LICENSE FOR FOREIGN FILING UNDER

Title 35, United States Code, Section 184

Title 37, Code of Federal Regulations, 5.11 & 5.15

GRANTED

The applicant has been granted a license under 35 U.S.C. 184, if the phrase "IF REQUIRED, FOREIGN FILING LICENSE GRANTED" followed by a date appears on this form. Such licenses are issued in all applications where the conditions for issuance of a license have been met, regardless of whether or not a license may be required as set forth in 37 CFR 5.15. The scope and limitations of this license are set forth in 37 CFR 5.15(a) unless an earlier license has been issued under 37 CFR 5.15(b). The license is subject to revocation upon written notification. The date indicated is the effective date of the license, unless an earlier license of similar scope has been granted under 37 CFR 5.13 or 5.14.

This license is to be retained by the licensee and may be used at any time on or after the effective date thereof unless it is revoked. This license is automatically transferred to any related applications(s) filed under 37 CFR 1.53(d). This license is not retroactive.

The grant of a license does not in any way lessen the responsibility of a licensee for the security of the subject matter as imposed by any Government contract or the provisions of existing laws relating to espionage and the national security or the export of technical data. Licensees should apprise themselves of current regulations especially with respect to certain countries, of other agencies, particularly the Office of Defense Trade Controls, Department of State (with respect to Arms, Munitions and Implements of War (22 CFR 121-128)); the Bureau of Industry and Security, Department of Commerce (15 CFR parts 730-774); the Office of Foreign AssetsControl, Department of Treasury (31 CFR Parts 500+) and the Department of Energy.

NOT GRANTED

No license under 35 U.S.C. 184 has been granted at this time, if the phrase "IF REQUIRED, FOREIGN FILING LICENSE GRANTED" DOES NOT appear on this form. Applicant may still petition for a license under 37 CFR 5.12, if a license is desired before the expiration of 6 months from the filing date of the application. If 6 months has lapsed from the filing date of this application and the licensee has not received any indication of a secrecy order under 35 U.S.C. 181, the licensee may foreign file the application pursuant to 37 CFR 5.15(b).

SelectUSA

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 U.S. offers tremendous resources and advantages for those who invest and manufacture goods here. Through SelectUSA, our nation works to promote and facilitate business investment. SelectUSA provides information assistance to the international investor community; serves as an ombudsman for existing and potential investors; advocates on behalf of U.S. cities, states, and regions competing for global investment; and counsels U.S. economic development organizations on investment attraction best practices. To learn more about why the United States is the best country in the world to develop technology, manufacture products, deliver services, and grow your business, visit http://www.SelectUSA.gov or call +1-202-482-6800.

Docket No.: 007742.00091\US
(PATENT)

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Patent Application of:

Sean Moore

Application No.: 15/614,956 Confirmation No.: 9334

Filed: June 6, 2017 Art Unit: N/A

For: FILTERING NETWORK DATA TRANSFERS Examiner: Not Yet Assigned

RESPONSE TO NOTICE TO FILE MISSING PART OF APPLICATION
AND
PRELIMINARY AMENDMENT UNDER 37 C.F.R. § 1.115

Mail Stop, Missing Parts
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Dear Commissioner:

In response to the Notice to File Missing Parts of Application mailed June 15, 2017, Applicants respectfully submit the fees associated with the application. Additionally, Applicants petition for any extension of time needed for entry of this paper. Please charge our Deposit Account No. 19-0733 in the amount of $800.00, which covers the fees set forth in 37 C.F.R. § 1.16(f), 1.16(a), 1.16(k), and 1.16(o).

Prior to examination on the merits, please amend the above-identified U.S. patent application as follows:

Amendments to the Claims are reflected in the listing of claims, which begins on page 2 of this paper.

Remarks/Arguments begin on page 9 of this paper.

Listing of Claims:

This Listing of Claims will replace all prior versions and listings of claims in this application.

1. (Currently Amended) A method comprising:

receiving, by a computing system ~~and from a computing device located in a first network~~ comprising memory and at least one processor, a plurality of packets, wherein the plurality of packets comprises a first portion of packets and a second portion of packets;

determining, based on a packet header field value, whether each packet of the plurality of packets comprises data corresponding to criteria specified by one or more packet-filtering rules;

responsive to a determination by the computing system that a packet header field value of the first portion of packets comprises data corresponding to criteria specified by the one or more packet-filtering rules configured to prevent a particular type of data transfer ~~from the first network to a second network, wherein the data indicates that the first portion of packets is destined for the second network:~~ applying, by the computing system and to each packet in the first portion of packets, a first operator, specified by the one or more packet-filtering rules, configured to drop packets associated with the particular type of data transfer; ~~and dropping, by the computing system, each packet in first portion of packets; and~~

determining, based on an application header field value, whether the second portion of packets comprises data corresponding to criteria specified by one or more operators specified by the one or more packet-filtering rules; and

responsive to ~~a determination by the computing system that~~ determining that the second portion of packets comprises data corresponding ~~that does not correspond~~ to the criteria specified by one or more operators, ~~wherein the data indicates that the second portion of packets is destined for a third network:~~ applying, by the computing system and to each packet in the second portion of packets, ~~and without applying the one or more packet-filtering rules configured to prevent the particular type of data transfer from the first network to the second network, a second operator configured to forward packets not associated with the particular type of data transfer toward the third network; and forwarding, by the computing system, each packet in the second portion of packets toward the third network~~ at least one packet transformation function configured to allow

or block each packet in the second portion of packets from continuing toward its respective destination.

2. (New) The method of claim 1, wherein:

the first portion of packets comprises a packet header field value indicating a protocol type associated with the particular type of data transfer, and corresponding to the criteria specified by the one or more packet-filtering rules; and

the second portion of packets comprises data indicating a protocol type not associated with the particular type of data transfer.

3. (New) The method of claim 1, wherein:

the first portion of packets comprises data indicating a first destination port number associated with the particular type of data transfer, and corresponding to the criteria specified by the one or more packet-filtering rules; and

the second portion of packets comprises data indicating a second destination port number not associated with the particular type of data transfer.

4. (New) The method of claim 1, wherein the application header field value identifies one or more hypertext transfer protocol (HTTP) methods, wherein the second portion of packets comprise a plurality of HTTP packets, wherein the second portion of packets comprises a third portion of packets and a fourth portion of packets, and wherein each packet of the third portion of packets comprises at least one GET method value, and each packet of the fourth portion of packets comprises at least one of a PUT method value, a POST method value, or a CONNECT method value, the method further comprising:

allowing each packet of the third portion of packets to continue toward its respective destination; and

blocking each packet of the fourth portion of packets from continuing toward its respective destination.

5. (New) The method of claim 1, the method further comprising:

allowing, based on a determination that the application header field value indicates a request for specified resource data, each packet of the second portion of packets indicating a request for specified resource data to continue toward its respective destination; and

blocking, based on a determination that the application header field value indicates a request to submit data to be processed by a specified resource, each packet of the second portion of packets indicating a request to submit data to be processed by the specified resource from continuing toward its respective destination.

6. (New) The method of claim 1, wherein applying the first operator configured to drop packets associated with the particular type of data transfer is performed responsive to a determination that the first portion of packets comprises data corresponding to a particular transport layer security (TLS) version value.

7. (New) The method of claim 1, wherein applying the first operator configured to drop packets associated with the particular type of data transfer is performed responsive to a determination that the first portion of packets comprises data corresponding to a particular real-time transport protocol value.

8. (New) An apparatus comprising:

at least one processor; and

memory storing instructions that, when executed by the at least one processor, cause the apparatus to:

receive a plurality of packets, wherein the plurality of packets comprises a first portion of packets and a second portion of packets;

determine, based on a packet header field value, whether the plurality of packets comprises data corresponding to criteria specified by one or more packet-filtering rules;

responsive to a determination that a packet header field value of the first portion of packets comprises data corresponding to criteria specified by the one or more packet-filtering rules configured to prevent a particular type of data transfer, apply, to each packet

in the first portion of packets, a first operator, specified by the one or more packet-filtering rules, configured to drop packets associated with the particular type of data transfer;

determine, based on an application header field value, whether the second portion of packets comprises data corresponding to criteria specified by one or more operators specified by the one or more packet-filtering rules; and

responsive to determining that the second portion of packets comprises data corresponding to the criteria specified by one or more operators, apply, to each packet in the second portion of packets, at least one packet transformation function configured to allow or block each packet in the second portion of packets from continuing toward its respective destination.

9. (New) The apparatus of claim 8, wherein:

the first portion of packets comprises a packet header field value indicating a protocol type associated with the particular type of data transfer, and corresponding to the criteria specified by the one or more packet-filtering rules; and

the second portion of packets comprises data indicating a protocol type not associated with the particular type of data transfer.

10. (New) The apparatus of claim 8, wherein:

the first portion of packets comprises data indicating a first destination port number associated with the particular type of data transfer, and corresponding to the criteria specified by the one or more packet-filtering rules; and

the second portion of packets comprises data indicating a second destination port number not associated with the particular type of data transfer.

11. (New) The apparatus of claim 8, wherein the application header field value identifies one or more hypertext transfer protocol (HTTP) methods, wherein the second portion of packets comprise a plurality of HTTP packets, wherein the second portion of packets comprises a third portion of packets and a fourth portion of packets, and wherein each packet of the third portion of packets comprising at least one GET method value, and each packet of the fourth portion of packets

comprising at least one of a PUT method value, a POST method value, or a CONNECT method value, and wherein the instructions, when executed by the at least one processor cause the apparatus to:

allow each packet of the third portion of packets to continue toward its respective destination; and

block each packet of the fourth portion of packets from continuing toward its respective destination.

12. (New) The apparatus of claim 8, wherein the instructions, when executed by the one or more computing devices, cause the apparatus to:

allow, based on a determination that the application header field value indicates a request for data from a specified resource, each packet of the second portion of packets indicating a request for data from a specified resource to continue toward its respective destination; and

block, based on a determination that the application header field value indicates a request to submit data to be processed by a specified resource, each packet of the second portion of packets indicating a request to submit data to be processed by a specified from continuing toward its respective destination.

13. (New) The apparatus of claim 8, wherein applying the first operator configured to drop packets associated with the particular type of data transfer is performed responsive to a determination that the first portion of packets comprises data corresponding to a particular transport layer security (TLS) version value.

14. (New) The apparatus of claim 8, wherein applying the first operator configured to drop packets associated with the particular type of data transfer is performed responsive to a determination that the first portion of packets comprises data corresponding to a particular real-time transport protocol value.

15. (New) One or more non-transitory computer-readable media comprising instructions that, when executed by one or more computing devices, cause the one or more computing devices to:

receive a plurality of packets, wherein the plurality of packets comprises a first portion of packets and a second portion of packets;

determine, based on a packet header field value, whether the plurality of packets comprises data corresponding to criteria specified by one or more packet-filtering rules;

responsive to a determination that a packet header field value of the first portion of packets comprises data corresponding to criteria specified by the one or more packet-filtering rules configured to prevent a particular type of data transfer, apply, to each packet in the first portion of packets, a first operator, specified by the one or more packet-filtering rules, configured to drop packets associated with the particular type of data transfer;

determine, based on an application header field value, whether the second portion of packets comprises data corresponding to criteria specified by one or more operators specified by the one or more packet-filtering rules; and

responsive to determining that the second portion of packets comprises data corresponding to the criteria specified by one or more operators, apply, to each packet in the second portion of packets, at least one packet transformation function configured to allow or block each packet in the second portion of packets from continuing toward its respective destination.

16. (New) The one or more non-transitory computer-readable media of claim 15, wherein:

the first portion of packets comprises a packet header field value indicating a protocol type associated with the particular type of data transfer, and corresponding to the criteria specified by the one or more packet-filtering rules; and

the second portion of packets comprises data indicating a protocol type not associated with the particular type of data transfer.

17. (New) The one or more non-transitory computer-readable media of claim 15, wherein:

the first portion of packets comprises data indicating a first destination port number associated with the particular type of data transfer, and corresponding to the criteria specified by the one or more packet-filtering rules; and

the second portion of packets comprises data indicating a second destination port number not associated with the particular type of data transfer.

18. (New) The one or more non-transitory computer-readable media of claim 15, wherein the application header field value identifies one or more hypertext transfer protocol (HTTP) methods, wherein the second portion of packets comprise a plurality of HTTP packets, wherein the second portion of packets comprises a third portion of packets and a fourth portion of packets, and wherein each packet of the third portion of packets comprising at least one GET method value, and each packet of the fourth portion of packets comprising at least one of a PUT method value, a POST method value, or a CONNECT method value, and wherein the instructions, when executed by the one or more computing devices, cause the one or more computing devices to:

allow each packet of the third portion of packets to continue toward its respective destination; and

block each packet of the fourth portion of packets from continuing toward its respective destination.

19. (New) The one or more non-transitory computer-readable media of claim 15, wherein applying the first operator configured to drop packets associated with the particular type of data transfer is performed responsive to a determination that the first portion of packets comprises data corresponding to a particular transport layer security (TLS) version value.

20. (New) The one or more non-transitory computer-readable media of claim 15, wherein applying the first operator configured to drop packets associated with the particular type of data transfer is performed responsive to a determination that the first portion of packets comprises data corresponding to a particular real-time transport protocol value.

REMARKS

By the instant preliminary amendment, claim 1 is amended, and claims 2-20 are added. No new matter has been added. Upon entry of this paper, claims 1-20 will be pending in the instant application. Applicant respectfully requests consideration of the application and allowance of all pending claims.

In view of the above amendment, applicant believes the pending application is in condition for allowance.

Dated: January 15, 2018

Respectfully submitted,

By: /Joshua D. Schneider/
Joshua D. Schneider
Registration No.: 67,194

BANNER & WITCOFF, LTD.
1100 13th St. NW
STE 1200
Washington, DC 20005
(202) 824-3000

PTO/SB/08a (03-15)
Approved for use through 07/31/2016. OMB 0651-0031
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE

Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it contains a valid OMB control number.

INFORMATION DISCLOSURE STATEMENT BY APPLICANT
(Not for submission under 37 CFR 1.99)

Application Number	15614956
Filing Date	2017-06-06
First Named Inventor	Sean Moore
Art Unit	
Examiner Name	
Attorney Docket Number	007742.00091\US

U.S.PATENTS Remove

Examiner Initial*	Cite No	Patent Number	Kind Code[1]	Issue Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1	7913303	B1	2011-03-22	Rouland et al.	
	2	8832832	B1	2014-09-09	Visbal	
	3	6279113	B1	2001-08-21	Vaidya	
	4	8009566	B2	2011-08-30	Zuk et al.	
	5	6826694		2004-11-30	Dutta et al.	
	6	7143438	B1	2006-11-28	Coss et al.	
	7	7296288	B1	2007-11-13	Hill et al.	

If you wish to add additional U.S. Patent citation information please click the Add button. Add

U.S.PATENT APPLICATION PUBLICATIONS Remove

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15614956
	Filing Date	2017-06-06
	First Named Inventor	Sean Moore
	Art Unit	
	Examiner Name	
	Attorney Docket Number	007742.00091\US

Examiner Initial*	Cite No	Publication Number	Kind Code[1]	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1	20150372977	A1	2015-12-24	Yin	
	2	20160119365	A1	2016-04-28	Barel	
	3	20160294870	A1	2016-10-06	Banerjee et al.	
	4	20140150051	A1	2014-05-29	Bharali et al	
	5	20090028160	A1	2009-01-29	Eswaran et al	
	6	20160205069	A1	2016-07-14	Blocher et al.	
	7	20110277034	A1	2011-11-10	Hanson	
	8	20130104236	A1	2013-04-25	Ray et al.	
	9	20140259170	A1	2014-09-11	Amsler	
	10	20150033336	A1	2015-01-29	Wang	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15614956
	Filing Date	2017-06-06
	First Named Inventor	Sean Moore
	Art Unit	
	Examiner Name	
	Attorney Docket Number	007742.00091\US

	11	20150106930	A1	2015-04-16	Honda et al.	
	12	20150244734	A1	2015-08-27	Olson et al	
	13	20150341389	A1	2015-11-26	Kurakami	
	14	20060133377	A1	2006-06-22	Jain	
	15	20140317397	A1	2014-10-23	Martini	
	16	20020112188	A1	2002-08-15	Syvanne	
	17	20030018591	A1	2003-01-23	Komisky	
	18	20030051026	A1	2003-03-13	Carter et al.	
	19	20030220940	A1	2003-11-27	Futoransky et al.	
	20	20040172529	A1	2004-09-02	Culbert	
	21	20040172557	A1	2004-09-02	Nakae et al.	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15614956
	Filing Date	2017-06-06
	First Named Inventor	Sean Moore
	Art Unit	
	Examiner Name	
	Attorney Docket Number	007742.00091\US

	22	20050071650	A1	2005-03-31	Jo et al.	
	23	20060080733	A1	2006-04-13	Khosmood et al.	
	24	20060085849	A1	2006-04-20	Culbert	

If you wish to add additional U.S. Published Application citation information please click the Add button. Add

FOREIGN PATENT DOCUMENTS Remove

Examiner Initial*	Cite No	Foreign Document Number[3]	Country Code[2] i	Kind Code[4]	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear	T[5]
	1							

If you wish to add additional Foreign Patent Document citation information please click the Add button Add

NON-PATENT LITERATURE DOCUMENTS Remove

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, pages(s), volume-issue number(s), publisher, city and/or country where published.	T[5]
	1	2017 Jun 07 - (US) Office Action - App 14/745,207	
	2	2015 Sept 4 (US) Notice of Allowance - App. 14/702,755 (007742.00036)	
	3	2017 Jun 07 - (WO) International Search Report and Written Opinion - App PCT/US2016/067111	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15614956
	Filing Date	2017-06-06
	First Named Inventor	Sean Moore
	Art Unit	
	Examiner Name	
	Attorney Docket Number	007742.00091\US

	4	2017 Aug 15 (WO) International Preliminary Report on Patentability - App. PCT/US2015/062691 (007742.00064)	
	5	2017 Aug 21 (AU) First Examination Report - App. 2015248067 (007742.00079)	
	6	2017 Sept 29 (CA) Examination Report - App. 2,772,630	
	7	2017 Oct 17 (WO) International Preliminary Report on Patentability - App. PCT/US2016/026339 (007742.00075)	
	8	2017 Nov 3 (US) Non-Final Office Action - App. 15/413,834 (007742.00085)	
	9	2017 Nov 21 (US) Notice of Allowance - App. 14/690,302 (007742.00042)	

If you wish to add additional non-patent literature document citation information please click the Add button Add

EXAMINER SIGNATURE

Examiner Signature		Date Considered	

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

[1] See Kind Codes of USPTO Patent Documents at www.USPTO.GOV or MPEP 901.04. [2] Enter office that issued the document, by the two-letter code (WIPO Standard ST.3). [3] For Japanese patent documents, the indication of the year of the reign of the Emperor must precede the serial number of the patent document. [4] Kind of document by the appropriate symbols as indicated on the document under WIPO Standard ST.16 if possible. [5] Applicant is to place a check mark here if English language translation is attached.

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15614956
	Filing Date	2017-06-06
	First Named Inventor	Sean Moore
	Art Unit	
	Examiner Name	
	Attorney Docket Number	007742.00091\US

CERTIFICATION STATEMENT

Please see 37 CFR 1.97 and 1.98 to make the appropriate selection(s):

That each item of information contained in the information disclosure statement was first cited in any communication from a foreign patent office in a counterpart foreign application not more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(1).

OR

☐ That no item of information contained in the information disclosure statement 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 information disclosure statement was known to any individual designated in 37 CFR 1.56(c) more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(2).

See attached certification statement.

The fee set forth in 37 CFR 1.17 (p) has been submitted herewith.

☒ A certification statement is not submitted herewith.

SIGNATURE

A signature of the applicant or representative is required in accordance with CFR 1.33, 10.18. Please see CFR 1.4(d) for the form of the signature.

Signature	/Joshua D. Schneider/	Date (YYYY-MM-DD)	2018-01-15
Name/Print	Joshua D. Schneider	Registration Number	67,194

This collection of information is required by 37 CFR 1.97 and 1.98. 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 1 hour 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, VA 22313-1450. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. **SEND TO: Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450.**

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 the Freedom of Information Act requires disclosure of these record s.

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 inspections 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.

PATENT COOPERATION TREATY

From the INTERNATIONAL SEARCHING AUTHORITY

To:
Wright, Bradley
BANNER & WITCOFF, LTD.
1100 13th Street, N.W. Suite 1200
Washington, DC 20005-4051
ETATS-UNIS D'AMERIQUE

PCT

NOTIFICATION OF TRANSMITTAL OF THE INTERNATIONAL SEARCH REPORT AND THE WRITTEN OPINION OF THE INTERNATIONAL SEARCHING AUTHORITY, OR THE DECLARATION

(PCT Rule 44.1)

Applicant's or agent's file reference 007742 00081	Date of mailing *(day/month/year)* 7 June 2017 (07-06-2017)
International application No. PCT/US2016/067111	**FOR FURTHER ACTION** See paragraphs 1 and 4 below
Applicant CENTRIPETAL NETWORKS, INC	International filing date *(day/month/year)* 16 December 2016 (16-12-2016)

1. [X] The applicant is hereby notified that the international search report and the written opinion of the International Searching Authority have been established and are transmitted herewith.

   Filing of amendments and statement under Article 19:
   The applicant is entitled, if he so wishes, to amend the claims of the International Application (see Rule 46):

   When? The time limit for filing such amendments is normally two months from the date of transmittal of the International Search Report.

   How? Directly to the International Bureau of WIPO, 34 chemin des Colombettes 1211 Geneva 20, Switzerland, Fascimile No.: (41-22) 338.82.70

   For more detailed instructions, see *PCT Applicant's Guide,* International Phase, paragraphs 9.004 - 9.011.

2. [ ] The applicant is hereby notified that no international search report will be established and that the declaration under Article 17(2)(a) to that effect and the written opinion of the International Searching Authority are transmitted herewith.

3. [ ] **With regard to any protest** against payment of (an) additional fee(s) under Rule 40.2, the applicant is notified that:

   [ ] the protest together with the decision thereon has been transmitted to the International Bureau together with any applicant's request to forward the texts of both the protest and the decision thereon to the designated Offices.

   [ ] no decision has been made yet on the protest; the applicant will be notified as soon as a decision is made.

4. **Reminders**

   The applicant may **submit comments on an informal basis on the written opinion of the International Searching Authority** to the International Bureau. These comments will be made available to the public after international publication. The International Bureau will send a copy of such comments to all designated Offices unless an international preliminary examination report has been or is to be established.

   Shortly after the expiration of **18 months from the priority date, the international application will be published** by the International Bureau. If the applicant wishes to avoid or postpone publication, a notice of withdrawal of the international application, or of the priority claim, must reach the International Bureau before the completion of the technical preparations for international publication (Rules 90*bis*.1 and 90*bis*.3).

   Within **19 months** from the priority date, but only in respect of some designated Offices, a demand for international preliminary examination must be filed if the applicant wishes to postpone the entry into the national phase **until 30 months** from the priority date (in some Offices even later); otherwise, the applicant must, **within 20 months** from the priority date, perform the prescribed acts for **entry into the national phase** before those designated Offices. In respect of other designated Offices, the time limit of **30 months** (or later) will apply even if no demand is filed within 19 months. For details about the applicable time limits, Office by Office, see www.wipo.int/pct/en/texts/time_limits.html and the *PCT Applicant's Guide*, National Chapters.

   Within **19 months from the priority date, the applicant may request that a supplementary international search be carried out** by a different International Searching Authority that offers this service (Rule 45*bis*.1). The procedure for requesting supplementary international search is described in the *PCT Applicant's Guide*, International Phase, paragraphs 8.006-8.032.

Name and mailing address of the International Searching Authority European Patent Office, P.B. 5818 Patentlaan 2 NL-2280 HV Rijswijk Tel. (+31-70) 340-2040 Fax: (+31-70) 340-3016	Authorized officer KOESTEL, Gilbert Tel: +31 (0)70 340-3544

RECEIVED
JUN 09 2017
BANNER & WITCOFF LTD

Form PCT/ISA/220 (July 2014)

PATENT COOPERATION TREATY

PCT

INTERNATIONAL SEARCH REPORT

(PCT Article 18 and Rules 43 and 44)

Applicant's or agent's file reference	**FOR FURTHER ACTION**	see Form PCT/ISA/220 as well as, where applicable, item 5 below.
007742 00081		
International application No.	International filing date *(day/month/year)*	(Earliest) Priority Date *(day/month/year)*
PCT/US2016/067111	16 December 2016 (16-12-2016)	23 December 2015 (23-12-2015)

Applicant

CENTRIPETAL NETWORKS, INC

This international search report has been prepared by this International Searching Authority and is transmitted to the applicant according to Article 18. A copy is being transmitted to the International Bureau.

This international search report consists of a total of 3 sheets.

[X] It is also accompanied by a copy of each prior art document cited in this report.

1. **Basis of the report**

a. With regard to the **language,** the international search was carried out on the basis of:

[X] the international application in the language in which it was filed

[ ] a translation of the international application into _____, which is the language of a translation furnished for the purposes of international search (Rules 12.3(a) and 23.1(b))

b. [ ] This international search report has been established taking into account the **rectification of an obvious mistake** authorized by or notified to this Authority under Rule 91 (Rule 43.6*bis*(a)).

c. [ ] With regard to any **nucleotide and/or amino acid sequence** disclosed in the international application, see Box No. I.

2. [ ] **Certain claims were found unsearchable** (See Box No. II)

3. [ ] **Unity of invention is lacking** (see Box No III)

4. With regard to the **title,**

[X] the text is approved as submitted by the applicant

[ ] the text has been established by this Authority to read as follows:

5. With regard to the **abstract,**

[X] the text is approved as submitted by the applicant

[ ] the text has been established, according to Rule 38.2, by this Authority as it appears in Box No. IV. The applicant may, within one month from the date of mailing of this international search report, submit comments to this Authority

6. With regard to the **drawings,**

a. the figure of the **drawings** to be published with the abstract is Figure No. 13

[ ] as suggested by the applicant

[ ] as selected by this Authority, because the applicant failed to suggest a figure

[X] as selected by this Authority, because this figure better characterizes the invention

b. [ ] none of the figures is to be published with the abstract

RECEIVED
JUN 09 2017
BANNER & WITCOFF LTD

INTERNATIONAL SEARCH REPORT

International application No
PCT/US2016/067111

A. CLASSIFICATION OF SUBJECT MATTER
INV. H04L29/06
ADD.

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)
H04L

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)
EPO-Internal, WPI Data

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	US 2014/317397 A1 (MARTINI PAUL MICHAEL [US]) 23 October 2014 (2014-10-23) paragraph [0024] - paragraph [0027]	1-25
X	US 2014/283004 A1 (MOORE SEAN [US]) 18 September 2014 (2014-09-18) paragraph [0036] - paragraph [0044] claims 1-4	1-25

☐ 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

"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
30 May 2017	07/06/2017

Name and mailing address of the ISA/	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Fax: (+31-70) 340-3016	Lamelas Polo, Yvan

1

Form PCT/ISA/210 (second sheet) (April 2005)

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No
PCT/US2016/067111

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 2014317397 A1	23-10-2014	CA 2909799 A1	23-10-2014
		EP 2989769 A1	02-03-2016
		US 8739243 B1	27-05-2014
		US 2014317397 A1	23-10-2014
		WO 2014172384 A1	23-10-2014
US 2014283004 A1	18-09-2014	AU 2014249055 A1	03-09-2015
		CA 2902158 A1	09-10-2014
		EP 2974212 A1	20-01-2016
		US 2014283004 A1	18-09-2014
		US 2015237012 A1	20-08-2015
		US 2016072709 A1	10-03-2016
		WO 2014164713 A1	09-10-2014

Information on Search Strategy - Pilot phase (see OJ 2015, A86)
The type of information contained in this sheet may change during the pilot for improving the usefulness of this new service.

Application Number
PCT/US2016/067111

TITLE: RULE-BASED NETWORK-THREAT DETECTION FOR ENCRYPTED COMMUNICATIONS

APPLICANT: CENTRIPETAL NETWORKS, INC

IPC CLASSIFICATION: H04L29/06

EXAMINER: Lamelas Polo, Yvan

CONSULTED DATABASES: WPI, EPODOC, NPL

CLASSIFICATION SYMBOLS DEFINING EXTENT OF THE SEARCH:

IPC:

CPC: H04L63/0227, H04L63/1416, H04L63/1441

FI/F-TERMS:

KEYWORDS OR OTHER ELEMENTS FEATURING THE INVENTION:
PROBLEM: network-threat detection for encrypted communications.
SOLUTION: receiving data indicating network-threat indicators and configuring packet-filtering rules to cause the packet-filtering system to identify packets comprising unencrypted data, and packets comprising encrypted data. When portion of the unencrypted data corresponding to one or more of the network-threat indicators, and the packet-filtering rules being configured to cause the packet-filtering system to determine, based on the portion of the unencrypted data, that the packets comprising encrypted data correspond to the one or more network-threat indicators.

EPO FORM P04442

PATENT COOPERATION TREATY

From the
INTERNATIONAL SEARCHING AUTHORITY

To:

see form PCT/ISA/220

PCT

WRITTEN OPINION OF THE INTERNATIONAL SEARCHING AUTHORITY
(PCT Rule 43*bis*.1)

Date of mailing
(day/month/year) see form PCT/ISA/210 (second sheet)

Applicant's or agent's file reference
see form PCT/ISA/220

FOR FURTHER ACTION
See paragraph 2 below

International application No.	International filing date *(day/month/year)*	Priority date *(day/month/year)*
PCT/US2016/067111	16.12.2016	23.12.2015

International Patent Classification (IPC) or both national classification and IPC
INV. H04L29/06

Applicant
CENTRIPETAL NETWORKS, INC

1. This opinion contains indications relating to the following items:

- ☒ Box No. I Basis of the opinion
- ☐ Box No. II Priority
- ☐ Box No. III Non-establishment of opinion with regard to novelty, inventive step and industrial applicability
- ☐ Box No. IV Lack of unity of invention
- ☒ Box No. V Reasoned statement under Rule 43*bis*.1(a)(i) with regard to novelty, inventive step and industrial applicability; citations and explanations supporting such statement
- ☐ Box No. VI Certain documents cited
- ☒ Box No. VII Certain defects in the international application
- ☒ Box No. VIII Certain observations on the international application

2. **FURTHER ACTION**

If a demand for international preliminary examination is made, this opinion will usually be considered to be a written opinion of the International Preliminary Examining Authority ("IPEA") except that this does not apply where the applicant chooses an Authority other than this one to be the IPEA and the chosen IPEA has notifed the International Bureau under Rule 66.1*bis*(b) that written opinions of this International Searching Authority will not be so considered.

If this opinion is, as provided above, considered to be a written opinion of the IPEA, the applicant is invited to submit to the IPEA a written reply together, where appropriate, with amendments, before the expiration of 3 months from the date of mailing of Form PCT/ISA/220 or before the expiration of 22 months from the priority date, whichever expires later.

For further options, see Form PCT/ISA/220.

RECEIVED
JUN 09 2017
BANNER & WITCOFF LTD

Name and mailing address of the ISA:	Date of completion of this opinion	Authorized Officer
European Patent Office P.B. 5818 Patentlaan 2 NL-2280 HV Rijswijk - Pays Bas Tel. +31 70 340 - 2040 Fax: +31 70 340 - 3016	see form PCT/ISA/210	Lamelas Polo, Yvan Telephone No. +31 70 340-0

Form PCT/ISA/237 (Cover Sheet) (January 2015)

Box No. I Basis of the opinion

1. With regard to the **language**, this opinion has been established on the basis of:

   ☒ the international application in the language in which it was filed.

   ☐ a translation of the international application into , which is the language of a translation furnished for the purposes of international search (Rules 12.3(a) and 23.1 (b)).

2. ☐ This opinion has been established taking into account the **rectification of an obvious mistake** authorized by or notified to this Authority under Rule 91 (Rule 43*bis*.1(a))

3. ☐ With regard to any **nucleotide and/or amino acid sequence** disclosed in the international application, this opinion has been established on the basis of a sequence listing:

   a. ☐ forming part of the international application as filed:

      ☐ in the form of an Annex C/ST.25 text file.

      ☐ on paper or in the form of an image file.

   b. ☐ furnished together with the international application under PCT Rule 13*ter*.1(a) for the purposes of international search only in the form of an Annex C/ST.25 text file.

   c. ☐ furnished subsequent to the international filing date for the purposes of international search only:

      ☐ in the form of an Annex C/ST.25 text file (Rule 13*ter*.1(a)).

      ☐ on paper or in the form of an image file (Rule 13*ter*.1(b) and Administrative Instructions, Section 713).

4. ☐ In addition, in the case that more than one version or copy of a sequence listing has been filed or furnished, the required statements that the information in the subsequent or additional copies is identical to that forming part of the application as filed or does not go beyond the application as filed, as appropriate, were furnished.

5. Additional comments:

Box No. V Reasoned statement under Rule 43*bis*.1(a)(i) with regard to novelty, inventive step or industrial applicability; citations and explanations supporting such statement

1. Statement

Novelty (N)	Yes:	Claims	
	No:	Claims	1-25
Inventive step (IS)	Yes:	Claims	
	No:	Claims	1-25
Industrial applicability (IA)	Yes:	Claims	1-25
	No:	Claims	

2. Citations and explanations

see separate sheet

Box No. VII Certain defects in the international application

The following defects in the form or contents of the international application have been noted:

see separate sheet

Box No. VIII Certain observations on the international application

The following observations on the clarity of the claims, description, and drawings or on the question whether the claims are fully supported by the description, are made:

see separate sheet

1 **Re item V**

Reasoned statement with regard to novelty, inventive step or industrial applicability; citations and explanations supporting such statement

Novelty and inventive step of the independent claims

1.1 **Independent claim 1**

1.1.1 The present application does not meet the criteria of Article 33(1) PCT, because the subject-matter of claim 1 is not new in the sense of Article 33(2) PCT.

1.1.2 Document D1 discloses (the references in parentheses referring to D1):

A method comprising:

receiving, by a packet-filtering system configured to filter packets in accordance with a plurality of packet-filtering rules, data indicating a plurality of network-threat indicators (fig.1(102,103) "Network Gateway" "Rules" cf. [0018] "When data on a network is encrypted (e.g., by SSL or TSL) the data can pose challenges to network security appliances and routines in that it makes it difficult to inspect the data as it is transferred between the end users and servers on the Internet. This can lead to problems such as viruses being transferred over secure connections and entering the network instead of being blocked because the gateway security appliances could not inspect the data"); **and**

configuring, by the packet-filtering system, the plurality of packet-filtering rules to cause the packet-filtering system to:

identify packets comprising unencrypted data ([0024] "the network gateway may intercept and examine a DNS request 108 from the browser device 106 that is addressed to the DNS 110");

identify packets comprising encrypted data ([0024] "The network gateway 102 may be configured to make this determination so that encrypted communication from the browser device 106"); **and**

determine, based on a portion of the unencrypted data corresponding to one or more network-threat indicators of the plurality of network-threat indicators, that the packets comprising encrypted data correspond to the one or more network-threat indicators ([0024] "Based on, for example, the URL or URI in the DNS request 108 and rules 103 indicating which destination should be decrypted and which should be passed directly to the Internet destination,

the network gateway 102 may determine" "The network gateway 102 may make the determining to pass encrypted communication through the MitM gateway 104 based on security policies or concerns").

1.1.3 The subject-matter of claim 1 is consequently not new (Article 33(2) PCT).

1.1.4 For the sake of completeness, it is pointed out that the objection of lack of novelty set out above could have also been substantiated, in view of the disclosure of prior art document D2 (see references applying to this document cited in the search report). In particular,

- D2, [0036]-[0044], claims 1-4.

1.2 **Independent claims 24,25**

A corresponding reasoning as above applies, mutatis mutandis, to the subject-matter of independent claims 24,25 which therefore is also considered not new (Article 33(2) PCT).

Novelty and inventive step of the dependent claims

Dependent claims 2-23 do not appear to contain any additional features which, in combination with those of the claims to which they refer, meet the requirements of PCT with respect to novelty and/or inventive step, because said claims are considered implementation related design options that are either known or directly derivable from the prior art documents D1,D2 (see D1-D2 and references applying to these documents cited in the search report) or a matter of normal design procedure for a man skilled in the art.

2 **Re item VII**

Certain defects in the international application

2.1 The features of the claims are not provided with reference signs placed in parentheses (Rule 6.2(b) PCT).

2.2 Documents D1,D2 which appear to represent the most relevant prior art, should be acknowledged in the description.

3 **Re item VIII**

Certain observations on the international application

3.1 References in patent applications to other documents may relate either to the background art or to part of the disclosure of the invention. The patent specification should, regarding the essential features of the invention, be self-contained, i.e. capable of being understood without reference to any other document. Consequently, it appears that the expression "which is herein incorporated by reference" in [0001],[0024],[0041],[0053] should be deleted.

3.2 The vague and imprecise statement "Numerous other embodiments, modifications, and variations within the scope and spirit of the appended claims" in the description on page [108] implies that the subject-matter for which protection is sought may be different to that defined by the claims, thereby resulting in lack of clarity (Article 6 PCT) when used to interpret them.

Possible steps after receipt of the international search report (ISR) and written opinion of the International Searching Authority (WO/ISA)

General information

For all international applications, the competent International Searching Authority **(ISA)** will establish an international search report **(ISR)** accompanied by a written opinion of the International Searching Authority **(WO/ISA)**. The **WO/ISA** may be responded to by

- filing informal comments with the **International Bureau of WIPO (IB)** (where no demand for international preliminary examination **(demand)** is filed)
- filing amendments under Art. 19 PCT (this can be done whether or not a **demand** is filed)
- filing amendments under Art. 34 PCT and/or formal observations in response to objections raised in the **WO/ISA** (where a **demand** *is* actually filed)

This document explains these possibilities.

Filing informal comments

After receipt of the **ISR and WO/ISA**, the applicant may file informal comments on the **WO/ISA, directly with the IB** (see International Search and Preliminary Examination Guidelines 2.15). These will be communicated to the designated/elected Offices, together with the International Preliminary Report on Patentability **(IPRP)** at 30 months from the priority date.

Amending claims under Art. 19 PCT

The applicant may file **amended claims** under Art. 19 PCT, **directly with the IB** by the later of the following dates:

- 2 months from the date of mailing of the **ISR** and the **WO/ISA**
- 16 months from the priority date

However, any such amendment received by the **IB** after the expiration of the applicable time limit shall be **considered to have been received on time** by the **IB**, if it reaches it **before** the technical preparations for international publication have been completed (the 15th day prior to the date of publication, see PCT Applicant's Guide, International Phase, 9.013).

For further information, please see Rule 46 PCT as well as form PCT/ISA/220.

Please also note that, when filing amended claims under Art. 19 PCT, such amendments shall be **accompanied by a letter** identifying the amendments made and also the basis for the amendments in the application as originally filed (Rule 46.5(b) PCT). Where a **demand** is filed, failure to comply with this requirement may result in the amendments being ignored in the International Preliminary Examination Report **(IPER)**, see Rule 70.2(c-*bis*) PCT.

BNSDOCID: <XS_____200704010CK_I_>

Filing a demand for international preliminary examination

In principle, the **WO/ISA** will be considered to be the written opinion of the International Preliminary Examining Authority (**IPEA**). Where the **WO/ISA** issued by the **EPO** as **ISA** gives a positive opinion on the international application and the invention to which it relates, filing a **demand** with the **EPO** as **IPEA** would normally be unnecessary, since a positive IPRP would anyway be established by the IB based on the WO/ISA (see also further below).

If the applicant wishes to file a **demand** (for example, to allow him to argue his case in international preliminary examination with regard to objections raised in a negative **WO/ISA** before the **IPEA** issues an **IPER**), this must be done before expiration of **3 months after the date of mailing of the ISR and WO/ISA** or **22 months after priority date,** whichever expires later (Rule 54*bis* PCT). Amendments under Art. 34 PCT can be filed with the IPEA, normally at the same time as filing the demand (Rule 66.1(b) PCT) or within the time limit set for reply to any written opinion issued during international preliminary examination by the **IPEA**.

If a **demand** is filed at the **EPO** as **IPEA** and no comments/amendments have been received by the time the **EPO** starts drawing up the **IPER** (Rule 66.4*bis* PCT), the **WO/ISA** will be transformed by the **IPEA** into an **IPER** (also called the **IPRP (Chapter II)** which would merely reflect the content of the **WO/ISA** (OJ 10/2011, 532). The **demand** can still be withdrawn (Art. 37 PCT).

Please also note that, when filing amendments under Art. 34 PCT, such amendments shall be accompanied by a letter which identifies the amendments made and also the basis for the amendments in the application as originally filed (Rule 66.8(a) PCT). Failure to comply with this requirement may result in the amendments being ignored in the **IPER (IPRP (Chapter II))**, see Rule 70.2(c-*bis*) PCT.

Filing a request for supplementary international search

The applicant may, with the **IB**, file a request for **supplementary international search** under Rule 45*bis*.1 PCT. The present **ISR** and **WO/ISA** may also be taken into account in the execution of that supplementary international search, provided that these are available to the Authority charged with this task before it starts the supplementary search (Rule 45*bis*.5 PCT).

This kind of request **cannot be filed specifying the ISA** who did the **international search**.

More information on this topic can be found in the **PCT Applicant's Guide**, Chapter 8 (http://www.wipo.int/pct/en/guide/ip08.html).

End of the international phase

Where no **demand** is filed, at the end of the international phase, the **IB** will transform the **WO/ISA** into the IPRP (PCT Chapter I) (Rule 44*bis* PCT), which will then be transmitted together with possible informal comments to the designated Offices. Where a demand is filed, the **WO/ISA** is not transformed into an **IPRP (Chapter I)** by the **IB**, but rather the **IPEA** will establish an **IPER**, (the IPER is the same as the IPRP (PCT Chapter II), see Rule 70.15 PCT).

NSDOCID: <XS_____200704010CK_I_>

PATENT COOPERATION TREATY

From the INTERNATIONAL BUREAU

PCT

NOTIFICATION CONCERNING TRANSMITTAL OF COPY OF INTERNATIONAL PRELIMINARY REPORT ON PATENTABILITY (CHAPTER I OF THE PATENT COOPERATION TREATY)
(PCT Rule 44bis.1(c))

To:

WRIGHT, Bradley
Suite 1200
1100 13th Street, NW
Washington, District of Columbia 20005
ETATS-UNIS D'AMERIQUE

Date of mailing *(day/month/year)*
24 August 2017 (24.08.2017)

Applicant's or agent's file reference
007742.00064

IMPORTANT NOTICE

International application No.	International filing date *(day/month/year)*	Priority date *(day/month/year)*
PCT/US2015/062691	25 November 2015 (25.11.2015)	10 February 2015 (10.02.2015)

Applicant
CENTRIPETAL NETWORKS, INC.

The International Bureau transmits herewith a copy of the international preliminary report on patentability (Chapter I of the Patent Cooperation Treaty).

The International Bureau of WIPO
34, chemin des Colombettes
1211 Geneva 20, Switzerland

Facsimile No. +41 22 338 82 70

Authorized officer
Athina Nickitas-Etienne

e-mail: pct.team4@wipo.int

Form PCT/IB/326 (January 2004)

PATENT COOPERATION TREATY

PCT

INTERNATIONAL PRELIMINARY REPORT ON PATENTABILITY
(Chapter I of the Patent Cooperation Treaty)

(PCT Rule 44*bis*)

Applicant's or agent's file reference 007742.00064	**FOR FURTHER ACTION**	See item 4 below
International application No. PCT/US2015/062691	International filing date *(day/month/year)* 25 November 2015 (25.11.2015)	Priority date *(day/month/year)* 10 February 2015 (10.02.2015)
International Patent Classification (8th edition unless older edition indicated) See relevant information in Form PCT/ISA/237		
Applicant CENTRIPETAL NETWORKS, INC.		

1. This international preliminary report on patentability (Chapter I) is issued by the International Bureau on behalf of the International Searching Authority under Rule 44 *bis*.1(a).

2. This REPORT consists of a total of 6 sheets, including this cover sheet.

   In the attached sheets, any reference to the written opinion of the International Searching Authority should be read as a reference to the international preliminary report on patentability (Chapter I) instead.

3. This report contains indications relating to the following items:

☒	Box No. I	Basis of the report
☐	Box No. II	Priority
☐	Box No. III	Non-establishment of opinion with regard to novelty, inventive step and industrial applicability
☐	Box No. IV	Lack of unity of invention
☒	Box No. V	Reasoned statement under Article 35(2) with regard to novelty, inventive step or industrial applicability; citations and explanations supporting such statement
☐	Box No. VI	Certain documents cited
☒	Box No. VII	Certain defects in the international application
☒	Box No. VIII	Certain observations on the international application

4. The International Bureau will communicate this report to designated Offices in accordance with Rules 44*bis*.3(c) and 93*bis*.1 but not, except where the applicant makes an express request under Article 23(2), before the expiration of 30 months from the priority date (Rule 44*bis* .2).

	Date of issuance of this report 15 August 2017 (15.08.2017)
The International Bureau of WIPO 34, chemin des Colombettes 1211 Geneva 20, Switzerland Facsimile No. +41 22 338 82 70	Authorized officer Athina Nickitas-Etienne e-mail: pct.team4@wipo.int

Form PCT/IB/373 (January 2004)

PATENT COOPERATION TREATY

From the
INTERNATIONAL SEARCHING AUTHORITY

To:

see form PCT/ISA/220

PCT

WRITTEN OPINION OF THE INTERNATIONAL SEARCHING AUTHORITY

(PCT Rule 43*bis*.1)

Date of mailing
(day/month/year) see form PCT/ISA/210 (second sheet)

Applicant's or agent's file reference
see form PCT/ISA/220

FOR FURTHER ACTION
See paragraph 2 below

International application No.	International filing date *(day/month/year)*	Priority date *(day/month/year)*
PCT/US2015/062691	25.11.2015	10.02.2015

International Patent Classification (IPC) or both national classification and IPC
INV. H04L12/701 H04L12/26

Applicant
CENTRIPETAL NETWORKS, INC.

1. This opinion contains indications relating to the following items:

- ☒ Box No. I Basis of the opinion
- ☐ Box No. II Priority
- ☐ Box No. III Non-establishment of opinion with regard to novelty, inventive step and industrial applicability
- ☐ Box No. IV Lack of unity of invention
- ☒ Box No. V Reasoned statement under Rule 43*bis*.1(a)(i) with regard to novelty, inventive step and industrial applicability; citations and explanations supporting such statement
- ☐ Box No. VI Certain documents cited
- ☒ Box No. VII Certain defects in the international application
- ☒ Box No. VIII Certain observations on the international application

2. **FURTHER ACTION**

If a demand for international preliminary examination is made, this opinion will usually be considered to be a written opinion of the International Preliminary Examining Authority ("IPEA") except that this does not apply where the applicant chooses an Authority other than this one to be the IPEA and the chosen IPEA has notifed the International Bureau under Rule 66.1*bis*(b) that written opinions of this International Searching Authority will not be so considered.

If this opinion is, as provided above, considered to be a written opinion of the IPEA, the applicant is invited to submit to the IPEA a written reply together, where appropriate, with amendments, before the expiration of 3 months from the date of mailing of Form PCT/ISA/220 or before the expiration of 22 months from the priority date, whichever expires later.

For further options, see Form PCT/ISA/220.

Name and mailing address of the ISA:	Date of completion of this opinion	Authorized Officer
European Patent Office D-80298 Munich Tel. +49 89 2399 - 0 Fax: +49 89 2399 - 4465	see form PCT/ISA/210	Pais Gonçalves, A Telephone No. +49 89 2399-0

Form PCT/ISA/237 (Cover Sheet) (January 2015)

Box No. I Basis of the opinion

1. With regard to the **language**, this opinion has been established on the basis of:

   ☒ the international application in the language in which it was filed.

   ☐ a translation of the international application into , which is the language of a translation furnished for the purposes of international search (Rules 12.3(a) and 23.1 (b)).

2. ☐ This opinion has been established taking into account the **rectification of an obvious mistake** authorized by or notified to this Authority under Rule 91 (Rule 43*bis*.1(a))

3. ☐ With regard to any **nucleotide and/or amino acid sequence** disclosed in the international application, this opinion has been established on the basis of a sequence listing:

   a. ☐ forming part of the international application as filed:

      ☐ in the form of an Annex C/ST.25 text file.

      ☐ on paper or in the form of an image file.

   b. ☐ furnished together with the international application under PCT Rule 13*ter*.1(a) for the purposes of international search only in the form of an Annex C/ST.25 text file.

   c. ☐ furnished subsequent to the international filing date for the purposes of international search only:

      ☐ in the form of an Annex C/ST.25 text file (Rule 13*ter*.1(a)).

      ☐ on paper or in the form of an image file (Rule 13*ter*.1(b) and Administrative Instructions, Section 713).

4. ☐ In addition, in the case that more than one version or copy of a sequence listing has been filed or furnished, the required statements that the information in the subsequent or additional copies is identical to that forming part of the application as filed or does not go beyond the application as filed, as appropriate, were furnished.

5. Additional comments:

Box No. V Reasoned statement under Rule 43*bis*.1(a)(i) with regard to novelty, inventive step or industrial applicability; citations and explanations supporting such statement

1. Statement

Novelty (N)	Yes:	Claims	1-25
	No:	Claims	
Inventive step (IS)	Yes:	Claims	
	No:	Claims	1-25
Industrial applicability (IA)	Yes:	Claims	1-25
	No:	Claims	

2. Citations and explanations

see separate sheet

Box No. VII Certain defects in the international application

The following defects in the form or contents of the international application have been noted:

see separate sheet

Box No. VIII Certain observations on the international application

The following observations on the clarity of the claims, description, and drawings or on the question whether the claims are fully supported by the description, are made:

see separate sheet

Reference is made to the following documents:

D1 US 2014/281030 A1 (CUI LIANG [CN] ET AL) 18 September 2014 (2014-09-18)

D2 US 2004/073655 A1 (KAN CHAO [US] ET AL) 15 April 2004 (2004-04-15)

section VIII

1 Claims 1, 24 and 25 do not fulfil the requirements of Article 6 PCT for the following reasons:

1.1 In the claims it is not clear what is meant by the expression "data identifying the host", how such data is determined or the purpose of generating and transmitting said data.

1.2 According to the description, it seems that this data is used to identify which packets are associated with a particular flow or session. However, this is not clear from the wording of the claims.

1.3 In addition, the desired scope of protection of independent claim 25 is not clearly defined due to the wording "one or more".

section V

2 Assuming the interpretation indicated in point 1.2 above, it seems that claim 1 would be directed to a method of monitoring packets received by a network device and packets transmitted by the device and correlating them in order to determined that the packets are associated with a flow.

2.1 Such a method is already known from the disclosure of D1, see e.g. [0002] to [0004] or D2, see e.g. [0037].

2.2 The only apparent distinguishing feature between claim 1 and e.g. D1 could be that the network device and the host are located in different networks.

2.3 However, such distinguishing feature is a simple modification of the basic concept disclosed in D1 lying within the general design competence of a person skilled in the art who would consider such measure to be at least implicitly disclosed in D1, see e.g. figures 7 and 10A.

2.4 In addition, these arguments are also valid for independent Claim 24, which contains the same combination of features as claim 1 in terms of an apparatus claim, i.e. the apparatus (system) for carrying out the method of claim 1.

2.5 Said arguments are also valid for independent claim 25 which is directed to computer-readable media comprising instructions essentially to carry out the method of claim 1.

2.6 Therefore, the subject-matter of Claims 1, 24 and 25 does not involve an **inventive step**, Article 33 (1), (3) PCT.

2.7 The additional features of the dependent claims seem to relate to minor design details and/or implementation measures, which are known or directly derivable from the above-referred documents or constitute normal practice of a person skilled in the art. These claims thus, either alone or in combination, do not seem to add anything new or of inventive significance to any of the previously-referred claims, see passages indicated in the search report.

section VII

3 Having regard to the above objections, it is not clear at present which part of the application could serve as a basis for a new and inventive subject-matter.

4 Should the Applicant nevertheless regard some particular matter of the application as having an inventive contribution over the prior art, then an independent claim relating to such subject-matter should be filed. The Applicant is also requested to indicate in the letter of reply the difference of said subject-matter vis-à-vis the state of the art and the significance thereof.

4.1 The above-cited documents D1 and D2 should be acknowledged and briefly discussed in the opening part of the description, Rule 5.1 (a) (ii) PCT, making clear any inventive contribution of the claimed invention over the prior art.

4.2 The independent claims should be drafted in a two-part form, having in their pre-characterizing part all features known from D1 (refer to point 2 above), Rule 6.3 (b) (ii) PCT.

4.3 The opening part of the description should be modified to bring it into agreement with the amended claim, Rule 5.1(a)(iii) PCT, if required.

4.4 The claims should include reference signs relating to the features referred to therein, Rule 6.2 (b) PCT.

PATENT COOPERATION TREATY

From the INTERNATIONAL BUREAU

PCT

NOTIFICATION CONCERNING TRANSMITTAL OF COPY OF INTERNATIONAL PRELIMINARY REPORT ON PATENTABILITY (CHAPTER I OF THE PATENT COOPERATION TREATY)
(PCT Rule 44bis.1(c))

To:

WRIGHT, Bradley
Suite 1200
1100 13th Street, NW
Washington, District of Columbia 20005
ETATS-UNIS D'AMERIQUE

Date of mailing *(day/month/year)*
26 October 2017 (26.10.2017)

Applicant's or agent's file reference
007742.00075

IMPORTANT NOTICE

International application No.	International filing date *(day/month/year)*	Priority date *(day/month/year)*
PCT/US2016/026339	07 April 2016 (07.04.2016)	17 April 2015 (17.04.2015)

Applicant
CENTRIPETAL NETWORKS, INC.

The International Bureau transmits herewith a copy of the international preliminary report on patentability (Chapter I of the Patent Cooperation Treaty).

The International Bureau of WIPO
34, chemin des Colombettes
1211 Geneva 20, Switzerland

Facsimile No. +41 22 338 82 70

Authorized officer
Agnès Wittmann-Regis

e-mail: pct.team6@wipo.int

Form PCT/IB/326 (January 2004)

PATENT COOPERATION TREATY

PCT

INTERNATIONAL PRELIMINARY REPORT ON PATENTABILITY
(Chapter I of the Patent Cooperation Treaty)

(PCT Rule 44*bis*)

Applicant's or agent's file reference 007742.00075	**FOR FURTHER ACTION**	See item 4 below
International application No. PCT/US2016/026339	International filing date *(day/month/year)* 07 April 2016 (07.04.2016)	Priority date *(day/month/year)* 17 April 2015 (17.04.2015)
International Patent Classification (8th edition unless older edition indicated) See relevant information in Form PCT/ISA/237		
Applicant CENTRIPETAL NETWORKS, INC.		

1. This international preliminary report on patentability (Chapter I) is issued by the International Bureau on behalf of the International Searching Authority under Rule 44 *bis*.1(a).

2. This REPORT consists of a total of 6 sheets, including this cover sheet.

   In the attached sheets, any reference to the written opinion of the International Searching Authority should be read as a reference to the international preliminary report on patentability (Chapter I) instead.

3. This report contains indications relating to the following items:

☒	Box No. I	Basis of the report
☐	Box No. II	Priority
☐	Box No. III	Non-establishment of opinion with regard to novelty, inventive step and industrial applicability
☐	Box No. IV	Lack of unity of invention
☒	Box No. V	Reasoned statement under Article 35(2) with regard to novelty, inventive step or industrial applicability; citations and explanations supporting such statement
☐	Box No. VI	Certain documents cited
☐	Box No. VII	Certain defects in the international application
☐	Box No. VIII	Certain observations on the international application

4. The International Bureau will communicate this report to designated Offices in accordance with Rules 44*bis*.3(c) and 93*bis*.1 but not, except where the applicant makes an express request under Article 23(2), before the expiration of 30 months from the priority date (Rule 44*bis* .2).

	Date of issuance of this report 17 October 2017 (17.10.2017)
The International Bureau of WIPO 34, chemin des Colombettes 1211 Geneva 20, Switzerland Facsimile No. +41 22 338 82 70	Authorized officer Agnès Wittmann-Regis e-mail: pct.team6@wipo.int

Form PCT/IB/373 (January 2004)

PATENT COOPERATION TREATY

From the
INTERNATIONAL SEARCHING AUTHORITY

To:

see form PCT/ISA/220

PCT

WRITTEN OPINION OF THE INTERNATIONAL SEARCHING AUTHORITY

(PCT Rule 43*bis*.1)

Date of mailing
(day/month/year) see form PCT/ISA/210 (second sheet)

Applicant's or agent's file reference
see form PCT/ISA/220

FOR FURTHER ACTION
See paragraph 2 below

International application No.	International filing date *(day/month/year)*	Priority date *(day/month/year)*
PCT/US2016/026339	07.04.2016	17.04.2015

International Patent Classification (IPC) or both national classification and IPC
INV. H04L29/06 H04L12/26

Applicant
CENTRIPETAL NETWORKS, INC.

1. This opinion contains indications relating to the following items:

- ☒ Box No. I Basis of the opinion
- ☐ Box No. II Priority
- ☐ Box No. III Non-establishment of opinion with regard to novelty, inventive step and industrial applicability
- ☐ Box No. IV Lack of unity of invention
- ☒ Box No. V Reasoned statement under Rule 43*bis*.1(a)(i) with regard to novelty, inventive step and industrial applicability; citations and explanations supporting such statement
- ☐ Box No. VI Certain documents cited
- ☐ Box No. VII Certain defects in the international application
- ☐ Box No. VIII Certain observations on the international application

2. **FURTHER ACTION**

If a demand for international preliminary examination is made, this opinion will usually be considered to be a written opinion of the International Preliminary Examining Authority ("IPEA") except that this does not apply where the applicant chooses an Authority other than this one to be the IPEA and the chosen IPEA has notifed the International Bureau under Rule 66.1*bis*(b) that written opinions of this International Searching Authority will not be so considered.

If this opinion is, as provided above, considered to be a written opinion of the IPEA, the applicant is invited to submit to the IPEA a written reply together, where appropriate, with amendments, before the expiration of 3 months from the date of mailing of Form PCT/ISA/220 or before the expiration of 22 months from the priority date, whichever expires later.

For further options, see Form PCT/ISA/220.

Name and mailing address of the ISA:	Date of completion of this opinion	Authorized Officer
European Patent Office D-80298 Munich Tel. +49 89 2399 - 0 Fax: +49 89 2399 - 4465	see form PCT/ISA/210	San Millán Maeso, J Telephone No. +49 89 2399-0

Form PCT/ISA/237 (Cover Sheet) (January 2015)

Box No. I Basis of the opinion

1. With regard to the **language**, this opinion has been established on the basis of:

   ☒ the international application in the language in which it was filed.

   ☐ a translation of the international application into , which is the language of a translation furnished for the purposes of international search (Rules 12.3(a) and 23.1 (b)).

2. ☐ This opinion has been established taking into account the **rectification of an obvious mistake** authorized by or notified to this Authority under Rule 91 (Rule 43*bis*.1(a))

3. ☐ With regard to any **nucleotide and/or amino acid sequence** disclosed in the international application, this opinion has been established on the basis of a sequence listing:

   a. ☐ forming part of the international application as filed:

      ☐ in the form of an Annex C/ST.25 text file.

      ☐ on paper or in the form of an image file.

   b. ☐ furnished together with the international application under PCT Rule 13*ter*.1(a) for the purposes of international search only in the form of an Annex C/ST.25 text file.

   c. ☐ furnished subsequent to the international filing date for the purposes of international search only:

      ☐ in the form of an Annex C/ST.25 text file (Rule 13*ter*.1(a)).

      ☐ on paper or in the form of an image file (Rule 13*ter*.1(b) and Administrative Instructions, Section 713).

4. ☐ In addition, in the case that more than one version or copy of a sequence listing has been filed or furnished, the required statements that the information in the subsequent or additional copies is identical to that forming part of the application as filed or does not go beyond the application as filed, as appropriate, were furnished.

5. Additional comments:

Box No. V Reasoned statement under Rule 43*bis*.1(a)(i) with regard to novelty, inventive step or industrial applicability; citations and explanations supporting such statement

1. Statement

Novelty (N)	Yes:	Claims	2-30, 33-43
	No:	Claims	1, 31, 32
Inventive step (IS)	Yes:	Claims	
	No:	Claims	1-43
Industrial applicability (IA)	Yes:	Claims	1-43
	No:	Claims	

2. Citations and explanations

see separate sheet

Re Item V

Reasoned statement with regard to novelty, inventive step or industrial applicability; citations and explanations supporting such statement

1. Reference is made to the following documents:

D1 **US 2004/093513 A1 (CANTRELL CRAIG [US] ET AL) 13 May 2004 (2004-05-13)**

D2 **US 2012/023576 A1 (SORENSEN AMANDA [US] ET AL) 26 January 2012 (2012-01-26)**

D3 **US 2008/163333 A1 (KASRALIKAR RAHUL [US]) 3 July 2008 (2008-07-03)**

2. The present application does not meet the criteria of Article 33(2) PCT, because the subject-matter of claims 1, 31 and 32 is not new.

2.1. Independent claim 1

Document D1 discloses (abstract) a method comprising:
- (paragraph 14) receiving, by a packet-filtering device, a plurality of packet-filtering rules configured to cause the packet-filtering device to identify packets corresponding to at least one of a plurality of network-threat indicators;
- (paragraphs 13-14 and 26) receiving, by the packet-filtering device, a plurality of packets; and for each packet of the plurality of packets and responsive to a determination by the packet-filtering device that the packet corresponds to one or more criteria, specified by a packet-filtering rule of the plurality of packet-filtering rules, that correspond to one or more network-threat indicators of the plurality of network-threat indicators:
- (abstract and paragraph 39) applying, by the packet-filtering device and to the packet, an operator specified by the packet-filtering rule and configured to cause the packet-filtering device to either prevent the packet from continuing toward its destination or allow the packet to continue toward its destination;
- (paragraphs 12, 16 and 17) generating, by the packet-filtering device, a packet-log entry comprising information from the packet-filtering rule that identifies the one or more network-threat indicators and indicating whether the packet-filtering device prevented the packet from continuing toward its destination or allowed the packet to continue toward its destination;

- (paragraphs 12, 16 and 17) generating, by the packet-filtering device and based on the packet-log entry, data indicating whether the packet-filtering device prevented the packet from continuing toward its destination or allowed the packet to continue toward its destination;
- (paragraphs 41-44) communicating, by the packet-filtering device and to a user device, the data; and indicating, based on the data, in an interface displayed by the user device, and in a portion of the interface corresponding to the packet-filtering rule and the one or more network-threat indicators, whether the packet-filtering device prevented the packet from continuing toward its destination or allowed the packet to continue toward its destination.

The subject-matter of claim 1 is therefore not new in the sense of Article 33(2) PCT.

Equivalent reasoning applies taking into account the disclosure of documents D2 or D3 (see the references to these documents in the international search report).

2.2. Independent claims 31 and 32

Equivalent reasoning applies, mutatis mutandis, to the subject-matter of the corresponding independent claims 31 and 32, which therefore are considered not new.

3. Dependent claims 2-30 and 33-43

The dependent claims do not contain any features which, in combination with the features of any claim to which they refer, meet the requirements of the PCT in respect of novelty and/or inventive step; since they are disclosed by documents D1, D2 (see the references applying to these documents cited in the search report) or they correspond to well-known design options which the skilled person would choose without involving an inventive step (Articles 33(2) and 33(3) PCT).

Electronic Patent Application Fee Transmittal

Application Number:	15614956
Filing Date:	06-Jun-2017
Title of Invention:	FILTERING NETWORK DATA TRANSFERS
First Named Inventor/Applicant Name:	Sean Moore
Filer:	Joshua Daniel Schneider/Jennifer Brady
Attorney Docket Number:	007742.00091\US

Filed as Small Entity

Filing Fees for Utility under 35 USC 111(a)

Description	Fee Code	Quantity	Amount	Sub-Total in USD($)
Basic Filing:				
UTILITY FILING FEE (ELECTRONIC FILING)	4011	1	70	70
UTILITY SEARCH FEE	2111	1	300	300
UTILITY EXAMINATION FEE	2311	1	360	360
Pages:				
Claims:				
Miscellaneous-Filing:				
LATE FILING FEE FOR OATH OR DECLARATION	2051	1	70	70
Petition:				

Description	Fee Code	Quantity	Amount	Sub-Total in USD($)
Patent-Appeals-and-Interference:				
Post-Allowance-and-Post-Issuance:				
Extension-of-Time:				
Extension - 5 months with $0 paid	2255	1	1500	1500
Miscellaneous:				
		Total in USD ($)		**2300**

Electronic Acknowledgement Receipt	
EFS ID:	31491275
Application Number:	15614956
International Application Number:	
Confirmation Number:	9334
Title of Invention:	FILTERING NETWORK DATA TRANSFERS
First Named Inventor/Applicant Name:	Sean Moore
Customer Number:	22907
Filer:	Joshua Daniel Schneider
Filer Authorized By:	
Attorney Docket Number:	007742.00091\US
Receipt Date:	15-JAN-2018
Filing Date:	06-JUN-2017
Time Stamp:	13:23:58
Application Type:	Utility under 35 USC 111(a)

Payment information:

Submitted with Payment	yes
Payment Type	DA
Payment was successfully received in RAM	$2300
RAM confirmation Number	011618INTEFSW00014193190733
Deposit Account	
Authorized User	
The Director of the USPTO is hereby authorized to charge indicated fees and credit any overpayment as follows:	

File Listing:

Document Number	Document Description	File Name	File Size(Bytes)/ Message Digest	Multi Part /.zip	Pages (if appl.)
1		Preliminary_Amendment_Under_37_CFR1_18930974.pdf	42076 e7b5c1a67b458bf19444d79270e8fdc59554ab65	yes	9
	Multipart Description/PDF files in .zip description				
	Document Description		**Start**	**End**	
	Preliminary Amendment		1	1	
	Claims		2	8	
	Applicant Arguments/Remarks Made in an Amendment		9	9	
Warnings:					
Information:					
2	Information Disclosure Statement (IDS) Form (SB08)	Information_Disclosure_Statement_Fillable_PDF_18930483.pdf	1036376 61a63ea6f8bcad56f06c6b054eabb248a9250d2d	no	7
Warnings:					
Information:					
3	Other Reference-Patent/App/Search documents	Final_Rejection__1449_18556085.pdf	1630810 db68866096f1242c5e0fd36f1f4cbee45d8dbf23	no	40
Warnings:					
Information:					
4	Other Reference-Patent/App/Search documents	Notice_of_Allowance_and_Fees_Due_PTOL-85__892_17481077.pdf	432120 9bbca6d1c47b31eb2c2f21430c2f8e3aa2f9e527	no	9
Warnings:					
Information:					
5	Non Patent Literature	Corr_from_WIPO__-_N__of_Transmittal_of_the_International_Search_Report_18615129.pdf	828763 978f48233eb8c59f605a36a1854546057d647889	no	13

Warnings:					
The page size in the PDF is too large. The pages should be 8.5 x 11 or A4. If this PDF is submitted, the pages will be resized upon entry into the Image File Wrapper and may affect subsequent processing					
Information:					
6	Non Patent Literature	IPRP_18691954.PDF	250545 b816ea799ff355591925013ab940a23195a6cfb1	no	7
Warnings:					
Information:					
7	Non Patent Literature	Examination_Report_No_1873 1811.PDF	286495 797507d9ddf499ada8c7b6509f12e42f802f481d	no	6
Warnings:					
Information:					
8	Non Patent Literature	CIPO_OA_dtd_09-29-2017_187 79397.PDF	339527 2a741fb28ed6f87aa782fa16bad9c78c823e51f3	no	5
Warnings:					
Information:					
9	Non Patent Literature	2014_Oct_17_International_Pr eliminary_Report_on_Patentab ility_18842240.PDF	240945 9375f98158b4e6a299a5c28d64957a2a1d478195	no	7
Warnings:					
Information:					
10	Other Reference-Patent/App/Search documents	Non-Final_Rejection__892__1449_1 8810768.pdf	2357334 6efcb7d31460f3799ae446f59c80655d33da5973	no	48
Warnings:					
Information:					
11	Other Reference-Patent/App/Search documents	Notice_of_Allowance_and_Fee s_Due_PTOL-85__INTVSUMEX_ _892__1449_18843642.pdf	5430704 dbe21ae40f9aff20daa5d7278daeec9100b4e73a	no	98
Warnings:					
Information:					
12	Fee Worksheet (SB06)	fee-info.pdf	39025 69c4432117dac85ad0ec782e2d05ddbbbfbaf0db	no	2

Warnings:	
Information:	
Total Files Size (in bytes):	12914720

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 a 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 APPLICATION FEE DETERMINATION RECORD

Substitute for Form PTO-875

Application or Docket Number: 15/614,956

APPLICATION AS FILED - PART I

FOR	(Column 1) NUMBER FILED	(Column 2) NUMBER EXTRA	SMALL ENTITY RATE($)	SMALL ENTITY FEE($)	OR	OTHER THAN SMALL ENTITY RATE($)	OTHER THAN SMALL ENTITY FEE($)
BASIC FEE (37 CFR 1.16(a), (b), or (c))	N/A	N/A	N/A	70		N/A	
SEARCH FEE (37 CFR 1.16(k), (i), or (m))	N/A	N/A	N/A	300		N/A	
EXAMINATION FEE (37 CFR 1.16(o), (p), or (q))	N/A	N/A	N/A	360		N/A	
TOTAL CLAIMS (37 CFR 1.16(i))	1 minus 20 =	*	x 40 =	0.00	OR		
INDEPENDENT CLAIMS (37 CFR 1.16(h))	1 minus 3 =	*	x 210 =	0.00			
APPLICATION SIZE FEE (37 CFR 1.16(s))	If the specification and drawings exceed 100 sheets of paper, the application size fee due is $310 ($155 for small entity) for each additional 50 sheets or fraction thereof. See 35 U.S.C. 41(a)(1)(G) and 37 CFR 1.16(s).			0.00			
MULTIPLE DEPENDENT CLAIM PRESENT (37 CFR 1.16(j))				0.00			
* If the difference in column 1 is less than zero, enter "0" in column 2.			TOTAL	730		TOTAL	

APPLICATION AS AMENDED - PART II

AMENDMENT A

	(Column 1) CLAIMS REMAINING AFTER AMENDMENT		(Column 2) HIGHEST NUMBER PREVIOUSLY PAID FOR	(Column 3) PRESENT EXTRA	SMALL ENTITY RATE($)	SMALL ENTITY ADDITIONAL FEE($)	OR	OTHER THAN SMALL ENTITY RATE($)	OTHER THAN SMALL ENTITY ADDITIONAL FEE($)
Total (37 CFR 1.16(i))	*	Minus	**	=	x =		OR	x =	
Independent (37 CFR 1.16(h))	*	Minus	***	=	x =		OR	x =	
Application Size Fee (37 CFR 1.16(s))									
FIRST PRESENTATION OF MULTIPLE DEPENDENT CLAIM (37 CFR 1.16(j))							OR		
					TOTAL ADD'L FEE		OR	TOTAL ADD'L FEE	

AMENDMENT B

	(Column 1) CLAIMS REMAINING AFTER AMENDMENT		(Column 2) HIGHEST NUMBER PREVIOUSLY PAID FOR	(Column 3) PRESENT EXTRA	RATE($)	ADDITIONAL FEE($)		RATE($)	ADDITIONAL FEE($)
Total (37 CFR 1.16(i))	*	Minus	**	=	x =		OR	x =	
Independent (37 CFR 1.16(h))	*	Minus	***	=	x =		OR	x =	
Application Size Fee (37 CFR 1.16(s))									
FIRST PRESENTATION OF MULTIPLE DEPENDENT CLAIM (37 CFR 1.16(j))							OR		
					TOTAL ADD'L FEE		OR	TOTAL ADD'L FEE	

* If the entry in column 1 is less than the entry in column 2, write "0" in column 3.
** If the "Highest Number Previously Paid For" IN THIS SPACE is less than 20, enter "20".
*** If the "Highest Number Previously Paid For" IN THIS SPACE is less than 3, enter "3".
The "Highest Number Previously Paid For" (Total or Independent) is the highest found in the appropriate box in column 1.

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 NUMBER	FILING or 371(c) DATE	GRP ART UNIT	FIL FEE REC'D	ATTY.DOCKET.NO	TOT CLAIMS	IND CLAIMS
15/614,956	06/06/2017		0.00	007742.00091\US	1	1

CONFIRMATION NO. 9334

22907
BANNER & WITCOFF, LTD.
1100 13th STREET, N.W.
SUITE 1200
WASHINGTON, DC 20005-4051

FILING RECEIPT



Date Mailed: 06/15/2017

Receipt is acknowledged of this non-provisional patent application. The application will be taken up for examination in due course. Applicant will be notified as to the results of the examination. Any correspondence concerning the application must include the following identification information: the U.S. APPLICATION NUMBER, FILING DATE, NAME OF APPLICANT, and TITLE OF INVENTION. Fees transmitted by check or draft are subject to collection. Please verify the accuracy of the data presented on this receipt. **If an error is noted on this Filing Receipt, please submit a written request for a Filing Receipt Correction. Please provide a copy of this Filing Receipt with the changes noted thereon. If you received a "Notice to File Missing Parts" for this application, please submit any corrections to this Filing Receipt with your reply to the Notice. When the USPTO processes the reply to the Notice, the USPTO will generate another Filing Receipt incorporating the requested corrections**

Inventor(s)

Sean Moore, Hollis, NH;

Applicant(s)

Centripetal Networks, Inc., Herndon, VA;

Power of Attorney: The patent practitioners associated with Customer Number 22907

Domestic Priority data as claimed by applicant

This application is a CON of 14/625,486 02/18/2015 PAT 9686193
which is a CON of 13/795,822 03/12/2013 PAT 9124552

Foreign Applications for which priority is claimed (You may be eligible to benefit from the **Patent Prosecution Highway** program at the USPTO. Please see http://www.uspto.gov for more information.) - None.
Foreign application information must be provided in an Application Data Sheet in order to constitute a claim to foreign priority. See 37 CFR 1.55 and 1.76.

Permission to Access Application via Priority Document Exchange: Yes

Permission to Access Search Results: Yes

Applicant may provide or rescind an authorization for access using Form PTO/SB/39 or Form PTO/SB/69 as appropriate.

If Required, Foreign Filing License Granted: 06/13/2017

The country code and number of your priority application, to be used for filing abroad under the Paris Convention, is **US 15/614,956**

Projected Publication Date: To Be Determined - pending completion of Missing Parts

Non-Publication Request: No

Early Publication Request: No
**** SMALL ENTITY ****
Title

FILTERING NETWORK DATA TRANSFERS

Preliminary Class

Statement under 37 CFR 1.55 or 1.78 for AIA (First Inventor to File) Transition Applications: No

PROTECTING YOUR INVENTION OUTSIDE THE UNITED STATES

Since the rights granted by a U.S. patent extend only throughout the territory of the United States and have no effect in a foreign country, an inventor who wishes patent protection in another country must apply for a patent in a specific country or in regional patent offices. Applicants may wish to consider the filing of an international application under the Patent Cooperation Treaty (PCT). An international (PCT) application generally has the same effect as a regular national patent application in each PCT-member country. The PCT process **simplifies** the filing of patent applications on the same invention in member countries, but **does not result** in a grant of "an international patent" and does not eliminate the need of applicants to file additional documents and fees in countries where patent protection is desired.

Almost every country has its own patent law, and a person desiring a patent in a particular country must make an application for patent in that country in accordance with its particular laws. Since the laws of many countries differ in various respects from the patent law of the United States, applicants are advised to seek guidance from specific foreign countries to ensure that patent rights are not lost prematurely.

Applicants also are advised that in the case of inventions made in the United States, the Director of the USPTO must issue a license before applicants can apply for a patent in a foreign country. The filing of a U.S. patent application serves as a request for a foreign filing license. The application's filing receipt contains further information and guidance as to the status of applicant's license for foreign filing.

Applicants may wish to consult the USPTO booklet, "General Information Concerning Patents" (specifically, the section entitled "Treaties and Foreign Patents") for more information on timeframes and deadlines for filing foreign patent applications. The guide is available either by contacting the USPTO Contact Center at 800-786-9199, or it can be viewed on the USPTO website at http://www.uspto.gov/web/offices/pac/doc/general/index.html.

For information on preventing theft of your intellectual property (patents, trademarks and copyrights), you may wish to consult the U.S. Government website, http://www.stopfakes.gov. Part of a Department of Commerce initiative, this website includes self-help "toolkits" giving innovators guidance on how to protect intellectual property in specific countries such as China, Korea and Mexico. For questions regarding patent enforcement issues, applicants may call the U.S. Government hotline at 1-866-999-HALT (1-866-999-4258).

LICENSE FOR FOREIGN FILING UNDER

Title 35, United States Code, Section 184

Title 37, Code of Federal Regulations, 5.11 & 5.15

GRANTED

The applicant has been granted a license under 35 U.S.C. 184, if the phrase "IF REQUIRED, FOREIGN FILING LICENSE GRANTED" followed by a date appears on this form. Such licenses are issued in all applications where the conditions for issuance of a license have been met, regardless of whether or not a license may be required as set forth in 37 CFR 5.15. The scope and limitations of this license are set forth in 37 CFR 5.15(a) unless an earlier license has been issued under 37 CFR 5.15(b). The license is subject to revocation upon written notification. The date indicated is the effective date of the license, unless an earlier license of similar scope has been granted under 37 CFR 5.13 or 5.14.

This license is to be retained by the licensee and may be used at any time on or after the effective date thereof unless it is revoked. This license is automatically transferred to any related applications(s) filed under 37 CFR 1.53(d). This license is not retroactive.

The grant of a license does not in any way lessen the responsibility of a licensee for the security of the subject matter as imposed by any Government contract or the provisions of existing laws relating to espionage and the national security or the export of technical data. Licensees should apprise themselves of current regulations especially with respect to certain countries, of other agencies, particularly the Office of Defense Trade Controls, Department of State (with respect to Arms, Munitions and Implements of War (22 CFR 121-128)); the Bureau of Industry and Security, Department of Commerce (15 CFR parts 730-774); the Office of Foreign AssetsControl, Department of Treasury (31 CFR Parts 500+) and the Department of Energy.

NOT GRANTED

No license under 35 U.S.C. 184 has been granted at this time, if the phrase "IF REQUIRED, FOREIGN FILING LICENSE GRANTED" DOES NOT appear on this form. Applicant may still petition for a license under 37 CFR 5.12, if a license is desired before the expiration of 6 months from the filing date of the application. If 6 months has lapsed from the filing date of this application and the licensee has not received any indication of a secrecy order under 35 U.S.C. 181, the licensee may foreign file the application pursuant to 37 CFR 5.15(b).

SelectUSA

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 U.S. offers tremendous resources and advantages for those who invest and manufacture goods here. Through SelectUSA, our nation works to promote and facilitate business investment. SelectUSA provides information assistance to the international investor community; serves as an ombudsman for existing and potential investors; advocates on behalf of U.S. cities, states, and regions competing for global investment; and counsels U.S. economic development organizations on investment attraction best practices. To learn more about why the United States is the best country in the world to develop technology, manufacture products, deliver services, and grow your business, visit http://www.SelectUSA.gov or call +1-202-482-6800.

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 NUMBER	FILING OR 371(C) DATE	FIRST NAMED APPLICANT	ATTY. DOCKET NO./TITLE
15/614,956	06/06/2017	Sean Moore	007742.00091\US

CONFIRMATION NO. 9334

22907
BANNER & WITCOFF, LTD.
1100 13th STREET, N.W.
SUITE 1200
WASHINGTON, DC 20005-4051

FORMALITIES LETTER

OC000000092045087

Date Mailed: 06/15/2017

NOTICE TO FILE MISSING PARTS OF NONPROVISIONAL APPLICATION

FILED UNDER 37 CFR 1.53(b)

Filing Date Granted

Items Required To Avoid Abandonment:

An application number and filing date have been accorded to this application. The item(s) indicated below, however, are missing. Applicant is given **TWO MONTHS** from the date of this Notice within which to file all required items below to avoid abandonment. Extensions of time may be obtained by filing a petition accompanied by the extension fee under the provisions of 37 CFR 1.136(a).

- The statutory basic filing fee is missing.
- The application search fee must be submitted.
- The application examination fee must be submitted.
- Surcharge as set forth in 37 CFR 1.16(f) must be submitted.
  The surcharge is due for any one of:
  - late submission of the basic filing fee, search fee, or examination fee,
  - late submission of inventor's oath or declaration,
  - filing an application that does not contain at least one claim on filing, or
  - submission of an application filed by reference to a previously filed application.

SUMMARY OF FEES DUE:

The fee(s) required within **TWO MONTHS** from the date of this Notice to avoid abandonment is/are itemized below. Small entity discount is in effect. If applicant is qualified for micro entity status, an acceptable Certification of Micro Entity Status must be submitted to establish micro entity status. (See 37 CFR 1.29 and forms PTO/SB/15A and 15B.)

- $ **70** basic filing fee.
- $ **70** surcharge.
- $ **300** search fee.
- $ **360** examination fee.
- $(**0**) previous unapplied payment amount.
- $ **800** TOTAL FEE BALANCE DUE.

Replies must be received in the USPTO within the set time period or must include a proper Certificate of Mailing or Transmission under 37 CFR 1.8 with a mailing or transmission date within the set time period. For more information and a suggested format, see Form PTO/SB/92 and MPEP 512.

Replies should be mailed to:

Mail Stop Missing Parts
Commissioner for Patents
P.O. Box 1450
Alexandria VA 22313-1450

Registered users of EFS-Web may alternatively submit their reply to this notice via EFS-Web, including a copy of this Notice and selecting the document description "Applicant response to Pre-Exam Formalities Notice". https://sportal.uspto.gov/authenticate/AuthenticateUserLocalEPF.html

For more information about EFS-Web please call the USPTO Electronic Business Center at 1-866-217-9197 or visit our website at http://www.uspto.gov/ebc.

If you are not using EFS-Web to submit your reply, you must include a copy of this notice.

Questions about the contents of this notice and the requirements it sets forth should be directed to the Office of Data Management, Application Assistance Unit, at **(571) 272-4000** or **(571) 272-4200** or **1-888-786-0101**.

/mabebe/

Doc code: IDS
Doc description: Information Disclosure Statement (IDS) Filed

PTO/SB/08a (03-15)
Approved for use through 07/31/2016. OMB 0651-0031
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE
Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it contains a valid OMB control number.

INFORMATION DISCLOSURE STATEMENT BY APPLICANT

(Not for submission under 37 CFR 1.99)

Application Number	15614956
Filing Date	2017-06-06
First Named Inventor	Sean Moore
Art Unit	N/A
Examiner Name	Not Yet Assigned
Attorney Docket Number	007742.00091\US

U.S.PATENTS Remove

Examiner Initial*	Cite No	Patent Number	Kind Code[1]	Issue Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1					

If you wish to add additional U.S. Patent citation information please click the Add button. Add

U.S.PATENT APPLICATION PUBLICATIONS Remove

Examiner Initial*	Cite No	Publication Number	Kind Code[1]	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear
	1					

If you wish to add additional U.S. Published Application citation information please click the Add button. Add

FOREIGN PATENT DOCUMENTS Remove

Examiner Initial*	Cite No	Foreign Document Number[3]	Country Code[2] i	Kind Code[4]	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear	T[5]
	1							

If you wish to add additional Foreign Patent Document citation information please click the Add button Add

NON-PATENT LITERATURE DOCUMENTS Remove

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, pages(s), volume-issue number(s), publisher, city and/or country where published.	T[5]

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15614956
	Filing Date	2017-06-06
	First Named Inventor	Sean Moore
	Art Unit	N/A
	Examiner Name	Not Yet Assigned
	Attorney Docket Number	007742.00091\US

	1	2016 May 06 - (US) Office Action - App 14/714,207	
	2	2016 May 13 - (US) Office Action - App 13/940,240	
	3	2016 Jun 14 - (US) Office Action - App 14/625,486	
	4	2016 Feb 25 - (AU) Office Action - App 2014249055	
	5	2016 Feb 24 - (AU) Office Action - App 2014228257	
	6	2016 Jun 09 - (WO) International Search Report - PCT/US2016/026339	
	7	2016 Jun 16 - (CA) Office Action - App 2,888,935	
	8	2016 Jul 11 - (EP) Office Action - App 14720824.3	
	9	2016 Jul 22 - (US) Office Action - App 14/921,718	
	10	2016 Jul 20 - (AU) Office Action - App 2013335255	
	11	2016 Oct 05 - (US) Notice of Allowance - App 14/698,560	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15614956
	Filing Date	2017-06-06
	First Named Inventor	Sean Moore
	Art Unit	N/A
	Examiner Name	Not Yet Assigned
	Attorney Docket Number	007742.00091\US

	12	2016 Sept 13 - (CA) Office Action - App 2,902,206	
	13	2016 Sept 14 - (CA) Office Action - App 2,897,737	
	14	2016 Sept 26 - (CA) Office Action - App 2,902,158	
	15	2016 Oct 26 - (US) Office Action - App 13/940,240	
	16	2016 Nov 21 - (US) Office Action - App 14/745,207	
	17	2016 Dec 05 - (US) Notice of Allowance - App 14/714,207	
	18	Singh, Rajeev et al. "Detecting and Reducing the Denial of Service attacks in WLANs", December 2011, World Congress on Information and Communication TEchnologies, pp. 968-973	
	19	2017 Feb 10 - (US) Notice of Allowance - App 14/625,486	
	20	2017 Feb 15 - (US) Notice of Allowance - App 14/921,718	
	21	2017 Apr 12 - (US) Office Action - App 14/757,638	
	22	2017 Mar 06 - (WO) International Search Report and Written Opinion - App PCT/US2016/068008	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15614956
	Filing Date	2017-06-06
	First Named Inventor	Sean Moore
	Art Unit	N/A
	Examiner Name	Not Yet Assigned
	Attorney Docket Number	007742.00091\US

If you wish to add additional non-patent literature document citation information please click the Add button Add

EXAMINER SIGNATURE

Examiner Signature		Date Considered	

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

[1] See Kind Codes of USPTO Patent Documents at www.USPTO.GOV or MPEP 901.04. [2] Enter office that issued the document, by the two-letter code (WIPO Standard ST.3). [3] For Japanese patent documents, the indication of the year of the reign of the Emperor must precede the serial number of the patent document. [4] Kind of document by the appropriate symbols as indicated on the document under WIPO Standard ST.16 if possible. [5] Applicant is to place a check mark here if English language translation is attached.

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15614956
	Filing Date	2017-06-06
	First Named Inventor	Sean Moore
	Art Unit	N/A
	Examiner Name	Not Yet Assigned
	Attorney Docket Number	007742.00091\US

CERTIFICATION STATEMENT

Please see 37 CFR 1.97 and 1.98 to make the appropriate selection(s):

That each item of information contained in the information disclosure statement was first cited in any communication from a foreign patent office in a counterpart foreign application not more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(1).

OR

☐ That no item of information contained in the information disclosure statement 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 information disclosure statement was known to any individual designated in 37 CFR 1.56(c) more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(2).

See attached certification statement.

The fee set forth in 37 CFR 1.17 (p) has been submitted herewith.

☒ A certification statement is not submitted herewith.

SIGNATURE

A signature of the applicant or representative is required in accordance with CFR 1.33, 10.18. Please see CFR 1.4(d) for the form of the signature.

Signature	/Bradley C. Wright/	Date (YYYY-MM-DD)	2017-06-06
Name/Print	Bradley C. Wright	Registration Number	38,061

This collection of information is required by 37 CFR 1.97 and 1.98. 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 1 hour 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, VA 22313-1450. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. **SEND TO: Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450.**

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 the Freedom of Information Act requires disclosure of these record s.

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 inspections 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.

Electronic Acknowledgement Receipt

EFS ID:	29409097
Application Number:	15614956
International Application Number:	
Confirmation Number:	9334
Title of Invention:	FILTERING NETWORK DATA TRANSFERS
First Named Inventor/Applicant Name:	Sean Moore
Customer Number:	22907
Filer:	Bradley C. Wright/Holly Parker
Filer Authorized By:	Bradley C. Wright
Attorney Docket Number:	007742.00091\US
Receipt Date:	06-JUN-2017
Filing Date:	
Time Stamp:	14:16:34
Application Type:	Utility under 35 USC 111(a)

Payment information:

Submitted with Payment	no

File Listing:

Document Number	Document Description	File Name	File Size(Bytes)/ Message Digest	Multi Part /.zip	Pages (if appl.)
1	Non Patent Literature	12871806_17748590.pdf	1117022 140263fcd5145fb1e2e2a099b1e98f18b8d6d360	no	10

Warnings:

Information:					
2	Non Patent Literature	12871806_17748585.pdf	897641 06520e07f43a9afbc0f7b7ad730dff8247a967af	no	15
Warnings:					
Information:					
3	Non Patent Literature	12871806_17748581.pdf	1165960 0c1b258923b72f80b1dc6cdf7b6363c8a95a44dd	no	10
Warnings:					
Information:					
4	Non Patent Literature	12871806_17748578.pdf	1551967 88a0f067f9a24decfbe884fa2df29cc25ab0d0b5	no	14
Warnings:					
Information:					
5	Non Patent Literature	12871806_17748577.pdf	1143829 25ac578781f5ed2e25f8b34aa201a16e82b83daa	no	10
Warnings:					
Information:					
6	Non Patent Literature	12871806_17748574.pdf	1347663 c2f845908e887514e60f22eea11e9ec7546ca129	no	11
Warnings:					
Information:					
7	Non Patent Literature	12871806_17748567.pdf	1222050 3962296f0c45ec3f80f4d5cbf25eedfe9ef404b1	no	10
Warnings:					
Information:					
8	Non Patent Literature	12871806_17748564.pdf	1118094 8e2f6a7b03a382382e26a106186d64dd475fcedb	no	16
Warnings:					
Information:					

9	Non Patent Literature	12871806_17748559.pdf	959293 0848229de6e141860751e8e14d6e021a7da1b25f	no	11
Warnings:					
Information:					
10	Non Patent Literature	12871806_17748527.pdf	655973 538c772956eacec011f7a184a5ac5a01ee34d6bd	no	8
Warnings:					
Information:					
11	Non Patent Literature	12871806_17748517.pdf	527673 ec48fa1268dddc8f6dcd05783730b341339a1371	no	5
Warnings:					
Information:					
12	Non Patent Literature	12871806_17748507.pdf	214277 aa3925bbab5f9517b3d068f7238a2985fbc4eec9	no	2
Warnings:					
Information:					
13	Non Patent Literature	12871806_17748503.pdf	1104955 9a812673586643b63a12e826cdd6cb53c332b895	no	18
Warnings:					
Information:					
14	Non Patent Literature	12871806_17748500.pdf	1428072 2b7e6fb1104bdca0e149137487d7ed7da7f11317	no	12
Warnings:					
Information:					
15	Non Patent Literature	12871806_17748492.pdf	1206677 9796972ed4e3a9278f61305fd084edf6b1d8e87c	no	12
Warnings:					
Information:					

16	Non Patent Literature	12871806_17748477.pdf	820708 3b06c3828c0a94ea80bec34368675ee9b0d9661f	no	8
Warnings:					
Information:					
17	Non Patent Literature	12871806_17748473.pdf	1310896 32621dc86ff0594ecbf95b9eae06e336a994092c	no	15
Warnings:					
Information:					
18	Non Patent Literature	12871806_17748470.pdf	732085 2a34455f7f90840a6f7a21d383c468f4c799cac2	no	7
Warnings:					
Information:					
19	Non Patent Literature	12871806_17748468.pdf	1693547 ba5c31882f9787e2ecd23ed1ce54a0f53f07b8df	no	21
Warnings:					
Information:					
20	Non Patent Literature	12871806_17748464.pdf	1210320 146d8a2fe7e3bb0cf21b700f047ad15909315d15	no	25
Warnings:					
Information:					
21	Non Patent Literature	12871806_17748458.pdf	3239881 e395eba28f8604973c9bb837217d4d755d203ca3	no	40
Warnings:					
Information:					
22	Non Patent Literature	12871806_17748456.pdf	920276 190c9f0448edc31547b0cc19fbb60ebdc4103a57	no	16
Warnings:					
Information:					

23	Non Patent Literature	12871806_17748450.pdf	1331868 48b1a01defbf29b5f288ee8887e8c4467304d3d8	no	14
Warnings:					
Information:					
24	Non Patent Literature	12871806_17748446.pdf	463011 fbdf3061cc05d35a338f7f55aea7b621fea7fa9f	no	5
Warnings:					
Information:					
25	Non Patent Literature	12871806_17748444.pdf	486111 d98688f170b4b3d00908b0480ae591948adf4064	no	8
Warnings:					
Information:					
26	Non Patent Literature	12871806_17748442.pdf	1820688 8cd6d88bcf14c95f95d05c5f0936c59af596164f	no	27
Warnings:					
Information:					
27	Non Patent Literature	12871806_17748729.pdf	125961 d0e799e835b53bc9542ab489078408f1ddfd2faa	no	4
Warnings:					
Information:					
28	Non Patent Literature	12871806_17748727.pdf	1457696 fdaf7e08ea22271791869cdfea47c8ae05e27489	no	16
Warnings:					
Information:					
29	Non Patent Literature	12871806_17748748.pdf	202311 d1be9bb8cdc413159ad1666cfa4319695486413e	no	6
Warnings:					
Information:					

30	Information Disclosure Statement (IDS) Form (SB08)	Information_Disclosure_Statement_Fillable_PDF_3_of_3_18554775.pdf	1035683 db81feb6c53f4eb796fadc2ba27d6de3e45858a9	no	6

Warnings:

Information:

A U.S. Patent Number Citation or a U.S. Publication Number Citation is required in the Information Disclosure Statement (IDS) form for autoloading of data into USPTO systems. You may remove the form to add the required data in order to correct the Informational Message if you are citing U.S. References. If you chose not to include U.S. References, the image of the form will be processed and be made available within the Image File Wrapper (IFW) system. However, no data will be extracted from this form. Any additional data such as Foreign Patent Documents or Non Patent Literature will be manually reviewed and keyed into USPTO systems.

Total Files Size (in bytes):	32512188

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 a 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.

Doc code: IDS
Doc description: Information Disclosure Statement (IDS) Filed

PTO/SB/08a (03-15)
Approved for use through 07/31/2016. OMB 0651-0031
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE
Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it contains a valid OMB control number.

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (**Not for submission under 37 CFR 1.99)**		
	Application Number	15614956
	Filing Date	2017-06-06
	First Named Inventor	Sean Moore
	Art Unit	N/A
	Examiner Name	Not Yet Assigned
	Attorney Docket Number	007742.00091\US

U.S.PATENTS							Remove
Examiner Initial*	Cite No	Patent Number	Kind Code[1]	Issue Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear	
	1						
If you wish to add additional U.S. Patent citation information please click the Add button.							Add

U.S.PATENT APPLICATION PUBLICATIONS							Remove
Examiner Initial*	Cite No	Publication Number	Kind Code[1]	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear	
	1						
If you wish to add additional U.S. Published Application citation information please click the Add button.							Add

FOREIGN PATENT DOCUMENTS								Remove
Examiner Initial*	Cite No	Foreign Document Number[3]	Country Code[2] i	Kind Code[4]	Publication Date	Name of Patentee or Applicant of cited Document	Pages,Columns,Lines where Relevant Passages or Relevant Figures Appear	T[5]
	1							
If you wish to add additional Foreign Patent Document citation information please click the Add button								Add

NON-PATENT LITERATURE DOCUMENTS			Remove
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, pages(s), volume-issue number(s), publisher, city and/or country where published.	T[5]

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15614956
	Filing Date	2017-06-06
	First Named Inventor	Sean Moore
	Art Unit	N/A
	Examiner Name	Not Yet Assigned
	Attorney Docket Number	007742.00091\US

	1	J. Xu et al., "Design and Evaluation of a High-Performance ATM Firewall Switch and Its Applications", IEEE Journal on Selected Areas in Communications, 17(6): 1190-1200, June 1999	
	2	C. Benecke, "A Parallel Packet Screen for High Speed Networks", Proceedings of the 15th Annual Computer Security Applications Conference, 1999	
	3	R. Funke et al., "Performance Evaluation of Firewalls in Gigabit-Networks", Proceedings of the Symposium on Performance Evaluation of Computer and Telecommunication Systems, 1999	
	4	S. Suri et al., "Packet Filtering in High Speed Networks", Proceedings of the Symposium on Discrete Algorithms, 969-970, 1999	
	5	U. Ellermann et al., "Firewalls for ATM Networks", Proceedings of INFOSEC'COM, 1998	
	6	V. Srinivasan et al., "Fast and Scalable Layer Four Switching", Proceedings of ACM SIGCOMM, 191-202, 1998	
	7	M. Degermark et al., "Small Forwarding Tables for Fast Routing Lookups", Proceedings of ACM SIGCOMM, 4-13, 1997	
	8	S,M. Bellovin et al., "Network Firewalls", IEEE Communications Magazine, 50-57, 1994	
	9	W.E. Leland et al., "On the Self-Similar Nature of Ethernet Traffic", IEEE Transactions on Networking, 2(1); 15, 1994	
	10	G. Brightwell et al., "Counting Linear Extensions is #P-Complete", Proceedings of the Twenty-Third Annual ACM Symposium on Theory of Computing, 1991	
	11	M. Al-Suwaiyel et al., "Algorithms for Trie Compaction", ACM Transactions on Database Systems, 9(2): 243-263, June 1984	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15614956
	Filing Date	2017-06-06
	First Named Inventor	Sean Moore
	Art Unit	N/A
	Examiner Name	Not Yet Assigned
	Attorney Docket Number	007742.00091\US

	12	D. Comer, "Analysis of a Heuristic for Full Trie Minimization", ACM Transactions on Database Systems, 6(3): 513-537, Sept. 1981	
	13	R.L. Graham et al., "Optimization and Approximation in Deterministic Sequencing and Scheduling: A Survey", Annals of Discrete Mathematics, 5: 287-326, 1979	
	14	E.L. Lawler, "Sequencing Jobs to Minimize Total Weighted Completion oTime Subject to Precedence Constraints", Annals of Discrete Mathematics, 2: 75-90, 1978	
	15	J.K. Lenstra et al., "Complexity of Scheduling Under Precedence Constraints", Operations Research, 26(1): 22-35,1978	
	16	R. Rivest, "On Self-Organizing Sequential Search Heuristics", Communications of the ACM, 19(2): 1976	
	17	W.E. Smith, "Various Optimizers for Single-Stage Productions", Naval Research Logistics Quarterly, 3: 59-66, 1956	
	18	Bellion, "High Performance Packet Classification", http://www.hipac.org (Publication Date Unknown)	
	19	2011 Oct 18 - (EP) Communication Pursuant to Article 94(3) - App 06 758 213.0	
	20	2011 Jun 09 - (US) Notice of Allowance - App 11/390,976	
	21	2012 Jun 26 - (EP) Extended Search Report - App 05857614.1	
	22	2012 Jun 09 - (AU) Notice of Acceptance - App 2006230171	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15614956
	Filing Date	2017-06-06
	First Named Inventor	Sean Moore
	Art Unit	N/A
	Examiner Name	Not Yet Assigned
	Attorney Docket Number	007742.00091\US

	23	2011 Nov 11 - (AU) Second Office Action - App 2006230171	
	24	2013 Jan 17 - (CA) Office Action - App 2,600,236	
	25	2013 Jan 16 - (CA) Office Action - App 2,594,020	
	26	2012 Nov 20 - (EP) Communication under rule 71(3) - App 06 758 213.0	
	27	2013 Apr 18 - (EP) Decision to Grant a European Patent - App 06758212.0	
	28	2011 Aug 25 - (US) Non Final Rejection - App 12/871,806	
	29	2012 Feb 06 - (US) Final Rejection - App 12/871,806	
	30	2012 Aug 07 - (US) Non Final Rejection - App 12/871,806	
	31	2012 Nov 26 - (US) Final Rejection - App 12/871,806	
	32	2013 Apr 04 - (US) Notice of Allowance - App 12/871,806	
	33	2015 Jan 14 - (EP) Extended Search Report - App 10819667.6	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15614956
	Filing Date	2017-06-06
	First Named Inventor	Sean Moore
	Art Unit	N/A
	Examiner Name	Not Yet Assigned
	Attorney Docket Number	007742.00091\US

	34	2014 May 26 - (CA) Office Action - App 2010297968	
	35	2015 May 25 - (AU) Notice of Acceptance - App 2010297968	
	36	2015 May 14 - (US) Non Final Rejection - App 13/940,240	
	37	2015 Nov 27 - (US) Final Rejection - App 13/940,240	
	38	2015 Jul 10 - (WO) Communication Relating to the Results of the Partial International Search for International App - PCT/US2015/024691	
	39	2015 Jul 14 - (WO) International Preliminary Report on Patentability - App PCT/US2013/072566	
	40	2016 Jan 28 - (WO) International Search Report and Written Opinion - App PCT/US2015/062691	
	41	International Search Report and Written Opinion for International App. No. PCT/US2015/024691, dated September 16, 2015.	
	42	International Preliminary Report on Patentability for International App. No. PCT/US2013/057502, dated April 28, 2015.	
	43	International Preliminary Report on Patentability for International App. No. PCT/US2014/023286, dated September 15, 2015.	
	44	International Preliminary Report on Patentability for International App. No. PCT/US2014/027723, dated September 15, 2015.	

