 101 of Fig. 1 may start up a scanner (a different scanner than scanning system 101) at the geographic location identified by scanning system 101.

[00127] Aspects of this disclosure may include reconstructing from the block storage volumes a state of the workload. Reconstructing, as used herein, may refer to rebuilding or reforming a state of the workload. A state of the workload, as used herein, may refer to a condition of a workload at a certain time. By way of example, scanning system 101 of Fig. 1 may rebuild the block storage volumes as an earlier version of any of workloads such as virtual machines 107A-107D, databases 109A-109D, storage 111A-111D, keystores 113A-113D of cloud infrastructure 106.

[00128] In some embodiments, the reconstructed state of the workload may include at least two of an indication of an installed application, a version of an installed application, an operating system configuration, an application configuration, a profile configuration, a log, or a database content. An indication of an installed application, as used herein, may refer to an identifier of a software. A version of an installed application, as used herein, may refer to a particular form, variant, edition, or revision of a piece or pieces of software. An operating system configuration, as used herein, may refer to a manner in which components are arranged to make up an operating system or a computer system. An application configuration, as used herein, may refer to a manner in which components are arranged to make up an application or a computer system. A profile configuration, as used herein, may refer to a manner in which components are arranged to make up a profile or a computer system. A log, as used herein, may refer to a detailed list of an application information, system performance, or user activities. Database content, as used herein, may refer to any collection of data, or information, that is specially organized for rapid search and retrieval by a computer. In some embodiments, scanning system 101 of Fig. 1 may reconstruct any workload, such as virtual machines 107A-107D, databases 109A-109D, storage 111A-111D, keystores 113A-113D of cloud infrastructure 106 from the block storage volumes to an indication of an installed application, a version of an installed application, an operating system configuration, an application configuration, a profile configuration, a log, or a database content.

[00129] Aspects of this disclosure may include assessing the reconstructed state of the workload to extract insights. Assessing the reconstructed state of the workload, as used herein, may refer viewing, utilizing, or evaluating the reconstructed state of the workload. Insights, as used herein, may refer to information or knowledge attained based on an analysis of some data. Extracting insights, as used herein, may refer to pulling information or data about, for example, a vulnerability associated with the workload or a composition of installed applications associated

with the workload. By way of example, scanning system 101 of Fig. 1 may evaluate the reconstructed state of any of workloads, such as virtual machines 107A-107D, databases 109A-109D, storage 111A-111D, keystores 113A-113D of cloud infrastructure 106 to pull valuable data and information.

[00130] In some embodiments, insights may include at least one of a vulnerability associated with the workload or a composition of installed applications associated with the workload. A vulnerability, as used herein, may refer to any weakness within an organization's information systems, internal controls, or system processes that can be exploited. A composition of installed applications associated with the workload, as used herein, may refer to a structure or group of accessible computer programs. By way of example, scanning system 101 of Fig. 1 may evaluate the reconstructed state of any workloads such as virtual machines 107A-107D, databases 109A-109D, storage 111A-111D, keystores 113A-113D of cloud infrastructure 106 to pull valuable data and information regarding a vulnerability with the workload.

[00131] In some embodiments, a scanner may be deployed at the geographical location. Deploying a scanner, as used herein, may refer to installing, activating, utilizing, or bringing a scanner into effective action . By way of example, Fig. 1 illustrates one example of a scanner, scanning system 101 of Fig. 1 that is deployed.

[00132] In some embodiments, scanning system 101 of Fig. 1 may update the reconstructed state of the workload based on at least one change to the block storage volumes . Updating the reconstructed state of the workload, as used herein, may refer to overhauling or refurbishing the reconstructed state of the workload. A change to the block storage volumes, as used herein, may refer to any shift, alteration, or modification to the block storage volumes.

[00133] **Fig. 3** illustrates a block diagram of method 300 for insight extraction, consistent with disclosed embodiments. In some embodiments, the method may include ten (or more or less) steps:

[00134] Block 302: Establish a trusted relationship between a source account in a cloud environment and a scanner account. In some embodiments, establishing a trusted relationship between a source account in a cloud environment and a scanner account may include an integration process (creating a connection between an account on scanning system 101 and an account on cloud infrastructure 106) discussed with reference to Fig. 2A. By way of one example, scanning system 101 of Fig. 1 may create a trusted relationship between a source account in a cloud environment and a scanner account.