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15614956
	Filing Date	2017-06-06
	First Named Inventor	Sean Moore
	Art Unit	N/A
	Examiner Name	Not Yet Assigned
	Attorney Docket Number	007742.00091\US

	45	Statement RE: Related Application, dated July 24, 2015	
	46	2015 Dec 22 - (US) Final Office Action - App 14/714,207	
	47	2016 Feb 26 - (US) Non Final Office Action - App 14/253,992	
	48	2016 Apr 15 - (US) Notice of Allowance - App 14/855,374	
	49	2015 Nov 02 - (AU) Office Action - App 2013372879	
	50	2016 Apr 26 - (US) Office Action - App 14/745,207	

If you wish to add additional non-patent literature document citation information please click the Add button Add

EXAMINER SIGNATURE

Examiner Signature		Date Considered	

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

[1] See Kind Codes of USPTO Patent Documents at www.USPTO.GOV or MPEP 901.04. [2] Enter office that issued the document, by the two-letter code (WIPO Standard ST.3). [3] For Japanese patent documents, the indication of the year of the reign of the Emperor must precede the serial number of the patent document. [4] Kind of document by the appropriate symbols as indicated on the document under WIPO Standard ST.16 if possible. [5] Applicant is to place a check mark here if English language translation is attached.

INFORMATION DISCLOSURE STATEMENT BY APPLICANT (Not for submission under 37 CFR 1.99)		
	Application Number	15614956
	Filing Date	2017-06-06
	First Named Inventor	Sean Moore
	Art Unit	N/A
	Examiner Name	Not Yet Assigned
	Attorney Docket Number	007742.00091\US

CERTIFICATION STATEMENT

Please see 37 CFR 1.97 and 1.98 to make the appropriate selection(s):

That each item of information contained in the information disclosure statement was first cited in any communication from a foreign patent office in a counterpart foreign application not more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(1).

OR

☐ That no item of information contained in the information disclosure statement 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 information disclosure statement was known to any individual designated in 37 CFR 1.56(c) more than three months prior to the filing of the information disclosure statement. See 37 CFR 1.97(e)(2).

See attached certification statement.

The fee set forth in 37 CFR 1.17 (p) has been submitted herewith.

☒ A certification statement is not submitted herewith.

SIGNATURE

A signature of the applicant or representative is required in accordance with CFR 1.33, 10.18. Please see CFR 1.4(d) for the form of the signature.

Signature	/Bradley C. Wright/	Date (YYYY-MM-DD)	2017-06-06
Name/Print	Bradley C. Wright	Registration Number	38,061

This collection of information is required by 37 CFR 1.97 and 1.98. 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 1 hour 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, VA 22313-1450. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. **SEND TO: Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450.**

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 the Freedom of Information Act requires disclosure of these record s.

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 inspections 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.

Electronic Acknowledgement Receipt

EFS ID:	29408951
Application Number:	15614956
International Application Number:	
Confirmation Number:	9334
Title of Invention:	FILTERING NETWORK DATA TRANSFERS
First Named Inventor/Applicant Name:	Sean Moore
Customer Number:	22907
Filer:	Bradley C. Wright/Holly Parker
Filer Authorized By:	Bradley C. Wright
Attorney Docket Number:	007742.00091\US
Receipt Date:	06-JUN-2017
Filing Date:	
Time Stamp:	14:10:37
Application Type:	Utility under 35 USC 111(a)

Payment information:

Submitted with Payment	no

File Listing:

Document Number	Document Description	File Name	File Size(Bytes)/ Message Digest	Multi Part /.zip	Pages (if appl.)
1	Foreign Reference	EP1677484A2_16712298.pdf	773874 47dd5a5c781f0fd2a220829489e9b07dca656db2	no	14

Warnings:

Information:					
2	Foreign Reference	EP1313290A1_16689905.pdf	1181323 0d673c0a51b64a7a9da6dec238ebe98d483c6031	no	18
Warnings:					
Information:					
3	Foreign Reference	WO2005046145_16811218.PDF	1189526 12556d88fc51d8167206348609ca3e7c4e53943b	no	27
Warnings:					
Information:					
4	Foreign Reference	EP2385676A1_16818890.pdf	780191 b81189390d037dff528b0afcc8f407a761e32de5	no	17
Warnings:					
Information:					
5	Foreign Reference	EP2498442A1_16818909.pdf	1749069 e86e61d1c020da22685c864e51bdf4c636777f30	no	28
Warnings:					
Information:					
6	Foreign Reference	KR20010079361A_17690002.pdf	755268 996a657ccd2abe1d4285baa60bb510a5b1dccd29	no	19
Warnings:					
Information:					
7	Foreign Reference	EP1006701A2_17690010.pdf	779545 f2da2b4b1c1220d3c5ce4d5982f3fece2efff587	no	15
Warnings:					
Information:					
8	Foreign Reference	WO2006105093A2_17690109.pdf	2750332 d8f7a6023f473296c2e990d0aadf80b708b2c230	no	62
Warnings:					
Information:					

9	Foreign Reference	WO2006093557A2_17690117.pdf	3097033 8c3b461d2321cbe7ec5f8f229c40c85cbf0f9440	no	72
Warnings:					
Information:					
10	Foreign Reference	12871806_17748754.pdf	3103892 30b788b05754645d32577638f64f4565dd006372	no	74
Warnings:					
Information:					
11	Foreign Reference	WO2011038420A2_17690375.pdf	2007006 6c4c9c4f60be9e685c65752da01e8856c0f6cc73	no	34
Warnings:					
Information:					
12	Foreign Reference	EP1484884A2_17690384.pdf	1943998 a07deab9e2c710976dc4bd58d26f822f871d1c5d	no	32
Warnings:					
Information:					
13	Foreign Reference	12871806_17748768.pdf	2289665 5ceeb63cf91f9aad8cf39d604e915d3d8d04db49	no	60
Warnings:					
Information:					
14	Foreign Reference	EP1864226B1_17690406.pdf	2030938 b36cae0ea8bd094f45d8a5285b295af62b60849a	no	44
Warnings:					
Information:					
15	Foreign Reference	12871806_17750217.pdf	2385931 edf220ad5b5dc6ca0e270598a7fffb36d983a9ce	no	56
Warnings:					
Information:					

16	Foreign Reference	WO2012146265A1_17715972.pdf	4103078 4a100c960bd1ab5910a84302cda0371ff1bb080d	no	78
Warnings:					
Information:					
17	Foreign Reference	WO2007109541A2_18096458.pdf	8952906 32d020e6fc053874951d75f280bdb76a36fda8dd	no	179
Warnings:					
Information:					
18	Non Patent Literature	NPL_-_ISR_from_00774200023_as_filed_16712474.PDF	879049 0c802d5b79a6d7cc80dc33c3685743441eceec44	no	8
Warnings:					
Information:					
19	Non Patent Literature	NPL-ISR_dated_24-6-147742_16806852.PDF	519073 8f3879c91d1dd61cd1a7d73ef98e53309964ad85	no	9
Warnings:					
Information:					
20	Non Patent Literature	NPL-ISR_16811120.PDF	8697440 386f3734edd4d1111a0ce5eb49f047d48a7df8e4	no	66
Warnings:					
Information:					
21	Non Patent Literature	NPL-ISR_dated_2606_16818779.PDF	744984 e1ca96f7cd5b5a2fa2cccc54367d9a3ea7ad69bc	no	11
Warnings:					
Information:					
22	Non Patent Literature	NPL-Cisco_16818946.PDF	2472590 e14c692da0b6acaa6bb75cbae061bdd60efaea86	no	26
Warnings:					
Information:					

23	Non Patent Literature	NPL_Reumann_as_filed_17029214.PDF	506812 7ce72f2f751d31ee0dbebfba3e269204ac309c28	no	5
Warnings:					
Information:					
24	Non Patent Literature	NPL_Greenwald_as_filed_17029228.PDF	1453816 f534647935f58bfb9e25964260695c7bb3863dda	no	14
Warnings:					
Information:					
25	Non Patent Literature	NPL-Mizuno_17259625.PDF	363361 4fd1c88496c3f87fd05dd88cbf06087f412ac8d6	no	3
Warnings:					
Information:					
26	Non Patent Literature	NPL-Moore_16818826.PDF	1932721 7f7362dd6ca63b62d069e3a66279ca0175c2da98	no	9
Warnings:					
Information:					
27	Non Patent Literature	NPL-Kindervag_16712444.PDF	2836187 7320c5df93a3c25807d980d1ff9302ac28250a18	no	27
Warnings:					
Information:					
28	Non Patent Literature	NPL-PaloAltoNetworks_16712439.PDF	2240313 1a612b6fe32ae1fdcce919ae95025d31865f5c76	no	11
Warnings:					
Information:					
29	Non Patent Literature	Non-Final_Rejection__892__1449_17666796.pdf	3313609 f94b21bf5072e9131e28903be6edbe2c064d83d3	no	82
Warnings:					
Information:					

30	Non Patent Literature	12871806_17760757.pdf	133950 d087d3fbbe5474766fb472b2fee3ab27b0b1d25c	no	3
Warnings:					
Information:					
31	Non Patent Literature	12871806_17748716.pdf	1885109 ff333809c196b7f7a899c06c24df7be6148b7b74	no	27
Warnings:					
Information:					
32	Non Patent Literature	12871806_17748705.pdf	72597 651be9b6f8516e40bd49cb8fea3b32608f57ad50	no	1
Warnings:					
Information:					
33	Non Patent Literature	12871806_17748704.pdf	280016 221a9d9fc8b69cd190f9a5e0c1fa7680eb10d7d4	no	8
Warnings:					
Information:					
34	Non Patent Literature	12871806_17748700.pdf	219866 2ff7ed11de580c33fbb1b4952953d81e9b627c07	no	6
Warnings:					
Information:					
35	Non Patent Literature	12871806_17748691.pdf	1056941 bfadf021c891e2d729069fdda5bb30ab7fbf6763	no	16
Warnings:					
Information:					
36	Non Patent Literature	12871806_17748689.pdf	93553 aeebc179c8fe4b8652ce76890598f0a71ce1aaa8	no	2
Warnings:					
Information:					

37	Non Patent Literature	12871806_17748686.pdf	66979 3884e538b6229e78e030c0dadc818f3747c9c505	no	1
Warnings:					
Information:					
38	Non Patent Literature	12871806_17748681.pdf	996456 64c55e7adf26bce0057805bbce8893d9484d48a2	no	30
Warnings:					
Information:					
39	Non Patent Literature	12871806_17748678.pdf	156743 31b7acfb7a47df115a561685241059d1d659011d	no	3
Warnings:					
Information:					
40	Non Patent Literature	12871806_17748674.pdf	954410 43e217a062cbd30d39f335589611d5c666afc0b0	no	22
Warnings:					
Information:					
41	Non Patent Literature	12871806_17748670.pdf	2407621 8036abd1045aaa8ace8ddc328d101c5e3756a533	no	40
Warnings:					
Information:					
42	Non Patent Literature	12871806_17748667.pdf	2047479 88a901174011cc43df79b97eefab85b253c4402e	no	29
Warnings:					
Information:					
43	Non Patent Literature	WO2006105093_17760770.pdf	130515 ffedf6884c4fc8189503e9ff67e119dd3c780109	no	3
Warnings:					
Information:					

44	Non Patent Literature	12871806_17748664.pdf	237193 7e270b000e2db4b9499e9c0955d46568991f37b2	no	4
Warnings:					
Information:					
45	Non Patent Literature	12871806_17748659.pdf	954305 3f4b7c0c39cc5a790a91fb62c5377170c2dd5d5b	no	11
Warnings:					
Information:					
46	Non Patent Literature	WO2006093557_17760776.pdf	330618 8b1a7002192e2958fc5a052a07ea100d87f170f4	no	6
Warnings:					
Information:					
47	Non Patent Literature	12871806_17748656.pdf	680644 f6452d529346441b5a7b66f39d8cb7b835fc821f	no	6
Warnings:					
Information:					
48	Non Patent Literature	12871806_17748652.pdf	809858 22ec24e861a9992648686b98ef88ea8c83a07bab	no	8
Warnings:					
Information:					
49	Non Patent Literature	12871806_17748644.pdf	843872 c9065af421ba9ec73f59a3c0abbae1bf99b75614	no	12
Warnings:					
Information:					
50	Non Patent Literature	12871806_17748638.pdf	1048051 62d2f5d39bb17ff27c66337e49aee4f6c9830c66	no	17
Warnings:					
Information:					

51	Non Patent Literature	12871806_17748635.pdf	910845 6e552f84cff6cf8d14434127ae71adb7df616dec	no	10
Warnings:					
Information:					
52	Non Patent Literature	12871806_17748632.pdf	1400185 d1485e17039f0972c9bca74d770ed585d940bd53	no	20
Warnings:					
Information:					
53	Non Patent Literature	12871806_17748622.pdf	1289250 fcdde68b1fc3570582a7a07ca4720c19537e39da	no	14
Warnings:					
Information:					
54	Non Patent Literature	12871806_17748618.pdf	585331 8f5ff661f447eb4f39db56058ccd1229bcaa0dfc	no	6
Warnings:					
Information:					
55	Non Patent Literature	12871806_17748614.pdf	1233391 f84378a5a084fed2aaf6cba80ececce4a98f2ba0	no	28
Warnings:					
Information:					
56	Non Patent Literature	12871806_17748610.pdf	266637 d6500988f54cbcebcbe36d73eca657359214084b	no	4
Warnings:					
Information:					
57	Non Patent Literature	12871806_17748605.pdf	888581 179b281ea068d86062d3084a0dbbc4632778ce46	no	10
Warnings:					
Information:					

58	Non Patent Literature	12871806_17748600.pdf	943167 06b16c35524d8f1d1a800b00e6f9293a6e11f14b	no	9
Warnings:					
Information:					
59	Non Patent Literature	12871806_17748594.pdf	1313706 cb35a896663a5ce778fd6ac66689f1bbdee57433	no	14
Warnings:					
Information:					
60	Information Disclosure Statement (IDS) Form (SB08)	Information_Disclosure_Statement_Fillable_PDF_2_of_3_18554772.pdf	1037188 8e04cbe2ff5e23f1f2dcb74f601412e893cf59ca	no	8
Warnings:					
Information:					

A U.S. Patent Number Citation or a U.S. Publication Number Citation is required in the Information Disclosure Statement (IDS) form for autoloading of data into USPTO systems. You may remove the form to add the required data in order to correct the Informational Message if you are citing U.S. References. If you chose not to include U.S. References, the image of the form will be processed and be made available within the Image File Wrapper (IFW) system. However, no data will be extracted from this form. Any additional data such as Foreign Patent Documents or Non Patent Literature will be manually reviewed and keyed into USPTO systems.

Total Files Size (in bytes):	91108587

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 a 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.

Espacenet

Bibliographic data: EP1677484 (A2) — 2006-07-05

Method and system for distributing security policies

No documents available for this priority number.

Inventor(s): PALEOLOGU EMANUEL [US]; YOUNGBLUT ERIC E [US]; IVANOV MAXIM A [US]; NAGAMPALLI NARASIMHA RAO S S [US]; SHETH SACHIN C [US]; KOTI SHIRISH R [US]; LIN YUN [US] ± (PALEOLOGU, EMANUEL, ; YOUNGBLUT, ERIC E, ; IVANOV, MAXIM A, ; NAGAMPALLI, NARASIMHA RAO, ; SHETH, SACHIN C, ; KOTI, SHIRISH R, ; LIN, YUN, ; NAGAMPALLI, NARASIMHA RAO S.S)

Applicant(s): MICROSOFT CORP [US] ± (MICROSOFT CORPORATION)

Classification: **- international:** ***H04L29/06***
- cooperative: **H04L63/0218**; **H04L63/0263**; **H04L63/20**

Application number: EP20050110420 20051107

Priority number(s): US20040993688 20041119

Also published as: EP1677484 (A3) US2006129808 (A1) US7509493 (B2) US2009172774 (A1) US7831826 (B2) more

Abstract of EP1677484 (A2)

A method and system for distributing and enforcing security policies is provided. A firewall agent executing at a host computer system that is to be protected receives security policies for the enforcement engines responsible for enforcing the security policies on the host computer system. A security policy has rules that each provide a condition and action to be performed when the condition is satisfied. A rule also has a rule type that is used by the distribution system to identify the security components that are responsible for enforcing the rules. To distribute the security policies that have been received at a host computer system, the firewall agent identifies to which enforcement engine a rule applies based in part on rule type. The firewall agent then distributes the rule to the identified enforcement engine, which then enforces the rule.

(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) EP 1 677 484 A2

(12) EUROPEAN PATENT APPLICATION

(43) Date of publication:
05.07.2006 Bulletin 2006/27

(51) Int Cl.:
H04L 29/06 *(2006.01)*

(21) Application number: **05110420.6**

(22) Date of filing: **07.11.2005**

(84) Designated Contracting States:
AT BE BG CH CY CZ DE DK EE ES FI FR GB GR HU IE IS IT LI LT LU LV MC NL PL PT RO SE SI SK TR
Designated Extension States:
AL BA HR MK YU

(30) Priority: **19.11.2004 US 993688**

(71) Applicant: **MICROSOFT CORPORATION**
Redmond, Washington 98052-6399 (US)

(72) Inventors:
- **Paleologu, Emanuel,**
  cO Microsoft Corporation
  Redmond,
  WA 98052 (US)
- **Youngblut, Eric E.,**
  c/o Microsoft Corporation
  Redmond,
  WA 98052 (US)
- **Ivanov, Maxim A.,**
  c/o Microsoft Corporation
  Redmond,
  WA 98052 (US)
- **Nagampalli, Narasimha Rao S.S.,**
  c/o Microsoft Corp
  Redmond,
  WA 98052 (US)
- **Sheth, Sachin C.,**
  c/o Microsoft Corporation
  Redmond,
  WA 98052 (US)
- **Koti, Shirish R.,**
  c/o Microsoft Corporation
  Redmond,
  WA 98052 (US)
- **Lin, Yun,**
  c/o Microsoft Corporation
  Redmond,
  WA 98052 (US)

(74) Representative: **Schuster, Thomas**
Grünecker, Kinkeldey,
Stockmair & Schwanhäusser
Anwaltssozietät
Maximilianstrasse 58
D-80538 München (DE)

(54) Method and system for distributing security policies

(57) A method and system for distributing and enforcing security policies is provided. A firewall agent executing at a host computer system that is to be protected receives security policies for the enforcement engines responsible for enforcing the security policies on the host computer system. A security policy has rules that each provide a condition and action to be performed when the condition is satisfied. A rule also has a rule type that is used by the distribution system to identify the security components that are responsible for enforcing the rules. To distribute the security policies that have been received at a host computer system, the firewall agent identifies to which enforcement engine a rule applies based in part on rule type. The firewall agent then distributes the rule to the identified enforcement engine, which then enforces the rule.

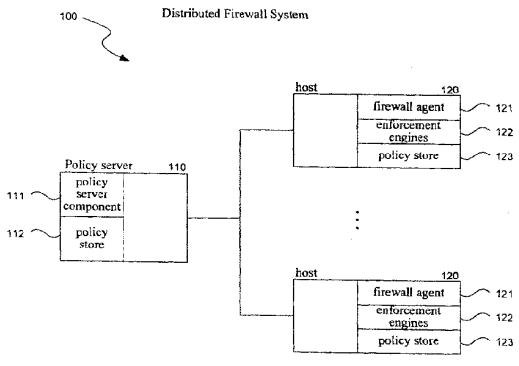


FIG. 1

EP 1 677 484 A2

Description

TECHNICAL FIELD

[0001] The described technology relates generally to systems that prevent the exploitation of vulnerabilities of computer systems.

BACKGROUND

[0002] Although the Internet has had great successes in facilitating communications between computer systems and enabling electronic commerce, the computer systems connected to the Internet have been under almost constant attack by hackers seeking to disrupt their operation. Many of the attacks seek to exploit vulnerabilities of software systems including application programs or other computer programs executing on those computer systems. Developers of software systems and administrators of computer systems of an enterprise go to great effort and expense to identify and remove vulnerabilities. Because of the complexity of software systems, however, it is virtually impossible to identify and remove all vulnerabilities before software systems are released. After a software system is released, developers can become aware of vulnerabilities in various ways. A party with no malicious intent may identify a vulnerability and may secretly notify the developer so the vulnerability can be removed before a hacker identifies and exploits it. If a hacker identifies a vulnerability first, the developer may not learn of the vulnerability until it is exploited—sometimes with disastrous consequences.

[0003] Regardless of how a developer finds out about a vulnerability, the developer typically develops and distributes to system administrators "patches" or updates to the software system that remove the vulnerability. If the vulnerability has not yet been exploited (e.g., might not be known to hackers), then a developer can design, implement, test, and distribute a patch in a disciplined way. If the vulnerability has already been widely exposed, then the developer may rush to distribute a patch without the same care that is used under normal circumstances. When patches are distributed to the administrators of the computer systems, they are responsible for scheduling and installing the patches to remove the vulnerabilities.

[0004] Unfortunately, administrators often delay the installation of patches to remove vulnerabilities for various reasons. When a patch is installed, the software system and possibly the computer system on which it is executing may need to be shut down and restarted. If the vulnerability is in a software system that is critical to the success of an organization, then the administrator needs to analyze the tradeoffs of keeping the software system up and running with its associated risk of being attacked and of shutting down a critical resource of the enterprise to install the patch. Some administrators may delay the installation of the patch because they fear that, because of a hasty distribution, it might not be properly tested and have unintended side effects. If the patch has an unintended side effect, then the software system, the computer system, or some other software component that is impacted by the patch may be shut down by the patch itself. Administrators need to factor in the possibility of an unintended side effect when deciding whether to install a patch. These administrators may delay installing a patch until experience by others indicates that there are no serious unintended side effects.

[0005] Intrusion detection systems have been developed that can be used to identify whether an attempt is being made to exploit a known vulnerability that has not yet been patched. These intrusion detection systems can be used to prevent exploitations of newly discovered vulnerabilities for which patches have not yet been developed or installed. These intrusion detection systems may define a "signature" for each way a vulnerability can be exploited. For example, if a vulnerability can be exploited by sending a certain type of message with a certain attribute, then the signature for that exploitation would specify that type and attribute. When a security enforcement event occurs, such as the receipt of a message, the intrusion detection system checks its signatures to determine whether any match the security enforcement event. If so, the intrusion detection system may take action to prevent the exploitation, such as dropping the message.

[0006] Signatures for newly discovered exploitations of vulnerabilities can be created in different ways. Developers of intrusion detection systems may create and distribute new signatures when they become aware of new exploitations. An administrator can then install the new signatures to prevent the exploitation. A developer may not, however, provide signatures for all known exploitations. For example, the vulnerability may be in a special-purpose application program that the developer does not support. To prevent exploitation of such a vulnerability, intrusion detection systems may allow administrators to create their own signatures.

[0007] A set of one or more signatures may be considered a security policy. Developers of intrusion detection systems may provide various security policies. For example, a developer may provide one security policy that defines signatures of vulnerabilities of an operating system and many other security policies that are specific to an application or a class of applications. Similarly, an administrator may define a security policy that is specific to custom applications used by the enterprise.

[0008] Because intrusions can occur at various points within an operating system or an application, intrusion detection systems have been developed to detect and prevent exploitation of vulnerabilities at each of these points. For example, an intrusion detection system may be developed to prevent exploitation of vulnerabilities that can be detected at the network layer of a communication protocol, and another intrusion detection system may be developed to prevent exploitation of vulnerabilities that can be detected when a file is accessed. Each

IPR2023-01353 Page 00517

developer of an intrusion detection system can provide their own implementation and security policies that need to be distributed to each computer system to be protected. In addition, any updates to the implementation or security policies also need to be distributed to the computer systems to be protected.

[0009] The distribution and updating of security policies for an enterprise with a large number of computer systems can be a complex and time-consuming task. Each security policy needs to be provided to each computer system and then to the appropriate intrusion detection system that is installed on the computer system. Since the developers of intrusion detection systems typically develop their systems independent of each other, each intrusion detection system may have a proprietary mechanism for defining and distributing security policies. A system administrator of an enterprise may need to be aware of each of these mechanisms to use the intrusion detection system effectively.

[0010] It would be desirable to have a mechanism that would provide a uniform way to distribute security policies received at a computer system to components responsible for enforcing the security policies.

SUMMARY

[0011] A method and system for distributing and enforcing security policies is provided. A distributed firewall system includes a policy server component, firewall agents, and enforcement engines. A firewall agent executing at a host computer system that is to be protected receives from the policy server component security policies for the enforcement engines responsible for enforcing the security policies on the host computer system. A security policy has rules that each provide a condition and action to be performed when the condition is satisfied. A rule also has a rule type (also referred to as a security type) that is used by the firewall agent to identify the enforcement engines that are responsible for enforcing the rules. The firewall agent may distribute the security policies to enforcement engines executing in user mode and in kernel mode. The enforcement engines may provide a layered approach to security enforcement. To distribute the security policies that have been received at a host computer system, the firewall agent identifies to which enforcement engine a rule applies based in part on rule type. The firewall agent then distributes the rule to the identified enforcement engine, which then enforces the rule.

BRIEF DESCRIPTION OF THE DRAWINGS

[0012] Figure 1 is a block diagram that illustrates the distributed firewall system in one embodiment.

[0013] Figure 2 is a block diagram that illustrates the components of the portions of the distributed firewall system executing on a host computer system in one embodiment.

[0014] Figure 3 is a flow diagram that illustrates the processing of the distribute policy component of the firewall agent in one embodiment.

[0015] Figure 4 is a flow diagram that illustrates the processing of the flow manager component in one embodiment.

[0016] Figure 5 is a flow diagram that illustrates the processing of an enforcement engine for a specific layer in one embodiment.

DETAILED DESCRIPTION

[0017] A method and system for distributing and enforcing security policies is provided. A distributed firewall system includes a policy server component, firewall agents, and enforcement engines. A firewall agent executing at a host computer system that is to be protected receives from a policy server security policies for the enforcement engines responsible for enforcing the security policies on the host computer system. A security policy has an execution scope and associated rules. An execution scope defines the attributes of a process or application to which the associated rules are to be applied. A rule provides a condition and action to be performed when the condition is satisfied. A rule also has a rule type (also referred to as a security type) that is used by the firewall agent to identify the enforcement engines that are responsible for enforcing the rules. The firewall agent may distribute the security policies to enforcement engines executing in user mode and in kernel mode. The enforcement engines may provide a layered approach to a distributed firewall. For example, enforcement engines may be installed on a host computer system to prevent intrusions at various layers of the ISO 7-layer reference model. The network layer, the transport layer, and the session layer may each have an enforcement engine that is developed to detect attempted intrusions at that layer. To distribute the security policies that have been received at a host computer system, the firewall agent identifies to which enforcement engine a rule applies based in part on rule type. The firewall agent then distributes the rule to the identified enforcement engine, which then enforces the rule. Upon receiving a rule, an enforcement engine may distribute the rule to its subcomponents as appropriate. For example, the firewall agent may distribute all rules destined for kernel-mode enforcement engines to a single kernel-mode security component that then distributes the rules to the enforcement engines. In this way, the distributed firewall system provides a single mechanism for distributing rules of security policies to the enforcement engines providing different layers of protection for a host computer system.

[0018] In one embodiment, a security policy includes rules that specify conditions, actions, and optionally exceptions. For example, a rule may indicate that an application may not receive messages on any network port other than port 80. The condition of the rule may be satisfied when a message is received on a network port

IPR2023-01353 Page 00518

other than port 80, and the action may result in the message being discarded. As another example, a rule may indicate that when an application attempts to send a message on network port 80, authorization is to be solicited from a user of the computing system on which the rule is enforced. The rules may be categorized into rule or security types based on their behavior. For example, rules with a network security type may be directed to security enforcement for network traffic. Each security type may have its own security component for enforcing rules of that security type. Languages for specifying security policies is described in U.S. Patent Application No. 10/882,438, entitled "Languages for Expressing Security Policies" and filed on July 1, 2004, which is hereby incorporated by reference.

[0019] The condition, action, and exception of the rules may be specified as expressions. A rule may be semantically expressed as "IF conditions THEN actions EXCEPT exceptions." Conditions of a rule are expressions of circumstances under which security enforcement actions of the rule are to be performed. An action is an expression of activity to be performed when the condition is satisfied. A rule may have multiple actions. An exception is an expression of when the actions may not be performed even though the condition is satisfied. A condition may be either static or dynamic. A static condition is one which refers, for example, to a hard-coded list of files. A dynamic condition is one which, for example, performs a query to determine a list of files matching a provided criterion. Security enforcement actions may include allowing a request that caused the security enforcement event, denying the request, soliciting input from a user, notifying the user, and so on. Other rule constructs are also possible. As an example, an "else" construct could be added to perform alternate actions when a condition is false.

[0020] Figure 1 is a block diagram that illustrates the distributed firewall system in one embodiment. An enterprise may have a network system 100 that includes a policy server computer system 110 and host computer systems 120. A system administrator uses a policy server component 111 of the policy server computer system to prepare and distribute security policies stored in a security policy store 112 to the host computer systems. Each host computer system includes a firewall agent 121, enforcement engines 122, and a policy store 123. The firewall agent provides a central mechanism through which a host computer system receives and distributes security policies to the enforcement engines of that host computer system. Upon receiving a security policy from the policy server computer system, the firewall agent of a host computer system stores the security policy in the policy store. The firewall agent then processes the security policies on a rule-by-rule basis and provides the rules to the appropriate enforcement engine. The firewall agent may combine various security policies, order the rules of the security policies, and dynamically notify the enforcement engines of changes to security policies or system configurations that will affect the enforcement of the security policies. Such combining, ordering, and notifying is described in U.S. Patent Application No. 10/966,800, entitled "Method and System for Merging Security Policies" and filed on October 14, 2004, which is hereby incorporated by reference. The enforcement engines then enforce the rules by applying them to network events. The enforcement engines may provide various type of security enforcement. An enforcement engine may detect an attempt to perform a certain behavior and block the attempt. For example, an application that is infected with a virus may attempt to delete certain files of the operating system. A security engine executing in the same process space as the application may detect and block the attempt to delete the file. A security engine for performing behavior blocking is described in U.S. Patent Application No. 10/832,798, entitled "A Method and System for Enforcing a Security Policy via a Security Virtual Machine" and filed on April 27, 2004, which is hereby incorporated by reference. The enforcement engines may be adapted to apply rules to network events of various protocols, such as IP, ICMP, TCP, FTP, DNS, HTTP, RPC, and so on. Techniques preventing exploitation of vulnerabilities are described in U.S. Patent Application No. 10/955,963, entitled "Method and System for Filtering Communications to Prevent Exploitation of a Software Vulnerability" and filed on September 30, 2004, which is hereby incorporated by reference.

[0021] Figure 2 is a block diagram that illustrates the components of the distributed firewall system executing on a host computer system in one embodiment. The host computer system includes firewall agent components 220, kernel-mode components 230, and application user-mode components 240. The firewall agent components include a receive policy component 221, a parse policy component 222, and a distribute policy component 223. The firewall agent components also include a policy store 224 and a configuration manager 225. The receive policy component receives security policies provided by the policy server computer system and stores the policies in the policy store. The parse policy component retrieves the policies from the policy store and converts them to a lower-level language for processing by the enforcement engines. The distribute policy component receives the parsed security policies and distributes the rules of the security policies to the appropriate enforcement engines. The configuration manager provides configuration information for use in determining which rules should be distributed to which enforcement engines based on the current configuration of the host computer system.

[0022] The kernel-mode components include enforcement engines 231, flow manager components 233, and protocol hooks 234. The components may be layered from layer 0 to layer N, which may correspond to various layers of the ISO 7-layer reference model. Each layer may have an associated protocol hook, flow manager, and enforcement engine that is specific to the layer, parses network events specific to that layer, and enforces the

IPR2023-01353 Page 00519

rules provided by the firewall agent to that layer. The rules for each layer are stored in a rule store 232 that may be specific for that layer. The flow manager components are responsible for intercepting network events and invoking the corresponding enforcement engine to enforce the rules. Each flow manager component receives network events from the corresponding protocol hook that is responsible for detecting network events at the corresponding layer. The application user-mode components may be provided as a dynamic link library that links into an application for providing layered security in a manner that is similar in overall architecture to the kernel-mode components.

[0023] The computing device on which the distribution system is implemented may include a central processing unit, memory, input devices (e.g., keyboard and pointing devices), output devices (e.g., display devices), and storage devices (e.g., disk drives). The memory and storage devices are computer-readable media that may contain instructions that implement the distribution system. In addition, the data structures and message structures may be stored or transmitted via a data transmission medium, such as a signal on a communications link. Various communication links may be used, such as the Internet, a local area network, a wide area network, a point-to-point dial-up connection, a cell phone network, and so on.

[0024] The distribution system may be implemented in various operating environments that include personal computers, server computers, hand-held or laptop devices, multiprocessor systems, microprocessor-based systems, programmable consumer electronics, digital cameras, network PCs, minicomputers, mainframe computers, distributed computing environments that include any of the above systems or devices, and so on. The host computer systems may be cell phones, personal digital assistants, smart phones, personal computers, programmable consumer electronics, digital cameras, and so on.

[0025] The distribution system may be described in the general context of computer-executable instructions, such as program modules, executed by one or more computers or other devices. Generally, program modules include routines, programs, objects, components, data structures, and so on that perform particular tasks or implement particular abstract data types. Typically, the functionality of the program modules may be combined or distributed as desired in various embodiments.

[0026] Figure 3 is a flow diagram that illustrates the processing of the distribute policy component of the firewall agent in one embodiment. Upon receiving policies from the parse policy component, the component identifies the enforcement engines to which the rules apply and distributes the rules to the identified enforcement engines. In blocks 301-307, the component loops selecting the security policies. In block 301, the component selects the next security policy. In decision block 302, if all the security policies have already been selected, then the component completes, else the component continues at block 303. In blocks 303-307, the component loops processing each rule within the selected security policy. In block 303, the component selects the next rule of the selected security policy. In decision block 304, if all the rules of the selected security policy have already been selected, then the component loops to block 301 to select the next security policy, else the component continues at block 305. In block 305, the component identifies the rule type (or security type) of the selected security rule. In block 306, the component identifies the enforcement engine that is responsible for enforcing the rules of the identified rule type. In block 307, the component provides the selected rule to the identified component and then loops to block 303 to select the next rule of the selected security policy.

[0027] Figure 4 is a flow diagram that illustrates the processing of the flow manager component in one embodiment. The flow manager component is passed a network event and invokes the appropriate enforcement engine for processing the network event. The flow manager component may be implemented as a separate flow manager for each layer that processes network events only for that layer or may be implemented as a single flow manager component that process network events for all layers. In this illustrated embodiment, the flow manager component processes all network events for all layers. In decision blocks 401-403, the component identifies the layer to which a network event applies. The component then invokes the associated enforcement engine for that layer in blocks 411-413. The component then completes. The flow manager may maintain a separate flow for each network connection. The enforcement engines and flow managers may be specific to the layer and the protocol implemented by the layer. For example, when a connection uses the Transmission Control Protocol ("TCP") or the User Datagram Protocol ("UDP") at the packet layer, the flow manager invokes the enforcement engine that is appropriate for the protocol. The invoked enforcement engines may instantiate data structures for tracking state information from one invocation to the next. Each application that is protected by user-mode enforcement engine may also have a flow manager with a separate flow for each connection.

[0028] Figure 5 is a flow diagram that illustrates the processing of an enforcement engine for a specific layer in one embodiment. The enforcement engine, which is a component of the distributed firewall system, is provided a network event, identifies the rules to apply to the event, determines whether the conditions of the rules are satisfied, and if so, performs the action associated with the rule. In blocks 501-504, the component loops selecting each rule associated with the layer of the enforcement engine. In block 501, the component selects the next rule. In decision block 502, if all the rules have already been selected, then the component completes, else the component continues at block 503. In decision block 503, if the condition of the rule is satisfied, then the component continues at block 504, else the component loops to block

IPR2023-01353 Page 00520

501 to select the next rule. In block 504, the component performs the action associated with the rule and then loops to block 501 to select the next rule. Each enforcement engine may be adapted to process the network events and rules that are appropriate to its layer. For example, an enforcement engine for the network layer may parse network packets and enforce rules related to the network packets. An enforcement engine for the application layer may detect unauthorized attempts to access or modify files relating to that application.

[0029] From the foregoing, it will be appreciated that specific embodiments of the distributed firewall system have been described herein for purposes of illustration, but that various modifications may be made without deviating from the spirit and scope of the invention. Accordingly, the invention is not limited except as by the appended claims.

Claims

1. A method in a computer system for distributing rules of security policies to enforcement engines for enforcing the security policies, the method comprising:

   providing at the computer system enforcement engines that implement different layers of security enforcement;
   receiving and storing at the computer system security policies having rules, each rule having a rule type;
   under control of a firewall agent executing on the computer system,
   retrieving the stored security policies; and
   for rules of a retrieved security policy,
   identifying an enforcement engine to which a rule applies based on the rule type of the rule; and
   providing the rule to the identified enforcement engine; and under control of the enforcement engines, enforcing the rules provided to the enforcement engine by the firewall agent,
   wherein the firewall agent provides a mechanism for distributing the rules to multiple enforcement engines of the computer system.

2. The method of claim 1 wherein the security policies are received from policy computer system that distributes the security policies for multiple computer systems.

3. The method of claim 1 wherein the enforcement engines provide a layered firewall.

4. The method of claim 1 wherein the enforcement engines include a behavior blocking security component.

5. The method of claim 1 wherein an enforcement engine includes subcomponents and wherein upon being provided with a rule, the enforcement engine provides the rule to a subcomponent.

6. The method of claim 1 wherein at least one enforcement engine executes in kernel mode and at least one enforcement engine executes in user mode.

7. The method of claim 1 wherein the firewall agent includes a user-mode subcomponent and a kernel-mode subcomponent, and the user-mode subcomponent distributes rules to the kernel-mode subcomponent for providing the rules to enforcement engines that execute in kernel mode.

8. The method of claim 1 wherein a flow manager component intercepts network events at the various layers and invokes an enforcement engine associated with a layer to enforce the rules for that layer.

9. The method of claim 8 wherein a single flow manager component handles network events from multiple layers,

10. The method of claim 8 wherein a flow manager component handles network events from only one layer.

11. The method of claim 1 wherein enforcement engines provide for security at various layers of a network protocol stack.

12. The method of claim 11 wherein layers of different enforcement engines are used for different connections.

13. A computer system architecture for distributing security policies, the architecture comprising:

    a policy server computer system for establishing security policies, the security policies having rules with rule types, and for providing the security policies to host computer systems for enforcement of the security policies at the host computer systems; and
    multiple host computer systems that include:

    enforcement engines for enforcing a type of security at the host computer system; and
    a firewall agent that receives the security policies from the policy server computer system, identifies the enforcement engine to which the rules apply based on rule type and enforcement engine type, and provides the rules to the identified enforcement engines for enforcement of the security policies.

IPR2023-01353 Page 00521

14. The computer system architecture of claim 13 wherein the enforcement engines are layered so that when one enforcement engine determines that the rules provided to it do not to apply to a network event, another enforcement engine determines whether the rules provided to it apply to the network event.

15. The computer system architecture of claim 14 wherein the layers include a protocol interception hook and an enforcement engine.

16. The computer system architecture of claim 13 wherein the firewall agent distributes rules to enforcement engines executing in user mode.

17. The computer system architecture of claim 13 wherein the firewall agent distributes rules to enforcement engines executing in kernel mode.

18. The computer system architecture of claim 13 wherein the enforcement engines are layered and including a flow manager component that handles network events at each layer and invokes an enforcement engine of the layer to enforce the rules of that layer on the network event.

19. The computer system architecture of claim 18 wherein the flow manager component handles network events for multiple layers.

20. The computer system architecture of claim 18 wherein the flow manager component handles network events for a single layer.

21. A host computer system for distributing rules of security policies to enforcement engines for enforcing the security policies of the host computer system, comprising:

multiple enforcement engines that implement different layers of firewall security enforcement at the host computer system by receiving and enforcing rules of security policies;
a component that receives at the host computer system security policies having rules, each rule having a rule type; and
a component that identifies enforcement engines to which a rule applies based on the rule type of the rule and that provides the rule to the identified security component;
wherein a mechanism is provided for distributing the rules to the layered enforcement engines of the host computer system.

22. The computer system of claim 21 wherein the security policies are received from a policy computer system that distributes the security policies for multiple host computer systems.

23. The computer system of claim 21 wherein the enforcement engine includes a behavior blocking security component.

24. The computer system of claim 21 wherein an enforcement engine includes subcomponents and wherein upon being provided with a rule, the enforcement engine provides the rule to a subcomponent.

25. The computer system of claim 21 wherein at least one enforcement engine executes in kernel mode and at least one enforcement engine executes in user mode.

26. The computer system of claim 21 wherein the component that provides the rules to the enforcement engines includes a user-mode subcomponent and a kernel-mode subcomponent, and the user-mode subcomponent distributes rules to the kernel-mode subcomponent for providing the rules to enforcement engine that execute in kernel mode.

27. The computer system of claim 21 including a flow manager component that intercepts network events at the various layers and invokes an enforcement engine associated with a layer to enforce the rules for that layer.

28. The computer system of claim 27 wherein a single flow manager component handles network events from multiple layers.

29. The computer system of claim 27 wherein a flow manager component handles network events from only one layer.

30. A computer-readable medium containing instructions for controlling a host computer system to distribute security policies to enforcement engines for enforcing the security policies, by a method comprising:

receiving and storing at the host computer system security policies having rules, each rule having a rule type;
under control of a firewall agent executing on the host computer system,
retrieving the stored security policies; and
for rules of a retrieved security policy,
identifying an enforcement engine to which a rule applies based on the rule type of the rule; and
providing the rule to the identified enforcement engine; and under control of the enforcement engines, enforcing the rules provided to the enforcement engines by the distribution component.

IPR2023-01353 Page 00522

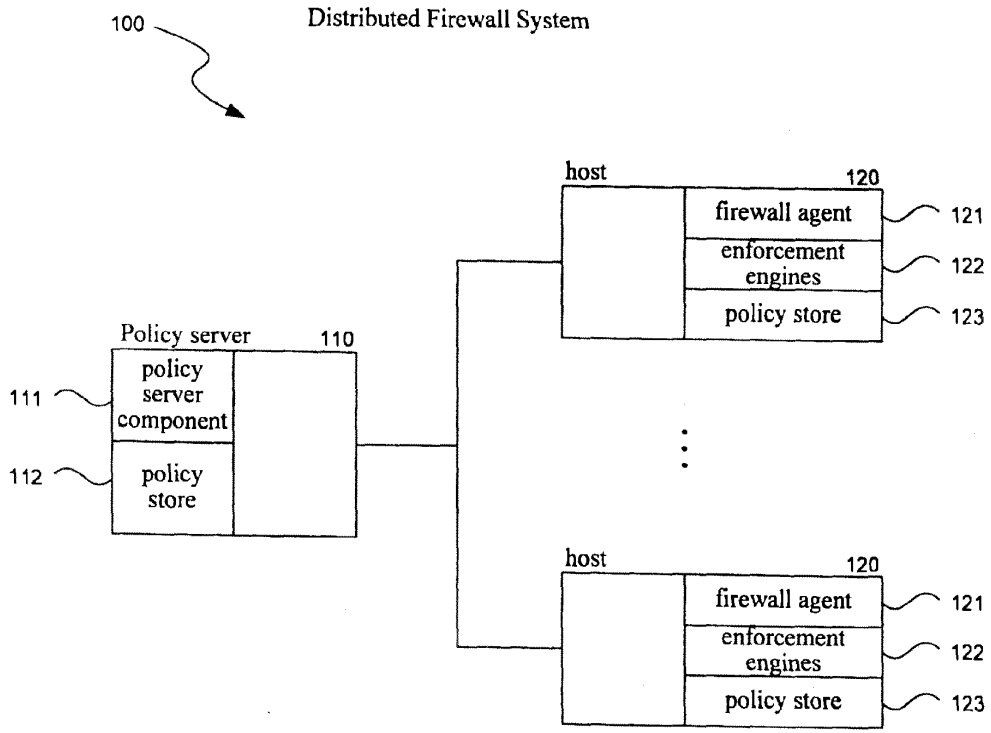

100
Distributed Firewall System
host
120
firewall agent
121
enforcement engines
122
policy store
123
Policy server
110
111
policy server component
112
policy store
host
120
firewall agent
121
enforcement engines
122
policy store
123

FIG. 1

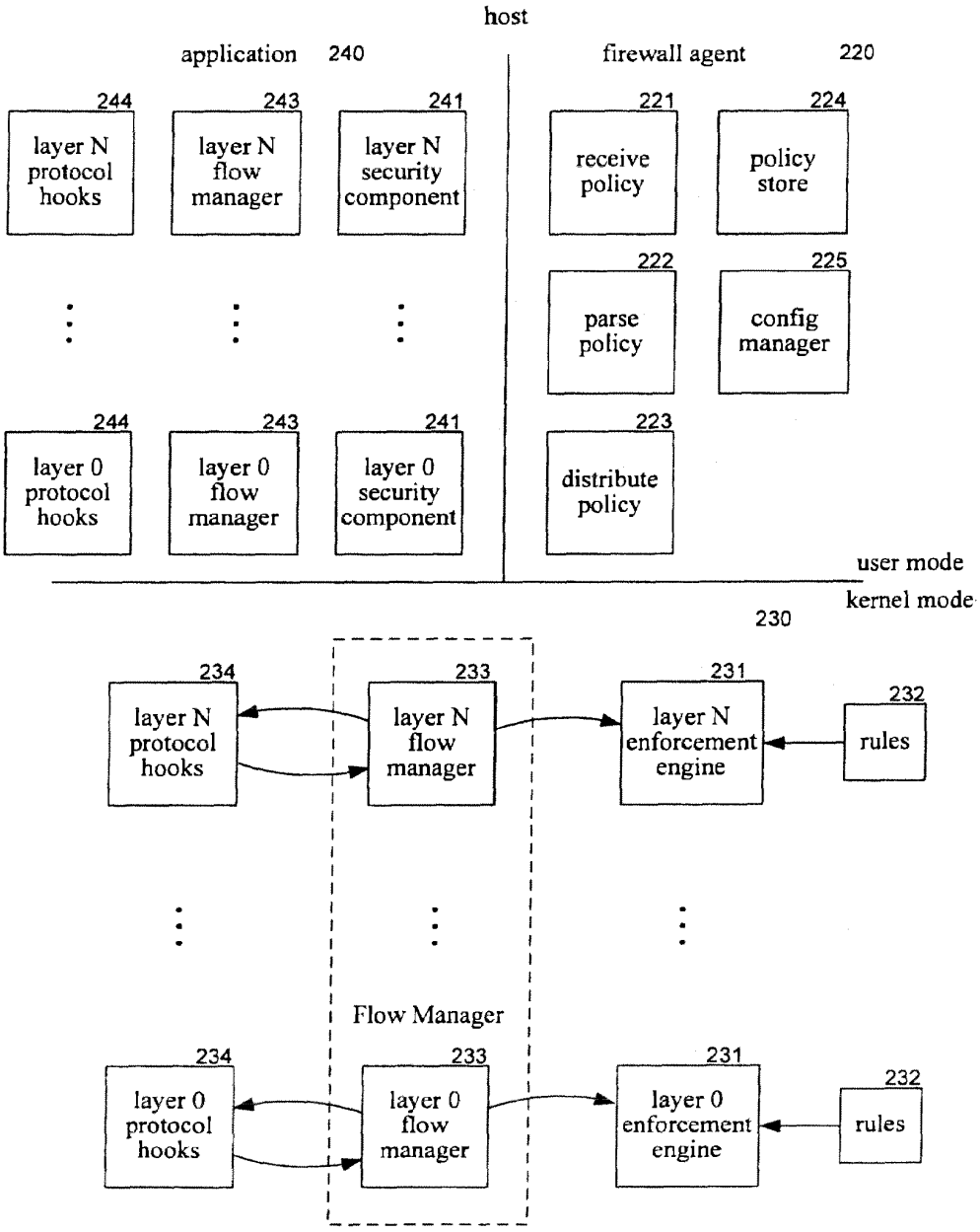


FIG. 2

IPR2023-01353 Page 00524

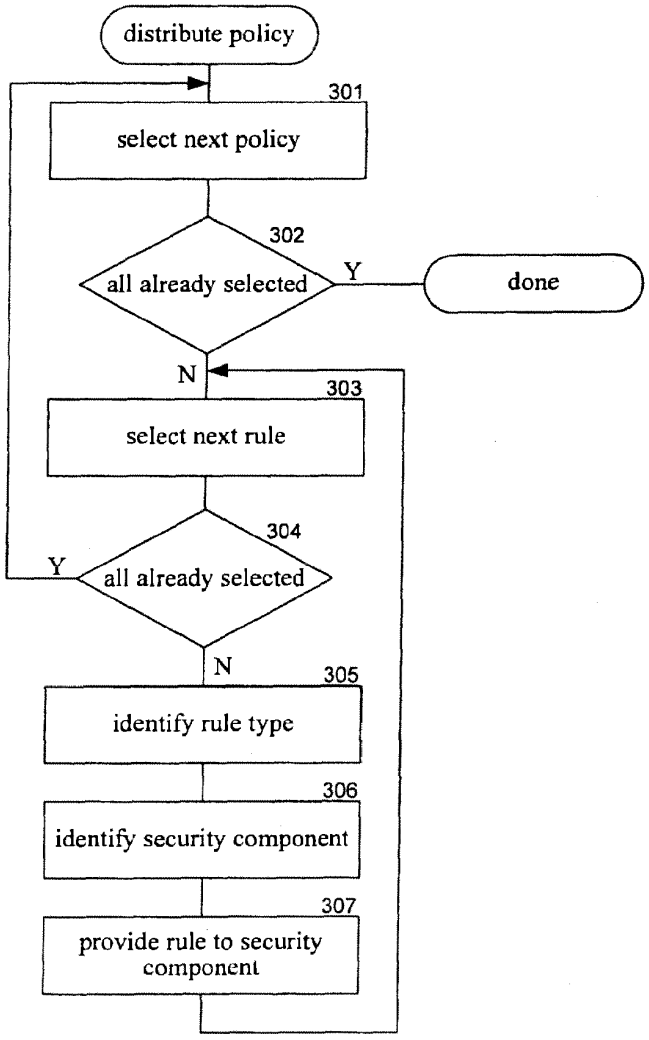


FIG. 3

IPR2023-01353 Page 00525

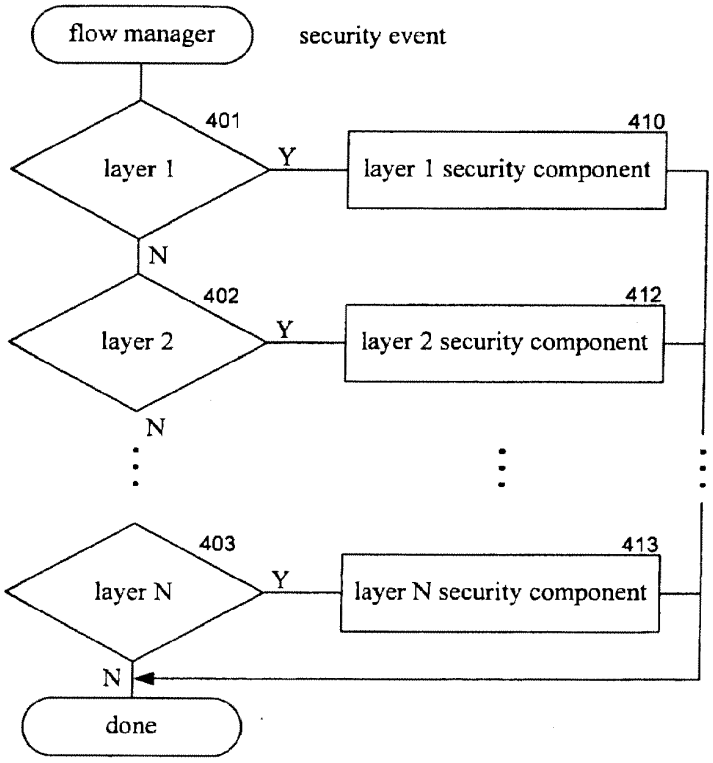


FIG. 4

IPR2023-01353 Page 00526

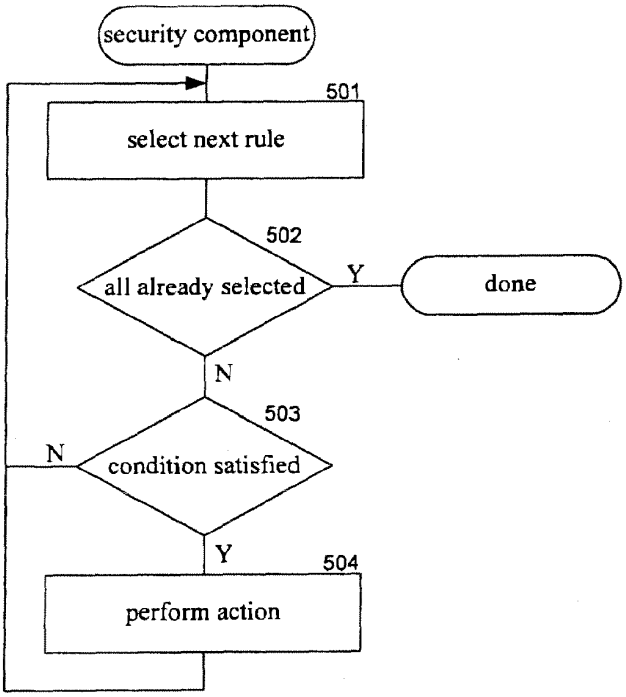


FIG. 5

IPR2023-01353 Page 00527

Espacenet

Bibliographic data: EP1313290 (A1) — 2003-05-21

A personal firewall with location dependent functionality

No documents available for this priority number.

Inventor(s): SYVAENNE TUOMO [FI] ± (SYVAENNE, TUOMO)

Applicant(s): STONESOFT CORP [FI] ± (STONESOFT CORPORATION)

Classification: **- international:** ***H04L29/06;*** (IPC1-7): H04L29/06
- cooperative: **H04L63/0227**; **H04L63/0263**; **H04L63/107**

Application number: EP20020102581 20021114

Priority number (s): US20010988356 20011119

Also published as: EP1313290 (B1) US2003097590 (A1) US7325248 (B2) DE60216218 (T2) AT346446 (T)

Abstract of EP1313290 (A1)

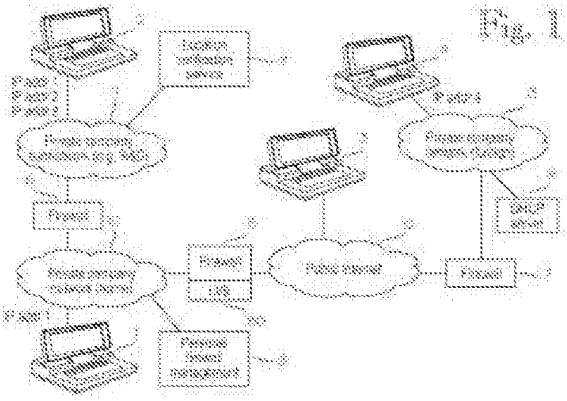


Fig. 1

A computer device is provided with a local security mechanism, a personal firewall, for protecting the computer device from attacks from a foreign network, in addition to or instead of a firewall in the internal network which protects the computer when connected to a home network. The personal firewall is provided with different sets of security rules for the home network and foreign networks. The personal firewall is arranged to detect its current location, i.e. determine to which network it is connected to at each particular moment. The personal firewall activates one of the given sets of security rules according to the detected current location of the computer device, i.e. the personal firewall automatically uses the security rules predefined for the network to which the computer device is connected at each particular moment.; Upon detecting a change in the location, the personal firewall immediately adapts to use security rules predefined for the new location.

Last updated: 11.12.2013 Worldwide Database 5.8.15.10; 93p

(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) EP 1 313 290 A1

(12) EUROPEAN PATENT APPLICATION

(43) Date of publication:
21.05.2003 Bulletin 2003/21

(51) Int Cl.[7]: **H04L 29/06**

(21) Application number: **02102581.2**

(22) Date of filing: **14.11.2002**

(84) Designated Contracting States:
AT BE BG CH CY CZ DE DK EE ES FI FR GB GR IE IT LI LU MC NL PT SE SK TR
Designated Extension States:
AL LT LV MK RO SI

(30) Priority: **19.11.2001 US 988356**

(71) Applicant: **Stonesoft Corporation**
00210 Helsinki (FI)

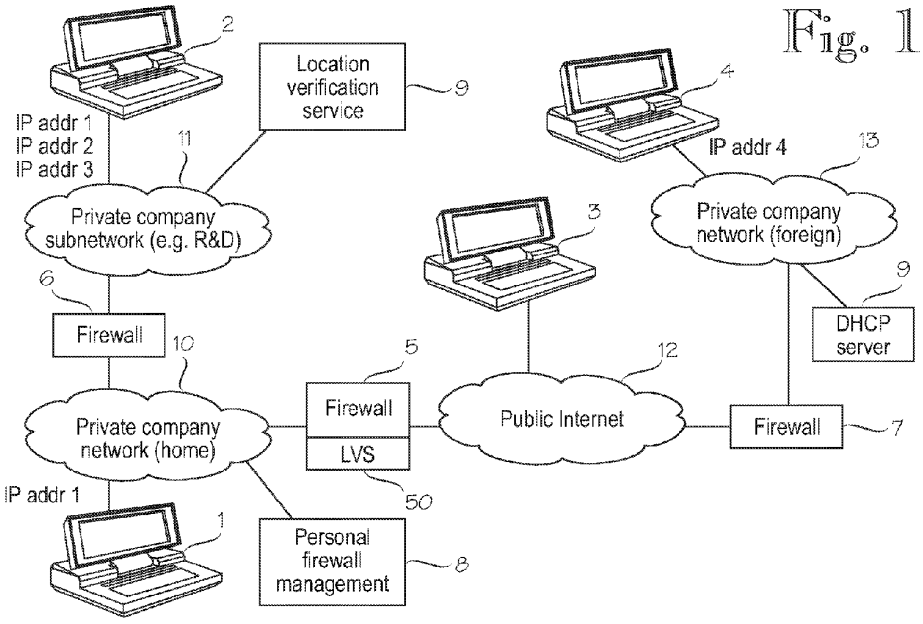
(72) Inventor: **Syvänne, Tuomo**
Fin-01450, Vantaa (FI)

(74) Representative: **Äkräs, Tapio Juhani**
Kolster Oy Ab,P.O. Box 148,
Iso Roobertinkatu 23
00121 Helsinki (FI)

(54) A personal firewall with location dependent functionality

(57) A computer device is provided with a local security mechanism, a personal firewall, for protecting the computer device from attacks from a foreign network, in addition to or instead of a firewall in the internal network which protects the computer when connected to a home network. The personal firewall is provided with different sets of security rules for the home network and foreign networks. The personal firewall is arranged to detect its current location, i.e. determine to which network it is connected to at each particular moment. The personal firewall activates one of the given sets of security rules according to the detected current location of the computer device, i.e. the personal firewall automatically uses the security rules predefined for the network to which the computer device is connected at each particular moment. Upon detecting a change in the location, the personal firewall immediately adapts to use security rules predefined for the new location.

Fig. 1



EP 1 313 290 A1

Printed by Jouve, 75001 PARIS (FR)

Description

Field of the Invention

[0001] The present invention relates to network security and, more particularly, to personal firewalls.

Background of the Invention

[0002] Traditionally, a firewall is considered as a set of components forming a gateway between two or more networks. Thus, a firewall has been a gateway which operates at the same time as a connector and a separator between the networks in a sense that the firewall keeps track of the traffic that passes through it from one network to another and restricts connections and packets that are defined as unwanted by the administrator of the system. Physically a firewall is a machine with appropriate software to do the tasks assigned to it. It can be a router, a personal computer (PC), or any other device that can be used for such purposes. Although firewalls are mostly used to connect Local Area Networks (LANs), i.e. internal networks, to the Internet and to protect against attackers or undesired traffic in general, they may also be used to separate and connect different segments of internal network for security purposes. The advantages of having a firewall are numerous. A firewall secures the network and can be used as a tool for monitoring the traffic especially from the outside to the inside of the network guarded by a firewall. Because all traffic intended for the internal network must pass through the firewall, most of the network security actions and policies can be concentrated in this particular point. This is of course a cost and administrative advantage.

[0003] Nowadays, laptop computers and other portable computer devices are widely used. While outside the internal network, the laptop cannot make benefit of the protection provided by the conventional "gateway-type" firewall. Therefore, approaches to improve security of a client located in a foreign network (a public network or an internal network of a foreign organisation) have been proposed. These approaches are based on protecting the laptop itself by means of a local security mechanism, called a personal firewall herein, installed in the laptop (in addition to or instead of a firewall in an internal network, which protects the computers connected to the internal network). The personal firewall may be implemented as software installed in the computer device, or as a separate electronic device connected to the computer device.

[0004] European patent application EP 0 952 715 discloses a firewall security device connected to an external communication port of a computer device. The incoming communications stream to the computer device from e.g. public networks is passed through the firewall security device. The firewall device applies standard security measures, thereby protecting the computer device.

[0005] There is a particular need for such protection by means of a personal firewall if the laptop is allowed to have a remote access, e.g. make a VPN (Virtual Private Network) connection to company network while being connected to a foreign network. In order to improve security of the VPN connections, one prior art solution is to enforce a protection level of a laptop, when a VPN tunnel to a company network is created. This means for example that, during a VPN connection, the IP address forwarding is not allowed, or that any connection attempts to the laptop are denied.

[0006] Clearly this is not enough, since the laptop must be protected as soon as it is connected to a foreign network, not only during a VPN connection. The laptops are often used by non-technical people, which increases the risk of overlooking security aspects. Laptops contain sensitive material, such as customer emails. If a laptop is unprotected, when connected to a foreign network, even for a short period of time, there is a risk of getting infected by a hostile application. Such application can be activated later, when the laptop is connected to an internal network and offer inside help for attacks.

[0007] Thus, there is a need to protect the laptop by means of a personal firewall always when the laptop is connected to a foreign network. However, when the laptop is connected to a company internal network, such personal firewall may unduly prevent some essential traffic. For example, the personal firewall should allow use of a laptop at home (internal) network and access to all services, such as disk-share. In a home network even non-IP protocols are sometimes used. Therefore, it is not feasible to have a personal firewall running at all times, at least not with the same configuration, since the protection needs in an internal network are different from those in a foreign network.

[0008] Some of the current solutions allow changing the set of rules used in the personal firewall, that is, they allow the user of the laptop to use different rule sets when connected to the internal network and when connected to a foreign network. However this is a manual operation. Since manual action is required, there is a high risk that operation is not done. Risk is even higher if the end user does not fully understand the need of a firewall.

Summary of the Invention

[0009] An object of the invention is to improve the security and flexibility of a personal firewall. The object is achieved by the invention disclosed in the attached independent claims. Preferred embodiments are disclosed in the attached dependent claims.

[0010] A computer device which can be connected to a home network (such as an internal network of a company or other organisation where the user is employed) and to a foreign network (such as a public network or an internal network of a foreign organisation) is provided with a local security mechanism, called a personal fire-

IPR2023-01353 Page 00531

wall herein, for protecting the computer device from attacks from a foreign network, in addition to or instead of a firewall in the internal network which protects the computer when connected to the internal network. The personal firewall is provided with different sets of security rules, at least one set of rules for the home network and at least one set of rules for foreign networks. In its simplest form, the set of rules for the home network contains no restrictions for the communication or use of service in the home network. The personal firewall is arranged to detect its current location, i.e. to determine the network to which it is connected at each particular moment. The personal firewall activates one of the given sets of security rules according to the detected current location of the computer device, i.e. the personal firewall automatically uses the security rules predefined for the network to which the computer device is connected at each particular moment. Upon detecting a change in the location, the personal firewall immediately adapts to use security rules predefined for the new location. A benefit of the invention is that the protection of a personal firewall is always enabled at the correct level, depending on the current location. On the other hand, when the computer device is located in the home network, a lower level of protection, or no protection at all, can be automatically provided by the personal firewall, so that the communication and services are not unduly restricted in the home network. Thus, the automated location-dependent management of different sets of rules offers optimal protection in different networks, while not unduly restricting operation in the home network.

[0011] The current location of the computer device is preferably determined on the basis of a currently used IP address of the computer device. This is based on the common practice that a computer device has a different IP address, either a fixed address or a dynamic address, in different networks. The IP address can thereby be utilized for identifying the current network and the location of the computer device.

[0012] However, there are situations where the IP address fails to indicate current location of the computer device. Therefore, in an embodiment of the invention, the current location determined on the basis of the current IP address of the computer device is verified by carrying out an additional location verification procedure with a predetermined network element. In a still further embodiment of the invention, availability of said predetermined network element related to the current IP address is checked. The predetermined network element is such that it responses only if the computer device is located in the network in which it is assumed to be on the basis of the current IP address. If the predetermined network element responses and identifies itself properly, the current location determined based on the current IP address is considered to be verified. Otherwise the computer device determines that the current IP address fails to indicate current location of the computer device. The additional verification process makes it even possible to automatically create a secured tunnel, such as a VPN tunnel to a home network even if the computer device uses the same IP address in the current location as in the internal (home) network. The present invention offers benefits even with stand alone personal firewalls wherein the security rules can be defined locally by the user, although the use of these rules is automated and location-dependent. However, more advantages are achieved when the basic invention is used with a central management of personal firewalls.

[0013] According to an aspect of the invention, security rules are defined, updated and distributed centrally by a centralized rule-based server. Especially the updating of the rules is challenging, because the rule updates must be applied as soon as possible, and therefore the process of updating rules in the personal firewalls must be automated. Updating of rules by push method from the centralized rule base server is not a sufficient option in this case. Use of DHCP (Dynamic Host Configuration Protocol), frequent travelling and the fact that at times the laptop may not be connected to any network makes it next to impossible for the centralized management to initiate contacts with the personal firewalls in the computer devices, because there is no way for the centralized management to know the IP address the computer device is using at a given moment. Therefore, according to an aspect of the invention, the personal firewall is configured to periodically query the availability of updated security rules from the centralized management. The queries should only be made, while the computer device is located in the home network, or optionally, when the computer device has a remote access (e.g. VPN connection) to the home network while being located in a foreign network. In other words, also the updating process is dependent on the current location of the computer device in a similar manner as the selection of the active rules, and similar methods can be utilized for determining the current location.

[0014] According to another aspect of the invention, log files containing information of a status and usage of resources of the computer device are handled in a centralized management location. This enables personnel aware of security aspects to verify whether there have been any attacks against the computer device or not. To that end, the personal firewall sends the log files to the central management, such as to a centralized log server, when the computer device is located in the home network. However, when the computer device is disconnected from the home network, the log files are collected and stored locally in the firewall. In order to enable central handling of the log files, the personal firewall transfers the collected log files to the central log server when such is available. This is performed automatically, whenever the computer device is located in, or optionally, connected to the home network. Again, the handling of the log files in the personal firewall is automated and location dependent in a way similar to the selection of active rules, and similar methods can be used for de-

IPR2023-01353 Page 00532

termining the current location of the computer device.

[0015] The present invention allows use of a computer device in a home (internal) network and access to all services, such as disc-share, and even use of non-IP protocols, which are often denied in foreign networks.

Brief description of the drawings

[0016] Preferred embodiments of the invention will now be described with reference to the attached drawings, in which

Figure 1 is a schematic block diagram of an exemplary network configuration where the present invention can be applied;
Figure 2 shows an exemplary protocol stack of a computer device containing a personal firewall according to the present invention,
Figure 3 illustrates exemplary selection rules and a security rule basis, and the association therebetween,
Figure 4 is a flow diagram illustrating a location-dependent rule base selection according to an embodiment of the invention;
Figures 5 and 6 are flow diagrams illustrating the location verification procedure according to an embodiment of the invention;
Figures 7 and 8 are flow diagrams illustrating the rule base updating according to an embodiment of the invention; and
Figure 9 is a flow diagram illustrating the handling of log files according to an embodiment of the invention.

Preferred embodiments of the invention

[0017] The present invention can be applied in personal firewalls in any computer device which can be moved and connected to different networks. Typically such devices are portable computer devices, such as laptop computers, PDAs, communicators, smart phones, intelligent telecommunication devices, etc. In the following illustrative embodiments of the invention, a laptop computer is used as an example of suitable computer devices.

[0018] Figure 1 shows a schematic block diagram of an exemplary network configuration. The configuration is shown only to facilitate the understanding and description of the present invention. The present invention is not intended to be restricted to any particular network configuration. Further, in order to improve clarity, only network elements which are somehow involved with the present invention are shown in Figure 1.

[0019] As illustrated in Figure 1, private local networks 10 and 13 are coupled to a public network, such as the Internet 12 via firewalls 5 and 7, respectively. Naturally, the coupling between the private networks and the public Internet 12 may include also routers and Internet service providers (ISPs not shown in Figure 1). As is well known in the art, private networks 10 and 13 may be, for example, company networks, such as local area networks (LANs) which connect users and resources, such as workstations, servers, printers and the like of the company. A private internal network may also consist of several sub-networks separated by internal firewalls. In the exemplary network configuration shown in Figure 1, the private company sub-network 11 is connected via a firewall 6 to the private local network 10. Such a sub-network 11 may be, for example, a dedicated network for a specific department of the organisation, such as the research and development (R&D) department which must have a restrictive access and higher protection level compared with other part of the company network. Sub-networks of the company, such as the local networks of organisation headquarters and branch offices may be interconnected by secure connections, such as virtual private network (VPN).

[0020] As already described above, the firewalls 5, 6 and 7 are gateways which operate at the same time as connectors and separators between the networks in a sense that the firewall keeps track of the traffic that passes through it from one network to another and restricts connections and packets that are defined as unwanted by the administrator of the system. Physically a firewall is a machine with appropriate software to perform the task assigned to it. It can be a router, a personal computer (PC), or whatever that can be used for such purposes.

[0021] However, the firewalls between the networks, or the implementations thereof, are not relevant to the present invention.

[0022] The present invention relates to protecting of the computer device, e.g. laptop itself, by means of a local security mechanism, called a personal firewall herein, installed on the laptop in addition to or instead of a firewall in a private network. The personal firewall may be implemented as software installed and run in the computer device, which is a preferred embodiment, or as a separate electronic device connected to the computer device. In Figure 1, the laptops 1, 2, 3 and 4 illustrate laptops provided with a personal firewall.

[0023] Figure 2 illustrates the basic principle of a personal firewall installed in a laptop. Physical and network layers 200 refer to all protocols and physical connections required for transferring protocol data units (PDUs) of the upper layers. The upper layers 200 include applications and any transmission protocols employed, such as Internet protocol (IP) transmission control protocol (TCP), NetPEUI, IPX, etc. Basically the personal firewall protection layer 201 operates in a manner analogue to a firewall between networks. More particularly, the personal firewall protection layer 201 operates at the same time as a connector and a separator between the underlying layers and the upper layer in a sense that the personal firewall keeps track of the traffic that passes through it from underlying layers to the upper layers, and

IPR2023-01353 Page 00533

vice versa, and restricts connections and packets that are defined as unwanted according to the security rules used. The personal firewall protection layer 201 is implemented or controlled by a personal firewall application 203 run in the laptop. In a preferred embodiment of the invention, the personal firewall application 203 carries out the location detection and the location-dependent functions described below, such as the selection of the active rule base according to the current location of the laptop. However, it should be appreciated that the present invention is not intended to be restricted to any specific practical implementation of the personal firewall.

[0024] In accordance with the principles of the present invention, the personal firewall has different sets of rules for the home network (such as the private company network 10) and foreign network, such as the public Internet 12, or the foreign private network 13, or a network of another department of the company. It is not relevant to the present invention what kind of security rules are applied, but some examples are given in Figure 3. For example, a rule base 301 for the foreign company network may list as allowed connections of protocols: hypertext transfer protocol (http), secured http (Https), domain name service (DNS), single message transfer protocol (SMTP) and a VPN connection with IPsec. In the preferred embodiment of the invention these rules are exclusive, in other words, other protocols and connections are denied and blocked by the personal firewall. For a default network, which may be the public Internet 12, the rule base 302 is similar to the rule base 301, except that the SMTP protocol is no longer allowed. For the home network 10, a rule base 300 is defined. The allowed protocols include, in addition to the http, https and the SMTP, also other transmission protocols, such as NetBEUI and IPX. The rule base 300 also allows a disc-share for predefined servers using NetBIOS. Other protocols and connections are denied. It is also possible that the rule base 300 allows all protocols and connections in the home network. Since the home network is protected by a company firewall, the use of a personal firewall in the home network may be regarded as unnecessary. However, the company firewall gives protection only against attacks from the outside of the home network, and the use of a personal firewall protection may be necessary for protecting against attacks from within the home network.

[0025] The different rule bases could be activated manually by a user. However, according to the basic principle of the present invention, the personal firewall automatically selects and activates the proper rule base according to the current location of the laptop.

[0026] Figure 4 is a flow diagram illustrating the selection of the rule base according to one embodiment of the present invention. The simplest way to determine the current location of the laptop is to do it on the basis of the currently used IP address only. This is possible in the cases where the laptop has a different IP address, either a fixed or a dynamic address, in different networks. As is well known in the art, a part of the IP address identifies the network, and can thus be used for detecting the current network of the laptop. The personal firewall may, for example, contain information on the IP address space of home network, and optionally, foreign networks, or a list of addresses available for the laptop in the home network.

[0027] When the current IP address of the laptop matches to a given address space or a list of addresses of the home network 10, for example, it can be assumed that the laptop is located in the home network 10 and the rule base 300 of the home network 10 is used. Thus, the current IP address is used as a selection rule for activating the rule base 300. However, there is some uncertainty in determining the location based on the current IP address only, and some approaches to overcome this problem are described with reference to further embodiments of the invention below.

[0028] Referring again to the generic flow diagram shown in Figure 4, the current IP address of the laptop is firstly determined in the step 401. The current IP address may be obtained simply by asking for it from the operating system of the laptop by means of using IP configuration routine. The current location of the laptop is monitored constantly, and therefore the personal firewall may be configured to periodically query the current IP address from the operating system. More preferably, the operating system of the laptop may be configured to inform any changes in the IP address to the personal firewall, and therefore a need for query the IP address time-to-time can be avoided. The step 401 may also include verification of the location determined based on the IP address by a verification procedure described below. In step 402, the personal firewall compares the current IP address with the current IP address stored in the personal firewall. If the IP address has not changed, the present active rule base can be maintained. However, if the IP address has changed, the personal firewall checks whether the new IP address matches to any IP address space or IP address belonging to one of the networks on the selection rule list in the personal firewall (step 403). If the new IP address does not belong to any of the networks on the selection rule list, the personal firewall considers the current network an unidentified network, and a default rule base 302 is selected (step 404). If the network cannot be identified and the default rule base must be used, it is normally assumed that the laptop is in a potentially hostile environment, most likely in the public Internet 12. Therefore, the default rule base is typically defined to provide the maximum protection needed. If the new IP address belongs to one of the networks defined on the selection rule list in step 403, it means that the network has been identified and a rule base linked to the identified network (or the corresponding selection rule) is selected and activated (step 405). In the simplest implementation, the selection rules include only the home network of the laptop and the cor-

IPR2023-01353 Page 00534

responding IP address space or list of addresses . If the current IP address belongs to the home network, the rule base 300 of the home network 10 is used. Otherwise the rule base 310 for foreign network or the default rule base 302 is used. In a more complicated implementation, there are selection rules (i.e. IP addresses and associated rule bases) also for at least one foreign network and/or different segments of the home network 10.

[0029] In the examples described above there are two or more rule bases which are enabled or disabled on the basis of the current location of the laptop. However, there are also alternative ways to implement different rule bases. One alternative is to provide only one rule base in which the rules are enabled and disabled in different combinations on the basis of the current location of the laptop.

[0030] As noted above, there are situations where the location (the current network) determined on the basis of the current IP address is uncertain, i.e. the IP address fails to indicate the current location of the laptop. If the IP address does not match the current network, use of the Internet protocol (IP) to attack against the laptop is not likely, and one may reason that in that case a personal firewall does not need to be used. However, there is still a possibility that there is an attack using other protocols, such NetBEUI or IPX. By detecting the situation where the IP address of the laptop is not an IP address of the current network, it is possible to block such protocols while in foreign networks. Further, NAT (network address translation) and private IP addresses are frequently used. This means that the same IP address is in use in several networks. In that case it is not enough to trust IP address information only when determining the location of the laptop. It is even possible that while being connected to a hostile network, the DHCP (dynamic host configuration protocol) gives familiar IP address to make it easier to attack the laptop. Basically, the DHCP enables individual computers on a network to connect to a DHCP server, such as the server 9 in Figure 1, and be assigned a dynamic IP address of the current network.

[0031] Thus, according to an aspect of the invention, in addition to the detection of location based on the current IP address described above, a further location verification procedure is carried out with a predetermined network element, which is preferably reachable only from the location to be verified. More generally, the network element is selected in such a way that it responses to the verification request only if the request originates from the location (i.e. the network) to be verified. Preferably, the specific network element is provided with a location verification service supporting the verification according to the invention. The verification procedure requires that the verification method be specified for the personal firewall, preferably at the same time as the different locations are specified. In other words, the methods to verify the location are specified for the personal firewall in the initial configuration, for example. It is also possible that the verification methods are updated or changed by means of the updating procedure described below, in a manner similar to other security rules.

[0032] A generic location verification procedure according to one embodiment of the invention is described with reference to Figures 5 and 6. Firstly, the current location is determined on the basis of the currently used IP address as described above (step 501). Next, the personal firewall selects a predetermined network element that should be available for verification from the determined current location (step 502). Then the personal firewall sends to the selected network element a request to send a response with some data proving the identity of the network element (step 503). Referring now to Figure 6, the verifying network element receives the verification request (step 601) from a personal firewall. Then, either always in response to the verification request or only if predetermined requirements are met, the network element sends the response with the required identity data to the personal firewall (step 602). Referring again to Figure 5, the personal firewall waits for a response (step 504), and if no response is received (preferably within a predetermined period of time), the location determined on the basis of the current IP address is rejected, and the location is determined to be unknown (step 505). In that case, a default location and an associated rule base, such as the rule base 302, can be used.

[0033] However, if the response is received from the network element in the step 504, the personal firewall verifies the identity of the network element on the basis of the received identity data, e.g. by comparing the received identity data with identity data stored in the personal firewall (step 506). If the verification of the identity is unsuccessful (step 507), the procedure proceeds to the step 505 described above. However, if the verification of the identity of the network element is successful, also the location of the laptop determined on the basis of the current IP address has been successfully verified and can be accepted.

[0034] Additionally, it is possible that one IP address is included in more than one selection rule in the personal firewall. In that case, if the verification of the laptop being located in a first network indicated by the current IP address fails, it is checked if the laptop is located in a second network indicated by the current IP address. There are various ways to implement the generic location verification procedure described above. The simplest way to implement the location verification service is to probe some known (known to the personal firewall) element. For example, it is possible to ask the MAC address of the known network element located in the home network and having a known IP address. The network element returns the MAC address in response, and if the AMC address is the one that it is assumed to be (e. g. matches with a MAC address stored in the personal firewall), it is verified that the laptop is located in the home network. If the MAC address is not the correct one, the laptop is determined to be outside the internal

IPR2023-01353 Page 00535

network.

[0035] One possible implementation is that the location verification service is implemented in such network element in an internal network which can be reached only from inside the internal network. For example, the firewall protecting the internal network, such as firewall 5 in Figure 1, may be configured to discard all connections from outside the internal network to this network element. If the personal firewall is able to reach this specific network element, it is verified that the personal firewall resides inside the internal network. Otherwise the personal firewall resides outside the internal network. There may be a plurality of such location verification services in a plurality of internal networks or sub-networks, and if the personal firewall cannot reach any of them, then it is assumed to be outside this plurality of internal networks or sub-networks.

[0036] The location verification service may be incorporated into the network firewall, such as the location verification service 50 in the firewall 5 in Figure 1. For example, verification requests only from the direction of the internal network (such as home network 10) may be responded by the location verification service 50. This implementation is rather straightforward, since the present firewalls are readily capable of detecting from which direction, that is, from which interface, a data packet is coming from.

[0037] In any case it is preferred that the personal firewall communicates with the location verification service by using some cryptographically strong method, such as public key encryption. For example SSL can be used. The certainty of the location verification can be further improved by setting the TTL (time-to-live) field in the location verification request to a relatively low value, so that the request is capable of reaching only a nearby location verification service. The TTL value is decremented each time the verification request passes through a router connecting different networks or network segments. If the TTL value is set to, for example, a zero value the verification request is not able to pass through a router to a different network or a network segment.

[0038] The use of additional location verification makes it even possible to automatically create a VPN tunnel to the home network even if the current location (a foreign network) is using the same IP address as in the internal network.

[0039] All the embodiments described above are effective both in standalone computers and in centrally managed computers. The central management of personal firewalls enables a uniform protection level in all computer devices using the private network. One feature of the central management is that preferably all of the personal firewalls have essentially similar security rules. It should also be possible to update these. It is preferable that rule updates are applied in the personal firewalls as soon as possible after they have been made in the central management. Because it is not sufficient to rely on the manual updating by the user, the process of updating the rules must be automated. However, distribution of the updated rules by a push transmission from the central management is not a sufficient option in a case where the personal firewalls can move from one network to another. Therefore, according to an aspect of the invention, the personal firewalls are arranged to periodically query the availability of updated rules from the central management. An updating procedure according to one embodiment of the invention is described with reference to Figures 7 and 8.

[0040] Firstly, a personal firewall measures a predetermined updating period, which can be any period of time, preferably one day or a few days (step 701). In step 702, the personal firewall checks whether the updating period has expired, and if not, the procedure returns to the step 701. When the updating period has expired, the personal firewall checks whether the current location of the laptop is in the home network (step 703) or in another subnetwork of the same company. The location determination is preferably based on the methods described above. If the current location is in the home network, the process proceeds directly to the step 705. However, if the current location is not in the home network, the personal firewall waits for the laptop to return to or establish a connection (e.g. VPN) to the home network (step 704), before proceeding to the step 705. In the step 705, the personal firewall sends a rule update query to the central management, such as the personal firewall management server 8 in Figure 1. Referring now to Figure 8, the personal firewall management 8 receives the rule update query from the personal firewall (step 801) and sends updated rules, if there are any, to the personal firewall (step 802). Referring again to Figure 7, the personal firewall checks whether the reply received from the personal firewall management 8 contains rule updatings (step 706), and if not, the process returns to the step 701 to measure the next updating period. However, if rule updatings have been received, the personal firewall updates the relevant rule bases stored in the laptop (step 707).

[0041] It is also preferable that the logs relating to the communication transactions of the laptop are handled in a central location. Since the laptops are frequently disconnected from the home network, logs must be collected locally. In order to enable central handling, the logs must be transferred to a central log server, such as the personal firewall management 8, when such is available. This should take place automatically when the computer device provided with a personal firewall is connected to the home network. Figure 9 illustrates log handling according to one embodiment of the invention.

[0042] Firstly, the personal firewall creates a log file each time the laptop is involved with a communication transaction, such as an Internet session (step 91). Then the personal firewall determines the current location of the laptop, preferably based on the location determining methods described above (steps 92 and 93). If the lo-

IPR2023-01353 Page 00536

cation of the laptop is in the home network or in another subnetwork of the same company, the personal firewall sends the log file to the central log server 8 immediately (step 94). However, if the current location of the laptop is not in the home network, the process proceeds to the step 95, where the log file is stored locally. Similarly, a number of log files is collected locally while the laptop is disconnected from the home network. When the personal firewall next time detects that the computer device is relocated in the home network, it sends the collected log files to the central log server 8. Optionally, the personal firewall may also send the collected log files to the personal firewall management 8 when the laptop has established a (e.g. VPN) connection to the home network.

[0043] It is apparent for those skilled in the art that the illustrative embodiments described are only examples and that various modifications can be made within the scope and spirit of the invention as defined in the appended claims.

Claims

1. A method of controlling a personal firewall in a client computer, said method comprising
   providing said personal firewall with at least one set of security rules to be used when said client computer is connected to a home network of said client computer, and at least one set of security rules to be used when said client computer is connected to foreign networks,
   monitoring the current location of said client computer based on an Internet Protocol (IP) address currently used by said client computer, and
   automatically selecting one of said sets of security rules by said personal firewall according to said current location of said client computer.

2. A method according to claim 1, wherein
   said step of providing comprises providing said client computer with said set of security rules in form of at least two rule bases,
   said step of selecting comprises enabling one of said rule bases at a time according to the current location of said client computer.

3. A method according to claim 1, wherein
   said step of providing comprises providing said client computer with said set of security rules in form of one rule base, and
   said step of selecting comprises enabling and disabling rules in said one rule base in different combinations according to the current location of said client computer.

4. A method according to claim 1, wherein said step of monitoring comprises
   storing in said client computer an IP address space of said home network,
   comparing the current IP address of said client computer with said IP address space, and
   if the current IP address of said client computer matches said IP address space, determining said personal firewall to be located in said home network.

5. A method according to any one of claims 1, 2 or 3, wherein said step of monitoring comprises
   storing in said client computer a list of IP addresses of said home network,
   comparing the current IP address of said client computer with said list of IP addresses, and
   if the current IP address of said client computer matches one of said addresses on said list, determining said client computer to be located in said home network.

6. A method according to any one of claims 1, 2 or 3, comprising
   verifying the current location determined on the basis of the current IP address of said client computer by carrying out a location verification procedure with a predetermined network element.

7. A method according to claim 6, wherein said step of verifying comprises
   checking availability of said predetermined network element related to the current IP address, said predetermined network element responding only if said client computer is located in the network in which it is assumed to be on the basis of the current IP address,
   verifying the current location determined based on said current IP address, if said predetermined network element responses with a specific identity data.

8. A method according to claim 7, wherein said specific identity data is a Media Access Control (MAC) address of said predetermined network element.

9. A method of managing a personal firewall in a client computer, comprising
   storing in said personal firewall at least one set of security rules to be used when said client computer is connected to a home network of said client computer, and at least one set of security rules to be used when said client computer is connected to foreign networks,
   storing updated sets of security rules, if any, in a centralized unit in said home network of said client computer,
   configuring said personal firewall to periodically query the availability of said updated sets of security rules from said centralized rule base server

IPR2023-01353 Page 00537

when being located in said home network, or when having a remote access to said home network while being located in a foreign network, and

loading said updated sets of security rules from said centralized rule base server to said personal firewall in response to said query, if such updated sets of security rules are available.

10. A method according to claim 9, comprising

monitoring the current location of said client computer based on an Internet Protocol (IP) address currently used by said client computer, and

automatically activating said periodical query, when the current location of said client computer is in said home network.

11. A method according to claim 9, comprising

monitoring the current location of said client computer based on an Internet Protocol (IP) address currently used by said client computer,

sending log files to a centralized log server from said personal firewall, when the current location of said client computer is in said home network, said log files containing information on communication transactions in said client computer,

collecting log files locally at said personal firewall, when the current location of said client computer is not in said home network, and

transferring said locally collected log files from said personal firewall to said centralized log server, when said client computer is connected to said home network.

12. A computer terminal, comprising

a personal firewall provided with at least one set of security rules to be used when said computer terminal is connected to a home network of said computer terminal, and at least one set of security rules to be used when said computer terminal is connected to foreign networks,

said personal firewall having a mechanism monitoring the current location of said computer terminal based on an Internet Protocol (IP) address currently used by said computer terminal, and

said personal firewall having a mechanism automatically selecting one of said sets of security rules by said personal firewall according to said current location of said computer terminal.

13. A computer terminal according to claim 12, further comprising

a mechanism verifying the current location determined on the basis of the current IP address of said computer terminal by carrying out a location verification procedure with a predetermined network element.

14. A computer-readable medium, containing a computer software which, when executed in a computer device, causes the computer device to provide a personal firewall routine comprising

storing at least one set of security rules to be used when said computer device is connected to a home network of said computer device, and at least one set of security rules to be used when said computer device is connected to foreign networks,

monitoring the current location of said computer device based on an Internet Protocol (IP) address currently used by said computer device, and

automatically selecting one of said sets of security rules by said personal firewall according to said current location of said computer device.

15. A computer-readable medium according to claim 14, said personal firewall routine further comprising

verifying the current location determined on the basis of the current IP address of said client computer by carrying out a location verification procedure with a predetermined network element

16. A computer device, comprising a personal firewall routine configured to store security rules for a home network of said computer device, and for foreign networks,

periodically query the availability of updated security rules from a centralized rule base server in said home network of said computer device when said computer device is located in said home network, and

download said updated security rules from said centralized rule base server, if such updated security rules are available in said centralized rule base server.

17. A computer device according to claim 16, said personal firewall routine being further configured to

activate said periodical query also when said computer device has a remote access to said home network while being located in a foreign network.

18. A computer device, comprising a personal firewall routine configured to

send log files to a centralized log server, when a current location of said computer device is in a home network of said computer device, said log files containing information on communication transactions in said computer device,

collect log files locally in said computer device, when the current location of said client device is not in said home network,

transfer said locally collected log files to said centralized log server, when said computer device is reconnected to said home network.

19. A computer-readable medium, containing computer software which, when executed in a computer de-

IPR2023-01353 Page 00538

vice, causes the computer device to provide a personal firewall routine comprising

storing security rules for a home network of said computer device, and for foreign networks,

periodically querying the availability of updated security rules from a centralized rule base server in said home network of said computer device when said computer device is located in said home network, and

downloading said updated security rules from said centralized rule base server, if such updated security rules are available in said centralized rule base server.

20. A computer-readable medium, containing computer software which, when executed in a computer device, causes the computer device to provide a personal firewall routine comprising

sending log files to a centralized log server, when a current location of said computer device is in a home network of said computer device, said log files containing information on communication transactions in said computer device,

collecting log files locally in said computer device, when the current location of said client device is not in said home network,

transferring said locally collected log files to said centralized log server, when said computer device is reconnected to said home network.

21. A computer program comprising program code means for performing all the steps of any one of claims 1 to 11 when the program is run on a computer.