[00135] Block 304: Using the established trust relationship, utilize at least one cloud provider API to identify workloads in the source account. In some embodiments, scanning system 101 of Fig. 1 may use an API to detect workloads such as virtual machines 107A-107D,

databases 109A-109D, storage 111A-111D, keystores 113A-113D, and load balancer 115 in the source account.

[00136] Block 306: Use the at least one cloud provider API to query a geographical location of at least one of the identified workloads. In some embodiments, scanning system 101 of Fig. 1 may query any of workloads virtual machines 107A-107D, databases 109A-109D, storage 111A-111D, keystores 113A-113D of cloud infrastructure 106 for their geographical location using a cloud provider API.

[00137] Block 308: Receive an identification of the geographic location. In some embodiments, scanning system 101 may indicate in the map additional information regarding the workload (e.g., a system or device). For example, scanning system 101 may provide: an identification of a data center (e.g., a name or number associated with the data center where the workload is located), at least one of a data center name (e.g., a name or address associated with the data center where the workload is located), Internet Protocol (IP) address, name of the cloud provider, or a unique identity (e.g., any other information related to the workload that may be used).

[00138] Block 310: Use the cloud provider APIs to access block storage volumes of the at least one workload. In some embodiments, scanning system 101 of Fig. 1 may access the block storage volume of workloads such as virtual machines 107A-107D, databases 109A-109D, storage 111A-111D, keystores 113A-113D of cloud infrastructure 106. In some embodiments, accessing the block storage may include taking the mean of generating snapshots.

[00139] Block 312: Determine a file-system of the at least one workload. In some embodiments, scanning system 101 of Fig. 1 may identify the type of file-system any of workloads (virtual machines 107A-107D, databases 109A-109D, storage 111A-111D, keystores 113A-113D of cloud infrastructure 106).

[00140] Block 314: Mount the block storage volumes on a scanner based on the determined file-system. In some embodiments, to mount the block storage volumes on the scanner, may include instructing the operating system to map a directory structure logically to a physical storage device. A storage volume may be mounted after it is attached and formatted for use by a server's operating system. Mounting may include creating a snapshot of the block storage..

[00141] Block 316: Activate a scanner at the geographic location. In some embodiments, scanning system 101 of Fig. 1 may start up a scanner (a different scanner than scanning system 101) at the geographic location identified by scanning system 101.

[00142] Block 318: Reconstruct from the block storage volumes a state of the workload. In some embodiments, scanning system 101 of Fig. 1 may rebuild the block storage volumes to

an earlier version of workloads such as virtual machines 107A-107D, databases 109A-109D, storage 111A-111D, keystores 113A-113D of cloud infrastructure 106.

[00143] Block 320: Assess the reconstructed state of the workload to extract insights. In some embodiments, scanning system 101 of Fig. 1 may evaluate the reconstructed state of workloads such as virtual machines 107A-107D, databases 109A-109D, storage 111A-111D, keystores 113A-113D of cloud infrastructure 106 to derive valuable data and information.

[00144] **Vulnerability Management Techniques**

[00145] **Fig. 4** depicts a cybersecurity system performing a side scanning function to protect against potential vulnerabilities. Referring to Fig. 4, a processor 400 configured to identify in a cloud environment one or more block storage volumes 401A-C communicatively connected to a cloud provider API where the block storage volumes 401A-C may be contained. When a block storage volume, e.g., 401A is identified, the processor 400 may be configured to perform the function of identifying the software installed [[403A]] 403A-C in or relating to block storage volume 401. Upon identifying installed software [[403A]] 403A-C, the processor may be further configured to identify the version of software [[405A]] 405A-C operating in the installed software ~~403A~~ 403A-C.. Based on the installed software ~~403A~~ 403A-C and the version of the software ~~405A~~ 405A-C, the processor may be configured to identify a listing of known vulnerabilities ~~407~~ 407A-C that may relate to the installed software ~~403A~~ 403A-C and the version installed ~~405A~~ 405A-C. When the listing of known vulnerabilities 407 is obtained, the system may list the known vulnerabilities ~~409A~~ 409A-C for that software version ~~411A~~ 411A-C, which can be communicated to the processor 400 of the end user. This information may be processed and displayed to an end user using processor 400.

[00146] **Fig. 5** of the disclosed embodiments describes a method of operating a cybersecurity system performing a side scanning function to protect against potential vulnerabilities. Referring to Fig. 5, a processor unit 500 may be configured to perform the method . When initiated either manually or automatically, processor unit 500 may be communicatively connected with a cloud provider API 501 to initiate a cybersecurity function as disclosed above. Contained digitally on a memory device and accessed by a cloud provider API 501 is a series of block storage volumes 503. Processor unit 500 may be capable of performing a vertical and horizontal scan for network access or security information of a block storage volume 503 to detect potential vulnerabilities to the block storage volume 503 or any processing unit 500 designed to access said block storage volume 503.

[00147] Based on the accessed block storage volume 500, processor unit 500 with the disclosed embodiments of a cybersecurity system may perform an identification of the type of software installed 505 connected with block storage volume 503. Following identification of the type of software installed 505, the processing unit may then identify an installed version 507 of

said installed software 505 for a version comprised of a unique identifier based on a combination of letters, numbers, or similar unique identifiers.

[00148] Upon recognition of an installed software version 507 of the installed software 505, processing unit 500 with the disclosed embodiments of the disclosed scanning system 101 may provide a list of known vulnerabilities 509 for review by said scanning system 101 and/or its end user and maintainer of said block storage volumes 503. Based on the list of known vulnerabilities 509, the disclosed scanning system 101 or its end user may identify one or more ports of accessibility to said block storage volume 503 that may be accessed by a known, associated vulnerability 513 from listed vulnerabilities 509 to determine an avenue for potential vulnerability 513 to access and infiltrate block storage volume 503.

[00149] In some embodiments, a cyber security scanning system for a cloud environment 101 may include a processor 500 to operate said system. This processor 500 may include central processing units and other similar computer-enabling equipment for processing and executing commands based on the information inputted to said system. The processor 500 may be communicatively connected to a computer network or series of networks to accomplish said cyber security function.

[00150] As an example embodiment, a processor unit 500 may be configured to use a cloud-provider API that may communicate with one or more specified computer-readable media across a digital network. This can be accomplished through internet protocols, internet control message protocols, transmission control protocol, or user datagram protocol. Cloud provider API 501 may be one of several forms of middleware, interface, middle layer, or other systems of interfacing applications. A processor 300 may be one or more computer processing units, central processing unit, server, microcomputer, mainframe, and any other manifestation of digital computing.

[00151] Further to one of several possible embodiments, a cloud-provider API 501 may be configured to access a block-storage volume 503 of a workload maintained in a cloud-storage environment. This may be accomplished through a system of computer-readable media communicatively connected. Said block-storage volume 303 may be contained on a Storage Area Network (SAN) or similar cloud-based memory storage environment. The block storage volume 503 may be contained in smaller storage volumes with an associated identifier unique to that portion of said block storage volume 503. In some embodiments, the block-storage volume 503 of a workload may have multiple paths for the storage volume to be reaggregated and retrieved quickly.

[00152] Among several embodiments, a scanning system 101 may comprise a system for identifying an installed software application in the accessed block-storage volume 503. This identification of installed software may be accomplished by accessing installed software 505 files through signature verification, root license, or authorized user lists. The installed software

505 may be located and identified within applications such as file storage, database storage, and virtual machine file system volumes. The identification of said installed software application 507 may be processed, analyzed, and communicated to the scanning system 101 for processing, cataloging, and protection through encryption and various methods of layered cyber defense.

[00153] Further, the scanning system 101 described herein may include functionality to analyze installed software applications to determine the associated software version. The software application version 507 may identify the software version based on unique version name, unique version number, and may be based on unique states of the currently installed computer software 505.

[00154] One of many embodiments disclosed above may include the scanning system 101 having the ability to access a data structure of known software vulnerabilities 509 for a plurality of versions of software applications. The known software vulnerabilities 509 may include, among others, missing data encryption, OS command injection, SQL injection, buffer overflow, missing authentication, missing authorization, unrestricted upload of dangerous file types, reliance on untrusted inputs in a security decision, cross-site scripting and forgery, download of codes without integrity checks, broken algorithms, URL redirection, path traversal, software bugs, weak passwords, and previously infected software. The scanning system 101 may be able to access and identify software vulnerabilities for mitigation, rectification, correction, and fortification.

[00155] In one embodiment, a cybersecurity system may also perform scanning according to scanning system 101 by performing a lookup of the identified installed software version 507 in the data structure to identify known vulnerabilities 509. This function can be performed by the scanning system 101 according to Fig. 1 performing a query of the installed software 505 for unique version number or designator and comparing to, amongst many things, a set of likely or potential vulnerabilities to that software version for potential deficiencies or cybersecurity threats known or suspected to similar software types and versions. This query may be performed according to a predetermined set of values, to include previously identified unique version numbers or designators that may contain the known list of previously identified vulnerabilities.