IPR2023-01353 Page 00539

Fig. 1

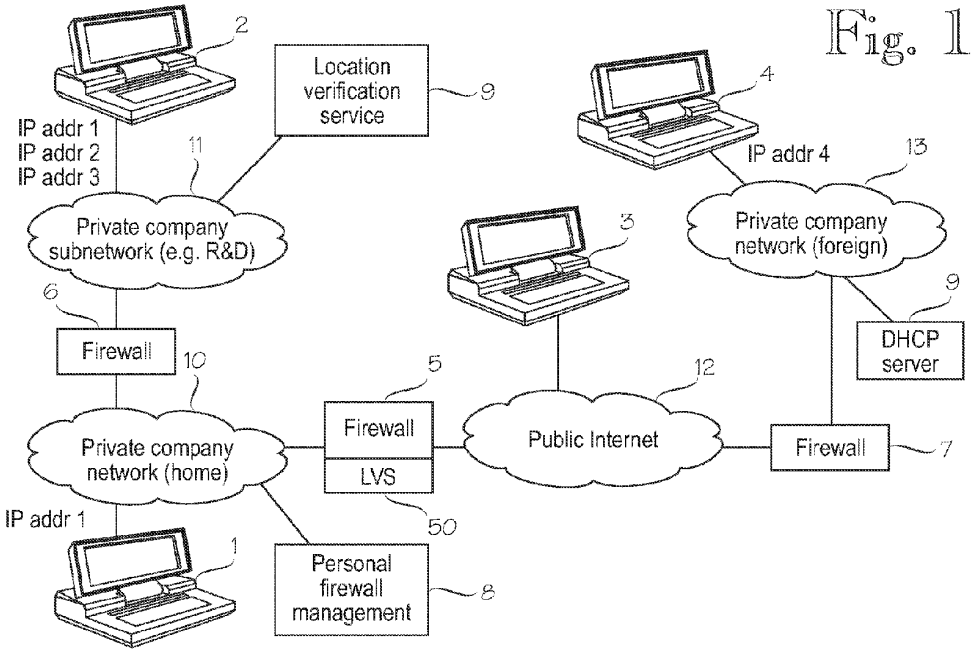


Fig. 2

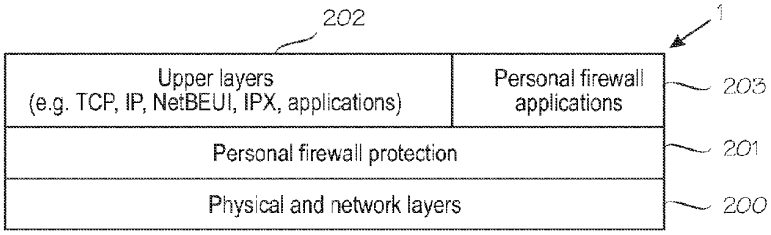
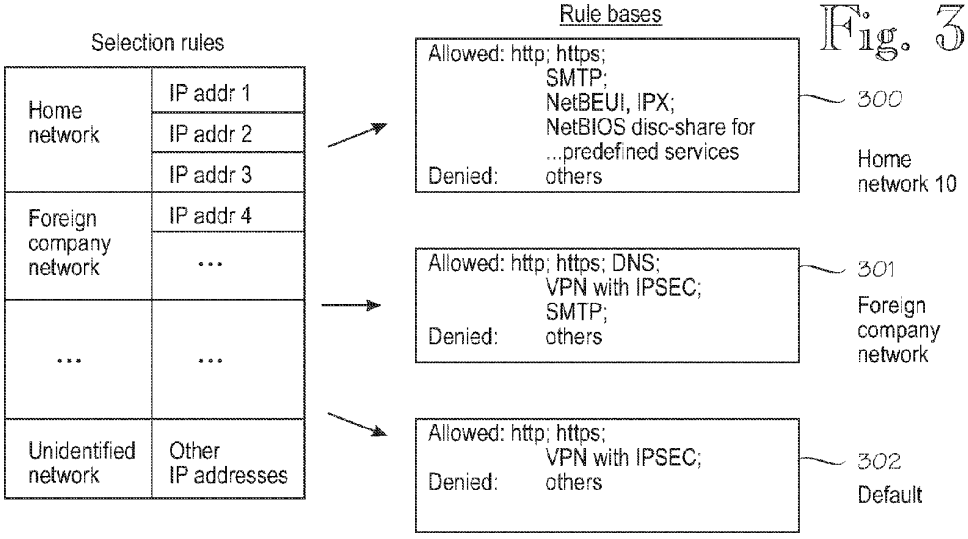


Fig. 3



IPR2023-01353 Page 00540

Fig. 4

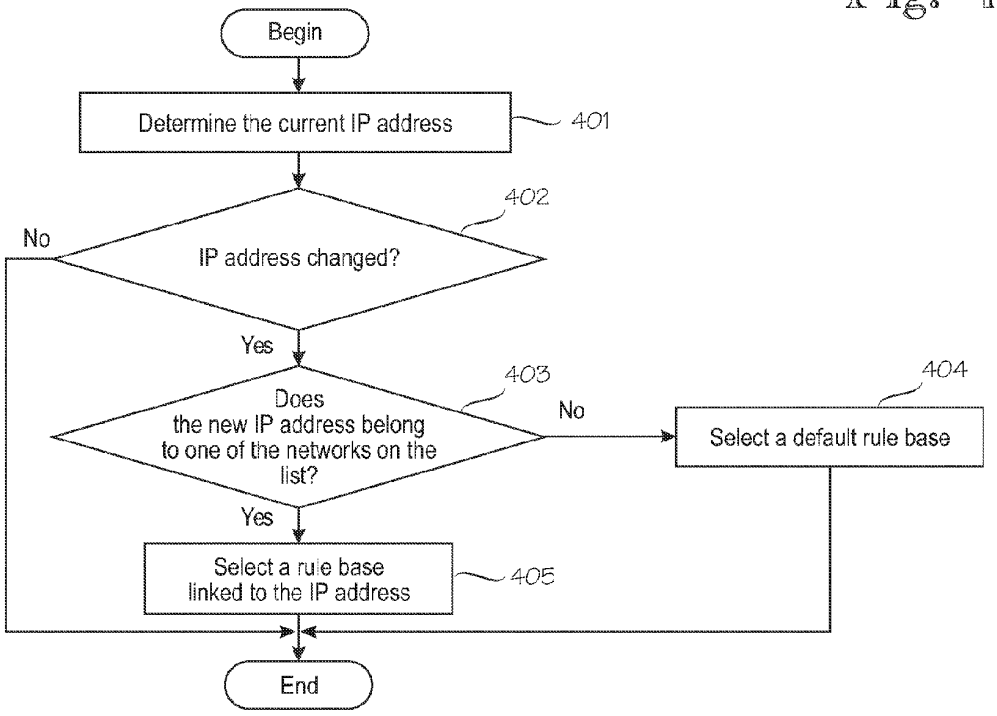
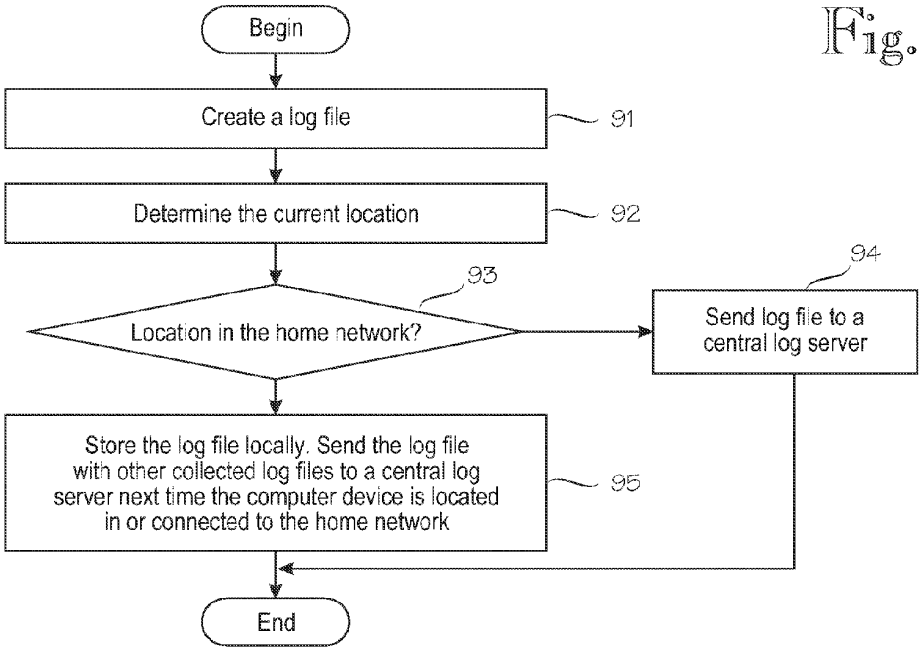


Fig. 9



IPR2023-01353 Page 00541

Fig. 5

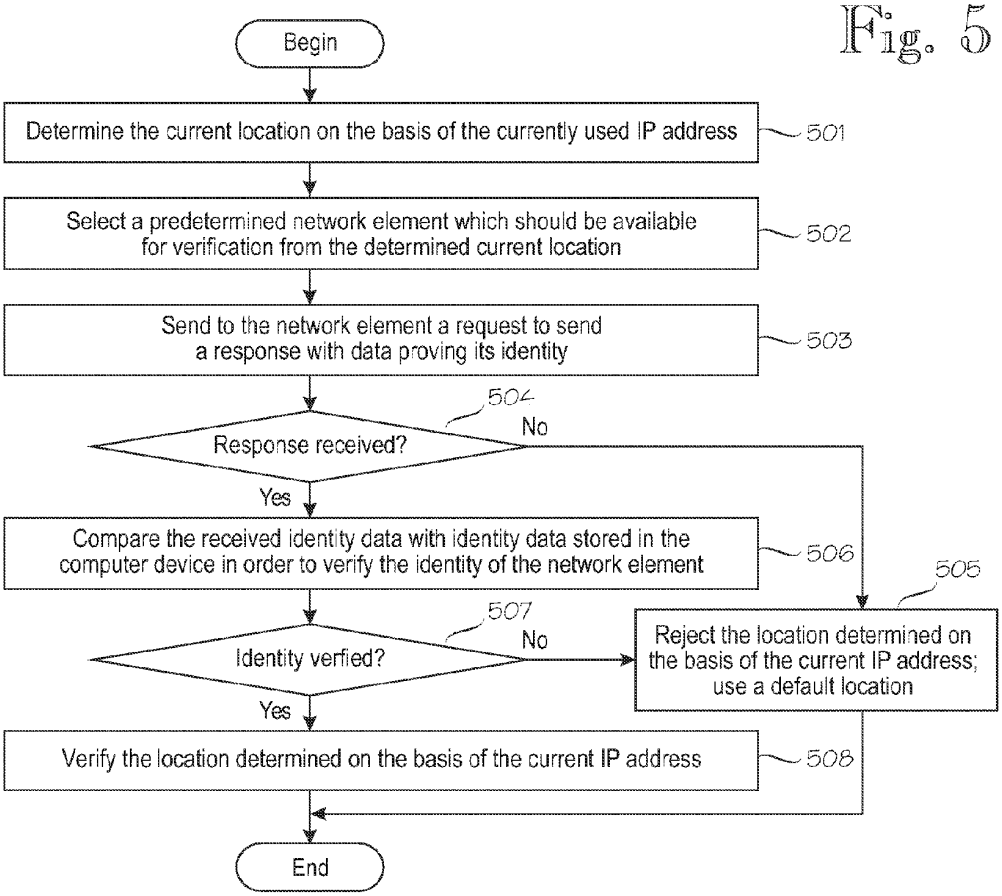


Fig. 6

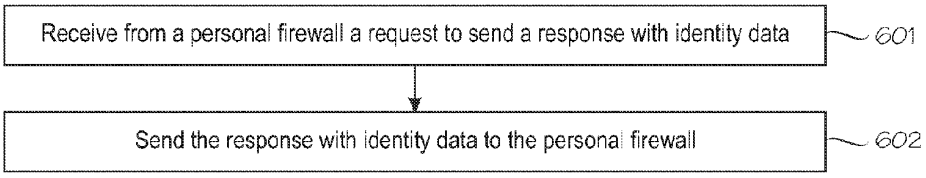
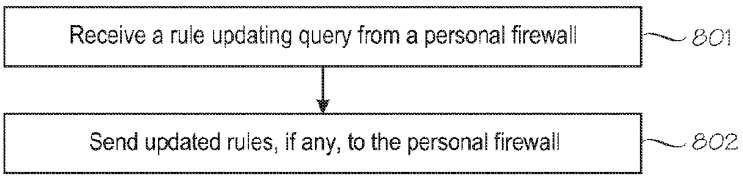
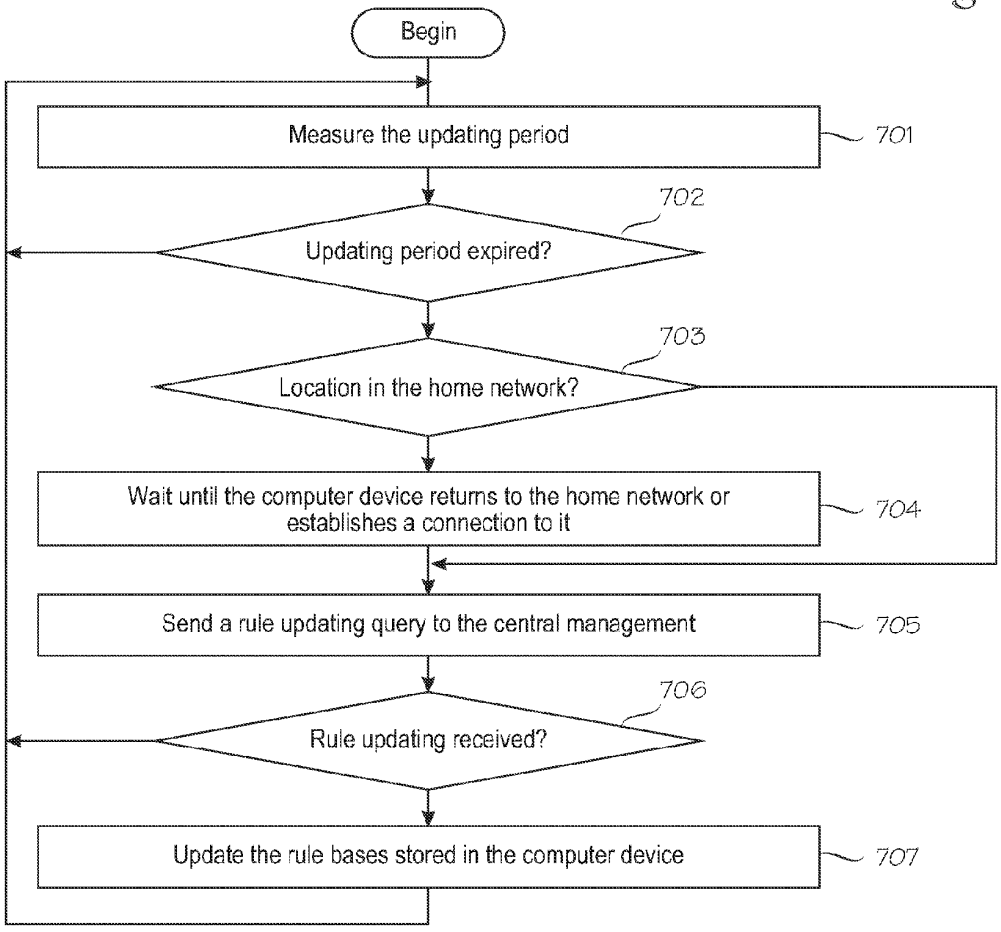


Fig. 8



IPR2023-01353 Page 00542

Fig. 7



IPR2023-01353 Page 00543

European Patent Office

EUROPEAN SEARCH REPORT

Application Number
EP 02 10 2581

DOCUMENTS CONSIDERED TO BE RELEVANT

Category	Citation of document with indication, where appropriate, of relevant passages	Relevant to claim	CLASSIFICATION OF THE APPLICATION (Int.Cl.7)
A	EP 0 854 621 A (AT & T CORP) 22 July 1998 (1998-07-22) * page 4, line 28 - line 47 * ---	1-21	H04L29/06
A	US 6 158 008 A (OWENS LESLIE DALE ET AL) 5 December 2000 (2000-12-05) * claim 1 * -----	1-21	

TECHNICAL FIELDS SEARCHED (Int.Cl.7)

H04L

The present search report has been drawn up for all claims

Place of search	Date of completion of the search	Examiner
THE HAGUE	14 March 2003	Veen, G

CATEGORY OF CITED DOCUMENTS

X : particularly relevant if taken alone
Y : particularly relevant if combined with another document of the same category
A : technological background
O : non-written disclosure
P : intermediate document

T : theory or principle underlying the invention
E : earlier patent document, but published on, or after the filing date
D : document cited in the application
L : document cited for other reasons

& : member of the same patent family, corresponding document

EPO FORM 1503 03.82 (P04C01)

IPR2023-01353 Page 00544

ANNEX TO THE EUROPEAN SEARCH REPORT ON EUROPEAN PATENT APPLICATION NO. EP 02 10 2581

This annex lists the patent family members relating to the patent documents cited in the above-mentioned European search report. The members are as contained in the European Patent Office EDP file on
The European Patent Office is in no way liable for these particulars which are merely given for the purpose of information.

14-03-2003

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
EP 0854621 A	22-07-1998	US 6233686 B1	15-05-2001
		CA 2226814 A1	17-07-1998
		EP 0854621 A1	22-07-1998
		JP 10229418 A	25-08-1998
US 6158008 A	05-12-2000	WO 9921339 A1	29-04-1999

EPO FORM P0459

For more details about this annex : see Official Journal of the European Patent Office, No. 12/82

IPR2023-01353 Page 00545

Espacenet

Bibliographic data: WO2005046145 (A1) — 2005-05-19

ADAPTABLE NETWORK BRIDGE

No documents available for this priority number.

Inventor(s): HALEN JOACIM ± (HALEN, JOACIM)

Applicant(s): ERICSSON TELEFON AB L M [SE] ± (TELEFONAKTIEBOLAGET LM ERICSSON (PUBL))

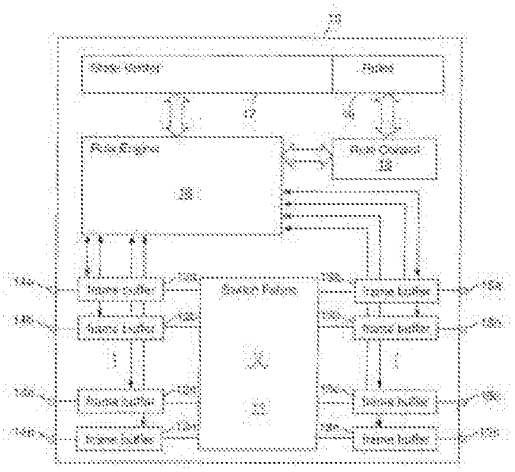
Classification: **- international:*H04L12/24; H04L12/46; H04L12/56;***
(IPC1-7): H04L12/46; H04L12/56
- cooperative: **H04L12/4625**; **H04L41/0893**; **H04L41/0896**; **H04L45/04**; **H04L45/56**; **H04L45/66**; **H04L49/351**

Application number: WO2004SE01611 20041105

Priority number (s): US20030517737P 20031106 ; US20040977724 20041029

Also published as: US2005111434 (A1) US7672318 (B2) JP2007534219 (A) JP4323523 (B2) EP1683313 (A1) more

Abstract of WO2005046145 (A1)



An adaptable network bridge such as an Ethernet bridge (10) includes a Rule Engine (16) that processes data frames according to a set of switching rules. Each switching rule identifies a type of frame to which the rule applies, and how the type of frame is to be processed. The bridge also includes means (18) for dynamically adding, subtracting, or modifying the switching rules in response to changes in the bridge's state information, or changes in context information. If the bridge includes a plurality of input/output ports (14, 15), the Rule Engine (16) may include a plurality of sets of switching rules, with each set of switching rules being applicable to a different port in the bridge. In this way, different network providers can share the same equipment, but have different network behavior.

Last updated: 11.12.2013 Worldwide Database 5.8.15.10; 93p

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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
19 May 2005 (19.05.2005)

PCT

(10) International Publication Number
WO 2005/046145 A1

(51) International Patent Classification[7]: H04L 12/56, 12/46

(21) International Application Number:
PCT/SE2004/001611

(22) International Filing Date:
5 November 2004 (05.11.2004)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
60/517,737 6 November 2003 (06.11.2003) US
10/977,724 29 October 2004 (29.10.2004) US

(71) Applicant: TELEFONAKTIEBOLAGET LM ERICSSON (publ) [SE/SE]; S-164 83 Stockholm (SE).

(72) Inventor: HALEN, Joacim; Anhaltsvägen 8, S-191 40 Sollentuna (SE).

(74) Agent: KARIN BOESTAD et al.; Ericsson AB, Patent Unit Core Networks Kista, S-164 80 Stockholm (SE).

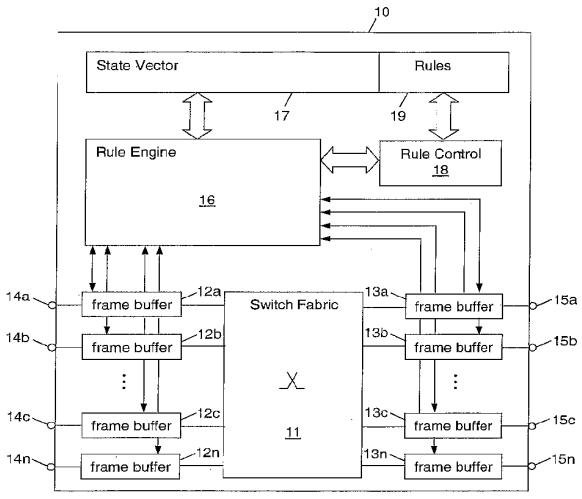
(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BW, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR,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

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HU, IE, IS, IT, LU, MC, NL, PL, PT, RO, SE, SI, SK, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Published:
— *with international search report*
— *before the expiration of the time limit for amending the claims and to be republished in the event of receipt of amendments*

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.

(54) Title: ADAPTABLE NETWORK BRIDGE



(57) Abstract: An adaptable network bridge such as an Ethernet bridge (10) includes a Rule Engine (16) that processes data frames according to a set of switching rules. Each switching rule identifies a type of frame to which the rule applies, and how the type of frame is to be processed. The bridge also includes means (18) for dynamically adding, subtracting, or modifying the switching rules in response to changes in the bridge's state information, or changes in context information. If the bridge includes a plurality of input/output ports (14, 15), the Rule Engine (16) may include a plurality of sets of switching rules, with each set of switching rules being applicable to a different port in the bridge. In this way, different network providers can share the same equipment, but have different network behavior.

WO 2005/046145 A1

ADAPTABLE NETWORK BRIDGE

CROSS-REFERENCE TO RELATED APPLICATIONS

This nonprovisional application claims priority under 35 U.S.C. § 119(e) on U.S. Provisional Application No. 60/517,737 entitled, *Programmable Bridge*, filed November 6, 2003, the disclosure of which is incorporated herein in its entirety.

BACKGROUND OF THE INVENTION

Technical Field of the Invention

The present invention relates generally to digital communication systems. More particularly, and not by way of limitation, the invention is directed to an adaptable network bridge having a Rule Engine that simplifies adapting the bridge to dynamically changing conditions.

Description of Related Art

Ethernet is a layer-2 packet-based transmission protocol that is primarily used in local area networks (LANs). Ethernet is the common name for the Institute of Electrical and Electronics Engineers (IEEE) 802.3 industry specification. Data is transmitted in Ethernet frames. Each frame includes a preamble with 64 bits utilized for synchronization. A Start of Frame Delimiter (SFD), a destination address, a source address, and a length/type identifier follow the preamble. Media Access Control (MAC) client data, together with a Packet Assembler/Disassembler (PAD) may vary in length from 46 to 1500 bytes (octets). A Frame Check Sequence (FCS) adds four more octets. Higher-level protocols reside in the content part of the frame. The frame size is counted from the destination address to the FCS, inclusive, and thus may vary between 64 and 1518 octets, not including a Virtual Local Area Network (VLAN) tag, which adds 4 octets.

Today most packet-based networks are built on Ethernet technology, as it is a cheap and reliable technology with good performance. Ethernet has been used in office networks for a long time and is now used to build Wide Area Networks (WANs) as well as small networks in homes. The communications

industry is now expanding the use of Ethernet to what is called "Public Ethernet" where millions of users are connected in the same network. Unfortunately, Ethernet does not scale to such large networks due to flooding of broadcasted messages. Some solutions have been proposed, but these solutions require modifications to the existing standards if they are to be implemented on standard equipment.

Ethernet bridging is a way to connect networks together to form a larger network. The standard for bridging is ANSI/IEEE 802.1d, which is hereby incorporated by reference herein. A bridge is a way to connect two separate network segments together in a protocol-independent way. Packets are forwarded based on Ethernet address, rather than IP address. Since forwarding is done at Layer 2, all protocols can go transparently through a bridge.

There are basically two approaches to adding new functionality to Ethernet bridges (and other Ethernet equipment as well): standardization thru IEEE, or proprietary solutions, which make the equipment non-standard. Adding new functionality and behavior to IEEE 802.1d MAC bridges, better known as Ethernet switches (or other Ethernet equipment), is a difficult and time-consuming process without any guarantees of success. Each new function must be proposed to and scrutinized by an IEEE standardization group (an existing working group or a new group has to be accepted and established). If one is lucky, the proposed function is accepted and added to the standard. However, since each new function requires a change in the standard, there is reluctance to add new functions. If the proposal is rejected, one is limited to a proprietary solution on non-standard equipment. The members of the standardization groups come from different vendors and other organizations, and although they are supposed to act in the interest of IEEE and the entire community, they may also act on agendas from their own organization/company that might be in conflict with the proposed functionality, thereby turning it down. It should also be noted that if a new function is added to the standard, it is then up to the different vendors to add the standardized functionality. A feature may be standardized but not made mandatory.

The proprietary approach also has its drawbacks. For example,

programmable network processors have been available for some time, and bridges may be purchased off the shelf containing programmable network processors. Proprietary solutions can be implemented on these. However, these network processors are vendor specific, and the solution will only work on that network processor or processor family. The solution will not be portable, and it will not be possible to run it on other vendors' equipment. Additionally, programming a network processor is done at a very low level, basically in assembly language, and the programming model is hard, requiring special skill. It should also be noted that a proprietary solution will make the equipment non-standard, and customers are more reluctant to buy non-standard equipment. There might also be inter-operational problems with other equipment.

SUMMARY OF THE INVENTION

The present invention is an adaptable network bridge having a Rule Engine that simplifies the adaptation of the bridge to dynamically changing conditions related, for example, to network conditions and user context. Simplicity is achieved by specifying how data frames should be processed and switched in terms of switching rules. Flexibility is achieved by enabling dynamic loading and updating of rules in the bridge via a protocol. In addition, one bridge may generate or modify rules for loading at another, neighboring, bridge. In a preferred embodiment, the bridge is an Ethernet bridge.

Thus, in one aspect, the invention is directed to an adaptable network bridge that includes a Rule Engine that processes data frames according to a set of switching rules. Each switching rule identifies a type of frame to which the rule applies, and how the type of frame is to be processed. The bridge also includes means for dynamically modifying the switching rules. In particular, the rules may be dynamically loaded and updated, for example, in response to changes in the bridge's state information, or changes in context information. In another embodiment, the bridge includes a plurality of input/output ports, and the Rule Engine includes a plurality of sets of switching rules, wherein each set of switching rules is applicable to a different port in the bridge.

In one aspect the rules are forced on the switch by external units such as a

Network Management System (NMS).

In a further aspect, a switch generates or modifies rules that are forced on other switches.

In yet another aspect, the invention is directed to a method of processing and switching Ethernet frames in an adaptable Ethernet bridge. The method includes the steps of defining a set of switching rules for the adaptable Ethernet bridge; receiving a given frame in the bridge; determining whether any of the rules are applicable to the given frame; and upon determining that one of the rules is applicable to the given frame, processing and switching the frame according to the applicable rule. The switching rules may be dynamically added, subtracted, or modified, for example, in response to changes in state or context information.

An important aspect of the invention is how the extended state vector can be utilized in new ways in the switching process. The switching process can be controlled and adapted in new ways allowing completely new ways to control networks.

BRIEF DESCRIPTION OF THE DRAWINGS

In the following, the essential features of the invention will be described in detail by showing preferred embodiments, with reference to the figures of the attached drawings.

FIG. 1 is a simplified block diagram of a Rule Engine utilized in the adaptable network bridge of the present invention;

FIG. 2 is a sequence diagram illustrating the steps of a process performed by the Rule Engine; and

FIG. 3 is an illustration of a network of adaptable Ethernet bridges controlled by a Network Management System (NMS).

DETAILED DESCRIPTION OF THE INVENTION

In the following description, for purposes of explanation and not limitation, specific details are set forth, such as particular embodiments, circuits, signal formats etc. in order to provide a thorough understanding of the present invention. It will be apparent to one skilled in the art that the present invention may be

practiced in other embodiments that depart from these specific details.

In existing Ethernet bridges, there are two identifiable phases in the switching process. In the first phase, parts of the frame header are checked for specific information. For example, the destination address of each frame may be checked against a forwarding table, priority bits, VLAN tags, and the like. In the second phase, different actions are taken within the bridge depending on the result of this check. For example, the frame may be sent to the port pointed to in the forwarding table, or the frame may be sent to all ports except the originating port, and so on. These steps characterize a process in which a frame is put through a filter that selects the action to take. Another important element is that the bridge stores and retrieves information in the form of a simple state vector in a forwarding table (i.e., memory), which the bridge is using in the process. However, the different parts of this process are fixed and may not be altered.

The present invention is based on a generalization of this switching process that adds several new key aspects to switching, enabling new ways of adapting and controlling the switching process as well as new ways to control Ethernet networks. The invention provides an adaptable Ethernet bridge having a Rule Engine that simplifies the addition or modification of functionality within the bridge. One of the purposes of the adaptable bridge is to provide a simple solution whereby anyone with reasonable skill can implement and add new functionality in a standardized way on standard equipment. Simplicity is achieved by specifying how Ethernet frames should be processed and switched in terms of switching rules. Flexibility is achieved by enabling dynamic loading and updating, inspection, and prioritization of rules in bridges via a protocol.

The generalized switching process consists of the Rule Engine processing Ethernet frames by applying the switching rules. A switching rule encapsulates a specific switching behavior that describes the filter and action processes together with the state vector. A switching rule thus consists of three parts:

1) A filter function that can filter any part or parts of a frame. The purpose of the filter is to decide if the rule is applicable to a given frame. In other words, the filter works as a predicate function on frames. In this decision process information in the state vector may be used.

2) An action function that describes how a frame header (or the entire frame) should be transformed and how the frame should be switched, i.e. to which port or ports the frame should be sent. In this process the action function may alter the state vector.

3) A state vector that the filter function and the action function can use in the switching process.

The switching rule can also contain other information such as version, priority specification, vendor information, and the like.

The state vector is an abstraction of several parts:

1) A local memory for a rule or a set of rules where rule-specific data can be stored and altered. Specific "management rules" make it possible to access that data from, for example, a management system.

2) A global bridge memory where bridge-specific data applicable to and accessible by all rules can be stored and altered. One example is the port-forwarding table.

3) Access to a set of bridge resources that can be implemented as library functions. Some examples are a function to retrieve the source port of the frame processed, a function that provides timestamps, counters and timers, and functions that describe the traffic flows and load in the bridge. The invention is not limited to these examples; other functions may be identified by those skilled in the art.

A switching rule describes the types of frames that match the rule and how these frames should be processed (e.g., switched or transformed before switching). Each incoming frame is checked against the set of rules stored in the bridge. If a rule matches, the frame is processed and switched according to the specification in the rule. If no rule matches the frame, a default rule may be utilized, or the frame may be dropped.

With the concept of rules, the switching process is defined as a simple and understandable programming model. Standardizing rule language and making it possible to dynamically load new rules in a bridge also makes it possible to add new functionality, adapting standard equipment to the needs at hand. If some functionality becomes obsolete, that functionality can be removed by removing the

rule or set of rules implementing it.

A powerful extension is to allow different rule sets to be valid for different ports enabling, for example, different network providers to share the same hardware but have different network behavior. Different embodiments of this extension include: a rules control mechanism that selects and prioritizes available rule sets based on port, time of day, load, and the like. Alternatively, this kind of information may be described in the filter functions, and the Rule Engine tries all rules until it finds a match.

Yet another powerful extension is to allow rules that monitor the state vector instead of being matched against Ethernet frames. That is, the filter function monitors the state vector, and when some condition is fulfilled, the action function is invoked. The monitoring process can be implemented by regularly processing these rules in the Rule Engine or in a separate Rule Engine.

Note that allowing the filter and action functions to have access to a state vector that goes beyond the forwarding table provides new and powerful ways to process and switch frames.

FIG. 1 is a simplified block diagram of an exemplary embodiment of the dynamically adaptable network bridge 10 of the present invention. The adaptable bridge includes many of the same components as existing bridges such as switch fabric 11, frame buffers 12a-12n and 13a-13n, and ports 14a-14n and 15a-15n. In addition, however, the adaptable bridge includes a Rule Engine 16, which replaces the implementation of some or all protocols. The adaptable bridge also includes a state vector memory 17 accessible by the Rule Engine, a rule control function 18, and a rules memory 19 accessible by the Rule Engine through the rule control function. Collectively, these new bridge components enable the adaptable bridge to receive and store rules from external units, store a state vector which may be dependent on external sources, and store rules that are imposed on other bridges.

Hybrid designs having both a Rule Engine and some of the protocols of existing bridges may also be implemented. The Rule Engine has processing capabilities, that is, the Rule Engine can (1) apply the filter function to a frame and calculate the result; (2) apply the action function if the rule matches; (3) access

the state vector; and (4) access the other resources of the switch. A frame entering the adaptable bridge through an input port such as port 14a and frame buffer 12a is acted upon by switching rules in the Rule Engine 16, including the filter function, action function, and state vector. The result is a calculated result frame, which is outputted through switch fabric 11, and an appropriate frame buffer and output port such as frame buffer 13a and output port 15a.

The rule control function 18 adds and removes rules dynamically, and orders or prioritizes THE set of rules. The rule control function may be implemented as protocols and secured against unwanted manipulation by using well known techniques intended for that purpose, for example, using digital signatures and encryption. Additionally, the rule control function may include a mechanism to transform (or translate) switching rules in human-readable form to an efficient form understandable by the Rule Engine 16.

The mechanisms to access different bridge resources and state information (i.e., state vectors), mechanisms to access library functions, and a protocol for externally accessing the state information are important parts of the adaptable bridge. The bridge resources and state information may be accessed via, for example, an Application Programming Interface (API). The protocol for externally accessing the state information may be implemented as a set of rules. The state of the bridge can affect switching rules dynamically, that is, a rule can behave differently depending on the current state. For example, frames can be switched differently depending on time of day, traffic pattern, traffic load, and the like. This behavior can also be controlled externally by, for example, a network management system (NMS).

FIG. 2 is a sequence diagram illustrating the steps of a method of processing an Ethernet frame by the Rule Engine 16. The process starts at step 21 and moves to step 22 where, for each incoming Ethernet frame, the port-specific rules associated with the Source Port are merged with the rules shared among all ports of the bridge (this merge can be done in advance by the priority module as a ordered list). The rules in the list are then tested to determine whether the frame matches the rule. This process begins at step 23, where a determination is made whether there are any more rules left to test. If not, and all

the rules have been tested, the frame is dropped at step 24. However, in the case of the first rule, there are rules left to test, so the method moves to step 25 where the filter function of the first rule in the list is applied to the frame. The filter function uses the information found in the frame together with the current state vector to determine whether the frame matches the rule. The filter function is a predicate, so the result at step 26 will be either true (the frame matches the rule) or false (the frame does not match the rule). If the result is false, the rule is discarded at step 27, and the method returns to step 23 where the test is performed on the next rule in the list.

The remainder of the rules are then tested until either no rules are left and the frame is dropped at step 24, or there is a match at step 26, and the method moves to step 28 where the action function of the rule is applied to the frame. The action function may use the library functions, the information in the frame, and the state vector to transform (or modify) the frame. In this process, the action function can use functions to modify the state vector. For example, a counter may be incremented, which counts the number of frames containing a specific high-level protocol coming from the port. At step 29, the action function determines the destination port or ports and sends the (possibly transformed) frame to the determined port or ports. The method then ends at step 30.

Another important aspect of the invention not shown in the sequence diagram is that other resources or modules in the bridge may continuously monitor and change the state vector in parallel. An extended Rule Engine, to the one described, may then invoke rules that, instead of being matched against Ethernet frames, monitor the state vector. When one or more predefined criteria are fulfilled (determined by the filter function), the action function generates new frames containing new rules to be sent to the bridge itself or to nearby bridges altering their behavior.

FIG. 3 is an illustration of a network 35 of adaptable Ethernet bridges controlled by a Network Management System (NMS) 36. The NMS monitors the state in the bridges and issues rules that adapt the behavior of certain bridges accordingly. For example, if $Bridge_{11}$ becomes heavily loaded with traffic, the NMS may issue a rule that will move some traffic from associated bridges (i.e.,

$Bridge_{21}$ – $Bridge_{23}$) to $Bridge_{12}$. For example, the rule may move every third frame or frames carrying a particular protocol to $Bridge_{12}$. Context information may also be added to the state vector, thereby allowing context to influence switching. Context information may contain, but is not limited to, information about traffic load in nearby bridges, forecasts or predictions of expected traffic, and the like. This feature is of particular interest in a mobile environment.

The NMS 36 is shown connected to the network by a dotted line because the adaptable bridges may automatically generate specific context rules and distribute them to nearby bridges as users move around in the network. When bridges automatically generate rules to other bridges (and to themselves), a self-managed network can be implemented in which bridges exchange rules dynamically. This capability can be used, for example, to setup a particular flow-path, distribute traffic during heavy load, or automatically redirect traffic in the case of network failures. For example, if Bridge11 receives a flow of high priority frames when it is already heavily loaded with traffic, it can issue a rule or set of rules to Bridge22 that move the flow to Bridge12 instead.

An Ethernet bridge is working in "wire-speed" and therefore it is desirable that the bridge introduce as little delay as possible. Adding a Rule Engine and basing the switching on rules may increase the delay. However, a number of techniques and mechanisms can be used to make the Rule Engine more efficient:

1) Filter functions can be tested in parallel, and a priority module can select the matching rule with highest priority. These computations can be performed because the filter functions are pure predicates, i.e. they contain no side effects.

2) Filter functions will in most cases be a conjunction (or disjunction) of comparisons. These can be directly mapped onto a pipelined matrix of comparison modules in hardware.

3) A similar pipelined hardware matrix realizing the language constructs used in the action functions can be implemented.

4) Some parts of the state vector, for example library functions reflecting bridge context and resources, can be implemented in hardware such as registers, thereby providing "instant" access to them.

In the preferred embodiment, the rule language used to describe the filter

and action functions is not a general purpose programming language. A programming language should only have the necessary constructs and mechanisms needed to express the solutions to the problems in the specified domain. If the language is too powerful, as a general-purpose language, it is possible to express an infinite loop (accidentally or on purpose) thereby increasing the delay to infinity. The preferred switching rule language, therefore, is a language primarily restricted to the domain of Ethernet switching (layer 2), with language constructs expressing the mechanisms used in the switching process. However, language constructs that are useful for other purposes like network management, constructs related to exchange of new rules, and so on, that are not in conflict with the basic requirements may also be included.

The present invention provides several advantages. The Ethernet standards can be implemented as switching rule sets, and as the standards evolve, the rule sets are updated. These updated rule sets can then easily be loaded into an adaptable network bridge. The rule sets can be vendor-independent and, for example, published by IEEE. Today, each vendor decides which protocols (standards) to support in a bridge, and unless the vendor issues an update whenever the standard is updated, users have to buy a new bridge if they need to follow the new standard.

In the present invention, the "programming language" of rules may be standardized thereby enabling anyone to add new functionality, as a switching rule, to any bridge with a Rule Engine. These rules can then be updated or removed if necessary. A proprietary solution can also be implemented as a rule or set of rules and loaded in any bridge with a Rule Engine. If a user later wants to use another solution, the rule set is simply replaced with a new rule set. It will be much less expensive to implement new functionality because, in most cases, the user only has to write some software in the form of a set of rules.

The present invention also greatly increases flexibility in networks because each bridge can be loaded with its own rule set down to the individual port level. The invention also provides the ability to adapt the functionality on each port based on time, user demands, service agreements, and the like. For example, different operators can share the same physical bridge (network) and provide

different services to their respective users.

Some examples of functionality that can be implemented with switching rules include the following:

1) Broadcast tunnels. It is simple to implement broadcast tunnels necessary for building Public Ethernet Networks.

2) Distributed router functionality added to a set of bridges.

3) Simple firewall functionality.

4) User adapted services on the port level.

5) Authentication and security mechanisms on the port level that are transparent to users.

6) Self-managed networks or semi-self managed networks.

This list is not intended to be exhaustive, and other functions may also be implemented with switching rules and loaded into any adaptable bridge with a Rule Engine.

As shown by the above examples, the state of the dynamically adaptable bridge can be utilized in the switching process to enable entirely new functionality and to provide new ways to use bridges in networks.

Although preferred embodiments of the present invention have been illustrated in the accompanying drawings and described in the foregoing Detailed Description, it is understood that the invention is not limited to the embodiments disclosed, but is capable of numerous rearrangements, modifications, and substitutions without departing from the scope of the invention. The specification contemplates any all modifications that fall within the scope of the invention defined by the following claims.

WHAT IS CLAIMED IS:

1. A dynamically adaptable network bridge comprising:

a Rule Engine that processes data frames according to a set of switching rules, wherein each switching rule identifies a type of frame to which the rule applies, and how the type of frame is to be processed; and

means for dynamically modifying the switching rules.

2. The dynamically adaptable network bridge of claim 1 wherein the means for dynamically modifying the switching rules includes a protocol for dynamically loading and updating the switching rules.

3. The dynamically adaptable network bridge of claim 1, wherein each switching rule includes a filter function for determining whether a given rule is applicable to a given frame.

4. The dynamically adaptable network bridge of claim 3, wherein each switching rule also includes an action function for specifying changes to the given frame's header and how the given frame should be switched.

5. The dynamically adaptable network bridge of claim 1, wherein each switching rule includes a state vector utilized for processing the type of frames to which each rule applies.

6. The dynamically adaptable network bridge of claim 5, wherein the state vector includes an abstraction of a local memory for storing rule-specific data for each switching rule.

7. The dynamically adaptable network bridge of claim 6, wherein specific management rules enable the Rule Engine to access the rule-specific data for each switching rule.

8. The dynamically adaptable network bridge of claim 6, wherein the state vector also includes an abstraction of a global bridge memory for storing bridge-specific data applicable to and accessible by all the switching rules.

9. The dynamically adaptable network bridge of claim 8, wherein the bridge-specific data includes a port-forwarding table.

10. The dynamically adaptable network bridge of claim 8, wherein the state vector also includes a set of library functions that implement bridge resources.

11. The dynamically adaptable network bridge of claim 10, wherein the set of library functions are selected from a group consisting of:

a function that retrieves a source port of a processed frame;

a function that provides timestamps;

a function that describes traffic flows in the bridge; and

a function that indicates the traffic load in the bridge.

12. The dynamically adaptable network bridge of claim 3, wherein the given frame is processed in accordance with the given rule if the filter function determines that the given rule is applicable to the given frame, and the given frame is dropped if the filter function determines that there is no rule applicable to the given frame.

13. The dynamically adaptable network bridge of claim 1, wherein the bridge includes a plurality of input/output ports, and the Rule Engine includes a plurality of sets of switching rules, wherein each set of switching rules is applicable to a different port in the bridge.

14. The dynamically adaptable network bridge of claim 1, wherein the bridge is an Ethernet bridge.

15. The dynamically adaptable network bridge of claim 1, wherein the means for dynamically modifying the switching rules includes a protocol that dynamically adds, subtracts, and modifies the switching rules.

16. The dynamically adaptable network bridge of claim 1, further comprising means for prioritizing rules within the set of switching rules.

17. The dynamically adaptable network bridge of claim 1, further comprising means for accessing state information for the bridge, wherein a given rule causes a given frame to be processed differently when the state information changes.

18. The dynamically adaptable network bridge of claim 17, wherein the given rule processes the given frame differently, depending on traffic load in the bridge.

19. The dynamically adaptable network bridge of claim 17, further comprising a port for receiving rules from a Network Management System (NMS), wherein the NMS monitors the state information for a plurality of adaptable bridges in a network, and issues rules that adapt the behavior of the bridge to optimize network performance.

20. The dynamically adaptable network bridge of claim 17, wherein the state information includes context information, wherein a given rule causes a given frame to be processed differently when the context information changes.

21. The dynamically adaptable network bridge of claim 1, wherein the bridge is one of a plurality of adaptable bridges in a network, and the bridge further comprises means for generating specific context rules and distributing the context rules to other adaptable bridges in the network.

22. A method of processing and switching data frames in an adaptable network bridge, said method comprising the steps of:

defining a set of switching rules for the adaptable bridge;

receiving a given frame in the bridge;

determining whether any of the rules are applicable to the given frame; and

upon determining that one of the rules is applicable to the given frame, processing and switching the frame according to the applicable rule.

23. The method of claim 22, further comprising dynamically adding, subtracting, and modifying the switching rules.

24. The method of claim 23, further comprising accessing state information for the bridge, wherein the step of dynamically adding, subtracting, and modifying the switching rules is performed in response to a change in the state information.

25. The method of claim 24, wherein the bridge is one of a plurality of adaptable bridges in a network, and the method further comprises the steps of:

receiving a new or modified switching rule from a Network Management System (NMS), wherein the NMS monitors the state information for a plurality of adaptable bridges in a network, and issues rules that adapt the behavior of the bridge to optimize network performance; and

processing and switching frames according to the new or modified switching rule received from the NMS.

26. The method of claim 24, further comprising adding context information to the state information, wherein the step of dynamically adding, subtracting, and modifying the switching rules is performed in response to a change in the context information.

27. The method of claim 22, wherein the bridge is one of a plurality of adaptable bridges in a network, and the method further comprises the steps of:

generating specific context rules; and

distributing the context rules to other adaptable bridges in the network.

28. The method of claim 22, wherein the bridge is an Ethernet bridge.

5

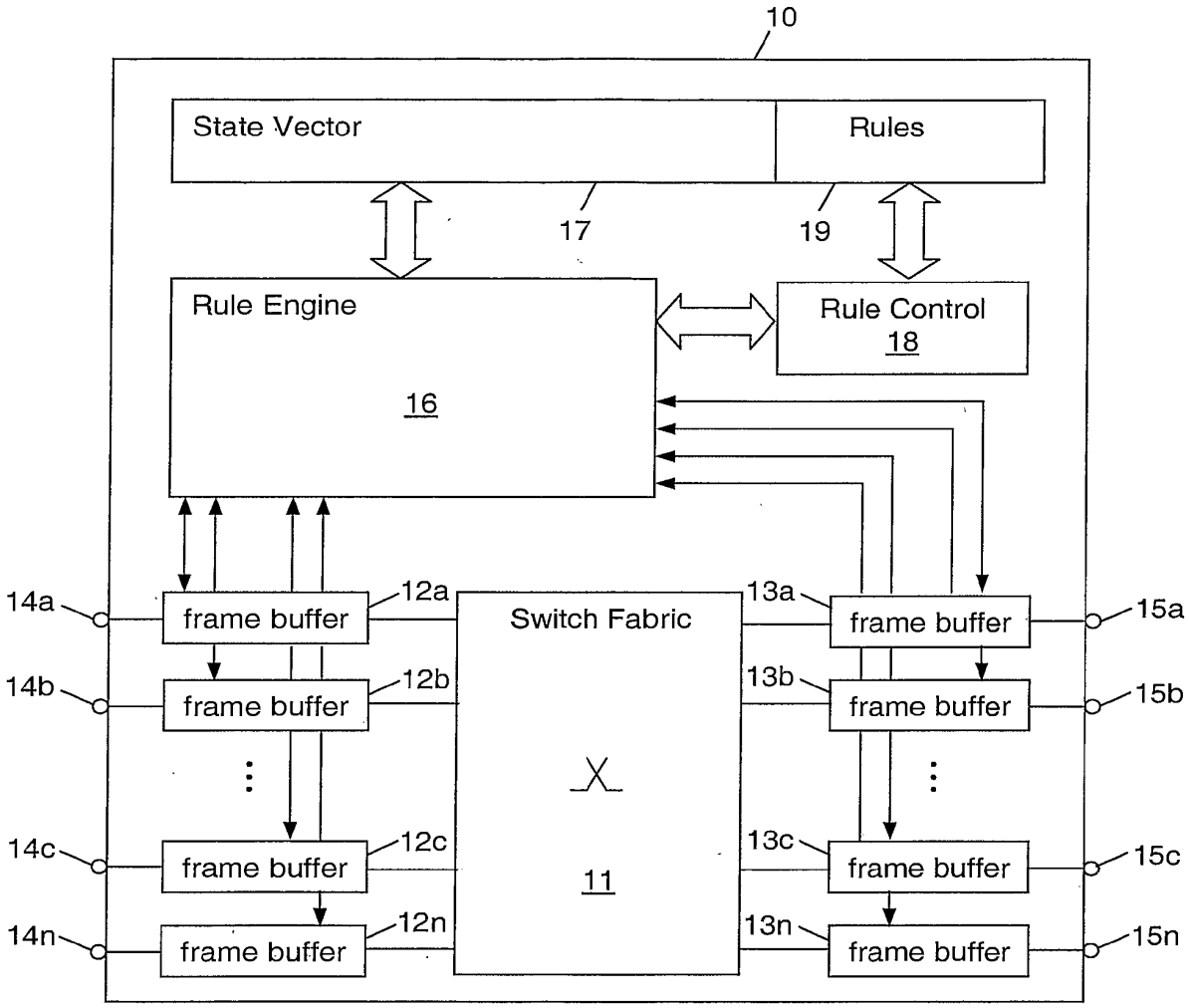

10
State Vector
Rules
17
19
Rule Engine
16
Rule Control
18
14a
frame buffer
12a
Switch Fabric
13a
frame buffer
15a
14b
frame buffer
12b
13b
frame buffer
15b
14c
frame buffer
12c
11
13c
frame buffer
15c
14n
frame buffer
12n
13n
frame buffer
15n

FIG. 1

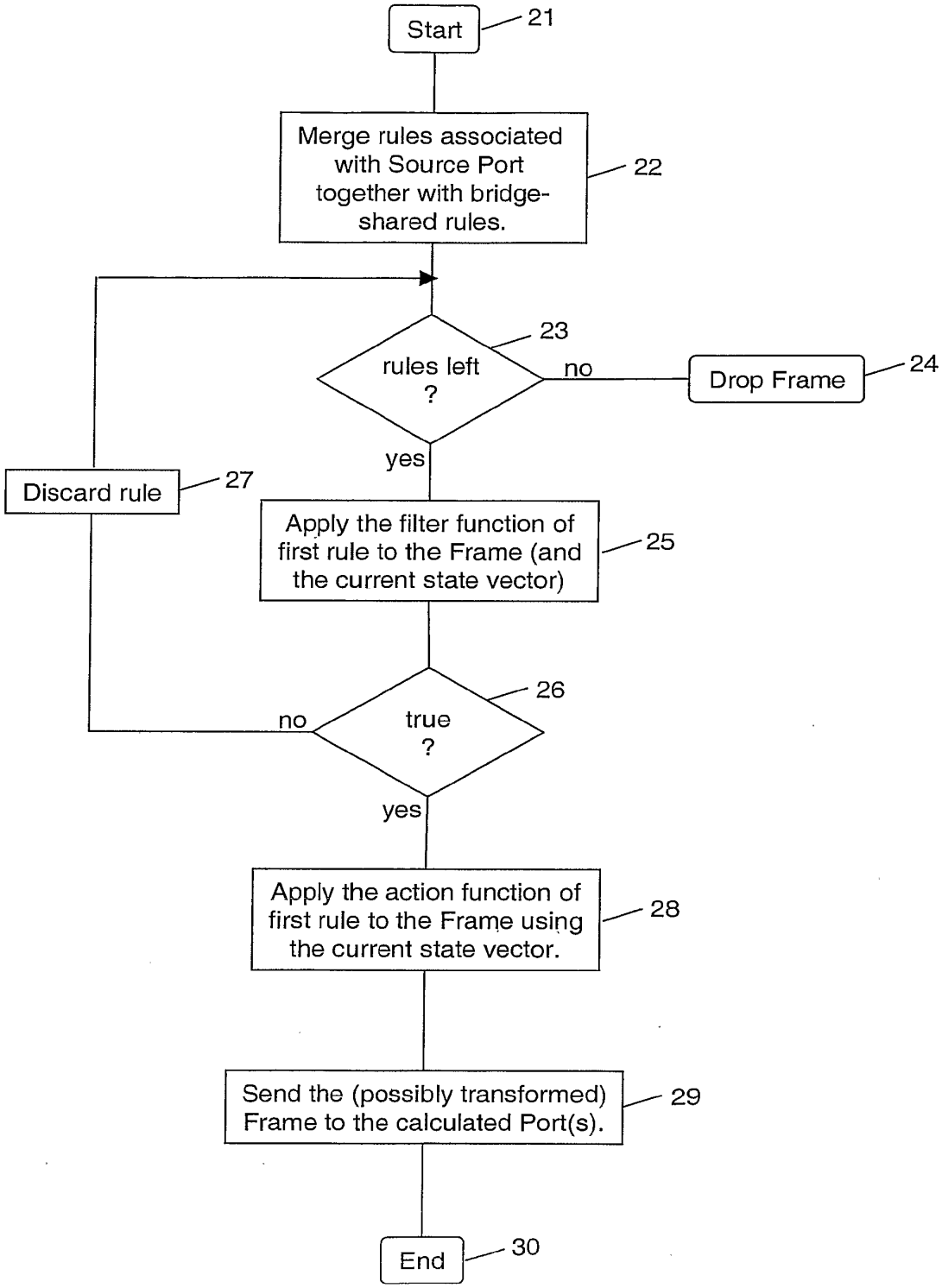

Start
21
Merge rules associated with Source Port together with bridge-shared rules.
22
rules left ?
23
no
Drop Frame
24
yes
Discard rule
27
Apply the filter function of first rule to the Frame (and the current state vector)
25
true ?
26
no
yes
Apply the action function of first rule to the Frame using the current state vector.
28
Send the (possibly transformed) Frame to the calculated Port(s).
29
End
30

FIG. 2

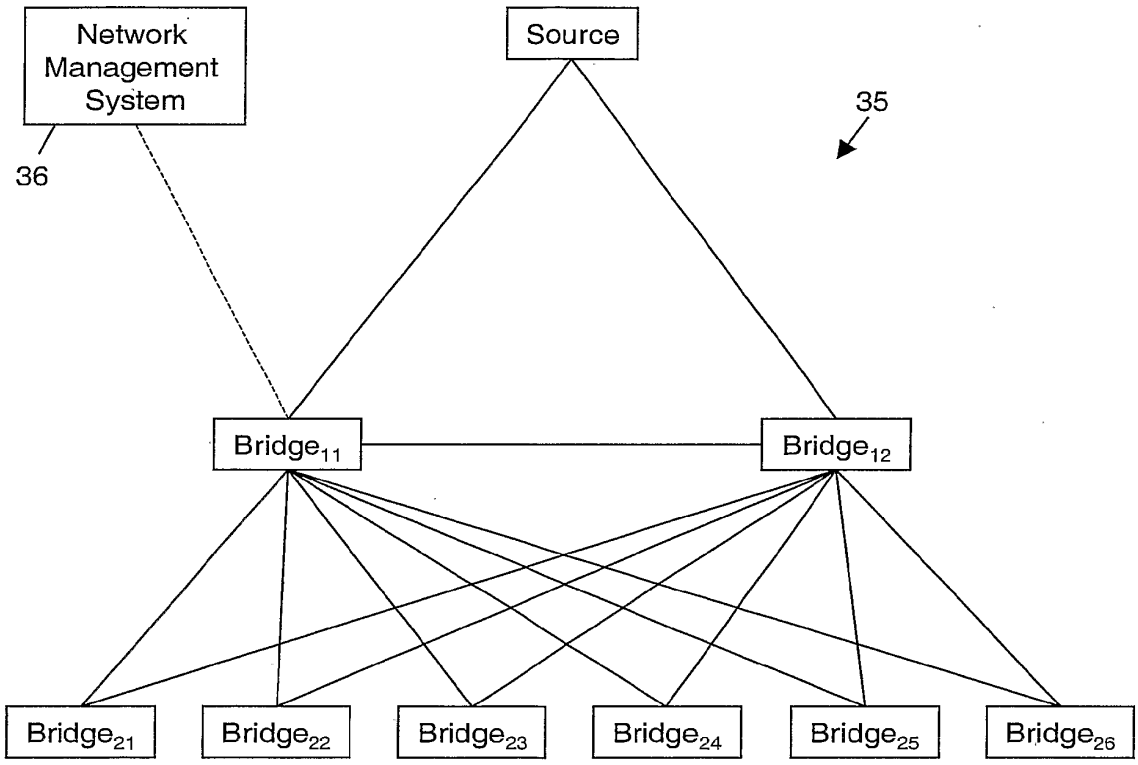

Network Management System
36
Source
35
Bridge11
Bridge12
Bridge21
Bridge22
Bridge23
Bridge24
Bridge25
Bridge26

FIG. 3

INTERNATIONAL SEARCH REPORT

International Application No
PCT/SE2004/001611

A. CLASSIFICATION OF SUBJECT MATTER
IPC 7 H04L12/56 H04L12/46

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)
IPC 7 H04L

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 practical, search terms used)

EPO-Internal, WPI Data, PAJ

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category °	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	US 5 790 554 A (PITCHER ET AL) 4 August 1998 (1998-08-04) abstract column 1, line 15 - column 1, line 67 column 2, line 39 - column 3, line 25 column 3, line 48 - column 4, line 42 column 4, line 60 - column 8, line 4 column 8, line 24 - column 8, line 26; claims 1,2,14,22,28; figures 2,3 ----- -/--	1-5, 12-20, 22-25, 27,28

[X] Further documents are listed in the continuation of box C.

[X] Patent family members are listed in annex.

° Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier document 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
21 March 2005	31/03/2005

Name and mailing address of the ISA	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL – 2280 HV Rijswijk Tel. (+31–70) 340–2040, Tx. 31 651 epo nl, Fax: (+31–70) 340–3016	Todorut, C

Form PCT/ISA/210 (second sheet) (January 2004)

IPR2023-01353 Page 00569

INTERNATIONAL SEARCH REPORT

International Application No
T/SE2004/001611

C.(Continuation) DOCUMENTS CONSIDERED TO BE RELEVANT

Category °	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	US 5 396 493 A (SUGIYAMA ET AL) 7 March 1995 (1995-03-07) abstract column 1, line 63 - column 3, line 11 column 4, line 11 - column 4, line 51 column 5, line 67 - column 6, line 39; claims 1,17; figures 1-4	1-28
X	WO 00/56024 A (BROADCOM CORPORATION; KADAMBI, SHIRI; AMBE, SHEKHAR; KALKUNTE, MOHAN;) 21 September 2000 (2000-09-21) abstract	1-4,22, 23
A	page 53, line 12 - page 65, line 21; claims 1,2	5-21, 24-28
A	US 2003/123387 A1 (JACKSON ANDREW LLOYD 'GB!) 3 July 2003 (2003-07-03) abstract column 1, paragraph 2 - column 1, paragraph 9 column 2, paragraph 16 - column 3, paragraph 25; claims 1,6,9; figure 4	1-28
A	US 5 959 976 A (KUO ET AL) 28 September 1999 (1999-09-28) abstract column 1, line 5 - column 2, line 40 column 3, line 8 - column 3, line 18 column 5, line 56 - column 6, line 10; claim 1; figures 3-5	1-28
A	EP 0 508 886 A (DIGITAL EQUIPMENT CORPORATION) 14 October 1992 (1992-10-14) abstract column 1, line 40 - column 3, line 54 column 4, line 45 - column 5, line 16 column 5, line 55 - column 6, line 53; claims 1,23; figures 5-8	1-28
A	US 5 515 376 A (MURTHY ET AL) 7 May 1996 (1996-05-07) abstract column 6, line 1 - column 9, line 21	1-28

1

Form PCT/ISA/210 (continuation of second sheet) (January 2004)

IPR2023-01353 Page 00570

INTERNATIONAL SEARCH REPORT

Information on patent family members

International Application No
/SE2004/001611

Patent document cited in search report		Publication date	Patent family member(s)		Publication date
US 5790554	A	04-08-1998	NONE		
US 5396493	A	07-03-1995	JP	6224918 A	12-08-1994
			KR	9614983 B1	23-10-1996
WO 0056024	A	21-09-2000	US	2001043611 A1	22-11-2001
			AU	3529500 A	04-10-2000
			AU	3752000 A	04-10-2000
			AU	4325800 A	04-10-2000
			EP	1161817 A2	12-12-2001
			EP	1169809 A2	09-01-2002
			EP	1159805 A2	05-12-2001
			WO	0056024 A2	21-09-2000
			WO	0056011 A2	21-09-2000
			WO	0056013 A2	21-09-2000
			US	6707817 B1	16-03-2004
			US	6810037 B1	26-10-2004
			US	6850521 B1	01-02-2005
			US	6707818 B1	16-03-2004
			US	2004174898 A1	09-09-2004
			US	2005047411 A1	03-03-2005
			US	2002039365 A1	04-04-2002
			AU	5442700 A	12-12-2000
			AU	5586400 A	12-12-2000
			AU	5737300 A	31-01-2001
			AU	6334400 A	22-01-2001
			EP	1181791 A1	27-02-2002
			EP	1181792 A1	27-02-2002
			EP	1212867 A2	12-06-2002
			EP	1192767 A2	03-04-2002
			WO	0072533 A1	30-11-2000
			WO	0102965 A2	11-01-2001
			WO	0072531 A1	30-11-2000
			WO	0101724 A2	04-01-2001
			US	2002093974 A1	18-07-2002
			US	2002186705 A1	12-12-2002
			US	6813268 B1	02-11-2004
			US	6735679 B1	11-05-2004
			US	6792500 B1	14-09-2004
			US	6842457 B1	11-01-2005
			US	6430188 B1	06-08-2002
			US	6859454 B1	22-02-2005
			US	2004170176 A1	02-09-2004
			US	2005036488 A1	17-02-2005
			US	2002031090 A1	14-03-2002
US 2003123387	A1	03-07-2003	NONE		
US 5959976	A	28-09-1999	NONE		
EP 0508886	A	14-10-1992	DE	69217103 D1	13-03-1997
			DE	69217103 T2	21-08-1997
			EP	0508886 A1	14-10-1992
			JP	5114905 A	07-05-1993
			US	5515513 A	07-05-1996
US 5515376	A	07-05-1996	EP	0710415 A1	08-05-1996
			JP	3335358 B2	15-10-2002

INTERNATIONAL SEARCH REPORT

Information on patent family members

International Application No
/SE2004/001611

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 5515376 A		JP 9500774 T	21-01-1997
		WO 9503659 A1	02-02-1995
		US 5610905 A	11-03-1997
		US 6545982 B1	08-04-2003

Espacenet

Bibliographic data: EP2385676 (A1) — 2011-11-09

Method for adapting security policies of an information system infrastructure

Inventor(s): CUPPENS FREDERIC [FR]; CUPPENS NORA [FR]; DUBUS SAMUEL [FR]; KANOUN WAEL [FR] ± (CUPPENS, FREDERIC, ; CUPPENS, NORA, ; DUBUS, SAMUEL, ; KANOUN, WAEL)

Applicant(s): ALCATEL LUCENT [FR]; INST TELECOM TELECOM BRETAGNE [FR] ± (ALCATEL LUCENT, ; INSTITUT TELECOM - TELECOM BRETAGNE)

Classification: **- international:** ***G06F21/57; H04L29/06***
- cooperative: **G06F21/577**; **H04L63/1408**; **H04L63/1433**; **H04L63/20**; H04L63/1441

Application number: EP20100290250 20100507

Priority number (s): EP20100290250 20100507

Also published as: US2013111548 (A1) KR20130005301 (A) JP2013525927 (A) WO2011138417 (A1) CN102934122 (A)

Abstract of EP2385676 (A1)

The present invention refers to a method for adapting security policies of an information system infrastructure in function of attacks wherein it comprises the steps of: - storing potential attacks and their associated risks in a data repository (125); - storing curative security policies (128) in response of the potential attacks in a data repository; - monitoring (101) entering contents representing data streams of the information system; - detecting (129) at least one attack in the information system; - assessing a success probability parameter (132) of the at least one detected attack and its associated cost impact parameter (133); -

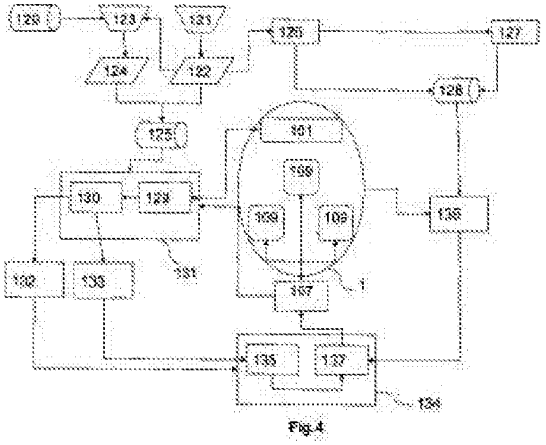


Fig.4

assessing an activation impact parameter (136) of at least one curative security policy in response to the at least one detected attack and its associated cost impact parameter; - deciding of the activation or deactivation (134) of a curative security policy in function of the success probability parameter of the, at least one, detected attack, of the activation impact parameter of at least one curative security policy and of the cost impact parameters of both the detected at least one attack and the at least one curative security policy.

(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) EP 2 385 676 A1

(12) EUROPEAN PATENT APPLICATION

(43) Date of publication:
09.11.2011 Bulletin 2011/45

(51) Int Cl.:
H04L 29/06 *(2006.01)* ***G06F 21/00*** *(2006.01)*

(21) Application number: **10290250.9**

(22) Date of filing: **07.05.2010**

(84) Designated Contracting States:
AL AT BE BG CH CY CZ DE DK EE ES FI FR GB GR HR HU IE IS IT LI LT LU LV MC MK MT NL NO PL PT RO SE SI SK SM TR
Designated Extension States:
BA ME RS

(71) Applicants:
- **Alcatel Lucent**
  75008 Paris (FR)
- **Institut Télécom - Télécom Bretagne**
  29238 BREST cedex 3 (FR)

(72) Inventors:
- **Cuppens, Frédéric**
  29238 Brest Cedex 3 (FR)
- **Cuppens, Nora**
  29238 Brest Cedex 3 (FR)
- **Dubus, Samuel**
  91620 Nozay (FR)
- **Kanoun, Wael**
  91620 Nozay (FR)

(74) Representative: **Croonenbroek, Thomas Jakob et al**
Cabinet Innovincia
11, avenue des Tilleuls
74200 Thonon-les-Bains (FR)

(54) Method for adapting security policies of an information system infrastructure

(57) The present invention refers to a method for adapting security policies of an information system infrastructure in function of attacks wherein it comprises the steps of:
- storing potential attacks and their associated risks in a data repository (125);
- storing curative security policies (128) in response of the potential attacks in a data repository;
- monitoring (101) entering contents representing data streams of the information system;
- detecting (129) at least one attack in the information system;
- assessing a success probability parameter (132) of the at least one detected attack and its associated cost impact parameter (133);
- assessing an activation impact parameter (136) of at least one curative security policy in response to the at least one detected attack and its associated cost impact parameter;
- deciding of the activation or deactivation (134) of a curative security policy in function of the success probability parameter of the, at least one, detected attack, of the activation impact parameter of at least one curative security policy and of the cost impact parameters of both the detected at least one attack and the at least one curative security policy.

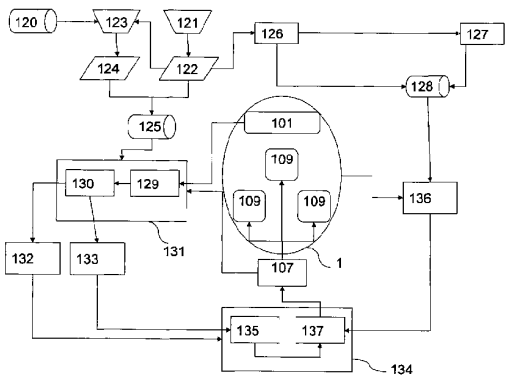


Fig.4

EP 2 385 676 A1

Description

[0001] The present invention relates to the field of information system protection and more precisely to the management of security policy in function of attacks undergone or being undergone by the information system.

[0002] Infrastructures of information systems need to be protected from harmful attacks leading to malicious events such as intrusions, data thefts, viruses or worms...

[0003] Due to the potential number of alerts generated by the existing attack detection systems with large information system infrastructures, it becomes impossible for operators to assess in real-time the risk of an attack and to decide of the suitable response to apply in response to the attack. Thus, automatic deployment of formally defined operational security policies starts to be considered in the protection of telecommunication and information infrastructures.

[0004] Fig.1 represents an example of such automatic protection of the state of the art.

[0005] The first step 101 corresponds to the detection of attacks toward the monitored information system which leads to the creation of elementary alerts (102). An alert correlation is then processed (103) to define correlated alerts (104) that are sent to a policy instantiation engine (105) to activate the appropriate security rules (106). These rules are sent to a policy decision point (107) which generates the configuration scripts (108) which are then used to configure policy enforcement points (109). Said policy enforcement points (109) are located in the information system 1 and apply the security rules in response to the detected attacks.

[0006] Such configuration of automatic policy activation suffers from drawbacks. Indeed, it is based only on correlated alerts and the number of correlated alerts may reach a very large number (up to thousands within a single day with large system) which would lead to thousands of security policy activations. Moreover, the deactivation of said security policies is not taken into account in the configurations of the state of the art such that a security policy may remain activated even if its impact on the users of the information system 1 is worth than the impact of the attack.

SUMMARY OF THE INVENTION

[0007] One object of the present invention is therefore to overcome the precited drawbacks of the state of the art and offer a method that allow to dynamically trig security policy activations only when it is necessary by taking into account a plurality of parameters influencing the activation decision and by defining a method that allow the deactivation of a security policy when it is necessary.

[0008] This is achieved by a method for adapting security policies of an information system infrastructure in function of attacks wherein it comprises the steps of:

- storing potential attacks and their associated risks in a data repository;
- storing curative security policies in response of the potential attacks in a data repository;
- monitoring entering contents representing data streams of the information system;
- detecting at least one attack in the information system;
- assessing a success probability parameter of the at least one detected attack and its associated cost impact parameter;
- assessing an activation impact parameter of at least one curative security policy in response to the at least one detected attack and its associated cost impact parameter;
- deciding of the activation or deactivation of a curative security policy in function of the success probability parameter of the, at least one, detected attack, of the activation impact parameter of at least one curative security policy and of the cost impact parameters of both the detected at least one attack and the at least one curative security policy.

[0009] According to another aspect of the invention, the step of storing potential attacks and their associated risks in a data repository comprises the steps of:

- defining the information system topology and attack detection signatures;
- defining a risk analysis of the information system that identifies potential attack objectives;
- specifying attack models to reach the identified attack objectives;
- storing said attack models in a data repository.

[0010] According to a further aspect of the invention, the step of storing curative security policies in response of the potential attacks in a data repository comprises:

- specifying at least one attack context;
- specifying curative security policies corresponding to the specified at least one attack context;
- storing said curative security policies in a data repository.

[0011] According to an additional aspect of the present invention, the step of assessing a success probability parameter of the at least one detected attack and its associated cost impact parameter comprises:

- generating attack strategy graphs based on the stored attack models and the detected at least one attack;
- assessing the probability for the attack to reach its objective;
- assessing the impact of the attack objective on the system security level and on the system quality of service (QoS) level;
- assessing the associated cost impact parameter of

IPR2023-01353 Page 00576

the attack objective;

[0012] According to another aspect of the present invention, the step of assessing an activation impact parameter of at least one curative security policy in response to the at least one detected attack and its associated cost impact parameter is based on the stored curative security policies and the state of the monitored information system.

[0013] According to a further aspect of the invention, the step of deciding of the activation or deactivation of a curative security policy in function of the success probability parameter of the, at least one, detected attack, of the activation impact parameter of at least one curative security policy and of the cost impact parameters of both the detected at least one attack and the at least one curative security policy is applied dynamically based on the evolution of the state of the monitored system.

[0014] According to an additional aspect of the invention, the cost impact parameters comprise:

- a quality of service (QoS) impact and,
- a security level degradation impact.

[0015] The present invention also refers to a monitoring and protecting equipment comprising:

- at least one data repository for:
  - storing potential attacks and their associated risks;
  - storing curative security policies in response of the potential attacks;
- processing means for:
  - monitoring entering contents representing data streams of the information system;
  - detecting at least one attack in the information system;
  - assessing a success probability parameter of the detected at least one attack and its associated cost impact parameter;
  - assessing at least one activation impact parameter of at least one curative security policy in response to the at least one detected attack and its associated cost impact parameter;
  - deciding of the activation of a curative security policy in function of the success probability parameter of the at least one detected attack, of the activation impact parameter of at least one curative security policy and of the cost impact parameters of both the at least oneattack and the at least one curative policy,
  - activating at least one curative security policy.

[0016] The present invention also refers to a monitoring and protecting equipment comprising:

- at least one data repository for:
  - storing potential attacks and their associated risks;
  - storing curative security policies in response of the potential attacks;
- processing means for:
  - monitoring entering contents representing data streams of the information system;
  - detecting at least one attack in the information system;
  - assessing a success probability parameter of the detected at least one attack and its associated cost impact parameter;
  - assessing at least one activation impact parameter of at least one curative security policy in response to the at least one detected attack and its associated cost impact parameter;
  - deciding of the deactivation of a curative security policy in function of the success probability parameter of the at least one detected attack, of the activation impact parameter of at least one curative security policy and of the cost impact parameters of both the at least one attack and the at least one curative policy,
  - deactivating at least one curative security policy.

BRIEF DESCRIPTION OF THE DRAWINGS

[0017]

FIG.**1** is a synoptic diagram of the different steps of a protective method for launching security policies against attacks of an information system according to the state of the art;

FIG.**2** is a synoptic diagram of the different steps of a protective method for adapting security policies of an information system infrastructure against attacks according to the present invention;

FIG.**3** is a diagram of an example of attack graph comprising the different attack steps and the objectives associated with these attacks;

FIG.4 is a synoptic diagram of the different steps of a detailed protective method for adapting security policies of an information system infrastructure against attacks according to the present invention;

DETAILED DESCRIPTION OF THE INVENTION

[0018] As used herein, the term "attack" refers to an event in a system that transgress the normal authorized usage of the system or exploits deliberately or accidentally a vulnerability in the system as for example a network scanning, a password cracking, a sending of malicious email (also called spam), a sending of a malformed internet protocol (IP) packet... Moreover, the expression

IPR2023-01353 Page 00577

attack with respect to an information system refers to attacks initiated from outside or inside of the information system (usually from an attacker machine) and directed toward said information system in order to produce dysfunctionings in said system.

[0019] As used herein, the term "SIP" refers to the acronym session Internet protocol.

[0020] As used herein, the term "IP" refers to the acronym Internet protocol.

[0021] As used herein, the term "QoS" refers to the acronym quality of service.

[0022] The embodiments of the present invention refer to a method for activating and deactivating security policies in order to protect an information system against malicious attacks wherein not only impacts of the attacks but also the success likelihood for the attack to reach its objective and impacts of security policies are taken into account in the activation/deactivation decision in order to minimize the impact, and in particular the cost, of attacks on the users of the information system.

[0023] Fig.2 represents the general steps of the method for adapting the protection of the information system 1.

[0024] Step 110 refers to the specification and the storing in a data repository of potential attacks and their associated risks. This first step allows to provide attack models that define the strategy and the objectives of the possible attacks.

[0025] Step 111 refers to specification and storing in a data repository of the foreseen security policies to apply in response to the possible attacks defined in the previous step.

[0026] Step 112 refers to the monitoring of entering content of the information system 1 corresponding to data streams and the attacks detection.

[0027] Step 113 refers to the assessment of a success probability of detected attacks and the cost impact associated with these attacks. Such assessment is achieved based on the attack models stored in step 110 and the attacks detected in step 112.

[0028] Step 114 refers to the assessment of the impact of an activation of a curative security policy in response to a detected attack and the cost impact associated with this activation. Such assessment is achieved based on the security policies stored in step 111 and the state of the monitored information system 1.

[0029] Step 115 refers to the decision of activating or deactivating a curative security policy in function of the assessments achieved in steps 113 and 114. Such decision corresponds to the comparison of the cumulative impacts of both the attack and the curative security policy on the information system 1.

[0030] The activation or deactivation decided in step 115 is then applied in step 116.

[0031] Moreover, it has to be noted that the steps 110 and 111 are preliminary steps which can be achieved offline whereas the steps 112, 113, 114, 115, 116 are dynamic steps which are achieved online such that any modification of the information system is taken into account in real time to decide of the activation or the deactivation of security policies.

[0032] Thus, the present invention, thanks to the assessments of both the attack consequences and the consequences of the activation of a security policy allows to determine if it is worthy to activate a curative security policy and when said policy is activated to determine if it is worthy to keep this policy activated in function of the evolution of the information system and the impact of this policy on said information system.

[0033] As described above, an attack is an event occurring in a system. In practice, an attack comprises different levels (or steps) that lead to the objective of the attack as described in fig.3 where an example of attack graph aimed at hacking a voice over internet protocol (VoIP) system is described.

[0034] The first level 201 corresponds to the sending of an email (malicious email or spam) toward a victim machine. Then the below (or next) levels represent the different steps to reach the objectives (213, 214, 215 and 216). The second level 202 refers to the gain of a remote shell in the victim machine (corresponding to the opening by the user of the malicious link or attachment of the email). The third level 203 corresponds to a bot (robot) infection which is the installation of a malicious software bot capable of simulating human activity that uses the remote shell. The bot can wait for future orders from the attacker to execute. As represented in fig.3, several remote shells and several corresponding bot infections may be develop in parallel in the information system.

[0035] The next level 204 corresponds to the discovery of the session initial protocol (SIP) discovery which is a scanning of the system (or network) to discover machines (computers in general) or servers using SIP protocol. Such attack may be done by a bot.

[0036] From this step, the attack may use two different ways depending on its objective.

[0037] The first way leads to step 205 which refers to the SIP fingerprinting which consists of identifying the type and the version of the software (operating systems, softphones, servers, etc...) installed on the SIP entities discovered at the previous level (204). This level may also be performed by a bot.

[0038] From step 205, the attack may have two different ways again, one leading to the objective of spam over IP (SPIT) 213 and comprising two levels, a discovery of the active users which is a scanning to determine the users of the VoIP system and which may be performed by a bot and a direct call 209 referring to a call to the victim user (by a bot) to perform spam over IP (SPIT).

[0039] The other possibility from level 205 is the discovery of the media access control (MAC) address of potential victim machines 207 (which may be performed by a bot).

[0040] The next step 208 is then the address resolution protocol (ARP) poisoning which forces the traffic (e.g. the established calls) between two victim users to pass through one bot. The attacker may then have access to

IPR2023-01353 Page 00578

all the traffic between both victim users. This leads either to a step of sniff audio 211 to sniff the audio packets transmitted between both users or a step of inject audio 212 to inject audio packets in the transmission between both users for altering the established call. The corresponding respective objectives being the conversation tapping 214 and the conversation injection 215.

[0041] From level 204 the second attack solution consists in a server flooding 210 wherein the bots flood the main server of the VoIP system to cause a denial of service (DoS) which is the objective 216.

[0042] It has to be noted that the closer to the objective is the attack, the higher the probability to reach the objective. Thus, when a spam (level 201) is detected, it is generally not worth activating a security policy as the probability for the attack to reach one of the objectives (213, 214, 215 or 216) is still low whereas if the server is flooded (level 210) for example, then the probability of reaching the objective 216 is very high.

[0043] The graphs such as presented in fig.3 are achieved automatically based on attack models stored in data repository and are used in the assessment of the impact of an attack and its probability to reach its objective as described previously.

[0044] In order to better understand the different steps of the invention, a detailed configuration of a possible embodiment of the invention will now be described based on fig.4, it refers to a more detailed presentation of the organization presented in fig.2.

[0045] Step 120 corresponds to the storing in a data repository of the information system topology and the attack or intrusion detection signatures (IDS). Step 121 is a risk analysis for the monitored system which is conducted by an expert to identify potential attack objectives 122. Based on these attack objectives and the system topology (stored in step 120), attack models are specified (by an expert) 123 as well as an attack context specification 126. Attack models such as presented in fig.4 are then set up 124 and stored as elementary models in a data repository 125.

[0046] On the other hand, the specification of attack contexts 126 allows to specify response contexts 127. For each identified attack context an appropriate response context is specified with the associated security rules. Said security rules and attack contexts are stored as security policies in a data repository 128. Said security policies refer, for example, to firewall activation, authentication request activation or user account blocking or any actions allowing to cure the information system or reduce the impact of the attack. Based on the stored attack models and security policies to apply in response to each attack, the automatic and dynamic part of the invention may be implemented.

[0047] The information system 1 is monitored by intrusion detection systems (IDS) 101 which correspond to any types of sensors capable of detecting an attack and of generating and sending alerts. Said sensors may be made of specific electronic devices (micro-controllers, application specific integrated systems (ASICs)...) or in an information technology (IT) environment, it may simply be a software run by a computer, a server or a router.

[0048] The alerts sent by the IDS are then aggregated 129 in an online correlation engine 131 which uses these aggregated alerts and the attack models stored in a data repository to generate attack graphs such as presented in fig.3. The generated graphs are used, on one hand, for the assessment of the probability for the attack to reach its objective 132 and, on the other hand, for the assessment of the impact of the attack 133.

[0049] The determination of the probability to reach the objective corresponds to the success likelihood for an attack to reach an objective. Thus, the success likelihood level of each attainable objective (predefined in the attack graph) of the attack is calculated. This level shows how close the attack is with respect to its objective.

[0050] The attack impact assessment corresponds to the determination of the impact of the attack on the security and the quality of service (QoS) of the information system 1. This gathers the impact on the confidentiality, the integrity and the availability and any parameters influencing the using of the information system 1. Such assessment is also based on the state of the information system 1.

[0051] The results of both assessments are then sent to a policy instantiation engine (PIE) 134 where the cumulative impact of the attacks is determined.

[0052] Besides, in the same way as the assessment of the impact of the attacks, an assessment of the impact of the curative security policies which can be used in response of the attacks 136 is done based on the stored security policies. Again, impact on both the security and the QoS of the information system as well as as the state of said system are taken into account in the assessment. The results of the assessment are also transmitted to the PIE 134 where the cumulative impact of the curative security policies is determined.

[0053] The policy instantiation engine 134 receives the assessments corresponding on one side of the attack impact and on the other side of the curative security policy impact which are processed respectively by an attack handler module 135 and a response handler module 137. For each type of impact, at least one metric is defined, and in the general case, a plurality of metrics corresponding to different aspects of the impact (on security, on QoS...) are defined and the assessed value of all these impact contributions are added (practically by integrated their value over time).

[0054] Based on both cumulative impacts computed by the attack handler module and the response handler module as well as the probability for the attack to reach its objective (also called success likelihood) and the state of the monitored information system 1, the PIE 134 determines if curative security policies have to be activated or deactivated.

[0055] Different activation/deactivation rules may be used, for example, a security policy may be activated if

IPR2023-01353 Page 00579

the success likelihood reached a given threshold, or if the attack impact reach a predetermined threshold. It can also be a combination of both (activation if the attack impact and the success likelihood reach given thresholds).

[0056] In general, a response policy must be activated when:

- the detected threat violated the security and operational policies and,
- the success likelihood (calculated dynamically, considering the attack progress and the state of the monitored system) of the threat exceeds a predefined threshold and,
- the impact of the threat exceeds a predefined threshold, and is greater than the cost of the associated response policy.

[0057] In the same way, if the success likelihood goes below a given threshold or the response impact goes over a predefined threshold or a combination of both, an activated security policy have to be deactivated. Such case may occur if the impact of a response policy induced more drawbacks than advantages with respect to the initial attack or if the risks associated with the attack have been dismissed or eradicated.

[0058] The order or command of activating or deactivating a security policy is then sent to response policy deployment point (PDP) 107 (also called policy decision point) wherein each security rule of a response policy is converted into scripts to configure the policy enforcement points (PEP) 109 which are located within the information system 1 and which are used to enforce the security policies (firewall activation or configuration, intrusion prevention systems activations, account permissions or access modifications...). The online steps (comprising the online correlation engine 131, the attack objective probability assessment 132, the attack impact assessment, the attack response assessment 136, the policy instantiation engine 134, the policy deployment point and the policy enforcement point) may be achieved by programs or software run by a computer or a server. Said online steps use the stored elements of the data repositories (attack model data repository 125 and response policy data repository 128) to determine dynamically the necessity of activation/deactivation of response security policy in function of the detected attacks and of the state of the monitored information system 1.

[0059] Thus, the present invention allows to provide a dynamic assessment of the impact of an attack as well as the impact of the security policy to apply in response to the attack providing therefore an optimized efficiency of the use of a security policy. Moreover, the idea of determining conditions for deactivating a security policy allows to improve the reactivity of the information system protection and to avoid an unnecessary degraded use of the information system. Indeed, security policy usually correspond to a degraded mode that limits the effects of the attack but also reduces the capacity of said information system and may disturb and affect the users of the information system leading to a loss of productivity or a reduction of quality of service (QoS) for the company using the information system.

Claims

1. Method for adapting security policies of an information system infrastructure in function of attacks wherein it comprises the steps of:

   - storing potential attacks and their associated risks in a data repository;
   - storing curative security policies in response of the potential attacks in a data repository;
   - monitoring entering contents representing data streams of the information system;
   - detecting at least one attack in the information system;
   - assessing a success probability parameter of the at least one detected attack and its associated cost impact parameter;
   - assessing an activation impact parameter of at least one curative security policy in response to the at least one detected attack and its associated cost impact parameter;
   - deciding of the activation or deactivation of a curative security policy in function of the success probability parameter of the, at least one, detected attack, of the activation impact parameter of at least one curative security policy and of the cost impact parameters of both the detected at least one attack and the at least one curative security policy.

2. Method for adapting security policies of an information system infrastructure in accordance with claim 1 wherein the step of storing potential attacks and their associated risks in a data repository comprises the steps of:

   - defining the information system topology and attack detection signatures;
   - defining a risk analysis of the information system that identifies potential attack objectives;
   - specifying attack models to reach the identified attack objectives;
   - storing said attack models in a data repository.

3. Method for adapting security policies of an information system infrastructure in accordance with claim 1 or 2 wherein the step of storing curative security policies in response of the potential attacks in a data repository comprises:

   - specifying at least one attack context;

IPR2023-01353 Page 00580

- specifying curative security policies corresponding to the specified at least one attack context;
- storing said curative security policies in a data repository.

4. Method for adapting security policies of an information system infrastructure in accordance with one of the previous claims wherein the step of assessing a success probability parameter of the at least one detected attack and its associated cost impact parameter comprises:

- generating attack strategy graphs based on the stored attack models and the detected at least one attack;
- assessing the probability for the attack to reach its objective;
- assessing the impact of the attack objective on the system security level and on the system quality of service (QoS) level;
- assessing the associated cost impact parameter of the attack objective;

5. Method for adapting security policies of an information system infrastructure in accordance with one of the previous claims wherein the step of assessing an activation impact parameter of at least one curative security policy in response to the at least one detected attack and its associated cost impact parameter is based on the stored curative security policies and the state of the monitored information system.

6. Method for adapting security policies of an information system infrastructure in accordance with one of the previous claims wherein the step of deciding of the activation or deactivation of a curative security policy in function of the success probability parameter of the, at least one, detected attack, of the activation impact parameter of at least one curative security policy and of the cost impact parameters of both the detected at least one attack and the at least one curative security policy is applied dynamically based on the evolution of the state of the monitored system.

7. Method for adapting security policies of an information system infrastructure in accordance with one of the previous claims wherein the cost impact parameters comprise:

- a quality of service (QoS) impact and,
- a security level degradation impact.

8. Monitoring and protecting equipment comprising:

- at least one data repository for:

- storing potential attacks and their associated risks;
- storing curative security policies in response of the potential attacks;

- processing means for:

- monitoring entering contents representing data streams of the information system;
- detecting at least one attack in the information system;
- assessing a success probability parameter of the detected at least one attack and its associated cost impact parameter;
- assessing at least one activation impact parameter of at least one curative security policy in response to the at least one detected attack and its associated cost impact parameter;
- deciding of the activation of a curative security policy in function of the success probability parameter of the at least one detected attack, of the activation impact parameter of at least one curative security policy and of the cost impact parameters of both the at least one attack and the at least one curative policy,
- activating at least one curative security policy.

9. Monitoring and protecting equipment comprising:

- at least one data repository for:

- storing potential attacks and their associated risks;
- storing curative security policies in response of the potential attacks;

- processing means for:

- monitoring entering contents representing data streams of the information system;
- detecting at least one attack in the information system;
- assessing a success probability parameter of the detected at least one attack and its associated cost impact parameter;
- assessing at least one activation impact parameter of at least one curative security policy in response to the at least one detected attack and its associated cost impact parameter;
- deciding of the deactivation of a curative security policy in function of the success probability parameter of the at least one detected attack, of the activation impact parameter of at least one curative security pol-

IPR2023-01353 Page 00581

icy and of the cost impact parameters of both the at least one attack and the at least one curative policy
- deactivating at least one curative security policy.

IPR2023-01353 Page 00582

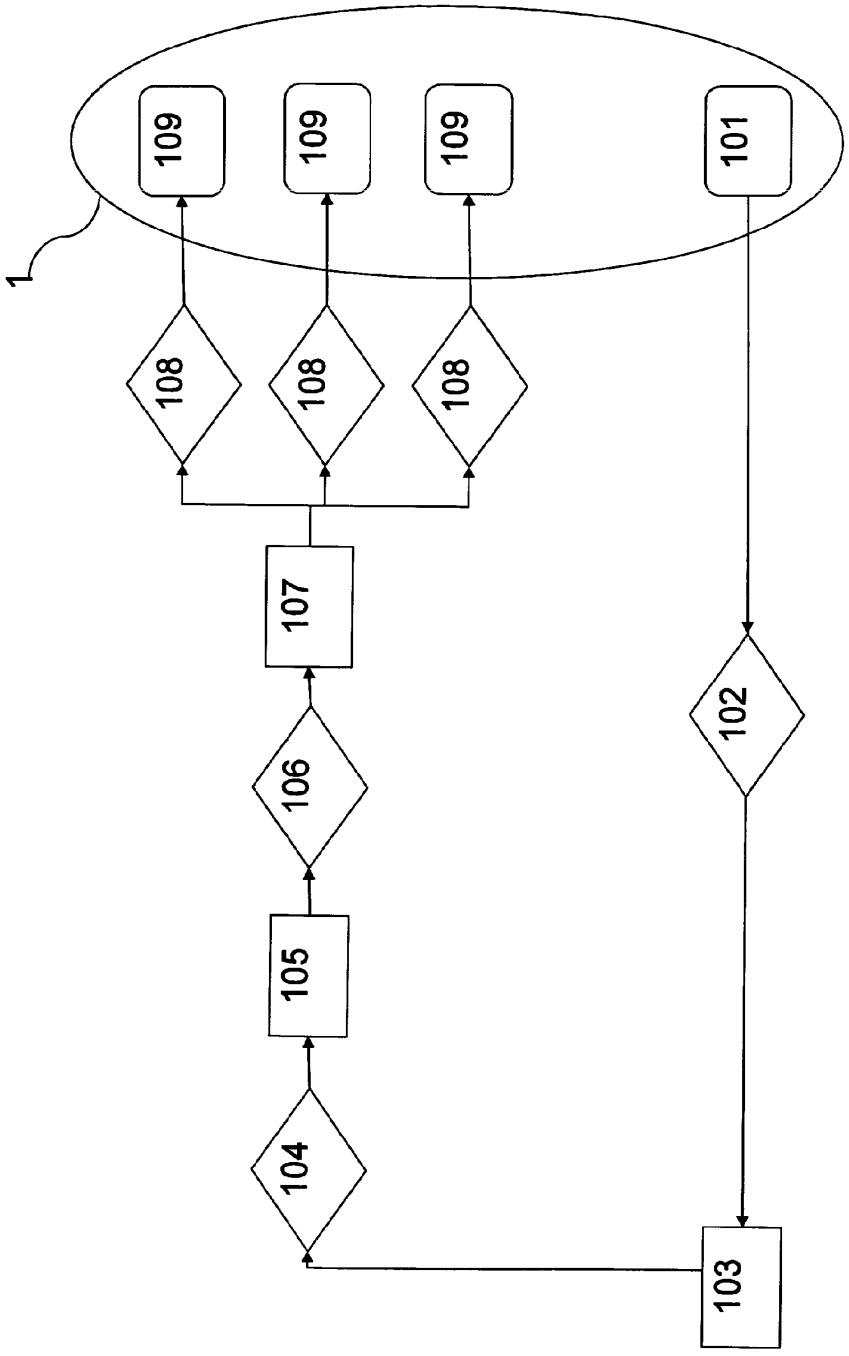


Fig.1

IPR2023-01353 Page 00583

IPR2023-01353 Page 00584

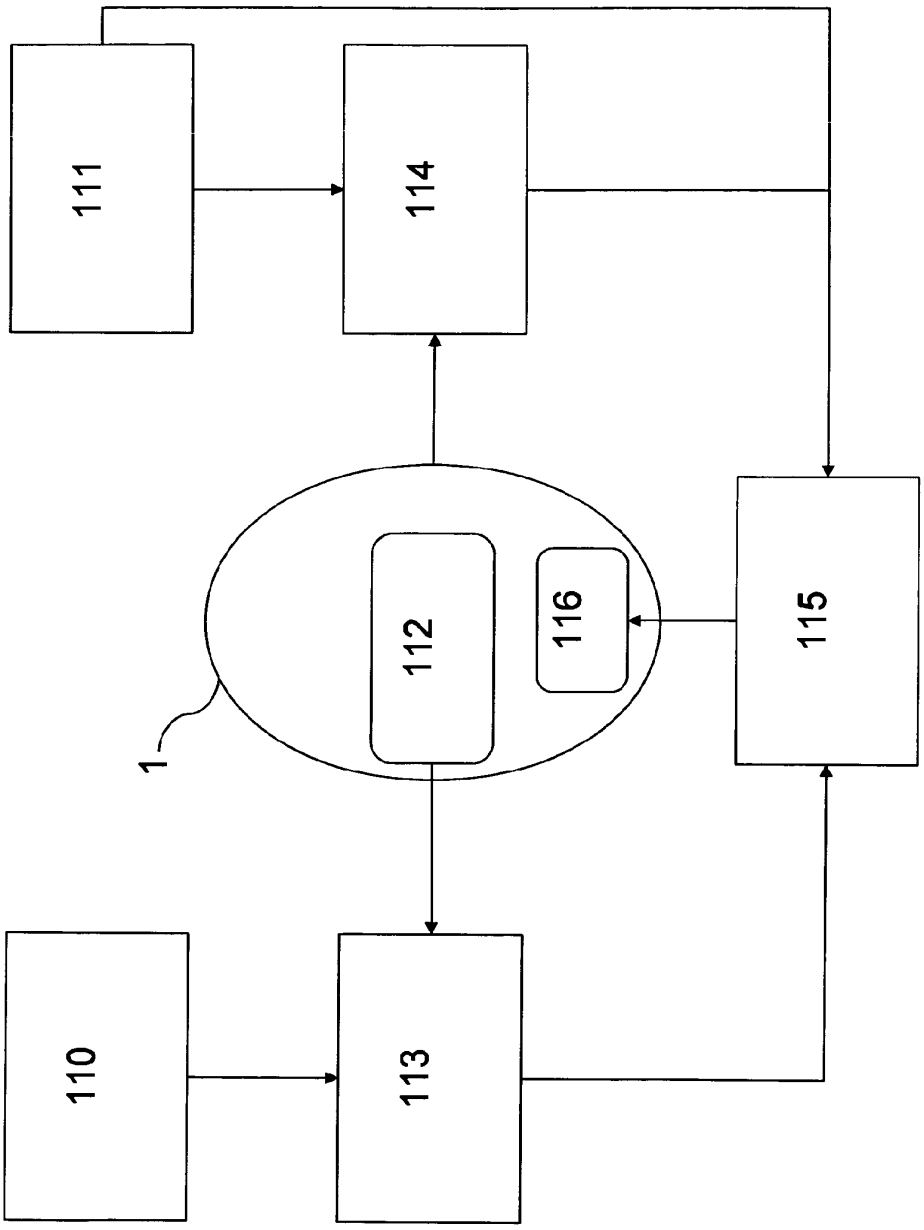


Fig.2

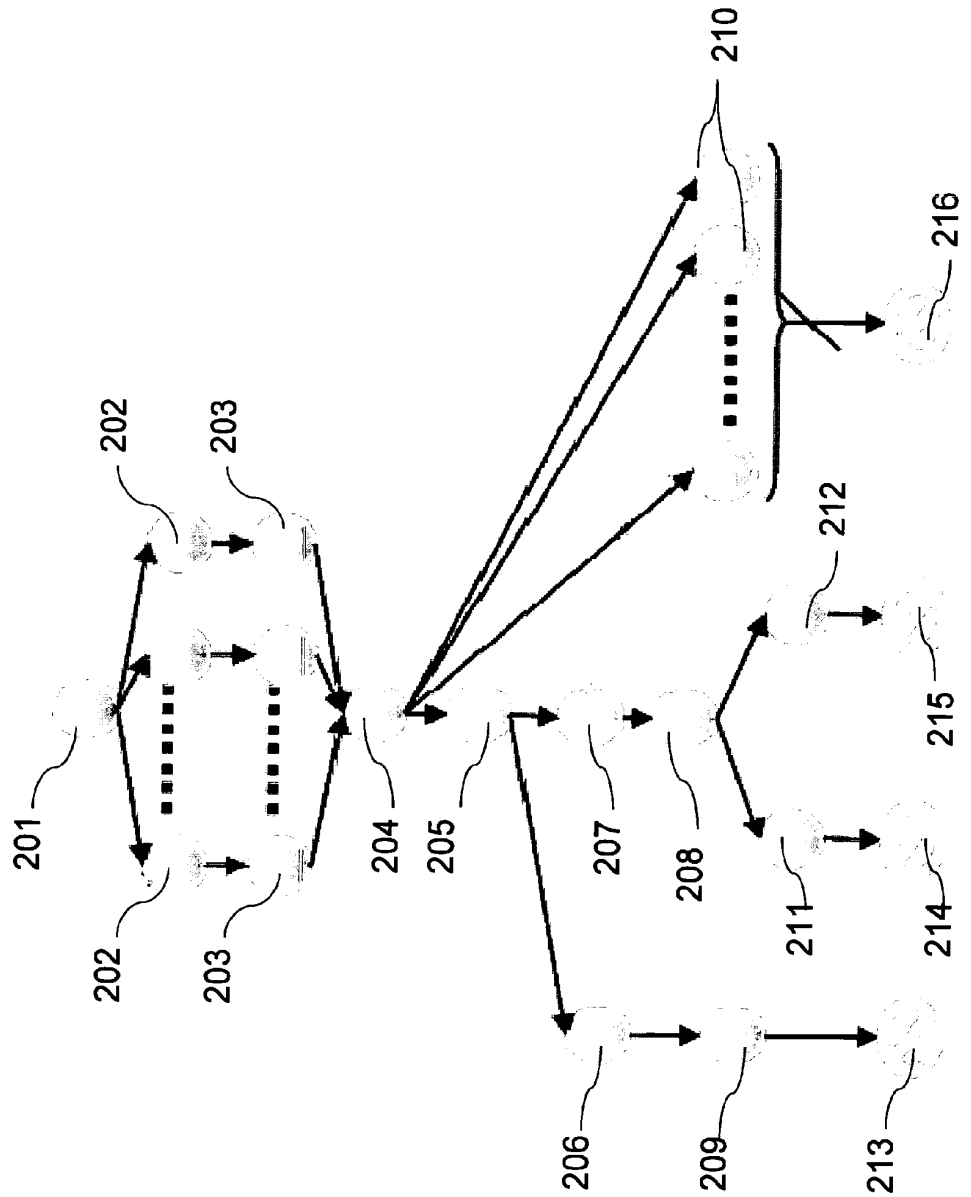


Fig.3

IPR2023-01353 Page 00585

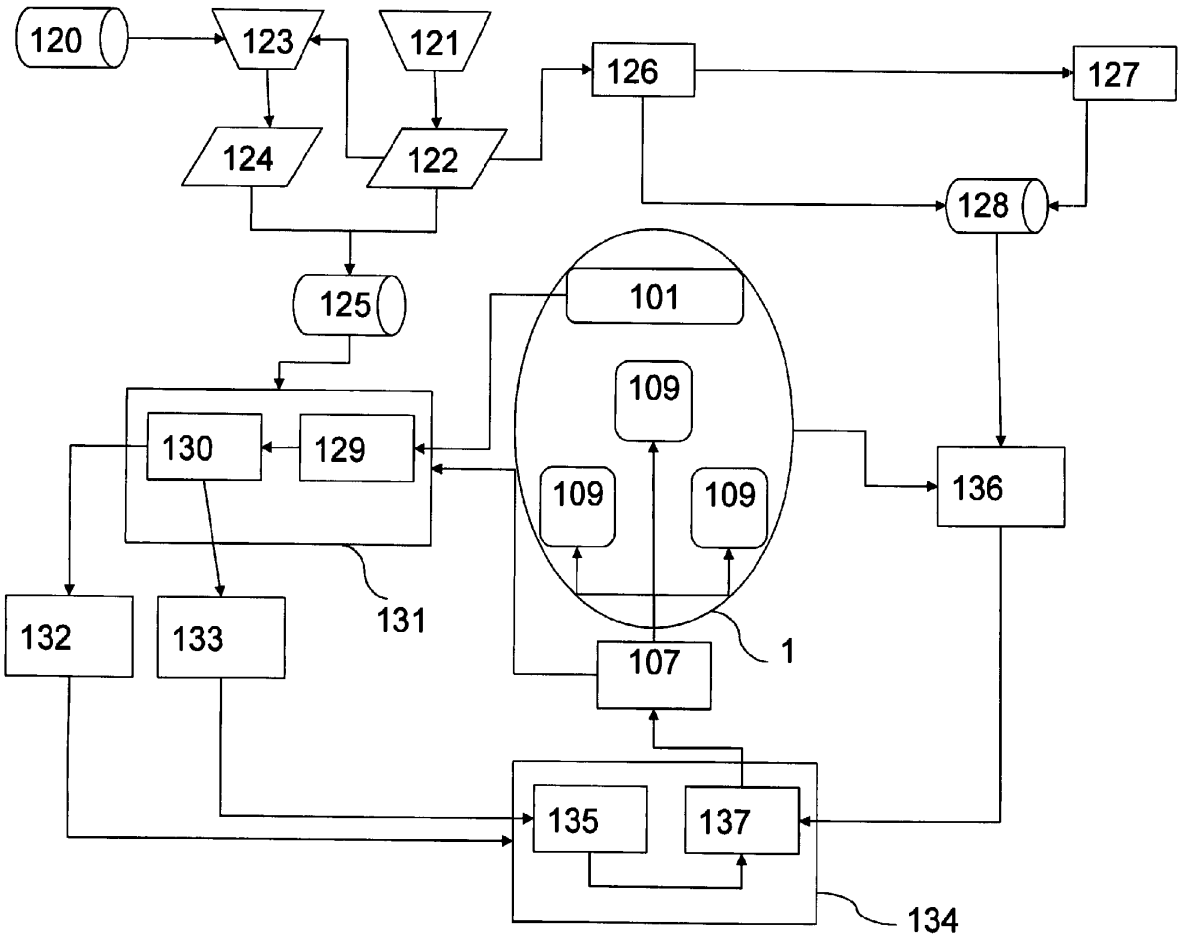

120
123
121
126
127
124
122
128
125
101
109
130
129
109
109
136
131
132
133
1
107
135
137
134

Fig.4

Europäisches Patentamt
European Patent Office
Office européen des brevets

EUROPEAN SEARCH REPORT

Application Number
EP 10 29 0250

DOCUMENTS CONSIDERED TO BE RELEVANT

Category	Citation of document with indication, where appropriate, of relevant passages	Relevant to claim	CLASSIFICATION OF THE APPLICATION (IPC)
X	WO 2004/015908 A2 (ALPHATECH INC [US]; KREIDL O PATRICK [US]; FRAZIER TIFFANY [US]) 19 February 2004 (2004-02-19) * abstract; figures 1-6; tables I, II * * paragraphs [0006], [0008], [0013] * * paragraphs [0021] - [0023] * * paragraphs [0037] - [0039] * * paragraphs [0052], [0053], [0062] * * paragraphs [0067], [0069], [0072] * * paragraphs [0073], [0076] * * paragraphs [0088] - [0110] *	1-9	INV. H04L29/06 G06F21/00
X	KANOUN W ET AL: "Success Likelihood of Ongoing Attacks for Intrusion Detection and Response Systems" COMPUTATIONAL SCIENCE AND ENGINEERING, 2009. CSE '09. INTERNATIONAL CONFERENCE ON, IEEE, PISCATAWAY, NJ, USA, 29 August 2009 (2009-08-29), pages 83-91, XP031543989 ISBN: 978-1-4244-5334-4 * abstract; figures 3,6,7 * * page 83, left-hand column, line 1 - right-hand column, line 37 * * page 84, left-hand column, line 42 - right-hand column, line 2 * * page 84, right-hand column, line 44 - page 85, left-hand column, line 15 * * page 87, left-hand column, line 35 - right-hand column, line 19 *	1-9	TECHNICAL FIELDS SEARCHED (IPC) H04L G06F G06N
A	WO 2007/027131 A2 (TELIASONERA AB [SE]; ENDERSZ VIKTOR [SE]) 8 March 2007 (2007-03-08) * figure 1 * * page 4, line 29 - page 8, line 34 *	1-9	

-/--

The present search report has been drawn up for all claims

Place of search	Date of completion of the search	Examiner
Munich	21 October 2010	Schossmaier, Klaus

CATEGORY OF CITED DOCUMENTS

X : particularly relevant if taken alone
Y : particularly relevant if combined with another document of the same category
A : technological background
O : non-written disclosure
P : intermediate document

T : theory or principle underlying the invention
E : earlier patent document, but published on, or after the filing date
D : document cited in the application
L : document cited for other reasons

& : member of the same patent family, corresponding document

EPO FORM 1503 03.82 (P04C01)

IPR2023-01353 Page 00587

Europäisches Patentamt
European Patent Office
Office européen des brevets

EUROPEAN SEARCH REPORT

Application Number
EP 10 29 0250

DOCUMENTS CONSIDERED TO BE RELEVANT

Category	Citation of document with indication, where appropriate, of relevant passages	Relevant to claim	CLASSIFICATION OF THE APPLICATION (IPC)
A	MIGUEL JOHN: "COMPOSITE COST/BENEFIT/RISK ANALYSIS METHODOLOGY" COMPUTER SECURITY: A GLOBAL CHALLENGE; PROCEEDINGS / PROCEEDINGS OF THE IFIP INTERNATIONAL CONFERENCE ON COMPUTER SECURITY,, 1 January 1984 (1984-01-01), pages 307-311, XP009139953 ISBN: 978-0-444-87618-8 * the whole document * -----	1-9	
			TECHNICAL FIELDS SEARCHED (IPC)

The present search report has been drawn up for all claims

Place of search	Date of completion of the search	Examiner
Munich	21 October 2010	Schossmaier, Klaus

CATEGORY OF CITED DOCUMENTS

X : particularly relevant if taken alone
Y : particularly relevant if combined with another document of the same category
A : technological background
O : non-written disclosure
P : intermediate document

T : theory or principle underlying the invention
E : earlier patent document, but published on, or after the filing date
D : document cited in the application
L : document cited for other reasons

& : member of the same patent family, corresponding document

EPO FORM 1503 03.82 (P04C01)

4

IPR2023-01353 Page 00588

ANNEX TO THE EUROPEAN SEARCH REPORT ON EUROPEAN PATENT APPLICATION NO. EP 10 29 0250

This annex lists the patent family members relating to the patent documents cited in the above-mentioned European search report. The members are as contained in the European Patent Office EDP file on
The European Patent Office is in no way liable for these particulars which are merely given for the purpose of information.