[00156] Among many embodiments, embodiments of the disclosed scanning system 101 may query the cloud provider API 501 to determine network accessibility information 511 related to the workload. In order to accomplish this query of the cloud provider API 301, the scanning system 101 may involve index of search results and display of said search results, followed by processing and grouping search results. Network accessibility information 511 may include connection quality, alternative paths between nodes in a network, and the ability to avoid blockage in said networks. The workloads associated with this query may include applications,

services, capabilities, and specific processes such as virtual machines, databases, containers, or Hadoop nodes, among others.

[00157] If the system detects a vulnerable application 513, one embodiment may identify one or more port on which said vulnerable application is accessible. In one of several embodiments, the scanning system 101 may detect a vulnerable application in one or more computation process. In another embodiment, the cybersecurity system may perform a network accessibility query in a separate process. Further, a disclosed embodiment may perform these separate functions in subsequent and sequential steps of the same process. A person having skill would understand that an authorized user or an authorized scanning system 101 can perform these functions concurrently and subsequently by an authorized user or an authorized cybersecurity system while performing the same function as the disclosed embodiment.

[00158] Upon gathering network accessibility information and the identified port to identify one or more vulnerabilities susceptible to attack from outside the workload, a disclosed embodiment would have the functionality to perform processes to gather, display, and mitigate a discovered vulnerability in order to minimize the likelihood and effectiveness of a cyber threat outside of and attempting to access a workload through a known or previously encountered type of cyber threat. This functionality may include collecting and organizing the vulnerabilities according to type or category of vulnerability, displaying the gathered data for an end user or maintainer, and implementing security features automatically or manually by a user or maintainer such as security patches, password or passcode changes and suggestions for users to do the same, and malicious code eradication.

[00159] As one of several possible embodiments, the scanning system 101 may also, upon identification of one or more vulnerabilities, implement remedial actions via one or more processors. Remedial actions may include, among other things, notification to an end user of an identified threat, compensation through a revised security code to mitigate the potential threat, publication of the identified threat and vulnerability in a log or record of detected vulnerabilities, and communication of the sensed vulnerability and threat to a server operator or maintainer to fortify the protections of workloads existing on similar environments.

[00160] In a similar embodiment to above, the remedial measure may include transmission of an alert to a device associated with an administrator. The alert may be, amongst others, written, auditory, and visual for processing and use by an administrator of said scanning system 101. Said administrator may take action based on the received alert, to include eliminating the cyber threat through mitigation measures, change in cybersecurity posture, or removing the workload from the cybersecurity threat environment.

[00161] In another embodiment, a query of the cloud provider API 501 to determine network accessibility information related to the workload may be performed by at least one processor 500 configured to examine data sources associated with the workload. The data

examined may include user data, system processing data, accessibility data, clock cycles, storage input/output, or similar data processors. A query of the cloud provider API 501 may be automated or manually-initiated. Based on the said query, the network accessibility information 511 related to the workload may change based on the data sources associated with said workload.

[00162] In the above processor and similar embodiments, further configuration may include a process to determine network accessibility information 511 based on the examined data sources. The examined data sources may include various cloud-based workloads, internet protocols, transmission communication protocols, or other methods and systems of memory and data storage.

[00163] As an exemplary embodiment, network accessibility information 511 includes at least one of: data from an external data source, cloud provider information, or at least one network capture log. These embodiments of an external data source may include data from the operating environment of the cloud-based environment, an external operating system for a computer processing unit 500, or other similar computer readable media. Cloud provider information may further include information that may identify the network accessibility information 511 vertically or horizontally to fully describe the associated workload. Network capture logs may be automated or manually updated to include possible vulnerabilities 513 and threats to the cloud-based storage medium.

[00164] Further, the disclosed embodiments may include an installed software application 505, with the at least one processor 500 configured to extract data from at least one of operating system packages, libraries, or program language libraries. This data may be extracted through a system query, random access algorithm, or similar automated process. Operating system packages may include systems operable on Microsoft, Apple, Linux, and similar operating systems. Libraries may consist of a series of files, folders, and databases of information stored on one of any indexed data repositories. A program language library may contain several of an exemplary program languages including but not limited to Javascript, Swift, Scala, Go, Python, Elm, Ruby, C#, C++ and other similar sources of software code.

[00165] As another exemplary embodiment, a scanning system 101 may also include a processor 500 configured to identify installed software application 505 based on the extracted data. The processor 500 may perform only this function or this function among many to accomplish the layered cybersecurity defense described herein this disclosed cloud-based security environment. The identification of installed software by said processor may include identifying the software by unique version number or designator, recognized source code, metadata associated with the installed software application files, or similar software-identifying information.

[00166] One embodiment may include an additional function wherein the installed software application 505 that has been identified includes one or more scripts. These scripts may be processed through various computer readable languages to include Javascript, C#, C++ and other forms of computer code.

[00167] One embodiment contemplated by the disclosed scanning system 101 may also include data structure includes aggregated vulnerability data 309. This aggregated vulnerability data 509 may be compiled by an end user or maintainer from within the cloud-based environment of the current data structure as well as compilation from similar data sets and aggregation of common threats to data structures likely to experience similar vulnerabilities. This aggregation of vulnerability data 509 may be contained within the data structure and it may be collectively aggregated to provide for a more robust and layered cybersecurity defense posture.

[00168] In an embodiment of the disclosed scanning system 101, the aggregated vulnerability data 509 may include data from one or more third-party vendors. These vendors may include operators of the cloud-based server environment, providers of networking and internet communication, methods of layered authentication, and other similar providers of services directly related and in communication with the cloud-based cybersecurity system.

[00169] As an additional exemplary embodiment, the aggregated vulnerability data 309 may include data collected by a scanner. This scanner may involve use of continuous or periodic monitoring of the workload. The scanner may perform security screenings of the various workloads vertically or horizontally to identify network identification information, port accessibility, and associated vulnerabilities. Any scan performed may be communicated to the scanning system 101 that may be responsible for performing and logging the results of the scan and may be able to initiate follow-on processes and protocols to protect the data contained in the workload that is the subject of scanning.

[00170] An embodiment of the disclosed scanning system 101 may also include aggregated vulnerability data 509 that may include at least one of an advisory, an exploit, a security announcement, or a known bug. An advisory may include notification to a system maintainer or user of the potential vulnerability, may log notice of the advisory, and may recommend possible user or maintainer actions to potentially address said advisory. An exploit may further consist of an automated system response designed to take advantage of the sensed vulnerability data. The exploit can be further reflected in the aggregated vulnerability data and protocols can be written into the cybersecurity infrastructure to prevent said exploit from gaining access and permissions to unauthorized areas of the workload storage environment.

[00171] A person having ordinary skill in the art would appreciate the above described embodiments are among many potential embodiments, to include a method of operating a

scanning system 101 similar to the one described above. A disclosed embodiment contemplates this method to be accomplished through manual user operation, automated computer processes, or similar manners of operation. These manners of operation and those contemplated similar to them would allow the scanning system 101 described and disclosed to execute its operations as the system above describes.

[00172] One of several embodiments of the disclosed method of operating a scanning system 101 may include a cyber security scanning system for a cloud environment 101 may include a processor 500 to operate said system. This processor may include central processing units and other similar computer-enabling equipment for processing and executing commands based on the information inputted to said system. The processor may be communicatively connected to a computer network or series of networks to accomplish said cyber security function.

[00173] As an example embodiment, a processor unit 500 may be configured to use a cloud-provider API 501 that may communicate with one or more specified computer-readable media across a digital network. This can be accomplished through internet protocols, internet control message protocols, transmission control protocol, or user datagram protocol. Cloud provider API 501 may be one of several forms of middleware, interface, middle layer, or other systems of interfacing applications. A processor may be one or more computer processing units, central processing unit, server, microcomputer, mainframe, and any other manifestation of digital computing.

[00174] Further to one of several possible embodiments, a cloud-provider API 501 may be configured to access a block-storage volume 503 of a workload maintained in a cloud-storage environment. This may be accomplished through a system of computer-readable media communicatively connected. Said block-storage volume 503 may be contained on a Storage Area Network (SAN) or similar cloud-based memory storage environment. The block storage volume 503 may be contained in smaller storage volumes with an associated identifier unique to that portion of said block storage volume 503. In some embodiments, the block-storage volume 503 of a workload may have multiple paths for the storage volume to be reaggregated and retrieved quickly.

[00175] Among several embodiments, a method of operating a scanning system 101 may comprise a system for identifying an installed software application 505 in the accessed block-storage volume 503. This identification of installed software 505 may be accomplished by accessing installed software files through signature verification, root license, or authorized user lists. The installed software 505 may be located and identified within applications such as file storage, database storage, and virtual machine file system volumes. The identification of said installed software application 505 may be processed, analyzed, and communicated to the

method of operating a scanning system 101 for processing, cataloging, and protection through encryption and various methods of layered cyber defense.