21-10-2010

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
WO 2004015908 A2	19-02-2004	AU 2003278706 A1	25-02-2004
		CA 2495147 A1	19-02-2004
		EP 1535422 A2	01-06-2005
WO 2007027131 A2	08-03-2007	NONE	

EPO FORM P0459

For more details about this annex : see Official Journal of the European Patent Office, No. 12/82

IPR2023-01353 Page 00589

Espacenet

Bibliographic data: EP2498442 (A1) — 2012-09-12

Methods, systems and devices for the detection and prevention of malware within a network

Inventor(s): RIESCHICK GARY [US]; DUNNE CAMERON ROSS [IE]; MCNAMEE ALAN [IE]; HOGAN JOE [IE] ± (RIESCHICK, GARY, ; DUNNE, CAMERON ROSS, ; MCNAMEE, ALAN, ; HOGAN, JOE)

Applicant(s): OPENET TELECOM LTD [IE] ± (OPENET TELECOM LTD)

Classification: **- international:** ***H04L12/24; H04L29/06; H04W12/08; H04W12/12;*** *H04W24/00; H04W88/16*
- cooperative: **H04L63/1441**; **H04W12/08**; **H04W12/12**; H04L63/1408; H04W88/16

Application number: EP20120158921 20120309

Priority number (s): US201161451775P 20110311 ; US201113159710 20110614

Also published as: US2012233656 (A1) US8726376 (B2)

Abstract of EP2498442 (A1)

Methods, systems and devices examine data flows in a communication system control network for known malware threats and suspicious properties typically associated with malware threats. A policy management system inside the control network accesses a user repository and a charging network, and performs pattern matching and/or observed behavior detection methods to determine if the data flows carry content (e.g., malware) that poses a security risk to network or wireless devices. The policy management system generates policy rules based on user preferences and risk-level. The policy management system sends the generated policy rules to a gateway/PCEF, which blocks the data flows, allows the data flows, or restricts the data flow based on the policy rules.

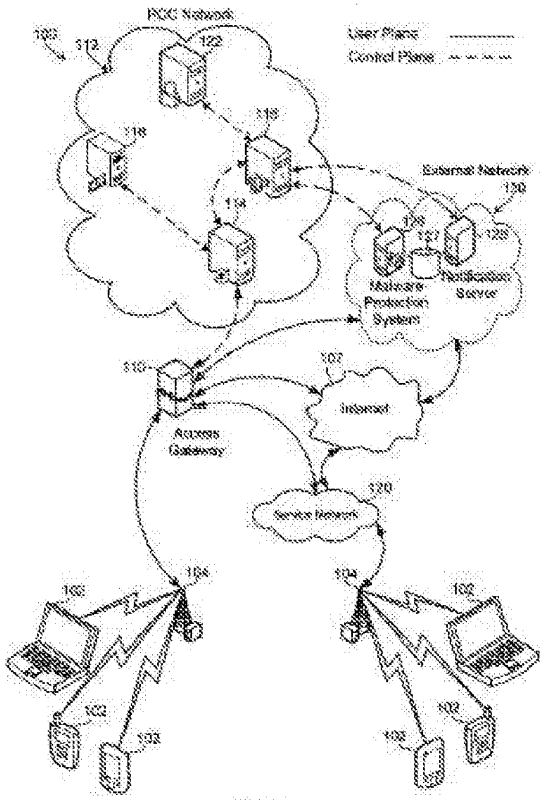

RCC Network
User Plane
Control Plane
External Network
Malware Protection System
Notification Service
Access Gateway
Internet
Service Network

FIG. 2

(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) EP 2 498 442 A1

(12) EUROPEAN PATENT APPLICATION

(43) Date of publication:
12.09.2012 Bulletin 2012/37

(21) Application number: **12158921.2**

(22) Date of filing: **09.03.2012**

(51) Int Cl.:
H04L 12/24 *(2006.01)* ***H04L 29/06*** *(2006.01)*
H04W 12/08 *(2009.01)* ***H04W 12/12*** *(2009.01)*
H04W 24/00 (2009.01) *H04W 88/16 (2009.01)*

(84) Designated Contracting States:
AL AT BE BG CH CY CZ DE DK EE ES FI FR GB GR HR HU IE IS IT LI LT LU LV MC MK MT NL NO PL PT RO RS SE SI SK SM TR
Designated Extension States:
BA ME

(30) Priority: **11.03.2011 US 201161451775 P**
14.06.2011 US 201113159710

(71) Applicant: **Openet Telecom Ltd.**
12 Dublin (IE)

(72) Inventors:
- **Rieschick, Gary**
  Louisburg, KS Kansas KS 66053 (US)
- **Dunne, Cameron Ross**
  County Kildare (IE)
- **McNamee, Alan**
  Dublin, 9 (IE)
- **Hogan, Joe**
  County Dublin (IE)

(74) Representative: **Kelly, Madeleine**
FRKelly
27 Clyde Road
Ballsbridge
Dublin 4 (IE)

(54) Methods, systems and devices for the detection and prevention of malware within a network

(57) Methods, systems and devices examine data flows in a communication system control network for known malware threats and suspicious properties typically associated with malware threats. A policy management system inside the control network accesses a user repository and a charging network, and performs pattern matching and/or observed behavior detection methods to determine if the data flows carry content (e.g., malware) that poses a security risk to network or wireless devices. The policy management system generates policy rules based on user preferences and risk-level. The policy management system sends the generated policy rules to a gateway/PCEF, which blocks the data flows, allows the data flows, or restricts the data flow based on the policy rules.

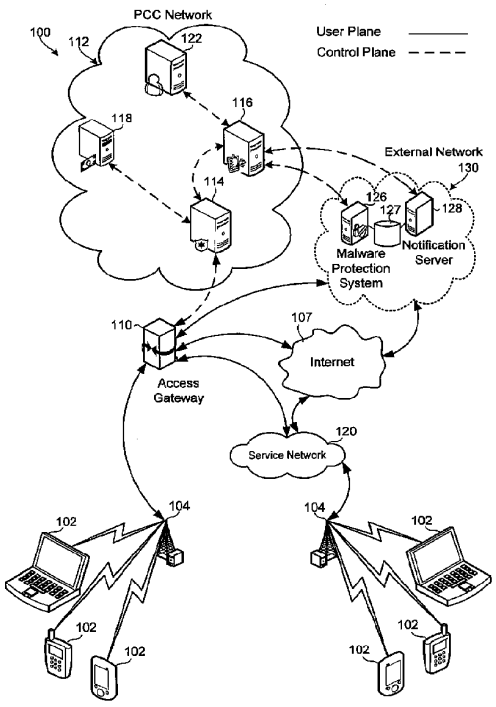


FIG. 2

EP 2 498 442 A1

Description

BACKGROUND

[0001] Recent advances in wireless and mobile technologies enable mobile-device users to access the Internet at any time and in any place. Internet usage habits are changing and many new types of Internet-enabled devices are emerging to meet the evolving user demands and changing Internet-usage habits. Many of these new and emerging Internet-enabled wireless devices use custom and/or non-traditional operating systems and access the Internet through mobile data networks. Further, many of these devices include powerful processors capable of executing third-party application software downloaded from digital application distribution platforms (e.g., Apple Inc.'s App Store, etc.). The emergence these new technologies and wireless devices presents additional security challenges for the telecommunications networks, mobile data networks and to the new wireless devices themselves.

[0002] Today, malicious software (herein "malware") may be used by thieves and hackers to steal confidential information, launch attacks on web servers and to flood networks with spam. Recently, there have been many high-profile cases of confidential customer information being stolen from businesses, and private wireless device users being subjected to online financial and identity thefts. Perhaps even more significantly, there has recently been an increase in wireless devices being infected by malware that forces the wireless device to participate in malicious activities without the user's knowledge or consent, causing users to incur higher usage-charges and leaving them vulnerable to potential legal actions. Recent press reports indicate this problem is increasing at an alarming rate. The emergence of these new types of Internet-enabled devices is likely to exacerbate these problems, as current malware detection and prevention techniques are ineffective at preventing or removing malware from wireless devices, especially the new breed of Internet-enabled wireless devices that have custom and/or non-traditional operating systems.

[0003] Further, the growth in popularity of application distribution platforms (herein "app ecosystems") allows wireless device users to more readily download third party software and execute the downloaded software with full administrative privileges on the mobile device. Such downloaded applications may be sourced from many different sources, such as email attachments, instant messaging systems, and device-to-device transfers. The emergence of app ecosystems presents additional security challenges that are distinct from the traditional security challenges presented by a malicious page being displayed within web browsers which could be managed by implementing a sandbox model.

[0004] Current malware detection and prevention techniques typically involve installing a protective software client application onto a traditional Internet-enabled device, such as a desktop computer or a laptop computer. Thus, current techniques aim to prevent malware by protecting user-level devices (e.g., laptops) rather than providing a comprehensive solution that can eliminate malware from within the network. Further, since many of the new and emerging Internet-enabled wireless devices do not have traditional operating systems, they cannot effectively employ existing malware detection and prevention techniques. As these new Internet-enabled devices grow in popularity, so will the importance of detecting and preventing malware from within the network.

SUMMARY

[0005] The various embodiments provide methods, systems and devices for receiving from one or more communication system control network components data flow parameters identifying characteristics of a data flow, performing a detection method to determine if the data flow carries malicious content, generating restrictive policy rules in response to determining the data flow carries malicious content, generating non-restrictive policy rules in response to determining the data flow does not carry malicious content, and pushing the generated policy rules to a communication system gateway configured to control the flow of packet-based data between a wireless device and an external network. In various embodiments, generating restrictive policy rules in response to determining the data flow carries malicious content includes generating restrictive policy rules that instruct the gateway to block the data flow and/or to block future data flows originating from the wireless device. The data flow may originate from a wireless device or an Internet server.

[0006] Various embodiment methods, systems and devices may perform charging operations that charge subscriber accounts for data flows allowed by the gateway and not charge subscriber accounts for the data flows blocked by the gateway.

[0007] Pattern matching detection methods may be performed in embodiment methods, systems and devices to determine if the data flow carries content that is a security risk. In an embodiment, pattern matching detection methods may be performed by comparing identified binary-file characteristic information with a binary-file characteristic of a known malware type to determine if the data flow carries content having binary-file characteristics of known malware, which may include comparing the identified binary-file characteristic information with a binary-file characteristic of a spam-sending Internet bot.

[0008] The embodiments include servers configured to implement the various methods through software programming, systems implementing such servers, and non-transitory computer-readable storage media on which are stored server-executable instructions implementing the various methods.

IPR2023-01353 Page 00593

BRIEF DESCRIPTION OF THE DRAWINGS

[0009] The accompanying drawings, which are incorporated herein and constitute part of this specification, illustrate exemplary aspects of the invention, and, together with the general description given above and the detailed description given below, serve to explain features of the invention.

[0010] FIG. 1 is a communication system block diagram illustrating network components of a communication system having a third generation partnership project (3GPP), evolved packet core (EPC) architecture suitable for use in the various embodiments.

[0011] FIG. 2 is a communication system block diagram illustrating network components and of an example communication-system having a policy and charging control (PCC) network in accordance with the various embodiments.

[0012] FIG. 3A is a communication system block diagram illustrating example communications between logical components in an example control network and an external network.

[0013] FIG. 3B is an communication system block diagram illustrating example communications between a malware protection system and external networks in accordance with various embodiments.

[0014] FIG. 4A is a process flow diagram illustrating an example method for receiving signaling flows and enforcing policy rules on the signaling flows in accordance with various embodiments.

[0015] FIG. 4B is a process flow diagram illustrating an example policy management system method for receiving parameters from a gateway and generating rules in accordance with the various embodiments.

[0016] FIG. 5 is a process flow diagram illustrating an example pattern matching detection method in accordance with an embodiment.

[0017] FIGs. 6A and 6B are process flow diagrams illustrating example observed behavior detection methods in accordance with various embodiments.

[0018] FIG. 7 is a component block diagram of a server suitable for use with an embodiment.

DETAILED DESCRIPTION

[0019] The various embodiments will be described in detail with reference to the accompanying drawings. Wherever possible, the same reference numbers will be used throughout the drawings to refer to the same or like parts. References made to particular examples and implementations are for illustrative purposes, and are not intended to limit the scope of the invention or the claims.

[0020] The word "exemplary" is used herein to mean "serving as an example, instance, or illustration." Any implementation described herein as "exemplary" is not necessarily to be construed as preferred or advantageous over other implementations.

[0021] As used herein, the terms "mobile device," "wireless device" and "user equipment (UE)" may be used interchangeably and refer to any one of various cellular telephones, smart-phones (e.g., iPhone®), personal data assistants (PDA's), palm-top computers, laptop computers, wireless electronic mail receivers (e.g., the Blackberry® and Treo® devices), VoIP phones, multimedia/Internet enabled cellular telephones, and similar electronic devices capable of sending and receiving wireless communication signals. A wireless device may include a programmable processor and memory.

[0022] As used herein, the term "bot" is used to refer to software applications that emulate human online-activity by automatically performing tasks typically performed by human computer users.

[0023] As used herein, the phrase "malware protection system" is used to refer to any system configured to detect, prevent and/or remove malware such as computer viruses, computer worms, Trojan horses, bots, spyware, adware, etc. Thus, a malware protection system may include a malware prevention system, a malware detection system, a malware removal system, or any other system that assists the detection, prevention and/or removal of malware, as well as combinations of such systems.

[0024] A number of different cellular and mobile communication services and standards are available or contemplated in the future, all of which may implement and benefit from the various embodiments. Such services and standards include, e.g., 3rd generation partnership project (3GPP), long term evolution (LTE) systems, 3rd generation wireless mobile communication technology (3G), 4th generation wireless mobile communication technology (4G), global system for mobile communications (GSM), universal mobile telecommunications system (UMTS), 3GSM, general packet radio service (GPRS), code division multiple access (CDMA) systems (e.g., cdmaone, CDMA2000TM), worldwide interoperability for microwave access (WiMAX), enhanced data rates for GSM evolution (EDGE), advanced mobile phone system (AMPS), digital AMPS (IS-136/TDMA), evolution-data optimized (EV-DO), digital enhanced cordless telecommunications (DECT), and integrated digital enhanced network (iden). Each of these technologies involves the transmission and reception of signaling and content messages. It should be understood that any references to terminology and/or technical details related to an individual standard or technology are for illustrative purposes only and are not intended to limit the scope of the claims to a particular broadcast communication system or technology unless specifically recited in the claim language.

[0025] As discussed above, Internet usage habits are changing and many new types of Internet-enabled devices are emerging to meet evolving user demands. The emergence of these devices presents new security challenges that cannot be addressed by existing antivirus/malware detection and removal systems. For example, current malware detection and prevention techniques attempt to prevent malware by protecting user-level devic-

IPR2023-01353 Page 00594

es (e.g., laptops) rather than from within the network. Since many new and emerging Internet-enabled devices do not having traditional operating systems, current malware detection and prevention techniques cannot efficiently detect or prevent malware.

[0026] The various embodiments provide a comprehensive security solution for detecting and preventing malware from within a mobile data network. The various embodiments protect mobile devices from becoming infected with malicious software (malware) and protect Internet servers and network components from attacks (e.g., denial of service attacks) and inconveniences (e.g., spam) caused by malware. The various embodiments detect both known and yet-to-be identified security threats by performing pattern matching and observed behavior detection methods. Various embodiments monitor network traffic for suspicious content or activities to detect malware and block suspicious data flows from within the network.

[0027] Various embodiments provide methods for detecting and preventing malware from within a data network by combining a policy management component, which has a global or near global view of network traffic and transactions, with a malware system detection component configured to recognize malware based upon patterns within the network traffic and transactions. The various embodiments may operate exclusively within the network and support any type of Internet device.

[0028] The various embodiments may detect transmissions of known and unknown malware in real time as the network patterns emerge. Various embodiments may enable defensive and ameliorative actions, such as modifying the transmissions of the malware or transmission generated by malware, which may include throttling malware transmissions. Various embodiments may prevent the transmission of malware once it has been detected. Various embodiments may alert wireless device users of detected malware. Various embodiments may alert telecommunications network operators of detected malware so they can take corrective actions and/or warn targets of an attack.

The various embodiments may include machine learning mechanisms to enable the malware protection system to learn about malware-generated transmissions and recognize malware patterns as they develop. Such machine learning capabilities may include tracking or correlating network activity levels of recognized threats as well as details of suspicious activities not yet recognized as threatening. Various embodiments may store user-specific malware detection (e.g., patterns in network traffic that occurred while a malware event was underway) and prevention information (e.g., countermeasures that proved effective in responding to a malware event), which may be used by the malware protection system for the detection and prevention of future malware events. Various embodiments may map specific malware threats identified to users of the network. Malware threats may be identified using any suitable mechanism, including the use of IP addresses.

[0029] The various embodiments may push client software capable of exchanging malware and security information with a control network to wireless devices. The malware and security information may include report details generated by the client, alerts generated by the system, instructions from the system to the client that relate to a specific threat to the client, and/or instructions that instruct the wireless device to change its behavior. In an embodiment, the malware and security information may include instructions that instruct a first application executing on the wireless device to cause a second application executing on the wireless device to change its behavior.

[0030] The various embodiments may detect spam by recognizing network traffic with certain characteristics, which may include destination information, protocols, and/or payload content. The various embodiments may protect networks by preventing wireless devices from being used as a source of malware attacks. Various embodiments may detect denial of service attacks by recognizing rapid increases in traffic addressed to a single or related destination from multiple sources. Various embodiments may prevent spam and denial of service attacks by throttling or blocking suspected malicious traffic.

[0031] The various embodiments may reduce the need for network operators to augment capacity and provide network congestion relief. The various embodiments may send detection and prevention information to telecommunications network operators, wireless devices and/or malware systems in real time.

[0032] The various embodiments may be implemented within a variety of mobile communication systems, an example of which is illustrated in FIG. 1. Wireless devices 102 may be configured to send and receive voice, data and control signals to and from a service network 36 (and ultimately the Internet) using a variety of communication systems/technologies (e.g., GPRS, UMTS, LTE, cdmaOne, CDMA2000™, etc). For example, general packet radio service (GPRS) data transmitted from a wireless device 102 is received by a base transceiver station (BTS) 12 and sent to a base station controller (BSC) and/or packet control unit (PCU) component (BSC/PCU) 18. Code division multiple access (CDMA) data transmitted from a wireless device 102 is received by a base transceiver station 12 and sent to a base station controller (BSC) and/or point coordination function (PCF) component (BSC/PCF) 22. Universal mobile telecommunications system (UMTS) data transmitted from a wireless device 102 is received by a NodeB 14 and sent to a radio network controller (RNC) 20. Long term evolution (LTE) data transmitted from a wireless device 102 is received by an eNodeB 16 and sent directly to a serving gateway (SGW) 28 located within the EPC 40.

[0033] The BSC/PCU 18, RNC 20 and BSC/PCF 22 components process the GPRS, UMTS and CDMS data, respectively, and send the processed data to a node within the EPC 40. More specifically, the BSC/PCU 18 and

IPR2023-01353 Page 00595

RNC 20 units send the processed data to a serving GPRS support node (SGSN) 24 and the BSC/PCF 22 sends the processed data to a packet data serving node (PDSN) and/or high rate packet data serving gateway (HSGW) component (PDSN/HSGW) 30. The PDSN/HSGW 30 may act as a connection point between the radio access network and the IP based PCEF/PGW 32. The SGSN 24 is responsible for routing the data within a particular geographical service area and may send signaling (control plane) information (e.g., information pertaining to call set-up, security, authentication, etc.) to a mobility management entity (MME) 26. The MME may request user and subscription information from a home subscriber server (HSS) 42, perform various administrative tasks (e.g., user authentication, enforcement of roaming restrictions, etc), select a SGW 28 and send administrative and/or authorization information to the SGSN 24.

[0034] Upon receiving the authorization information from the MME 26 (e.g., authentication complete, identifier of a selected SGW, etc.), the SGSN 24 sends the GPRS/ UMTS data to a selected SGW 28. The SGW 28 stores information about the data (e.g., parameters of the IP bearer service, network internal routing information, etc.) and forwards user data packets to a policy control enforcement function (PCEF) and/or packet data network gateway (PGW) 32. The PCEF/ PGW 32 sends signaling information (control plane) to a policy control rules function 34. The PCRF 34 accesses subscriber databases, creates a set of intelligent policy rules and performs other specialized functions (e.g., interacts with online/offline charging systems, application functions, etc.) and sends the intelligent policy rules to the PCEF/ PGW 32 for enforcement. The PCEF/ PGW 32 implements the policy rules to control the bandwidth, the quality of service and the characteristics of the data, services and flows being communicated between the service network 36 and the end users.

[0035] FIG. 2 illustrates various logical components in an example communication system 100 suitable for implementing the various embodiments. Wireless devices 102 may be configured to communicate via cellular telephone network, a radio access network (e.g., UTRAN, RAN, etc.), WiFi network, WiMAX network, and/or other well known technologies (e.g., GPRS, UMTS, LTE, cdmaOne, CDMA2000TM). Wireless devices 102 may be configured to transmit and receive voice, data and control signals to and from a base station (e.g., base transceiver station, NodeB, eNodeB, etc.) 104, which may be coupled to a controller (e.g., cellular base station, radio network controller, service gateway, etc.) operable to communicate the voice, data, and control signals between mobile devices and to other network destinations. The base station 104 may communicate with an access gateway 110, which serves as the primary point of entry and exit of wireless device traffic and connects the wireless devices 102 to their immediate service provider and/or packet data networks (PDNs). The access gateway 110 may be a part of the policy control and charging network 112 or external to the policy control and charging network 112, and may include one or more of a controller, a gateway, a serving gateway (SGW), a packet data network gateway (PGW), an evolved packet data gateway (ePDG), a packet data serving node (PDSN), a serving general packet radio service support node (SGSN), a policy and charging enforcement function (PCEF), or any combination of the features/functions provided thereof. The access gateway 110 may be implemented in a single computing device or in many computing devices, either within a single network or across a wide area network, such as the Internet.

[0036] The access gateway 110 may forward the voice, data, and control signals to other network components as user data packets, provide connectivity to external packet data networks, manage and store contexts (e.g. network internal routing information, etc.) and act as an anchor between different technologies (e.g., 3GPP and non-3GPP systems). The access gateway 110 may also coordinate the transmission and reception of data to and from the Internet 107, and the transmission and reception of voice, data and control information to and from an external service network 120 connected to the Internet 107 and other base stations 104 and wireless devices 102. The access gateway 110 may also route control/signaling information to a policy and charging control (PCC) network 112, which may be a part of an implementation of an evolved packet core (EPC)/long term evolution (LTE) architecture.

[0037] The PCC network 112 may include a gateway and/or policy and charging enforcement function (PCEF) 114 component for controlling the flow of Internet protocol (IP) data between mobile devices 102 and external networks (e.g., Internet 107, service network 120, etc.). The gateway/PCEF 114 may contain a deep packet inspection (DPI) component (not illustrated) configured to obtain additional application-level information about a particular IP data flow and provide the information to other components in the PCC network 112. In various embodiments, the gateway/PCEF 114 may be integrated with, or a part of, the access gateway 110, either within the PCC network 112 or outside of the PCC network 112.

[0038] The policy and charging control network 112 may also include a policy management system 116, a charging system 118 and a user repository 122. The policy management system 116 may communicate with a malware protection system 126 and a notification server 128 in a network (External Network) 130 outside the telecommunications provider network. The notification server 128 may be configured to receive requests from the policy management system 116 (e.g., PCRF) and send messages to mobile devices 102 using various technologies/mechanisms, such as email, short message service (SMS) texts, instant messaging (IM) systems, interactive voice response (IVR) systems. The notification server 128 may also send messages in a mobile device-specific format, which may be unique to each mobile device type. The notification server 128 may choose

IPR2023-01353 Page 00596

a communications technology/mechanism on a case-by-case basis. For example, the notification server 128 may choose a communications technology/mechanism depending on the user's preferences and/or mobile device type. Messages may be sent through the Internet 107 or the access gateway 110. In an embodiment, the notification server 128 may send messages to a client application running on the mobile devices 102 (e.g., via the Internet 107, access gateway 110, etc.), which may send a response to the notification server 128 for forwarding to the policy management system 116.

[0039] In an embodiment, the notification server 128 may be configured to send messages to mobile devices 102 "in-band" or "out-of-band." The in-band notifications may use the same communication mechanism used for receiving message and the out-of-band notifications may use a different communication mechanism than that used for receiving message. For example, if a user is browsing web pages and receives an in-band notification, then the notification may appear as a new web page.

[0040] The malware protection system 126 may be a dedicated malware system or server, which is sometime referred to herein as a "malware expert system." The malware protection system 126 may be implemented within one or a number of dedicated servers located within the communication network 100, or within one or a number of dedicated servers located external from but with communication links to the communication network 100 (as illustrated). In a further embodiment, some or all elements of the malware protection system 126 functionality may be implemented within a server performing another network function, such as the policy management system 116, a charging system 118 or a user repository 122.

[0041] The malware protection system 126 may maintain a repository of known security risks, gather real-time information regarding emerging security/malware threats, identify security/malware threats by monitoring traffic within the communication network 100, and generate security/malware threat counter-measures. Such security/malware threat counter-measures may include malware removal software and/or malware repair software. In an embodiment, the malware protection system 126 may be configured to generate security/malware threat counter-measures, malware removal software and/or malware repair software in a format specific to each mobile device 102 type. The mobile device formats may be defined by configuration or determined in real time by the policy control system 116, or by the malware protection system 126 based on information received in real time from the policy control system 116.

[0042] The malware protection system 126 may also include a local malware prevention memory or database 127 for each policy management system 116. Likewise, the policy management system 116 may include a local policy management memory (not illustrated), which may be populated by the policy management system 116, the malware protection system 126, or both. In an embodiment, the malware protection system 126 may push data to the policy management memory of the policy management system 116. In an embodiment, the policy management system 116 may periodically request information from the malware protection system 126 and populate the policy management memory with the received information.

[0043] FIG. 3 A is a system architecture diagram illustrating communications between logical components in an example control network 150 (e.g., PCC network) and an external network 155. FIG. 3 illustrates that the control network 150 may include one or more components for providing a policy and charging enforcement function (PCEF) 114, a policy management system 116, a charging system 118, and a user repository 122. The gateway/PCEF 114 may be a part of, or perform operations typically associated with, a gateway GPRS support node (GGSN), a packet data network gateway (PGW), an access gateway 110, or other similar components. In an embodiment, the policy management system 116 may include a policy and charging rules function (PCRF). In an embodiment, the charging system 118 may include an off-line charging system (OFCS) and an on-line charging system (OCS). In an embodiment, the user repository 122 may include a subscriber profile repository (SPR).

[0044] The gateway/PCEF 114 may serve as the primary enforcement point, as a gateway and as a router providing a routing mechanism between the Internet and the radio infrastructure/radio access network. The gateway/PCEF 114 may receive signaling flows and use information contained therein to select an optimal route for a particular type of communication, session, or quality of service (QoS). The gateway/PCEF 114 may also enforce various policies on the received flows, which may include querying, coordinating and/or adjusting various network resources based on a set of policy rules provided by the policy management system 116.

[0045] The policy management system 116 may identify appropriate policy rules for a given situation and send the identified policy rules to the gateway/PCEF 114 for enforcement. Each policy rule may govern the services, quality of service (QoS) and/or bandwidth that are made available to a particular subscriber in a particular data call and/or session. The policy rules may also govern when certain services are made accessible to the subscriber (e.g. weekdays from 9AM to 5PM, etc.) and how long the subscriber may access to those services (e.g., 15 minutes at a time, a total of two hours, etc.). The policy rules may further govern whether specific IP data flows should be "blocked," "allowed" or "allowed with constraints," based on current and potential security risks/malware threats, as discussed in further detail below.

[0046] In an embodiment, the policy management system 116 may generate, compile and select a set of business, technology and security rules that define the policies that are to be enforced. In doing so, the policy management system 116 may communicate with the user repository 122 and/or the charging system 118. The user

IPR2023-01353 Page 00597

repository 112 may store subscriber information (e.g., customer IDs, preferences, subscription levels, etc.) and user-specific information, such as details of a user's subscription to malware prevention services and historical details of a user's exposure to specific malware threats. The charging system 118 may store information pertaining to how much wireless data is being used by a wireless subscriber and information pertaining to whether the subscriber has sufficient funds/credits/access units to receive a requested service/flow. The policy management system 116 may use the information stored by the user repository 122 and charging system 118 to identify potential security risks/malware threats and user preferences for responding to identified security risks/malware threats and make rule decisions based on a combination of the risk/threat and user preference. The policy management system 116 may make the rule decisions on a per-subscriber, per-session and/or per-service basis. For example, the policy management system 116 may use subscriber information (e.g., subscriber's city of residence), the current usage parameters (e.g., day of week, time of day, peak hours, etc.) and the subscriber's service level (e.g., Gold, Bronze, etc.) in conjunction with identified security risks (e.g., known virus, irregular usage patterns, etc.) and user preferences to generate and/or select one or more rules or rule groups, which are pushed to the gateway/PCEF 114 for enforcement.

[0047] The policy management system 116 may also communicate with external systems 155, such as a malware protection system 126 and a notification server 128. As mentioned above, the malware protection system 126 may maintain a repository of known malware threats, gather information regarding emerging malware threats, identify current and potential malware threats and generate malware threat counter-measures, malware removal software and/or malware repair software to neutralize the security threats. The policy management system 116 may send information to the malware protection system 126 to aid in the malware prevention operations. Likewise, the malware protection system 126 may send security related information to the policy management system 116 to aid in the rule making decisions/operations.

[0048] FIG. 3B is a system architecture diagram illustrating communications between policy management systems and malware protection systems. Specifically, FIG. 3B illustrates that the malware protection system 126 may receive information from the policy management system 116, and share information with other policy management systems (e.g., policy management systems 166) or external antivirus/malware protection systems (e.g., malware detection system 162).

[0049] In an embodiment, the policy management system 116 may provide malware detection feedback to the malware protection system 126. In an embodiment, the malware detection feedback may include detection rate information extracted from the IP data flows. The detection rate information may identify the detection rates of known malware threats and include additional parameters (e.g., wireless device type, location of a wireless device, etc.) readily accessible to components within the control network (e.g., policy management system 116, gateway/PCEF 114, etc.). These communication system control network components are generally unavailable to external services and servers (e.g., antivirus servers, malware detection systems, etc.). Such additional parameters may be carried by the IP data flows, extracted by the gateway/PCEF 114 and/or the policy management system 116 and sent to the malware protection system 126. The malware protection system 126 may then use this information to update its malware protection services (e.g., to update lists, generate new prevention software, etc.).

[0050] In an embodiment, the policy management system 116 may send the malware protection system 126 information relating to IP data flows with suspicious characteristics. This information may be used by the malware protection system 126 to identify emerging and unidentified malware threats that are not yet prevalent enough to be known to the malware protection system 126. The malware protection system 126 may also use this information to develop preemptive or preventative counter-measures for the identified malware threats.

[0051] In an embodiment, the policy management system 116 may send malware protection system sample data extracted from IP traffic flows within the detection feedback, which the malware protection system 126 may use for analyzing the malware threats.

[0052] In an embodiment, the malware protection system 126 can subsequently share information received from the policy management system 116 with other policy management systems (e.g., policy management systems 166) in other mobile data networks (e.g., roaming/home networks, external networks, etc.) and/or with independent malware protection systems (e.g., malware detection system 162) operating in private data networks, such as corporate networks, etc.

[0053] FIG. 4A illustrates an example gateway/PCEF method 400 for receiving signaling flows and enforcing policy rules on the signaling flows in accordance with the various embodiments. This embodiment method 400 may be implemented in a server configured with server-executable instructions configured to cause the server to perform the operations of the method steps. In step 402, the gateway/PCEF may receive a data flow originating from, or destined for, a wireless device. In step 406, the gateway/PCEF may examine the data flows and/or the network traffic, identify key characteristics of the data flows and/or network traffic, generate parameters based on the key characteristics and send the generated parameters to a policy managements system for analysis. In step 408, the gateway/PCEF may receive one or more policy rules from the policy management system. In step 410, the gateway/PCEF may use the received policy rules to enforce the various policies discussed above, such as blocking, restricting, or allowing

IPR2023-01353 Page 00598

the data flow to be sent to, or received from, a wireless device.

[0054] FIG. 4B illustrates an example policy management system method 450 for receiving parameters from a gateway/PCEF and generating/selecting rules in accordance with the various embodiments. In step 452, the policy management system may receive parameters generated by the gateway/PCEF. In step 454, the policy management system may use the received parameters to determine if the data flows carry content (e.g., malware) that poses a security risk to network and/or wireless devices. For example, the policy management system may use the parameters to perform pattern matching detection methods or observed behavior detection methods to determine if the data flow carries malicious content or is otherwise a security risk. The policy management system may retrieve information from a local memory and/or request/receive information from external memories/systems/servers (e.g., user repository, charging system, a malware protection system, etc.) that may be used to determine if the data flows carry content (e.g., malware) associated with a potential or known security risk. In step 456, the policy management system may generate/select policy rules based on determining the data flows carry content that poses a "known risk," a "potential risk," or "no risk." In step 458, the policy management system may send the generated/selected policy rules to the gateway/PCEF for enforcement.

[0055] As mentioned above, the policy management system may retrieve information from a local memory as well as from external memories/systems/servers. In an embodiment, the policy management system may first access the local memory before requesting information from the external memories/systems/servers to reduce latency and improve efficiency. In an embodiment, if the local memory identifies the data flow as carrying a potential security risk, the policy management system may forgo accessing the external memories/systems/servers and generate rules based on the information in the local memory. This reduces latency and improves efficiency because fewer operations and external communications are required to make intelligent policy rule decisions.

[0056] FIG. 5 illustrates an example pattern matching detection method 500, which may be performed by the policy management system to determine if an IP data flow carries content (e.g., malware) that poses a security risk. In step 502, the policy management system may receive parameters identifying certain key characteristics of the IP data flows and/or network traffic. In step 504, the policy management system may examine the parameters to identify/extract information that may be used to determine if the data flow carries content that poses an unacceptable security risk to a wireless device, the network, or both. For example, in step 504, the policy management system may extract binary-file characteristic information, uniform resource locator (URL) information (e.g., from the HTTP-headers of IP packets), IP address information and/or simple mail transfer protocol (SMTP) information (e.g., from SMTP headers/bodies of outgoing emails) from the received parameters.

[0057] In determination step 506, the policy management system may compare the extracted binary-file characteristic information with binary-file characteristics of known malware types (e.g., bots, viruses, worms, Trojan Horses, etc.) to determine if the data flow carries content having binary-file characteristics of known malware (e.g., bots, viruses, worms, Trojan Horses, etc). If the policy management system determines that the data flow carries content that has binary-file characteristics in common with known malware (i.e., determination step 506 = "Yes"), in step 550, the policy management system may generate restrictive policies rules (e.g., "block content," "allow with constraints," etc.) based on the severity of the risk and push the generated rules to the gateway/PCEF. In an embodiment, the policy management system may generate restrictive policy rules that instruct the gateway/PCEF to block the IP data flow upon determining that an IP data flow carries content having binary-file characteristics of known malware threats (e.g., viruses, Trojan Horses, etc.).

[0058] If the policy management system determines that the data flow does not carry content having binary-file characteristics of known malware types (i.e., determination step 506 = "No"), in determination step 510, the policy management system may compare extracted URL information against a list of URLs known to be under attack (e.g., denial-of-service attack, etc.) to determine if the destination of data flow content is a server under attack. If the policy management system determines that the data flow carries or is likely to carry content destined for a server under attack (i.e., determination step 510 = "Yes"), in step 550, the policy management system may generate policy rules (e.g., "block content," "allow with constraints," etc.) based on various factors (e.g., severity of the risk, subscriber's usage history, subscription level, likelihood request is genuine, etc.) and push the generated rules to the gateway/PCEF. In an embodiment, the policy management system may generate an "allow with constraints" rule upon determining that the IP data flow represents a malware threat (e.g., carries content destined for a server under attack), but does not contain a binary that is a malware threat (e.g., carries content having binary-file characteristics of known malware). In an embodiment, the policy management system may generate rules that include constraints that set an upper threshold on the rate of IP data flows and/or introduce an artificial latency to the IP data flow. In this manner, the policy management system may generate intelligent policy rules to prevent attacks on servers, in real time and from within the network, without shutting off all communications to a particular Internet site.

[0059] If the policy management system determines that the data flow does not carry content destined for a server under attack (i.e., determination step 510 = "No"), in determination step 512, the policy management system may compare SMTP information of emails carried

IPR2023-01353 Page 00599

by a data-flow with known spam heuristics (e.g., incorrect SMTP headers, origin IP address blacklisting, email not relayed, message body contains blacklisted keywords, message attachment scanning using Optical Character Recognition (OCR) techniques, Artificial Intelligence (AI) based methods such as Bayesian filtering, or any combination of these) to determine if the emails originate from a spam-sending bot. If the policy management system determines that the data-flow carries an email that originates from a spam-sending bot (i.e., determination step 512 = "Yes"), in step 550, the policy management system may generate restrictive policy rules that block or restrict the data flow and push the generated rules to the gateway/PCEF for enforcement. In an embodiment, the policy management system may generate an "allow with constraints" rule upon determining that the IP data flow carries content from a known spam bot. In this manner, the policy management system may generate intelligent policy rules to reduce spam, in real time and from within the network, without filtering desired emails/communications.

[0060] If the policy management system determines that the IP data flow does not carry an email that originates from a spam-sending bot (i.e., determination step 512 = "No"), in determination step 514, the policy management system may compare the IP address information with a list of known offender IP addresses to determine if the IP data flow originates from, or is destined for, a mobile device that is compromised (e.g., infected with malware, a virus, etc.) or blacklisted (e.g., known to intentionally engage in malicious activities). If the policy management system determines that the IP address information is associated with an IP address in the list of known offender IP addresses (i.e., determination step 514 = "Yes), in step 550, the policy management system may generate policy rules that block or restrict the data flow and push the generated rules to the gateway/PCEF for enforcement. In an embodiment, the policy management system may generate an "allow with constraints" rule upon determining that the IP data flow carries content originating from, or destined for, a mobile device that is compromised or blacklisted. In this manner, the policy management system may generate intelligent policy rules to prevent the wireless device, a network server, or both from engaging in malicious activities, without blocking all the services/communications between a wireless device and a network component.

[0061] If the policy management system determines that the IP address information is not associated with an IP address in the list of known offender IP addresses (i.e., determination step 514 = "No"), in determination step 516, the policy management system may compare the URL information to a list of known offender network addresses (e.g., URLs) to determine if the data flow is destined for a website/server known to engage in malicious activities. If the policy management system determines that the URL information is associated with a URL in the list of known offender URLs (i.e., determination step 516 = "Yes"), in step 550, the policy management system may generate restrictive policy rules and push the generated rules to the gateway/PCEF for enforcement. In an embodiment, the policy management system may generate an "allow with constraints" rule upon determining that the IP data flow carries content destined for a website/server known to engage in malicious activities, without blocking all access to the website/server.

[0062] If the policy management system determines that the URL information is not associated with a URL in the list of known offender URLs (i.e., determination step 516 = "No"), in step 560, the policy management system may generate rules to allow the flow, or perform additional operations (e.g., observed behavior detection methods) to determine if the data flow carries content (e.g., malware) that poses a security risk to a wireless device, network server, or both.

[0063] In an embodiment, the policy management system may perform observed behavior detection methods to determine if the data flow carries content (e.g., malware) that may be harmful or a security risk to the network/wireless devices. The policy management system may perform the observed behavior detection methods instead of, or in addition to, the pattern matching detection methods discussed above. Observed behavior detection methods allow the policy management system to detect emerging and yet-unidentified security risks (e.g., malware threats) that are not yet prevalent enough to be known to the malware protection system or third party antivirus/security systems. The observed behavior detection methods may exploit the policy management system's access to the control network information to monitor IP data flows for suspicious characteristics and to identify security risks. For example, the policy management system may access information stored in a user repository and/or charging system in a control network to identify and neutralize security risks that cannot be identified or neutralized by systems that do not have access to user/charging information (e.g., components outside the control network).

[0064] FIGs. 6A-6B illustrate example observed behavior detection methods that may be performed by the policy management system to determine if an IP data flow is a security risk/malware threat. Performing the observed behavior oriented detection methods may include monitoring a number of data flows matching a specific behavior and/or detecting an unexpected rise in a number of requests matching a defined behavior from a specific wireless device. The specific and defined behaviors may include any of the behaviors described below, including unexpected rise in requests, requests to a specific port 80 of a specific IP address, increases in HTTP requests to a specific URL, unexpected rise in requests to a single destination, unexpected rise in outbound emails and/or DNS queries, etc.

[0065] FIG. 6A illustrates an example observed behavior detection method 600 for determining if an IP data flow carries content that may contribute to a denial of

IPR2023-01353 Page 00600

service attack. In step 602, the policy management system may receive parameters identifying certain key characteristics of the IP data flows and/or network traffic. In step 604, the policy management system may analyze the parameters and update memories, counters and/or threshold values used to monitor sudden and unexpected rises in requests to a single destination. These counters may be used to store running totals, averages, and/or maximum values for one or more different time periods (e.g., the total number of requests to a destination IP address in the last 24 hours, or the average number of requests to a destination IP address per month). Therefore, the counters can be used to store or derive request rates. The counters may be specific to the user, or they may be generalized across groups of users. Therefore, a sudden and unexpected rise in requests to a single destination can be defined in terms of a counter value surpassing a pre-defined rate. This predefined rate can be expressed in terms of the counters (e.g., a sudden and unexpected rise can be defined as the current rate being greater than a tripling of the maximum hourly rate ever experienced). The associated thresholds may be defined as constants, or in terms of the counters (e.g., a threshold can be defined as a quadrupling of the maximum hourly rate ever experienced). Thus, there can be a sudden and unexpected rise in the number of requests to a single destination, but it may not exceed a threshold. In step 606, the policy management system may monitor the number of requests by performing IP level and application level monitoring operations. IP level monitoring operations may include monitoring requests to a specific port 80 of a specific IP address. Application level monitoring operations may include monitoring increases in HTTP requests to a specific URL. In determination step 608, the policy management system may determine if there has been a sudden and unexpected rise in requests to a single destination. This determination may be based upon a counter value surpassing a pre-defined rate, as described previously. If the policy management system determines that there has not been a sudden and unexpected rise in requests to a particular address (i.e., determination step 608 = "No"), in step 612, the policy management system may generate policy rules and push the generated rules to the gateway/PCEF. If the policy management system determines that there has been a sudden and unexpected rise in requests to a single destination (i.e., determination step 608 = "Yes"), in determination step 610, the policy management system may determine if the rise in requests exceeds a predetermined threshold. If the rise in requests does not exceed the predetermined threshold (i.e., determination step 610 = "No"), in step 612, the policy management system may generate policy rules and push the generated rules to the gateway/PCEF. If the rise in requests exceeds the predetermined threshold (i.e., determination step 610 = "Yes"), the policy management system may conclude that a denial of service attack is emerging, and in step 614, register the destination as being under attack, generate restrictive policy rules to block or limit current and future flows from accessing the registered destination, and push the generated rules to the gateway/PCEF for implementation.

[0066] FIG. 6B illustrates an example observed behavior detection method 650 for determining if an IP data flow carries content from spam engines. In step 652, the policy management system may receive parameters identifying certain key characteristics of the IP data flows and/or network traffic from a gateway/PCEF. In step 654, the policy management system may analyze the parameters and update local databases and/or counters with information that may be used to monitor sudden rises in outbound emails and/or sudden rises in domain name system (DNS) queries for mail exchanger (MX) records by a single mobile device. In step 656, the policy management system may monitor the number of outbound emails and/or DNS queries by performing IP-level and application-level monitoring operations. IP-level monitoring operations performed by the policy management system may include monitoring/counting the number of requests by a specific mobile device to port 25 of different IP addresses. Application-level monitoring operations may include monitoring the number of SMTP requests being sent directly to an email recipient's SMTP server from a single mobile device. The IP-level and application-level monitoring operations may be used to identify data flows carrying content originating from a spam engine because "well-behaved" email clients generally send all their outbound email via a single SMTP server, whereas spam engines typically send outbound emails directly to the email recipient's SMTP server.

[0067] In determination step 658, the policy management system may determine if there has been a sudden and unexpected rise in outbound emails and/or in domain name system (DNS) queries for mail exchanger (MX) records by a single mobile device. This determination may be based upon a counter value surpassing a pre-defined rate, as described previously. If the policy management system determines that there has not been a sudden and unexpected rise in outbound emails/queries (i.e., determination step 658 = "No"), in step 662, the policy management system may generate policy rules and push the rules to the gateway/PCEF. If the policy management system determines that there has been a sudden and unexpected rise in outbound emails and/or in domain name system (DNS) queries (i.e., determination step 658 = "Yes"), in determination step 660, the policy management system may determine if the rise in requests exceeds a predetermined threshold. If the rise exceeds the predetermined threshold (i.e., determination step 660 = "Yes"), in step 664, the policy management system may identify the data flow as carrying content from a spam engine, register the spam engine (e.g., update a spam engine list in memory, etc.), generate restrictive policy rules to block or limit data flows originating from the spam engine, and push the rules to the gateway/PCEF for enforcement. If the rise does not exceed the

IPR2023-01353 Page 00601

predetermined threshold (i.e., determination step 660 = "No"), in step 664, the policy management system may generate policy rules (e.g., allow, allow with constraints, etc.) and push the rules to the gateway/PCEF.

[0068] As mentioned above, the policy management system may perform detection methods (e.g., pattern matching, observed behavior, etc.) and generate restrictive policy rules based on the type of malware detected. For example, the policy management system may generate a "block" rule if an IP data flow carries content having binary-file characteristics of known malware threats (e.g., viruses, Trojan Horses, etc.) and an "allow with constraints" rule if the IP data flow represents a malware threat, but does not contain a binary that is a malware threat (e.g., the IP data flow is part of a denial of service attack, the IP data flow originates from a spam bot, etc.). The "allow with constraints" rule may define requirements (i.e., constraints) for allowing certain IP data flows to pass through the gateway/PCEF. For example, the constraints may require the gateway/PCEF to introduce an artificial latency to certain IP data flows or to set an upper threshold on certain IP data flow rates.

[0069] The "allow with constraints" type rules, and the constraints they impose, allow the policy management system to customize each rule to the preferences of a particular subscriber, wireless device type or type value, network component or network client. For example, the policy management system may access user preference parameters that define which constraints are to be used in various situations, as well as the actions, conditions and variables (e.g., risk levels, thresholds, data usage levels, costs, etc.) associated with each constraint. In an embodiment, a unique/separate set of user preference parameters may be maintained for each wireless subscriber, device, device type, network component, or client. In this manner telecommunications network operators and their customers may customize responses to malware events, which may be implemented automatically and in real time, consistent with the type of threat and how the network operator and/or customer prefers to respond. For example, some organizations cannot to afford to shut off all requests for service during a denial of service attack, and thus would prefer constraints on the volume or data rates imposed on incoming service requests.

[0070] In an embodiment, the constraints imposed by the policy management system may be proportional to a malware threat or threat-level as determined by the malware protection system or specified by an operator. Methods for determining a malware threat-level within malware protection system are disclosed herein. In an embodiment, the constraints imposed by the policy management system may become progressively more restrictive or constrained, in real-time, as a malware threat or threat-level rises as determined by malware protection system, an operator or other systems configured to identify the threat level.

[0071] User preference parameters may define the actions and conditions associated with each constraint in "allow with constraints" type rules. Thus, the user preference parameters may be used by the policy management system to generate policy rules that instruct the gateway/PCEF to perform several different types of operations (i.e., actions) as well as set conditions for performing those actions. For example, the policy management system may access the user preference parameters to determine that the correct course of action (i.e., the correct rule) for the gateway/PCEF upon the detection of low-risk malware is to send an alert message to the wireless device without blocking the IP data flow. Likewise, the user preference parameters may be used by the policy management system to generate rules that instruct the gateway/PCEF to block the current IP data flow and all future IP data flows until the malware has been manually removed from the mobile device.

[0072] In an embodiment, the policy management system may perform quarantining operations by storing details of IP data flows, aggregated details of the IP data flows, or sequestering the actual content of the IP data flows, that were allowed, constrained, or blocked. The policy management system may store source information, destination information and time information for each IP data flow that is constrained. For example, the policy management system may track the number of IP data flows containing infected binaries within a predetermined time period (e.g., within the last month) and store the number value in a memory. The policy management system may also store emails deemed to be spam. This information may be stored such that a wireless device user whose mobile device is associated with these IP data flows may subsequently view the quarantining details. In an embodiment, the policy management system may be configured to receive messages from the wireless device user that trigger further quarantining actions (e.g., removing a stored IP data flow, un-quarantining stored IP data flow, etc.). For example, the user may send the policy management system a message to reclassify an email as not spam, in which case the email may be pushed to the mobile device by the policy management system. In an embodiment, the user may classify the IP data flow in a manner that causes the policy management system to allow delivery of future similar IP data flows.

[0073] In the various embodiments, user preference parameters may be generated from information from a subscriber account in the user repository, by interacting with the user (e.g., via SMS messages sent through the notification server 126, etc.) to determine his/her preferences in real-time, by using contextual information (e.g., roaming in third party network, home network, etc.), or any combination thereof. For example, a wireless device user may upload user preference parameters to the user repository that instructs the policy management system to send an alert to the user's wireless device warning of potential risks if the wireless device is not roaming in a third-party mobile data network and to block the current IP data flow and all future IP data flows if the wireless

IPR2023-01353 Page 00602

device is roaming in a third-party mobile data network. In this manner, cost conscientious users may prevent a potential malware program from accessing online services while roaming in an expensive third-party mobile data network.

[0074] In an embodiment, the user preference parameters may be based on a malware threat model. The malware threat model may be defined in terms of multiple independent factors, such as the probability of infection, the type of infection, the severity of the threat posed by the infection, and the difficulty associated with removing the infection. These factors may be weighted and combined to produce an overall threat level. The user may then specify his/her preference in terms of this overall threat level. Alternatively, the user may specify his/her preference in terms of one of the factors within the threat model (e.g., a user may be willing to be exposed to a type of infection that installs Adware, whereas he/she may not be willing to be exposed to a type of infection that corrupts or destroys data stored on the wireless device).

[0075] In an embodiment, the policy management system and/or gateway/PCEF may be configured to charge users/subscribers differently for the IP data flows depending on the actions. For example, a user/subscriber may not be charged for any IP data flows that were blocked by the gateway.

[0076] In an embodiment, a malware prevention and removal client (MPRC) software application may be pushed to the wireless devices. The policy management system may, upon determining that an IP data flow is associated with a malware threat, notify the malware protection system of the wireless device's identity and the detected malware. The malware protection system may then update, or instruct, the MPRC to take preventative, recovery, or removal actions specific to the wireless device type or type value and the detected malware threat. For example, the malware protection system may instruct the MPRC to disable the wireless device's data capabilities and/or isolate the mobile device from the mobile data network until the malware is removed or is otherwise no longer a threat. In an embodiment, the policy management system may send a notification based on the status of the user's subscription to the malware prevention service.

[0077] As discussed above with reference to FIG. 3, the policy management system may exchange information with both a user repository and a charging system. In an embodiment, the policy management system may directly charge the subscription accounts of beneficiaries (e.g., wireless device users, web servers, telecommunication clients, etc.) of the above mentioned malware detection/prevention services by sending a charging message to deduct funds from each subscriber's service account. For example, the policy management system may send a charge message to charge a wireless device user's wireless subscription account each time malware is detected and/or removed from the user's wireless device.

[0078] The policy management system may support multiple different charging models to support any required business models. The charging models may include providing the above mentioned services free-of-charge, charging on an annual basis, on a pay-as-you-consume basis, or any combination thereof. If the services are charged on an annual basis, the policy management system may determine the status of the users' annual subscriptions (e.g., last time payment was made, account status, subscription level, etc.) by accessing the user repository and sending charging messages to the charging system to charge the customers as required. If the services are charged on a pay-as-you-consume basis, the policy management system may send a charging request to the charging system each time the service is provided (e.g., each time a message is send waning of potential malware). In an embodiment, each service may be charged separately. For example, the policy management system may send charging messages to the charging system annually for the detection services and send charging messages to the charging system each time malware is removed.

[0079] In an embodiment, the policy management system may be configured to automatically adapt to a network wide stance if the detection of malware indicates that the network is under attack. The adaption of the network wide stance may be in stages, such that the network-wide protections are progressive. In an embodiment, the policy management system may change and/or bulk push policies/rules into the network if the network the detection of malware indicates that the network is under attack. Policies/rules may be pushed to network elements upon detection of a new malware threat or upon a malware threat level achieving a predefined threshold such that the network adopts an "under attack" pre-defined configuration. The initiation of such "under attack" responses or configuration may be based on the detection of malware and/or policy counters exceeding a predefined threshold number or instances of attack.

[0080] The various embodiments and logical components may be implemented on any of a variety of commercially available server devices, such as the server 700 illustrated in FIG. 7. Such a server 700 typically includes a processor 701 coupled to volatile memory 702 and a large capacity nonvolatile memory, such as a disk drive 703. The server 700 may also include a floppy disc drive, compact disc (CD) or DVD disc drive 706 coupled to the processor 701. The server 700 may also include network access ports 704 coupled to the processor 701 for establishing data connections with a network 705, such as a local area network coupled to other broadcast system computers and servers.

[0081] The processor 701 may be any programmable microprocessor, microcomputer or multiple processor chip or chips that can be configured by software instructions (applications) to perform a variety of functions, including the functions of the various embodiments described below. Multiple processors 701 may be provided,

IPR2023-01353 Page 00603

such as one processor dedicated to wireless communication functions and one processor dedicated to running other applications. Typically, software applications may be stored in the internal memory 702, 703 before they are accessed and loaded into the processor 701. The processor 701 may include internal memory sufficient to store the application software instructions.

[0082] The foregoing method descriptions and the process flow diagrams are provided merely as illustrative examples and are not intended to require or imply that the steps of the various embodiments must be performed in the order presented. As will be appreciated by one of skill in the art the order of steps in the foregoing embodiments may be performed in any order. Words such as "thereafter," "then," "next," etc. are not intended to limit the order of the steps; these words are simply used to guide the reader through the description of the methods. Further, any reference to claim elements in the singular, for example, using the articles "a," "an" or "the" is not to be construed as limiting the element to the singular.

[0083] The various illustrative logical blocks, modules, circuits, and algorithm steps described in connection with the embodiments disclosed herein may be implemented as electronic hardware, computer software, or combinations of both. To clearly illustrate this interchangeability of hardware and software, various illustrative components, blocks, modules, circuits, and steps have been described above generally in terms of their functionality. Whether such functionality is implemented as hardware or software depends upon the particular application and design constraints imposed on the overall system. Skilled artisans may implement the described functionality in varying ways for each particular application, but such implementation decisions should not be interpreted as causing a departure from the scope of the present invention.

[0084] The hardware used to implement the various illustrative logics, logical blocks, modules, and circuits described in connection with the aspects disclosed herein may be implemented or performed with a general purpose processor, a digital signal processor (DPC), an application specific integrated circuit (ASIC), a field programmable gate array (FPGA) or other programmable logic device, discrete gate or transistor logic, discrete hardware components, or any combination thereof designed to perform the functions described herein. A general-purpose processor may be a microprocessor, but, in the alternative, the processor may be any conventional processor, controller, microcontroller, or state machine. A processor may also be implemented as a combination of computing devices, e.g., a combination of a DPC and a microprocessor, a plurality of microprocessors, one or more microprocessors in conjunction with a DPC core, or any other such configuration. Alternatively, some steps or methods may be performed by circuitry that is specific to a given function.

[0085] In one or more exemplary aspects, the functions described may be implemented in hardware, software, firmware, or any combination thereof. If implemented in software, the functions may be stored on or transmitted over as one or more instructions or code on a non-transitory computer-readable medium. The steps of a method or algorithm disclosed herein may be embodied in a processor-executable software module which may reside on a non-transitory computer-readable storage medium. Non-transitory computer-readable storage media may be any available media that may be accessed by a computer (e.g., a server) or processor of a computer or server. By way of example, and not limitation, non-transitory computer-readable media may include RAM, ROM, EEPROM, CD-ROM or other optical disk storage, magnetic disk storage or other magnetic storage devices, or any other storage medium that may be used to store desired program code in the form of processor-executable instructions or data structures and that may be accessed by a computer. Disk and disc, as used herein, includes compact disc (CD), laser disc, optical disc, digital versatile disc (DVD), floppy disk, and blu-ray disc where disks usually reproduce data magnetically, while discs reproduce data optically with lasers. Combinations of the above should also be included within the scope of non-transitory computer-readable media. Additionally, the operations of a method or algorithm may reside as one or any combination or set of codes and/or instructions on a non-transitory machine readable medium and/or computer-readable medium, which may be incorporated into a computer program product.

[0086] The preceding description of the disclosed embodiments is provided to enable any person skilled in the art to make or use the present invention. Various modifications to these embodiments will be readily apparent to those skilled in the art, and the generic principles defined herein may be applied to other embodiments without departing from the scope of the invention. Thus, the present invention is not intended to be limited to the embodiments shown herein but is to be accorded the widest scope consistent with the following claims and the principles and novel features disclosed herein.

Claims

1. A communication method, comprising:

   receiving in a server from one or more communication system control network components data flow parameters identifying characteristics of a data flow;

   performing in the server a detection method to determine if the data flow carries malicious content;

   generating restrictive policy rules in response to determining the data flow carries malicious content;

   generating non-restrictive policy rules in response to determining the data flow does not

IPR2023-01353 Page 00604

carry malicious content; and
pushing the generated policy rules to a communication system gateway configured to control a flow of packet-based data between a wireless device and an external network.

2. The method of claim 1, wherein generating restrictive policy rules in response to determining the data flow carries malicious content comprises generating restrictive policy rules based on a subscriber's usage history and a subscription level.

3. The method of claim 1, wherein generating restrictive policy rules in response to determining the data flow carries malicious content comprises generating constraints that are proportional to a threat-level associated with the malicious content and generating restrictive policy rules that impose the generated constraints on the communication system gateway.

4. The method of claim 1, further comprising transmitting malware detection feedback to a network component in a second external network.

5. The method of claim 1, further comprising interacting with a wireless device user to determine user preferences in real-time via alert messages sent through a notification server to receive user preference parameters.

6. The method of claim 1, wherein the data flow originates from the wireless device, and generating restrictive policy rules in response to determining the data flow carries malicious content comprises generating restrictive policy rules that instruct the communication system gateway to block future data flows originating from the wireless device.

7. The method of claim 1, wherein performing a detection method to determine if the data flow carries content that is a security risk comprises performing a pattern matching detection method.

8. The method of claim 7, wherein:

receiving in a server from one or more communication system control network components data flow parameters identifying characteristics of a data flow comprises receiving parameters identifying binary-file characteristic information; and
performing a pattern matching detection method comprises comparing the identified binary-file characteristic information with a binary-file characteristic of a known malware type to determine if the data flow carries content having binary-file characteristics of known malware.

9. The method of claim 7, wherein:

receiving in a server from one or more communication system control network components data flow parameters identifying characteristics of a data flow comprises receiving parameters identifying source and destination information; and
performing in the server a pattern matching detection method comprises comparing the identified source and destination information against a list of known offender sources and destinations to determine if the data flow originates from a wireless device known to engage in malicious activity and if the data flow is destined for a website/server known to engage in malicious activities.

10. The method of claim 7, wherein:

receiving in a server from one or more communication system control network components data flow parameters identifying characteristics of a data flow comprises receiving parameters identifying destination information; and
performing in the server a pattern matching detection method comprises comparing the identified destination information against a list of destinations known to be under attack to determine if a destination of the data flow is a network server under attack.

11. The method of claim 7, wherein:

receiving in a server from one or more communication system control network components data flow parameters identifying characteristics of a data flow comprises receiving parameters identifying messaging information; and
performing in the server a pattern matching detection method comprises comparing the identified messaging information with spam heuristics.

12. The method of claim 1, wherein performing in the server a detection method to determine if the data flow carries content that is a security risk comprises performing a behavior oriented detection method.

13. The method of claim 12, wherein performing a behavior oriented detection method comprises accessing information stored in one of a user repository of a control network and a charging system of a control network.

14. The method of claim 12, wherein performing a behavior oriented detection method comprises updating a counter value to monitor a number of data flows

IPR2023-01353 Page 00605

matching a specific behavior.

15. The method of claim 12, wherein performing a behavior oriented detection method comprises performing monitoring operations in order to detect an unexpected rise in a number of requests matching a defined behavior from a specific wireless device.

16. A server, comprising:

a memory; and
a processor coupled to the memory, wherein the processor is configured with processor-executable instructions to perform operations comprising:

receiving data flow parameters identifying characteristics of a data flow;
performing a detection method to determine if the data flow carries malicious content;
generating restrictive policy rules in response to determining the data flow carries malicious content;
generating non-restrictive policy rules in response to determining the data flow does not carry malicious content; and
pushing the generated policy rules to a communication system gateway configured to control a flow of packet-based data between a wireless device and an external network.

17. A non-transitory computer readable medium having stored thereon processor-executable software instructions configured to cause a processor to perform operations comprising:

receiving from one or more communication system control network components data flow parameters identifying characteristics of a data flow;
performing a detection method to determine if the data flow carries malicious content;
generating restrictive policy rules in response to determining the data flow carries malicious content;
generating non-restrictive policy rules in response to determining the data flow does not carry malicious content; and
pushing the generated policy rules to a communication system gateway configured to control a flow of packet-based data between a wireless device and an external network.

IPR2023-01353 Page 00606

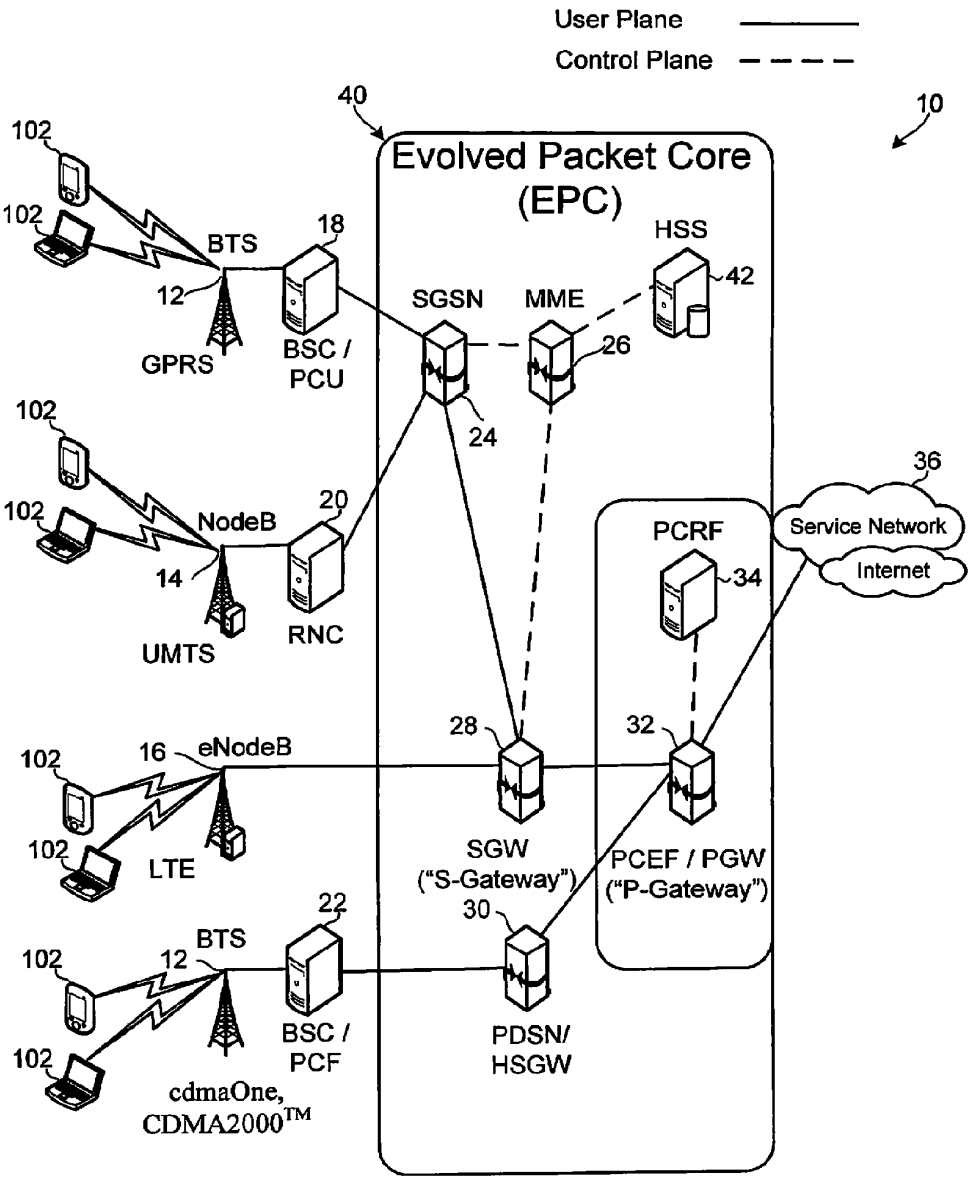

User Plane
Control Plane
40
10
Evolved Packet Core
(EPC)
102
102
BTS
18
12
GPRS
BSC /
PCU
SGSN
MME
HSS
42
26
24
102
102
NodeB
20
14
UMTS
RNC
36
Service Network
Internet
PCRF
34
28
32
102
102
16
eNodeB
LTE
SGW
("S-Gateway")
PCEF / PGW
("P-Gateway")
30
22
102
102
12
BTS
BSC /
PCF
PDSN/
HSGW
cdmaOne,
CDMA2000™

FIG. 1

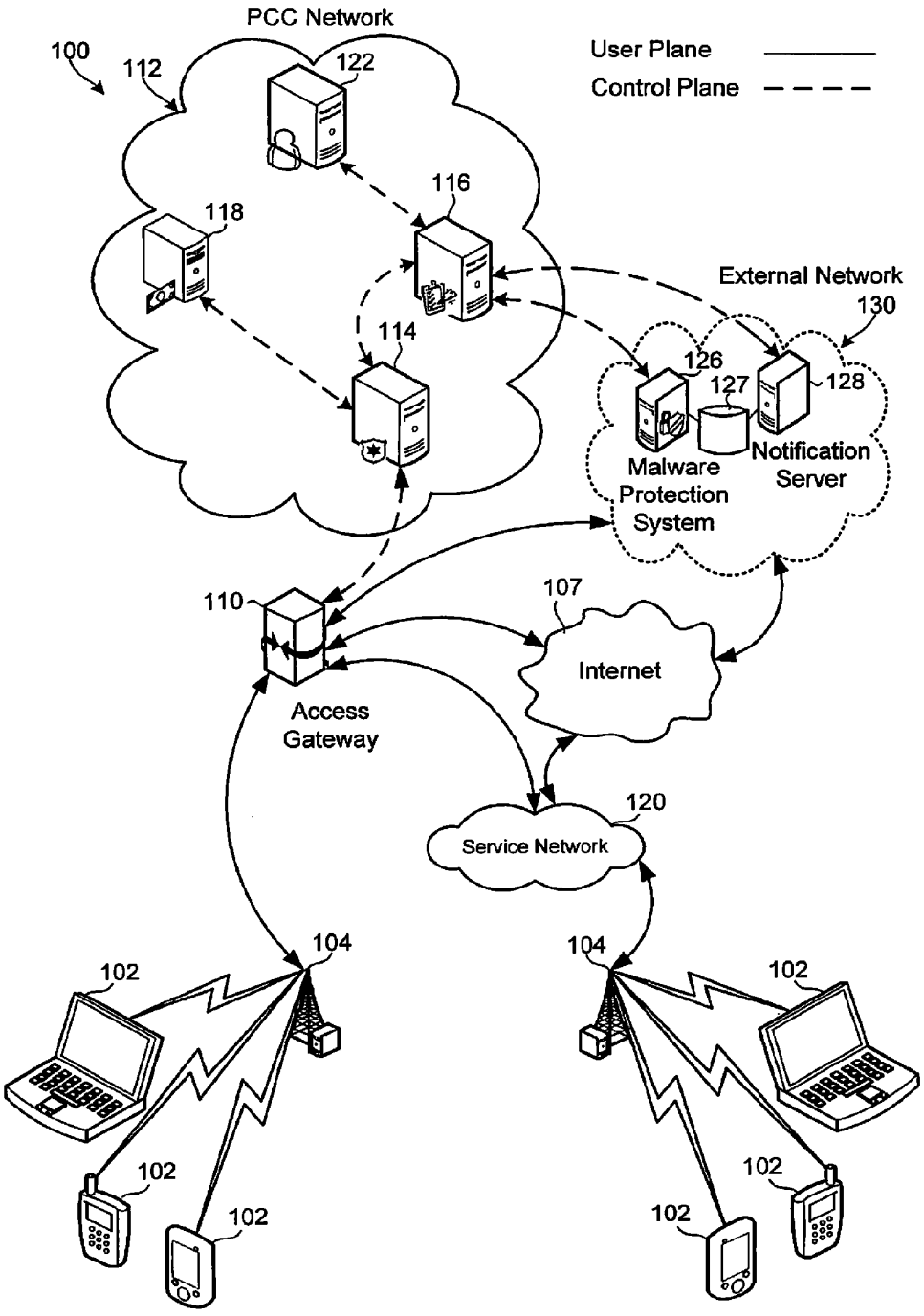


FIG. 2

IPR2023-01353 Page 00608

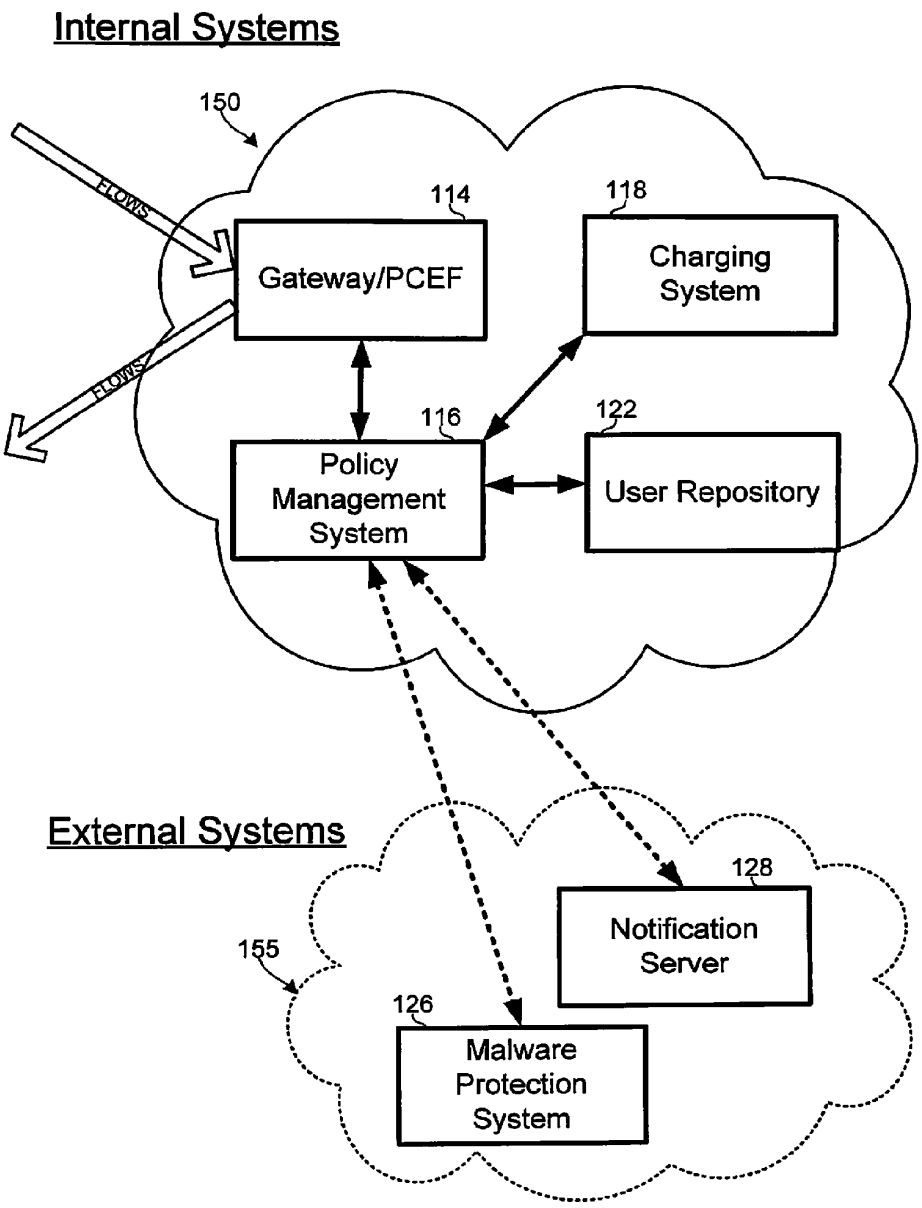


FIG. 3A

IPR2023-01353 Page 00609

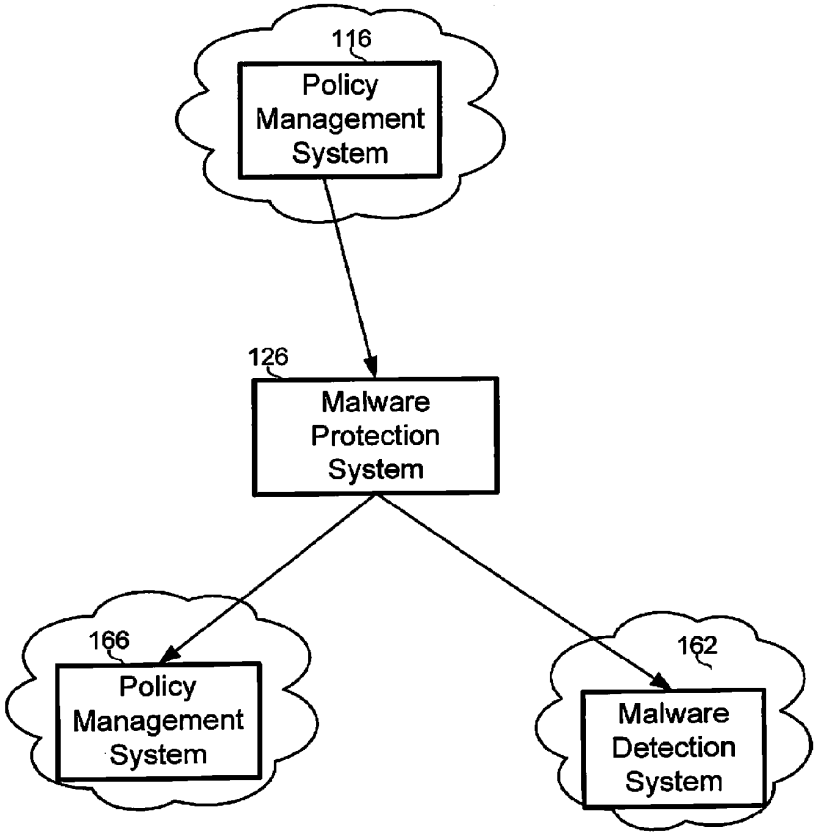


FIG. 3B

IPR2023-01353 Page 00610

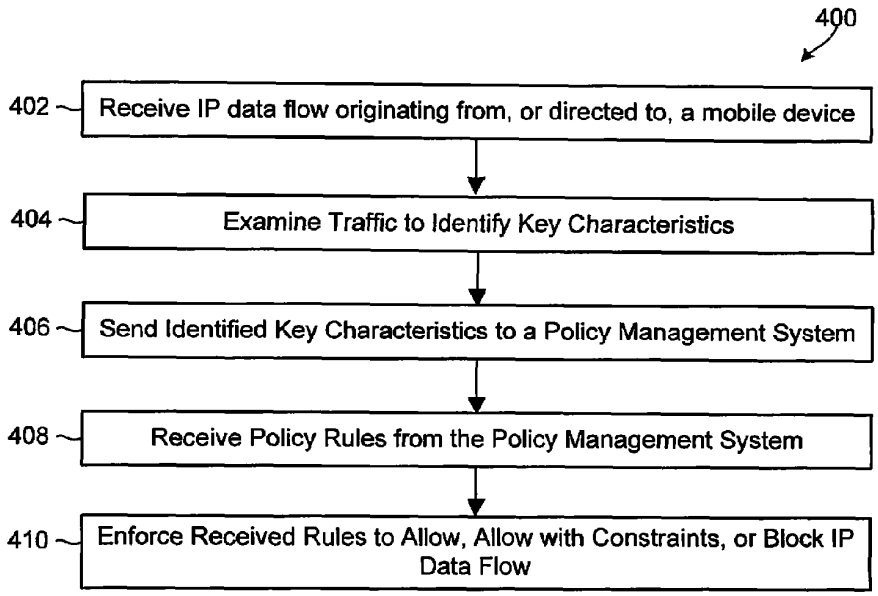


FIG. 4A

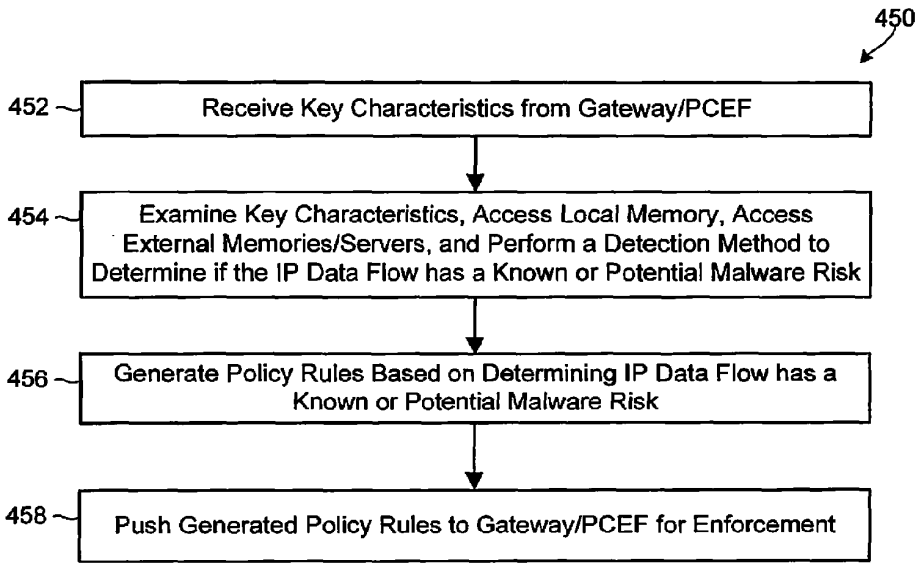


FIG. 4B

IPR2023-01353 Page 00611

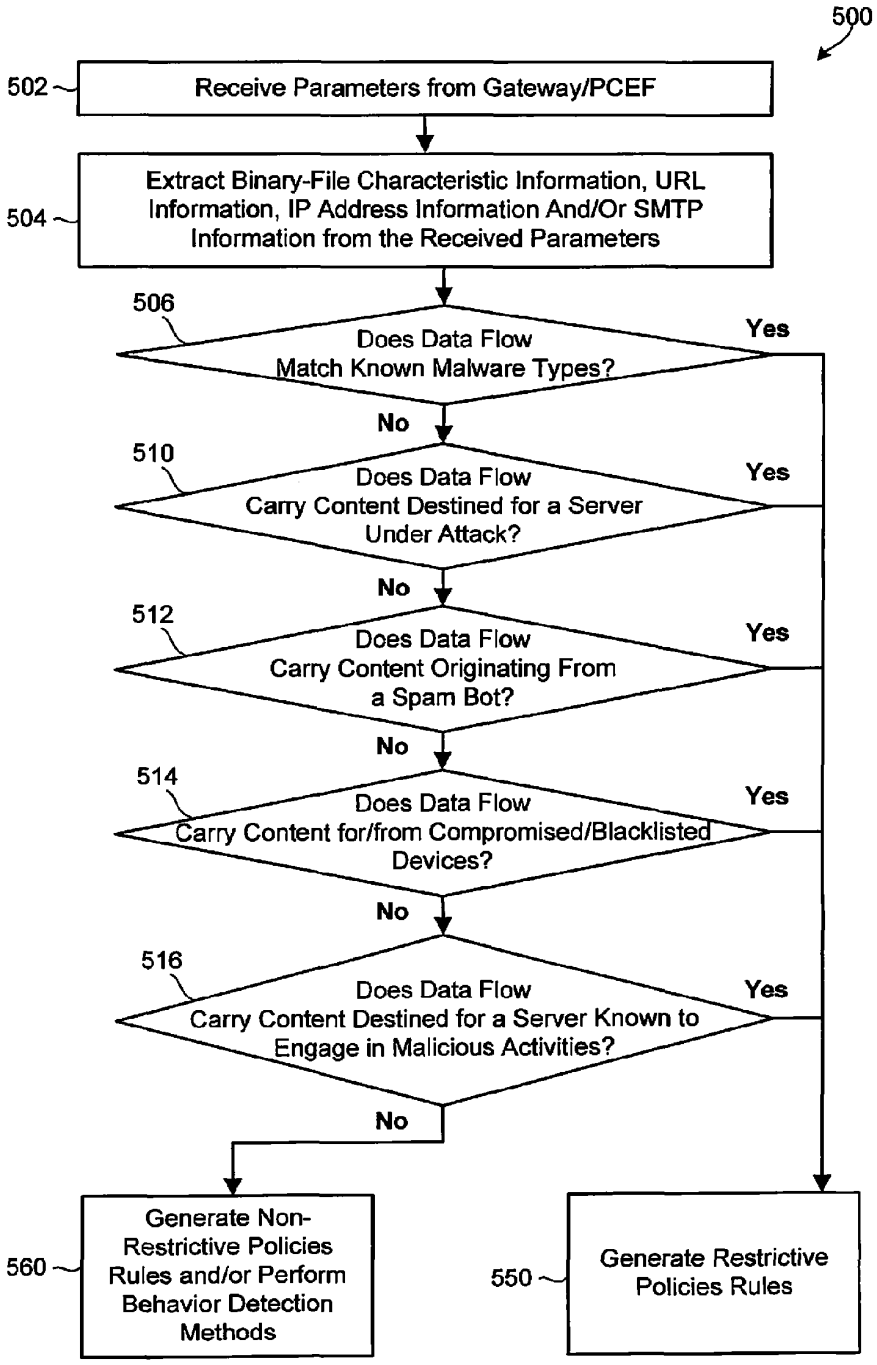


FIG. 5

IPR2023-01353 Page 00612

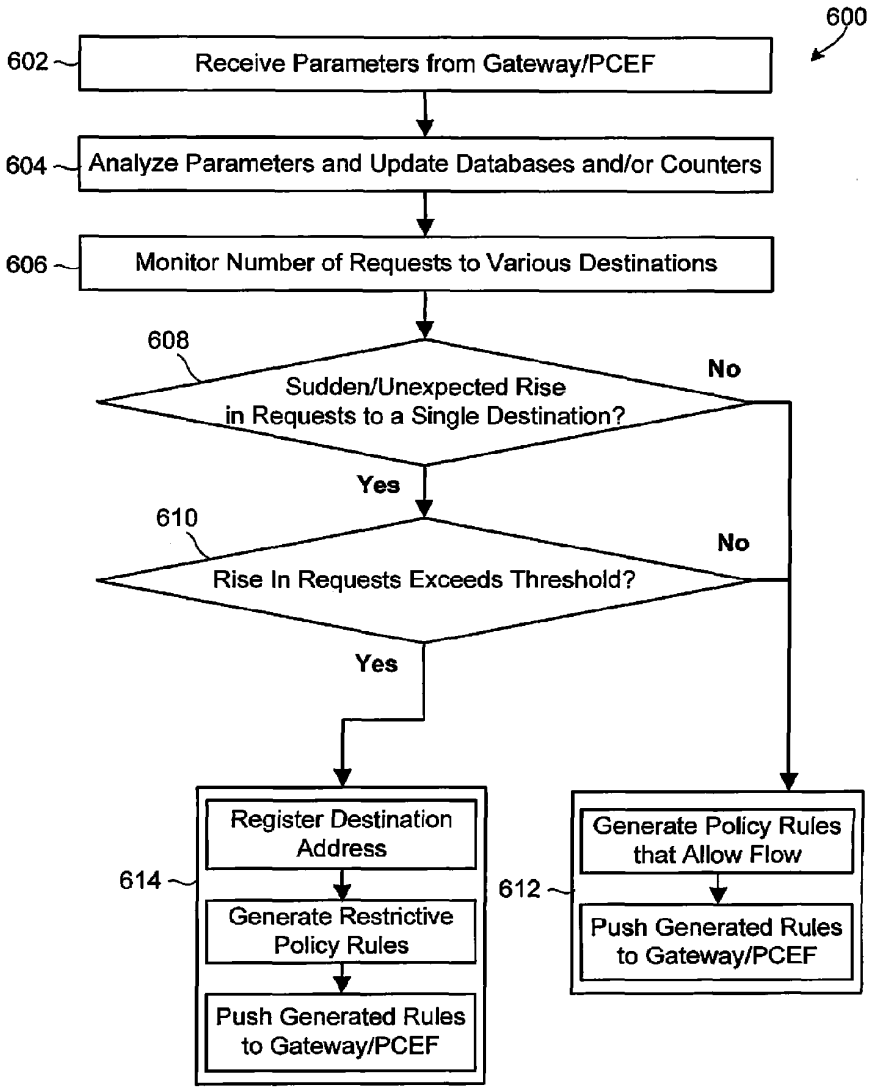

600
602
Receive Parameters from Gateway/PCEF
604
Analyze Parameters and Update Databases and/or Counters
606
Monitor Number of Requests to Various Destinations
608
Sudden/Unexpected Rise in Requests to a Single Destination?
No
Yes
610
Rise In Requests Exceeds Threshold?
No
Yes
614
Register Destination Address
Generate Restrictive Policy Rules
Push Generated Rules to Gateway/PCEF
612
Generate Policy Rules that Allow Flow
Push Generated Rules to Gateway/PCEF

FIG. 6A

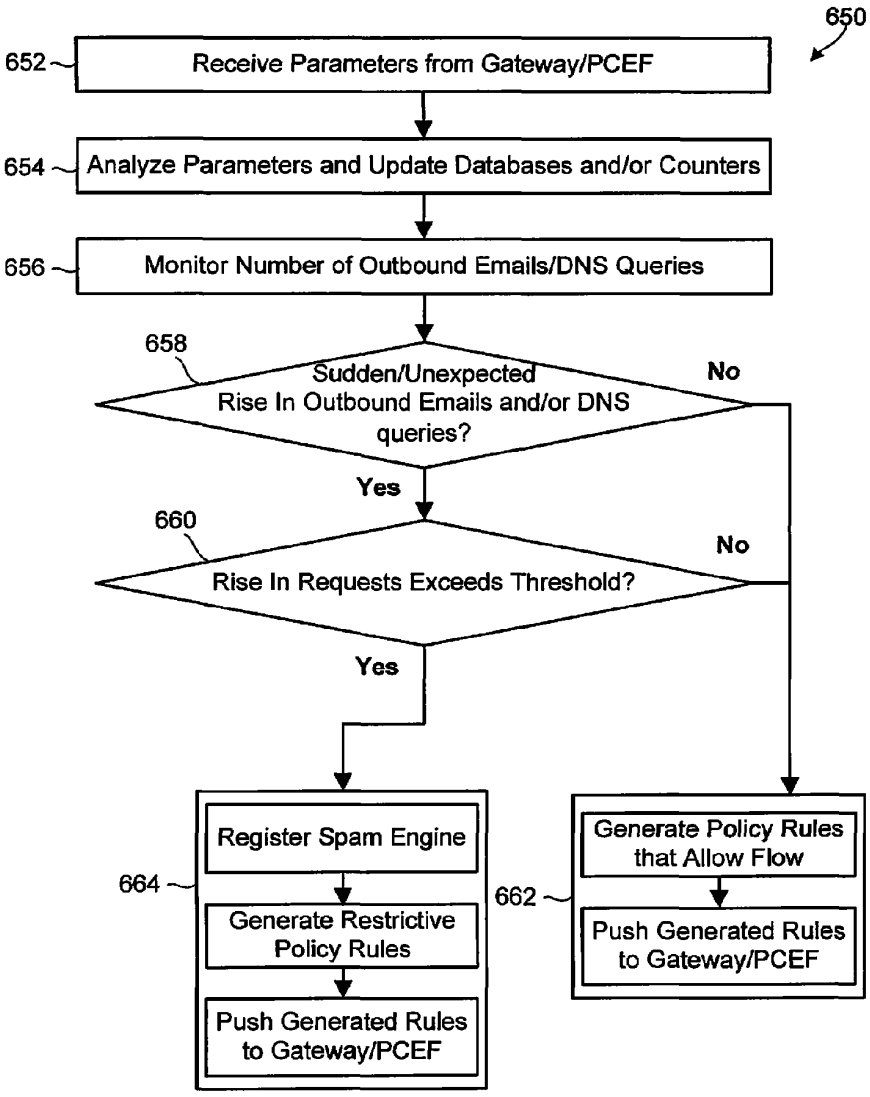


FIG. 6B

IPR2023-01353 Page 00614

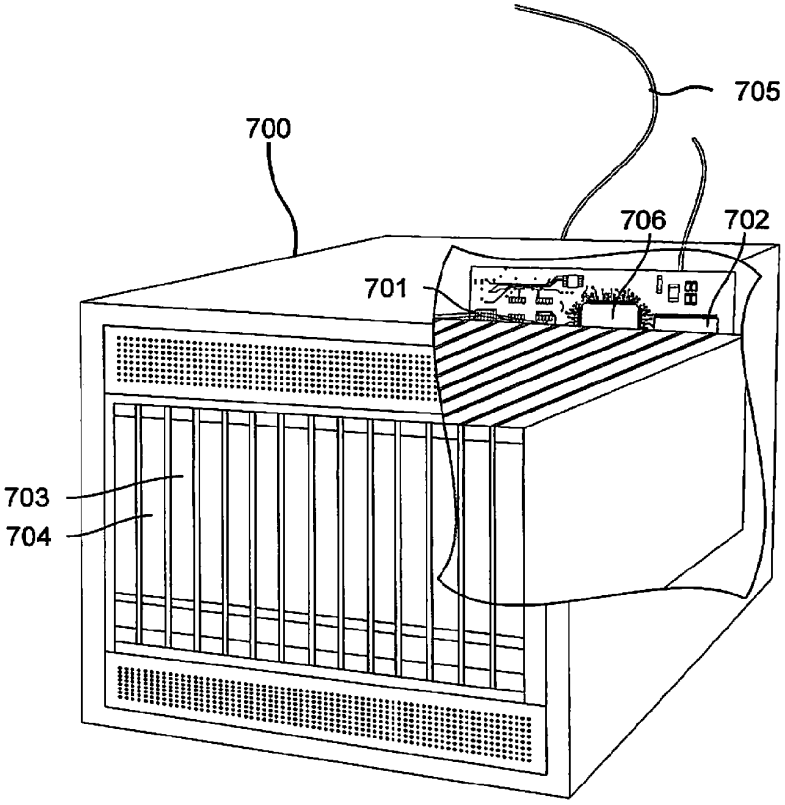


FIG. 7

IPR2023-01353 Page 00615

Europäisches Patentamt
European Patent Office
Office européen des brevets

EUROPEAN SEARCH REPORT

Application Number
EP 12 15 8921

DOCUMENTS CONSIDERED TO BE RELEVANT

Category	Citation of document with indication, where appropriate, of relevant passages	Relevant to claim	CLASSIFICATION OF THE APPLICATION (IPC)
X	WO 2009/132700 A1 (ERICSSON TELEFON AB L M [SE]; STENFELT JOHN [SE]) 5 November 2009 (2009-11-05) * page 6, lines 4-10 * * page 8, lines 6-10 * * page 15, line 25 - page 17, line 16 * * page 17, lines 29-31 * * claim 12; figures 1,4 * -----	1-17	INV. H04L12/24 H04L29/06 H04W12/08 H04W12/12 ADD. H04W24/00 H04W88/16
X	US 2004/193912 A1 (LI HONG C [US] ET AL) 30 September 2004 (2004-09-30) * paragraphs [0024], [0039] - [0041] * * figure 3 * -----	1-17	
A	US 2007/006309 A1 (HERBERT HOWARD C [US] ET AL SMITH NED M [US] ET AL) 4 January 2007 (2007-01-04) * paragraph [0026]; figure 2 * -----	1-17	

TECHNICAL FIELDS SEARCHED (IPC)

H04L
H04W

The present search report has been drawn up for all claims

Place of search	Date of completion of the search	Examiner
Munich	20 June 2012	Pajatakis, Emmanouil

CATEGORY OF CITED DOCUMENTS

X : particularly relevant if taken alone
Y : particularly relevant if combined with another document of the same category
A : technological background
O : non-written disclosure
P : intermediate document

T : theory or principle underlying the invention
E : earlier patent document, but published on, or after the filing date
D : document cited in the application
L : document cited for other reasons

& : member of the same patent family, corresponding document

1

EPO FORM 1503 03.82 (P04C01)

IPR2023-01353 Page 00616

ANNEX TO THE EUROPEAN SEARCH REPORT ON EUROPEAN PATENT APPLICATION NO. EP 12 15 8921

This annex lists the patent family members relating to the patent documents cited in the above-mentioned European search report. The members are as contained in the European Patent Office EDP file on
The European Patent Office is in no way liable for these particulars which are merely given for the purpose of information.