[00176] Further, a method of operating a scanning system 101 described herein may include functionality to analyze installed software applications 505 to determine the associated software version 507. The software application version 507 may identify the software version based on unique version name, unique version number, and may be based on unique states of the currently installed computer software.

[00177] One of many embodiments disclosed above may include the method of operating a scanning system 101 having the ability to access a data structure of known software vulnerabilities 509 for a plurality of versions of software applications 507. The known software vulnerabilities 509 may include, among others, missing data encryption, OS command injection, SQL injection, buffer overflow, missing authentication, missing authorization, unrestricted upload of dangerous file types, reliance on untrusted inputs in a security decision, cross-site scripting and forgery, download of codes without integrity checks, broken algorithms, URL redirection, path traversal, software bugs, weak passwords, and previously infected software. The cyber security system may be able to access and identify software vulnerabilities 309 for mitigation, rectification, correction, and fortification.

[00178] In one embodiment, a method of operating a scanning system 101 may also perform scanning according to scanning system 101 by performing a lookup of the identified installed software version 507 in the data structure to identify known vulnerabilities 309. This function can be performed by the scanning system 101 according to Fig. 1 performing a query of the installed software 505 for unique version number 507 or designator and comparing to, amongst many things, a set of likely or potential vulnerabilities to that software version for potential deficiencies or cybersecurity threats known or suspected to similar software types and versions. This query may be performed according to a predetermined set of values, to include previously identified unique version numbers or designators that may contain the known list of previously identified vulnerabilities.

[00179] Among many embodiments, embodiments of the disclosed method of operating a scanning system 101 may query the cloud provider API 501 to determine network accessibility information 511 related to the workload. In order to accomplish this query of the cloud provider API 501, the scanning system 101 may involve index of search results and display of said search results, followed by processing and grouping search results. Network accessibility information 511 may include connection quality, alternative paths between nodes in a network, and the ability to avoid blockage in said networks. The workloads associated with this query may include applications, services, capabilities, and specific processes such as virtual machines, databases, containers, or Hadoop nodes, among others.

[00180] If the system detects a vulnerable application, one embodiment may identify one or more port on which said vulnerable application is accessible. In one of several embodiments, the scanning system 101 may detect a vulnerable application in one or more computation process. In another embodiment, the scanning system 101 may perform a network accessibility query in a separate process. Further, a disclosed embodiment may perform these separate functions in subsequent and sequential steps of the same process. A person having skill would understand that an authorized user or an authorized cybersecurity system may perform these functions concurrently and subsequently while performing the same function as the disclosed embodiment.

[00181] Upon gathering network accessibility information 511 and the identified port 513 to identify one or more vulnerabilities susceptible to attack from outside the workload, a disclosed embodiment would have the functionality to perform processes to gather, display, and mitigate a discovered vulnerability in order to minimize the likelihood and effectiveness of a cyber threat outside of and attempting to access a workload through a known or previously encountered type of cyber threat. This functionality may include collecting and organizing the vulnerabilities according to type or category of vulnerability, displaying the gathered data for an end user or maintainer, and implementing security features automatically or manually by a user or maintainer such as security patches, password or passcode changes and suggestions for users to do the same, and malicious code eradication.

[00182] A person having ordinary skill in the art would similarly understand that the above disclosed system can be disclosed using a suitable non-transitory computer-readable medium performing each of the disclosed functions. The disclosed embodiment may include a non-transitory computer readable medium with a scanning system 101 similar to the one described above on a non-transitory computer-readable medium. A disclosed embodiment contemplates this system to be accomplished through a medium that may contain a central processing unit, virtual machine, or a similar non-transitory medium. These manners of operation and those contemplated similar to them would allow the scanning system 101 described and disclosed to execute its operations as the system above describes.

[00183] One of several embodiments of the disclosed non-transitory computer readable medium with a scanning system 101 may include a cyber security scanning system for a cloud environment 101 may include a processor 500 to operate said system. This processor 500 may include central processing units and other similar computer-enabling equipment for processing and executing commands based on the information inputted to said system. The processor 500 may be communicatively connected to a computer network or series of networks to accomplish said cyber security function.

[00184] As an example embodiment, a processor unit 300 may be configured to use a cloud-provider API 501 that may communicate with one or more specified computer-readable

media across a digital network. This can be accomplished through internet protocols, internet control message protocols, transmission control protocol, or user datagram protocol. Cloud provider API 501 may be one of several forms of middleware, interface, middle layer, or other systems of interfacing applications. A processor 500 may be one or more computer processing units, central processing unit, server, microcomputer, mainframe, and any other manifestation of digital computing.

[00185] Further to one of several possible embodiments, a cloud-provider API 501 may be configured to access a block-storage volume 503 of a workload maintained in a cloud-storage environment. This may be accomplished through a system of computer-readable media communicatively connected. Said block-storage volume 503 may be contained on a Storage Area Network (SAN) or similar cloud-based memory storage environment. The block storage volume 503 may be contained in smaller storage volumes with an associated identifier unique to that portion of said block storage volume 503. In some embodiments, the block-storage volume 503 of a workload may have multiple paths for the storage volume to be reaggregated and retrieved quickly.

[00186] Among several embodiments, a non-transitory computer readable medium with a scanning system 101 may comprise a system for identifying an installed software application 505 in the accessed block-storage volume 503. This identification of installed software 505 may be accomplished by accessing installed software files through signature verification, root license, or authorized user lists. The installed software 505 may be located and identified within applications such as file storage, database storage, and virtual machine file system volumes. The identification of said installed software application 505 may be processed, analyzed, and communicated to the method of operating a scanning system 101 for processing, cataloging, and protection through encryption and various methods of layered cyber defense.

[00187] Further, a non-transitory computer readable medium with a scanning system 101 described herein may include functionality to analyze installed software applications 505 to determine the associated software version 507. The software application version 507 may identify the software version based on unique version name, unique version number, and may be based on unique states of the currently installed computer software.

[00188] One of many embodiments disclosed above may include the non-transitory computer readable medium with a scanning system 101 having the ability to access a data structure of known software vulnerabilities 509 for a plurality of versions of software applications 507. The known software vulnerabilities 509 may include, among others, missing data encryption, OS command injection, SQL injection, buffer overflow, missing authentication, missing authorization, unrestricted upload of dangerous file types, reliance on untrusted inputs in a security decision, cross-site scripting and forgery, download of codes without integrity checks, broken algorithms, URL redirection, path traversal, software bugs, weak passwords,

and previously infected software. The scanning system 101 may be able to access and identify software vulnerabilities for mitigation, rectification, correction, and fortification.

[00189] In one embodiment, a non-transitory computer readable medium with a scanning system 101 may also perform scanning according to scanning system 101 by performing a lookup of the identified installed software version in the data structure to identify known vulnerabilities 509. This function can be performed by the scanning system 101 according to Fig. 1 performing a query of the installed software for unique version number 307or designator and comparing to, amongst many things, a set of likely or potential vulnerabilities 309 to that software version 307 for potential deficiencies or cybersecurity threats known or suspected to similar software types and versions. This query may be performed according to a predetermined set of values, to include previously identified unique version numbers or designators that may contain the known list of previously identified vulnerabilities 509.

[00190] Among many embodiments, embodiments of the disclosed non-transitory computer readable medium with a scanning system 101 may query the cloud provider API 501 to determine network accessibility information 511 related to the workload. In order to accomplish this query of the cloud provider API 501, the scanning system 101 may involve index of search results and display of said search results, followed by processing and grouping search results. Network accessibility information 511 may include connection quality, alternative paths between nodes in a network, and the ability to avoid blockage in said networks. The workloads associated with this query may include applications, services, capabilities, and specific processes such as virtual machines, databases, containers, or Hadoop nodes, among others.

[00191] If the system detects a vulnerable application, one embodiment may identify one or more ports 515 on which said vulnerable application is accessible. In one of several embodiments, the scanning system 101 may detect a vulnerable application in one or more computation processes. In another embodiment, the scanning system 101 may perform a network accessibility query in a separate process. Further, a disclosed embodiment may perform these separate functions in subsequent and sequential steps of the same process. A person having skill would understand that an authorized user or an authorized cybersecurity system can perform these functions concurrently and subsequently while performing the same function as the disclosed embodiment.

[00192] Upon gathering network accessibility information 511 and the identified port 515 to identify one or more vulnerabilities susceptible to attack from outside the workload, a disclosed embodiment would have the functionality to perform processes to gather, display, and mitigate a discovered vulnerability in order to minimize the likelihood and effectiveness of a cyber threat outside of and attempting to access a workload through a known or previously encountered type of cyber threat. This functionality may include collecting and organizing the

vulnerabilities according to type or category of vulnerability, displaying the gathered data for an end user or maintainer, and implementing security features automatically or manually by a user or maintainer such as security patches, password or passcode changes and suggestions for users to do the same, and malicious code eradication.