20-06-2012

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
WO 2009132700 A1	05-11-2009	EP 2304915 A1	06-04-2011
		US 2011041182 A1	17-02-2011
		WO 2009132700 A1	05-11-2009
US 2004193912 A1	30-09-2004	CN 1768518 A	03-05-2006
		DE 112004000428 B4	26-11-2009
		DE 112004000428 T5	09-02-2006
		GB 2412540 A	28-09-2005
		JP 2006521598 A	21-09-2006
		JP 2009187587 A	20-08-2009
		KR 20050118223 A	15-12-2005
		TW I253571 B	21-04-2006
		US 2004193912 A1	30-09-2004
		WO 2004095801 A1	04-11-2004
US 2007006309 A1	04-01-2007	TW I323102 B	01-04-2010
		US 2007006309 A1	04-01-2007
		WO 2007002875 A1	04-01-2007

EPO FORM P0459

For more details about this annex : see Official Journal of the European Patent Office, No. 12/82

IPR2023-01353 Page 00617

Espacenet

Bibliographic data: KR20010079361 (A) — 2001-08-22

FIRE WALL SYSTEM BASED ON NETWORK STATUS AND METHOD THEREOF

Inventor(s): JANG HUI JIN [KR]; KIM SANG WOOK [KR]; PARK BO SEOK [KR]; PARK IN SEONG [KR]; SONG BYEONG UK [KR] ± (JANG, HUI JIN, ; KIM, SANG WOOK, ; PARK, BO SEOK, ; PARK, IN SEONG, ; SONG, BYEONG UK)

Applicant(s): KIM SANG WOOK [KR] ± (KIM, SANG WOOK)

Classification: **- international:** ***H04L12/22; H04L12/26; H04L12/70;*** (IPC1-7): H04L12/56

- cooperative:

Application number: KR20010040894 20010709

Priority number(s): KR20010040894 20010709

Abstract of KR20010079361 (A)

PURPOSE: A fire wall system based on network status and a method thereof are provided to interrupt all illegal packet or traffic flowing into an internal network through an external information communication networks by analyzing the status of network traffic from various angles and differentially reflecting a traffic control policy together with recognition for status through a status graph. CONSTITUTION: A fire wall system based on network status consists of a network traffic collection unit(100), a static status information collection unit(200), a dynamic status information analysis unit(300), a traffic monitor unit(400), a rule generation unit(500), and a traffic control unit(600). The network traffic collection unit(100) classifies the network packets collected from an external information communication network by hosts, protocols and connections. The static status information collection unit(200), comprised of a statistics analyzer(230), a status unit table(220) and a packet manager(210), analyzes the information of the network packets transmitted from the network traffic collection unit(100), updates the information of the status unit table(220), and processes traffic information. The dynamic status information analysis unit(300), containing a traffic log table(310) and a traffic history analyzer(320), analyzes and records the long-term inclination and pattern of the network traffic of the traffic information processed at the static status information collection unit(200). The traffic monitor unit(400) monitors the trend of network traffic using each information provided from the static status information collection unit(200) and the dynamic status information analysis unit(300), recognizes an abnormal situation and outputs a state transition signal. The rule generation unit(500), composed of a policy manager(510), a status graph table(520) and an action manager(530), receives the state transition signal from the traffic monitor unit(400), creates a status graph, generates a packet filtering rule using the graph, and transfers the rule to the

traffic control unit(600) so that the traffic control unit(600) can cope with the abnormal situation.

(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 특2001-0079361
(43) 공개일자 2001년08월22일

(51) 국제분류코드
H04L 12/56 (2006.01)

(21) 출원번호 10-2001-0040894

(22) 출원일자 2001년07월09일

(71) 출원인
김상욱
대구광역시 수성구 수성1가 신세계타운 10동 501호, 대한민국

(72) 발명자
김상욱
대구광역시 수성구 수성1가 신세계타운 10동 501호, 대한민국
송병욱
대구광역시북구산격3동1313-37, 대한민국
박인성
경상북도경주시석장동858번지, 대한민국
장희진
대구광역시수성구지산동서한청산맨션102동1105호, 대한민국
박보석
대구광역시동구신암1동652-11, 대한민국

(74) 대리인
이영필
이해영

(77) 심사관
있음

전체 청구항 수 : 총 9 항

(54)네트워크 상태 기반의 방화벽 장치 및 그 방법

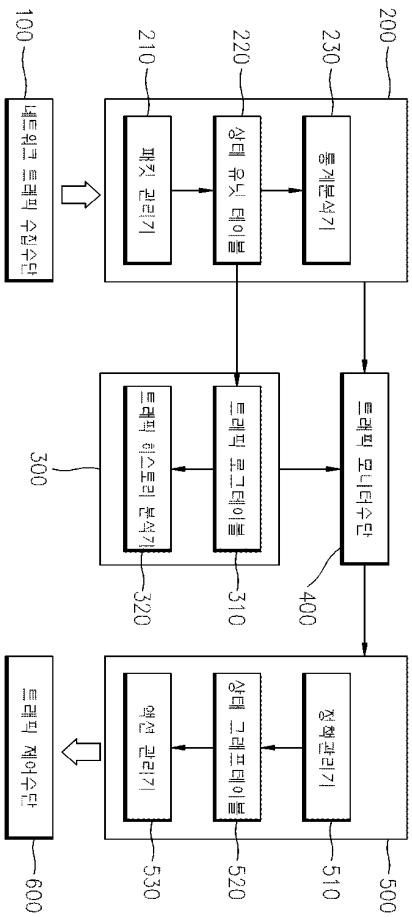
(57)요 약

본 발명은 외부 정보 통신망으로부터 유입되는 모든 패킷의 정보를 분석하여 유효성과 불법성을 분석하고, 내부 네트워크의 비정상적인 상황을 유도하는 트래픽 패턴을 인식하며, 비정상적 상황을 파악하고 네트워크 관리 정책에 따라 대응하기 위한 네트워크 상태 기반의 방화벽 장치 및 그 방법에 관한 것으로,

본 발명의 장치는, 외부 정보 통신망으로부터 수집한 네트워크 패킷을 호스트, 프로토콜, 커넥션별로 구분하여 출력하는 네트워크트래픽수집수단; 상기 네트워크트래픽수집수단에서 전송되는 상기 네트워크 패킷의 정보를 분석하고, 상태 유닛 테이블 정보를 갱신하여 트래픽 정보를 가공하는 정적상태정보수집수단; 상기 정적상태정보수집수단에서 가공된 상기 트래픽 정보의 네트워크 트래픽의 장기적인 성향과 패턴을 분석하고 기록하는 동적상태정보분석수단; 상기 정적상태정보수집수단과 상기 동적상태정보분석수단에서 각각 제공하는 정보를 이용하여 상기 네트워크트래픽의 동향을 감시하고 비정상적인 상황을 인식하여 상태 전이 신호를 출력하는 트래픽모니터수단; 및 상기 트래픽모니터수단에서 출력되는 상태 전이 신호를 받아서 생성된 상태 그래프 테이블을 이용하여 비정상적인 네트워크 상황을 정상적으로 유도하기 위한 패킷 필터링 룰을 생성하는 룰생성수단을 포함하는데 특징이 있다. 또한, 본 발명의 방법은, 게이트웨이를 통과하는 네트워크 패킷을 수집하여 상태 유닛별로 분류하는 제 1 과정; 상기 제 1 과정에서 수집된 상기 네트워크 패킷을 통계 및 패턴 분석 처리하여 현재의 트래픽 상황에 대한 정보를 생성하는 제 2 과정; 상기 제 2 과정에서 생성된 트래픽 상황에 대한 정보를 분석하고 감시하여 네트워크의 비정상적인 상황의 발생을 인지하여 상태 전이 신호를 생성하는 제 3 과정; 상기 제 3 과정에서 생성된 상태 전이 신호에 따라 상태 전이를 발생시켜, 그에 따른 정책을 재택하고 새로운 패킷 필터링 룰을 생성하는 제 4 과정; 및 상기 제 4 과정이 완료되면 현재 적용되고 있는 정책과 필터링 룰을 기록하고 상태 정보와 로그를 저장하는 제 5 과정을 포함하는데 특징이 있다.

키워드 : 트래픽, 상태 유닛, 통계 분석, 히스토리, 로그, 정책, 상태 그래프, 액션

대 표 도 - 도 2



This Facsimile Page has been artificially created from KIPO data.

청구의 범위

청구항 1

외부 정보 통신망으로부터 수집한 네트워크 패킷을 호스트, 프로토콜, 커넥션별로 구분하여 출력하는 네트워크트래픽수집수단;

상기 네트워크트래픽수집수단에서 전송되는 상기 네트워크 패킷의 정보를 분석하고, 상태 유닛 테이블 정보를 갱신하여 트래픽 정보를 가공하는 정적상태정보수집수단;

상기 정적상태정보수집수단에서 가공된 상기 트래픽 정보의 네트워크 트래픽의 장기적인 성향과 패턴을 분석하고 기록하는 동적상태정보분석수단;

상기 정적상태정보수집수단과 상기 동적상태정보분석수단에서 각각 제공하는 정보를 이용하여 상기 네트워크 트래픽의 동향을 감시하고 비정상적인 상황을 인식하여 상태 전이 신호를 출력하는 트래픽모니터수단; 및

상기 트래픽모니터수단에서 출력되는 상태 전이 신호를 받아서 생성된 상태 그래프 테이블을 이용하여 비정상적인 네트워크 상황을 정상적으로 유도하기 위한 패킷 필터링 룰을 생성하는 룰생성수단를 포함하는 것을 특징으로 하는 네트워크 상태 기반의 방화벽 장치.

청구항 2

제 1 항에 있어서, 상기 정적상태정보수집수단은

상기 네트워크 패킷의 정보를 분석하는 패킷관리기; 및

상기 패킷관리기에서 분석된 상기 네트워크 패킷의 정보에 따라 상태 유닛 테이블의 정보를 갱신하는 통계분석기를 포함함을 특징으로 하는 네트워크 상태 기반의 방화벽 장치.

청구항 3

제 1 항에 있어서, 상기 동적상태정보분석수단은

발생하는 모든 이벤트에 대한 로그를 기록하여 관리자에게 네트워크와 방화벽의 관리에 대한 참고 자료를 제공하는 트래픽로그테이블; 및

상기 트래픽로그테이블에서 제공되는 참고 자료를 받아서 상기 네트워크 트래픽의 패턴과 그에 따른 대응 방식을 기록, 분석하는 트래픽히스토리분석기를 포함함을 특징으로 하는 네트워크 상태 기반의 방화벽 장치.

청구항 4

제 1 항에 있어서, 상기 룰생성수단은

상기 트래픽모니터수단에 의하여 생성된 상태 전이 신호를 받아 상태 그래프 테이블을 생성하는 정책관리기; 및

상기 정책관리기에서 생성된 상태 그래프 테이블에서 일어난 상태 전이에 따라 현재의 상황 지시자가 새로운 상태 노드로 이동하면 채택되는 새로운 네트워크 트래픽 관리 정책에 따라 현재 적용되고 있는 패킷 데이터 룰을 제거하거나 새로운 룰을 추가하는 액션관리기를 포함함을 특징으로 하는 네트워크 상태 기반의 방화벽 장치.

청구항 5

제 1 항에 있어서,

비정상적인 네트워크 상황을 정상적으로 유도하기 위한 패킷 필터링 룰을 전달받아 네트워크의 비정상적 상황에 대응하는 트래픽제어수단을 더 포함함을 특징으로 하는 네트워크 상태 기반의 방화벽 장치.

청구항 6

게이트웨이를 통과하는 네트워크 패킷을 수집하여 상태 유닛별로 분류하는 제 1 과정;

상기 제 1 과정에서 수집된 상기 네트워크 패킷을 통계 및 패턴 분석 처리하여 현재의 트래픽 상황에 대한 정보를 생성하는 제 2 과정;

상기 제 2 과정에서 생성된 트래픽 상황에 대한 정보를 분석하고 감시하여
네트워크의 비정상적인 상황의 발생을 인지하여 상태 전이 신호를 생성하는 제 3 과정;

상기 제 3 과정에서 생성된 상태 전이 신호에 따라 상태 전이를 발생시켜, 그에 따른 정책을 채택하고 새로운 패킷 필터링 룰을 생성하는 제 4 과정; 및

상기 제 4 과정이 완료되면 현재 적용되고 있는 정책과 필터링 룰을 기록하고 상태 정보와 로그를 저장하는 제 5 과정을 포함하는 것을 특징으로 하는 네트워크 상태 기반의 방화 방법.

청구항 7

제 6 항에 있어서, 상기 제 2 과정은

수집되는 상기 네트워크 패킷을 서브 네트워크에 포함되어 있는 호스트, 프로토콜, TCP 및 UDP 커넥션별로 분류하고, 각각의 특성에 따른 정보를 차별하여 분석하는 제 1 단계;

수집되는 상기 네트워크 패킷이 이미 생성되어 있는 상태 유닛에 포함된다면 상기 네트워크 패킷의 정보를 추출하여 상태 유닛의 정보를 갱신하고, 상태 유닛 테이블에 존재하지 않는 것이라면 상기 네트워크 패킷의 정보에 따라 상태 유닛을 생성하는 제 2 단계;

상기 네트워크 패킷의 시간적, 논리적 상관 관계를 트래픽 히스토리 정보 테이블을 이용하여 유효성을 검사하고 논리적 오류가 발견되면 상기 네트워크 패킷을 패기하는 제 3 단계;

트래픽의 전체적인 형태를 파악하기 위하여 유효성이 검사된 상기 네트워크

패킷의 정보를 대상으로 통계 정보를 생성하는 제 4 단계;

트래픽의 전체적 형태와 장기간적 동향을 파악하기 위하여 트래픽 패턴의 기록 및 그에 따른 대응 형태 등을 트래픽 히스토리 테이블에 기록하는 제 5 단계;

네트워크와 방화벽의 관리에 참조하기 위하여 상기 네트워크 패킷의 유효성 검사와 통계 정보 생성, 상태 유닛 테이블의 관리에 수반되는 로그를 로그 테이블에 기록하는 제 6 단계; 및

네트워크 상태의 비정상적 상황을 인식하기 위한 트래픽모니터수단에 트래픽 상태 정보를 제공하는 제 7 단계를 포함하는 것을 특징으로 하는 네트워크 상태 기반의 방화 방법.

청구항 8

제 6 항에 있어서, 상기 제 3 과정은

정적상태정보수집수단으로부터 제공되는 트래픽 상태 정보에서 정적으로 설정되어 있는 트래픽 정상 조건의 보정을 위한 평균 트래픽 정보를 추출하는 제 1 단계;

상기 제 1 단계에서 추출된 평균 트랙픽 정보를 이용하여 트래픽 정상 조건을 보정하는 제 2 단계;

트래픽 정상 조건과 트래픽 상태 정보를 비교하여 현재 네트워크의 상황의 비정상 여부를 판단하는 제 3 단계;

네트워크의 상태가 비정상적 상황임이 판명될 경우 새로운 트래픽 제어 정책의 채택을 위한 상태 전이 신호를 생성하는 제 4 단계; 및

상기 제 4 단계에서 생성된 상태 전이 신호를 룰생성수단의 상태 그래프에 전달하는 제 5 단계를 포함함을 특징으로 하는 네트워크 상태 기반의 방화 방법.

청구항 9

제 6 항에 있어서, 상기 제 4 과정은 트래픽모니터수단에서 전달된 상태 전이 신호에서 현재 네트워크 상태에 대한 정보를 추출하고, 비정상적인 요인을 추출하여 상태 그래프 테이블에 적용하는 제 1 단계;

상태 그래프 테이블에서 현재 상태 노드에서 전이 가능한 노드를 선별하여, 그 중에 상태 전이 신호의 정보를 수용할 수 있는 노드를 추출한 후 상태 전이를 수행하는 제 2 단계;

상태 그래프 테이블을 참조하여 새로운 상태 노드의 트래픽 제어 정책을 채택하고, 그에 따른 패킷 필터링 룰을 생성하는 제 3 단계; 및

상기 제 3 단계에서 새롭게 생성된 패킷 필터링 룰을 트래픽제어수단에 전달하여 네트워크의 비정상적 상황에 대응하는 제 4 단계를 포함함을 특징으로 하는 네트워크 상태 기반의 방화 방법.

명세서

발명의 명칭

네트워크 상태 기반의 방화벽 장치 및 그 방법{Apparatus for firewall of network status based Method thereof}

도면의 간단한 설명

[0001] 도 1은 일반적인 홈 네트워크 구조에서의 방화벽 배치도,

[0002] 도 2는 본 발명에 따른 네트워크 상태 기반의 방화벽 장치의 구성도,

[0003] 도 3은 본 발명에 따른 네트워크 상태 기반의 방화벽 장치를 구현하는 방법의 전체 흐름도,

[0004] 도 4는 도 3에 도시된 네트워크 상태 기반의 방화벽 장치를 구현하는 방법의 상태 정보 수집을 위한 흐름도,

[0005] 도 5는 도 3에 도시된 네트워크 상태 기반의 방화벽 장치를 구현하는 방법의 상태 전이 신호 생성을 위한 흐름도,

[0006] 도 6은 도 3에 도시된 네트워크 상태 기반의 방화벽 장치를 구현하는 방법의 패킷 필터링 룰 생성을 위한 흐름도.

[0007] <도면의 주요 부분에 대한 부호의 설명>

[0008] 100 : 네트워크트래픽수집수단 200 : 정적상태정보수집수단

[0009] 300 : 동적상태정보분석수단 400 : 트래픽모니터수단

[0010] 500 : 룰생성수단 600 : 트래픽제어수단

발명의 상세한 설명

[0011] 본 발명은 네트워크 상태 기반의 방화벽 장치 및 그 방법에 관한 것으로, 보다 상세하게는 외부 정보 통신망으로부터 유입되는 모든 패킷의 정보를 분석하여 유효성과 불법성을 분석하고, 내부 네트워크의 비정상적인 상황을 유도하는 트래픽 패턴을 인식하며, 비정상적 상황을 파악하고 네트워크 관리 정책에 따라 대응하기 위한 네트워크 상태 기반의 방화벽 장치 및 그 방법에 관한 것이다.

[0012] 최근에는 네트워크가 가정으로까지 확산되어 블루투스, 하비, 지니와 같은 홈네트워킹에 필요한 제반 기술 및 응용 분야가 개발되고 있다. 이와 같은 홈네트워크가 네트워크 분야에서 한 주류를 형성하면서 이질적인 기기들의 집합체인 홈네트워크를 제어, 관리하기 위한 에이전트가 필요한데 반하여, 에이전트로의 불법적 접근은 가정의 안전을 해칠 수도 있다. 이러한 기업, 가정 및 그 밖의 네트워크에서 외부로부터의 안전을 지키기 위해서는 네트워크의 특성에 따라 침입탐지, 침입차단, 역추적, 바이러스 방지 등의 많은 기술을 요구한다. 그 중 가장 기본적이고 모든 네트워크에 적용할 수 있는 것이 침입 차단 기술을 이용한 방화벽 시스템인 바, 방화벽은 외부로부터 내부 네트워크 자원을 보호하기 위해 필수적인 요소이며, 네트워크 내부의 구조와는 관계없이 외부와 내부의 접점에서 내부로의 패킷 흐름을 통제하고, 내부 사용자의 통신에 투명성을 제공하여야 한다. 도 1은 일반적인 홈 네트워크 구조에서의 방화벽 배치도

로서, 홈 네트워크 구조는 액세스 네트워크(10), 홈 게이트웨이 및 방화벽(11), 홈서버(12), PLC(13), DTV(14),
IEEE1394카메라(15), DVCR(16), 디지털오디오(17), 게임기(18), 수도(19), 보안기(20), 불루투스카메라(21), 팬(Fan)(22), 전등(23), 세탁기(24), 냉장고(25), 윈도우/커튼(26), 웹패드(27), PDA(28), 휴대폰(29), 화상전화기(30), 프린터(31), 스캐너(32), 팩스(33), PC(34), 콘센트(35) 등으로 구성되는데, 방화벽(11)은 그 발전 과정에 따라 세대별로 구분할 수 있는데, 제 1 세대 방화벽인 패킷 필터링은 각 패킷의 헤더 정보들에 기반하여 네트워크 사이의 전달을 허가하거나 거부하는 프로세스이다. 이와 같은 패킷 필터링은 한 서비스가 외부 네트워크에 항상 열려 있기 때문에 정적 상태를 갖으며, 장점으로는 오버헤드가 적고, 높은 성능을 보이며 구현하는데 비용이 적거나 들지 않고 트래픽 관리가 쉬운데 반하여, 단점으로는 서비스를 위해 열려있는 포트가 외부 클라이언트로부터 직접적인 연결을 허가하며, 서비스를 제공하는 포트가 외부 클라이언트에게 항상 열려있고, 복잡한 환경에서 신속한 관리가 어려우며, 특정 IP를 막더라도 IP 눈속임으로 공격당하기 쉽고, 사용자 인증과 투명성을 제공하지 않는다. 또한, 제 1 세대 방화벽과 비교되는 제 2 세대 방화벽의 가장 큰 차이점은 제 1 세대 방화벽에서는 방화벽(11)쪽의 서버 프로그램이 클라이언트쪽의 클라이언트 소프트웨어와 통신하면서 클라이언트의 존재를 감지해야만 하는 바, 서버(12)와 클라이언트들 각각에 대해 소프트웨어를 설치해야하는 번거러움이 있으며, 방화벽은 인지된 클라이언트들에 대해서만 방화벽 기능을 제공하는데 반하여, 제 2 세대 방화벽은 이러한 단점을 해결하여 클라이언트에 존재를 감지할 필요 없이 내부 호스트들에 대한 방화벽 기능을 수행하는 투명성을 제공하며, 또 하나의 제 1 세대 방화벽의 단점인 지속적 포트 개방
으로 인한 공격자의 공격 기회를 줄이기 위해 동적으로 제어하도록 설계되는데, 이와 같은 제 2 세대 방화벽 역시 제 1 세대 방화벽과 마찬가지로 사용자의 인증을 제공할 수 없고 IP 눈속임 공격에 대응하지 못하는 문제점이 있다.

[0013] 따라서, 본 발명은 이와 같은 문제점을 해결하기 위한 것으로서, 본 발명의 목적은 네트워크 트래픽의 상태를 다양한 관점에서 분석하고 상태 그래프를 통하여 상황에 대한 인식과 함께 트래픽 제어 정책을 차별적으로 반영함으로써, 외부 정보 통신망을 통하여 내부 네트워크로 유입되는 모든 불법적인 패킷 또는 패킷의 연속으로 이루어지는 트래픽을 차단하고, 내부 네트워크의 비정상적인 상황을 초래하는 패킷 흐름을 제어하여 원활하고 안전한 네트워크 상태를 유지할 수 있는 네트워크 상태 기반의 방화벽 장치 및 그 방법을 제공하는데 있다.

[0014] 본 발명의 다른 목적은 네트워크의 전반적인 상황을 파악할 수 있는 다양한 정보 및 로그를 제공하는 네트워크 상태 기반의 방화벽 장치 및 그 방법을 제공하는데 있다.

[0015] 이와 같은 목적을 달성하기 위한 본 발명의 특징은, 외부 정보 통신망으로부터 수집한 네트워크 패킷을 호스트, 프로토콜, 커넥션별로 구분하여 출력하는 네트워크트래픽수집수단; 상기 네트워크트래픽수집수단에서 전송되는 상기 네트워크 패킷의 정보를 분석하고, 상태 유닛 테이블 정보를 갱신하여 트래픽 정보를 가공하는 정적상태정보수집수단; 상기 정적상태정보수집수단에서 가공된 상기 트래픽 정보의
네트워크 트래픽의 장기적인 성향과 패턴을 분석하고 기록하는 동적상태정보분석수단; 상기 정적상태정보수집수단과 상기 동적상태정보분석수단에서 각각 제공하는 정보를 이용하여 상기 네트워크 트래픽의 동향을 감시하고 비정상적인 상황을 인식하여 상태 전이 신호를 출력하는 트래픽모니터수단; 및 상기 트래픽모니터수단에서 출력되는 상태 전이 신호를 받아서 생성된 상태 그래프 테이블을 이용하여 비정상적인 네트워크 상황을 정상적으로 유도하기 위한 패킷 필터링 룰을 생성하는 룰생성수단를 포함하는데 있다.

[0016] 바람직하게, 상기 정적상태정보수집수단은 상기 네트워크 패킷의 정보를 분석하는 패킷관리기; 및 상기 패킷관리기에서 분석된 상기 네트워크 패킷의 정보에 따라 상태 유닛 테이블의 정보를 갱신하는 통계분석기를 포함함을 특징으로 한다.

[0017] 바람직하게, 상기 동적상태정보분석수단은 발생하는 모든 이벤트에 대한 로그를 기록하여 관리자에게 네트워크와 방화벽의 관리에 대한 참고 자료를 제공하는 트래픽로그테이블; 및 상기 트래픽로그테이블에서 제공되는 참고 자료를 받아서 상기 네트워크 트래픽의 패턴과 그에 따른 대응 방식을 기록, 분석하는 트래픽히스토리분석기를 포함함을 특징으로 한다.

[0018] 바람직하게, 상기 룰생성수단은 상기 트래픽모니터수단에 의하여 생성된 상태 전이 신호를 받아 상태 그래프 테이블을 생성하는 정책관리기; 및 상기 정책관리기에서 생성된 상태 그래프 테이블에서 일어난 상태 전이에 따라 현재의 상황 지시자가 새로운 상태 노드로 이동하면 채택되는 새로운 네트워크 트래픽 관리 정책에 따라 현재 적용되고 있는 패킷 데이터 룰을 제거하거나 새로운 룰을 추가하는
액션관리기를 포함함을 특징으로 한다.

[0019] 바람직하게, 비정상적인 네트워크 상황을 정상적으로 유도하기 위한 패킷 필터링 룰을 전달받아 네트워크의 비정상적 상황에 대응하는 트래픽제어수단을 더 포함함을 특징으로 한다.

[0020] 본 발명의 다른 특징은, 게이트웨이를 통과하는 네트워크 패킷을 수집하여 상태 유닛별로 분류하는 제 1 과정; 상기 제 1 과정에서 수집된 상기 네트워크 패킷을 통계 및 패턴 분석 처리하여 현재의 트래픽 상황에 대한 정보를 생성하는 제 2 과정; 상기 제 2 과정에서 생성된 트래픽 상황에 대한 정보를 분석하고 감시하여 네트워크의 비정상적인 상황의 발생을 인지하여 상태 전이 신호를 생성하는 제 3 과정; 상기 제 3 과정에서 생성된 상태 전이 신호에 따라 상태 전이를 발생시켜, 그에 따른 정책을 채택하고 새로운 패킷 필터링 룰을 생성하는 제 4 과정; 및 상기 제 4 과정이 완료되면 현재 적용되고 있는 정책과 필터링 룰을 기록하고 상태 정보와 로그를 저장하는 제 5 과정을 포함하는데 있다.

[0021] 바람직하게, 상기 제 2 과정은 수집되는 상기 네트워크 패킷을 서브 네트워크에 포함되어 있는 호스트, 프로토콜, TCP 및 UDP 커넥션별로 분류하고, 각각의 특성에 따른 정보를 차별하여 분석하는 제 1 단계; 수집되는 상기 네트워크 패킷이 이미 생성되어 있는 상태 유닛에 포함된다면 상기 네트워크 패킷의 정보를 추출하여 상태 유닛의 정보를 갱신하고, 상태 유닛 테이블에 존재하지 않는 것이라면 상기 네트워크 패킷의 정보에 따라 상태 유닛을 생성하는 제 2 단계; 상기 네트워크 패킷의 시간적, 논리적 상관 관계를 트래픽 히스토리 정보 테이블을 이용하여 유효성을 검사하고 논리적 오류가 발견되면 상기 네트워크 패킷을 패기하는 제 3 단계; 트래픽의 전체적인 형태를 파악하기 위하여 유효성이 검사된 상기 네트워크 패킷의 정보를 대상으로 통계 정보를 생성하는 제 4 단계; 트래픽의 전체적 형태와 장기간적 동향을 파악하기 위하여 트래픽 패턴의 기록 및 그에 따른 대응 형태 등을 트래픽 히스토리 테이블에 기록하는 제 5 단계; 네트워크와 방화벽의 관리에 참조하기 위하여 상기 네트워크 패킷의 유효성 검사와 통계 정보 생성, 상태 유닛 테이블의 관리에 수반되는 로그를 로그 테이블에 기록하는 제 6 단계; 및 네트워크 상태의 비정상적 상황을 인식하기 위한 트래픽모니터수단에 트래픽 상태 정보를 제공하는 제 7 단계를 포함하는데 있다.

[0022] 바람직하게, 상기 제 3 과정은 정적상태정보수집수단으로부터 제공되는 트래픽 상태 정보에서 정적으로 설정되어 있는 트래픽 정상 조건의 보정을 위한 평균 트래픽 정보를 추출하는 제 1 단계; 상기 제 1 단계에서 추출된 평균 트랙픽 정보를 이용하여 트래픽 정상 조건을 보정하는 제 2 단계; 트래픽 정상 조건과 트래픽 상태 정보를 비교하여 현재 네트워크의 상황의 비정상 여부를 판단하는 제 3 단계; 네트워크의 상태가 비정상적 상황임이 판명될 경우 새로운 트래픽 제어 정책의 채택을 위한 상태 전이 신호를 생성하는 제 4 단계; 및 상기 제 4 단계에서 생성된 상태 전이 신호를 룰생성수단의 상태 그래프에 전달하는 제 5 단계를 포함함을 특징으로 한다.

[0023] 바람직하게, 상기 제 4 과정은 트래픽모니터수단에서 진달된 상태 전이 신호에서 현재 네트워크 상태에 대한 정보를 추출하고, 비정상적인 요인을 추출하여 상

태 그래프 테이블에 적용하는 제 1 단계; 상태 그래프 테이블에서 현재 상태 노드에서 전이 가능한 노드를 선별하여, 그 중에 상태 전이 신호의 정보를 수용할 수 있는 노드를 추출한 후 상태 전이를 수행하는 제 2 단계; 상태 그래프 테이블을 참조하여 새로운 상태 노드의 트래픽 제어 정책을 채택하고, 그에 따른 패킷 필터링 룰을 생성하는 제 3 단계; 및 상기 제 3 단계에서 새롭게 생성된 패킷 필터링 룰을 트래픽제어수단에 전달하여 네트워크의 비정상적 상황에 대응하는 제 4 단계를 포함함을 특징으로 한다.

[0024] 이하, 본 발명의 바람직한 실시예에 대하여 첨부 도면을 참조하여 상세히 설명한다.

[0025] 도 2는 본 발명에 따른 네트워크 상태 기반의 방화벽 장치의 구성도이다.

[0026] 도 2에 있어서, 본 발명은 네트워크트래픽수집수단(100), 정적상태정보수집수단(200), 동적상태정보분석수단(300), 트래픽모니터수단(400), 룰생성수단(500), 트래픽제어수단(600)으로 구성된다.

[0027] 네트워크트래픽수집수단(100)은 도 1의 액세스네트워크(10)인 외부 정보 통신망으로부터 수집한 네트워크 패킷을 호스트, 프로토콜, 커넥션별로 구분하여 출력하며, 정작상태정보수집수단(200)은 네트워크 패킷의 정보를 분석하는 패킷관리기(210) 및 패킷관리기(210)에서 분석된 네트워크 패킷의 정보에 따라 상태 유닛 테이블(220)의 정보를 갱신하는 통계분석기(230)를 포함하여, 네트워크트래픽수집수단(100)에서 전송되는 네트워크 패킷의 정보를 분석하고, 상태 유닛 테이블 정보를 갱신하여 트래픽 정보를 가공한다.

[0028] 동적상태정보분석수단(300)은 발생하는 모든 이벤트에 대한 로그를 기록하여 관리자에게 네트워크와 방화벽의 관리에 대한 참고 자료를 제공하는 트래픽로그테이블(310) 및 트래픽로그테이블(310)에서 제공되는 참고 자료를 받아서 네트워크 트래픽의 패턴과 그에 따른 대응 방식을 기록, 분석하는 트래픽히스토리분석기(320)를 포함하여, 정적상태정보수집수단(200)에서 가공된 트래픽 정보의 네트워크 트래픽의 장기적인 성향과 패턴을 분석하고 기록하며, 트래픽모니터수단(400)은 정적상태정보수집수단(200)과 동적상태정보분석수단(300)에서 각각 제공하는 정보를 이용하여 네트워크 트래픽의 동향을 감시하고 비정상적인 상황을 인식하여 상태 전이 신호를 출력한다.

[0029] 한편, 룰생성수단(500)은 트래픽모니터수단(400)에 의하여 생성된 상태 전이 신호를 받아 상태 그래프를 생성하는 정책관리기(510) 및 정책관리기(510)에서 생성된 상태 그래프 테이블(520)에서 일어난 상태 전이에 따라 현재의 상황 지시자가 새로운 상태 노드로 이동하면 채택되는 새로운 네트워크 트래픽 관리 정책에 따라 현재 적용되고 있는 패킷 데이터 룰을 제거하거나 새로운 룰을 추가하는 액션관리기(530)를 포함하여, 트래픽모니터수단(400)에서 출력되는 상태 전이 신호를 받아서 생성된 상태 그래프를 이용하여 비정상적인 네트워크 상황을 정상적으로 유도하기 위한 패킷 필터링 룰을 생성하여 트래픽제어수단(600)에 전달하여, 트래픽제어수단(600)으로 하여금 네트워크의 비정상적 상황에 대응하도록 한다.

[0030] 도 3은 본 발명에 따른 네트워크 상태 기반의 방화벽 장치를 구현하는 방법의 전체 흐름도이다.

[0031] 도 3에 있어서, 도 1에 도시된 게이트웨이(11)를 통과하는 네트워크 패킷을 수집하여 상태 유닛별로 분류하여(단계 S100), S100 단계에서 수집된 네트워크 패킷의 상태 정보를 수집하여, 통계 및 패턴 분석 처리하고 현재의 트래픽 상황에 대한 정보를 생성한다(단계 200). S200 단계에서 생성된 트래픽 상황에 대한 정보를 분석하고 감시하여 네트워크의 비정상적인 상황의 발생을 인지하여 상태 전이 신호를 생성하며(단계 S300), S300 단계에서 생성된 상태 전이 신호에 따라 상태 전이를 발생시켜, 그에 따른 정책을 채택하고 새로운 패킷 필터링 룰을 생

성한다(단계 S400). S400 단계에서 새로운 패킷 필터링 룰이 생성되면 현재 적용되고 있는 정책과 필터링 룰을 기록하고 상태 정보와 로그를 저장한다(단계 S500).

[0032] 도 4는 도 3에 도시된 네트워크 상태 기반의 방화벽 장치를 구현하는 방법의 상태 정보 수집을 위한 흐름도이다.

[0033] 도 4에 있어서, 수집되는 네트워크 패킷을 서브 네트워크에 포함되어 있는 호스트, 프로토콜, TCP 및 UDP 커넥션별로 분류하고, 각각의 특성에 따른 정보를 차별하여 분석한다(단계 S201). S201 단계에서 네트워크 정보를 분석한 후에, 수집되는 네트워크 패킷이 이미 생성되어 있는 상태 유닛에 포함된다면 네트워크 패킷의 정보를 추출하여 상태 유닛의 정보를 갱신하고, 상태 유닛 테이블(220)에 존재하지 않는 것이라면 네트워크 패킷의 정보에 따라 상태 유닛을 생성한다(단계 S202).

[0034] S202 단계에서 상태 유닛이 생성되면 네트워크 패킷의 시간적, 논리적 상관 관계를 트래픽 히스토리분석기(320)에 내장된 트래픽 히스토리 정보 테이블을 이용
하여 유효성을 검사하고(단계 S203), S203 단계에서 논리적 오류가 발견되는가에 따라 유효한가 판단한다(단계 S204). S204 단계에서 유효하지 않으면 네트워크 패킷을 패기하고(단계 S205), S204 단계에서 유효하면 트래픽의 전체적인 형태를 파악하기 위하여 유효성이 검사된 상기 네트워크 패킷의 정보를 대상으로 통계 정보를 생성한다(단계 206).

[0035] S206 단계에서 통계 정보를 처리하면 트래픽의 전체적 형태와 장기간적 동향을 파악하기 위하여 트래픽 패턴의 기록 및 그에 따른 대응 형태 등을 트래픽히스토리분석기(320)에 내장된 트래픽 히스토리 테이블에 기록하고(단계 S207), 네트워크와 방화벽의 관리에 참조하기 위하여 네트워크 패킷의 유효성 검사와 통계 정보 생성, 상태 유닛 테이블의 관리에 수반되는 로그를 트래픽 로그 테이블(310)에 기록한다(단계 S208). S208 단계에서 상태 정보 생성 로그 처리가 되면 네트워크 상태의 비정상적 상황을 인식하기 위한 트래픽모니터수단(400)에 트래픽 상태 정보를 제공한다(단계 S209).

[0036] 도 5는 도 3에 도시된 네트워크 상태 기반의 방화벽 장치를 구현하는 방법의 상태 전이 신호 생성을 위한 흐름도이다.

[0037] 도 5에 있어서, 정적상태정보수집수단(200)으로부터 제공되는 트래픽 상태 정보에서 정적으로 설정되어 있는 트래픽 정상 조건의 보정을 위한 평균 트래픽 정보를 추출하여(단계 S301), S301 단계에서 수집, 추출된 평균 트랙픽 정보를 이용하여 트래픽 정상 조건을 보정한다(단계 S302).

[0038] S302 단계에서 트래픽 정상 조건이 보정되면 트래픽 비정상 검사를 수행함으
로써(단계 S303), 트래픽 정상 조건과 트래픽 상태 정보를 비교하여 현재 네트워크의 상황의 비정상 여부를 판단한다(단계 S304). S304 단계에서 네트워크의 상태가 비정상적 상황이 아니면 S301 단계로 복귀하고, S304 단계에서 네트워크의 상태가 비정상적 상황임이 판명될 경우 새로운 트래픽 제어 정책의 채택을 위한 상태 전이 신호를 생성하여 룰생성수단(500)의 상태 그래프(520)에 전달한다(단계 S305).

[0039] 도 6은 도 3에 도시된 네트워크 상태 기반의 방화벽 장치를 구현하는 방법의 패킷 필터링 룰 생성을 위한 흐름도이다.

[0040] 도 6에 있어서, 트래픽모니터수단(400)에서 룰생성수단(500)에 전달된 상태 전이 신호를 분석하고(단계 S401), S401 단계에서 상태 전이 신호가 분석되면 현재 네트워크 상태에 대한 정보를 추출하여(단계 S402), 비정상적인 요인을 추출하여 상태 그래프 테이블에 적용한다. S402 단계에서 트래픽 상황 정보가 추출되면 상태 그래프 테이블(520)에서 현재 상태 노

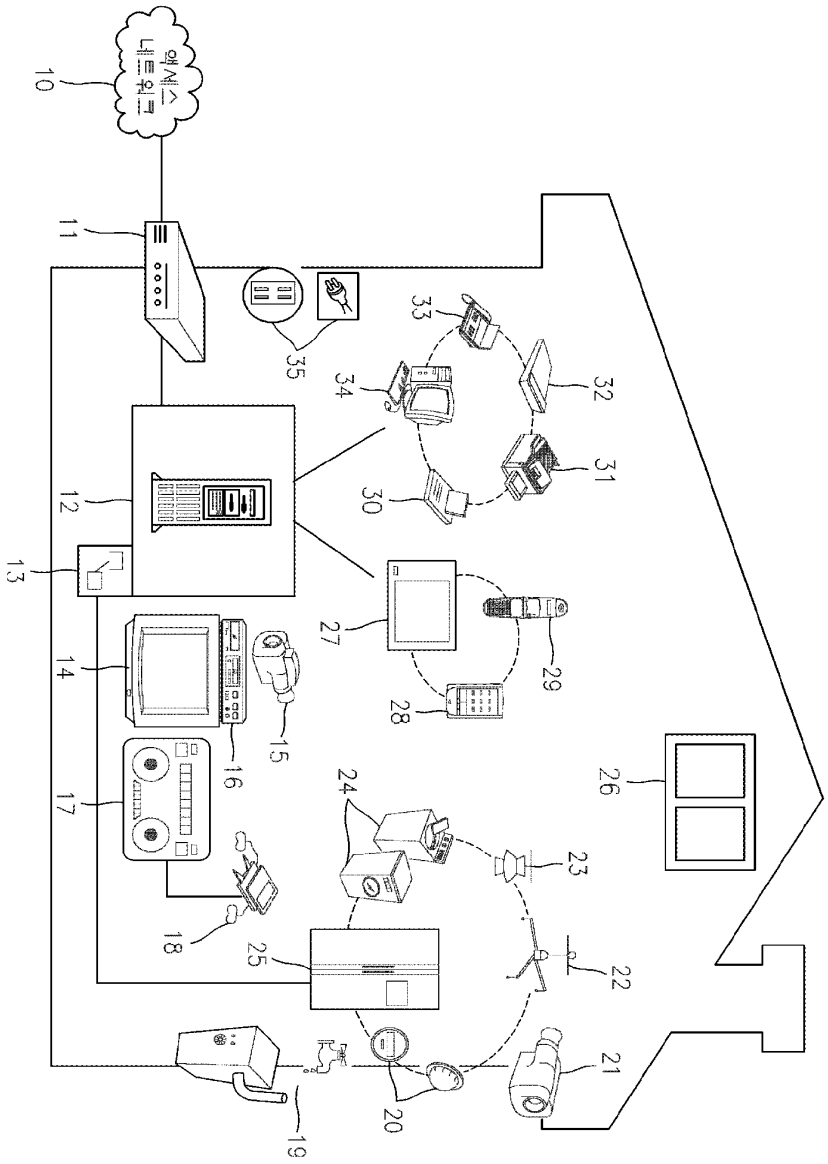
드에서 전이 가능한 노드를 선별하여(단계 S403), 그 중에 상태 전이 신호의 정보를 수용할 수 있는 노드를 추출한 후 상태 전이가 가능한가 판단한다(단계 S404).

[0041] S404 단계에서 상태 전이가 가능하면 상태 전이를 수행하여(단계 S405), 상태 그래프 테이블(520)을 참조하여 새로운 상태 노드의 트래픽 제어 정책을 채택하고, 그에 따른 패킷 필터링 룰을 생성한다(단계 S406). S406 단계에서 새롭게 생성된 패킷 필터링 룰을 트래픽제어수단(600)에 전달하여 네트워크의 비정상적 상황에 대응하기 위하여 트래픽 제어를 수행한다(단계 S407).

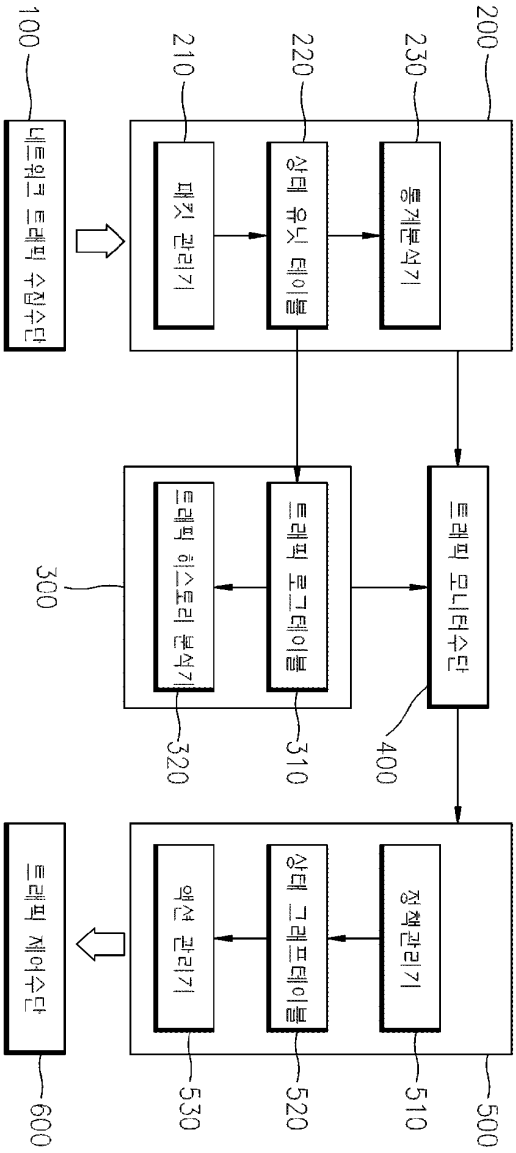
[0042] 이와 같이, 본 발명의 상세한 설명에서는 구체적인 실시예에 관해 설명하였으나, 본 발명의 범주에서 벗어나지 않는 한도 내에서 여러가지 변형이 가능함은 물론이다. 그러므로, 본 발명의 범위는 설명된 실시예에 국한되어 정해져서는 안되며 후술하는 특허청구범위 뿐만 아니라 이 특허청구범위와 균등한 것들에 의해 정해져야 한다.

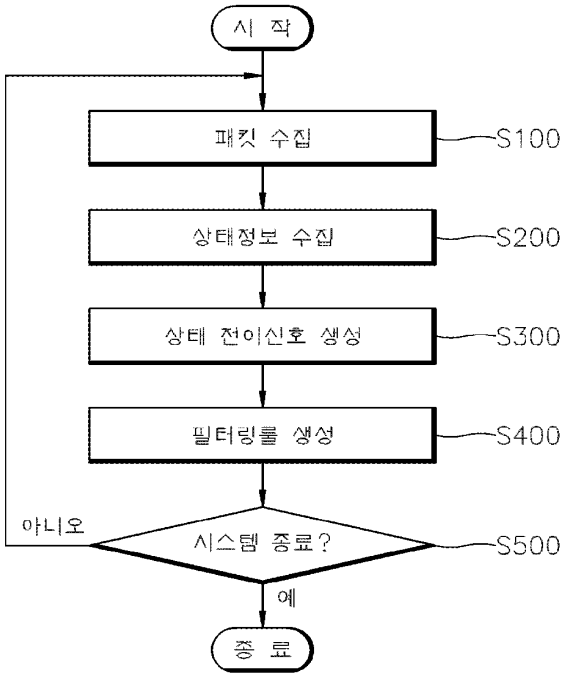
[0043] 결국, 본 발명에 의한 네트워크 상태 기반의 방화벽 장치 및 그 방법에 따르면 다음과 같은 이점이 발생한다.

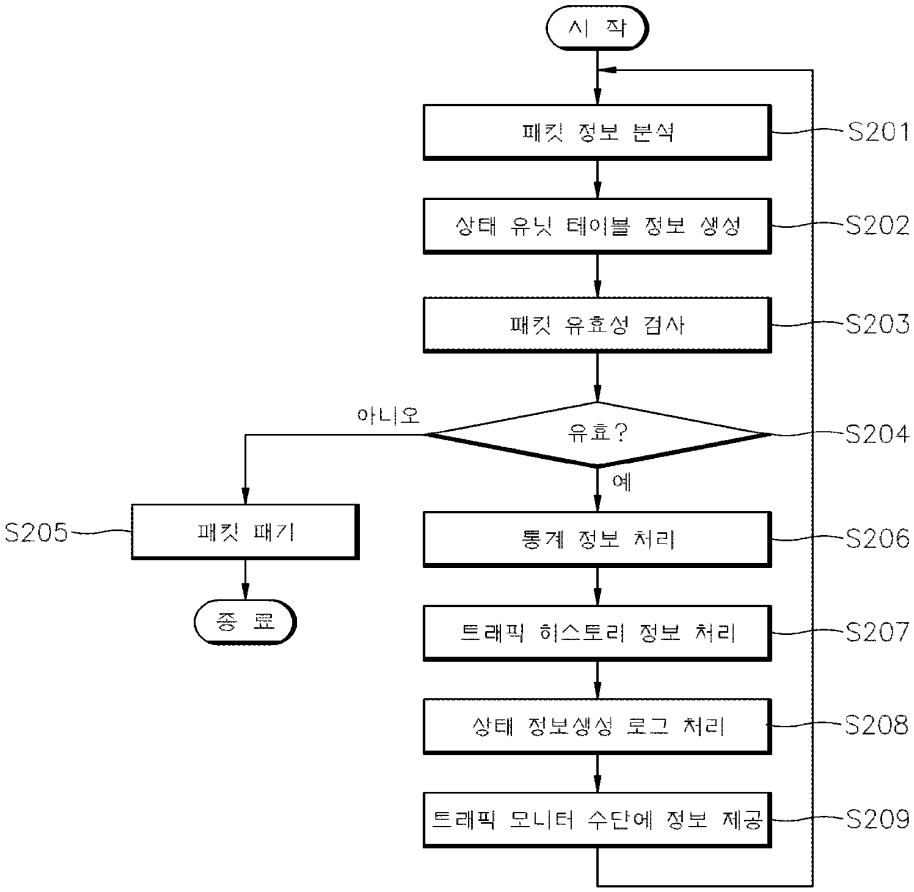
[0044] 즉, 불법적인 패킷 조작을 통한 해킹 및 침입 시도를 차단하고 비정상적 패킷 흐름을 이용하여 내부 네트워크의 혼란을 일으키는 서비스 거부 공격과 같은 공격에 신속하고 효과적으로 대응함으로써 내부 호스트와 네트워크의 안전 및 신뢰성을 보장할 수 있다.

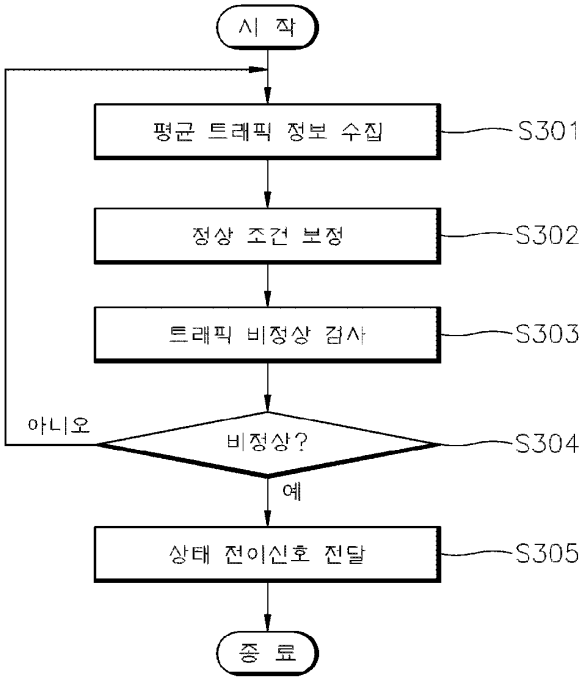


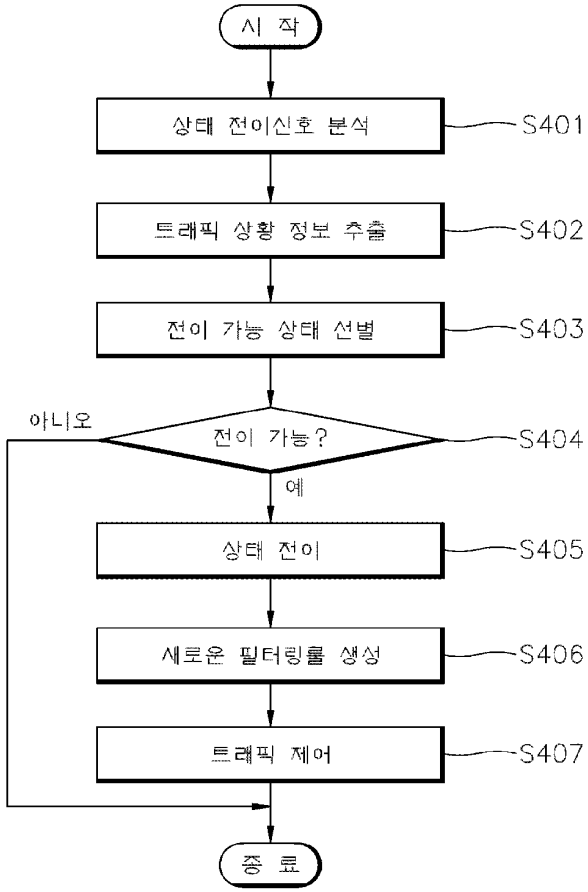
IPR2023-01353 Page 00631


100
네트워크 트래픽 수집수단
200
210
패킷 관리기
220
상태 유닛 테이블
230
통계분석기
300
310
트래픽 로그테이블
320
트래픽 히스토리 분석기
400
트래픽 모니터수단
500
510
정책관리기
520
상태 그래프테이블
530
액션 관리기
600
트래픽 제어수단


시 작
패킷 수집
S100
상태정보 수집
S200
상태 전이신호 생성
S300
필터링룰 생성
S400
시스템 종료?
S500
아니오
예
종 료


시 작
패킷 정보 분석
S201
상태 유닛 테이블 정보 생성
S202
패킷 유효성 검사
S203
유효?
S204
아니오
예
S205
패킷 폐기
종 료
통계 정보 처리
S206
트래픽 히스토리 정보 처리
S207
상태 정보생성 로그 처리
S208
트래픽 모니터 수단에 정보 제공
S209


시 작
평균 트래픽 정보 수집
S301
정상 조건 보정
S302
트래픽 비정상 검사
S303
아니오
비정상?
S304
예
상태 전이신호 전달
S305
종 료


시 작
상태 전이신호 분석
S401
트래픽 상황 정보 추출
S402
전이 가능 상태 선별
S403
아니오
전이 가능?
S404
예
상태 전이
S405
새로운 필터링률 생성
S406
트래픽 제어
S407
종 료

Espacenet

Bibliographic data: EP1006701 (A2) — 2000-06-07

Adaptive re-ordering of data packet filter rules

Inventor(s): KRISHNAN P [US]; RAZ DANNY [US]; SUGLA BINAY [US] ± (KRISHNAN, P, ; RAZ, DANNY, ; SUGLA, BINAY)

Applicant(s): LUCENT TECHNOLOGIES INC [US] ± (LUCENT TECHNOLOGIES INC)

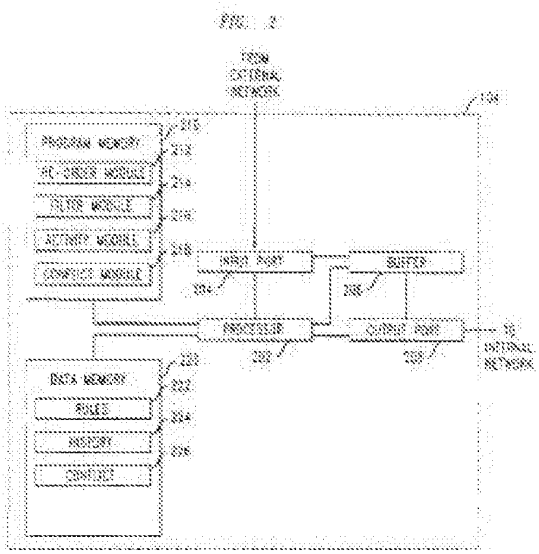
Classification: **- international:** ***G06F13/00; G06F17/30; H04L29/06;*** (IPC1-7): H04L29/06
- cooperative: **H04L63/0227**; **H04L63/0263**

Application number: EP19990309593 19991130

Priority number (s): US19980204464 19981203

Also published as: EP1006701 (A3) EP1006701 (B1) CA2287689 (A1) CA2287689 (C) US2003051165 (A1) more

Abstract of EP1006701 (A2)



A packet data filter which stores ordered rules and sequentially applies the rules to received data packets to determine the disposition of the data packet. The packet filter maintains a match count in memory which indicates the number of times each rule matched an incoming data packet. Periodically, at the initiation of a user, or based on operating parameters of the filter, the rules are automatically re-ordered based on the match count. As a result of the re-ordering, rules with higher match counts are moved earlier in the sequential evaluation order and rules with lower match counts are moved later in the sequential evaluation order. As such, rules which are more likely to match incoming data packets are evaluated earlier, thus avoiding the evaluation of later rules.; In order to prevent a re-ordering which would change the overall security policy of the packet filter, pairs of rules are compared to determine if they conflict (i.e., the swapping of the two rules would result in a change in the overall security policy). During re-ordering, the swapping of conflicting rules is prevented.

(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) EP 1 006 701 A2

(12) EUROPEAN PATENT APPLICATION

(43) Date of publication:
07.06.2000 Bulletin 2000/23

(51) Int Cl.[7]: **H04L 29/06**

(21) Application number: **99309593.4**

(22) Date of filing: **30.11.1999**

(84) Designated Contracting States:
AT BE CH CY DE DK ES FI FR GB GR IE IT LI LU MC NL PT SE
Designated Extension States:
AL LT LV MK RO SI

(30) Priority: **03.12.1998 US 204464**

(71) Applicant: **LUCENT TECHNOLOGIES INC.**
Murray Hill, New Jersey 07974-0636 (US)

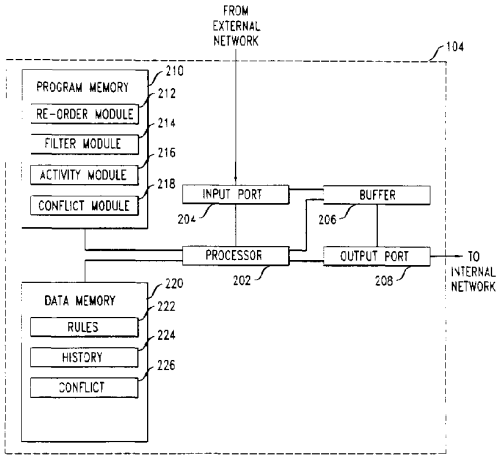
(72) Inventors:
- **Krishnan, P.**
  Matawan, New Jersey 07747 (US)
- **Raz, Danny**
  Aberdeen, New Jersey 07747 (US)
- **Sugla, Binay**
  Aberdeen, New Jersey 07747 (US)

(74) Representative:
Buckley, Christopher Simon Thirsk et al
Lucent Technologies (UK) Ltd,
5 Mornington Road
Woodford Green, Essex IG8 0TU (GB)

(54) Adaptive re-ordering of data packet filter rules

(57) A packet data filter which stores ordered rules and sequentially applies the rules to received data packets to determine the disposition of the data packet. The packet filter maintains a match count in memory which indicates the number of times each rule matched an incoming data packet. Periodically, at the initiation of a user, or based on operating parameters of the filter, the rules are automatically re-ordered based on the match count. As a result of the re-ordering, rules with higher match counts are moved earlier in the sequential evaluation order and rules with lower match counts are moved later in the sequential evaluation order. As such, rules which are more likely to match incoming data packets are evaluated earlier, thus avoiding the evaluation of later rules. In order to prevent a re-ordering which would change the overall security policy of the packet filter, pairs of rules are compared to determine if they conflict (i.e., the swapping of the two rules would result in a change in the overall security policy). During re-ordering, the swapping of conflicting rules is prevented.

FIG. 2



EP 1 006 701 A2

Printed by Jouve, 75001 PARIS (FR)

Description

Field of the Invention

[0001] The present invention relates to data packet filters. More particularly, the present invention relates to adaptive re-ordering of data packet filter rules to improve the performance of the filter while maintaining a security policy.

Background of the Invention

[0002] Data networks are becoming increasingly prevalent. For example, many companies have internal data networks to allow for intra-company communication. Such communication includes email, documents, voice, video, and multimedia. Further, these internal data networks are generally connected to an external data network (e.g. Internet), to allow for the exchange of information between the internal and external networks. However, as a result of the interconnection of data networks, security has become a major concern. The unauthorized access to a company's data network can result in the loss of valuable proprietary information. Further, unauthorized access can also result in attacks on the company's data network and computing system, resulting in loss of data and computer system crashes.

[0003] A computer programmed to implement a security policy for a data network is called a firewall. Typically, firewalls are located at strategic points in the network such that all incoming and outgoing data traffic must pass through a firewall. At these points, all network traffic is trapped and examined by the firewall to determine if the traffic should be allowed to pass through the firewall. There are various techniques for implementing a firewall. One such implementation is a computer programmed to implement a data packet filter. A data packet filter examines all received data packets, and either denies or allows passage of the data packet based on the contents of the data packet. In one implementation, this decision is based on rules stored in the data packet filter which define which data packets are allowed to pass and which data packets will be blocked. Each stored rule defines certain parameters of data packets (e.g. source and destination) and also includes the disposition of data packets which match these parameters. For example, a rule may indicate that any data packet from source X will be blocked, while another rule may indicate that any data packet from source Y to destination Z will be allowed to pass. The rules are stored in sequential order in the data packet filter and every data packet received by the firewall is tested against the rules in sequential order. The first rule whose parameters match the received packet is applied to the packet and the packet is treated as indicated by the rule.

[0004] In such a rules based data packet filter, a system administrator must first define a security policy to be implemented by the firewall, and then must program the data packet filter with rules which will implement that security policy. One major problem with these types of firewalls is that sophisticated security policies require a large number of rules in order to implement the policy. Since every data packet which passes through the firewall must be checked against these rules, the performance of the firewall degrades as the number of rules increases.

[0005] One known solution to the performance problem in a data packet filter firewall is to use a memory cache. In such a system, when a data packet arrives, the relevant parameters (e.g. source and destination) of the data packet are stored in a cache. In addition, after the packet filter rules have been applied to the received data packet, the disposition (e.g. allow or deny) is also stored in the cache associated with the relevant parameters of the received data packet. Thereafter, if a data packet is received with parameters which are the same as parameters previously stored in the cache, the firewall can apply the associated disposition without applying all the rules to the data packet. This enhances performance in view of the fact that for certain applications, ongoing communications will occur between two computers, and there is no need to check every data packet exchanged between the computers during the communication session (i.e., connection). Thus, while this technique improves performance for data packets exchanged during connections, the technique does not improve performance for new connections.

[0006] Packet filters may also be implemented on a client computer, for example a personal computer running World Wide Web (WWW) browsing software (e.g., Microsoft Explorer or Netscape Navigator). Many WWW sites contain material which parents may deem unsuitable for children. As such, several companies offer filtering software which implements packet filters for blocking unsuitable material. These packet filters implement a security policy through the use of rules as described above. As the number of rules increases, these client based packet filters suffer from the same performance problems as described above.

[0007] What is needed is a technique for improving the performance of a data packet filter as the number of rules required to implement a security policy increases.

Summary of the Invention

[0008] As described above, a data packet filter stores a plurality of ordered rules which are sequentially applied to received data packets to implement a security policy. In accordance with the invention, the rules are automatically re-

IPR2023-01353 Page 00640

ordered to improve the performance of the packet filter. Rules which match incoming data packets more frequently are moved earlier in the ordering, and rules which match incoming data packets less frequently are moved later in the ordering. Since the first rule that matches a received data packet controls the disposition of the packet, once a rule matches a packet, the remaining rules need not be evaluated. By re-ordering the rules, data packets are matched against rules more quickly, and the performance of the data packet filter is improved.

[0009] In one embodiment of the invention, the packet filter dynamically maintains, during operation of the packet filter, a count of the number of times each rule matches a received data packet. The packets are then re-ordered based on these counts with the rules having higher match counts being moved earlier in the rules order. Such re-ordering is made on the premise that past history of the packet filter provides a reliable indication of future operation. Thus, by dynamically maintaining match counts during operation, the packet filter can adaptively re-order its rules based on past operating parameters. Such adaptive re-ordering is beneficial to protect against so-called denial of service attacks, as described in further detail below.

[0010] In one embodiment of the invention, rules are re-ordered by swapping pairs of rules in the rules order. However, the sequence of the rules plays a part in defining the security policy of the packet filter. As such, the indiscriminate swapping of rules may result in changing the security policy of the packet filter. Such a result is undesirable and must be prevented. As such, rules are evaluated to determine re-orderings which are not allowed because they result in a change in the security policy. In one embodiment, such an evaluation is made by comparing pairs of rules to determine which pairs of rules conflict with each other. Two rules are said to conflict with each other if swapping the rules would result in changing the security policy of the packet filter. The result of this conflict evaluation is stored in the memory of the packet filter and subsequent re-ordering refers to this information in order to prevent the swapping of conflicting rules.

[0011] The re-ordering in accordance with the invention may be performed periodically, at the initiation of a user, or based on the some operating parameters of the packet filter (e.g. when the performance of the packet filter drops below a threshold).

[0012] The principles of the present invention may be applied to various types of computing devices. For example, and without limitation, the invention may be implemented in a dedicated network firewall which filters data packets for many computers and networks, or in a stand-alone personal computer such that the packet filter only filters data packets for the personal computer.

[0013] These and other advantages of the invention will be apparent to those of ordinary skill in the art by reference to the following detailed description and the accompanying drawings.

Brief Description of the Drawings

[0014]

Fig. 1 shows a network of the type in which the present invention may be implemented;
Fig. 2 shows a computer configured to implemented a firewall in accordance with the present invention;
Fig. 3 shows the format of an exemplary data packet;
Fig. 4 shows an example of a rule set;
Fig. 5 shows a conflict table; and
Fig. 6 shows a personal computer in which the principles of the present invention may be implemented.

Detailed Description

[0015] A network of the type in which the present invention may be implemented is shown in Fig. 1. An internal data network, such as a corporate intranet, is shown as network 102. An external data network, such as the Internet, is shown as network 106. Firewall 104 is shown positioned such that all data traffic exchanged between internal network 102 and external network 106 must pass through the firewall 104. These data networks communicate with each other using the well known Internet Protocol (IP) for data packet communication. Assume that firewall 104 is administered by the same entity which administers internal network 102. The purpose of firewall 104 is to implement the security policy of the internal network 102.

[0016] The firewall 104 may be implemented by a properly configured computer, further details of which are shown in Fig. 2. Fig. 2 shows firewall 104 comprising a processor 202 which controls the overall operation of the firewall 104. Processor 202 is connected to input port 204, buffer 206, output port 208, program memory 210, and data memory 220. Program memory 210 and data memory 220 may be any type of well known computer storage device. In operation, processor 202 controls the operation of the firewall 104 by executing computer program instructions stored in program memory 210. Program memory 210 includes the following program modules: re-order module 212, filter module 214, activity module 216, and conflict module 218. These modules contain computer program code which, when executed

IPR2023-01353 Page 00641

by processor 202, implements a packet filter in accordance with the present invention. It is to be understood that the program modules shown in program memory 210 are high level representations of functional computer program modules. In various implementations, these modules could include program source code, object code, or a combination of source and object code. These modules will be discussed in further detail below. Of course, a computer implementing a firewall would have other well known components as well. Such other components are not shown in Fig. 2 for clarity.

[0017] The operation of firewall 104 is as follows. When a data packet is received at firewall 104 from external network 106, the data packet is received at input port 204. The format of an exemplary data packet 302 is shown in Fig. 3. Data packet 302 contains a header portion 304 and data portion 306. The header portion 304 contains the following fields: protocol 308, source address 310, destination address 312, source port 314, and destination port 316. The protocol field 308 indicates the protocol with which the current packet is being transmitted. For example, the protocol field 308 may indicate that the data packet 302 is being transmitted using the well known TCP/IP protocol. The source address 310 indicates the IP address of the source of the data packet 302. IP addressing is well known in data networking. IP addresses are 32 bits, and are most commonly represented in a dotted notation as 4 integers, each between 0 and 255, and each separated by a dot (e.g. 125.180.104.4). The destination address 312 indicates the IP address of the destination of the data packet 302. The source port 314 indicates the port number of the source computer. A port number is an integer between 0 and 65,535, and generally is associated with an application that runs on a particular machine. For example, web servers are typically associated with port number 80. The destination port 316 indicates the port number of the destination computer. The data portion 306 of the packet 302 contains the payload of the packet which will be used by an application executing on the destination computer.

[0018] Upon receipt of the data packet at input port 204, the data packet 302 is stored in buffer 206 while the processor 202 executes the filter module 214 in order to apply the filter rules 222 stored in data memory 220 against the received data packet. The filter module 214 applies the rules to the received data packet in sequential order. The first rule that matches the data packet is the rule which will decide whether the data packet is allowed or denied. Once a rule matches the data packet, there is no need to apply the remaining rules.

[0019] An example of a rule set which would be stored as rules 222 is shown in Fig. 4. As shown in Fig. 4, each rule has a sequence number, which indicates the order in which the rules will be applied against an incoming data packet. Each rule also has fields indicating protocol, source address, destination address, source port, destination port, and an action. Each rule indicates the action to be taken when a data packet is received which matches the information in the rule's fields. A packet is said to match the rule if the header information in the packet falls within the defined ranges of values in the above described fields. For example, rule number 1 indicates that any packet received using the TCP protocol with a source address of 125.30.20.200 is to be denied (i.e. blocked), regardless of the contents of the other fields of the packet. The "*" in a rule field indicates a wildcard which will match any value. Rule number 2 indicates that any packet received using the TCP protocol with a source address in the range 135.180.104.4 to 135.180.104.225, a destination address in the range 135.180.204.4 to 135.180.204.255, a source port in the range 1024-65535, and a destination port of 80 is to be allowed (i.e. passed to its destination). Rule number 3 indicates that any packet received using the TCP protocol from a source address in the range 125.180.100.0 - 125.180.100.255 is to be allowed, regardless of the values in the other fields. Finally, rule number 4 indicates that all data packets are to be denied. Since rules are applied in sequential order, rule number 4 will only be applied if none of the rules 1-3 match the incoming data packet. The inclusion of a rule such as rule number 4 is common practice in firewalls and implements a security policy that all packets are denied unless specifically allowed. As would readily be recognized by one skilled in the art, a wide variety of security policies may be implemented by storing different rule sets as rules 222 in data memory 220. It is also noted that the action indicated in a rule may be something other than allow or deny. For example, it is possible that an action indicates that packets matching the rule be forwarded to some other location for further processing.

[0020] Thus, while the received data packet is stored in buffer 206, processor 202 sequentially applies rules 222 to the data packet until a rule matches the data packet. If the matched rule indicates that the data packet is to be allowed, then the packet is transferred to output port 208 and is allowed to pass to the internal network 102. If the matched rule indicates that the data packet is to be denied, then the packet is deleted from the buffer 206 and is thus not allowed to pass to the internal network 102. It is noted that although the rules are applied in sequential order, they will not necessarily be stored internally in data memory 220 in that sequential order. For various reasons, the internal representation of the rules may be different from the logical representation shown in fig. 4. However, regardless of the internal representation, each rule will be associated with a sequence number or other indication of order of evaluation (e.g. priority), such that the filter module 214 will apply the rules to a received data packet in a given sequence.

[0021] As the number of stored rules 222 increases, the performance of the firewall 104 generally degrades because it takes longer to apply the rules to each incoming data packet. The present invention solves this performance problem by re-ordering the stored rules 222 such that rules which are more likely to match incoming data packets are placed earlier in the sequential order (i.e. assigned a lower sequence number). This improves the performance of the firewall because once a rule is matched, the remaining rules are not applied, thus avoiding the processing time required to apply those remaining rules. If rules which are more likely to match data packets can be moved earlier in the rules

IPR2023-01353 Page 00642

sequence, performance of the firewall will improve.

[0022] However, prior to re-ordering the rules, it must be determined which rules may be re-ordered without having an affect on the overall security policy of the firewall 104. Due to the nature of the rules and the sequential application of the rules to received data packets, certain re-orderings may result in a change in the overall security policy. Such re-orderings must be prevented. For example, referring to the example rule set of fig. 4, rule number 4 could not be swapped with rule 1, because such a swap will result in rule number 4 being applied first. Since rule number 4 indicates that all data packets are to be denied, placing it first in the rules sequence would result in all data packets being denied. None of the other rules would be reached to allow the data packets to pass. Therefore, it is necessary to determine which rules may be re-ordered while still maintaining the security policy of the firewall.

[0023] In order to determine which rules may be re-ordered, conflict module 218 is executed by processor 202. The conflict module analyzes the rule set 222 to determine whether pairs of rules are in conflict such that the swapping of the pair would result in a change in the security policy. Generally, two rules are said to conflict with each other if at least one data packet could match either one of the rules and the actions of the rules are not the same. In the embodiment described herein, the conflict algorithm to determine whether a Rule-A conflicts with a Rule-B is as follows:

```
If ACTION of Rule-A is not the same as ACTION of Rule-B then

      if the PROTOCOL of Rule-A does not intersect with the
      PROTOCOL OF Rule-B then
```


IPR2023-01353 Page 00643

```
                    if the SOURCE ADDRESS of Rule-A does not intersect
                    with the SOURCE ADDRESS of Rule-B then

                          if the DESTINATION ADDRESS of Rule-A does not
                          intersect with the DESTINATION ADDRESS of Rule-
                          B then

                                if the SOURCE PORT of Rule-A does not
                                intersect with the SOURCE PORT of Rule-B
                                then

                                      if the DESTINATION PORT of Rule-A
                                      does not intersect with the
                                      DESTINATION PORT of Rule-B then

                                            no conflict between Rule-A
                                            and Rule-B
                                      else
                                else
                          else
                    else
              else
                    conflict between Rule-A and Rule-B.
        else

              no conflict.
```

[0024] In accordance with the algorithm, a particular of a rule is said to *intersect* with a particular field of another rule if there is any intersection in the range of values of the two fields. For example, with respect to fields which contains IP addresses, the fields will intersect if there is any overlap between the addresses. For example, an IP address field containing the address range 123.30.20.100 intersects with an IP address field containing the address range 123.30.20.70 - 123.30.20.150 in the address range 123.30.20.70 - 123.30.20.100. For example, the IP address 123.30.20.85 would fall within both ranges. Thus, in accordance with the algorithm, if the actions of two rules are different, and if the there is an intersection in any of the fields of the two rules, then the rules conflict. Otherwise, the rules do not conflict. Of course, one skilled in the art could modify the above algorithm in order to detect the conflict between rules with a different set of fields. Such an algorithm would analyze each field of each rule to determine if there were an intersection. If there is any intersection, and if the action of the two rules is different, then the rules conflict.

[0025] It is noted that the conflict module 218 is executed once when the rule set 222 changes (i.e. a rule is added or deleted). The results of the execution of the conflict module 218 are stored in data memory 220 as conflict data 226 which indicates, for every possible pair of rules, whether that pair conflicts. In one embodiment, a two-dimensional conflict table such as conflict table 500 shown in Fig. 5 could be stored as conflict data 226 as a result of the execution of conflict module 218. For each pair of rules, the intersection point in the conflict table 500 indicates whether the rules are in conflict (C) or not in conflict (NC). For example, the intersection of row 3 and column 2 in conflict table 500

IPR2023-01353 Page 00644

contains NC indicating that rule 3 and rule 2 do not conflict. If a rule is added, only the interaction of the new rule with the other rules needs to be checked with the conflict module 218. The interaction of the other rules will not change. It is noted that the conflict table 500 is symmetrical in that the intersection of column x and row y will contain the same value as the intersection of column y and row x. As such, only half of the table needs to be stored in conflict data 226 in data memory 220. However, the entire table is shown in Fig. 5 for purposes of this description.

[0026] Returning now to the re-ordering of rules, as was stated above, in accordance with the invention, the performance of a packet data filter is improved by re-ordering the stored rules 222 such that rules which are more likely to match incoming data packets are placed earlier in the sequential order. In accordance with one aspect of the invention, the decision as to which rules are more likely to match incoming data packets is made with reference to past history of the firewall 104. In one embodiment, this past history is a record of the number of times each rule has matched an incoming data packet. In accordance with this embodiment, during operation of the firewall 104, an activity module 216 dynamically maintains a running count of the number of times each rule matched an incoming packet. This count is stored as a table in the history section 224 of data memory 220. The table associates a count with each rule. Thus, rules associated with higher counts matched incoming packets more often than rules associated with lower counts. It is noted that the counts can periodically be reinitialized to zero.

[0027] The re-ordering of the rule set 222 may be performed periodically (e.g. every 30 minutes), may be initiated by a system administrator, or may be initiated by the firewall 104 based on certain operating parameters of the firewall 104. For example, the firewall 104 may initiate a re-ordering when the performance of the firewall 104 drops below a certain predetermined threshold. The re-ordering is performed by the re-order module 212 by swapping rules having a lower count in history section 224 with rules having a higher count in history section 224. This results in rules which matched data packets more often being placed earlier in the sequential ordering.

[0028] One algorithm for swapping rules is as follows.

```
For i = 1 to N

    J = N

    While J > 0

        If match count for Rule J > match count for Rule J-1
        then

            if Rule J does not conflict with Rule J-1 then

                Swap Rule J with Rule J-1.

        J = J - 1.
```

This algorithm attempts to move the rules with a higher match count earlier in the sequence of rules. One constraint of the re-ordering of rules is that rules are only swapped if doing so will not change the overall security policy of the rule set (i.e. the rules do not conflict). While this algorithm uses a well known bubble sort type of algorithm, one skilled in the art could readily implement various types of re-ordering algorithms.

[0029] One benefit of the present invention is that the filter rules are re-ordered based on the activity of the firewall. As such, the firewall is dynamically adaptable to changing operating conditions. As an example, one type of attack on a computer system is known as a denial of service attack, which is aimed at reducing the availability of network elements. One technique for mounting such an attack is to continuously transmit packets to a network, which packets are known to be blocked by the network's firewall. If the rule which rejects such packets is late in the sequential ordering of rules, then every time such a packet is received by the firewall, the firewall must apply many rules before rejecting the packet. For example, if the rule which will eventually reject the data packet is rule number 100, then each time

IPR2023-01353 Page 00645

such a packet is received, the firewall must apply 100 rules prior to rejecting the packet. By sending many such packets, an attacker may seriously impair the performance of the firewall.

[0030] The present invention can improve the performance during a denial of service attack as follows. If an attacker is continuously sending packets which are rejected by rule number 100, then the match count for rule number 100 will increase quickly. Further, since one of the parameters for initiating a re-ordering of rules is the performance of the firewall, then a denial of service attack will substantially degrade the performance such that a re-ordering will be initiated. At such time, since the match count of rule 100 will be high, the re-ordering module will attempt to move rule 100 earlier in the sequential ordering. For example, if the rule were moved to rule 5, then further receipt of the attacking packets will result in the packets being blocked after only 5 rules, rather than 100, an improvement of 95%.

[0031] It is to be understood that although the invention has been described in the context of a firewall implemented in a network as shown in Fig. 1, the principles of the present invention may also be implemented in a personal computer configured to filter data packets received from a data network. In one well known configuration, a personal computer is connected to the Internet using a modem and a dial-up connection to the Internet via the telephone network. Since it is often desirable to limit the Internet sites which a user of a personal computer may access, there are various filter programs available which filter incoming data packets based on filtering rules. The principles of the present invention may be implemented in such a personal computer. Such an embodiment of the invention is shown in Fig. 6 as personal computer 600. Personal computer 600 includes a processor 610 connected to a modem 612 which is used to establish a connection to a data network, such as the Internet. In one embodiment, the computer 600 executes a browser program 604 which allows the computer 600 to communicate with World Wide Web (WWW) sites and to receive information from such sites. In accordance with the invention, the computer 600 also includes a packet filter program module 606, which contains a re-order module, filter module, activity module, and intersection module, which operate to implement a packet filter as described above. Also, the computer 600 contains data memory 608 including rules, history, and conflict data, also as described above. A user of the computer 600 interacts with the computer 600 via keyboard 614, graphical input device 618 (e.g. mouse), and display 616 in a conventional manner. In operation, a user of computer 600 may request information from the Internet, and the packet filter module 606 will operate as described above in order to filter data packets in accordance with stored filter rules. The filter rules may be re-ordered to improve the performance of the packet filter, as long as such re-ordering does not change the overall security policy of computer 600.

[0032] It is noted that network routers may optionally be configured to filter data packets in a manner similar to that described above in connection with firewalls. Thus, the principles of the present invention may be implemented in a network router which is configured for data packet filtering.

[0033] The foregoing Detailed Description is to be understood as being in every respect illustrative and exemplary, but not restrictive, and the scope of the invention disclosed herein is not to be determined from the Detailed Description, but rather from the claims as interpreted according to the full breadth permitted by the patent laws. It is to be understood that the embodiments shown and described herein are only illustrative of the principles of the present invention and that various modifications may be implemented by those skilled in the art without departing from the scope and spirit of the invention.

Claims

1. A method for operation of a data packet filter which implements a security policy by filtering data packets by sequentially evaluating a plurality of ordered rules, said method comprising the step of:
   automatically re-ordering said ordered rules.

2. The method of claim 1 wherein said step of re-ordering is based on prior activity of said data packet filter.

3. The method of claim 2 further comprising the step of:

   dynamically maintaining, during operation of said data packet filter, match data indicating the number of times each of said rules is matched by a data packet; and
   wherein said step of re-ordering is based on said match data.

4. The method of claim 3 wherein said rules are re-ordered such that rules which are matched more often are evaluated earlier.

5. The method of claim 1 wherein said step of automatically re-ordering is performed periodically.

6. The method of claim 1 wherein said step of automatically re-ordering is performed when the performance of said

IPR2023-01353 Page 00646

data packet filter falls below a threshold.

7. The method of claim 1 wherein said step of re-ordering is performed by swapping pairs of rules, said method further comprising the step of:
determining pairs of rules which can be swapped without changing said security policy.

8. The method of claim 7 wherein said step of re-ordering is performed such that said security policy is not changed.

9. The method of claim 7 wherein said step of determining further comprises the step of determining the intersection of fields of pairs of said rules.

10. A data packet filter for implementing a security policy comprising:

a memory for storing a plurality of ordered rules;
a filter module for sequentially applying said rules to received data packets; and
a re-ordering module for re-ordering said ordered rules.

11. The data packet filter of claim 10 wherein said memory further stores information representing past activity of said data packet filter and wherein said re-ordering is based on said past activity.

12. The data packet filter of claim 11 wherein said information representing past activity comprises match data indicating the number of times each of said rules is matched by a data packet.

13. The data packet filter of claim 12 further comprising:
an activity module for dynamically maintaining said match data.

14. The data packet filter of claim 10 further comprising:
a conflict module for determining pairs of said rules which can be swapped without altering said security policy.

15. The data packet filter of claim 14 wherein said re-ordering module only swaps pairs of rules which can be swapped without altering said security

16. The data packet filter of claim 14 wherein said conflict module determines pairs of said rules which can be swapped without altering said security policy by determining the intersection of fields of pairs of rules.

17. A computer readable medium storing computer program instructions which are executable on a computer system implementing a data packet filter, said computer program instructions comprising instructions defining the steps of:

filtering received data packets by sequentially applying a plurality of ordered rules, said rules defining a security policy; and
automatically re-ordering said ordered rules.

18. The computer readable medium of claim 17 wherein said computer program instructions further comprise instructions defining the steps of:
re-ordering said rules based on prior activity of said computer system.

19. The computer readable medium of claim 18 wherein said computer program instructions further comprise instructions defining the steps of:

dynamically maintaining, during operation of said computer system, match data indicating the number of times each of said rules is matched by a data packet; and
re-ordering said rules based on said match data.

20. The computer readable medium of claim 19 wherein said computer program instructions further comprise instructions defining the steps of:
re-ordering said rules such that rules which are matched more often are applied earlier.

21. The computer readable medium of claim 17 wherein said computer program instructions further comprise instruc-

IPR2023-01353 Page 00647

tions defining the step of:
periodically automatically re-ordering said rules.

22. The computer readable medium of claim 17 wherein said computer program instructions further comprise instructions defining the step of:
automatically re-ordering said rules when the performance of said computer system falls below a threshold.

23. The computer readable medium of claim 17 wherein said computer program instructions further comprise instructions defining the step of:
determining pairs of rules which can be swapped without changing said security policy.

24. The computer readable medium of claim 23 wherein said computer program instructions further comprise instructions defining the step of:
re-ordering said rules by swapping only those rules which may be swapped without changing said security policy.

25. The computer readable medium of claim 23 wherein said computer program instructions further comprise instructions defining the step of:
determining pairs of said rules which can be swapped without changing said security policy by determining the intersection of fields of pairs of rules.

26. A firewall for filtering data traffic between a first network and a second network comprising:

an input port for receiving data packets;
a memory storing a plurality of ordered rules defining a security policy of said firewall;
a filter module for sequentially applying said ordered rules to received data packets; and
a re-order module for automatically re-ordering said plurality of rules.

27. The firewall of claim 26 further comprising:
a conflict module for determining whether pairs of rules can be swapped without altering said security policy.

28. The firewall of claim 27 wherein said conflict module determines whether pairs of rules can be swapped without altering said security policy by determining the intersection of fields of pairs of rules.

29. The firewall of claim 27 wherein said re-order module only swaps pairs of rules which can be swapped without altering said security policy.

30. The firewall of claim 26 further comprising an activity module for dynamically maintaining activity information about the operation of said firewall.

31. The firewall of claim 30 wherein said activity information comprises match data indicating the number of times each of said rules is matched by a data packet.

32. The firewall of claim 30 wherein said re-order module re-orders said plurality of rules based on said activity information.

IPR2023-01353 Page 00648

FIG. 1

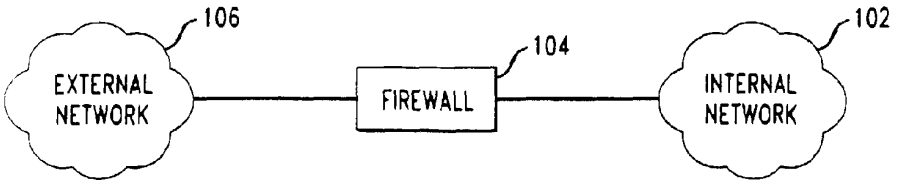
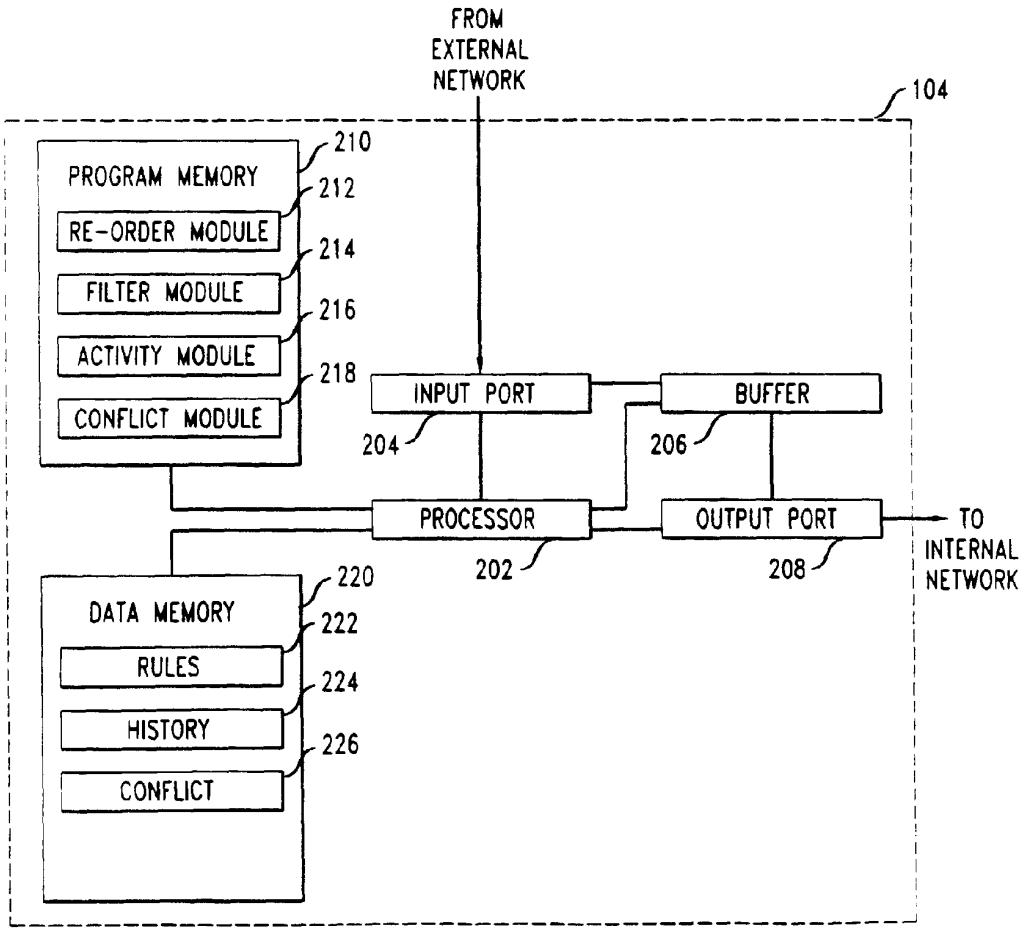


FIG. 2



IPR2023-01353 Page 00649

FIG. 3

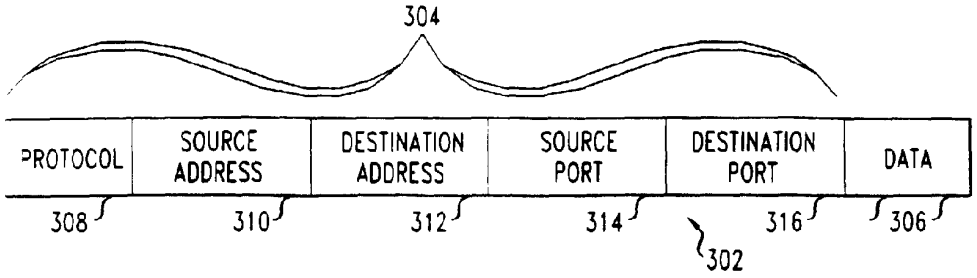


FIG. 4

SEQ. NO.	PROTOCOL	SOURCE ADDRESS	DESTINATION ADDRESS	SOURCE PORT	DESTINATION PORT	ACTION
1	TCP	125.30.20.200	*	*	*	DENY
2	TCP	135.180.104.4-135.180.104.255	135.180.204.4-135.180.204.255	1024-65535	80	ALLOW
3	TCP	125.180.100.*	*	*	*	ALLOW
4	*	*	*	*	*	DENY

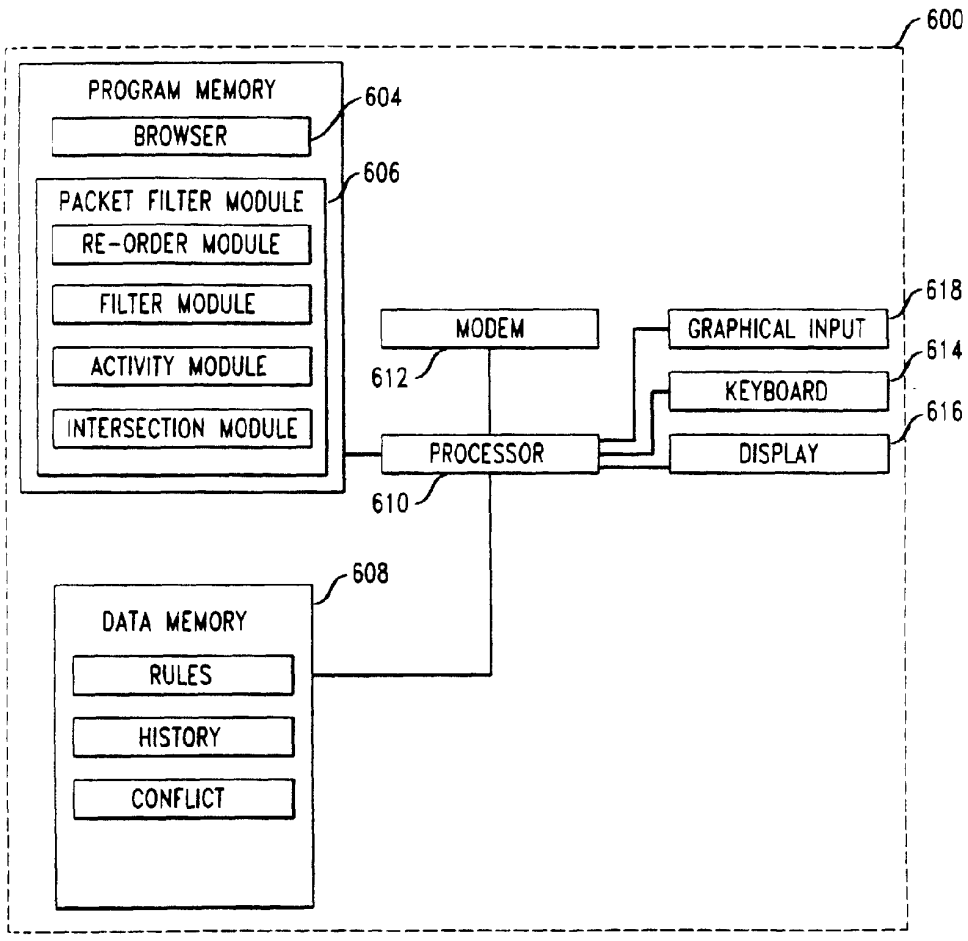
FIG. 5

	1	2	3	4	5	
1	NC	C	C	NC	C	
2	C	NC	NC	C	NC	
3	C	NC	NC	NC	C	
4	NC	C	NC	NC	C	
5	C	NC	C	C	NC	

500

IPR2023-01353 Page 00650

FIG. 6



IPR2023-01353 Page 00651

Espacenet

Bibliographic data: WO2006105093 (A2) — 2006-10-05

METHODS, SYSTEMS, AND COMPUTER PROGRAM PRODUCTS FOR NETWORK FIREWALL POLICY OPTIMIZATION

Inventor(s): FULP ERRIN W [US]; TARSA STEPHEN J [US] ± (FULP, ERRIN, W, ; TARSA, STEPHEN, J)

Applicant(s): UNIV WAKE FOREST [US]; FULP ERRIN W [US]; TARSA STEPHEN J [US] ± (WAKE FOREST UNIVERSITY, ; FULP, ERRIN, W, ; TARSA, STEPHEN, J)

Classification: **- international:** ***G06F15/16***
- cooperative: **H04L63/0263**; **H04L63/20**

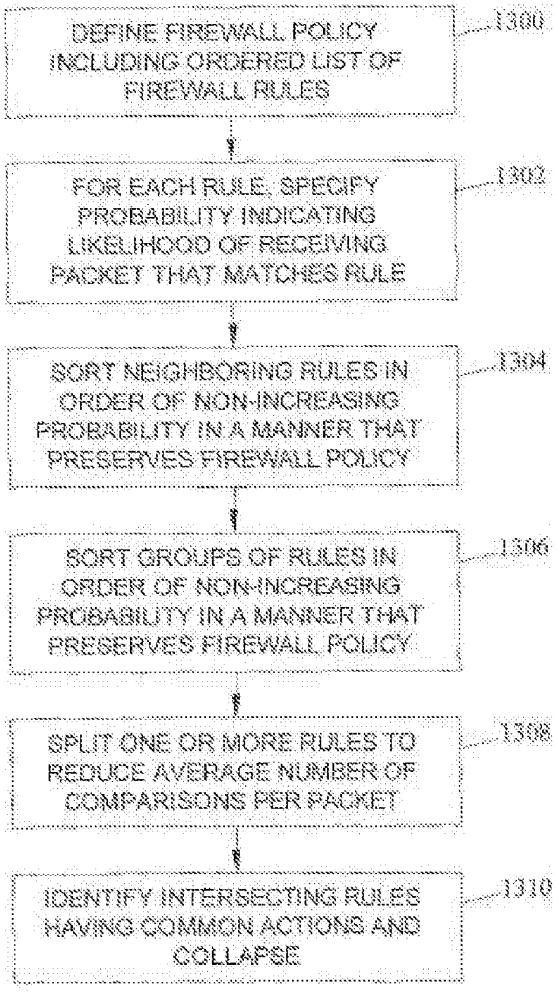
Application number: WO2006US11291 20060328

Priority number (s): US20050665664P 20050328

Also published as: WO2006105093 (A3) US2006248580 (A1) US8042167 (B2) EP1864226 (A2) EP1864226 (A4) more

Abstract of WO2006105093 (A2)

Methods, systems, and computer program products for firewall policy optimization are disclosed. According to one method, a firewall policy including an ordered list of firewall rules is defined. For each rule, a probability indicating a likelihood of receiving a packet matching the rule is determined. The rules are sorted in order of non-increasing probability in a manner that preserves the firewall policy.


DEFINE FIREWALL POLICY INCLUDING ORDERED LIST OF FIREWALL RULES
1300
FOR EACH RULE, SPECIFY PROBABILITY INDICATING LIKELIHOOD OF RECEIVING PACKET THAT MATCHES RULE
1302
SORT NEIGHBORING RULES IN ORDER OF NON-INCREASING PROBABILITY IN A MANNER THAT PRESERVES FIREWALL POLICY
1304
SORT GROUPS OF RULES IN ORDER OF NON-INCREASING PROBABILITY IN A MANNER THAT PRESERVES FIREWALL POLICY
1306
SPLIT ONE OR MORE RULES TO REDUCE AVERAGE NUMBER OF COMPARISONS PER PACKET
1308
IDENTIFY INTERSECTING RULES HAVING COMMON ACTIONS AND COLLAPSE
1310

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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
5 October 2006 (05.10.2006)

PCT

(10) International Publication Number
WO 2006/105093 A2

(51) International Patent Classification:
G06F 15/16 (2006.01)

(21) International Application Number:
PCT/US2006/011291

(22) International Filing Date: 28 March 2006 (28.03.2006)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
60/665,664 28 March 2005 (28.03.2005) US

(71) Applicant *(for all designated States except US)*: **WAKE FOREST UNIVERSITY** [US/US]; OFFICE OF TECHNOLOGY ASSET MANAGEMENT, WAKE FOREST UNIVERSITY HEALTH SCIECNES, 200 East First Street, Suite 101, Winston Salem, North Carolina 27101 (US).

(72) Inventors; and

(75) Inventors/Applicants *(for US only)*: **FULP, Errin, W.** [US/US]; 308 N. Hawthorne Road, Winston-Salem, North Carolina 27104 (US). **TARSA, Stephen, J.** [US/US]; 135 King Phillips Path, Duxbury, Massachusetts 02332-3503 (US).