[00193] Another disclosed embodiment may include a method of using a cloud provider API 501, accessing block storage volume 503 of a workload maintained in a cloud storage environment. The method may include manual user operation, automated system operation, systematic and random operating parameters for the system to provide its cybersecurity and similar security functions. The accessed block storage volume may be contained as a collection of block units organized together, may be a series of individual blocks that can be reorganized to form a new storage volume, and may have the ability to be disaggregated and reaggregated as necessary to accomplish its storage and cybersecurity functions.

[00194] The disclosed method may further comprise a system that may analyze the identified installed software application 505 to determine an associated software version 507. In analyzing an installed software application 305, the disclosed embodiment may include a method of querying the installed software application to access a software version 507 that may include a series of letters, numbers, and other identifying characters to differentiate the version of software currently operative and its associated identifying characteristics and capabilities.

[00195] A method of the disclosed may also include accessing a data structure of known software vulnerabilities 509 for a plurality of versions of software applications 507. The disclosed method may determine a data structure of known software vulnerabilities 509 of one among many operative versions of software installed in the monitored storage system as compared to historical data of similar storage systems.

[00196] The disclosed method as described herein may further include performing a lookup of the identified installed software version 507 in the data structure to identify known vulnerabilities 509. The lookup of identified installed software 507 may be previously indexed based on unique version identifier, and the indexed information may include known vulnerability information 509, network accessibility information 511, network protocols, and administrator identifier information. Based on the said indexed known vulnerabilities 509, the scanning system 101 disclosed may take actions to mitigate a cybersecurity threat and reinforce cybersecurity defenses at the type of cybersecurity threat, as previously discussed.

[00197] Further to the above, a disclosed method may also include querying the cloud provider API 501 to determine network accessibility information 511 related to the workload. A query of the cloud provider API 501 to determine network accessibility information 511 related to the workload may be automatically or manually initiated. Network accessibility information 511 similar to the above embodiment may include connection quality, alternative paths between nodes in a network, and the ability to avoid blockage in said networks. The workloads

associated with this query may include applications, services, capabilities, and specific processes such as virtual machines, databases, containers, and Hadoop nodes, among others.

[00198] A disclosed method of one embodiment also may provide a method of identifying at least one port 515 on which the vulnerable application is accessible. The identification of one port 515 on which the vulnerable application is accessible may be for the purpose of identifying a cyber weakness, and it may include the ability to change the port currently accessible to mitigate the said cyber weakness.

[00199] Yet another disclosed embodiment of a disclosed method may include using the network accessibility information 511 and the identified at least one port 515 to identify one or more vulnerabilities 513 susceptible to attack from outside the workload. The vulnerability 513 identified as susceptible to attack from outside the workload may include, among others, missing data encryption, OS command injection, SQL injection, buffer overflow, missing authentication, missing authorization, unrestricted upload of dangerous file types, reliance on untrusted inputs in a security decision, cross-site scripting and forgery, download of codes without integrity checks, broken algorithms, URL redirection, path traversal, software bugs, weak passwords, and previously infected software.

[00200] A method of the disclosed embodiment may further include implementing a remedial action in response to the identified one or more vulnerabilities 513. Said remedial action may include, among other things, notification to an end user of an identified threat, compensation through a cybersecurity patch, publish of the identified threat and vulnerability in a log or record of detected vulnerabilities, and communication of the sensed vulnerability and threat to a server operator and maintainer to fortify the protections of workloads existing on similar environments.

[00201] The disclosed method may also include wherein the remedial measure includes transmitting an alert to a device associated with an administrator. Said alert may be, amongst others, written, auditory, and visual for processing and use by an administrator of said cybersecurity system. Said alert may further be logged and catalogued for future identification of known threats and vulnerabilities to similar software application versions.

[00202] A disclosed method of the present embodiment may also provide querying the cloud provider API 501 to determine the network accessibility information 311 related to the workload and examining data sources associated with the workload. A query of the cloud provider API 501 to determine network accessibility information 511 related to the workload may be automatically or manually initiated. Network accessibility information 511 similar to the above embodiment may include connection quality, alternative paths between nodes in a network, and the ability to avoid blockage in said networks. The workloads associated with this query may include applications, services, capabilities, and specific processes such as virtual machines, databases, containers, and Hadoop nodes, among others.