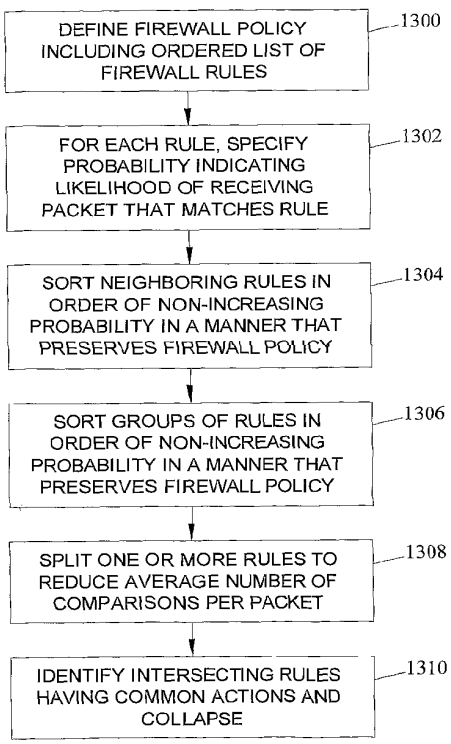
(74) Agent: **HUNT, Gregory, A.**; JENKINS, WILSON, TAYLOR & HUNT, P.A., Suite 1200 University Tower, 3100 Tower Boulevard, Durham, North Carolina 27707 (US).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BW, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, LY, MA, MD, MG, MK, MN, MW, MX, MZ, NA, NG, NI, NO, NZ, OM, PG, PH, PL, PT, RO, RU, SC, SD, SE, SG, SK, SL, SM, SY, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, YU, ZA, ZM, ZW.

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM),

[Continued on next page]

(54) Title: METHODS, SYSTEMS, AND COMPUTER PROGRAM PRODUCTS FOR NETWORK FIREWALL POLICY OPTIMIZATION



(57) Abstract: Methods, systems, and computer program products for firewall policy optimization are disclosed. According to one method, a firewall policy including an ordered list of firewall rules is defined. For each rule, a probability indicating a likelihood of receiving a packet matching the rule is determined. The rules are sorted in order of non-increasing probability in a manner that preserves the firewall policy.

WO 2006/105093 A2

European (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HU, IE, IS, IT, LT, LU, LV, MC, NL, PL, PT, RO, SE, SI, SK, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Published:

— *without international search report and to be republished upon receipt of that report*

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.

DESCRIPTION

METHODS, SYSTEMS, AND COMPUTER PROGRAM PRODUCTS FOR NETWORK FIREWALL POLICY OPTIMIZATION

RELATED APPLICATIONS

This application claims the benefit of U.S. Provisional Patent Application Serial No. 60/665,664, filed March 28, 2005; the disclosure of which is incorporated herein by reference in its entirety.

GOVERNMENT INTEREST

This invention was made with Government support under Grant No. DE-FG02-03ER25581 awarded by U.S. Department of Energy, Mathematical and Computational Information Sciences Division. The Government has certain rights in the invention.

TECHNICAL FIELD

The subject matter described herein relates to optimizing polices for network firewalls. More particularly, the subject matter described herein relates to methods, systems, and computer program products for optimizing performance of network firewalls.

BACKGROUND ART

Network firewalls remain the forefront defense for most computer systems. Guided by a security policy, these devices provide access control, auditing, and traffic control [3, 30, 31]. A security policy is a list of ordered rules, as seen in Table 1, that defines the action to perform on matching packets. For example, an accept action passes the packet into or from the secure network, while deny causes the packet to be discarded. In many implementations, the rule set is stored internally as a linked list. A packet is sequentially compared to the rules, starting with the first, until a match is found; otherwise, a default action is performed [30, 31]. This is referred to as a first-match policy and is used in many firewall systems including the Linux firewall implementation iptables [25].

IPR2023-01353 Page 00656

No.	Proto.	Source IP	Source Port	Destination IP	Destination Port	Action	Prob.
1	UDP	1.1.*	*	*	80	deny	0.01
2	TCP	2.*	*	1.*	90	accept	0.02
3	UDP	*	*	1.*	*	accept	0.10
4	TCP	2.*	*	1.*	20	accept	0.17
5	UDP	1.*	*	*	*	accept	0.20
6	*	*	*	*	*	deny	0.50

Table 1: Example Security Policy Consisting of Multiple Ordered Rules

Traditional firewall implementations consist of a single, dedicated machine, similar to a router, that sequentially applies rules to each arriving packet. However, packet filtering represents a significantly higher processing load than routing decisions [24, 29, 31]. For example, a firewall that interconnects two 100 Mbps networks would have to process over 300,000 packets per second [30]. Successfully handling these traffic loads becomes more difficult as rule sets become more complex [4, 22, 31]. Furthermore, firewalls must be capable of processing even more packets as interface speeds increase. In a high-speed environment (e.g. Gigabit Ethernet), a single firewall can easily become a bottleneck and is susceptible to DoS attacks [4, 9, 13, 14]. An attacker could simply inundate the firewall with traffic, delaying or preventing legitimate packets from being processed.

One approach to increase firewall performance focuses on improving hardware design. Current research is investigating different distributed firewall designs to reduce processing delay [4, 9, 22], and possibly provide service differentiation [11]. Another approach focuses on improving performance via better firewall software [6, 7, 12, 16, 17, 24]. Similar to approaches that address the longest matching prefix problem for packet classification [8, 10, 25, 28], solutions typically represent the firewall rule set in different fashions (e.g. tree structures) to improve performance. While both approaches, together or separate, show great promise, each requires radical changes to the firewall system, and therefore are not amenable to current or legacy systems.

Accordingly, there exists a need for improved methods, systems, and computer program products for network firewall policy optimization.

SUMMARY

The subject matter described herein includes methods, systems, and computer program products for network firewall policy optimization. According to one method, a firewall policy including an ordered list of firewall rules is defined. For each rule, a probability indicating a likelihood of receiving a packet matching the rule is defined. Neighboring rules are sorted in order of non-increasing probability in a manner that preserves the firewall policy.

As used herein, the term "firewall" refers to a logical entity that is adapted to filter packets at the ingress and/or egress portions of a network based on a policy. The term "firewall" is intended to include systems that block packets from entering or leaving a network and systems that perform intrusion detection and intrusion protection functions for packets entering or leaving a network. Thus, the methods and systems described herein for firewall policy optimization can be used to optimize policies for intrusion detection and intrusion protection firewalls.

The subject matter described herein includes a method to improve firewall performance and lower packet delay that can be applied to both legacy and current systems. In one implementation, a firewall rule set may be re-ordered to minimize the average number of rule comparisons to determine the action, while maintaining the integrity of the original policy. Integrity is preserved if the reordered and original rules always arrive at the same result. To maintain integrity, a firewall rule set may be modeled as a Directed Acyclical Graph (DAG), where vertices are firewall rules and edges indicate precedence relationships. Given this representation, any linear arrangement of the DAG is proven to maintain the original policy integrity. Unfortunately, determining the optimal rule order from all the possible linear arrangements is proven to be *NP*-complete, since it is equivalent to sequencing jobs with precedence constraints for a single machine [15]. Although determining the optimal order is *NP*-complete, a heuristic will be described below to order firewall rules that reduces the average number of comparisons while maintaining integrity. Simulation

IPR2023-01353 Page 00658

results show the proposed reordering method yields rule orders that are comparable to optimal; thus, provides a simple means to significantly improve firewall performance and lower packet delay.

The subject matter described herein for network firewall policy optimization may be implemented using any combination of hardware, software, or firmware. For example, the subject matter described herein may be implemented using a computer program product comprising computer executable instructions embodied in a computer readable medium. Exemplary computer readable media suitable for implementing the subject matter described herein include disk memory devices, chip memory devices, programmable logic devices, application specific integrated circuits, and downloadable electrical signals. In addition, a computer program product that implements the subject matter described herein may be implemented using a single device or computing platform or may be distributed across multiple devices or computing platforms.

BRIEF DESCRIPTION OF THE DRAWINGS

Preferred embodiments of the subject matter described herein will now be explained with reference to the accompanying drawings of which:

Figure 1A is a block diagram of a rule list Directed Acyclical Graph (DAG) for the firewall rules in Table 1;

Figure 1B is a linear arrangement of firewall rules corresponding to the original rule order;

Figure 2A is a policy DAG representing a reordered version of the firewall rule set in Figure 1 after sorting;

Figure 2B is a policy DAG for the rule set illustrated in Table 1 illustrating an optimal rule order;

Figure 3A is a graph illustrating average numbers of packet comparisons versus intersection percentage for different orderings of a firewall policy;

Figure 3B is a graph illustrating the percent difference in average numbers of packet comparisons between sorted and optimal sortings of a firewall policy;

IPR2023-01353 Page 00659

Figure 4A is a graph of percent difference comparisons between original and sorted versions of a firewall policy;

Figure 4B is a graph illustrating the performance impact of considering rule intersection and actions in performing firewall rule sorting;

Figure 5A is a table illustrating a firewall policy before sorting;

Figure 5B is a table illustrating a sorted version of the firewall policy of Figure 5A;

Figure 6A is a policy trie representation of the firewall policy of Figure 5B;

Figure 6B is a table representing the firewall policy of Figure 5A after TCP and UDP sub-trie rotation;

Figure 7A is a policy trie illustrating the splitting of rule r_5 into rules r_{5a} and r_{5b};

Figure 7B is a table illustrating the firewall policy of Figure 5A after splitting rule r_5 into rules r_{5a} and r_{5b};

Figure 8A is a graph illustrating firewall performance with and without rule splitting;

Figure 8B is a graph illustrating locations and probabilities where splitting is better;

Figure 9A is a policy DAG of a firewall rule set;

Figure 9B is a policy DAG illustrating compression of rules r_1 and r_5;

Figure 10A is a policy DAG of a firewall rule list;

Figure 10B is an original policy trie before sorting;

Figure 10C is a policy trie illustrating the firewall rules after rotating sub-tries T_1 and T_2;

Figure 11A is a policy DAG illustrating a firewall policy;

Figure 11B is a policy trie illustrating firewall rules of the policy before sorting;

Figure 11C is a policy trie illustrating the firewall policy after exchanging nodes r_1 and r_2 and nodes r_7 and r_8;

Figure 11D is a policy trie illustrating the firewall policy after exchanging sub-tries T_3 and T_4;

Figure 11E is a policy trie illustrating the firewall policy after exchanging sub-tries T_1 and T_2;

IPR2023-01353 Page 00660

Figure 12A is a policy DAG illustrating an original order for a firewall policy;

Figure 12B is a policy DAG illustrating a sorted version of the firewall policy of Figure 12A after exchanging neighbors;

Figure 12C is a policy DAG illustrating a sorted version of the firewall policy after exchanging lists L_1 including rules r_1 and r_3 and lists L_2 including rules r_2 and r_4;

Figure 13 is a flow chart illustrating exemplary steps for network firewall policy optimization according to an embodiment of the subject matter described herein;

Figure 14 is a block diagram illustrating exemplary components of a system for firewall policy optimization according to an embodiment of the subject matter described herein; and

Figure 15 is a block diagram illustrating a plurality of firewall nodes for implementing a firewall policy according to an embodiment of the subject matter described herein.

DETAILED DESCRIPTION OF THE INVENTION

Modeling Firewall Security Policies

As described above, a firewall rule set, also known as a firewall policy, is traditionally an ordered list of firewall rules. Firewall policy models have been the subject of recent research [1, 2, 16]; however, the primary purpose is anomaly detection and policy verification. In contrast, the policy model described in this herein is designed for firewall performance optimization and integrity. Firewall performance refers to reducing the average number of comparisons required to determine an action, while integrity refers to maintaining the original policy intent. Although improving the worst-case performance is important, it is not possible without changing the list-based representation [16, 24].

Firewall Rule and Policy Models

In the examples described herein, a rule r is modeled as an ordered tuple of sets, $r = (r[1], r[2], ..., r[k])$. Order is necessary among the tuples since

comparing rules and packets requires the comparison of corresponding tuples. Each tuple $r[l]$ is a set that can be fully specified, specify a range, or contain wildcards '*' in standard prefix format. For the Internet, security rules are commonly represented as a 5-tuple consisting of: protocol type, IP source address, source port number, IP destination address, and destination port number [30, 31]. Given this model, the ordered tuples can be supersets and subsets of each other, which forms the basis of precedence relationships. In addition to the prefixes, each filter rule has an action, which is to accept or deny. However, the action will not be considered when comparing packets and rules. Similar to a rule, a packet (IP datagram) d can be viewed as an ordered k-tuple $d = (d[1], d[2], ..., d[k])$; however, ranges and wildcards are not possible for any packet tuple.

Using the previous rule definition, a standard security policy can be modeled as an ordered set (list) of n rules, denoted as $R = \{r_1, r_2, ..., r_n\}$. A packet d is sequentially compared against each rule r_i starting with the first, until a match is found ($d \Rightarrow r_i$) then the associated action is performed. A match is found between a packet and rule when every tuple of the packet is a subset of the corresponding tuple in the rule.

Definition Packet d matches r_i if

$$d \Rightarrow r_i \quad \text{iff} \quad d[l] \subseteq r_i[l], \quad l = 1, ..., k$$

The rule list R is comprehensive if for every possible legal packet d a match is found using R. Furthermore, two rule lists R and R' are equivalent if for every possible legal packet d the same action is performed by the two rule lists. If R and R' are different (e.g. a reorder) and the lists are equivalent, then the policy integrity is maintained.

As previously mentioned, a rule list has an implied precedence relationship where certain rules must appear before others if the integrity of the policy is to be maintained. For example consider the rule list in Table 1. Rule r_1 must appear before rules r_3 and r_5, likewise rule r_6 must be the last rule in the policy. If for example, rule r_3 was moved to the beginning of the policy, then it will shadow [2] the original rule r_1. However, there is no precedence relationship between rules r_2 and r_4 given in Table 1. Therefore, the relative ordering of these two rules will not impact the policy integrity and can be

IPR2023-01353 Page 00662

changed to improve performance. The present example assumes that the original policy is free from any anomalies. Likewise, when a policy is reordered to improve performance it should not introduce any anomalies, which will occur if precedence relationships are not maintained. As a result, a model is needed to effectively represent precedence relationships.

Modeling Rule List Precedence Relationships

The precedence relationship between rules in a policy will be modeled as a Directed Acyclical Graph (DAG) [23, 18]. Such graphs have been successfully used to represent the relative order of individual tasks that must take place to complete a job (referred to as a task graph model). Since certain rules must appear before others to maintain policy integrity, this structure is well suited for modeling the precedence of firewall rules. Let $G = (R, E)$ be a rule list DAG for a rule list R, where vertices are rules and edges E are the precedence relationships (constraint). A precedence relationship, or edge, exists between rules r_i and r_j, if $i < j$, the actions for each rule are different, and the rules intersect.

Definition The intersection of rule r_i and r_j, denoted as $r_i \cap r_j$ is

$$r_i \cap r_j = (r_i[l] \cap r_j[l]), \qquad l = 1, \ldots, k$$

Therefore, the intersection of two rules results in an ordered set of tuples that collectively describes the packets that match both rules. The rules r_i and r_j intersect if every tuple of the resulting operation is non-empty. In contrast, the rules r_i and r_j do not intersect, denoted as $r_i \not\cap r_j$, if at least one tuple is the empty set. Note the intersection operation is symmetric; therefore, if r_i intersects r_j, then r_j will intersect r_i. The same is true for rules that do not intersect.

For example consider the rules given in Table 1, the intersection of r_1 and r_3 yields (UDP, 1.1.*, *, 1.*, 80). Again, the rule actions are not considered in the intersection or match operation. Since these two rules intersect, a packet can match both rules for example d = (UDP, 1.1.1.1, 80, 1.1.1.1, 80). Furthermore, the actions of the two rules are different. Therefore, the relative order must be maintained between these two rules and an edge drawn from r_1 to r_3 must be present in the rule list DAG, as seen in Figures 1A and 1B. More

IPR2023-01353 Page 00663

particularly, Figure 1A illustrates a rule list DAG for the rules in Table 1. Figure 1B illustrates a linear arrangement of the rules in Figure 1A. In Figures 1A and 1B, the vertices represent rules, the circles represents an accept rules, and the squares represents deny rules. Edges that connect the vertices represent precedence requirements. As can be seen from Figure 1B, because of the edge between rules r_1 and r_3, precedence between rules r_1 and r_3 must be maintained. In contrast consider the intersection of rules r_3 and r_5. These two rules intersect, indicating packets belonging to the set (UDP, 1.*, *, 1.*, *) would match both rules. However, it does not matter which of the two rules a packet matches first, since the action is the same for both rules. Therefore, an edge does not exist between rules r_3 and r_5 in the diagram. Similarly, rules r_2 and r_4 do not intersect due to the fifth tuple (destination port). A packet cannot match both rules indicating the relative order can change; therefore, an edge will not exist between them.

The match operation can be used to identify precedence relationships, but it cannot do so in every case. Consider a partial-match example [1], where r_a = (UDP, *, 80, 10.*, 90, accept) and r_b = (UDP, 10.*, 80, *, 90, deny). The intersection of r_a and r_b is (UDP, 10.*, 80, 10.*, 90); therefore a packet, such as d = (UDP, 10.10.10.10, 80, 10.10.10.10, 90), can match both rules. If r_a appears before r_b then the packet d is accepted, but if r_b occurs before r_a then d is rejected. As a result, the order of r_a and r_b in the original policy must be maintained. However, the match operation is unable to identify the precedence in this example. A partial match exists in between rules r_3 and r_5 in Table 1, but as previously discussed an edge does not exist between the rules since the actions are the same.

Using the rule list DAG representation a linear arrangement is sought that improves the firewall performance. As depicted in Figure 1B, a linear arrangement (permutation or topological sort) is a list of DAG vertices where all the successors of a vertex appear in sequence after that vertex [23]. Therefore it follows that a linear arrangement of a rule list DAG represents a rule order, if the vertices are read from left to right. Furthermore, it is proven in the following theorem that any linear arrangement of a rule list DAG maintains integrity.

IPR2023-01353 Page 00664

Theorem Any linear arrangement of a rule list DAG maintains integrity.

Proof Assume a rule list DAG *G* is constructed from the security policy *R* that is free of anomalies. Consider any two rules r_i and r_j in the policy, where $i < j$. If an edge between r_i and r_j in *G* does not exist, then a linear arrangement of *G* can interchange the order of the two rules. An edge will not exist if the rules do not intersect; however, a reorder will not affect integrity since a packet cannot match both rules. Shadowing is not introduced due to the reorder since the intersection operation is symmetric. An edge will not exist if the two rules intersect but have the same action; however, a reorder will not affect integrity since the same action will occur regardless of which rule is matched first. If an edge does exist between the rules, then their relative order will be maintained in every linear arrangement of *G*; thus maintaining precedence and integrity.

Rule List Optimization

As described above, it is important to inspect packets as quickly as possible given increasing network speeds and QoS requirements. Using the rule list DAG to maintain policy integrity, a linear arrangement is sought that minimizes the average number of comparisons required. However, this will require information not present in the firewall rule list. Certain firewall rules have a higher probability of matching a packet than others. As a result, it is possible to develop a policy profile over time that indicates frequency of rule matches (similar to cache hit ratio). Let $P = \{p_1, p_2, ..., p_n\}$ be the policy profile, where p_i is the probability that a packet will match rule *i* (first match in the policy). Furthermore, assume a packet will always find a match, $\sum_{i=1}^{n} p_i = 1$; therefore *R* is comprehensive. Using this information, the average number of rule comparisons required is

$$E[n] = \sum_{i=1}^{n} i \cdot p_i \qquad (1)$$

For example, the average number of comparisons required for the rule set in Table 1 is 5.03.

Given a rule list DAG $G = (R, E)$ and policy profile $P = \{p_1, p_2, ..., p_n\}$ a linear arrangement π of G is sought that minimizes Equation 1. In the absence of precedence relationships, the average number of comparisons is minimized if the rules are sorted in non-increasing order according to the probabilities [26], which is also referred to as Smith's algorithm [27]. Precedence constraints cause the problem to be more realistic; however, such constraints also make determining the optimal permutation more problematic.

Determining the optimal rule list permutation can be viewed as job scheduling for a single machine with precedence constraints [15, 21]. The notation for such scheduling problems is $\alpha|\beta|\gamma|\delta$, where α is the number of machines, β is the precedence (or absence of) which can be represented as a DAG, γ is a restriction on processing time, and δ is the optimality criterion [15]. Determining the optimal rule order is similar to the $1|\beta|1|\sum \omega C_i$ scheduling problem, or optimality criterion, where ω is a weight associated with a job (for example, importance) and C_i is the completion time. As previously noted, the $1|||\sum \omega C_i$ problem can be solved in linear time the using Smith's algorithm [27], which orders jobs according to non-decreasing $\frac{t_i}{\omega}$ ratio, where t_i is the processing time of job i. In this case set $t_i = 1$ and $\omega = p_i \ \forall i$. However, Lawler [19] and Lenstra et al. [21] proved $1|\beta|1|\sum \omega C_i$ to be *NP*-complete via the linear arrangement problem, which implies determining the optimal firewall rule order is also *NP*-complete. Note, determining the number of possible permutations has been proven to be *NP*-hard [5].

Theorem $1|\beta|1|\sum \omega C_i$ α *Determining the optimal order of a firewall rule list*

Proof Consider the $1|\beta|1|\sum \omega C_i$ problem. Each of n jobs J_i, $i \in I$, has to be processed without preemption on a single machine that can handle at most one job at a time. For each $i \in I$, let ω be the associated weight. Furthermore, let $G = (V, E)$ be a DAG that represents the precedence order of the jobs J_i. Assume the processing time of each job equals 1 time unit, the weights to be $0 \leq \omega \leq 1$ such that $\sum \omega = 1$, and β, which is G, to be a rule list DAG. In this case, the

optimization criterion $\sum \omega \cdot C_i$ is the same as $\sum p_i \cdot i$, which is given in equation 1. Clearly, the optimal firewall rule ordering problem has a solution if and only if $1|\beta|1|\sum \omega C_i$ has a solution. Therefore, determining the optimal permutation of firewall rules is *NP*-complete.

Exemplary Rule Sorting Algorithm

Although determining the optimal rule permutation was proven to be *NP*-complete, reducing the average number of comparisons required to process a packet remains an important objective. As previously discussed, a sorting algorithm must maintain the precedence relationships among rules. Of course an exhaustive search is possible if the number of rules is small (generate and test all possible topological orders); however as proven in the previous section, this is not feasible with a realistic rule list.

One exemplary algorithm starts with the original the rule set, then sorts neighboring rules based on nonincreasing probabilities. However, an exchange of neighbors should never occur if the rules intersect and have different actions. This test preserves any precedence relationships in the policy. For example, the following sorting algorithm uses such a comparison to determine if neighboring rules should be exchanged. Note a_i denotes the action associated with the ith rule.

```
done = false
while(!done)
    done = true
    for(i = 1; i <n; i++)
        if(p_i < p_i+1 AND (r_i ∩ r_i+1 OR a_i == a_i+1))then
            exchange rules, actions, and probabilities
            done = false
        endif
    endfor
endwhile
```

Other sorting algorithms are possible if the completion time of the sort is an issue; however, the method presented is easy to implement and only requires a simple neighbor comparison. Assume the match probabilities for the rule list given in Table 1. Applying the sorting algorithm to this rule list results in the ordering depicted in Figure 2A, which has 10% fewer comparisons on

average. However when using the algorithm, it is possible that one rule can prevent another rule from being reordered. For example, rule r_1 prevents rule r_5 from being placed closer to the beginning of the rule set. However, if both rules r_1 and r_5 are placed closer to the beginning of the policy while maintaining their relative order, the average number of comparison will be further reduced, 15% fewer. This is the optimal order for the 6 rule set, which is depicted in Figure 2B. Although this simple sorting algorithm is unable to move groups of rules, it can still improve the performance of the firewall system. Its effectiveness is measured experimentally in the next section.

Experimental Results

In this section, the average number of rule comparisons required after sorting the rules is compared with average number of comparisons required using the original order and the optimal order. Note the optimal rule ordering was determined via an exhaustive search. As a result, it was not feasible to determine the optimal ordering once the number of rules equaled 15 given the number of permutations to consider. Two different variations of the rule sorting method described in the preceding section were implemented. The difference between the methods was the comparison used to determine if neighboring rules should be exchanged. The first sorting algorithm exchanged neighboring rules if they were out of order and did not intersect. The if-condition was as follows.

if ($p_i < p_{i+1}$ **AND** $r_i \not\cap r_{i+1}$) **then**

Therefore, rule actions were not considered and the method will be referred to as *non-action sort*. The second sorting method did consider the rule action (as described in the preceding section) and will be referred to as *action sort*. The comparison between the two sorting algorithms will indicate the importance of considering rule actions when ordering rules.

In the first experiment, lists of 10 firewall rules were generated with random precedence relationships. The match probability of each rule was given by a Zipf distribution [20], which assigns probabilities according to the rank of an item. For this simulation, the last rule had the highest probability

which is consistent with most policies (last rules are more general). As a result, the original order yields the worst average number of comparisons. The intersection percentage measured the percentage of rules that intersect in the policy. This metric gives a value for the dependency level in the policy; however note, rule action is not considered when calculating this metric. Rules were equally likely to have an accept or deny action and the results (average number of rule comparisons) for a particular intersection percentage were averaged over 1000 simulations.

Results of the first experiment are given in Figures 3A and 3B. More particularly, Figure 3A is a graph illustrating the average number of packet comparisons versus intersection percentage for the original rule sorting, the non-action sort, the action sort, and the optimal sort. Figure 3B is a graph illustrating the percent difference in the average number of comparisons required between the sorted and optimal firewall policy configurations. The average number of comparisons required was lower for the sorted and the optimal lists when the intersection percentage was low, as seen in Figure 3A. This was expected since there is a large number of possible rule permutations (few edges in the DAG). When the intersection percentage approached 100%, the values converged to the number required for the original list. This is due to the limited number of possible rule orders, only one order in the extreme case. The percent difference between the sorted and optimal order is shown in Figure 3B. At zero intersection percentage the sorted and optimal orders are equal, since any ordering is possible. Similarly at an intersection percentage of 100, the sorted and optimal orders are equal since only one order is possible. Between these two extremes, the action sorting algorithm remains close to the optimal value. In contrast, the non-action sort had a maximum difference of 33%.

The benefit of rule sorting on larger policies is also of interest; however as previously described, it was not possible to determine the optimal ordering once the number of rules approaches 15. Therefore, the second experiment used larger rule sets, but only compared the original order and the sorted orders. The number of rules ranged from 10 to 1000, while the matching probabilities and intersection percentages were the same as the previous

IPR2023-01353 Page 00669

experiment. The results of this experiment are depicted in Figure 4A. The sorted rule sets always performed equal to or better than the original order, while the action sort consistently performed better than the non-action sort. As noted in the previous experiment, the percent difference is very large given few intersections (e.g. 80% decrease for 1000 rules with 0% dependency), but approaches zero as all the rules intersect. As the number of rules increases, sorting is increasingly beneficial, although only at low intersection percentages. The benefit of sorting drops more significantly as the intersection percentage increases with larger rule sets. This is primarily due to the low matching probabilities of each rule, which requires a complete reordering to have a significant impact on the average number of comparisons. As a result, large rule sets can benefit from sorting if the intersection percentage is low.

The impact of considering rule actions when sorting is illustrated in Figure 4B. In this experiment lists of 1000 rules were generated, where the intersection percentages ranged from 0% to 100% and the percentage of rules with the same action varied from 50% to 100%. The performance of the action sort to the original list was then compared. As the percentage of rules with the same action increased, the percent difference (reduction) in the average number of comparisons also increased. This is due to the increased number of permutations possible when the rule actions are increasingly the same (fewer edges in the rule list DAG to consider). This is depicted in Figure 4B where a policy with a 100% intersection percentage can significantly reduce the number of comparisons (80% reduction) if all the rules have the same action. This performance increase is not possible with the non-action sort. Therefore, considering the rule action increases the number of possible rule orders, thereby providing more possibilities to improve firewall performance.

Conclusions

Network firewalls enforce a security policy by comparing arriving packets to a list of rules. An action, such as accept or deny, is then performed on the packet based on the matching rule. Unfortunately packet filtering can impose significant delays on traffic due to the complexity and size of rule sets.

IPR2023-01353 Page 00670

Therefore, improving firewall performance is important, given network Quality of Service (QoS) requirements and increasing network traffic loads.

The sections above describe rule ordering methods to improve the performance of network firewalls. Assuming each rule has a probability of a packet matching, firewall rules should be sorted such that the matching probabilities are non-increasing. This reduces the average number of comparisons and the delay across the firewall. However, a simple sort is not possible given precedence relations across rules. It is common in a security policy that two rules may match the same packet yet have different actions. It is this precedence relationship between rules that must be maintained to preserve integrity. The method described above uses Directed Acyclical Graphs (DAG) to represent the precedence order rules must maintain. Given this representation, a topological sort can be used to determine the optimal order (minimum average number of comparisons); however, the examples above prove this problem to be *NP*-complete (similar to job scheduling for a single nonpreemptive machine with precedence constraints). As an alternative, a simple sorting method was introduced that maintained the precedence order of the rules. Simulation results indicate this method can significantly reduce the average number of comparisons required and is comparable to optimal ordering.

Several areas exist for future research in optimizing firewall rule lists. The sorting method proposed above is based on a simple algorithm. Although it can offer an improvement over the original rule order, algorithms that can move groups of rules could provide a larger reduction in the average number of comparisons. The effect of stateful firewalls should also be addressed in future research. Security can be enhanced with connection state and packet audit information. For example, a table can be used to record the state of each connection, which is useful for preventing certain types of attacks (e.g., TCP SYN flood) [30, 31]. The impact of such rules on the firewall needs to be investigated and whether sorting can be done on-line to reflect temporal changes. In addition, more research is needed to determine more accurate probability distributions for packet matching and dependency percentages.

IPR2023-01353 Page 00671

Given this information, better algorithms can be designed and more realistic simulations can be performed.

Trie-Based Network Firewall Optimization Methods and Rule Splitting

Overview

As described above, while assuring firewall policy integrity is critical, performance is equally important given increasing network speeds and traffic volumes. Firewall processing delay can be reduced via hardware as well as policy optimization, where rules are manipulated to reduce the number of comparisons. For a given firewall installation, it can be determined that certain security rules have a higher probability of matching a packet than others. As a result, it is possible to develop a *policy profile* over time that indicates frequency of rule matches (similar to cache hit ratio). Given this information, trie-based methods are described herein to minimize the average number of comparisons required per packet. Although most firewall systems still utilize an ordered list representation, the proposed enhancements still are applicable to current firewall systems since a policy trie can be converted into an ordered list using an inorder traversal.

Rule Sorting

As described above, rule sorting of neighboring rules based on ascending probabilities in a manner that considers rules intersection and rule actions can be used to improve firewall performance by reducing the number of packet comparisons. The methods described above use Directed Acyclic Goal Graphs (DAGs) to represent firewalls. The methods described in this section will use the rule sorting algorithms in combination with policy trie representations to perform sorting of groups of rules in a manner that improves firewall performance.

In this example, it is assumed that a security policy contains a list of n ordered rules, $\{r_1, r_2, ..., r_n\}$. In addition, let p_i represent the probability that a packet will match rule i (first match in the policy). Therefore, the policy is comprehensive, which means every packet will have a match, if $\sum_{i=1}^{n} p_i = 1$.

IPR2023-01353 Page 00672

Using this information, the average number of rule comparisons required can be calculated as

$$E[n] = \sum_{i=1}^{n} i \cdot p_i$$

The average number of comparisons is minimized if the rules are sorted in non-ascending order according to the probabilities. However, this can only be achieved if the sorting algorithm considers the probabilities and whether one rule is a superset of another. For example, sorting algorithms must use the following comparison to determine if neighboring rules should be interchanged Note, a_i denotes the action associated with the ith rule.

if ($p_i < p_{i+1}$ **AND** $r_i \not\subset r_{i+1}$ **OR** $a_i == a_{i+1}$)) **then**

Sorting rules in this fashion can have positive impact on the average number of comparisons required. Figure 5A is a table illustrating a firewall policy prior to rule sorting. The expected number of comparisons for the policy of Figure 5A is 4.26 comparisons per packet. Figure 5B is a table illustrating the firewall policy of Figure 5A after sorting using the above-described algorithm. In Figure 5B, the expected number of comparisons per packet has been reduced to 3.94. Thus, using the method described above, the number of comparisons per packet can be reduced by 8%.

However, it can be seen from Figures 5A and 5B that one rule can prevent another rule from being reordered. For example in Figure 5B, rule r_1 prevents rule r_4 from being placed closer to the beginning of the rule set. However, if both rules r_1 and r_4 are placed closer to the beginning of the policy while maintaining their relative order, the average number of comparison will be reduced.

To solve this problem, rule sets can be sorted a using policy tries. First, the rule list is converted into an equivalent policy trie. Each sub-trie will have an associated probability $\overline{p}$ that is average probability (hit-ratio) of the rules comprising the sub-trie. Sub-tries can be rotated around their parent node to increase performance, using the method described earlier. Since the policy trie (or equivalent rule set) is tested from left to right, rotation should occur if the

probability of the right sub-trie is greater than the left sub-trie. As a result, the average number of comparisons required will be reduced and the policy integrity is maintained. Figure 6A is a trie representation of the rules in Figure 5B. Figure 6B is a table representing resulting rule set after rotating the TCP and UDP sub-tries. The TCP sub-trie in Figure 6A has an expected packet matching probability of 0.095 while the UDP sub-trie has an expected packet matching probability of 0.155. Thus, the UDP sub-trie and the TCP sub-trie should be rotated so that the UDP sub-trie becomes the left sub-trie and the TCP sub-trie becomes the right sub-trie. In the table illustrated in Figure 6B, which lists the ordering of the rules after rotating the sub-tries, the average number of comparisons per packets has been reduced to 3.70.

Rule Splitting

Rule splitting takes a general rule and creates more specific rules that collectively perform the same action over the same set of packets. Here, rule splitting is used to reduce the average number of comparisons. For example in Figure 7A, rule r_5 is split into two separate rules, r_{5a} for UDP and r_{5b} for TCP. Figure 7B is a table corresponding to the policy trie of Figure 7A. Once the rules are positioned based on their probabilities and their relation to other rules, the average number of rule comparisons is reduced to 2.98 (after sort, trie rotation, and splitting) which is a 30% less.

It many not be advantageous to split a general rule since it adds another rule to the policy. For example, assume a policy contains 20 rules where the first 19 rules have the same probability. Assume the last rule can be split and the new specific rule has a probability that is $\frac{3}{4}$ of the last rule. The impact of the probability of the last rule and the location of the new split rule, m, is depicted in Figures 8A and 8B. More particularly, Figure 8A is a graph illustrating the average number of comparisons for firewalls with and without rule splitting. Figure 8B is a graph illustrating average number of comparisons versus different locations of the new rule and probabilities. The average number of comparisons is reduced as the split rule is located closer to the first rule (surface decreases as m approaches one). Furthermore, splitting yields

IPR2023-01353 Page 00674

better results when the general rule has a high probability. However as illustrated in Figure 8B, the closer the specific rule is to the location of the original rule the average number of comparisons increases, which is the penalty of adding one more rule to the policy.

The effect of splitting a single rule can be described mathematically as follows. Consider n rules where rule r_n can be split into rules $r_{n,1}$ and $r_{n,r}$ (whose union is the original rule). In addition, assume the split rule $r_{n,1}$ will be located at the mth position ($1 \leq m < n$), while rule $r_{n,r}$ will remain at the nth location. The goal is to determine the best location m, which yields a lower average number of comparisons as compared to the original rule set. This can be defined mathematically in the following formula.

$$\sum_{i=1}^{n} i \cdot p_i > \sum_{i=1}^{m-1} i \cdot p_i + m \cdot p_{n,l} + \sum_{i=m}^{n-1} (i+1) \cdot p_i + (n+1) \cdot p_{n,r}$$

The left side of the inequality is the average number of comparisons for the original rule set. The right hand side of the inequality is the average number of comparisons with the specific rule at location m. If it is assumed that the rules located between m and n have an equal probability (denoted as p) the previous equation can be solved for m.

$$m < \frac{n \cdot p - n \cdot p_n + (n+1) \cdot p_{n,r}}{p - p_{n,l}}$$

The new rule must be located between the first and m^{th}; however, its final location will depend on the relationship with the other rules (cannot be placed before any rule for which it is a superset). This result can be applied iteratively to multiple rules and repeatedly to the same rule.

Rule Optimization for QoS

In the preceding sections, optimizing the entire security policy has been discussed. However, it may be desirable to optimize the rules for a certain type of traffic, in order to reduce the average number of comparisons this traffic encounters. This can be done by organizing the policy trie via traffic

IPR2023-01353 Page 00675

classification, then optimizing the sub-tries. The result is an ordering where the rules for the most important traffic are tested first.

Network Firewall Optimization Using Rule Compression

Another method for network firewall optimization is referred to as rule compression. Given a security policy R, one objective in optimizing firewall performance is to reduce number of rules in the policy while maintaining the original integrity. A method for reducing the number of rules called *rule compression*, which combines and replaces two rules in the policy will now be described. Although compression does reduce the number of rules, it can only be done if the integrity of the original policy is maintained.

Consider two rules r_i and r_j in policy R, where r_i appears before r_j. The two rules can be compressed if and only if the following conditions are true

1. The two rules intersect, $r_i \cap r_j \neq \emptyset$ (an edge between r_i and r_j exists in the policy DAG) (Tuples that are proper subsets and supersets are considered. Compression may still occur if the rules do not intersect or are adjacent in the set space, the resulting rule may contain ranges.)
2. The action of r_i and r_j is the same.
3. Rule r_i is not dependent on any rule r_k that is directly or indirectly dependent on r_j (only one path from r_i to r_j exists in the policy DAG).

The result of compression is a new rule, $r_{i,j}$ whose tuples are the union of the corresponding tuples of rules r_i and r_j. The new rule $r_{i,j}$ replaces both r_i and r_j in R; however, the location of $r_{i,j}$ in R may require the relocation of other rules. Let D_j be the ordered set of rules that appear after r_i that directly or indirectly depend upon r_j. The new rule is placed at the original location of r_i in R and the rules D_j are placed before $r_{i,j}$.

For example, consider the policy given in Table 2. It is possible to compress rules r_1 and r_5, creating the new rule $r_{1,5}$ =[TCP, *, *, 2.*, *, accept]. The ordered set D_j in this example consists of the rules r_3 and r_4. As a result, the new rule $r_{1,5}$ is placed at the original location of r_1 while the rules in D_j are placed before the new rule. Figures 9A and 9B respectively illustrate the

IPR2023-01353 Page 00676

original policy DAG for Table 2 and the policy DAG after rules r_1 and r_5 are compressed.

No.	Proto.	Source IP	Source Port	Destination IP	Destination Port	Action
1	TCP	1.1.*	*	2.2.*	*	accept
2	TCP	1.*	*	*	80	deny
3	TCP	3.3.*	*	*	80	accept
4	TCP	3.*	*	*	*	deny
5	TCP	*	*	2.*	90	accept

Table 2: Example Policy Consisting of Multiple Ordered Rules

Theorem *Compressing rules r_i and r_j of policy R will maintain integrity if the three conditions are met and method described is used.*

Proof Before r_i and r_j are compressed, relocate the rules D_j immediately before r_i and D_i. This relocation will maintain integrity due to the third requirement for compression (no rule in D_i can be dependent on any rule in D_j). Now place r_j directly after r_i. This relocation will not affect integrity since no rule in D_i can have an edge to r_j (again, due to the third condition). Compressing the neighboring rules r_i and r_j creates $r_{i,j}$. This does not affect integrity since the resulting rule only matches the packets that match r_i or r_j in the original policy. Therefore, the result is the compression of rule r_i and r_j, and the integrity of the policy is maintained.

Determining if Compression Should Occur

Although compression may be possible, it may not improve the performance of the policy due to the relocation of rules. Compression should only occur if the average number of comparisons required for the new policy R_c is less than the original policy R; therefore, $E[R_c] < E[R]$.

Firewall Rule Sub-Trie and Sub-List Ordering

Policy Sub-Trie Ordering

Additional methods for firewall optimization are referred to as firewall sub-trie and sub-list ordering. Given a policy trie T, it may be desirable to order all sub-tries such that the average number of tuple comparisons is reduced and policy integrity is maintained. Consider a policy trie T that contains sub-tries T_i and T_j having the same parent node, as seen in Figures 10A-10C. Let P_i denote the sum of the probabilities of the rules contained in sub-trie i, while C_i denotes the number of comparisons required to completely traverse sub-trie i (which is equal to the number of branches). In order to reduce the average number of tuple comparisons, sub-tries that share the same parent node should be ordered such that the following two conditions are observed.

1. Sub-tries that share a parent node are ordered such that the P_i values are non-ascending (higher match probabilities occur first, from left to right).
2. If sub-tries that share a parent node have the same probability (P_i equals P_j), then order the sub-tries such that the C_i values are non-descending (sub-tries consisting of fewer comparisons occur first, from left to right).

These conditions are recursively applied throughout the policy trie as seen in Figures 11A-11D. However, these conditions do not necessarily maintain policy integrity.

Consider a group of n sub-tries that share the same parent node that are numbered sequentially from left to right. Consider any two non-intersecting sub-tries T_i and T_{i+k} in this group that are out of order. If two sub-tries do not intersect, denoted as $T_i \pitchfork T_j$, then no rule in one sub-trie will intersect with any rule in the other sub-trie, $r_k \pitchfork r_l$, $\forall r_k \in T_i$, $\forall r_l \in T_j$. The following *integrity condition* must be observed when reordering the sub-tries. The two sub-tries can be exchanged (rotated about the parent) if and only the sub-tries T_i through T_{i+k-1} do not intersect with T_{i+k} and the sub-tries T_{i+1} through T_{i+k} do not intersect with T_i. Similar to finding the linear sequence of a policy DAGs, these conditions maintain the policy trie integrity. This is observed in Figure 10A,

where no edge exists between any rule in T_1 and T_2. More particularly, Figure 10A is a policy DAG of a six-rule firewall list. Figure 10B is a policy trie representation of the firewall rule list represented by Figure 10A. In Figure 10B, the average number of tuple compares required per packet is 5.5. Figure 10C illustrates the policy trie after rotating sub-tries T_1 and T_2. In Figure 10C, the average number of tuple compares required is 4.5. In Figures 10A-10C, the following probabilities are assumed:

$P_1 = 0.5$

$C_1 = 4$

$P_2 = 0.5$; and

$C_2 = 2$.

The average number of tuple comparisons is an estimate based on the longest trie path.

Using the conditions for comparing and ordering, the following sorting algorithm sorts neighboring sub-tries that share the same parent node.

```
m = a parent node in T
done = false
while(!done)
    done = true
    for each sub-trie Ti having parent m AND a right neighboring sub-trie
        if((Pi < Pi+1 AND (Pi == Pi+1 AND Ci > Ci+1)) AND Ti ⋔ Ti+1) then
        interchange Ti and Ti+1
        done = false
        endif
    endfor
endwhile
```

Figures 11A-11D illustrate policy trie rotation for an eight-rule firewall policy. More particularly, Figure 11A illustrates the policy DAG representation of an eight-rule firewall policy. Figure 11B is a policy trie representation of the original firewall policy. In this example, ordering occurs in three stages, starting at the bottom level and moving towards the top level of the trie. In Figure 11C, the leaves are ordered at the lowest level. More particularly, nodes r_1 and r_2 are exchanged with nodes r_7 and r_8. The average number of tuple compares for the ordering represented by Figure 11C is 10.45. In Figure 11D, the sub-

IPR2023-01353 Page 00679

tries on the second lowest level (T_3, T_4, T_5, and T_6) are ordered. More particularly, Figure 11D illustrates the policy trie after exchanging sub-tries T_3 and T_4. The average number of tuple compares for the representation in Figure 11D is 10.275. In Figure 11E, the sub-tries on the second highest level, (T_1 and T_2) are ordered. Figure 11E illustrates the final ordering after exchanging T_1 and T_2. In the policy trie of Figure 11E, the average number of tuple compares required is 6.775. The average number of tuple comparisons is an estimate based on the longest trie path.

Policy Sub-List Ordering

The preceding section describes conditions for sorting policy sub-tries, which allow the exchange of groups of rules (sub-tries). The same conditions can be applied to list based policies, where sub-lists are ordered to improve the average number of rule comparisons.

In the sections above relating to rule sorting, a method is described to exchange neighboring rules in a list-based security policy, as seen in Figure 12B. However, it was observed that when ordering rules in this fashion it is possible that one rule can block the exchange of another. If groups of rules were allowed to be exchanged the average number of rule comparisons could be further reduced.

Given a policy list L, it may be desirable to order all sub-lists such that the average number of rule comparisons is reduced and policy integrity is maintained. Consider a policy list L that contains sub-tries L_i and L_j. Let P_i denote the sum of the probabilities of the rules contained in sub-list i, while C_i denotes the number of rules in sub-list i. In order to reduce the average number of rule comparisons, sub-lists should be ordered such that the following two conditions are observed.

1. Sub-lists are ordered such that the P_i values are non-ascending (higher match probabilities occur first, from left to right).
2. If sub-lists have the same probability (P_i equals P_j), then order the sub-lists such that the C_i values are non-descending (sub-lists consisting of fewer comparisons occur first, from left to right).

These conditions are applied throughout the policy list; however as with policy tries, these conditions do not necessarily maintain policy integrity.

Consider a group of n sub-lists that are numbered sequentially from left to right. Consider any two non-intersecting sub-lists L_i and L_{i+k} in this group that are out of order. If two sub-lists do not intersect, denoted as $L_i \pitchfork L_j$, then no rule in one sub-list will intersect with any rule in the other sub-list, $r_k \pitchfork r_l$, $\forall r_k \in L_i$, $\forall r_l \in L_j$. The following *integrity condition* must be observed when reordering the sub-lists. The two sub-lists can be exchanged if and only if the sub-lists L_i through L_{i+k-1} do not intersect with any rule in L_{i+k} and sub-lists L_{i+1} through L_{i+k} do not intersect with L_i. Similar to finding the linear sequence of a policy DAGs, these conditions maintain the policy trie integrity. For example, consider the sub-lists $L_1 = \{r_1, r_3\}$ and $L_2 = \{r_2, r_4\}$ given in Figure 12B. No edge exists between any rule in L_1 and L_2. The two sub-lists are out of order with respect to P and should be exchanged. The resulting list is given in Figure 12C. Figure 12A illustrates the original policy order and results in an average number of rule comparisons of 3.4. In the sorted version in Figure 12B, the average number of rule comparisons is 3.3. In Figure 12C, the average number of rule comparisons has been reduced to 2.7. Using the conditions for comparing and ordering, the following sorting algorithm sorts neighboring sub-lists.

```
done = false
while(!done)
    done = true
    for each sub-list Li having a right neighboring sub-list
        if ((Pi < Pi+1 OR (Pi == Pi+1 AND Ci > Ci+1)) AND Li ⋔ Li+1)then
            interchange rules and probabilities
            done = false
        endif
    endfor
endwhile
```

Summary of Firewall Policy Optimization Techniques and Methods

As stated above, the methods and systems described herein for a network firewall policy optimization can be implemented using any combination of hardware, software, and/or firmware. Alternatively, the steps for firewall policy optimization can be performed manually with the number of rules being optimized is small enough. Figure 13 is a flow chart illustrating an exemplary process for firewall policy optimization according to an embodiment of the subject matter described herein. Referring to Figure 13, in block **1300**, a firewall policy including an ordered list of firewall rules is defined. In block **1302**, for each rule, a probability indicating a likelihood of receiving a packet that matches each rule is specified. In block **1304**, neighboring rules are sorted in order of non-increasing probability in a manner that preserves firewall policy. In block **1306**, groups of rules are sorted in order of non-increasing probability in a manner that preserves firewall policy. For example, any of the policy-trie-based methods described above may be used. In block **1308**, one or more rules are split to reduce the number of average packet comparisons. In block **1310**, intersecting rules having common actions are identified and collapsed into single rules.

As described above, the subject matter described herein may be implemented in hardware, software, and/or firmware. In one implementation, the subject matter described herein for firewall policy optimization may be implemented on a general purpose computing platform. Figure 14 is a block diagram of a general purpose computing platform including software for firewall policy optimization according to an embodiment of the subject matter described herein. Referring to Figure 14, computing platform **1400** may be a general purpose computer, such as a personal computer, that includes a microprocessor **1402**, memory **1404**, and I/O interfaces **1406**. Microprocessor **1400** may be any suitable microprocessor, such as any of the Intel or AMD families of microprocessors. Memory **1404** may include volatile memory for running programs and persistent memory, such as one or more disk storage devices. I/O interface **1406** may include interfaces with I/O devices, such as user input devices and output devices.

IPR2023-01353 Page 00682

In the illustrated example, software that may be resident in memory **1404** includes a firewall policy editor **1408** and a firewall policy optimizer **1410**. Firewall policy editor **1408** may allow a user to define a firewall policy. For example, firewall policy editor **1408** may allow a user to define a firewall policy by specifying different rules. The rules may be specified in a graphical manner, for example using policy DAGs as described above. Alternatively, firewall policy editor **1408** may allow a user to input rules in a tabular manner, as illustrated in any of the tables described herein. Firewall policy optimizer **1410** may implement any or all of the firewall policy optimization techniques to order rules entered via firewall policy editor **1408** in a manner that preserves policy integrity and that enhances firewall performance.

A firewall rule set that is optimized using the subject matter described herein may be implemented on any firewall system that includes one or more firewalls. For example, an optimized firewall rule set according to embodiments of the subject matter described herein may be implemented using any of the hierarchical, multi-node firewall systems described in commonly assigned, co-pending U.S. patent application no. 11/316,331, filed December 22, 2005, the disclosure of which is incorporated herein by reference in its entirety. Figure 15 illustrates an example of a multi-node firewall system suitable for implementing the firewall rules according to an embodiment of the subject matter described herein. Referring to Figure 15, a plurality of firewall nodes **1500**, **1502**, **1504**,and **1506** may collectively implement any of the optimized firewall policies described herein. For example, the firewall nodes may collectively implement different portions of a firewall policy data structure including an ordered list of firewall rules. A firewall policy engine **1508** resident on each firewall node may filter packets using its local portion **1510** of the firewall policy data structure. A control node **1512** may include a firewall policy optimizer **1514** that measures the average number of comparisons per packets for the current rule configuration and may dynamically reorder the rules to reduce the average number of comparisons per packet. For example, firewall policy optimizer **1514** may utilize any of the methods described herein to rearrange rules and improve firewall performance. Rules may be tested for rearrangement at user-specified

intervals or when average number of packet comparisons increases by a user-specified amount.

References

The disclosure of each of the following references is hereby incorporated herein by reference in its entirety.

[1] E. Al-Shaer and H. Hamed. Firewall Policy Management Advisor for Anomaly Detection and Rule Editing. In *Proceedings of the IFIP/IEEE International Symposium on Integrated Network Management*, 2003.

[2] E. Al-Shaer and H. Hamed. Modeling and Management of Firewall Policies. *IEEE Transactions on Network and Service Management*, 1(1), 2004.

[3] S. M. Bellovin and W. Cheswick. Network Firewalls. *IEEE Communications Magazine*, pages 50–57, Sept. 1994.

[4] C. Benecke. A Parallel Packet Screen for High Speed Networks. In *Proceedings of the 15th Annual Computer Security Applications Conference*, 1999.

[5] G. Brightwell and P. Winkler. Counting Linear Extensions is #P-Complete. In *Proceedings of the Twenty-Third Annual ACM Symposium on Theory of Computing*, 1991.

[6] M. Christiansen and E. Fleury. Using Interval Decision Diagrams for Packet Filtering. Technical report, BRICS, 2002.

[7] D. Decasper, Z. Dittia, G. Parulkar, and B. Plattner. Router Plugins: A Software Architecture for Next-Generation Routers. *IEEE/ACM Transactions on Networking*, 8(1), February 2000.

[8] M. Degermark, A. Brodnik, S. Carlsson, and S. Pink. Small Forwarding Tables for Fast Routing Lookups. In *Proceedings of ACM SIGCOMM*, pages 4 – 13, 1997.

[9] U. Ellermann and C. Benecke. Firewalls for ATM Networks. In Proceedings of *INFOSEC'COM*, 1998.

[10] A. Feldmann and S. Muthukrishnan. Tradeoffs for Packet Classification. In Proceedings of the *IEEE INFOCOM*, pages 397 – 413, 2000.

IPR2023-01353 Page 00684

[11] E. W. Fulp. Firewall Architectures for High Speed Networks. Technical Report 20026, Wake Forest University Computer Science Department, 2002.

[12] E. W. Fulp. Firewall Policy Models Using Ordered-Sets. Technical report, Wake Forest University Computer Science Department, 2004.

[13] R. Funke, A. Grote, and H.-U. Heiss. Performance Evaluation of Firewalls in Gigabit-Networks. In *Proceedings of the Symposium on Performance Evaluation of Computer and Telecommunication Systems*, 1999.

[14] S. Goddard, R. Kieckhafer, and Y. Zhang. An Unavailability Analysis of Firewall Sandwich Configurations. In *Proceedings of the 6th IEEE Symposium on High Assurance Systems Engineering*, 2001.

[15] R. L. Graham, E. L. Lawler, J. K. Lenstra, and A. H. G. R. Kan. Optimizing and Approximation in Determinstic Sequencing and Scheduling: A Survey. *Annals of Discrete Mathematics*, 5:287 – 326, 1979.

[16] A. Hari, S. Suri, and G. Parulkar. Detecting and Resolving Packet Filter Conflicts. In *Proceedings of IEEE INFOCOM*, pages 1203–1212, 2000.

[17] HiPAC. High Performance Packet Classification. http://www.hipac.org.

[18] E. Horowitz, S. Sahni, and D. Mehta. *Fundamentals of Data Structures in C++*. Compuer Science Press, 1995.

[19] E. L. Lawler. Sequencing Jobs to Minimize Total Weighted Completion Time Subject to Precedence Constraints. *Annals of Discrete Mathematics*, 2:75 – 90, 1978.

[20] W. E. Leland, M. S. Taqqu, W. Willinger, and D. V. Wilson. On the Self-Similar Nature of Ethernet Traffic. *IEEE Transactions on Networking*, 2:1 – 15, 1994.

[21] J. K. Lenstra and A. H. G. R. Kan. Complexity of Scheduling under Precedence Constraints. *Operations Research*, 26(1):22 – 35, 1978.

[22] O. Paul and M. Laurent. A Full Bandwidth ATM Firewall. In Proceedings of the 6th European Symposium on *Research in Computer Security ESORICS'2000*, 2000.

[23] B. R. Preiss. *Data Structures and Algorithms with Object-Oriented Design Patterns in C++*. John Wiley & Sons, 1999.

[24] L. Qui, G. Varghese, and S. Suri. Fast Firewall Implementations for Software and Hardware-Based Routers. In *Proceedings of ACM SIGMETRICS*, June 2001.

[25] V. P. Ranganath and D. Andresen. A Set-Based Approach to Packet Classification. In *Proceedings of the IASTED International Conference on Parallel and Distributed Computing and Systems*, pages 889–894, 2003.

[26] R. Rivest. On Self-Organizing Sequential Search Heuristics. *Communications of the ACM*, 19(2), 1976.

[27] W. E. Smith. Various Optimizers for Single-Stage Production. *Naval Research Logistics Quarterly*, 3:59 – 66, 1956.

[28] V. Srinivasan, G. Varghese, S. Suri, and M. Waldvogel. Fast and Scalable Layer Four Switching. In *Proceedings of ACM SIGCOMM*, pages 191–202, 1998.

[29] S. Suri and G. Varghese. Packet Filtering in High Speed Networks. In *Proceedings of the Symposium on Discrete Algorithms*, pages 969 – 970, 1999.

[30] R. L. Ziegler. *Linux Firewalls*. New Riders, second edition, 2002.

[31] E. D. Zwicky, S. Cooper, and D. B. Chapman. *Building Internet Firewalls*. O'Reilly, 2000.

It will be understood that various details of the invention may be changed without departing from the scope of the invention. Furthermore, the foregoing description is for the purpose of illustration only, and not for the purpose of limitation.

CLAIMS

What is claimed is:

1. A method for producing a performance optimized firewall policy, the method comprising:
   (a) defining a firewall policy including an ordered list of firewall rules;
   (b) for each rule, defining a probability indicating the likelihood of receiving a packet matching each rule; and
   (c) sorting neighboring rules in order of non-increasing probability in a manner that preserves the firewall policy.
2. The method of claim 1 wherein sorting neighboring rules in order of non-increasing probability in a manner that preserves the firewall policy includes, for each pair of neighboring rules, rearranging the rules if a first rule of the pair has a lower probability than a second rule of the pair and the first rule of the pair is not a subset of the second rule of the pair with a different action.
3. The method of claim 1 comprising sorting groups of two or more rules in order of non-increasing probability in a manner that preserves the firewall policy.
4. The method of claim 3 wherein sorting groups of two or more rules in a manner that preserves firewall policy includes assigning each group of rules to a sub-trie in a policy trie, computing a sum of packet matching probabilities for rules in each sub-trie, and, in response to determining that a first sub-trie has a lower sum than a second sub-trie, rotating the first and second sub-tries about a common parent so that rules in the second sub-trie will be applied before the rules in the first sub-trie in the firewall policy .
5. The method of claim 4 comprising, in response to determining that the first and second sub-tries have equal sums, determining whether the second sub-trie has an lower number of branches than the first sub-trie, and, in response to determining that the second sub-trie has a lower number of branches than the first sub-trie, rotating the first and second sub-tries about a common parent so that the rules in the second sub-trie will be applied before the rules in the first sub-trie in the firewall policy.

6. The method of claim 3 wherein sorting groups of two or more rules in a manner that preserves the firewall policy includes:

(a) identifying first and second sub-lists of rules in the ordered list of firewall rules, the first sub-list preceding the second sub-list in the ordered list of firewall rules;

(b) determining a sum of packet matching probabilities for each of the first and second sub-lists;

(c) determining whether the first and second sub-lists intersect; and

(d) in response to determining that the sum of the packet matching probabilities for the second sub-list is greater than the sum of the packet matching probabilities for the first sub-list and that the first and second sub-lists do not intersect, switching the order of the first and second sub-lists in the ordered list of firewall rules.

7. The method of claim 6 comprising, in response to determining that the sums of the packet matching probabilities for the first and sub-lists are equal, determining whether the second sub-list has a lower number of rules than the first sub-list, and, in response to determining that the second sub-list has a lower number of rules than the first sub-list, switching the order of the first and second sub-lists in the ordered list of firewall rules.

8. The method of claim 1 comprising splitting at least one of the firewall rules to reduce an average number of comparisons required per received packet.

9. The method of claim 1 comprising identifying intersecting rules having a common action and collapsing the rules into a single rule representing a union of the intersecting rules.

10. The method of claim 1 wherein sorting neighboring rules includes sorting rules such that comparisons for a particular class of packets are reduced.

11. The method of claim 1 comprising assigning the rules to vertices in a directed acyclical graph (DAG), determining precedence relationships between the rules, assigning the precedence relationships to edges that connect the vertices in the DAG, and wherein sorting the neighboring

rules in order of non-increasing probability in a manner that preserves the firewall policy includes sorting the neighboring rules in a manner that preserves the precedence relationships represented by the DAG.

12. The method of claim 1 wherein the rules comprise intrusion detection rules.

13. The method of claim 1 wherein the rules comprise intrusion protection rules.

14. A system for firewall policy optimization, the system comprising:

(a) a firewall policy editor for allowing a user to define a firewall policy including an ordered list of firewall rules and for receiving input regarding probabilities indicating the likelihood of receiving a packet matching each rule; and

(b) a firewall policy optimizer for sorting neighboring rules in order of non-increasing probability in a manner that preserves the firewall policy.

15. The system of claim 14 wherein the firewall policy optimizer is adapted to, for each pair of neighboring rules, rearrange the rules if a first rule of the pair has a lower probability than a second rule of the pair and the first rule of the pair is not a subset of the second rule of the pair with a different action.

16. The system of claim 14 wherein the firewall policy optimizer is adapted to sort groups of two or more rules in order of non-increasing probability in a manner that preserves the firewall policy.

17. The system of claim 16 wherein the firewall policy optimizer is adapted to assign each group of rules to a sub-trie in a policy trie, to compute a sum of packet matching probabilities for rules in each sub-trie, and, in response to determining that a first sub-trie has a lower sum than a second sub-trie, to rotate the first and second sub-tries about a common parent so that rules in the second sub-trie will be applied before the rules in the first sub-trie in the firewall policy.

18. The system of claim 17 wherein the firewall policy optimizer is adapted to, in response to determining that the first and second sub-tries have equal sums, determine whether the second sub-trie has an lower

IPR2023-01353 Page 00689

number of branches than the first sub-trie, and, in response to determining that the second sub-trie has a lower number of branches than the first sub-trie, rotate the first and second sub-tries about a common parent so that the rules in the second sub-trie will be applied before the rules in the first sub-trie in the firewall policy.

19. The system of claim 16 wherein the firewall policy optimizer is adapted to:

(a) identify first and second sub-lists of rules in the ordered list of firewall rules, the first sub-list preceding the second sub-list in the ordered list of firewall rules;

(b) determine a sum of packet matching probabilities for each of the first and second sub-lists;

(c) determine whether the first and second sub-lists intersect; and

(d) in response to determining that the sum of the packet matching probabilities for the second sub-list is greater than the sum of the packet matching probabilities for the first sub-list and that the first and second sub-lists do not intersect, switch the order of the first and second sub-lists in the ordered list of firewall rules.

20. The system of claim 19 wherein the firewall policy optimizer is adapted to, in response to determining that the sums of the packet matching probabilities for the first and sub-lists are equal, determine whether the second sub-list has a lower number of rules than the first sub-list, and, in response to determining that the second sub-list has a lower number of rules than the first sub-list, switch the order of the first and second sub-lists in the ordered list of firewall rules.

21. The system of claim 14 wherein the firewall policy optimizer is adapted to split at least one of the firewall rules to reduce an average number of comparisons required per received packet.

22. The system of claim 14 wherein the firewall policy optimizer is adapted to identify intersecting rules having a common action and collapse the rules into a single rule representing a union of the intersecting rules.

23. The system of claim 14 wherein the firewall policy optimizer is adapted to sort neighboring rules such that comparisons for a particular class of packets are reduced.

24. The system of claim 14 wherein the firewall policy optimizer is adapted to assign the rules to vertices in a directed acyclical graph (DAG), determine precedence relationships between the rules, assign the precedence relationships to edges that connect the vertices in the DAG, and to sort the neighboring rules in a manner that preserves the precedence relationships represented by the DAG.

25. The system of claim 14 wherein the rules comprise intrusion detection rules.

26. The system of claim 14 wherein the rules comprise intrusion protection rules.

27. A network firewall comprising:

(a) a firewall policy data structure for storing an ordered list of firewall rules;

(b) a firewall policy engine for implementing a firewall policy by comparing packets to the rules in the order specified by the firewall policy data structure; and

(c) a firewall policy optimizer for optimizing performance of the network firewall by reordering the rules in a manner that reduces an average number of rule comparisons per packet and hat preserves the firewall policy.

28. The network firewall of claim 27 wherein the firewall policy optimizer is adapted to measure average number of comparisons per packet for a rule ordering specified by the firewall policy data structure and to dynamically re-order to rules to reduce the average number of comparisons per packet.

29. A computer program product comprising computer executable instructions embodied in a computer readable medium for performing steps comprising:

(a) defining a firewall policy including an ordered list of firewall rules;

IPR2023-01353 Page 00691

(b) for each rule, defining a probability indicating the likelihood of receiving a packet matching each rule; and

(c) sorting neighboring rules in order of non-increasing probability in a manner that preserves the firewall policy.

30. The computer program product of claim 29 wherein sorting neighboring rules in order of non-increasing probability in a manner that preserves the firewall policy includes, for each pair of neighboring rules, rearranging the rules if a first rule of the pair has a lower probability than a second rule of the pair and the first rule of the pair is not a subset of the second rule of the pair with a different action.

31. The computer program product of claim 29 comprising sorting groups of two or more rules in order of non-increasing probability in a manner that preserves the firewall policy.

32. The computer program product of claim 31 wherein sorting groups of two or more rules in a manner that preserves firewall policy includes assigning each group of rules to a sub-trie in a policy trie, computing a sum of packet matching probabilities for rules in each sub-trie, and, in response to determining that a first sub-trie has a lower sum than a second sub-trie, rotating the first and second sub-tries about a common parent so that rules in the second sub-trie will be applied before the rules in the first sub-trie in the firewall policy .

33. The computer program product of claim 32 comprising, in response to determining that the first and second sub-tries have equal sums, determining whether the second sub-trie has an lower number of branches than the first sub-trie, and, in response to determining that the second sub-trie has a lower number of branches than the first sub-trie, rotating the first and second sub-tries about a common parent so that the rules in the second sub-trie will be applied before the rules in the first sub-trie in the firewall policy.

34. The computer program product of claim 31 wherein sorting groups of two or more rules in a manner that preserves the firewall policy includes:

(a) identifying first and second sub-lists of rules in the ordered list of firewall rules, the first sub-list preceding the second sub-list in the ordered list of firewall rules;

(b) determining a sum of packet matching probabilities for each of the first and second sub-lists;

(c) determining whether the first and second sub-lists intersect; and

(d) in response to determining that the sum of the packet matching probabilities for the second sub-list is greater than the sum of the packet matching probabilities for the first sub-list and that the first and second sub-lists do not intersect, switching the order of the first and second sub-lists in the ordered list of firewall rules.

35. The computer program product of claim 34 comprising, in response to determining that the sums of the packet matching probabilities for the first and sub-lists are equal, determining whether the second sub-list has a lower number of rules than the first sub-list, and, in response to determining that the second sub-list has a lower number of rules than the first sub-list, switching the order of the first and second sub-lists in the ordered list of firewall rules.

36. The computer program product of claim 29 comprising splitting at least one of the firewall rules to reduce an average number of comparisons required per received packet.

37. The computer program product of claim 29 comprising identifying intersecting rules having a common action and collapsing the rules into a single rule representing a union of the intersecting rules.

38. The computer program product of claim 29 wherein sorting neighboring rules includes sorting rules such that comparisons for a particular class of packets are reduced.

39. The computer program product of claim 29 comprising assigning the rules to vertices in a directed acyclical graph (DAG), determining precedence relationships between the rules, assigning the precedence relationships to edges that connect the vertices in the DAG, and wherein sorting the neighboring rules in order of non-increasing probability in a manner that preserves the firewall policy includes sorting the

neighboring rules in a manner that preserves the precedence relationships represented by the DAG.

40. The computer program product of claim 29 wherein the rules comprise intrusion detection rules.

41. The computer program product of claim 29 wherein the rules comprise intrusion protection rules.

IPR2023-01353 Page 00694

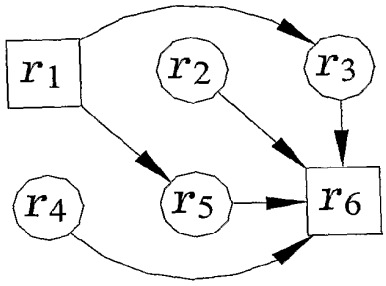


FIG. 1A

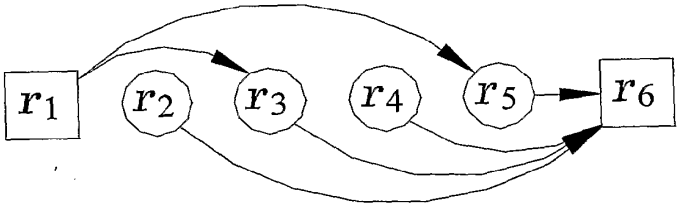


FIG. 1B

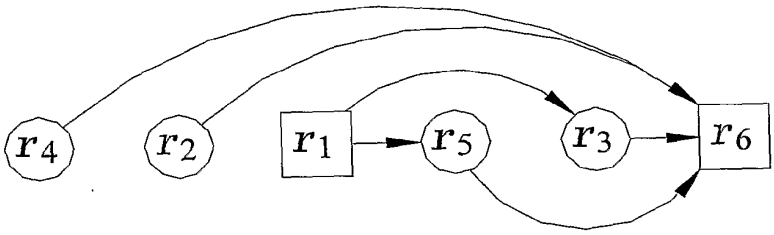


FIG. 2A

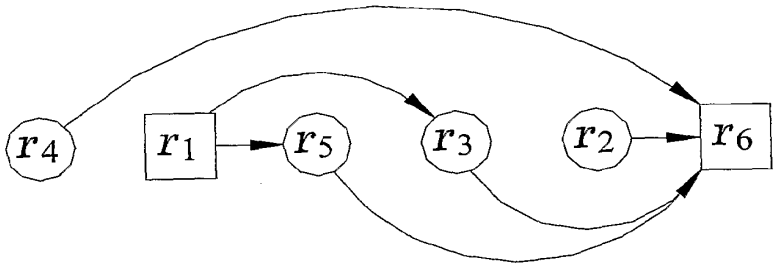


FIG. 2B

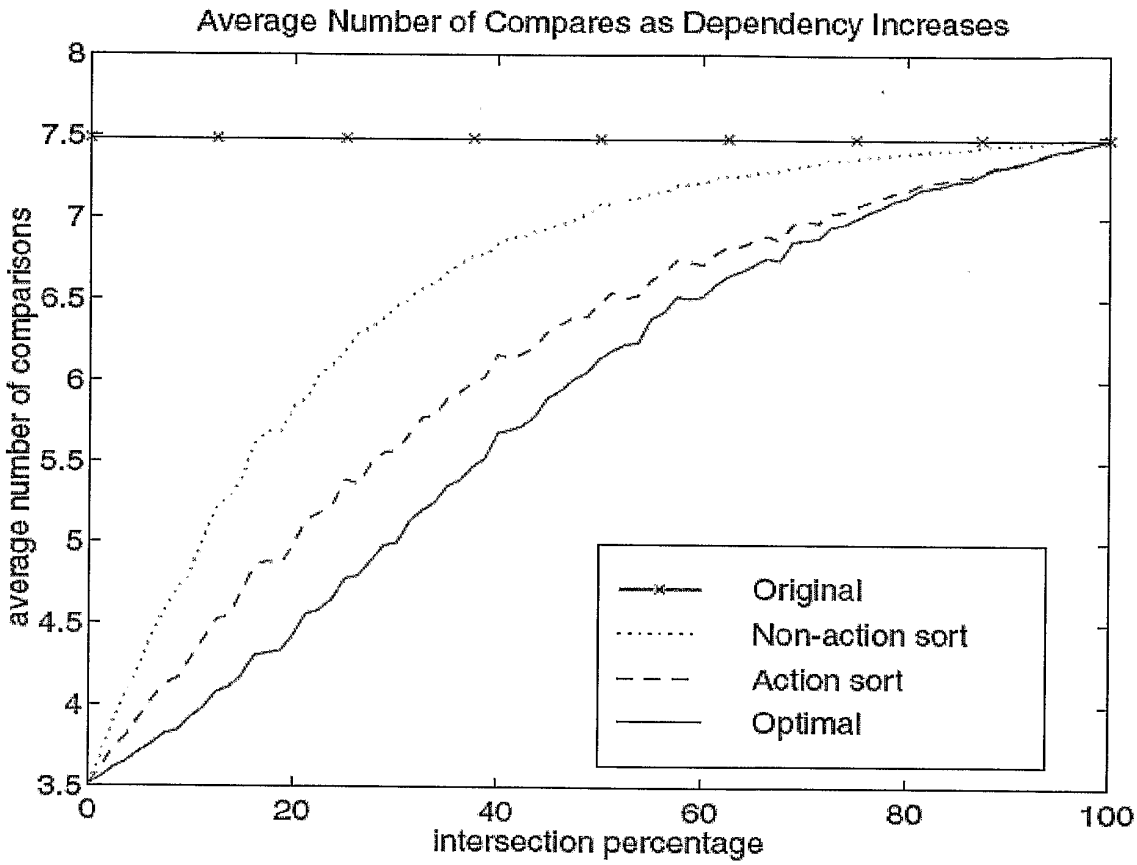

Average Number of Compares as Dependency Increases
average number of comparisons
intersection percentage
8
7.5
7
6.5
6
5.5
5
4.5
4
3.5
0
20
40
60
80
100
Original
Non-action sort
Action sort
Optimal

FIG. 3A

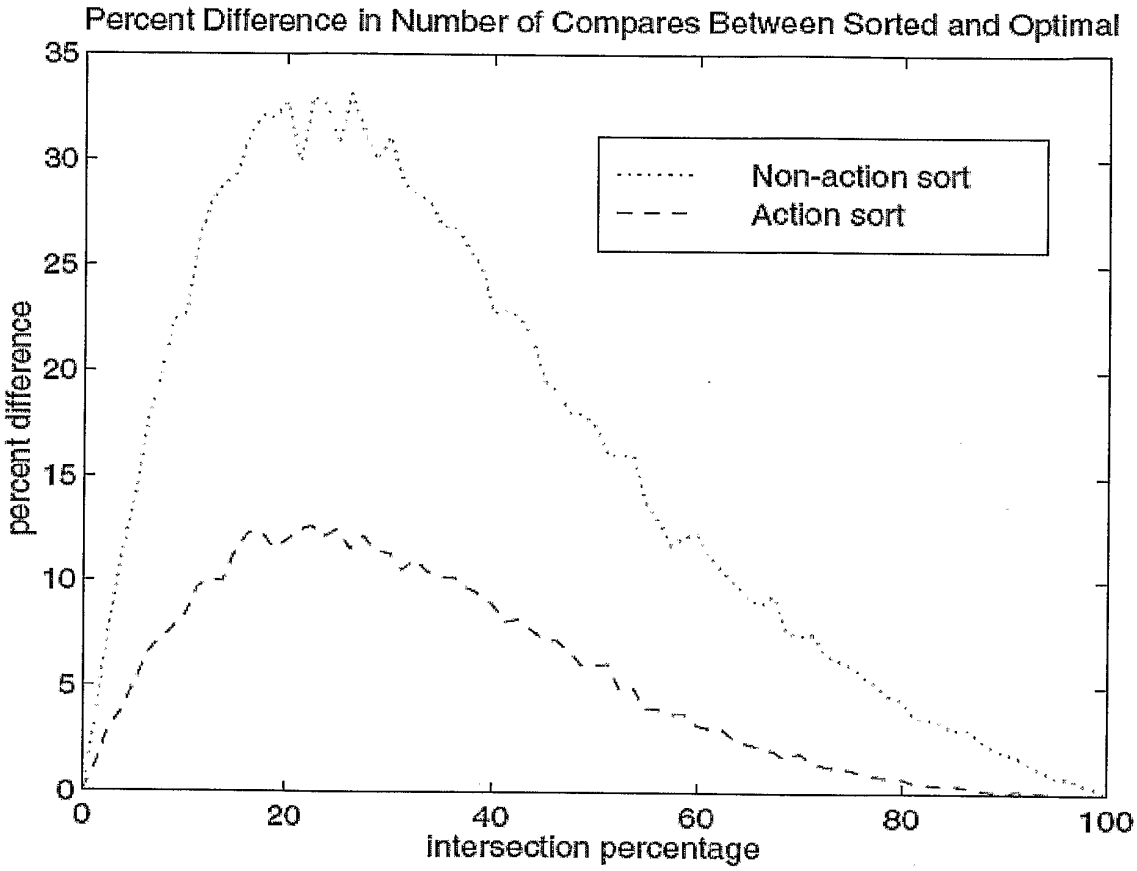

Percent Difference in Number of Compares Between Sorted and Optimal
percent difference
intersection percentage
Non-action sort
Action sort
0
5
10
15
20
25
30
35
0
20
40
60
80
100

FIG. 3B

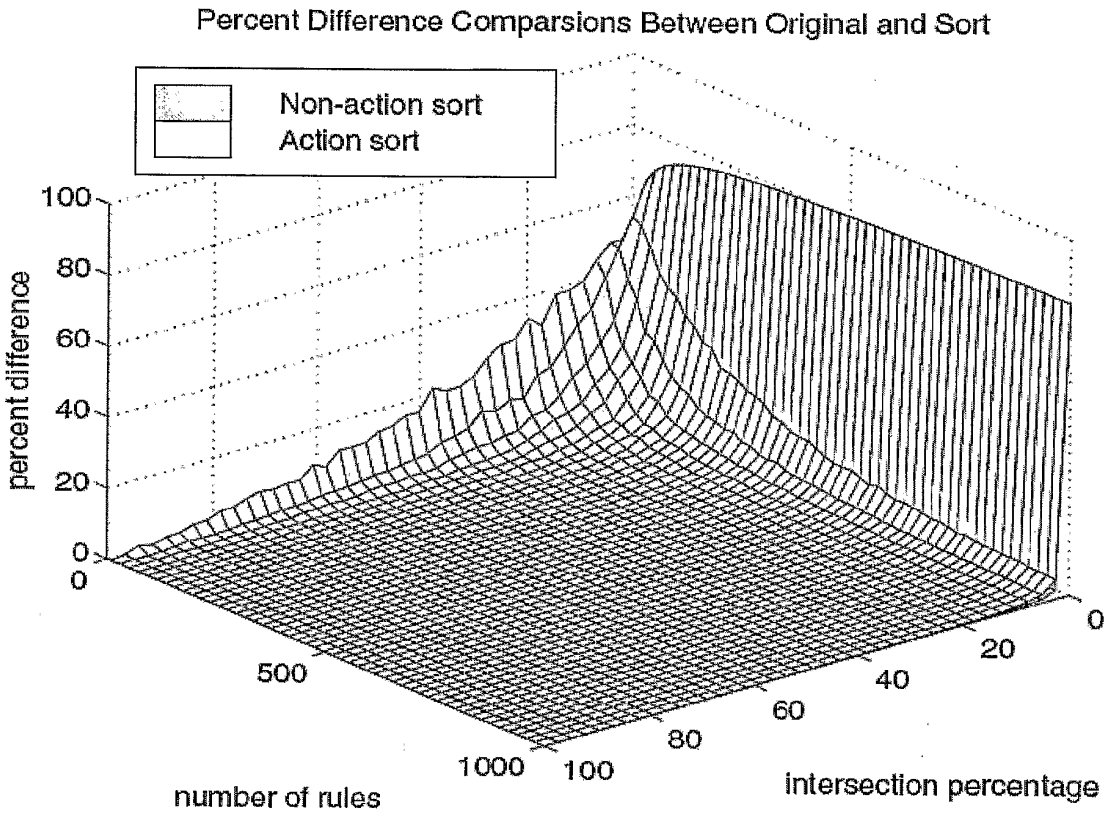

Percent Difference Comparsions Between Original and Sort
Non-action sort
Action sort
percent difference
100
80
60
40
20
0
0
500
1000
number of rules
0
20
40
60
80
100
intersection percentage

FIG. 4A

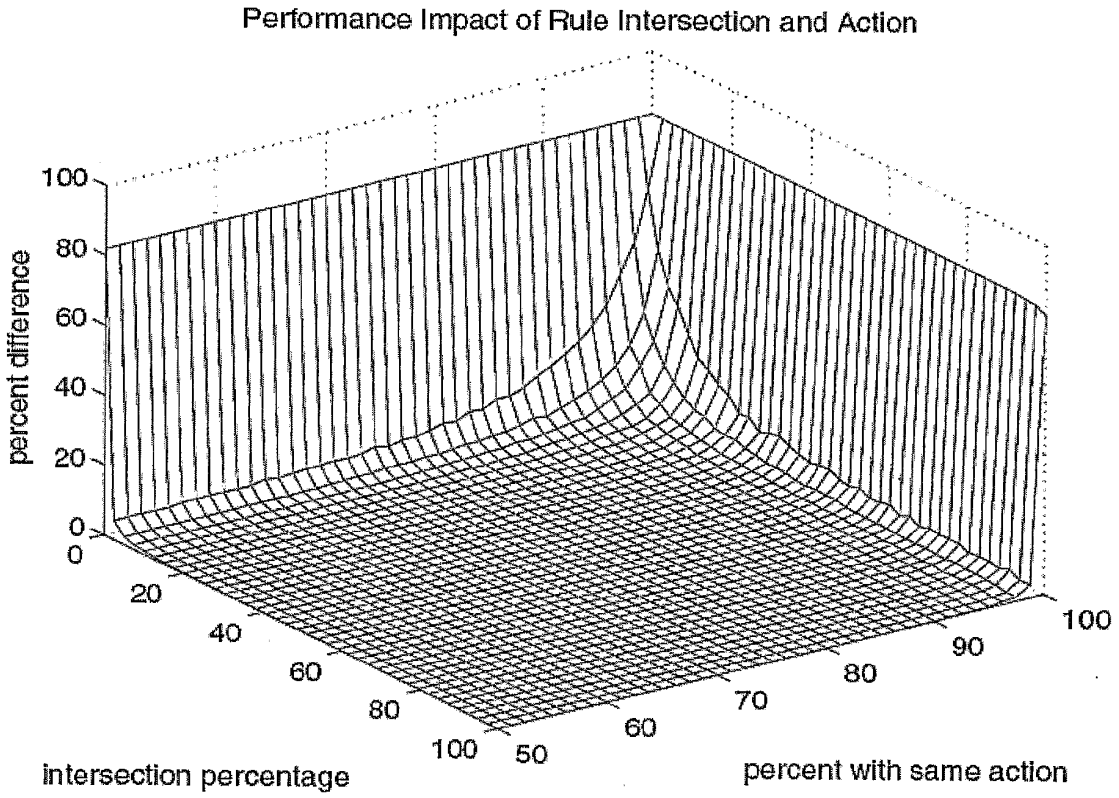

Performance Impact of Rule Intersection and Action
percent difference
100
80
60
40
20
0
0
20
40
60
80
100
intersection percentage
50
60
70
80
90
100
percent with same action

FIG. 4B

r_1	(UDP,	1.1.*,	ACCEPT)	$p_1 = 0.01$
r_2	(TCP,	1.*,	ACCEPT)	$p_2 = 0.02$
r_3	(TCP,	2.*,	ACCEPT)	$p_3 = 0.17$
r_4	(UDP,	1.*,	DENY)	$p_4 = 0.30$
r_5	(*,	*,	DENY)	$p_5 = 0.50$

FIG. 5A

r_3	(TCP,	2.*,	ACCEPT)	$p_3 = 0.17$
r_2	(TCP,	1.*,	ACCEPT)	$p_2 = 0.02$
r_1	(UDP,	1.1.*,	ACCEPT)	$p_1 = 0.01$
r_4	(UDP,	1.*,	DENY)	$p_4 = 0.30$
r_5	(*,	*,	DENY)	$p_5 = 0.50$

FIG. 5B

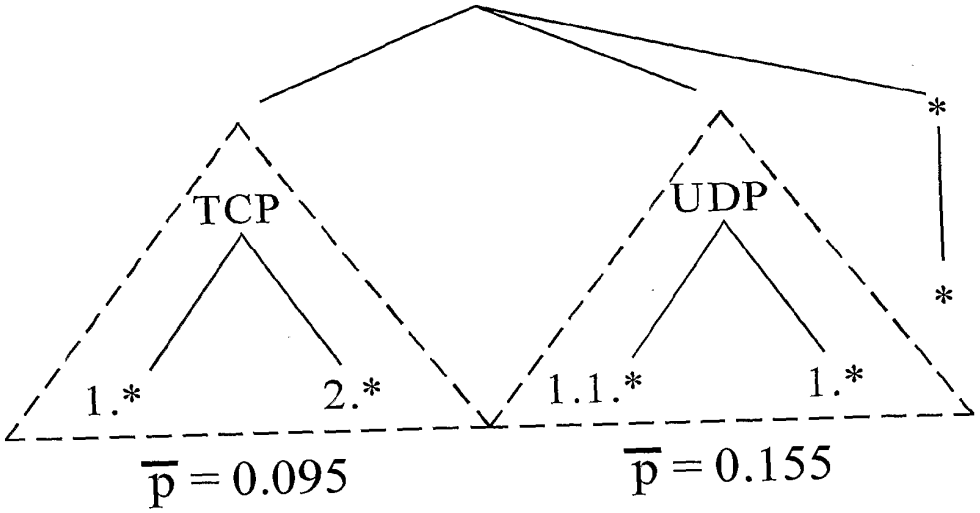


FIG. 6A

r_1	(UDP,	1.1.*,	ACCEPT)	$p_1 = 0.01$
r_4	(UDP,	1.*,	DENY)	$p_4 = 0.30$
r_3	(TCP,	2.*,	ACCEPT)	$p_3 = 0.17$
r_2	(TCP,	1.*,	ACCEPT)	$p_2 = 0.02$
r_5	(*,	*,	DENY)	$p_5 = 0.50$

FIG. 6B

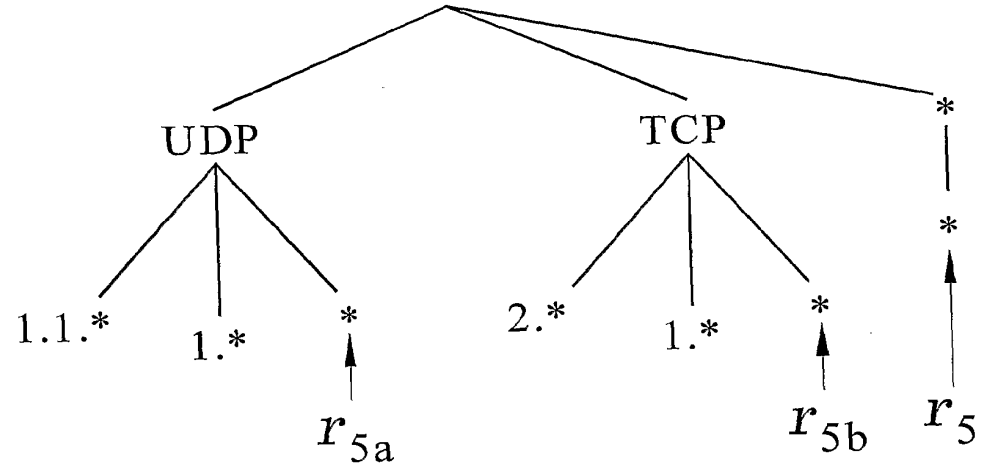


FIG. 7A

r_1	(UDP,	1.1.*,	ACCEPT)	$p_1 = 0.01$
r_4	(UDP,	1.*,	ACCEPT)	$p_4 = 0.30$
r_{5a}	(UDP,	*,	DENY)	$p_{5a} = 0.47$
r_3	(TCP,	2.*,	ACCEPT)	$p_3 = 0.17$
r_2	(TCP,	1.*,	ACCEPT)	$p_2 = 0.02$
r_{5b}	(TCP,	*,	DENY)	$p_{5b} = 0.03$

FIG. 7B

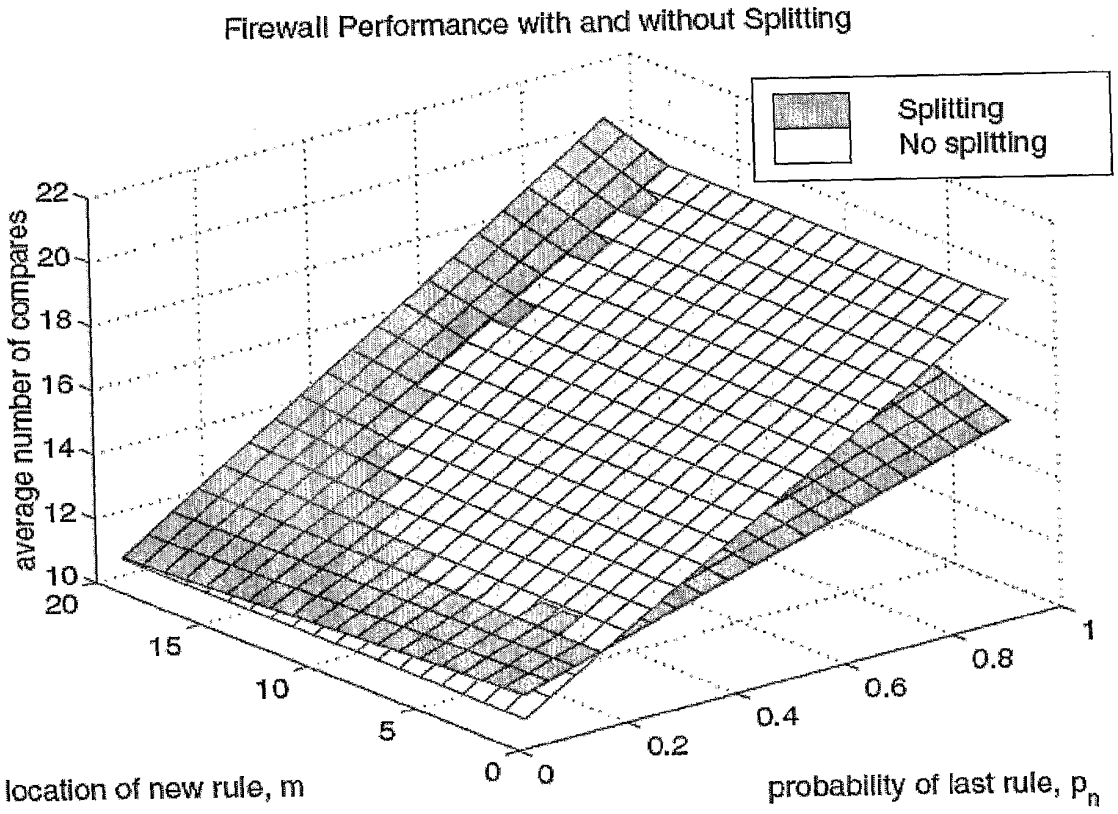

Firewall Performance with and without Splitting
Splitting
No splitting
average number of compares
22
20
18
16
14
12
10
20
15
10
5
0
location of new rule, m
0
0.2
0.4
0.6
0.8
1
probability of last rule, p_n

FIG. 8A

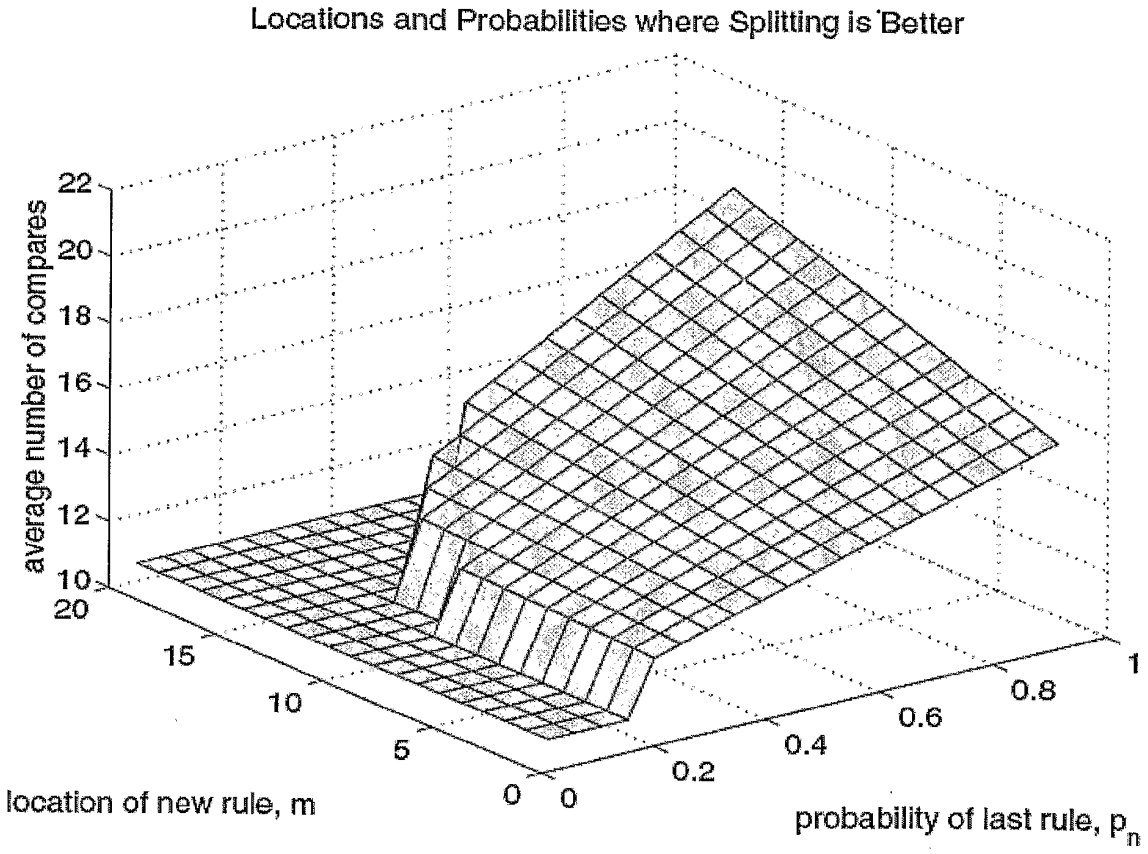

Locations and Probabilities where Splitting is Better
average number of compares
22
20
18
16
14
12
10
20
15
10
5
0
location of new rule, m
0
0.2
0.4
0.6
0.8
1
probability of last rule, p_n

FIG. 8B

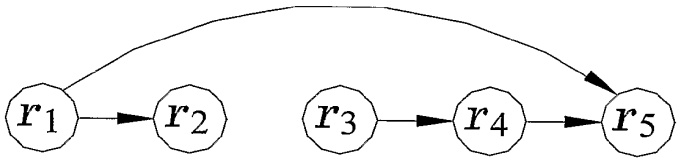


FIG. 9A

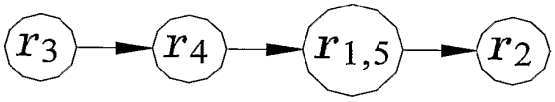


FIG. 9B

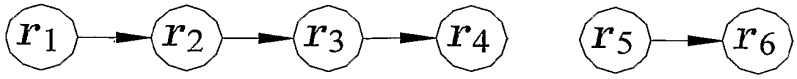


FIG. 10A

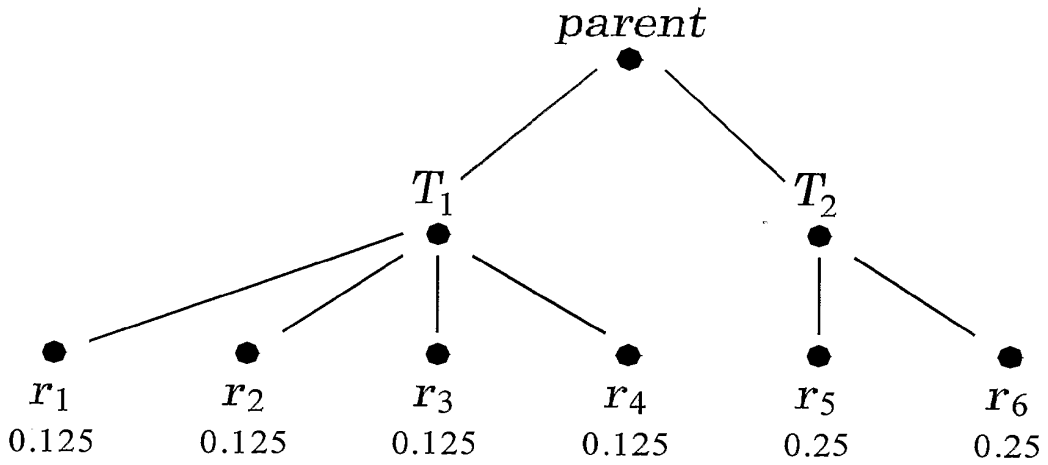


FIG. 10B

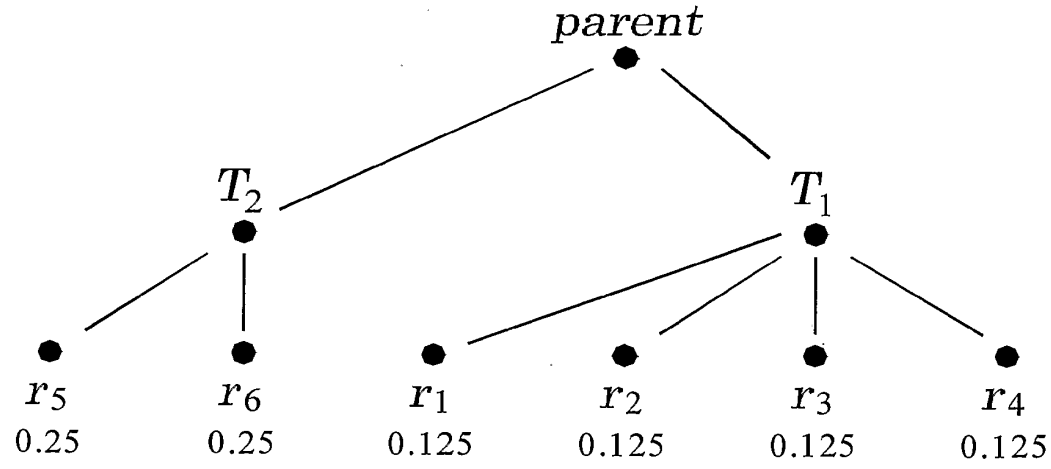


FIG. 10C

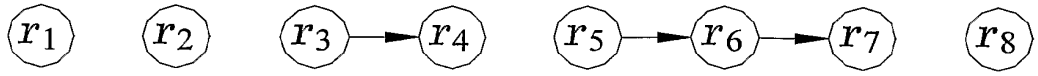


FIG. 11A

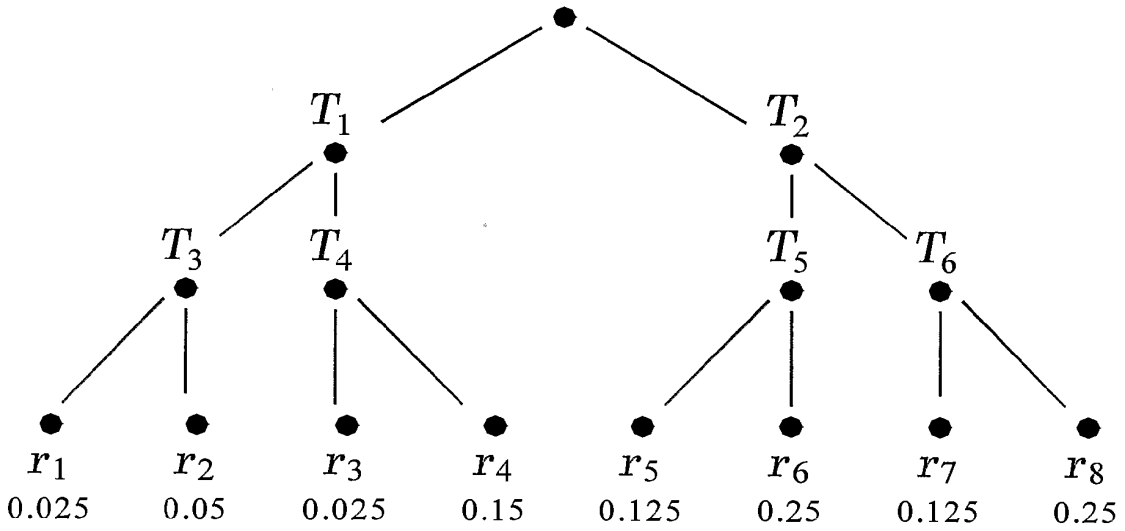


FIG. 11B

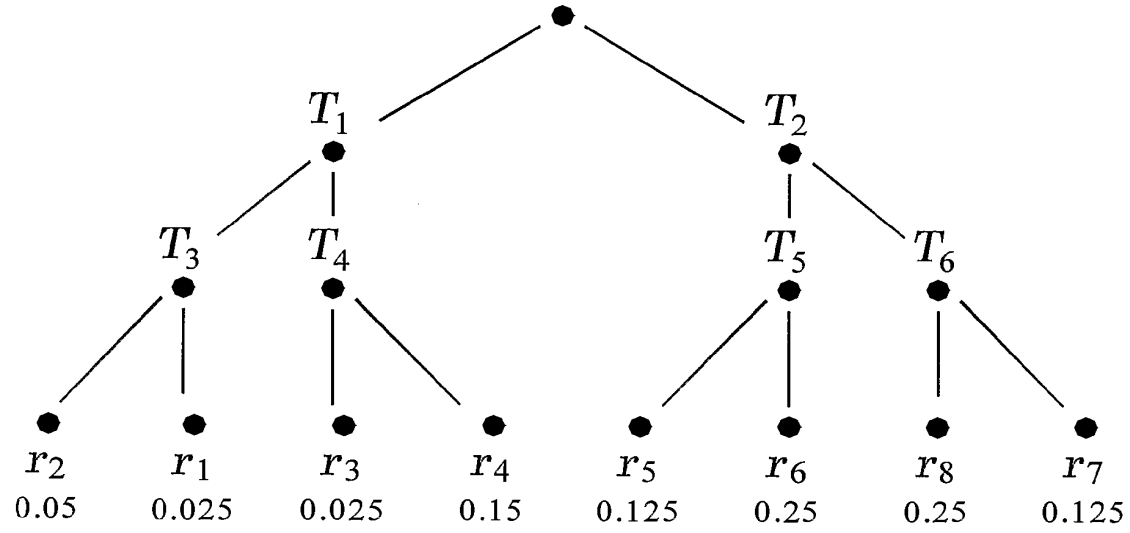


FIG. 11C

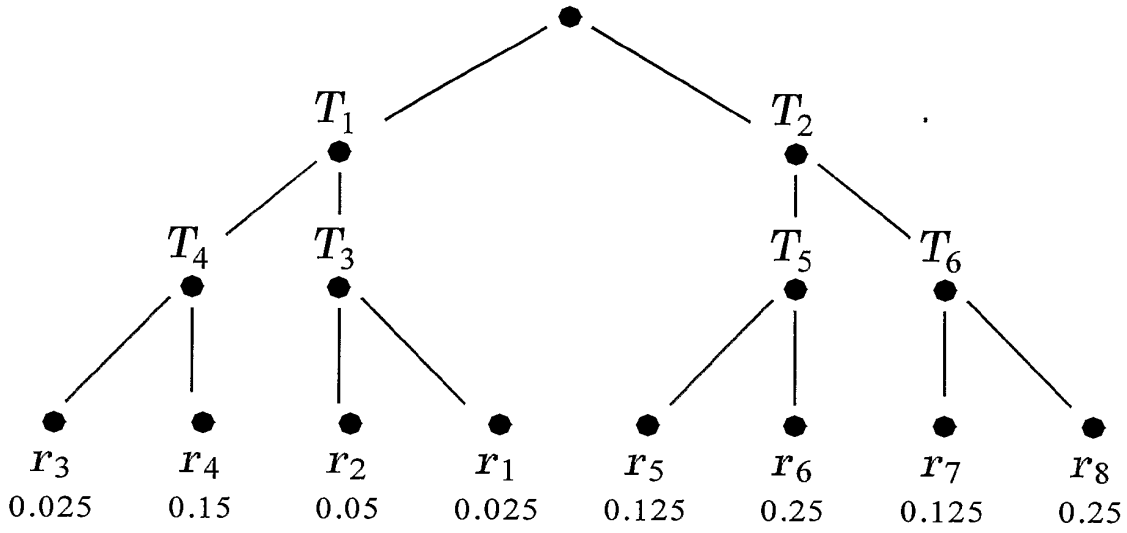


FIG. 11D

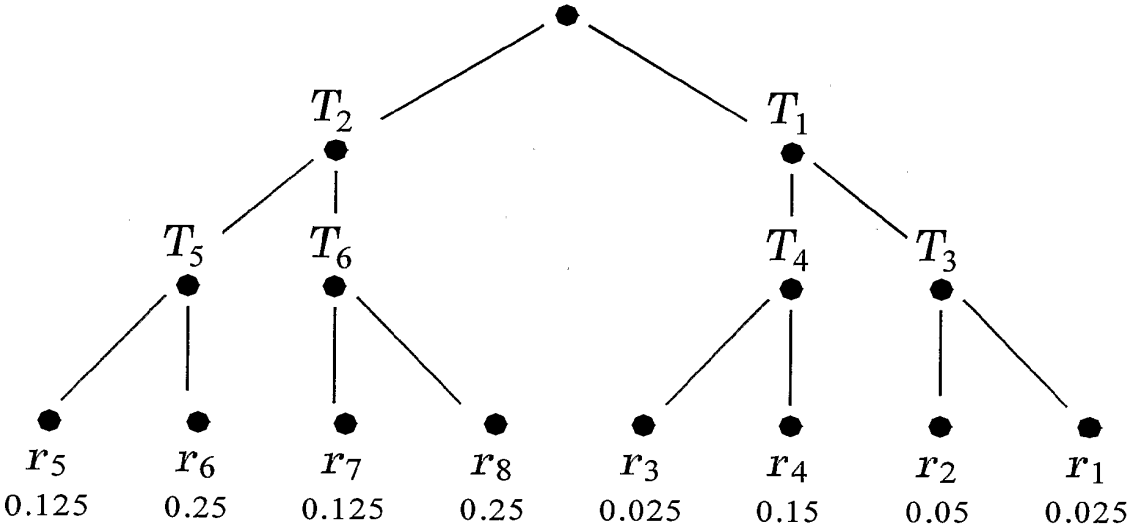


FIG. 11E

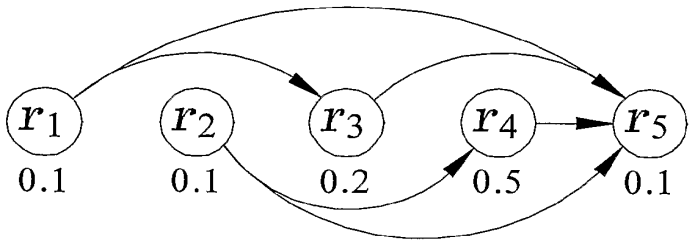


FIG. 12A

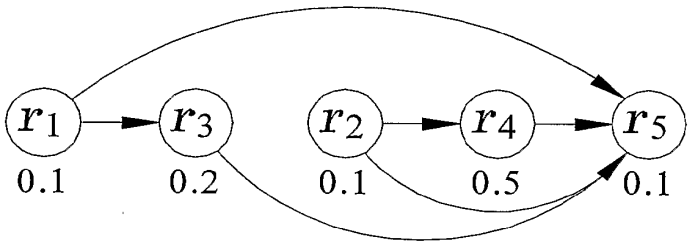


FIG. 12B

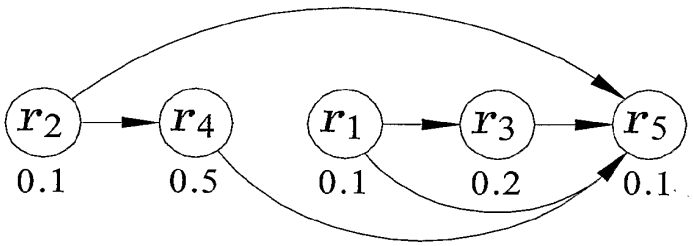


FIG. 12C

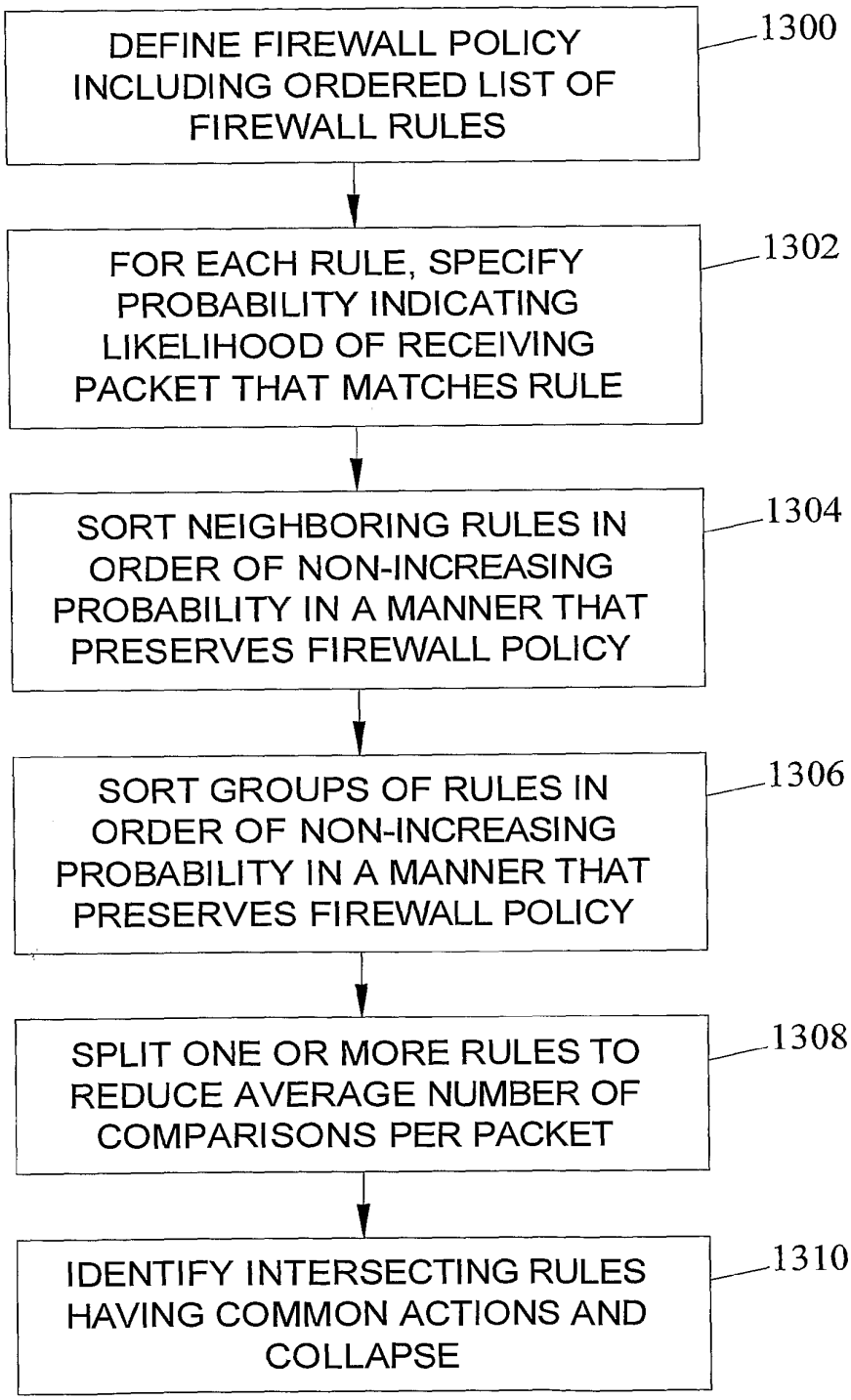

DEFINE FIREWALL POLICY INCLUDING ORDERED LIST OF FIREWALL RULES
1300
FOR EACH RULE, SPECIFY PROBABILITY INDICATING LIKELIHOOD OF RECEIVING PACKET THAT MATCHES RULE
1302
SORT NEIGHBORING RULES IN ORDER OF NON-INCREASING PROBABILITY IN A MANNER THAT PRESERVES FIREWALL POLICY
1304
SORT GROUPS OF RULES IN ORDER OF NON-INCREASING PROBABILITY IN A MANNER THAT PRESERVES FIREWALL POLICY
1306
SPLIT ONE OR MORE RULES TO REDUCE AVERAGE NUMBER OF COMPARISONS PER PACKET
1308
IDENTIFY INTERSECTING RULES HAVING COMMON ACTIONS AND COLLAPSE
1310

FIG. 13

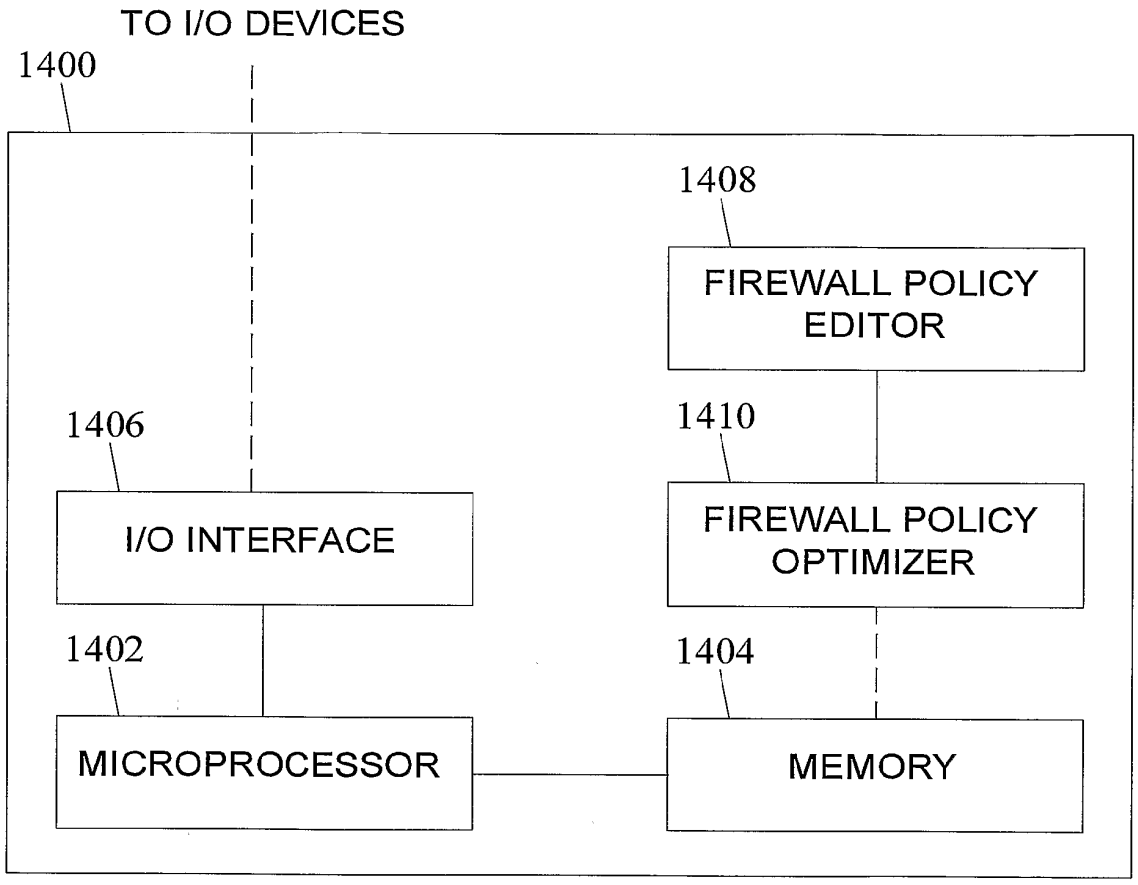

TO I/O DEVICES
1400
1408
FIREWALL POLICY EDITOR
1406
1410
I/O INTERFACE
FIREWALL POLICY OPTIMIZER
1402
1404
MICROPROCESSOR
MEMORY

FIG. 14

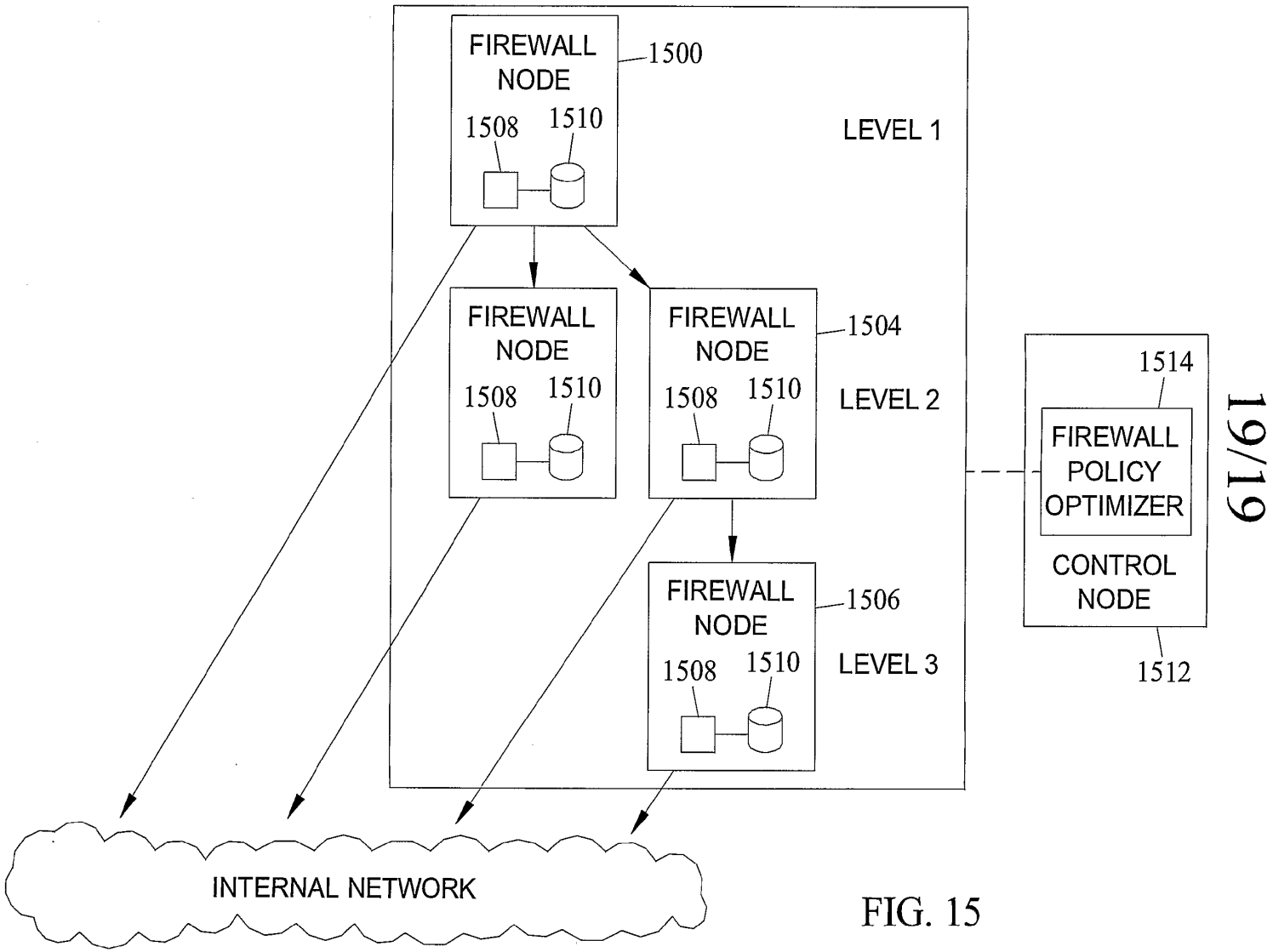

FIREWALL NODE
1500
1508
1510
LEVEL 1
FIREWALL NODE
1508
1510
FIREWALL NODE
1504
1508
1510
LEVEL 2
1514
FIREWALL POLICY OPTIMIZER
CONTROL NODE
1512
FIREWALL NODE
1506
1508
1510
LEVEL 3
INTERNAL NETWORK

FIG. 15

Espacenet

Bibliographic data: WO2006093557 (A2) — 2006-09-08

METHOD, SYSTEMS, AND COMPUTER PROGRAM PRODUCTS FOR IMPLEMENTING FUNCTION-PARALLEL NETWORK FIREWALL

Inventor(s): FULP ERRIN W [US]; FARLEY RYAN J [US] + (FULP, ERRIN, W, ; FARLEY, RYAN, J)

Applicant(s): UNIV WAKE FOREST [US]; FULP ERRIN W [US]; FARLEY RYAN J [US] + (WAKE FOREST UNIVERSITY, ; FULP, ERRIN, W, ; FARLEY, RYAN, J)

Classification: **- international:**
- cooperative: H04L63/0218; H04L63/0263

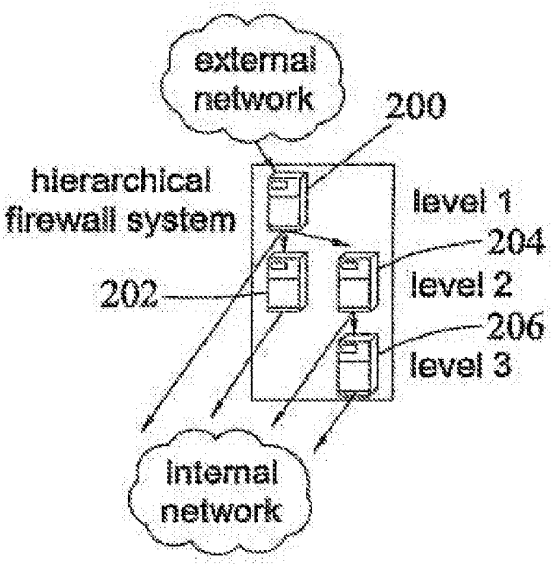
Application number: WO2005US47008 20051222

Priority number (s): US20040638436P 20041222

Also published as: WO2006093557 (A3) US2006195896 (A1) US8037517 (B2) EP1839188 (A2) EP1839188 (A4) more

Abstract of WO2006093557 (A2)

Methods, systems, and computer program products for providing function-paralell firewalls are disclosed. According to one aspect, a function-parallel firewall includes a first firewall node for filtering received packets using a first portion of a rule set including a plurality of rules. The first portion includes less than all of the rules in the rule set. At least one second firewall node filters packets using a second portion of the rule set. The second portion includes at least one rule in the rule set that is not present in the first portion. The first and second portions together include all of the rules in the rule set.



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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
8 September 2006 (08.09.2006)

PCT

(10) International Publication Number
WO 2006/093557 A2

(51) International Patent Classification: Not classified

(21) International Application Number:
PCT/US2005/047008

(22) International Filing Date:
22 December 2005 (22.12.2005)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
60/638,436 22 December 2004 (22.12.2004) US

(71) Applicant *(for all designated States except US)*: **WAKE FOREST UNIVERSITY** [US/US]; OFFICE OF TECHNOLOGY ASSET MANAGEMENT, WAKE FOREST UNIVERSITY HEALTH SCIENCES, 200 East First Street, Suite 101, Winston Salem, North Carolina 27101 (US).

(72) Inventors; and

(75) Inventors/Applicants *(for US only)*: **FULP, Errin, W.** [US/US]; 308 N. Hawthorne Road, Winston-salem, North Carolina 27104 (US). **FARLEY, Ryan, J.** [US/US]; 212 Radford Street, Winston-salem, North Carolina 27106 (US).

(74) Agent: **HUNT, Gregory, A.**; JENKINS, WILSON & TAYLOR, P.A., SUITE 1400 UNIVERSITY TOWER, 3100 Tower Boulevard, Durham, North Carolina 27707 (US).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BW, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, LY, MA, MD, MG, MK, MN, MW, MX, MZ, NA, NG, NI, NO, NZ, OM, PG, PH, PL, PT, RO, RU, SC, SD, SE, SG, SK, SL, SM, SY, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, YU, ZA, ZM, ZW.

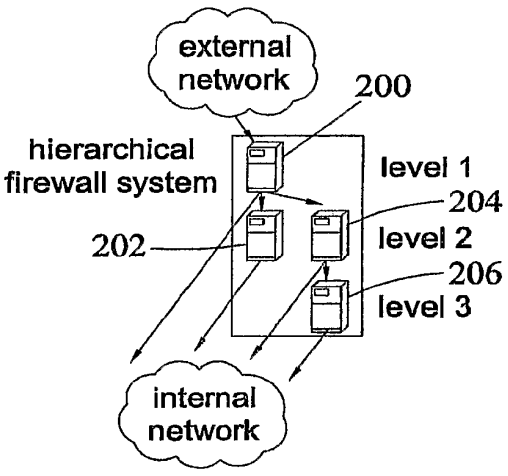
(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HU, IE, IS, IT, LT, LU, LV, MC, NL, PL, PT, RO, SE, SI, SK, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Published:

— *without international search report and to be republished upon receipt of that report*

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.

(54) Title: METHOD, SYSTEMS, AND COMPUTER PROGRAM PRODUCTS FOR IMPLEMENTING FUNCTION-PARALLEL NETWORK FIREWALL



(57) Abstract: Methods, systems, and computer program products for providing function-paralell firewalls are disclosed. According to one aspect, a function-parallel firewall includes a first firewall node for filtering received packets using a first portion of a rule set including a plurality of rules. The first portion includes less than all of the rules in the rule set. At least one second firewall node filters packets using a second portion of the rule set. The second portion includes at least one rule in the rule set that is not present in the first portion. The first and second portions together include all of the rules in the rule set.

WO 2006/093557 A2

DESCRIPTION

METHOD, SYSTEMS, AND COMPUTER PROGRAM PRODUCTS FOR IMPLEMENTING FUNCTION-PARALLEL NETWORK FIREWALL

RELATED APPLICATIONS

This application claims the benefit of U.S. Provisional Patent Application Serial No. 60/638,436, filed December 22, 2004; the disclosure of which is incorporated herein by reference in its entirety.

GOVERNMENT INTEREST

This invention was made with Government support under Grant No. DE-FG02-03ER25581 awarded by U.S. Department of Energy, Mathematical and Computational Information Sciences Division. The U.S. Government has certain rights in the invention.

TECHNICAL FIELD

The subject matter described herein relates to implementing network firewalls. More particularly, the subject matter described herein relates to methods, systems, and computer program products for implementing a function-parallel network firewall.

BACKGROUND ART

Firewalls remain the frontier defense for securing networks that are vital to private industry, government agencies, and the military. They enforce a security policy by inspecting and filtering traffic arriving at or departing from a secure network [3, 21, 22]. Meaningful inspections often involve a complex process, and if firewalls are to remain effective, they must adapt to constantly changing network demands, technology, and security threats.

As firewall technology evolves to meet new demands and threats, it must continue to act transparently to legitimate users, with little or no effect on the perceived network performance. The networks between two communicating machines should be invisible, especially if traffic requires specific network Quality of Service (QoS), such as bounds on the packet delay, jitter, and throughput. The firewall should process the legitimate traffic quickly, efficiently,

IPR2023-01353 Page 00716

and never violate the desired QoS. Unfortunately, the firewall can easily become a bottleneck, given increasing traffic loads and network speeds [12, 14, 18, 20, 21]. Packets must be inspected and compared against complex rule sets and tables, which is a time-consuming process. In addition, audit files must be updated with current connection information. As a result, current firewalls have difficulty maintaining QoS guarantees. Since a firewall can be overwhelmed with traffic, it is also susceptible to Denial of Service (DoS) attacks [21]. Such attacks merely overload/saturate the firewall with illegitimate traffic. Audit files and state tables quickly fill available storage space and legitimate traffic queued behind suspicious traffic encounters long delays. Legitimate users notice poor network performance, and, in the worst case, the secure network is disconnected from the outside world.

Inspecting traffic sent between networks, a firewall provides access control, auditing, and traffic control based on a security policy [3, 21, 22]. The security policy is a list of ordered rules, as seen in Table 1, that defines the action to perform on matching packets. A rule r can be viewed as a k-tuple $(r[1], r[2], r[3], ..., r[k])$, where $r[i]$ is specified as a variable length prefix.

No.	Proto.	Source		Destination		Action
		IP	Port	IP	Port	
1	TCP	140.*	*	*	80	accept
2	TCP	150.*	*	120	80	accept
3	TCP	140.*	*	130	20	accept
4	UDP	150.*	*	*	3030	accept
5	*	*	*	*	*	deny

Table 1: Example Security Policy Consisting of Multiple Ordered Rules

In the examples described herein, rules will initially be represented as the 5-tuple (protocol, IP source address, source port number, IP destination address, destination port number). Fields can be fully specified or contain wildcards '*' in standard prefix format. For example the prefix 192.* would represent any IP address that has 192 as the first dotted-decimal number. In addition to the prefixes, each filter rule has an action, which is to accept or deny. An accept action passes the packet into or from the secure network,

IPR2023-01353 Page 00717

while deny causes the packet to be discarded. Rules are applied in order to every arriving packet until a match is found; otherwise, a default action is performed [21, 22].

Unfortunately, security policies may contain anomalies, such as a packet that matches multiple rules, where the matching rules specify conflicting actions. Security policy anomaly detection and correction is the subject of continued research [13, 21] and is not the focus of the present subject matter. Given the type of security policy required, rule sets can become quite large and complex. Security can be further enhanced with connection state and packet audit information. For example, a table can be used to record the state of each connection, which is useful for preventing certain types of attacks (e.g., TCP SYN flood) [21, 22].

Traditional firewall implementations consist of a single dedicated machine, similar to a router, that sequentially applies the rule set to each arriving packet. However, packet filtering represents a significantly higher processing load than routing decisions [15, 18, 22]. For example, a firewall that interconnects two 100 Mbps networks would have to process over 300,000 packets per second [21]. Successfully handling this high traffic becomes more difficult as rule sets become more complex [4, 14, 22]. Furthermore, firewalls must be capable of processing even more packets as interface speeds increase. In a high-speed environment (e.g. Gigabit Ethernet), a single firewall can easily become a bottleneck and is susceptible to DoS attacks [4, 7, 10, 12]. An attacker could simply inundate the firewall with traffic, delaying or preventing legitimate packets from being processed. Building a faster single firewall is possible [8, 15, 18, 19, 20]; however, the benefits are temporary (traffic loads and interface speeds are increasing); it is not scalable; it is a single point of failure; and it is generally not cost-effective for all installations.

A data-parallel firewall (or firewall sandwich) where each firewall node implements an entire firewall rule set is another approach to increase the speed of processing traffic [4, 12, 14, 22]. As shown in Figure 1, the system consists of multiple identical firewalls **100** connected in parallel. Each machine implements the complete security policy and arriving packets are distributed across the machines for processing in parallel [4]. A packet distributor **102**

implements a load balancing algorithm to distribute different packets to different firewall nodes. How the load-balancing algorithm distributes packets is vital to the system and is typically implemented as a high-speed switch in commercial products [11, 12]. Although data-parallel firewalls achieve higher throughput than traditional firewalls [4] and have a redundant design, they have difficulty maintaining QoS across networks. For example, legitimate traffic can encounter delays if it is queued behind traffic that requires more processing. Under these circumstances, users notice poor network performance, which is growing concern as more network applications require QoS assurances. Furthermore, stateful inspection requires all traffic from a certain connection or exchange to traverse the same parallel machine, which is difficult to perform at high speeds [14]. Therefore, new firewall architectures are needed to meet the demands of future networks and the challenges of increasing security threats.

SUMMARY

According to one aspect, the subject matter described herein includes a function-parallel firewall. The function-parallel firewall includes a first firewall node for filtering received packets using a portion of a rule set. The portion of the rule set may be less than all of the rules in the rule set. At least one second firewall node filters packets using at least one rule in the rule set that is not present in the portion implemented by the first firewall node. The first and second potions together include all of the rules in the rule set.

The term "firewall node," as used herein refers to a physical entity that implements one or more packet filtering rules. For example, a firewall node may be a single-processor computing device that implements the packet filtering rules. Alternatively, a firewall node may be a processor within a multi-processor computing device that implements one or more packet filtering rules. Thus, a multi-processor device may include multiple firewall nodes.

According to another aspect, the subject matter described herein includes a stateful function-parallel firewall system. The system includes a first function-parallel firewall subsystem including a plurality of firewall nodes implementing a first rule set for filtering packets arriving at a network. The first function-parallel firewall subsystem includes at least first and second firewall nodes respectively implementing first and second portions of the first rule set.

The first portion includes at least one rule that is not present in the second portion. A second function-parallel firewall subsystem includes a second set of firewall nodes. The second set of firewall nodes implements a firewall rule set for filtering packets departing from the network. The second function-parallel firewall subsystem includes at least third and fourth firewall nodes. The third and fourth firewall nodes implement at least one rule that is not present in the second portion of the second rule set. The function-parallel firewall subsystems share state information regarding connections established through the first and second sets of firewall nodes.

According to another aspect, the subject matter described herein includes a firewall grid. The firewall grid includes a plurality of firewall nodes that are physically connected to each other via a network for filtering packets. A controller controls logical connections between the firewall nodes. The controller is adapted to configure the firewall nodes in a logical firewall hierarchy without changing physical connections between the firewall nodes.

According to yet another aspect, the subject matter described herein includes a method for distributing rules in a function-parallel firewall. The method includes defining a rule set for a function-parallel firewall. The method further includes assigning rules in the rule set to nodes and branches in a trie data structure. Each node in the trie data structure corresponds to a data field and each branch represents a value for each data field to be compared to be values in corresponding fields of received packets. The trie data structure is pruned in a manner that preserves ordering of the rules in the rule set.

According to yet another aspect, the subject matter described herein includes a method for assigning rules to a plurality of firewall nodes. The method includes assigning rules to nodes in a directed acyclical graph (DAG). Relationships between rules are represented by edges in the DAG. Rules are distributed among a plurality of firewall nodes using the DAG in a function-parallel manner and such that relationships between the rules specified in the DAG are preserved.

As used herein, the term "firewall" refers to a logical entity that is adapted to filter packets at the ingress and/or egress portions of a network based on a policy. The term "firewall" is intended to include systems that block packets from entering or leaving a network and systems that perform intrusion

IPR2023-01353 Page 00720

detection and intrusion protection functions for packets entering or leaving a network. Thus, the term "function-parallel firewall" is also intended to include an intrusion detection system or an intrusion protection system that operates in a function-parallel manner as described herein.

The subject matter described herein can be implemented using any combination of hardware, software, or firmware. In one exemplary implementation, the subject matter described herein may be implemented using a computer program product comprising computer executable instructions embodied in a computer readable medium. Exemplary computer readable media suitable for implementing the subject matter described herein include disk memory devices, chip memory devices, programmable logic devices, application specific integrated circuits, and downloadable electrical signals. In addition, a computer program product that implements the subject matter described herein may be located on a single device or computing platform or may be distributed across multiple devices or computing platforms.

One objective of the subject matter described herein is to provide a new security architecture, referred to herein as a hierarchical firewall, that provides the functionality of a traditional firewall (enforcing a security policy) and includes the following improvements:

1. **Suitable for high speed networks and scalable for large traffic loads**. Firewalls must continue to perform security tasks quickly and efficiently, appearing transparent to legitimate users under varying traffic loads.

2. **Maintains QoS requirements and traffic differentiation**. A growing number of network applications require QoS for their operation. Firewalls must not violate these guarantees while performing their security tasks.

3. **Greatly lessens the impact of DoS attacks**. Firewalls are candidates for DoS attacks, since they serve as the point of entry between two networks. New architectures must be capable of handling these new threats.

4. **Highly available**. Firewalls introduce a single point of failure between two networks. New firewall architectures must provide redundancy that prevents segmenting the secure network from the external network.

According to one aspect, the subject matter described herein includes a set of hierarchically configured computers that collectively enforce a desired security policy. Given this distributed configuration, traffic is quickly divided among the machines based on perceived threat. In one exemplary implementation, traffic that is considered safe may be promptly forwarded into the secure network, while the remaining traffic may be forwarded to different machines in the hierarchy for further scrutiny. Hence, traffic is segregated and queued based on security threat, yielding minimal delays for legitimate traffic. The system is also robust and highly available, since multiple machines are employed for the architecture and low cost, since Linux-installed PCs and open source software can be used for its implementation [21]. The proposed architecture is believed to be applicable to a wide variety of agencies in the public and private sectors.

The firewall architecture described herein will provide a scalable solution to varying traffic loads and QoS requirements, which are increasingly important as computing becomes more ubiquitous and applications become more diverse. The architecture is of interest to the networking community, since the architecture will result in new security policy models and optimization methods. The architecture is also relevant to the military and business community, since it provides a new, robust and cost-effective solution to security threats.

Some of the objects of the subject matter described herein having been stated hereinabove, and which are addressed in whole or in part by the subject matter described herein, other objects will become evident as the description proceeds when taken in connection with the accompanying drawings as best described hereinbelow.

BRIEF DESCRIPTION OF THE DRAWINGS

Preferred embodiments of the subject matter described herein will now be explained with reference to the accompanying drawings of which:

Figure 1 is a block diagram of a conventional data-parallel firewall;

Figure 2 is a block diagram illustrating an exemplary function-parallel firewall architecture according to an embodiment of the subject matter described herein;

Figure 3A is graph comparing simulation results for average packet delay for different firewall architectures;

Figure 3B is a graph comparing simulation results for maximum packet delay for different firewall architectures;

Figure 4A is a graph illustrating average packet delay for different packet types for a data-parallel firewall;

Figure 4B is a graph illustrating simulation results for packet delay for different packet types for a function-parallel hierarchical firewall according to an embodiment of the subject matter described herein;

Figure 5 is a data structure diagram illustrating a full trie representation of the security policy illustrated in Table 1;

Figure 6 is a data structure diagram illustrating a pruned trie representation of the security policy given in Table 1 where each packet is processed within four field comparisons;

Figure 7A is a data structure diagram illustrating a trie representing two security rules assuming that the default action per node is deny;

Figure 7B is a data structure diagram illustrating a reordered version of the trie data structure illustrated in Figure 7A;

Figure 8A is a diagram illustrating a Jackson model for a function-parallel hierarchical firewall according to an embodiment of the subject matter described herein;

Figure 8B is a diagram illustrating a Jackson model for a data-parallel firewall;

Figure 9 is a graph illustrating simulation results for average delay for different firewall architectures;

Figure 10 is a block diagram illustrating a stateful hierarchical firewall according to an embodiment of the subject matter described herein;

Figure 11A is a block diagram illustrating physical connections of a firewall grid according to an embodiment of the subject matter described herein;

IPR2023-01353 Page 00723

Figure 11B is a block diagram illustrating logical connections of the firewall grid illustrated in Figure 11A according to an embodiment of the subject matter described herein;

Figure 12 is a policy directed acyclical graph (DAG) representation of the firewall rules given in Table 2;

Figure 13A is a block diagram illustrating rule distribution among a data-parallel firewall;

Figure 13B is a block diagram illustrating a function-parallel firewall and rule distribution among nodes in the function-parallel firewall according to an embodiment of the subject matter described herein;

Figure 14A is a block diagram illustrating one exemplary rule distribution among nodes in a function-parallel firewall according to an embodiment of the subject matter described herein;

Figure 14B illustrates another distribution of rules among nodes in a function-parallel firewall according to an embodiment of the subject matter described herein;

Figure 14C illustrates an illegal rule distribution among nodes in a function-parallel firewall;

Figure 15A is a block diagram illustrating non-redundant rule distribution among nodes in a function-parallel firewall according to an embodiment of the subject matter described herein;

Figure 15B is a block diagram illustrating redundant rule distribution among nodes in a function-parallel firewall according to an embodiment of the subject matter described herein;

Figure 16A illustrates a policy DAG where deny rules are indicated by a square node;

Figure 16B is a block diagram illustrating initial gateless rule distribution among nodes in a function-parallel firewall according to an embodiment of the subject matter described herein;

Figure 16C is a block diagram illustrating final gateless rule distribution among nodes in a function-parallel firewall according to an embodiment of the subject matter described herein;

Figure 17A is a graph illustrating simulation results for average packet delay as packet arrival rate increases for different firewall designs;

IPR2023-01353 Page 00724

Figure 17B is a graph illustrating simulation results for packet delay as the arrival rate and number of rules increase for different firewall designs;

Figure 18A is a graph illustrating simulation results for average packet delay for different firewall architectures;

Figure 18B is a graph illustrating simulation results for maximum packet delay for different firewall architectures; and

Figure 19 is a flow chart illustrating exemplary steps for filtering packets using a rule set distributed among nodes in a function-parallel manner according to an embodiment of the subject matter described herein.

DETAILED DESCRIPTION OF THE INVENTION

According to one aspect of the subject matter described herein may include multiple firewall machines configured in a hierarchy, as shown in Figure 2. In the example illustrated in Figure 2, a plurality of firewall nodes **200**, **202**, **204**, and **206** collectively implement a firewall policy. Each firewall node or set of nodes implements a different level of the firewall policy in a hierarchy. For example, firewall node **200** may implement a first level of a firewall policy that includes rules that are not present in subsequent levels. If a packet passes the first level of screening, the packet may be forwarded immediately to the internal network without impacting other firewall nodes. If the packet is identified as requiring further scrutiny, the packet may be forwarded to the next level where it is screened by firewall node **202** or **204** using rules that were not applied by firewall node **200**. If the packet passes level 2 screening, the packet may be forwarded directly to the internal network. If the packet is indicated as requiring further screening, the packet will be passed to level 3 where level 3 screening is applied by firewall node **206**.

At each level in the firewall hierarchy illustrated in Figure 2, firewall nodes may implement different portions of a packet filtering rule set. An architecture in which different firewall nodes implement different portions of a firewall rule set is referred to herein as a function-parallel architecture. A function-parallel firewall architecture in which packet filtering rules are distributed among firewall nodes such that if a packet passes or fails at a

certain level, the packet may bypass later levels is referred to herein as a hierarchical firewall architecture.

In the architecture illustrated in Figure 2, at each level, machines may be configured to perform a specific security task quickly and efficiently. In one implementation, the firewall rules may be distributed across the different hierarchy levels such that increased scrutiny is placed at lower levels. Since the rules are distributed among the levels, no machine in the hierarchy is required to implement the entire rule set. In contrast, a data-parallel firewall architecture requires all machines to implement the complete rule set.

As described above, in the exemplary architecture illustrated in Figure 2, packets start at the highest level and progress downward through the hierarchy. At each level, a packet can be forwarded to the internal network if deemed safe, forwarded to the next lower level for more processing, or dropped. The benefit of this divide-and-conquer approach is that traffic will incur delays consistent with its perceived security threat; therefore, traffic that is deemed safe will not be delayed by traffic that requires more processing. As a result, the hierarchical firewall can provide differentiated service and support QoS guarantees, unlike current designs. Furthermore, traffic that is forwarded to the internal network can be shaped at the particular firewall level, constraining the amount of traffic entering the network and lessening the effect of DoS attacks.

For example, in the hierarchical firewall depicted in Figure 2, packets are forwarded from the external network to the first level, where initial security checks are performed, such as a quick validity check and comparisons of IP addresses. Packets with trusted IP addresses are immediately forwarded to the internal network, while the remaining packets are sent to the next level. At the second level, packets are separated based on transport protocols and/or port numbers. Again, if a packet is deemed safe it is immediately forwarded to the secure network; otherwise, they are sent to the next level for auditing. Log files may be used at one or more of the firewall nodes indicate which connections are open and have been requested. Each level may also drop packets based on perceived threat.

The preceding example illustrates a simple hierarchical firewall; however, the size and complexity can vary. In general, the hierarchy attempts to quickly divide traffic and provide differentiated service. More levels can be used for

IPR2023-01353 Page 00726

stronger security policies and to distribute traffic more efficiently. Determining the appropriate number of levels, the number of machines per level, and how to distribute the rule set are implementation details that may be used to optimize a function-parallel firewall according to an embodiment of the subject matter described herein.

Preliminary Experimental Results

In this section, simulation is used to compare a hierarchical firewall with a traditional single-machine firewall and a data-parallel firewall [4, 22]. Each architecture implemented the same 325-rule security policy, where each rule required 1 unit of time to complete. The single firewall architecture consisted of one machine that processed each packet received from the external network. The data-parallel architecture consisted of four machines, as shown in Figure 1, each implementing the same 325-rule set. The hierarchical firewall consisted of four machines configured in three levels, as shown in Figure 2. Rules were distributed across the levels such that the first level implemented 25 rules; the second level implemented 50 rules each; and the third level implemented 75 rules. The group of rules performed at a certain machine will be referred to as a security task. The hierarchical firewall incurred an additional delay of one time unit when a packet was transmitted from machine to machine. Packets randomly arrived to the system using a Poisson distribution with a mean of 30 units and would randomly match one of the 325 rules based on a uniform distribution. For classification purposes, each packet was associated with the security task it matched (which group of rules). In this experiment, the performance parameters analyzed are the overall average packet delay, the maximum packet delay, and the average delay experienced by each packet type.

Experimental results for the different architectures are shown in Figures 3 and 4. More particularly, Figure 3A illustrates average packet delay for a single firewall design, for the data-parallel firewall design illustrated in Figure 1, and for the hierarchical firewall design illustrated in Figure 2. Figure 3B illustrates maximum packet delay for the single, data-parallel, and hierarchical firewall designs. Figure 4A illustrates average packet delay for the data-parallel firewall design for different packet types. Figure 4B illustrates average packet

delay for the hierarchical firewall design for different packet types. Packet types refer to packets of a classified based on which security task was satisfied. Unlike the data-parallel firewall, the hierarchical design consistently yields different delays for each packet type.

For both delay measurements, the hierarchical architecture performed better than the single firewall and the data-parallel firewall. The single firewall had an average delay 31 times greater than the hierarchical architecture, which demonstrates how a single firewall machine can quickly become a bottleneck. Although the data-parallel firewall performed better than the single firewall, it had an average and maximum delay 6 times longer than the hierarchical architecture. In the parallel configuration, traffic that needed little processing often queued behind traffic that required more processing, which resulted in longer delays. In contrast, the hierarchical system consistently yielded better results, since traffic was distributed among the levels based on processing requirements.

The average delay experienced by each packet type is also important and is illustrated in Figures 4A and 4B. Delays in the hierarchical firewall were proportional to the amount of processing required; for example, security task 1 traffic had the shortest delay, security task 2 traffic had the second shortest delay, security task 3 traffic had the third shortest delay; and security task 4 traffic had the highest delay. Even though security tasks 2 and 3 were on the same level, security task 3 traffic had a delay slightly higher that security task 2 traffic, since it was queued with security task 4 traffic. Therefore, the hierarchical firewall is capable of providing differentiated service to multiple packet types/classes. In contrast, the delays varied for each packet type in the data-parallel firewall. Packets were constantly delayed by lower type/class traffic (e.g., security task 1 packets delayed by security task 4 packets); thus, no differentiation was achieved. Although these results are preliminary, they indicate the hierarchical architecture has merit, offering significantly better performance than current firewall systems.

IPR2023-01353 Page 00728

Refinements

The previous section described an initial hierarchical firewall architecture for employing a security policy. Although preliminary results indicate the architecture has merit, refinements can be made so that the design can achieve its full potential. One refinement relates to the hierarchical architecture. Important questions include determining the appropriate number of levels and how to effectively distribute the security policy rules. Another refinement may include providing stateful packet inspection (maintaining connection tables) and increasing redundancy. Finally, another refinement may include examining other high-speed firewall architectures, such as a dynamic firewall grid. Analytical models and simulation will be used for design and analysis; in addition, a prototype will be implemented along with analysis tools and software.

Architecture Design and Rule Set Distribution

Important questions about the firewall architecture include the number of levels, the number of machines per level, and the number of security tasks per machine. The security policy is central for addressing these questions and, for this reason, will be formally defined and modeled. The model will be optimized to reduce the physical size, minimize the traversal time, and provide differentiated service. Afterward, the optimized policy model will be used to implement the actual hierarchical firewall.

One method for designing the hierarchy is to represent the security policy graphically as a tree data structure. This representation will aid the translation to the actual firewall, since it has a hierarchical form. Although a variety of tree data structures exist, a trie will be initially investigated [1]. A trie is a special type of data structure that has the advantages of fast retrievals and quick unsuccessful search determination, which are important for enforcing a security policy. In a security policy trie, each path from the root node to a leaf corresponds to one rule. The nodes of the trie correspond to the rule fields, while branches are possible values. Figure 5 is a trie data structure for the rules in Table 1. For example, in Figure 5, the root node is the protocol field. The next level represents the source IP address, while the leaf nodes represent

IPR2023-01353 Page 00729

the rule action. Rules that have the same field value at a node will share a branch for that value.

The trie in Figure 5 referred to as full trie, since every noncycle path from the root to a leaf has the same length, and every leaf lies at the same depth [6]. Unlike other policy models, the rule order must be preserved if the trie is to be successfully translated to a hierarchical firewall. The action to apply to a packet is determined by a preorder traversal. Starting at the root node, branches are compared to the fields of the packet. Since multiple branches may match a packet, they are traversed in order from left to right. If a packet field does not match any possible branches, it moves back to the previous node (parent), where the next branch is tested. Once the policy trie is developed, it can be translated into a hierarchical firewall. The nodes of the trie represent the machines in the hierarchy, while the branches represent the links between nodes. The value associated with each branch indicates the packet field value required to traverse it. Although a full trie representation of a security policy can be created, it is not necessarily optimal. Further trie manipulation is required create an effective hierarchical firewall, such as preventing routing to parents and limiting trie traversal time.

Pruning yields a trie that never requires routing to parent nodes, which provides faster packet processing [1, 15]. Assuming a full trie consists of n nodes and k levels (where k <= n), the worst case packet processing would be O(n*k) tuple-compares. In contrast, a pruned trie requires only O(k + n) tuple-compares in the worst case which is a processing reduction of 1/k.. Pruning is accomplished by replacing general rules with descendants. Rule r_1 is a descendant of rule r_2, if $r_2[i]$ is a prefix of $r_1[i]$ for $i = 1, \ldots, k$. Therefore, pruning occurs by pushing a rule down to all its descendents, then deleting the rule. An example of a pruned version of the trie illustrated in Figure 5 is given in Figure 6, where [*, *, *, deny] is a general rule that can be pushed down to every other rule. Although the original branch has been removed and added to the other branches, the order is preserved, if branches are tested from left to right. Current pruning algorithms do not consider rule order, since tries were primarily developed for storing data [6]. New pruning techniques that preserve order may be developed and used to distribute rules among nodes in a functional-parallel firewall according to an embodiment of the subject matter described

herein. Pruning may also cause the number of nodes in the hierarchy to increase, since a general rule is duplicated at different levels. For this reason, other techniques are needed to reduce the trie size.

Reducing the trie size is important, since delay increases each time a packet is forwarded from one node to another. The trie can be reduced by compressing nodes, which is achieved by removing single branching paths or consolidating sub-ties into one node [2, 6, 15]. As a result, a node may compare multiple rule fields or perform multiple security tasks per packet. For example, in Figure 5, the rule [UDP, 150.*, *, 3030, accept] is considered a leaf chain [6] and can be compressed into one node (a single machine). Two other methods commonly used to reduce the size of a trie are Directed Acyclic Graphs (DAG) and level reordering [1, 6, 15]. If the same sub-trie appears in multiple places, then the trie structure can be changed to a DAG. The possible use of DAG is seen in Figure 6, where nodes labeled $Port^a$ and $Port^b$ are equivalent sub-tries. Parent nodes of both sub-tries could point to just one, eliminating the need for the other.

Another method for reducing the trie size is level reordering. As seen in Figures 7A and 7B, the order of the levels can significantly affect the number of nodes in the trie. More particularly, in Figure 7A, an original trie representing the security rule [TCP, 120.*, 210.*, 90, accept] and [TCP, 130.*, 210.*, 90, accept] is illustrated. Both Figures 7A and 7B assume that the default action per node is deny. Since some of the rules in the original trie in Figure 7A include common elements, the rules can be reordered so that the number of branches in the subtrie are reduced. For example, in Figure 7A, the destination IP address and port for each branch of the subtrie are the same. Accordingly, if these common portions of the rule are implemented at higher levels, the duplicate nodes in the trie can be reduced, as illustrated by Figure 7B. More particularly, in Figure 7B, the nodes for destination IP and port are placed before the node for source IP address. As can be seen from Figure 7B, reordering the levels produces a smaller trie. Using a DAG can also reduce the number of nodes.

The problem of finding an ordering that guarantees a minimum size trie has been shown to be NP-complete; however, several heuristics have been developed [2, 6]. It should be noted that DAGs and reordering only reduce the

number of nodes in the hierarchy and do not reduce the depth of the trie, which influences the traversal time.

Regardless of the applicability of these different trie modifications, they must be performed in concert to achieve an optimal hierarchical design. For example, the architecture utilizes multiple nodes to distribute traffic load and provide differentiated service; however, the trie path length (depth) impacts the packet traversal time. Therefore, compression and ordering must be performed together to provide differentiated service and optimize packet delay.

To achieve these goals, information not explicitly given in the security policy (e.g., QoS, level of trust, and processing time) may be used. For example, it is advantageous to place security tasks that require the most time at the lowest levels. This prevents packets from queueing behind traffic that requires more processing time; thus, lowering delay. A weight w_i can be assigned to each security task s_i (a single rule or multiple rules) based on the amount of time it requires. For example, an auditing security task requires more time (updating system files) than those that only perform field comparisons. Using these weights, the trie can be ordered to minimize the average traversal time. For example assuming i and j are hierarchy levels, where $i < j$, the trie can be ordered such that the average weight per level increases from the highest to the lowest level $\overline{w}_i < \overline{w}_j, \forall i \neq j$. In addition, the trie can be designed to separate traffic based on QoS or level of trust. Traffic would traverse different sub-tries, based on the type of service required, providing class isolation and different delays. The algorithms described in this section can then be applied to these sub-tries independently to reduce size and traversal time. These ideas were used to design the hierarchical firewall simulated in the preceding section, which showed excellent results. New algorithms for trie compression based on traversal time with constraints on QoS will be developed for optimizing function-parallel firewall design.

Packet traversal time remains a key performance metric for firewall systems. For this reason, queueing models will be used to analyze the delay of different firewall designs. The node delay encountered by a packet consists of the processing time, queueing delay, and transmission time. The processing delay is the time required to perform the security task. Assume node i must

process R_i rules per packet and can perform *xi* rules per unit time, then the service rate is x_i/R_i packets per unit time. Queueing is the delay associated with waiting for processing and depends on the system state. The transmission delay is the time required to transmit the packet, which depends on the packet's length and the line speed. The maximum delay experienced by a packet traversing the firewall is the sum of the node delays along the path from the root to a leaf.

If it is assumed the packet arrival and processing (transmission and security task) rates are exponentially distributed, the node can be modeled as an M/M/1 queue [5, 17]. The firewall can then be considered an open network of M/M/1 queues, where probabilities are assigned to each link to indicate the likelihood of moving to the next node. This is a Jackson network model, which has a product-form solution for the average delay [5, 17]. The average end-to-end delay for N cascading nodes (traversal path) can be computed as

$$E(T) = \sum_{i=1}^{N} \frac{1}{\mu_i - \lambda_i} \tag{1}$$

where $1/\mu_i$ is the service time (processing and transmission) and λ_i is the arrival rate to node *i*. Figures 8A and 8B respectively illustrate Jackson models for hierarchical and data-parallel firewalls. In Figures 8A and 8B, it is assumed that the total arrival rate to each system is λ packets per unit time, each system implements the same R rule security policy, and each node can perform x rules per unit time. For the hierarchical firewall, shown in Figure 8A, the policy rules are evenly distributed to each node; therefore, the service rate is $3x/R$ per node. The traffic arrival rate to node A is λ packets per unit time, half of the packets are forwarded to node B, while the other half is forwarded to node C. Although not considered in this example, some traffic may be forwarded into the secure network or dropped by node A; therefore in this case, the sum of the rates to nodes B and C would be less than λ. Using these values, the average end-to-end delay for either path in the hierarchical firewall is

$$E_h(T) = \frac{1}{\frac{3\chi}{R} - \lambda} + \frac{1}{\frac{3\chi}{R} - \frac{\lambda}{2}} \tag{2}$$

For the data-parallel configuration, illustrated in Figure 8B, traffic arrives at the packet distributor, which evenly distributes the traffic to each node. Each node implements the complete rule set *R*; therefore, the service rate is *x/R*. The end-to-end delay across any node in the data-parallel firewall is

$$E_p(T) = \frac{1}{\frac{\chi}{R} - \frac{\lambda}{3}} \quad (3)$$

Using these models, the effect of increasing traffic load λ on the average end-to-end delay, for the two architectures is shown in Figure 9. For this example, the arrival rate λ ranged from 1 to 50 packets per unit time. The rule set *R* consisted of 240 rules, while *x* equaled 5000 rules per unit time. Each of the hierarchical nodes incurred an additional delay of five time units to compensate for inter-node transmission. As expected, the average delay increases as the network load increases for both architectures; however, the average delay is consistently lower for the hierarchical firewall. This is due to the decreased rule set per node and the reduced arrival rates for nodes B and C. The average is an upper bound for the hierarchical firewall. Allowing node A to forward packets to the secure network or to drop packets would reduce the load on the second level and decrease delays. The arrival rate is less per node in the data-parallel firewall; however, the system has a higher end-to-end delay because each node must implement the complete rule set. The theoretical results are consistent with the experimental results described above. These models provide an important tool for analyzing the effect of load and rule distribution on end-to-end delay. In addition to analytical results, simulation will be used to measure the performance of the hierarchical designs under various conditions.

Stateful Firewalls

Another aspect of the subject matter described herein includes a stateful hierarchical firewall. The preceding design is considered stateless, or static, since the security action taken is based solely on the packet currently inspected. A stateful firewall provides better security by tracking and controlling network flows (packets belonging to a particular network connection or

IPR2023-01353 Page 00734

exchange) [16, 21, 22]. In one exemplary implementation, a stateful firewall can be achieved by storing state information concerning each flow in a table, and then according it with the security policy. For example, each time a TCP connection is established from the internal network to the external network, it is logged in a table. The table may contain the source and destination addresses, port numbers, sequencing information, and connection flags. Afterwards, packets arriving from the external network claiming to be part of an existing TCP connection can be checked against the table. If a packet belongs to an existing connection or exchange, it can be accepted. Otherwise, it is dropped. Table entries expire, if packets belonging to an exchange are not observed by the firewall over a period of time [21].

Therefore, a stateful firewall is dynamic and limits the hackers' opportunity to exploit certain services. The increased security does require additional memory and time for table manipulation, which is problematic for high-speed networks. Furthermore, the firewall must process all traffic arriving and departing the secure network to adequately maintain the connection tables 21]. To solve these problems, the proposed firewall system will be divided into two hierarchies, one for arriving traffic and another for departing traffic, as illustrated in Figure 10. In Figure 10, a stateful firewall system includes first and second hierarchical firewall subsystems **1000** and **1002** and a control node **1004**. First and second hierarchical firewall subsystems **1000** and **1002** may be function-parallel firewall systems including a plurality of firewall nodes with different rules distributed among different nodes as described above. In one exemplary implementation, hierarchical firewall subsystem **1000** may include at least two firewall nodes that implement different portions of a first firewall rule set. Similarly, hierarchical firewall subsystem **1002** may include at least two firewall nodes that implement different portions of a second firewall rule set. Firewall subsystem **1000** may filter packets entering the network. Firewall subsystem **1002** may filter packets departing from the network. Firewall subsystems **1000** and **1002** may share state information regarding connections established through the nodes within each system. Control node **1004** may store the state information and may be accessible by the nodes in hierarchical firewall subsystems **1000** and **1002**. Rules can be distributed to the

hierarchies and the designs can be optimized using the methods described in preceding section.

However, a new problem with the stateful model is the need to distribute state information across nodes. For example, nodes that scrutinize arriving traffic must be aware of any new connections permitted by nodes that inspect departing traffic. Furthermore, nodes must be informed of table entries that expire over time. To communicate state information, a central database machine, such as control node **1004**, may collect and distribute the information (table entries) to the appropriate nodes. In one exemplary implementation, control node **1004** may not perform table queries on behalf of nodes. Individual nodes will only query their local tables, which prevents the central machine from becoming a bottleneck. In addition, the central machine will only send information (table entries) to nodes that might require it. As a result of this policy, the table searched by an individual node is reduced (contains only relevant information) as well as the search time. In an alternate implementation, control node **1004** may store the connection table and may perform queries for packets received by firewall subsystems **1000** and **1002**.

Survivability and Firewall Grids

Another aspect of the subject matter described herein focuses on increasing the survivability of the hierarchical design. Although the proposed design has the benefit of increased packet throughput and minimal delay, the reliability of the system is decreased, since more machines (nodes) are deployed. For example, if the highest level node fails (is unable to inspect more traffic or goes off-line), then the entire hierarchy is unable to process packets. For this reason, the hierarchy is considered an n of n system, where all *n* machines must be functioning for the system to operate correctly [4, 12].

Survivability of the system can be improved with the use of redundant (parallel) nodes at different levels. For example, the first level may include multiple machines in parallel, where each machine implements the same security task (e.g., an IP check). An arriving packet would be sent to only one of the parallel machines, where it would be accepted, dropped, or forwarded to the next level for further processing. If one of the parallel nodes fails, then the

IPR2023-01353 Page 00736

remaining nodes can process the traffic. An additional benefit of parallel processing is the equal distribution of traffic among nodes, which can result in higher throughput and lower delays. Details such as where to locate parallel nodes and the number of machines required for each parallel configuration may depend on the desired degree of reliability versus cost of deploying redundant machines.

Yet another aspect of the subject matter described herein includes a grid of configurable firewalls (nodes) that can be used to efficiently process traffic, while being highly available. In one exemplary implementation, the grid may include a network of highly interconnected nodes (possibly via a high-speed switch or router) that can be logically deployed in various configurations (e.g., parallel, hierarchical, or hybrid). The advantage of the firewall grid is its ability to quickly adapt to new demands. When security needs change (e.g., due to policy changes, increased traffic volume, or node failure), these machines can be deployed or configured to maintain throughput and QoS requirements. For example, in a logical hierarchy, if a higher level encounters an increased traffic load, less frequently used redundant machines at lower levels can be reconfigured to process packets at the higher level. Unlike the hierarchical model, physical changes are not required, since all the machines are interconnected. Only the logical connection between nodes must be updated, which can be performed quickly. Like the stateful hierarchical firewall, the initial grid model will use a central machine to coordinate node configuration and communication. This central machine will play an important role in the system (balancing processing demand per node) and may constrain the efficiency of the firewall grid. Future versions of the firewall grid will attempt to place more intelligence in the nodes and allow them to interact independently and adapt dynamically under the guidance of a general security policy.

Figures 11A and 11B illustrate an example of a firewall grid according to an embodiment of the subject matter described herein. Referring to Figure 11A, a plurality of firewall nodes **1100-1114** are connected via a control node **1116**. Each firewall node **1100-1114** may be configured to implement different portions of a firewall rule set, as controlled by control node **1116**. Each of the firewall nodes is physically connected to the other firewall nodes via physical

connections **1118**. Physical connections **1118** may be local area network connections, such as Ethernet connections. As such, control node **1116** may be an Ethernet switch.

In Figure 11B, firewall nodes **1100**-**1114** are connected via logical connections **1120**. Logical connections **1120** may be implemented by control node **1116** without changing physical connections **1118**. For example, control node **1116** may configure the output port to which firewall **1106** is connected to receive frames from firewalls **1100**, **1102**, and **1104**. Similar internal configurations can be implemented for the remaining firewalls illustrated in Figure 11B. In Figure 11B, logical connections **1120** between the firewalls implement a hierarchical, function-parallel firewall design, such as that illustrated in Figure 1. In addition, the design illustrated in Figure 11B is also data-parallel at each level. For example, each firewall node at each level may implement a common rule or rule set. For example, the rule set implemented by the firewall nodes in level 1 is illustrated in Figure 11B as R1. The firewall nodes in level 2 implement a rule set represented by R2, and the firewall nodes in level 3 implement a rule set represented by R3. If any of the firewall nodes in a given level fails, the remaining firewall nodes in that level may take over operation for the failed firewall. Alternatively, as described above, if a failure or a bottleneck occurs at one level, a less-used firewall at another level may be logically configured to operate at the level where the failure or congestion occurred. Packet distributors **1122** may be configured to distribute the firewall rules between the firewalls at each level. Thus, as illustrated by Figures 11A and 11B, logical connections between firewall nodes and a firewall grid may be dynamically altered (i.e., changed on the fly without ceasing packet filtering operation) by a controller node without changing the physical connections between the firewalls.

Firewall Implementation

In addition to the analytical models and simulations described above, an implementation of a hierarchical firewall to measure performance will now be described. One exemplary implementation may utilize Linux-based PCs and Ethernet connections. Each PC will contain multiple Ethernet cards and be

loaded with the Linux 2.4 kernel. The 2.4 kernel has support for various packet queueing disciplines, traffic control methods, and packet processing (e.g., iproute2, tc, and iptable). Collectively, these functions are required for advanced packet filtering. For example, iproute2 and iptable can route packets based on information other than the destination address, while tc can mark and shape traffic. PCs will serve as hierarchical nodes, sources and receivers of network traffic, and hierarchy controllers.

Results from an initial implementation of a three-node hierarchical firewall indicate that these open-source components are suitable for high-speed networks. The Computer Science Department at Wake Forest University has 10 PCs available for this project, where each has a 600 Mhz processor and multiple 100 Mbps Ethernet cards. These machines will serve as an initial testbed for system development; however, faster machines and high-speed network connections are required for actual system testing. For this reason, this project requests the purchase of 6 additional PCs (2.4 Ghz processors), each with multiple Gigabit Ethernet cards (3 cards per machine). These new machines will be configured as the firewall, while the current machines will serve as traffic generators and receivers. For comparison purposes, parallel and single-machine firewalls will also be implemented using this equipment. Any software required will be developed and provided by the Computer Science Department.

Firewall Rule Distribution for Function-Parallel Firewall Design

As stated above, one aspect with creating an optimal function-parallel firewall design includes distributing rules among firewalls in the firewall rule set. Firewall rules are preferably distributed among firewall nodes in a manner that preserves firewall policy integrity. The following sections describe an exemplary firewall rule set and different options for distributing the firewall rule set among firewall nodes in a function-parallel firewall design.

Firewall Policy

In this section and the following sections, a rule r is modeled as an ordered tuple of sets, $r = (r[1], r[2], ..., r[k])$. Order is necessary among the tuples since comparing rules and packets requires the comparison of

corresponding tuples. Each tuple $r[l]$ is a set that can be fully specified, specify a range, or contain wildcards '*' in standard prefix format. For the Internet, security rules are commonly represented as a 5-tuple consisting of: protocol type, IP source address, source port number, IP destination address, and destination port number [21, 22]. For example, the prefix 192.* would represent any IP address that has 192 as the first dotted-decimal number. Given this model, the ordered tuples can be supersets and subsets of each other, which forms the basis of precedence relationships. In addition to the prefixes, each filter rule has an action, which is to accept or deny. However, the action will not be considered when comparing packets and rules. Similar to a rule, a packet (IP datagram) d can be viewed as an ordered k-tuple $d = (d[1], d[2], ..., d[k])$; however, ranges and wildcards are not possible for any packet tuple.

Using the previous rule definition, a standard security policy can be modeled as an ordered set (list) of n rules, denoted as $R = \{r_1, r_2, ..., r_n\}$. A packet d is sequentially compared against each rule r_i starting with the first, until a match is found ($d \Rightarrow r_i$) then the associated action is performed. A match is found between a packet and rule when every tuple of the packet is a subset of the corresponding tuple in the rule.

Definition Packet d matches r_i if

$$d \Rightarrow r_i \text{ iff } d[l] \subseteq r_i[l],\ l = 1, ..., k$$

The rule list R is comprehensive if for every possible legal packet d a match is found using R. Furthermore, two rule lists R and R' are equivalent if for every possible legal packet d the same action is performed by the two rule lists. If R and R' are different (e.g. a reorder) and the lists are equivalent, then the policy integrity is maintained.

As described above, a rule list has an implied precedence relationship where certain rules must appear before others if the integrity of the policy is to be maintained. Table 2 shown below illustrates an exemplary rule set that will be used herein to illustrate precedence relationships and rule assignment according to an embodiment of the subject matter described herein.

No.	Proto.	Source		Destination		Action	Prob.
		IP	Port	IP	Port		
1	UDP	1.1.*	*	*	80	deny	0.01
2	TCP	1.*	*	1.*	90	accept	0.02
3	TCP	2.*	*	2.*	20	accept	0.25
4	UDP	1.*	*	*	*	accept	0.22
5	*	*	*	*	*	deny	0.50

Table 2: Example Security Policy Consisting of Multiple Ordered Rules

In Table 2, rule r_1 must appear before rule r_4, likewise rule r_5 must be the last rule in the policy. If for example, rule r_4 was moved to the beginning of the policy, then it will shadow [23] the original rule r_1. Shadowing is an anomaly that occurs when a rule r_j matches a preceding rule r_i, where $i < j$. As a result of the relative order r_j will never be utilized. However, there is no precedence relationship between rules r_1, r_2, or r_3 given in Table 2. Therefore, the relative order of these three rules will not impact the policy integrity and can be changed to improve firewall performance.

As described in [28], the precedence relationship between rules in a policy can be modeled as a Policy Directed Acyclical Graph (DAG). Let $G = (R,E)$ be a *policy DAG* for a rule list R, where vertices are rules and edges E are the precedence relationships (constraint). A precedence relationship, or edge, exists between rules r_i and r_j, if $i < j$ and the rules intersect [28].

Definition The intersection of rule r_i and r_j, denoted as $r_i \cap r_j$ is

$$r_i \cap r_j = (r_i[l] \cap r_j[l]),\ l = 1, \ldots, k$$

Therefore, the intersection of two rules results in an ordered set of tuples that collectively describes the packets that match both rules. The rules r_i and r_j intersect if every tuple of the resulting operation is non-empty. In contrast, the rules r_i and r_j do not intersect, denoted as $r_i \not\cap r_j$, if at least one tuple is the empty set. It should be noted that the intersection operation is symmetric; therefore, if r_i intersects r_j, then r_j will intersect r_i. The same is true for rules that do not intersect.

For example, for the rules given in Table 2, the intersection of r_1 and r_4 yields (UDP, 1.1.*, *, *, 80). Again, the rule actions are not considered in the

intersection or match operation. Since these two rules intersect, a packet can match both rules for example d = (UDP, 1.1.1.1, 80, 2.2.2.2, 80). The relative order must be maintained between these two rules and an edge drawn from r_1 to r_4 must be present in the DAG, as seen in Figure 12. In Figure 12, the policy DAG represents the rules in Table 2. Vertices **1200** indicate rules while edges **1202** indicate precedence relationships. Rules r_2 and r_3 in Table 2 provide an example of rules that can be re-ordered without changing the firewall policy. These two rules do not intersect due to the fifth tuple (destination port). A packet cannot match both rules indicating the relative order can change; therefore, an edge will not exist between them.

As described above, traditional firewall implementations consist of a single, dedicated machine, similar to a router, that applies the rule set to each arriving packet. However, packet filtering represents a significantly higher processing load than routing decisions [15, 18, 22]. For example, a firewall that interconnects two 100 Mbps networks would have to process over 300,000 packets per second [21]. Successfully handling these traffic loads becomes more difficult as rule sets become more complex [4, 14, 22]. Furthermore, firewalls must be capable of processing even more packets as interface speeds increase. In a high-speed environment (e.g. Gigabit Ethernet), a single firewall can easily become a bottleneck and is susceptible to DoS attacks [4, 7, 10, 11]. Building a faster single firewall is possible [8, 15, 18, 19, 20]; however, the benefits are temporary (traffic loads and interface speeds are increasing); it is not scalable; it is a single point of failure; and it is generally not cost-effective for all installations.

Parallelization has been demonstrated to offer a scalable approach for improving the performance of network firewalls. As seen in Figure 13A, parallel designs consist of multiple firewalls, referred to as firewall nodes, connected in an array. In Figure 13A, firewall nodes **1300** each implement a common rule set. A packet distributor **1302** load shares packets among firewall nodes **1300**. In Figure 13B, a function-parallel design is illustrated where firewall nodes **1304** implement different portions of a firewall rule set. A packet duplicator **1306** duplicates each packet to each firewall node. A gate **1308** controls packets entering the network based on control information received from firewall nodes

1304. The operation of the system illustrated in Figure 13B will be described in more detail below.

The two designs depicted in Figures 13A and 13B are different based on what is distributed: packets or rules. For example, the design developed by Benecke et al consisted of multiple identical firewall nodes connected in parallel, as shown in Figure 13A. Each firewall node implements the complete security policy *R*, and arriving packets are distributed across the firewall nodes for processing (one packet is sent to one firewall node) [4]; therefore different packets are processed in parallel. Using terminology developed for parallel computing, this design is considered data-parallel since the data (packets) is distributed across the firewall nodes [25]. How the load-balancing algorithm distributes the packets is vital to the data-parallel system and typically implemented as a specialized high-speed switch [11, 12]. Although data-parallel firewalls have been shown to achieve higher throughput than traditional firewalls and have a redundant design [4], they suffer from two major disadvantages. First, these designs have difficulty maintaining QoS across networks. For example, legitimate traffic can encounter delays if it is queued behind traffic that requires more processing. Under these circumstances, users notice poor network performance, which is a growing concern as more network applications require QoS assurances. Second, stateful inspection requires all traffic from a certain connection or exchange to traverse the same firewall node, which is difficult to perform at high speeds using the data-parallel approach [4, 14].

Rule Distribution and Function-Parallel Firewall

As described above, new scalable firewall designs are needed to cope with increasing traffic loads, network speeds, and security threats. To address these issues, one implementation of the subject matter described herein includes a firewall design that consists of multiple firewalls connected in parallel and a gate machine, as shown in Figure 13B. Again in this design, the individual firewalls will be referred to as firewall nodes. However unlike the data-parallel design that distributes the packets, the function-parallel design distributes the policy rules across the firewall nodes. The distribution is done such that policy integrity is maintained. Therefore, a single traditional firewall

and a function-parallel firewall will always give the same result for any packet. The rules distributed to a firewall node will be referred to as the local rule set or local policy. Using parallel computing terminology this design will be referred to as function-parallel, since the rules are distributed across the firewall nodes [25].

The general operation of one exemplary function-parallel system can be described as follows. When a packet arrives to the function-parallel system, the packet is forwarded to every firewall node. Each firewall node processes the packet using its local rule list. The firewall node then signals the gate indicating either no match was found or provides the rule number and action if a match was found. Therefore, no match is a valid response and is required for this basic function-parallel design. The gate keeps track of the results and determines the final action to perform on the packet. For example, a first match policy can be implemented by applying the action of the lowest numbered matching rule to the packet.

The performance of this basic design can be improved by allowing the firewall nodes to pipeline packets [25]. For example, once a firewall node has completed processing a packet, it can immediately begin processing the next packet. This allows the firewall nodes to work independently, thus increasing the overall system performance. However, pipelining requires the gate to keep track of multiple packets simultaneously. Performance can also be increased if the gate can signal the firewall nodes that further processing of a certain packet is no longer needed (similar to short-circuit evaluation). This occurs when the appropriate match has been found by a firewall node before the other firewall nodes have completed processing the same packet. Short-circuit evaluation requires the gate to know how the rules are distributed as well as the dependencies, which is discussed in detail below.

The function-parallel design has several significant advantages over traditional and data-parallel firewalls. First, the function-parallel design results in the faster processing since every firewall node is utilized to process a single packet. Reducing the processing time, instead of the arrival rate (the data-parallel paradigm), yields better performance that will be proven theoretically and demonstrated experimentally. Second, unlike the data-parallel design, function-parallel can maintain state information about existing connections.

Maintaining state can be viewed as the addition of a new rule corresponding to a requested connection. Unlike the data-parallel design, the new rule can be placed in any firewall node since a packet will be processed by every firewall node. Third, the function-parallel design has the unique ability to provide service differentiation between classes of traffic; therefore, the design can provide QoS requirements. Finally, the function-parallel system can be implemented using any current firewall system (as discussed below, the gate can be removed), without any specialized hardware. The disadvantage of the system is the limited scalability, since the system cannot have more firewall nodes than rules, unless data-redundancy is provided. However, given the size of most firewall policies range in the thousands to hundred of thousands of rules, which is expected to increase, the scalability limit is not an important concern.

The operation and performance of the function-parallel design depends on the correct distribution of policy rules. Rule distribution techniques that maintain integrity, increase processing speed, allow redundancy and robustness, and eliminate the gate are described in the following sections.

Rule Distribution and Policy Integrity

As described above, the function-parallel design distributes the security policy rules across an array of firewall nodes. Using this design, all the firewall nodes process an arriving packet in parallel using its local rule set. The rules must be distributed such that the integrity of the policy is maintained. Again, it will be assumed in the examples described herein that a first-match policy is desired. Therefore, the distribution of the rules must not introduce any anomalies into the original rule list, such as shadowing.

Let m be the number of firewall nodes in the system and j be a firewall node $j = 1...m$. Each firewall node has a local rule set that is a portion of the security policy. Denote the local rule list for firewall node j as R_j that has n_j rules. The firewall node will process an arriving packet using the local rule list in a top-down fashion to find the first match. It should be noted that a software implementation of the firewall node according to the subject matter described herein is not required to have a rule list. The function-parallel design is independent of software implementation. In this scenario integrity is maintained

under two conditions: every rule is assigned to at least one firewall node, and a policy DAG edge never traverses upward within a firewall node. The first condition assures every rule in the original policy exists in the function-parallel system, while the second condition prevents rule shadowing within a firewall node. Consider a two-firewall node system and the five-rule policy given in Table 2. Possible rule distributions are depicted in Figures 14A-14C. The distribution illustrated in Figure 14A will be referred to as vertical, while the distribution illustrated in Figure 14B will be referred to as horizontal. In both cases the five rules are present and all policy DAG edges are downward; thus integrity is maintained. In contrast, Figure 14C is an illegal distribution because rule r_4 shadows r_1 (upward policy DAG edge).

The gate can also utilize the policy DAG to determine the first match, and if a short-circuit evaluation can occur. For example consider the distribution illustrated in Figure 14A. Assume a packet matches r_3 in the second firewall node and this result is reported to the gate. Using the policy DAG, the gate can determine this is the first match, since it is impossible for a packet that matches r_3 to also match any rules that preceded it in the original policy (rules r_1 or r_2). The gate can inform the remaining firewall nodes to stop processing the packet.

Optimal Rule Distribution

The previous section described the distribution requirements for maintaining policy integrity. However it was proven in [27] that any linear arrangement of a policy DAG maintains integrity, which also holds for a local policy DAG. Furthermore as evident in the previous section, many different distributions are possible. Given the possible rule distributions for the function-parallel design, a distribution is sought that minimizes the average number of comparisons required per packet [27].

Achieving this goal will require information not present in the firewall rule list. Certain firewall rules have a higher probability of matching a packet than others. As a result, it is possible to develop a policy profile over time that indicates frequency of rule matches (similar to cache hit ratio). Let $P = \{p_1, p_2, \ldots, p_n\}$ be the policy profile, where p_i be the probability that a packet will match rule r_i (first match in the policy). Furthermore, assume a packet will always find

a match, $\sum_{i=1}^{n} p_i = 1$; therefore R is comprehensive. Using this information, the average number of rule comparisons required for the policy R using a standard single machine firewall is

$$E(R) = \sum_{i=1}^{n} i \cdot p_i \quad (4)$$

This equation also applies to firewall nodes. However, the rules of a local policy include rules of the original policy; therefore, a mapping between the local and original policy rules is needed. Let $r_{i,j}$ represent the ith rule of firewall node j, that corresponds a certain rule in the original rule list R. Consider the local rule list for the second firewall node depicted in Figure 14A. The local policy is $R_2 = \{r_4, r_5\}$, where $r_{1,2} = r_4$ and $r_{2,2} = r_5$. Similarly, let $p_{i,j}$ be the probability of matching the ith rule of firewall node j. Therefore the average number of comparisons required for firewall node j is

$$E(R_j) = \sum_{i=1}^{n_j} i \cdot p_{i,j} \quad (5)$$

Using the previous equation, the average number of comparisons required for the function-parallel system is

$$E_f(R) = \sum_{j=1}^{m} E(R_j) = \sum_{j=1}^{m} \sum_{i=1}^{n_j} i \cdot p_{i,j} \quad (6)$$

For example, the average number of comparisons required for the distribution given in Figure 14A is 2.24, while the average number of comparisons for the distribution given in Figure 14B is 2.45.

Given a policy DAG $G = (R,E)$ and policy profile P, a distribution is sought that minimizes Equation 6. In the absence of precedence relationships, the average number of comparisons is minimized if the rules are sorted in non-increasing order according to the probabilities [28], which is also referred to as Smith's algorithm [29]. The sorted rules are then distributed in a horizontal fashion, as depicted in Figure 14B. Precedence constraints cause the problem

to be more realistic; however, precedence constraints also make determining the optimal permutation more problematic. In [27] determining the optimal rule order for a single machine firewall is proven to be *NP*-complete; therefore, so is determining the optimal function-parallel rule distribution. However, [27] presents a simple sorting algorithm to improve the optimal order, that can be modified to apply to function-parallel rule distribution. This modified algorithm sorts the policy rules using the method described in [27], then distributes the rules in a horizontal fashion.

Redundancy

Similar to the traditional and data-parallel firewalls, a disadvantage of the current function-parallel design is its inability to withstand a single firewall node failure. If a single firewall node fails in the function-parallel system, then part of the policy is lost and integrity is not maintained. A simple solution commonly used for traditional firewalls is to duplicate the entire system; however, this solution is cost prohibitive, not efficient, and difficult to manage.

Redundancy can be provided in the function-parallel design by replicating rules, instead of firewall nodes. Consider the function-parallel system depicted in Figure 15A, that includes four firewall nodes and an eight-rule policy distributed in a horizontal manner. If the local rule list of the right neighbor is added to the end of every firewall node, as illustrated in Figure 15B, then any non-continuous firewall nodes can fail and integrity will be maintained. It should be noted that the criteria for maintaining integrity still apply: every rule must be implemented at a node and no upward edges in the local policy DAG. Furthermore, the redundant design requires short-circuit evaluations to prevent multiple firewall nodes from evaluating the same rule.

Gateless Design

The gate plays an important role in the function-parallel design. It accepts information from the firewall nodes to determine the correct action to apply to a packet. Unfortunately, the gate is a special device that must coordinate with the individual firewall nodes in the system, which may be problematic to implement. However, it is possible to eliminate the gate and allow the firewall nodes to operate independently without any coordination. The

significant advantage is that the gateless function-parallel design can be implemented with any firewall system (from open source to proprietary).

Eliminating the gate requires a rule distribution such that for any given packet, only one machine will ever accept and the remaining machines will always deny. As described above, rule actions are either accept or deny. This can be seen in the policy DAG by denoting deny rules with a square and accept rules with a circle. To assure only one machine will accept a packet, an accept rule in a firewall node cannot intersect with another accept rule in any other firewall node. As described above, the intersection of two rules describes the set of packets that can match both rules. If an edge spans the local rules of firewall nodes, then the set of packets would be accepted by both firewall nodes; thus, duplicate packets would be accepted. In addition, every drop rule must be present in each firewall node to assure the remaining firewall nodes will drop the packet. Given these two requirements a policy DAG edge will not span firewall nodes, as illustrated in Figures 16B and 16C.

For example, consider a two-firewall node function-parallel system that implements the five-rule policy given in Table 2. The policy DAG is given in Figure 16A, where the first and last rules are deny. An initial gateless rule distribution is given in Figure 16B which has no edges spanning the firewall nodes. Given the local policy DAG, deny rules can be removed if: no inbound edges exist for the deny rule; and all out-going edges for the deny rule connect to a superset deny rule. Using these constraints, the first rule in the second firewall node depicted in Figure 16B can be removed (it is a subset of r_5 and has no other edges). This gives the final gateless rule distribution shown in Figure 16C. As a result, the first firewall node in this distribution will only process UDP traffic, while the second machine only processes TCP. The rules can be divided with respect to other tuples, and can be done to provide QoS guarantees.

Theoretical Results

Assuming arrivals and service times are exponentially distributed, a firewall system can be considered an open network of M/M/1 queues (Jackson network) [5, 17]. Probabilities can be assigned to each link to indicate the likelihood of moving to the next node, which are given from the policy profile (hit

ratio) described above. The average end-to-end delay for q cascading machines (traversal path) can be computed as

$$E(T) = \sum_{i=1}^{q} \frac{1}{\mu_i - \lambda_i} \tag{7}$$

where $1/\mu_i$ is the service time (processing and transmission) and λi is the arrival rate to node i. As a result, we have a theoretical model for the average delay across a firewall system. Consider the data-parallel and functional-parallel firewall designs given in Figures 13A and 13B. Assume each system consists of m machines and implements the same n rule security policy. Let the total arrival rate to each system be λ packets per unit time and each node performs x rules per unit time.

For the data-parallel firewall, traffic arrives at the packet distributor, which evenly distributes the traffic. As a result the arrival rate to each machine is λ m. The service rate is x n, since each node implements the complete rule set. The end-to-end delay across any machine in the parallel firewall is

$$E_d(T) = \frac{1}{\frac{x}{n} - \frac{\lambda}{m}} \tag{8}$$

In contrast, rules are distributed across each machine, in the function-parallel firewall. Furthermore, all traffic arriving to the system is forwarded to each machine. If we assume the rules in each machine are independent (no policy DAG edges spanning machines), then the end-to-end delay across any machine in the functional-parallel firewall is

$$E_f(T) = \frac{1}{\frac{m \cdot x}{n} - \lambda} \tag{9}$$

The reduction in delay compared to a data-parallel firewall system is 1 m; therefore, the functional-parallel system has the potential to be m times faster than a (data) parallel system. However, the scalability of the functional-parallel system is dependent on the rule set.

Figures 17A and 17B compare the performance of function-parallel, data-parallel, and single firewall designs. More particularly, Figure 17A compares the average among the different firewall designs as the arrival rate λ increases. As can be seen from Figure 17A, the function-parallel design achieves lower average delay as the arrival rate increases than the data-parallel design. Figure 17B illustrates the average delay as arrival rate and the number of rules increase. In Figure 17B, the function-parallel design again achieves better simulated performance than the data-parallel design.

Experimental Results

The previous section described models that provide insight to the average delay across different firewall designs. This information is beneficial in guiding initial design; however, we are also interested in the maximum delay encountered. For this reason, a discrete event simulator has been developed to model the different systems.

The data-parallel firewall and function-parallel firewall designs were simulated, where both systems implemented a 50-rule policy and each consisted of 5 firewall nodes. Three classes of traffic were modeled, first, second, and background. The first and second classes represent legitimate traffic that was eventually accepted by the firewall system, while background traffic was typically dropped. The two designs were simulated under normal conditions, where each traffic type had the same volume. Figures 18A and 18B shows the simulation results. As seen in Figure 18A and predicted by the theoretical models, the average delay encountered by packets in the functional-parallel design was lower than the data-parallel designs. The function-parallel design also provided lower maximum delays than the data-parallel, as seen in Figure 18B. In summary, the theoretical models and simulation results indicate function-parallel is a better design than a traditional single firewall system or a data-parallel system; yet, function-parallel requires no more computing resources than data-parallel.

Summary

Thus, as described above, the subject matter described herein includes methods, systems, and computer program products for providing a function-

IPR2023-01353 Page 00751

parallel firewall where different portions of a firewall rule set are distributed among different firewall nodes. Figure 19 is a flow chart illustrating exemplary steps of a process for filtering packets using a rule set distributed among firewall nodes in a function-parallel manner. Referring to Figure 19, in step **1900**, packet filtering rules of a rule set are distributed among nodes in a function-parallel manner so that different nodes implement different portions of the rule set. In step **1902**, packets to be filtered are replicated to each of the nodes. In step **1904**, the packets to be filtered are replicated to each of the nodes. In a hierarchical design, step **1904** may be implemented at each level in the hierarchy. If a packet passes or fails at one level, the packet may be forwarded into a network or dropped, depending on the distribution of the rule set. In step **1904**, the rules are applied to filter the packets.

The steps illustrated in Figure 19 and in all of the examples described herein may be applied to firewall systems that filter packets based on packet header information, intrusion detection systems that filter packets based on packet payload information, or intrusion protection systems that filter packets based on a combination of packet header and packet payload information.

The following references correspond to the citations in this document. The disclosure of each of these references is hereby incorporated herein by reference it its entirety.

References

[1] A. V. Aho, J.E. Hopcroft, and J. D. Ullman. *Data Structures and Algorithms*. Addison-Wesley, 1987.

[2] M. Al-Suwaiyel and E. Horowitz. Algorithms for Trie Compaction. *ACM Transactions on Database Systems*, 9(2):243–263, June 1984.

[3] S. M. Bellovin and W. Cheswick. Network Firewalls. *IEEE Communications Magazine*, pages 50–57, Sept. 1994.

[4] C. Benecke. A Parallel Packet Screen for High Speed Networks. In *Proceedings of the 15th Annual Computer Security Applications Conference*, 1999.

[5] G. Bolch, S. Greiner, H. de Meer, and K. S. Trivedi. *Queueing Networks and Markov Chains*. John Wiley and Sons, Inc., 1998.

[6] D. Comer. Analysis of a Heuristic for Full Trie Minimization. *ACM Transactions on Database Systems*, 6(3):513–537, Sept. 1981.

[7] U. Ellermann and C. Benecke. Firewalls for ATM Networks. In *Proceedings of INFOSEC'COM*, 1998.

[8] D. Eppstein and S. Muthukrishnan. Internet Packet Filter Management and Rectangle Geometry. In *Proceedings of the Symposium on Discrete Algorithms*, pages 827–835, 2001.

[9] E. Fulp, Z. Fu, D. S. Reeves, S. F. Wu, and X. Zhang. Preventing Denial of Service Attacks on Quality of Service. In *Proceedings of the 2001 DARPA Information Survivability Conference and Exposition II*, 2001.

[10] R. Funke, A. Grote, and H.-U. Heiss. Performance Evaluation of Firewalls in Gigabit-Networks. In *Proceedings of the Symposium on Performance Evaluation of Computer and Telecommunication Systems*, 1999.

[11] X. Gan, T. Schroeder, S. Goddard, and B. Ramamurthy. LSMAC vs. SLNAT: Scalable ClusterbasedWeb Servers. *Journal of Networks, Software Tools, and Applications*, 3(3):175–185, 2000.

[12] S. Goddard, R. Kieckhafer, and Y. Zhang. An Unavailability Analysis of Firewall Sandwich Configurations. In *Proceedings of the 6th IEEE Symposium on High Assurance Systems Engineering*, 2001.

[13] A. Hari, S. Suri, and G. Parulkar. Detecting and Resolving Packet Filter Conflicts. In *Proceedings of IEEE INFOCOM*, pages 1203–1212, 2000.

[14] O. Paul and M. Laurent. A Full Bandwidth ATM Firewall. In *Proceedings of the 6th European Symposium on Research in Computer Security ESORICS'2000*, 2000.

[15] L. Qui, G. Varghese, and S. Suri. Fast Firewall Implementations for Software and Hardware-Based Routers. In *Proceedings of ACM SIGMETRICS*, June 2001.

[16] G. V. Rooij. Real Stateful TCP Packet Filtering in IP Filter. In *Proceedings of the 10th USENIX Security Symposium*, 2001.

[17] M. Schwartz. *Telecommunication Networks: Protocols, Modeling, and Analysis*. Addison-Wesley, 1987.

[18] S. Suri and G. Varghese. Packet Filtering in High Speed Networks. In *Proceedings of the Symposium on Discrete Algorithms*, pages 969 – 970, 1999.

[19] P. Warkhede, S. Suri, and G. Varghese. Fast Packet Classification for Two-Dimensional Conflict-Free Filters. In *Proceedings of IEEE INFOCOM*, pages 1434–1443, 2001.

[20] J. Xu and M. Singhal. Design and Evaluation of a High-Performance ATM Firewall Switch and Its Applications. *IEEE Journal on Selected Areas in Communications*, 17(6):1190–1200, June 1999.

[21] R. L. Ziegler. *Linux Firewalls*. New Riders, second edition, 2002.

[22] E. D. Zwicky, S. Cooper, and D. B. Chapman. *Building Internet Firewalls*. O'Reilly, 2000.

[23] E. Al-Shaer and H. Hamed. Modeling and Management of Firewall Policies. *IEEE Transactions on Network and Service Management*, 1(1), 2004.

[24] S. M. Bellovin and W. Cheswick. Network Firewalls. *IEEE Communications Magazine*, pages 50–57, Sept. 1994.

[25] D. E. Culler and J. P. Singh. *Parallel Computer Architecture: A Hardware/Software Approach*. Morgan Kaufman, 1999.

[26] E. W. Fulp. Firewall Architectures for High Speed Networks. Technical Report 20026, Wake Forest University Computer Science Department, 2002.

[27] E. W. Fulp. Optimization of Firewall Policies Using Directed Acyclical Graphs. Submitted for publication, 2004.

[28] R. Rivest. On Self-Organizing Sequential Search Heuristics. *Communications of the ACM*, 19(2), 1976.

[29] W. E. Smith. Various Optimizers for Single-Stage Production. *Naval Research Logistics Quarterly*, 3:59 – 66, 1956.

It will be understood that various details of the present subject matter may be changed without departing from the scope of the present subject matter. Furthermore, the foregoing description is for the purpose of illustration only, and not for the purpose of limitation.

CLAIMS

What is claimed is:

1. A function-parallel firewall comprising:
   (a) a first firewall node for filtering received packets using a first portion of a rule set including a plurality of rules, the first portion including less than all of the rules in the rule set; and
   (b) at least one second firewall node for filtering packets using a second portion of the rule set, the second portion including at least one rule in the rule set that is not present in the first portion, wherein the first and second portions together include all of the rules in the rule set.
2. The function-parallel firewall of claim 1 wherein the first and second portions respectively include rules present in first and second levels of a firewall hierarchy.
3. The function-parallel firewall of claim 2 wherein the first and second firewall nodes are connected to each other such that the rules in the first level are applied to received packets before the rules in the second level.
4. The function-parallel firewall of claim 3 wherein the first firewall node is adapted to forward packets that pass one of the rules in the first portion to an internal network in a manner that bypasses the second firewall node.
5. The function-parallel firewall of claim 3 wherein the first firewall node is adapted to discard packets that fail one of the rules in the first portion.
6. The function-parallel firewall of claim 3 wherein the rules in the second portion are more processor-intensive than the rules in the first portion.
7. The function-parallel firewall of claim 1 comprising:
   (a) a packet distributor for replicating each packet to the first and second firewall nodes; and
   (b) a gate machine for collecting results of applying the first and second portions of the rule set to each packet from the first and second firewall nodes and for making a forwarding decision based on the results.

IPR2023-01353 Page 00755

8. The function-parallel firewall of claim 7 wherein each of the first and second firewall nodes is adapted to process the packets in a pipelined manner and wherein the gate machine is adapted to maintain state information for multiple packets simultaneously.

9. The function-parallel firewall of claim 7 wherein the gate machine is adapted to signal the second firewall node when processing of a packet is no longer needed due to matching of the packet with one of the rules used by the first firewall node.

10. The function-parallel firewall of claim 1 wherein the first and second firewall nodes implement rules of the same hierarchical level.

11. The function-parallel firewall of claim 1 wherein at least one of the rules in the first portion is present in the second portion to provide redundancy.

12. The function-parallel firewall of claim 1 wherein the first and second firewall nodes are configured to implement a gateless design wherein the rules are distributed among the first and second firewall nodes such that for any given packet, one of the first and second firewall nodes will accept the packet and the other of the first and second firewall nodes will deny the packet.

13. The function-parallel firewall of claim 1 wherein the first and second firewall nodes are implemented on different physical machines.

14. The function-parallel firewall of claim 1 wherein the first and second firewall nodes are implemented on different processors in the same machine.

15. The function-parallel firewall of claim 1 wherein the first and second firewall nodes are adapted to simultaneously process copies of the same packets.

16. The function-parallel firewall of claim 1 wherein the rule set includes firewall rules for filtering packets based on packet headers.

17. The function-parallel firewall of claim 1 wherein the rule set includes intrusion detection rules for filtering packets based on packet payloads.

18. The function-parallel firewall of claim 1 wherein the rule set includes intrusion protection rules for filtering packets based on a combination of packet headers and packet payloads.

19. A stateful function-parallel firewall system comprising:

(a) a first function-parallel firewall subsystem including a plurality of firewall nodes implementing a first rule set for filtering packets arriving in a network, wherein the first function-parallel firewall subsystem includes at least first and second firewall nodes respectively implementing first and second portions of the first rule set, the first portion including at least one rule that is not present in the second portion; and

(b) a second function-parallel firewall sub-system including a second set of firewall nodes implementing a second rule set for filtering packets departing from the network, wherein the second function-parallel firewall subsystem includes at least third and fourth firewall nodes, the third and fourth firewall nodes implementing first and second portions of the second rule set, the first portion of the second rule set including at least one rule that is not present in the second portion of the second rule set, wherein the first and second function-parallel firewall subsystems share state information regarding connections established through the first and second sets of firewall nodes.

20. The stateful function-parallel firewall system of claim 19 comprising a database node coupled to the first and second function-parallel firewall subsystems, wherein the database node stores the state information, wherein the first set of firewall nodes obtains state information regarding connections established through the second set of firewall nodes from the database node, and wherein the second set of firewall nodes obtains state information regarding connections established through the first set of firewall nodes from the database node.

21. The stateful function-parallel firewall system of claim 20 wherein the first and second sets of firewall nodes store copies of the state information maintained by the database node.

22. The stateful function-parallel firewall system of claim 19 wherein the first and second function-parallel firewall subsystems each include a gateless design wherein rules are distributed among individual firewall nodes

such that a received packet will be accepted by a single firewall node and will be denied by remaining firewall nodes.

23. The stateful function-parallel firewall system of claim 19 wherein the firewall nodes of the first subsystem are adapted to simultaneously process copies of the same packets arriving in the network and wherein the firewall nodes in the second subsystem are adapted to simultaneously process copies of packets departing from the network.

24. The stateful function-parallel firewall system of claim 19 wherein the first and second rule sets include firewall rules for filtering packets based on packet header information.

25. The stateful function-parallel firewall system of claim 19 wherein the first and second rule sets include intrusion detection rules for filtering packets based on payloads of the packets.

26. The stateful function-parallel firewall system of claim 19 wherein the first and second rule sets include intrusion protection rules for filtering packets based on a combination of packet header and packet payload information.

27. A firewall grid comprising:

(a) a plurality of firewall nodes being physically connected to each other via a network for filtering packets; and

(b) a controller for controlling logical connections between the firewall nodes, wherein the controller is adapted to configure the firewall nodes in a logical firewall hierarchy without changing physical connections between the firewall nodes.

28. The firewall grid of claim 27 wherein the firewall nodes are adapted to simultaneously process copies of the same packets.

29. The firewall grid of claim 27 wherein the logical firewall hierarchy comprises a function-parallel hierarchy where individual firewall nodes implement different portions of a rule set.

30. The firewall grid of claim 27 wherein the controller is adapted to detect congestion at a first level in the firewall hierarchy and is adapted to reconfigure the logical connections between the firewall nodes to add a firewall node from another level in the firewall hierarchy to the level at which congestion was detected.

31. A system for providing network access control based on a function-parallel policy, the system comprising:
    (a) a first node for filtering received packets using a first portion of a rule set including a plurality of rules, the first portion including less than all of the rules in the rule set; and
    (b) at least one second node for filtering packets using a second portion of the rule, the second portion including at least one rule in the rule set that is not present in the first portion, wherein the first and second portions together include all of the rules in the rule set.

32. The system of claim 31 wherein the rule set includes firewall rules for filtering packets based on packet header information.

33. The system of claim 31 wherein the rule set includes intrusion detection rules for filtering packets based on packet payload information.

34. The system of claim 31 wherein the rule set includes intrusion protection rules for filtering packets based on a combination of packet header and packet payload information.

35. A method for controlling access to a network based on a set of packet filtering rules distributed in a function-parallel manner, the method comprising:
    (a) distributing packet filtering rules of a rule set among a plurality of different nodes in a function-parallel manner so that at least some of the different nodes implement different portions of the rule set;
    (b) replicating packets to each of the nodes; and
    (c) applying the rules to filter the packets.

36. The method of claim 35 wherein the packet filtering rules comprise firewall rules for filtering packets based on packet header information.

37. The method of claim 35 wherein the packet filtering rules comprise intrusion detection rules for filtering packets based on packet payload information.

38. The method of claim 35 wherein the packet filtering rules comprise intrusion protection rules for filtering packets based on a combination of packet header and packet payload information.

IPR2023-01353 Page 00759

39. The method of claim 35 comprising, receiving a first packet at a first node, determining whether the packet matches a first rule implemented by the first node, and, in response to determining that a match occurs, notifying the other nodes of the match.

40. The method of claim 36 comprising, at the other nodes, ceasing processing of the first packet.

41. A method for distributing rules in a function-parallel firewall, the method comprising:

(a) defining a rule set for a function-parallel firewall;

(b) assigning rules in the rule set to nodes and branches in a trie data structure, wherein each node in the trie data structure corresponds to a data field and each branch represents a value for each data field to be compared to values in corresponding fields in received packets; and

(c) pruning the trie data structure in a manner that preserves ordering of the rules in the rule set.

42. The method of claim 41 wherein pruning the trie data structure includes pushing a rule to its descendants and deleting the original rule.

43. The method of claim 41 comprising reducing the depth of the trie using level reordering.

44. The method of claim 41 wherein assigning rules in the rule set to nodes and branches in the trie data structure includes assigning rules requiring longer processing time to lower levels in the trie.

45. A method for assigning rules to a plurality of firewall nodes, the method comprising:

(a) assigning packet filtering rules to nodes in a directed acyclical graph (DAG);

(b) representing relationships between rules by edges in the DAG; and

(c) distributing the rules among a plurality of firewall nodes using the DAG such that different firewall nodes implement different rules and such that relationships between the rules specified in the DAG are preserved.

46. The method of claim 45 wherein distributing the rules among a plurality of firewall nodes using the DAG includes distributing the rules such that intersecting accept rules are not present in more than one firewall node.

47. The method of claim 45 wherein distributing the rules among a plurality of firewall nodes using the DAG includes distributing the rules such that the edges to not span multiple firewall nodes.

48. The method of claim 45 wherein distributing the rules among a plurality of firewall nodes using the DAG includes replicating deny rules in each of the firewall nodes.

49. The method of claim 48 comprising eliminating a deny rule implemented by a first firewall node in response to determining that the deny rule for the first firewall node includes no inbound edges and outbound edges from the first deny rule terminate only in a superset deny rule.

50. A computer program product comprising computer executable instructions embodied in a computer readable medium for performing steps comprising:

(a) distributing packet filtering rules of a rule set among a plurality of different nodes in a function-parallel manner so that at least some of the different nodes implement different portions of the rule set;

(b) replicating packets to each of the nodes; and

(c) applying the rules to filter the packets.

51. A computer program product comprising computer executable instructions embodied in a computer readable medium for performing steps comprising:

(a) defining a rule set for a function-parallel firewall;

(b) assigning rules in the rule set to nodes and branches in a trie data structure, wherein each node in the trie data structure corresponds to a data field and each branch represents a value for each data field to be compared to values in corresponding fields in received packets; and

(c) pruning the trie data structure in a manner that preserves ordering of the rules in the rule set.

52. A computer program product comprising computer executable instructions embodied in a computer readable medium for performing steps comprising:

(a) assigning rules to nodes in a directed acyclical graph (DAG);

(b) representing relationships between rules by edges in the DAG; and

(c) distributing the rules among a plurality of firewall nodes using the DAG such that the relationships between the rules specified in the DAG are preserved.

IPR2023-01353 Page 00762

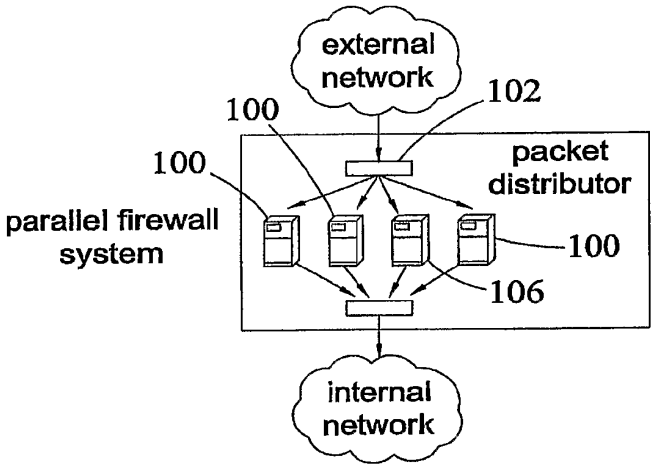


FIG. 1
(PRIOR ART)

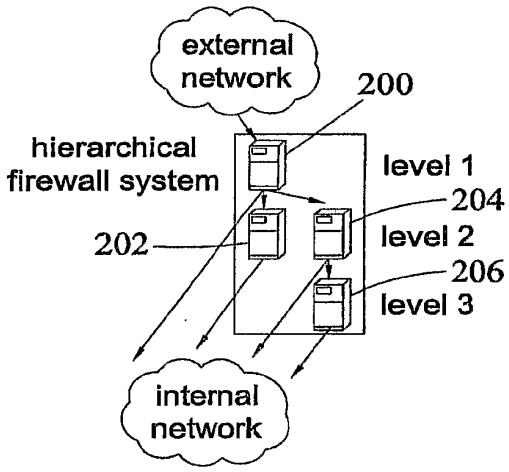

external network
200
hierarchical firewall system
level 1
204
level 2
202
206
level 3
internal network

FIG. 2

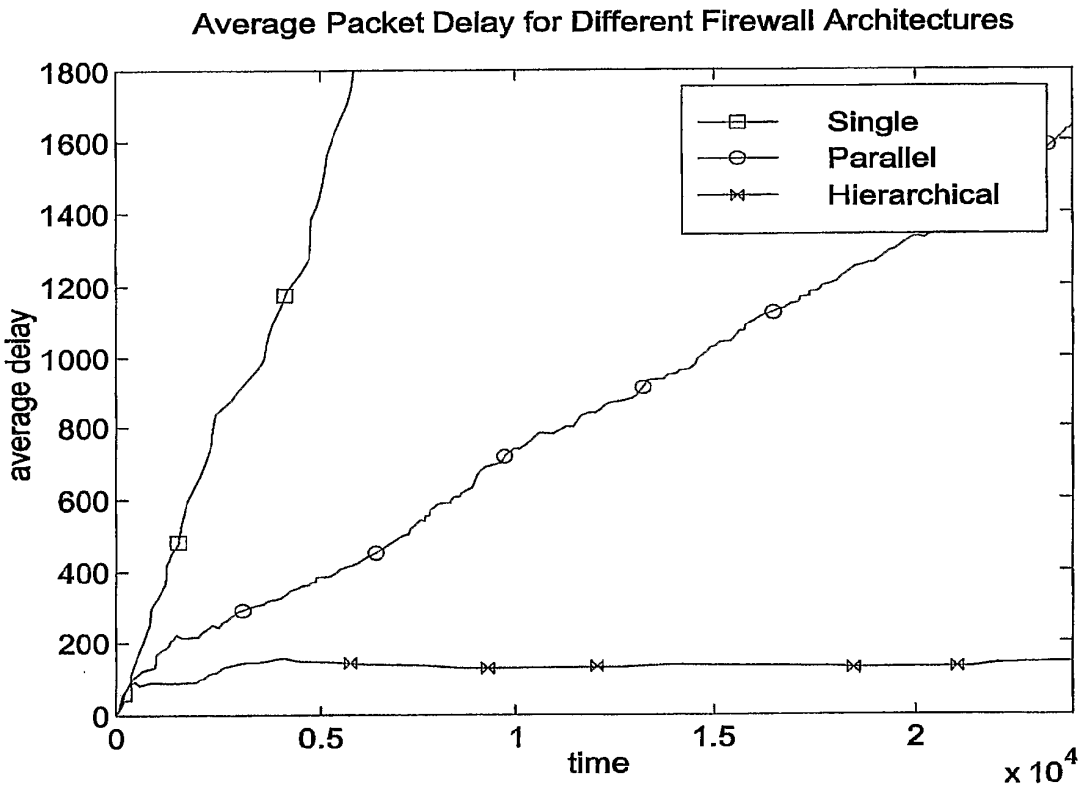

Average Packet Delay for Different Firewall Architectures
average delay
time
x 10^4
Single
Parallel
Hierarchical
0
200
400
600
800
1000
1200
1400
1600
1800
0
0.5
1
1.5
2

FIG. 3A

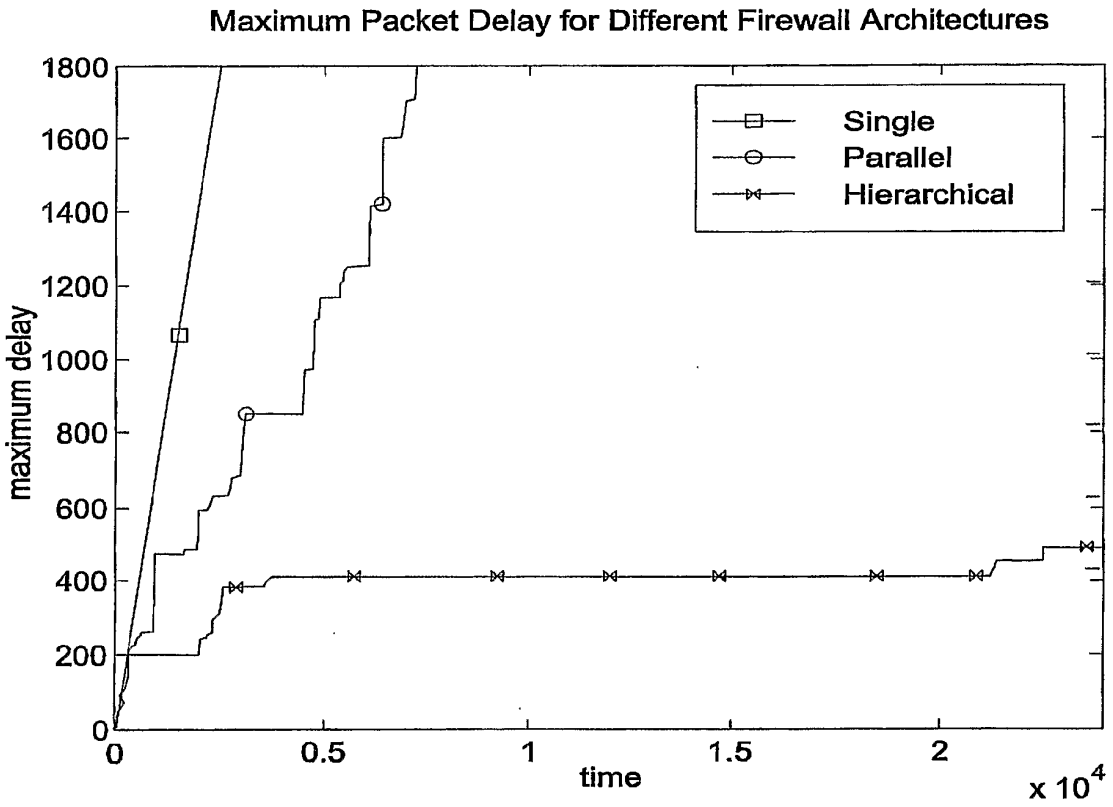

Maximum Packet Delay for Different Firewall Architectures
maximum delay
time
x 10^4
Single
Parallel
Hierarchical
0
200
400
600
800
1000
1200
1400
1600
1800
0
0.5
1
1.5
2

FIG. 3B

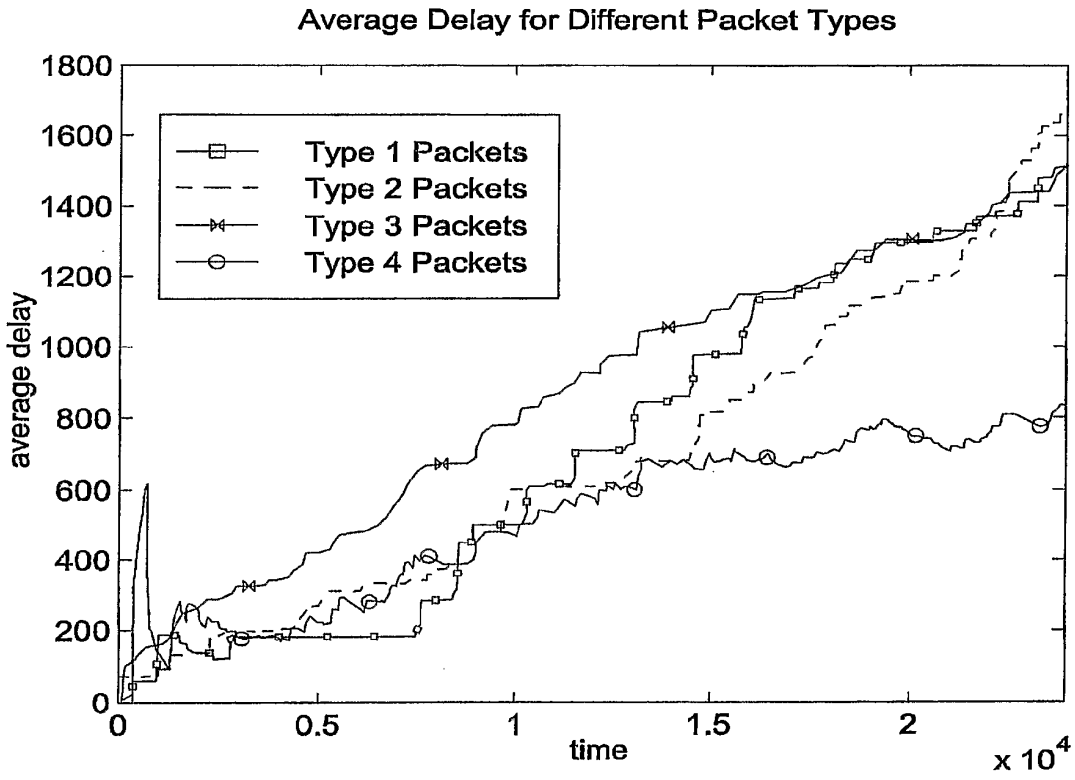

Average Delay for Different Packet Types
Type 1 Packets
Type 2 Packets
Type 3 Packets
Type 4 Packets
average delay
time
x 10^4
0
200
400
600
800
1000
1200
1400
1600
1800
0.5
1
1.5
2

FIG. 4A

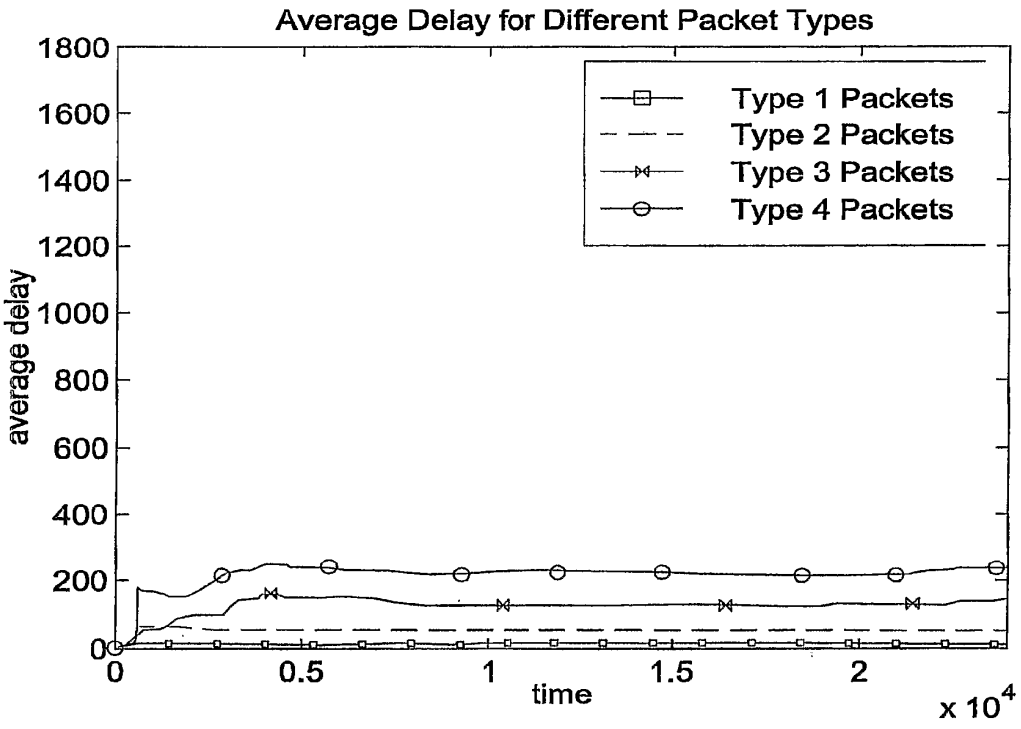

Average Delay for Different Packet Types
1800
1600
1400
1200
1000
800
600
400
200
0
average delay
0
0.5
1
1.5
2
time
x 10^4
Type 1 Packets
Type 2 Packets
Type 3 Packets
Type 4 Packets

FIG. 4B

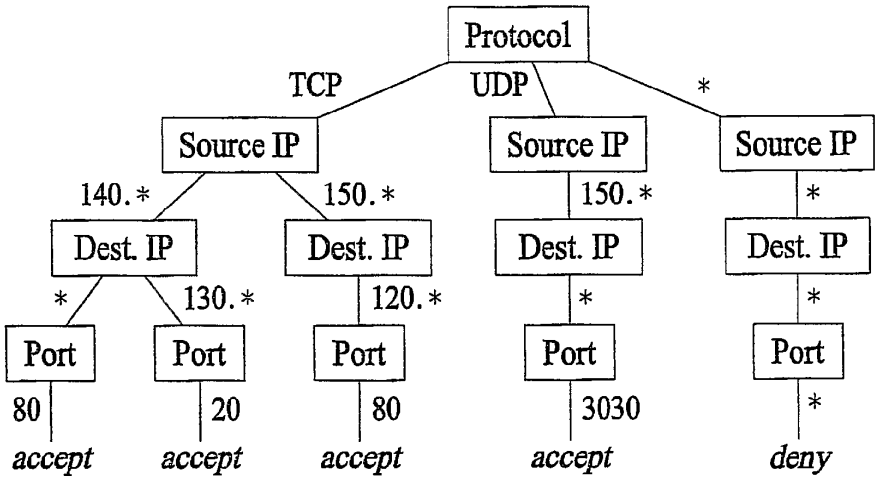

Protocol
TCP
UDP
*
Source IP
Source IP
Source IP
140.*
150.*
150.*
*
Dest. IP
Dest. IP
Dest. IP
Dest. IP
*
130.*
120.*
*
*
Port
Port
Port
Port
Port
80
20
80
3030
*
accept
accept
accept
accept
deny

FIG. 5

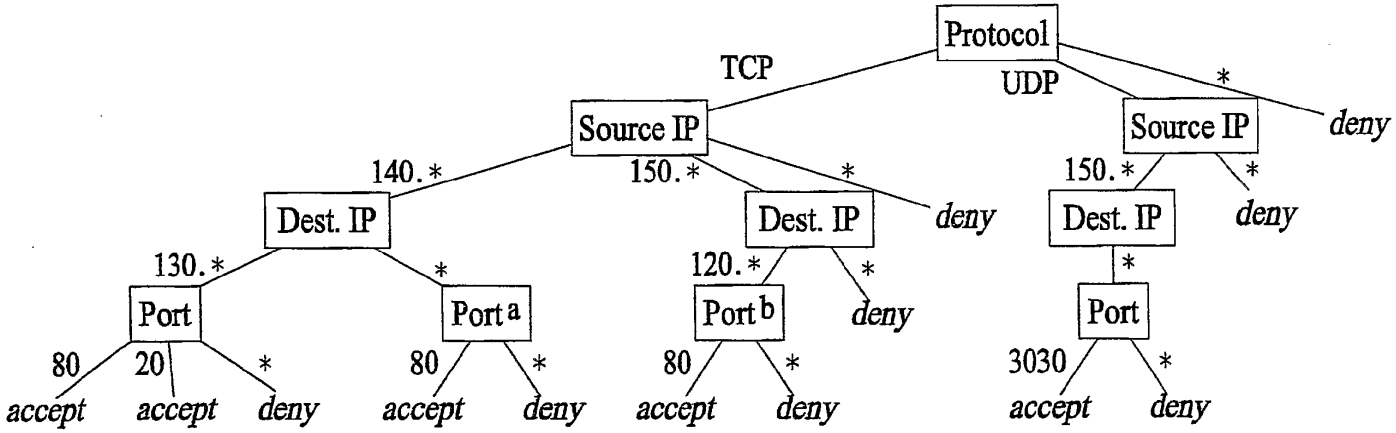

Protocol
TCP
UDP
*
deny
Source IP
Source IP
140.*
150.*
*
150.*
*
Dest. IP
Dest. IP
deny
Dest. IP
deny
130.*
*
120.*
*
*
Port
Port a
Port b
deny
Port
80
20
*
80
*
80
*
3030
*
accept
accept
deny
accept
deny
accept
deny
accept
deny

FIG. 6

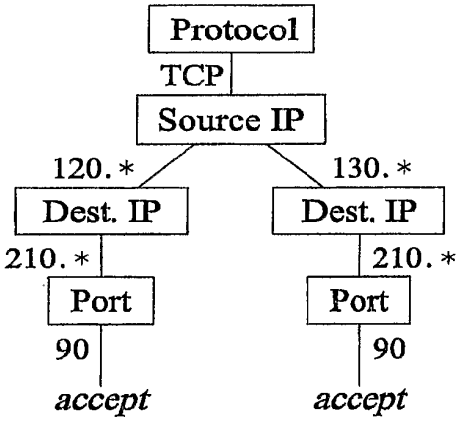

Protocol
TCP
Source IP
120.*
130.*
Dest. IP
Dest. IP
210.*
210.*
Port
Port
90
90
accept
accept

FIG. 7A

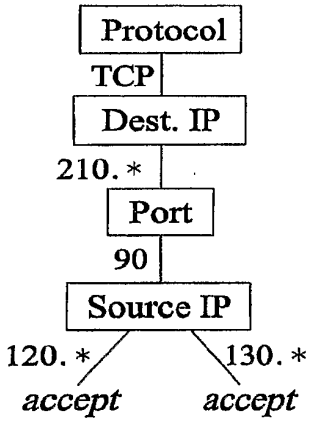

Protocol
TCP
Dest. IP
210.*
Port
90
Source IP
120.*
130.*
accept
accept

FIG. 7B

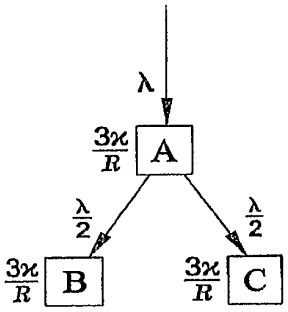

λ
3κ/R
A
λ/2
λ/2
3κ/R
B
3κ/R
C

FIG. 8A

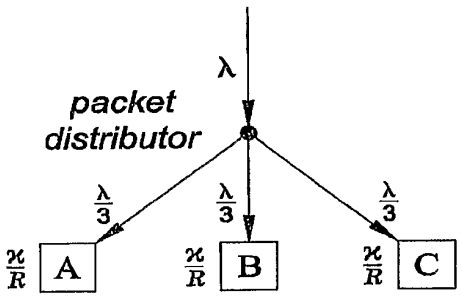

λ
packet
distributor
λ/3
λ/3
λ/3
κ/R
A
κ/R
B
κ/R
C

FIG.8B

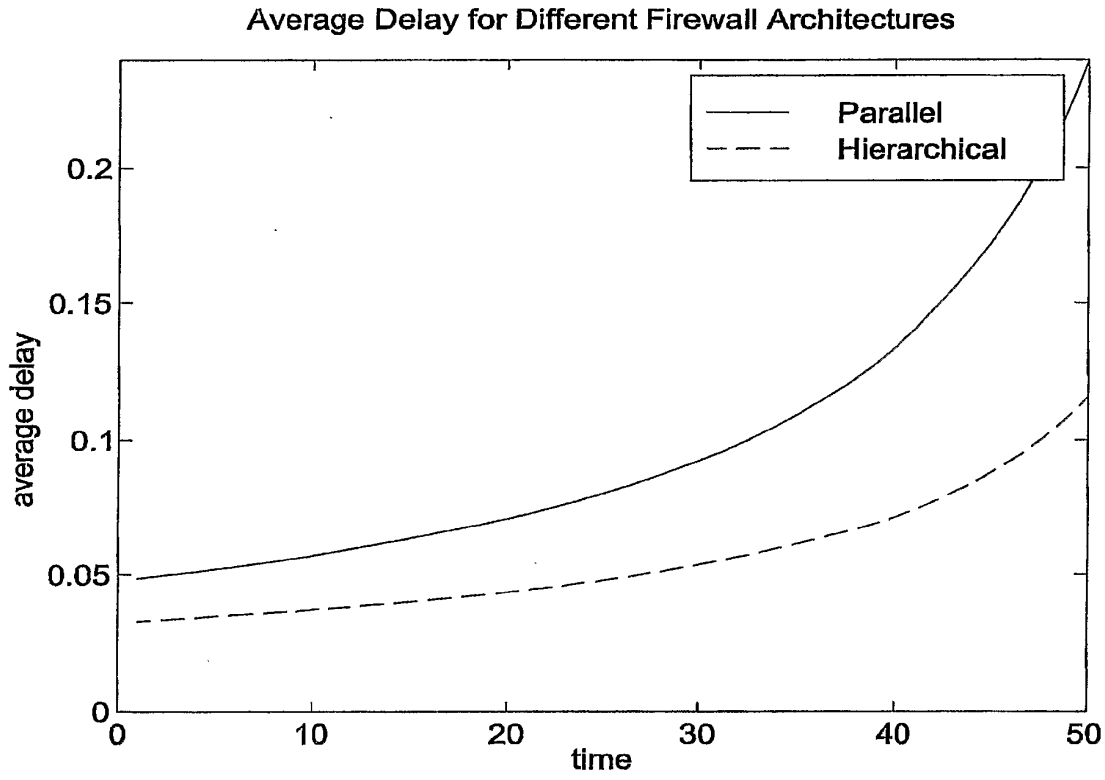

Average Delay for Different Firewall Architectures
Parallel
Hierarchical
average delay
0
0.05
0.1
0.15
0.2
0
10
20
30
40
50
time

FIG. 9

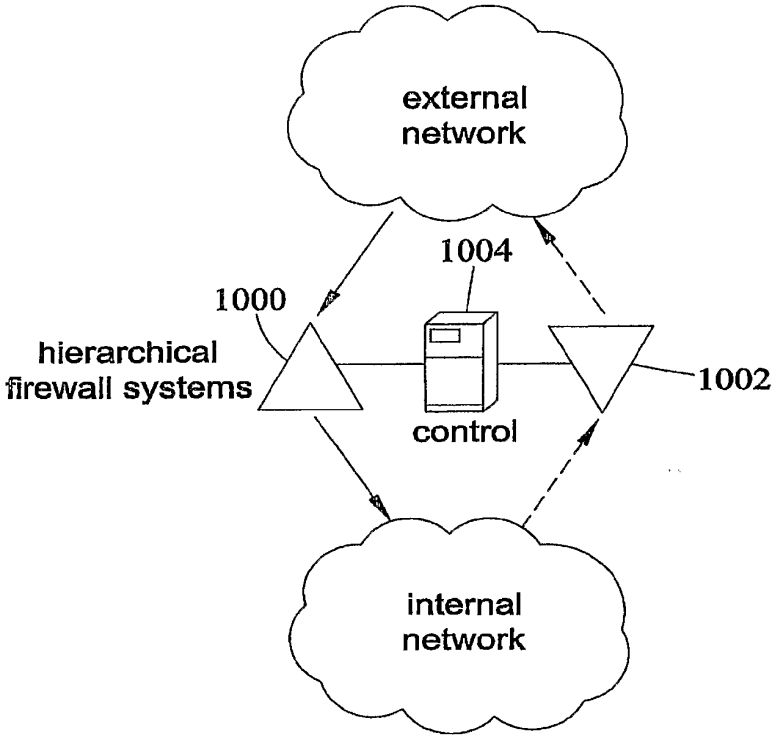

external network
1004
1000
hierarchical firewall systems
1002
control
internal network

FIG. 10

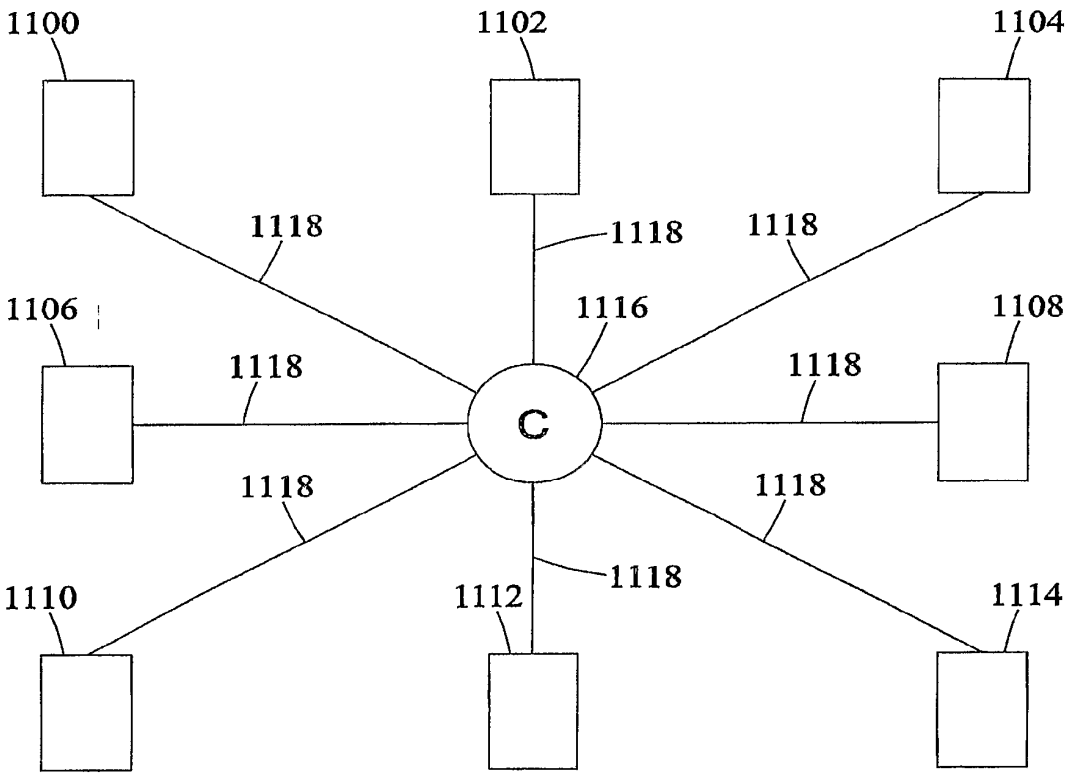

1100
1102
1104
1118
1118
1118
1106
1116
1108
1118
1118
C
1118
1118
1110
1112
1118
1114

FIG. 11A

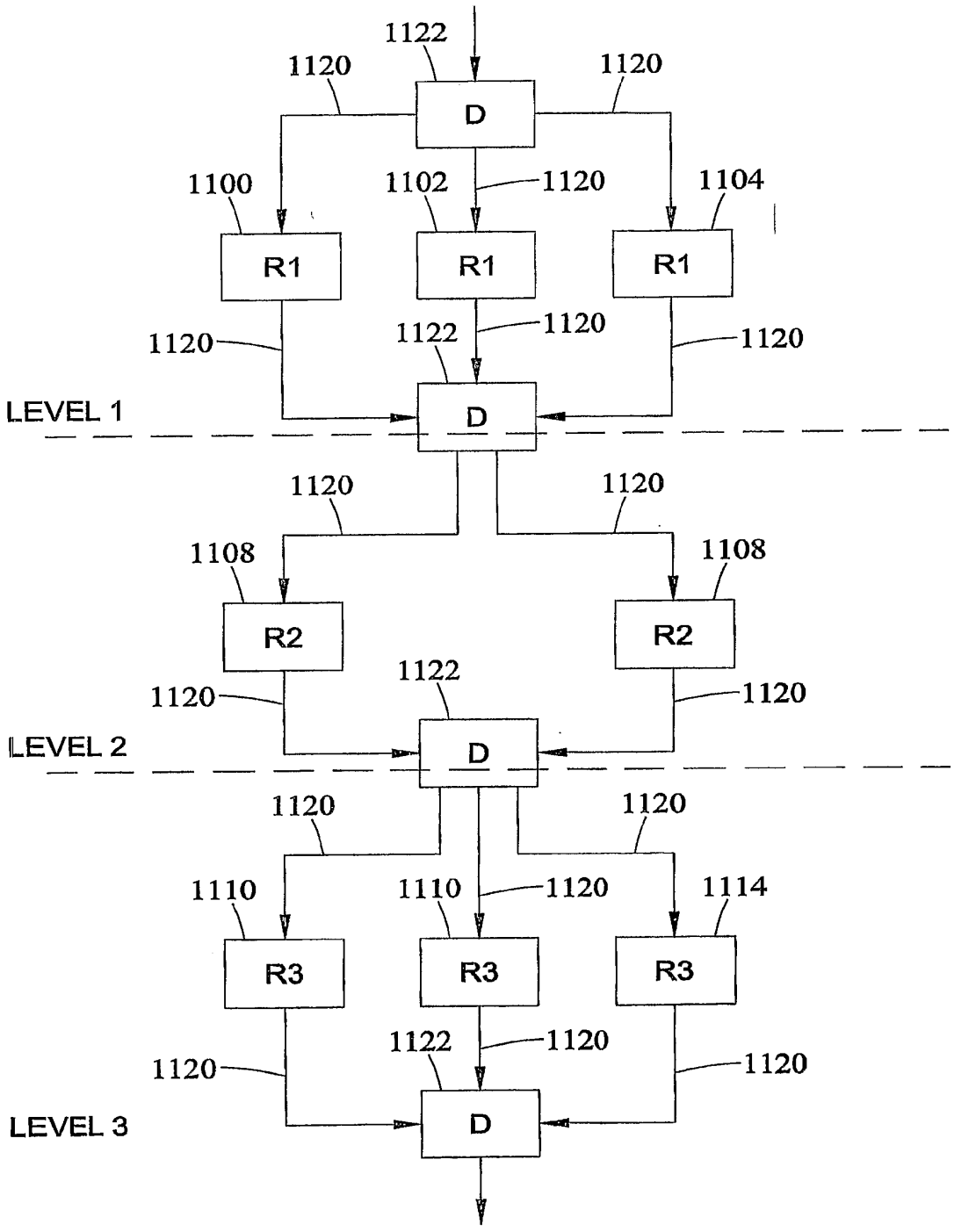

1122
1120
1120
D
1100
1102
1120
1104
R1
R1
R1
1120
1122
1120
1120
LEVEL 1
D
1120
1120
1108
1108
R2
R2
1122
1120
1120
LEVEL 2
D
1120
1120
1110
1110
1120
1114
R3
R3
R3
1122
1120
1120
1120
LEVEL 3
D

FIG. 11B

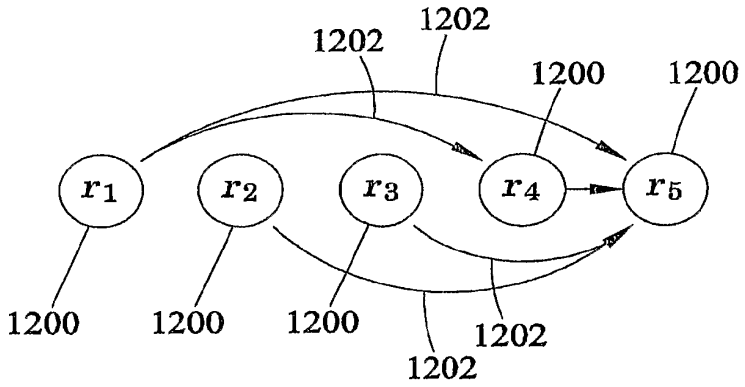


FIG. 12

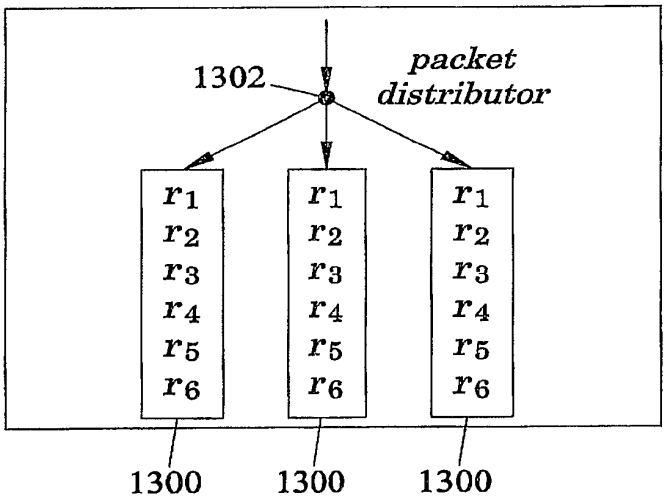


FIG. 13A

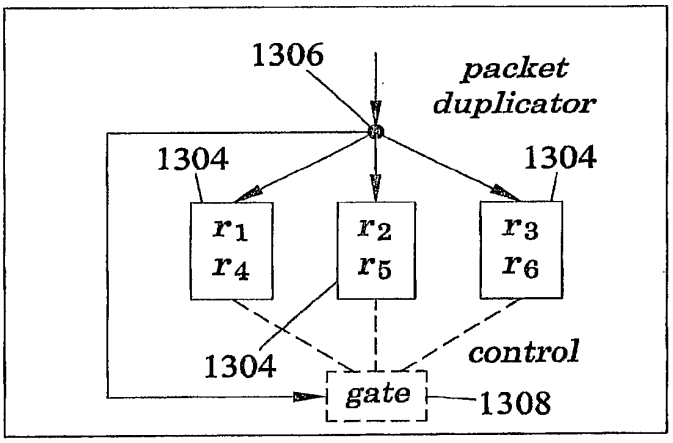


FIG. 13B

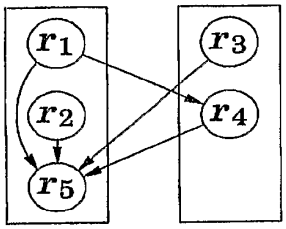


FIG. 14A

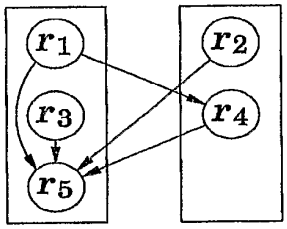


FIG. 14B

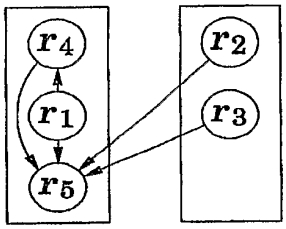


FIG. 14C

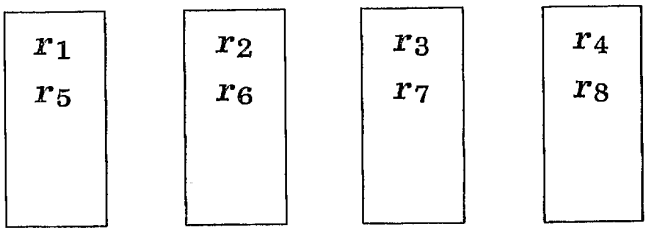


FIG. 15A

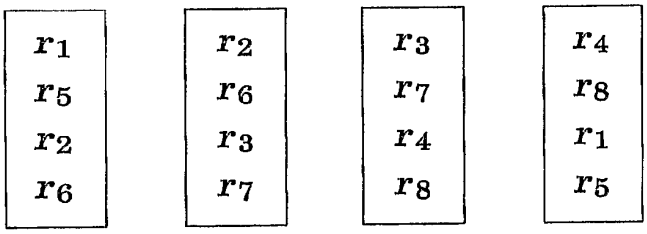


FIG. 15B

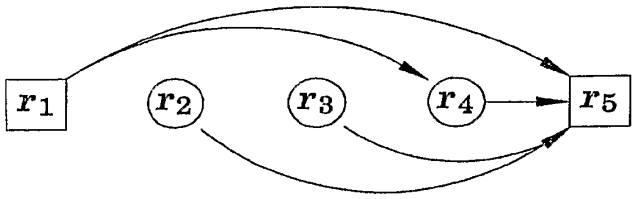


FIG. 16A

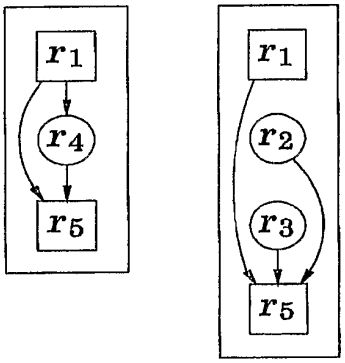


FIG. 16B

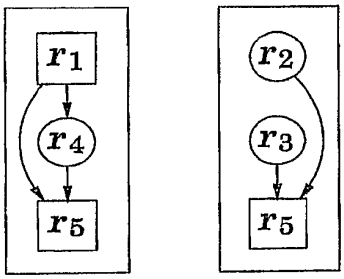


FIG. 16C

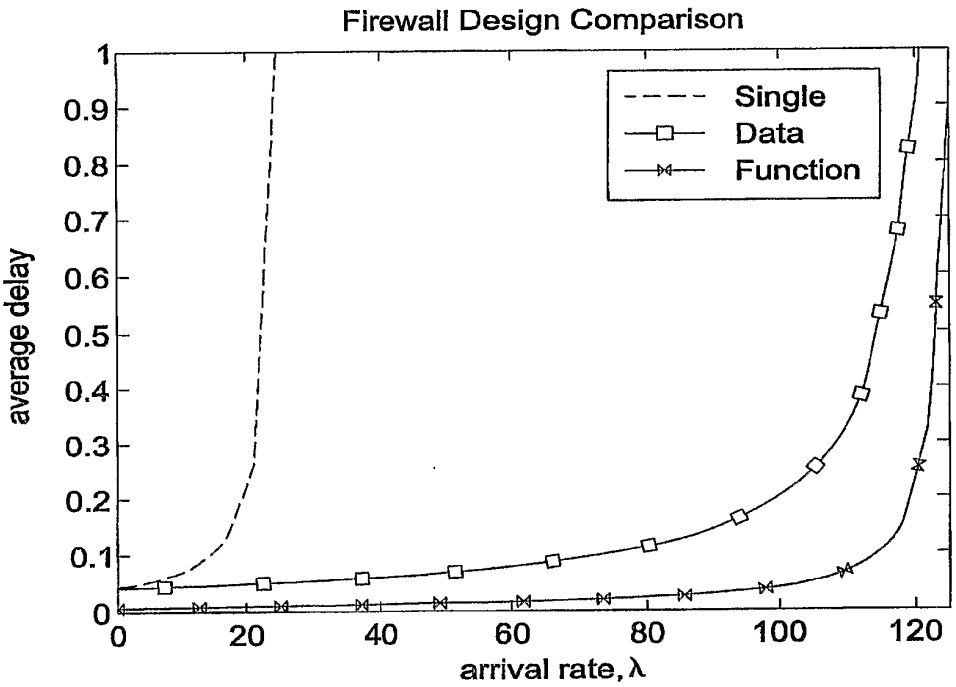

Firewall Design Comparison
Single
Data
Function
average delay
arrival rate, λ
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
0
20
40
60
80
100
120

FIG. 17A

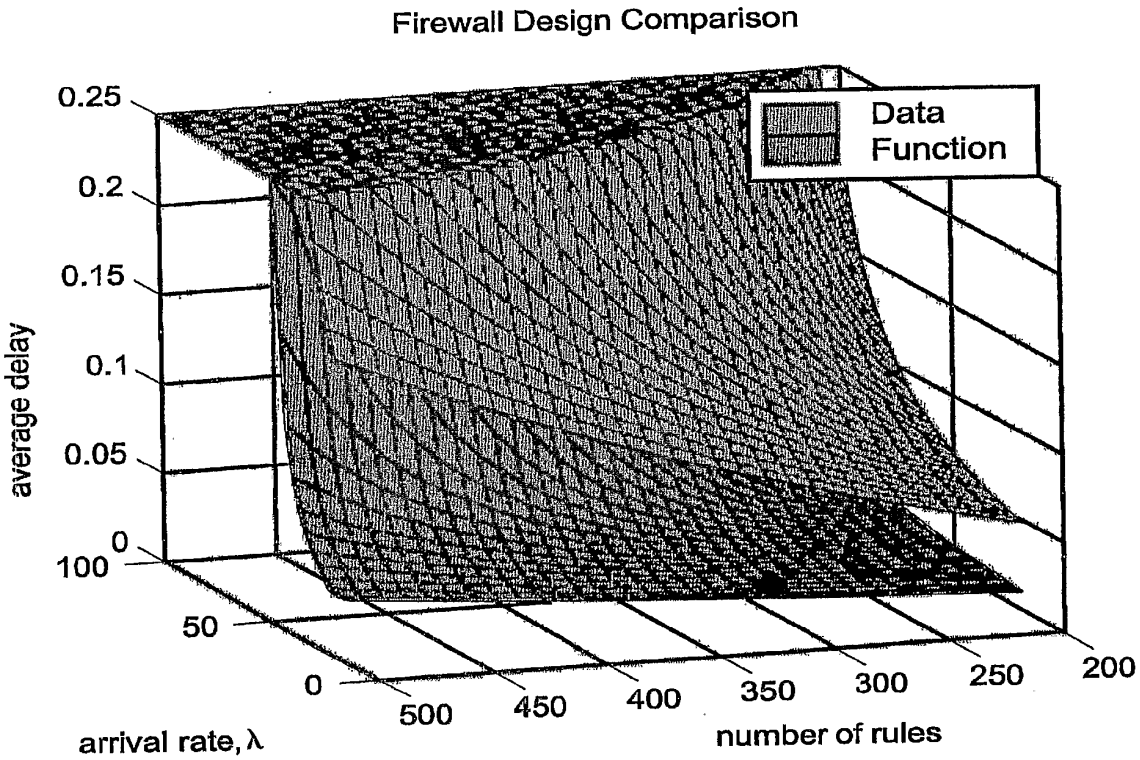

Firewall Design Comparison
Data
Function
average delay
0.25
0.2
0.15
0.1
0.05
0
100
50
0
arrival rate, λ
500
450
400
350
300
250
200
number of rules

FIG. 17B

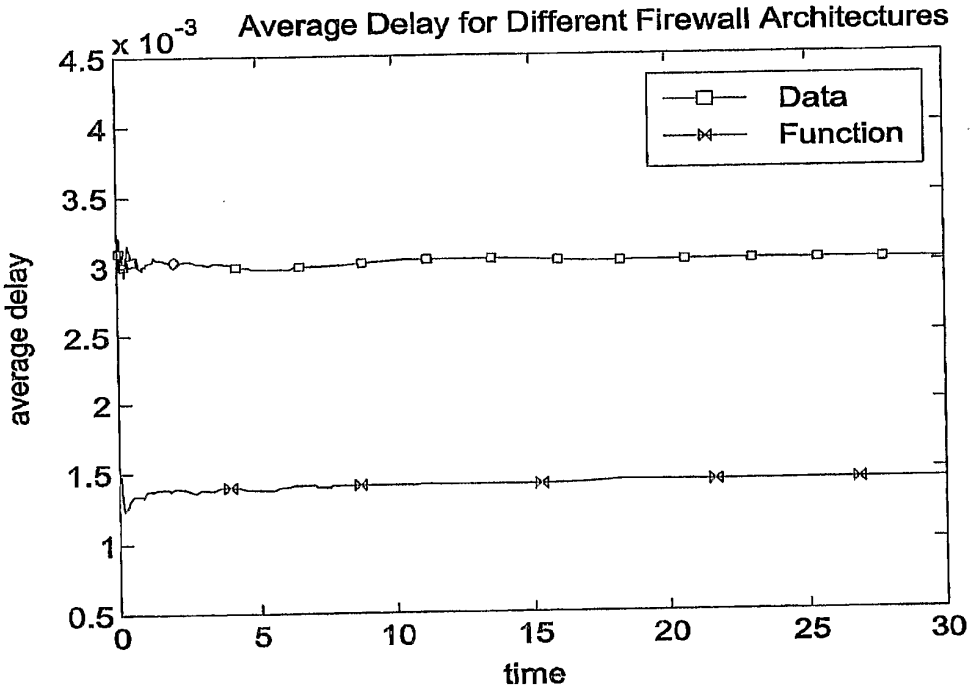

Average Delay for Different Firewall Architectures
x 10^-3
4.5
4
3.5
3
2.5
2
1.5
1
0.5
average delay
0
5
10
15
20
25
30
time
Data
Function

FIG. 18A

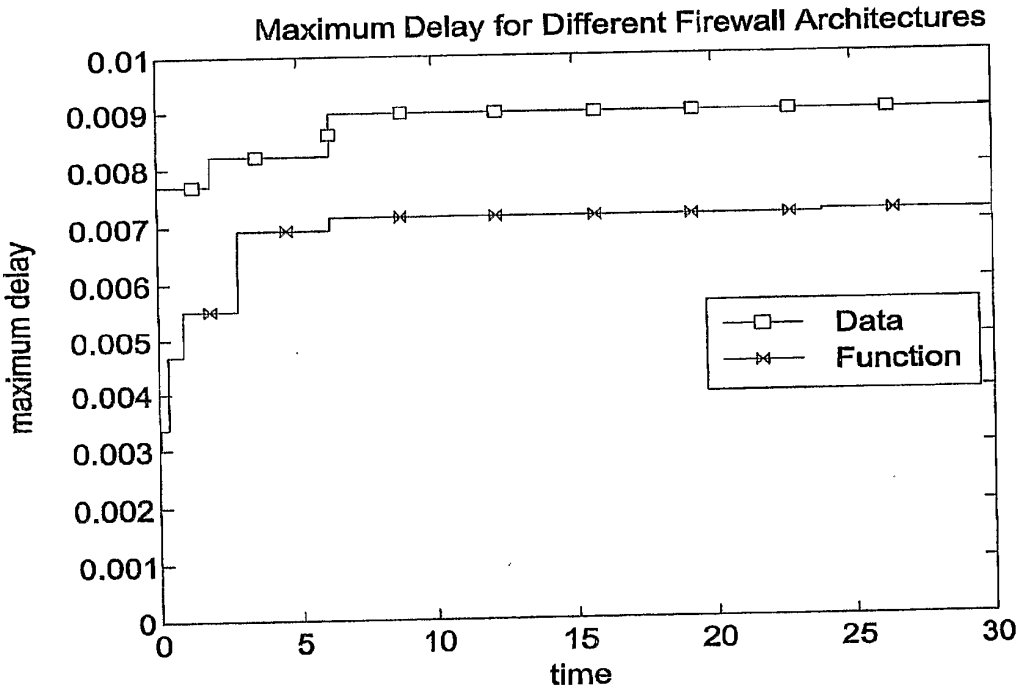

Maximum Delay for Different Firewall Architectures
maximum delay
time
Data
Function
0.01
0.009
0.008
0.007
0.006
0.005
0.004
0.003
0.002
0.001
0
0
5
10
15
20
25
30

FIG. 18B

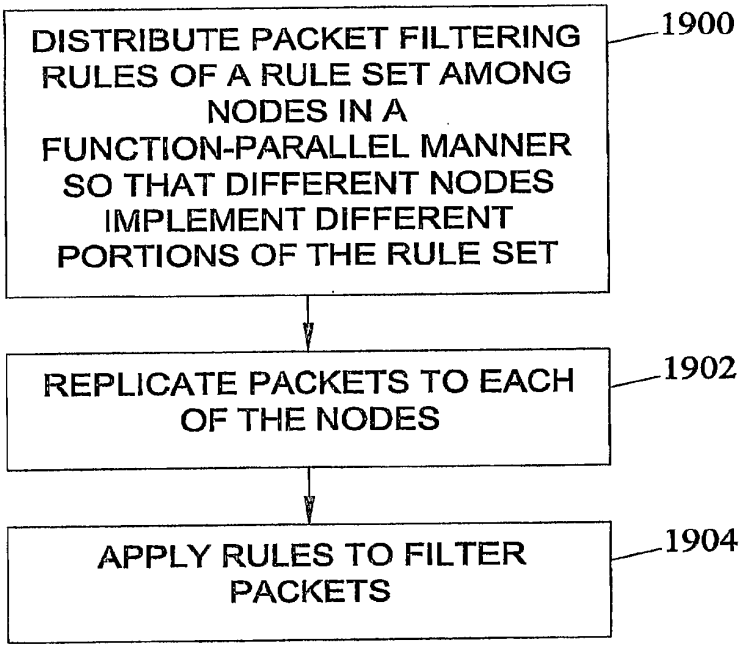

DISTRIBUTE PACKET FILTERING RULES OF A RULE SET AMONG NODES IN A FUNCTION-PARALLEL MANNER SO THAT DIFFERENT NODES IMPLEMENT DIFFERENT PORTIONS OF THE RULE SET
1900
REPLICATE PACKETS TO EACH OF THE NODES
1902
APPLY RULES TO FILTER PACKETS
1904

FIG. 19

Australian Government
IP Australia

LETTERS PATENT

STANDARD PATENT

2005328336

I, Victor Portelli, the Commissioner of Patents, grant a Standard Patent with the following particulars:

Name and Address of Patentee(s):

Wake Forest University
Office of Technology Asset Management, Wake Forest University Health Services, 200 East First Street, Suite 101, Winston Salem, North Carolina, 27101, United States of America

Name of Actual Inventor(s):

Fulp, Errin W. and Farley, Ryan J..

Title of Invention:

Method, systems, and computer program products for implementing function-parallel network firewall

Term of Letters Patent:

Twenty years from 22 December 2005

Priority Details :

Number	*Date*	*Filed with*
60/638,436	22 December 2004	US

COMMONWEALTH OF AUSTRALIA
AUSTRALIA
PATENT OFFICE

Dated this 5th day of January 2012

V J Portelli

Victor Portelli
Commissioner of Patents

PATENTS ACT 1990

(12) STANDARD PATENT (11) Application No. **AU 2005328336 B2**
(19) AUSTRALIAN PATENT OFFICE

(54) Title
Method, systems, and computer program products for implementing function-parallel network firewall

(51) International Patent Classification(s)
G06F 17/00 (2006.01)

(21) Application No: **2005328336** (22) Date of Filing: **2005.12.22**

(87) WIPO No: **WO06/093557**

(30) Priority Data

(31) Number	(32) Date	(33) Country
60/638,436	**2004.12.22**	**US**

(43) Publication Date: **2006.09.08**
(44) Accepted Journal Date: **2011.09.15**

(71) Applicant(s)
Wake Forest University

(72) Inventor(s)
Fulp, Errin W.;Farley, Ryan J.

(74) Agent / Attorney
Griffith Hack, Level 10 167 Eagle Street, Brisbane, QLD, 4001

(56) Related Art
US 2002/0038339 A1

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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
8 September 2006 (08.09.2006)

PCT

(10) International Publication Number
WO 2006/093557 A3

(51) International Patent Classification:
G06F 17/00 (2006.01)

(21) International Application Number:
PCT/US2005/047008

(22) International Filing Date:
22 December 2005 (22.12.2005)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
60/638,436 22 December 2004 (22.12.2004) US

(71) Applicant *(for all designated States except US)*: **WAKE FOREST UNIVERSITY** [US/US]; OFFICE OF TECHNOLOGY ASSET MANAGEMENT, WAKE FOREST UNIVERSITY HEALTH SCIENCES, 200 East First Street, Suite 101, Winston Salem, North Carolina 27101 (US).

(72) Inventors; and
(75) Inventors/Applicants *(for US only)*: **FULP, Errin, W.** [US/US]; 308 N. Hawthorne Road, Winston-salem, North Carolina 27104 (US). **FARLEY, Ryan, J.** [US/US]; 212 Radford Street, Winston-salem, North Carolina 27106 (US).

(74) Agent: **HUNT, Gregory, A.**; JENKINS, WILSON & TAYLOR, P.A., SUITE 1400 UNIVERSITY TOWER, 3100 Tower Boulevard, Durham, North Carolina 27707 (US).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BW, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, LY, MA, MD, MG, MK, MN, MW, MX, MZ, NA, NG, NI, NO, NZ, OM, PG, PH, PL, PT, RO, RU, SC, SD, SE, SG, SK, SL, SM, SY, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, YU, ZA, ZM, ZW.

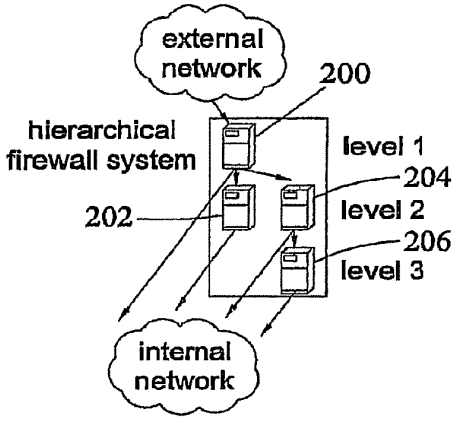
(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HU, IE, IS, IT, LT, LU, LV, MC, NL, PL, PT, RO, SE, SI, SK, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Published:
— *with international search report*

(88) Date of publication of the international search report:
18 January 2007

[Continued on next page]

(54) Title: METHOD, SYSTEMS, AND COMPUTER PROGRAM PRODUCTS FOR IMPLEMENTING FUNCTION-PARALLEL NETWORK FIREWALL



(57) Abstract: Methods, systems, and computer program products for providing function-paralell firewalls are disclosed. According to one aspect, a function-parallel firewall includes a first firewall node for filtering received packets using a first portion of a rule set including a plurality of rules. The first portion includes less than all of the rules in the rule set. At least one second firewall node filters packets using a second portion of the rule set. The second portion includes at least one rule in the rule set that is not present in the first portion. The first and second portions together include all of the rules in the rule set.

WO 2006/093557 A3

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.

DESCRIPTION

METHOD, SYSTEMS, AND COMPUTER PROGRAM PRODUCTS FOR IMPLEMENTING FUNCTION-PARALLEL NETWORK FIREWALL

RELATED APPLICATIONS

This application claims the benefit of U.S. Provisional Patent Application Serial No. 60/638,436, filed December 22, 2004; the disclosure of which is incorporated herein by reference in its entirety.

GOVERNMENT INTEREST

This invention was made with Government support under Grant No. DE-FG02-03ER25581 awarded by U.S. Department of Energy, Mathematical and Computational Information Sciences Division. The U.S. Government has certain rights in the invention.

TECHNICAL FIELD

The subject matter described herein relates to implementing network firewalls. More particularly, the subject matter described herein relates to methods, systems, and computer program products for implementing a function-parallel network firewall.

BACKGROUND ART

Firewalls remain the frontier defense for securing networks that are vital to private industry, government agencies, and the military. They enforce a security policy by inspecting and filtering traffic arriving at or departing from a secure network [3, 21, 22]. Meaningful inspections often involve a complex process, and if firewalls are to remain effective, they must adapt to constantly changing network demands, technology, and security threats.

As firewall technology evolves to meet new demands and threats, it must continue to act transparently to legitimate users, with little or no effect on the perceived network performance. The networks between two communicating machines should be invisible, especially if traffic requires specific network Quality of Service (QoS), such as bounds on the packet delay, jitter, and throughput. The firewall should process the legitimate traffic quickly, efficiently,

IPR2023-01353 Page 00790

and never violate the desired QoS. Unfortunately, the firewall can easily become a bottleneck, given increasing traffic loads and network speeds [12, 14, 18, 20, 21]. Packets must be inspected and compared against complex rule sets and tables, which is a time-consuming process. In addition, audit files must be updated with current connection information. As a result, current firewalls have difficulty maintaining QoS guarantees. Since a firewall can be overwhelmed with traffic, it is also susceptible to Denial of Service (DoS) attacks [21]. Such attacks merely overload/saturate the firewall with illegitimate traffic. Audit files and state tables quickly fill available storage space and legitimate traffic queued behind suspicious traffic encounters long delays. Legitimate users notice poor network performance, and, in the worst case, the secure network is disconnected from the outside world.

Inspecting traffic sent between networks, a firewall provides access control, auditing, and traffic control based on a security policy [3, 21, 22]. The security policy is a list of ordered rules, as seen in Table 1, that defines the action to perform on matching packets. A rule r can be viewed as a k-tuple ($r[1]$, $r[2]$, $r[3]$, ..., $r[k]$), where $r[i]$ is specified as a variable length prefix.

No.	Proto.	Source		Destination		Action
		IP	Port	IP	Port	
1	TCP	140.*	*	*	80	accept
2	TCP	150.*	*	120	80	accept
3	TCP	140.*	*	130	20	accept
4	UDP	150.*	*	*	3030	accept
5	*	*	*	*	*	deny

Table 1: Example Security Policy Consisting of Multiple Ordered Rules

In the examples described herein, rules will initially be represented as the 5-tuple (protocol, IP source address, source port number, IP destination address, destination port number). Fields can be fully specified or contain wildcards '*' in standard prefix format. For example the prefix 192.* would represent any IP address that has 192 as the first dotted-decimal number. In addition to the prefixes, each filter rule has an action, which is to accept or deny. An accept action passes the packet into or from the secure network,

while deny causes the packet to be discarded. Rules are applied in order to every arriving packet until a match is found; otherwise, a default action is performed [21, 22].

Unfortunately, security policies may contain anomalies, such as a packet that matches multiple rules, where the matching rules specify conflicting actions. Security policy anomaly detection and correction is the subject of continued research [13, 21] and is not the focus of the present subject matter. Given the type of security policy required, rule sets can become quite large and complex. Security can be further enhanced with connection state and packet audit information. For example, a table can be used to record the state of each connection, which is useful for preventing certain types of attacks (e.g., TCP SYN flood) [21, 22].

Traditional firewall implementations consist of a single dedicated machine, similar to a router, that sequentially applies the rule set to each arriving packet. However, packet filtering represents a significantly higher processing load than routing decisions [15, 18, 22]. For example, a firewall that interconnects two 100 Mbps networks would have to process over 300,000 packets per second [21]. Successfully handling this high traffic becomes more difficult as rule sets become more complex [4, 14, 22]. Furthermore, firewalls must be capable of processing even more packets as interface speeds increase. In a high-speed environment (e.g. Gigabit Ethernet), a single firewall can easily become a bottleneck and is susceptible to DoS attacks [4, 7, 10, 12]. An attacker could simply inundate the firewall with traffic, delaying or preventing legitimate packets from being processed. Building a faster single firewall is possible [8, 15, 18, 19, 20]; however, the benefits are temporary (traffic loads and interface speeds are increasing); it is not scalable; it is a single point of failure; and it is generally not cost-effective for all installations.

A data-parallel firewall (or firewall sandwich) where each firewall node implements an entire firewall rule set is another approach to increase the speed of processing traffic [4, 12, 14, 22]. As shown in Figure 1, the system consists of multiple identical firewalls **100** connected in parallel. Each machine implements the complete security policy and arriving packets are distributed across the machines for processing in parallel [4]. A packet distributor **102**

IPR2023-01353 Page 00792

2005328336 22 Aug 2011

implements a load balancing algorithm to distribute different packets to different firewall nodes. How the load-balancing algorithm distributes packets is vital to the system and is typically implemented as a high-speed switch in commercial products [11,12]. Although data-parallel firewalls achieve higher throughput than traditional firewalls [4] and have a redundant design, they have difficulty maintaining QoS across networks. For example, legitimate traffic can encounter delays if it is queued behind traffic that requires more processing. Under these circumstances, users notice poor network performance, which is growing concern as more network applications require QoS assurances. Furthermore, stateful inspection requires all traffic from a certain connection or exchange to traverse the same parallel machine, which is difficult to perform at high speeds [14]. Therefore, new firewall architectures are needed to meet the demands of future networks and the challenges of increasing security threats.

SUMMARY

In accordance with the present invention, there is provided a function-parallel firewall system comprising (a) a first firewall node for filtering received packets using a first portion of a rule set including a plurality of rules, the first portion including less than all of the rules in the rule set; and (b) at least one second firewall node for filtering packets using a second portion of the rule set, the second portion including at least one rule in the rule set that is not present in the first portion, wherein the first and second portions together include all of the rules in the rule set, wherein the first and second firewall nodes are configured to implement a gateless design wherein the rules are distributed among the first and second firewall nodes such that for a given packet, one of the first and second firewall nodes will accept the packet and the other of the first and second firewall nodes will deny the packet.

In accordance with the present invention, there is provided a method for controlling access to a network based on a set of packet filtering rules distributed in a function-parallel manner, the method comprising (a) distributing packet filtering rules of a rule set among a plurality of different nodes in a function-parallel manner so that at least some of the different nodes implement different portions of the rule set; (b) replicating packets to each of the nodes; and (c) applying the rules to filter the packet using a gateless design wherein the rules are distributed among first and second firewall nodes such that for a given packet, one of the first and second firewall nodes will accept the packet and the other of the first and second firewall nodes will deny the packet.

In accordance with the present invention, there is provided a computer program product comprising computer executable instructions embodied in a computer readable medium for performing steps comprising (a) a first firewall node for filtering received packets using a first portion of a rule set including a plurality of rules, the first portion including less than all of the rules in the rule set; and (b) at least one second firewall node for filtering packets using a second portion of the rule set, the second portion including at least one rule in the rule set that is not present in the first portion, wherein the first and second portions together include all of the rules in the rule set, wherein the first and second firewall nodes are configured to implement a gateless design wherein the rules are distributed among the first and second firewall nodes such that for any given packet, one of the first and second firewall nodes will accept the packet and the other of the first and second firewall nodes will deny the packet.

According to one aspect, the subject matter described herein includes a function-parallel firewall. The function-parallel firewall includes a first firewall node for filtering received packets using a portion of a rule set. The portion of the rule set may be less than all of the rules in the rule set. At least one second firewall node filters packets using at least one rule in the rule set that is not present in the portion implemented by the first firewall node. The first and second potions together include all of the rules in the rule set.

The term "firewall node," as used herein refers to a physical entity that implements one or more packet filtering rules. For example, a firewall node may be a single-processor computing device that implements the packet filtering rules. Alternatively, a firewall node may be a processor within a multi-processor computing device that implements one or more packet filtering rules. Thus, a multi-processor device may include multiple firewall nodes.

According to another aspect, the subject matter described herein includes a stateful function-parallel firewall system. The system includes a first function-parallel firewall subsystem including a plurality of firewall nodes implementing a first rule set for filtering packets arriving at a network. The first function-parallel firewall subsystem includes at least first and second firewall nodes respectively implementing first and second portions of the first rule set.

The first portion includes at least one rule that is not present in the second portion. A second function-parallel firewall subsystem includes a second set of firewall nodes. The second set of firewall nodes implements a firewall rule set for filtering packets departing from the network. The second function-parallel firewall subsystem includes at least third and fourth firewall nodes. The third and fourth firewall nodes implement at least one rule that is not present in the second portion of the second rule set. The function-parallel firewall subsystems share state information regarding connections established through the first and second sets of firewall nodes.

According to another aspect, the subject matter described herein includes a firewall grid. The firewall grid includes a plurality of firewall nodes that are physically connected to each other via a network for filtering packets. A controller controls logical connections between the firewall nodes. The controller is adapted to configure the firewall nodes in a logical firewall hierarchy without changing physical connections between the firewall nodes.

According to yet another aspect, the subject matter described herein includes a method for distributing rules in a function-parallel firewall. The method includes defining a rule set for a function-parallel firewall. The method further includes assigning rules in the rule set to nodes and branches in a trie data structure. Each node in the trie data structure corresponds to a data field and each branch represents a value for each data field to be compared to be values in corresponding fields of received packets. The trie data structure is pruned in a manner that preserves ordering of the rules in the rule set.

According to yet another aspect, the subject matter described herein includes a method for assigning rules to a plurality of firewall nodes. The method includes assigning rules to nodes in a directed acyclical graph (DAG). Relationships between rules are represented by edges in the DAG. Rules are distributed among a plurality of firewall nodes using the DAG in a function-parallel manner and such that relationships between the rules specified in the DAG are preserved.

As used herein, the term "firewall" refers to a logical entity that is adapted to filter packets at the ingress and/or egress portions of a network based on a policy. The term "firewall" is intended to include systems that block packets from entering or leaving a network and systems that perform intrusion

IPR2023-01353 Page 00796

detection and intrusion protection functions for packets entering or leaving a network. Thus, the term "function-parallel firewall" is also intended to include an intrusion detection system or an intrusion protection system that operates in a function-parallel manner as described herein.

The subject matter described herein can be implemented using any combination of hardware, software, or firmware. In one exemplary implementation, the subject matter described herein may be implemented using a computer program product comprising computer executable instructions embodied in a computer readable medium. Exemplary computer readable media suitable for implementing the subject matter described herein include disk memory devices, chip memory devices, programmable logic devices, application specific integrated circuits, and downloadable electrical signals. In addition, a computer program product that implements the subject matter described herein may be located on a single device or computing platform or may be distributed across multiple devices or computing platforms.

One objective of the subject matter described herein is to provide a new security architecture, referred to herein as a hierarchical firewall, that provides the functionality of a traditional firewall (enforcing a security policy) and includes the following improvements:

1. **Suitable for high speed networks and scalable for large traffic loads.** Firewalls must continue to perform security tasks quickly and efficiently, appearing transparent to legitimate users under varying traffic loads.

2. **Maintains QoS requirements and traffic differentiation.** A growing number of network applications require QoS for their operation. Firewalls must not violate these guarantees while performing their security tasks.

3. **Greatly lessens the impact of DoS attacks.** Firewalls are candidates for DoS attacks, since they serve as the point of entry between two networks. New architectures must be capable of handling these new threats.

4. **Highly available**. Firewalls introduce a single point of failure between two networks. New firewall architectures must provide redundancy that prevents segmenting the secure network from the external network.

According to one aspect, the subject matter described herein includes a set of hierarchically configured computers that collectively enforce a desired security policy. Given this distributed configuration, traffic is quickly divided among the machines based on perceived threat. In one exemplary implementation, traffic that is considered safe may be promptly forwarded into the secure network, while the remaining traffic may be forwarded to different machines in the hierarchy for further scrutiny. Hence, traffic is segregated and queued based on security threat, yielding minimal delays for legitimate traffic. The system is also robust and highly available, since multiple machines are employed for the architecture and low cost, since Linux-installed PCs and open source software can be used for its implementation [21]. The proposed architecture is believed to be applicable to a wide variety of agencies in the public and private sectors.

The firewall architecture described herein will provide a scalable solution to varying traffic loads and QoS requirements, which are increasingly important as computing becomes more ubiquitous and applications become more diverse. The architecture is of interest to the networking community, since the architecture will result in new security policy models and optimization methods. The architecture is also relevant to the military and business community, since it provides a new, robust and cost-effective solution to security threats.

Some of the objects of the subject matter described herein having been stated hereinabove, and which are addressed in whole or in part by the subject matter described herein, other objects will become evident as the description proceeds when taken in connection with the accompanying drawings as best described hereinbelow.

BRIEF DESCRIPTION OF THE DRAWINGS

Preferred embodiments of the subject matter described herein will now be explained with reference to the accompanying drawings of which:

Figure 1 is a block diagram of a conventional data-parallel firewall;

Figure 2 is a block diagram illustrating an exemplary function-parallel firewall architecture according to an embodiment of the subject matter described herein;

Figure 3A is graph comparing simulation results for average packet delay for different firewall architectures;

Figure 3B is a graph comparing simulation results for maximum packet delay for different firewall architectures;

Figure 4A is a graph illustrating average packet delay for different packet types for a data-parallel firewall;

Figure 4B is a graph illustrating simulation results for packet delay for different packet types for a function-parallel hierarchical firewall according to an embodiment of the subject matter described herein;

Figure 5 is a data structure diagram illustrating a full trie representation of the security policy illustrated in Table 1;

Figure 6 is a data structure diagram illustrating a pruned trie representation of the security policy given in Table 1 where each packet is processed within four field comparisons;

Figure 7A is a data structure diagram illustrating a trie representing two security rules assuming that the default action per node is deny;

Figure 7B is a data structure diagram illustrating a reordered version of the trie data structure illustrated in Figure 7A;

Figure 8A is a diagram illustrating a Jackson model for a function-parallel hierarchical firewall according to an embodiment of the subject matter described herein;

Figure 8B is a diagram illustrating a Jackson model for a data-parallel firewall;

Figure 9 is a graph illustrating simulation results for average delay for different firewall architectures;

Figure 10 is a block diagram illustrating a stateful hierarchical firewall according to an embodiment of the subject matter described herein;

Figure 11A is a block diagram illustrating physical connections of a firewall grid according to an embodiment of the subject matter described herein;

IPR2023-01353 Page 00799

Figure 11B is a block diagram illustrating logical connections of the firewall grid illustrated in Figure 11A according to an embodiment of the subject matter described herein;

Figure 12 is a policy directed acyclical graph (DAG) representation of the firewall rules given in Table 2;

Figure 13A is a block diagram illustrating rule distribution among a data-parallel firewall;

Figure 13B is a block diagram illustrating a function-parallel firewall and rule distribution among nodes in the function-parallel firewall according to an embodiment of the subject matter described herein;

Figure 14A is a block diagram illustrating one exemplary rule distribution among nodes in a function-parallel firewall according to an embodiment of the subject matter described herein;

Figure 14B illustrates another distribution of rules among nodes in a function-parallel firewall according to an embodiment of the subject matter described herein;

Figure 14C illustrates an illegal rule distribution among nodes in a function-parallel firewall;

Figure 15A is a block diagram illustrating non-redundant rule distribution among nodes in a function-parallel firewall according to an embodiment of the subject matter described herein;

Figure 15B is a block diagram illustrating redundant rule distribution among nodes in a function-parallel firewall according to an embodiment of the subject matter described herein;

Figure 16A illustrates a policy DAG where deny rules are indicated by a square node;

Figure 16B is a block diagram illustrating initial gateless rule distribution among nodes in a function-parallel firewall according to an embodiment of the subject matter described herein;

Figure 16C is a block diagram illustrating final gateless rule distribution among nodes in a function-parallel firewall according to an embodiment of the subject matter described herein;

Figure 17A is a graph illustrating simulation results for average packet delay as packet arrival rate increases for different firewall designs;

IPR2023-01353 Page 00800

Figure 17B is a graph illustrating simulation results for packet delay as the arrival rate and number of rules increase for different firewall designs;

Figure 18A is a graph illustrating simulation results for average packet delay for different firewall architectures;

Figure 18B is a graph illustrating simulation results for maximum packet delay for different firewall architectures; and

Figure 19 is a flow chart illustrating exemplary steps for filtering packets using a rule set distributed among nodes in a function-parallel manner according to an embodiment of the subject matter described herein.

DETAILED DESCRIPTION OF THE INVENTION

According to one aspect of the subject matter described herein may include multiple firewall machines configured in a hierarchy, as shown in Figure 2. In the example illustrated in Figure 2, a plurality of firewall nodes **200**, **202**, **204**, and **206** collectively implement a firewall policy. Each firewall node or set of nodes implements a different level of the firewall policy in a hierarchy. For example, firewall node **200** may implement a first level of a firewall policy that includes rules that are not present in subsequent levels. If a packet passes the first level of screening, the packet may be forwarded immediately to the internal network without impacting other firewall nodes. If the packet is identified as requiring further scrutiny, the packet may be forwarded to the next level where it is screened by firewall node **202** or **204** using rules that were not applied by firewall node **200**. If the packet passes level 2 screening, the packet may be forwarded directly to the internal network. If the packet is indicated as requiring further screening, the packet will be passed to level 3 where level 3 screening is applied by firewall node **206**.

At each level in the firewall hierarchy illustrated in Figure 2, firewall nodes may implement different portions of a packet filtering rule set. An architecture in which different firewall nodes implement different portions of a firewall rule set is referred to herein as a function-parallel architecture. A function-parallel firewall architecture in which packet filtering rules are distributed among firewall nodes such that if a packet passes or fails at a

IPR2023-01353 Page 00801

certain level, the packet may bypass later levels is referred to herein as a hierarchical firewall architecture.

In the architecture illustrated in Figure 2, at each level, machines may be configured to perform a specific security task quickly and efficiently. In one implementation, the firewall rules may be distributed across the different hierarchy levels such that increased scrutiny is placed at lower levels. Since the rules are distributed among the levels, no machine in the hierarchy is required to implement the entire rule set. In contrast, a data-parallel firewall architecture requires all machines to implement the complete rule set.

As described above, in the exemplary architecture illustrated in Figure 2, packets start at the highest level and progress downward through the hierarchy. At each level, a packet can be forwarded to the internal network if deemed safe, forwarded to the next lower level for more processing, or dropped. The benefit of this divide-and-conquer approach is that traffic will incur delays consistent with its perceived security threat; therefore, traffic that is deemed safe will not be delayed by traffic that requires more processing. As a result, the hierarchical firewall can provide differentiated service and support QoS guarantees, unlike current designs. Furthermore, traffic that is forwarded to the internal network can be shaped at the particular firewall level, constraining the amount of traffic entering the network and lessening the effect of DoS attacks.

For example, in the hierarchical firewall depicted in Figure 2, packets are forwarded from the external network to the first level, where initial security checks are performed, such as a quick validity check and comparisons of IP addresses. Packets with trusted IP addresses are immediately forwarded to the internal network, while the remaining packets are sent to the next level. At the second level, packets are separated based on transport protocols and/or port numbers. Again, if a packet is deemed safe it is immediately forwarded to the secure network; otherwise, they are sent to the next level for auditing. Log files may be used at one or more of the firewall nodes indicate which connections are open and have been requested. Each level may also drop packets based on perceived threat.

The preceding example illustrates a simple hierarchical firewall; however, the size and complexity can vary. In general, the hierarchy attempts to quickly divide traffic and provide differentiated service. More levels can be used for

IPR2023-01353 Page 00802

stronger security policies and to distribute traffic more efficiently. Determining the appropriate number of levels, the number of machines per level, and how to distribute the rule set are implementation details that may be used to optimize a function-parallel firewall according to an embodiment of the subject matter described herein.

Preliminary Experimental Results

In this section, simulation is used to compare a hierarchical firewall with a traditional single-machine firewall and a data-parallel firewall [4, 22]. Each architecture implemented the same 325-rule security policy, where each rule required 1 unit of time to complete. The single firewall architecture consisted of one machine that processed each packet received from the external network. The data-parallel architecture consisted of four machines, as shown in Figure 1, each implementing the same 325-rule set. The hierarchical firewall consisted of four machines configured in three levels, as shown in Figure 2. Rules were distributed across the levels such that the first level implemented 25 rules; the second level implemented 50 rules each; and the third level implemented 75 rules. The group of rules performed at a certain machine will be referred to as a security task. The hierarchical firewall incurred an additional delay of one time unit when a packet was transmitted from machine to machine. Packets randomly arrived to the system using a Poisson distribution with a mean of 30 units and would randomly match one of the 325 rules based on a uniform distribution. For classification purposes, each packet was associated with the security task it matched (which group of rules). In this experiment, the performance parameters analyzed are the overall average packet delay, the maximum packet delay, and the average delay experienced by each packet type.

Experimental results for the different architectures are shown in Figures 3 and 4. More particularly, Figure 3A illustrates average packet delay for a single firewall design, for the data-parallel firewall design illustrated in Figure 1, and for the hierarchical firewall design illustrated in Figure 2. Figure 3B illustrates maximum packet delay for the single, data-parallel, and hierarchical firewall designs. Figure 4A illustrates average packet delay for the data-parallel firewall design for different packet types. Figure 4B illustrates average packet

IPR2023-01353 Page 00803

delay for the hierarchical firewall design for different packet types. Packet types refer to packets of a classified based on which security task was satisfied. Unlike the data-parallel firewall, the hierarchical design consistently yields different delays for each packet type.

For both delay measurements, the hierarchical architecture performed better than the single firewall and the data-parallel firewall. The single firewall had an average delay 31 times greater than the hierarchical architecture, which demonstrates how a single firewall machine can quickly become a bottleneck. Although the data-parallel firewall performed better than the single firewall, it had an average and maximum delay 6 times longer than the hierarchical architecture. In the parallel configuration, traffic that needed little processing often queued behind traffic that required more processing, which resulted in longer delays. In contrast, the hierarchical system consistently yielded better results, since traffic was distributed among the levels based on processing requirements.

The average delay experienced by each packet type is also important and is illustrated in Figures 4A and 4B. Delays in the hierarchical firewall were proportional to the amount of processing required; for example, security task 1 traffic had the shortest delay, security task 2 traffic had the second shortest delay, security task 3 traffic had the third shortest delay; and security task 4 traffic had the highest delay. Even though security tasks 2 and 3 were on the same level, security task 3 traffic had a delay slightly higher that security task 2 traffic, since it was queued with security task 4 traffic. Therefore, the hierarchical firewall is capable of providing differentiated service to multiple packet types/classes. In contrast, the delays varied for each packet type in the data-parallel firewall. Packets were constantly delayed by lower type/class traffic (e.g., security task 1 packets delayed by security task 4 packets); thus, no differentiation was achieved. Although these results are preliminary, they indicate the hierarchical architecture has merit, offering significantly better performance than current firewall systems.

IPR2023-01353 Page 00804

Refinements

The previous section described an initial hierarchical firewall architecture for employing a security policy. Although preliminary results indicate the architecture has merit, refinements can be made so that the design can achieve its full potential. One refinement relates to the hierarchical architecture. Important questions include determining the appropriate number of levels and how to effectively distribute the security policy rules. Another refinement may include providing stateful packet inspection (maintaining connection tables) and increasing redundancy. Finally, another refinement may include examining other high-speed firewall architectures, such as a dynamic firewall grid. Analytical models and simulation will be used for design and analysis; in addition, a prototype will be implemented along with analysis tools and software.

Architecture Design and Rule Set Distribution

Important questions about the firewall architecture include the number of levels, the number of machines per level, and the number of security tasks per machine. The security policy is central for addressing these questions and, for this reason, will be formally defined and modeled. The model will be optimized to reduce the physical size, minimize the traversal time, and provide differentiated service. Afterward, the optimized policy model will be used to implement the actual hierarchical firewall.

One method for designing the hierarchy is to represent the security policy graphically as a tree data structure. This representation will aid the translation to the actual firewall, since it has a hierarchical form. Although a variety of tree data structures exist, a trie will be initially investigated [1]. A trie is a special type of data structure that has the advantages of fast retrievals and quick unsuccessful search determination, which are important for enforcing a security policy. In a security policy trie, each path from the root node to a leaf corresponds to one rule. The nodes of the trie correspond to the rule fields, while branches are possible values. Figure 5 is a trie data structure for the rules in Table 1. For example, in Figure 5, the root node is the protocol field. The next level represents the source IP address, while the leaf nodes represent

IPR2023-01353 Page 00805

the rule action. Rules that have the same field value at a node will share a branch for that value.

The trie in Figure 5 referred to as full trie, since every noncycle path from the root to a leaf has the same length, and every leaf lies at the same depth [6]. Unlike other policy models, the rule order must be preserved if the trie is to be successfully translated to a hierarchical firewall. The action to apply to a packet is determined by a preorder traversal. Starting at the root node, branches are compared to the fields of the packet. Since multiple branches may match a packet, they are traversed in order from left to right. If a packet field does not match any possible branches, it moves back to the previous node (parent), where the next branch is tested. Once the policy trie is developed, it can be translated into a hierarchical firewall. The nodes of the trie represent the machines in the hierarchy, while the branches represent the links between nodes. The value associated with each branch indicates the packet field value required to traverse it. Although a full trie representation of a security policy can be created, it is not necessarily optimal. Further trie manipulation is required create an effective hierarchical firewall, such as preventing routing to parents and limiting trie traversal time.

Pruning yields a trie that never requires routing to parent nodes, which provides faster packet processing [1, 15]. Assuming a full trie consists of n nodes and k levels (where k <= n), the worst case packet processing would be O(n*k) tuple-compares. In contrast, a pruned trie requires only O(k + n) tuple-compares in the worst case which is a processing reduction of 1/k.. Pruning is accomplished by replacing general rules with descendants. Rule r_1 is a descendant of rule r_2, if $r_2[i]$ is a prefix of $r_1[i]$ for $i = 1, ..., k$. Therefore, pruning occurs by pushing a rule down to all its descendents, then deleting the rule. An example of a pruned version of the trie illustrated in Figure 5 is given in Figure 6, where [*, *, *, deny] is a general rule that can be pushed down to every other rule. Although the original branch has been removed and added to the other branches, the order is preserved, if branches are tested from left to right. Current pruning algorithms do not consider rule order, since tries were primarily developed for storing data [6]. New pruning techniques that preserve order may be developed and used to distribute rules among nodes in a functional-parallel firewall according to an embodiment of the subject matter described

IPR2023-01353 Page 00806

herein. Pruning may also cause the number of nodes in the hierarchy to increase, since a general rule is duplicated at different levels. For this reason, other techniques are needed to reduce the trie size.

Reducing the trie size is important, since delay increases each time a packet is forwarded from one node to another. The trie can be reduced by compressing nodes, which is achieved by removing single branching paths or consolidating sub-ties into one node [2, 6, 15]. As a result, a node may compare multiple rule fields or perform multiple security tasks per packet. For example, in Figure 5, the rule [UDP, 150.*, *, 3030, accept] is considered a leaf chain [6] and can be compressed into one node (a single machine). Two other methods commonly used to reduce the size of a trie are Directed Acyclic Graphs (DAG) and level reordering [1, 6, 15]. If the same sub-trie appears in multiple places, then the trie structure can be changed to a DAG. The possible use of DAG is seen in Figure 6, where nodes labeled $Port^a$ and $Port^b$ are equivalent sub-tries. Parent nodes of both sub-tries could point to just one, eliminating the need for the other.

Another method for reducing the trie size is level reordering. As seen in Figures 7A and 7B, the order of the levels can significantly affect the number of nodes in the trie. More particularly, in Figure 7A, an original trie representing the security rule [TCP, 120.*, 210.*, 90, accept] and [TCP, 130.*, 210.*, 90, accept] is illustrated. Both Figures 7A and 7B assume that the default action per node is deny. Since some of the rules in the original trie in Figure 7A include common elements, the rules can be reordered so that the number of branches in the subtrie are reduced. For example, in Figure 7A, the destination IP address and port for each branch of the subtrie are the same. Accordingly, if these common portions of the rule are implemented at higher levels, the duplicate nodes in the trie can be reduced, as illustrated by Figure 7B. More particularly, in Figure 7B, the nodes for destination IP and port are placed before the node for source IP address. As can be seen from Figure 7B, reordering the levels produces a smaller trie. Using a DAG can also reduce the number of nodes.

The problem of finding an ordering that guarantees a minimum size trie has been shown to be NP-complete; however, several heuristics have been developed [2, 6]. It should be noted that DAGs and reordering only reduce the

IPR2023-01353 Page 00807

number of nodes in the hierarchy and do not reduce the depth of the trie, which influences the traversal time.

Regardless of the applicability of these different trie modifications, they must be performed in concert to achieve an optimal hierarchical design. For example, the architecture utilizes multiple nodes to distribute traffic load and provide differentiated service; however, the trie path length (depth) impacts the packet traversal time. Therefore, compression and ordering must be performed together to provide differentiated service and optimize packet delay.

To achieve these goals, information not explicitly given in the security policy (e.g., QoS, level of trust, and processing time) may be used. For example, it is advantageous to place security tasks that require the most time at the lowest levels. This prevents packets from queueing behind traffic that requires more processing time; thus, lowering delay. A weight w_i can be assigned to each security task s_i (a single rule or multiple rules) based on the amount of time it requires. For example, an auditing security task requires more time (updating system files) than those that only perform field comparisons. Using these weights, the trie can be ordered to minimize the average traversal time. For example assuming i and j are hierarchy levels, where $i < j$, the trie can be ordered such that the average weight per level increases from the highest to the lowest level $\overline{w}_i < \overline{w}_j, \forall i \neq j$. In addition, the trie can be designed to separate traffic based on QoS or level of trust. Traffic would traverse different sub-tries, based on the type of service required, providing class isolation and different delays. The algorithms described in this section can then be applied to these sub-tries independently to reduce size and traversal time. These ideas were used to design the hierarchical firewall simulated in the preceding section, which showed excellent results. New algorithms for trie compression based on traversal time with constraints on QoS will be developed for optimizing function-parallel firewall design.

Packet traversal time remains a key performance metric for firewall systems. For this reason, queueing models will be used to analyze the delay of different firewall designs. The node delay encountered by a packet consists of the processing time, queueing delay, and transmission time. The processing delay is the time required to perform the security task. Assume node i must

IPR2023-01353 Page 00808

process R_i rules per packet and can perform xi rules per unit time, then the service rate is x_i/R_i packets per unit time. Queueing is the delay associated with waiting for processing and depends on the system state. The transmission delay is the time required to transmit the packet, which depends on the packet's length and the line speed. The maximum delay experienced by a packet traversing the firewall is the sum of the node delays along the path from the root to a leaf.

If it is assumed the packet arrival and processing (transmission and security task) rates are exponentially distributed, the node can be modeled as an M/M/1 queue [5, 17]. The firewall can then be considered an open network of M/M/1 queues, where probabilities are assigned to each link to indicate the likelihood of moving to the next node. This is a Jackson network model, which has a product-form solution for the average delay [5, 17]. The average end-to-end delay for N cascading nodes (traversal path) can be computed as

$$E(T) = \sum_{i=1}^{N} \frac{1}{\mu_i - \lambda_i} \tag{1}$$

where $1/\mu_i$ is the service time (processing and transmission) and λ_i is the arrival rate to node i. Figures 8A and 8B respectively illustrate Jackson models for hierarchical and data-parallel firewalls. In Figures 8A and 8B, it is assumed that the total arrival rate to each system is λ packets per unit time, each system implements the same R rule security policy, and each node can perform x rules per unit time. For the hierarchical firewall, shown in Figure 8A, the policy rules are evenly distributed to each node; therefore, the service rate is $3x/R$ per node. The traffic arrival rate to node A is λ packets per unit time, half of the packets are forwarded to node B, while the other half is forwarded to node C. Although not considered in this example, some traffic may be forwarded into the secure network or dropped by node A; therefore in this case, the sum of the rates to nodes B and C would be less than λ. Using these values, the average end-to-end delay for either path in the hierarchical firewall is

$$E_h(T) = \frac{1}{\frac{3x}{R} - \lambda} + \frac{1}{\frac{3x}{R} - \frac{\lambda}{2}} \tag{2}$$

IPR2023-01353 Page 00809

For the data-parallel configuration, illustrated in Figure 8B, traffic arrives at the packet distributor, which evenly distributes the traffic to each node. Each node implements the complete rule set R; therefore, the service rate is x/R. The end-to-end delay across any node in the data-parallel firewall is

$$E_p(T) = \frac{1}{\frac{x}{R} - \frac{\lambda}{3}} \quad (3)$$

Using these models, the effect of increasing traffic load λ on the average end-to-end delay, for the two architectures is shown in Figure 9. For this example, the arrival rate λ ranged from 1 to 50 packets per unit time. The rule set R consisted of 240 rules, while x equaled 5000 rules per unit time. Each of the hierarchical nodes incurred an additional delay of five time units to compensate for inter-node transmission. As expected, the average delay increases as the network load increases for both architectures; however, the average delay is consistently lower for the hierarchical firewall. This is due to the decreased rule set per node and the reduced arrival rates for nodes B and C. The average is an upper bound for the hierarchical firewall. Allowing node A to forward packets to the secure network or to drop packets would reduce the load on the second level and decrease delays. The arrival rate is less per node in the data-parallel firewall; however, the system has a higher end-to-end delay because each node must implement the complete rule set. The theoretical results are consistent with the experimental results described above. These models provide an important tool for analyzing the effect of load and rule distribution on end-to-end delay. In addition to analytical results, simulation will be used to measure the performance of the hierarchical designs under various conditions.

Stateful Firewalls

Another aspect of the subject matter described herein includes a stateful hierarchical firewall. The preceding design is considered stateless, or static, since the security action taken is based solely on the packet currently inspected. A stateful firewall provides better security by tracking and controlling network flows (packets belonging to a particular network connection or

IPR2023-01353 Page 00810

exchange) [16, 21, 22]. In one exemplary implementation, a stateful firewall can be achieved by storing state information concerning each flow in a table, and then according it with the security policy. For example, each time a TCP connection is established from the internal network to the external network, it is logged in a table. The table may contain the source and destination addresses, port numbers, sequencing information, and connection flags. Afterwards, packets arriving from the external network claiming to be part of an existing TCP connection can be checked against the table. If a packet belongs to an existing connection or exchange, it can be accepted. Otherwise, it is dropped. Table entries expire, if packets belonging to an exchange are not observed by the firewall over a period of time [21].

Therefore, a stateful firewall is dynamic and limits the hackers' opportunity to exploit certain services. The increased security does require additional memory and time for table manipulation, which is problematic for high-speed networks. Furthermore, the firewall must process all traffic arriving and departing the secure network to adequately maintain the connection tables 21]. To solve these problems, the proposed firewall system will be divided into two hierarchies, one for arriving traffic and another for departing traffic, as illustrated in Figure 10. In Figure 10, a stateful firewall system includes first and second hierarchical firewall subsystems **1000** and **1002** and a control node **1004**. First and second hierarchical firewall subsystems **1000** and **1002** may be function-parallel firewall systems including a plurality of firewall nodes with different rules distributed among different nodes as described above. In one exemplary implementation, hierarchical firewall subsystem **1000** may include at least two firewall nodes that implement different portions of a first firewall rule set. Similarly, hierarchical firewall subsystem **1002** may include at least two firewall nodes that implement different portions of a second firewall rule set. Firewall subsystem **1000** may filter packets entering the network. Firewall subsystem **1002** may filter packets departing from the network. Firewall subsystems **1000** and **1002** may share state information regarding connections established through the nodes within each system. Control node **1004** may store the state information and may be accessible by the nodes in hierarchical firewall subsystems **1000** and **1002.** Rules can be distributed to the

IPR2023-01353 Page 00811

hierarchies and the designs can be optimized using the methods described in preceding section.

However, a new problem with the stateful model is the need to distribute state information across nodes. For example, nodes that scrutinize arriving traffic must be aware of any new connections permitted by nodes that inspect departing traffic. Furthermore, nodes must be informed of table entries that expire over time. To communicate state information, a central database machine, such as control node **1004**, may collect and distribute the information (table entries) to the appropriate nodes. In one exemplary implementation, control node **1004** may not perform table queries on behalf of nodes. Individual nodes will only query their local tables, which prevents the central machine from becoming a bottleneck. In addition, the central machine will only send information (table entries) to nodes that might require it. As a result of this policy, the table searched by an individual node is reduced (contains only relevant information) as well as the search time. In an alternate implementation, control node **1004** may store the connection table and may perform queries for packets received by firewall subsystems **1000** and **1002**.

Survivability and Firewall Grids

Another aspect of the subject matter described herein focuses on increasing the survivability of the hierarchical design. Although the proposed design has the benefit of increased packet throughput and minimal delay, the reliability of the system is decreased, since more machines (nodes) are deployed. For example, if the highest level node fails (is unable to inspect more traffic or goes off-line), then the entire hierarchy is unable to process packets. For this reason, the hierarchy is considered an n of n system, where all *n* machines must be functioning for the system to operate correctly [4, 12].

Survivability of the system can be improved with the use of redundant (parallel) nodes at different levels. For example, the first level may include multiple machines in parallel, where each machine implements the same security task (e.g., an IP check). An arriving packet would be sent to only one of the parallel machines, where it would be accepted, dropped, or forwarded to the next level for further processing. If one of the parallel nodes fails, then the

IPR2023-01353 Page 00812

remaining nodes can process the traffic. An additional benefit of parallel processing is the equal distribution of traffic among nodes, which can result in higher throughput and lower delays. Details such as where to locate parallel nodes and the number of machines required for each parallel configuration may depend on the desired degree of reliability versus cost of deploying redundant machines.

Yet another aspect of the subject matter described herein includes a grid of configurable firewalls (nodes) that can be used to efficiently process traffic, while being highly available. In one exemplary implementation, the grid may include a network of highly interconnected nodes (possibly via a high-speed switch or router) that can be logically deployed in various configurations (e.g., parallel, hierarchical, or hybrid). The advantage of the firewall grid is its ability to quickly adapt to new demands. When security needs change (e.g., due to policy changes, increased traffic volume, or node failure), these machines can be deployed or configured to maintain throughput and QoS requirements. For example, in a logical hierarchy, if a higher level encounters an increased traffic load, less frequently used redundant machines at lower levels can be reconfigured to process packets at the higher level. Unlike the hierarchical model, physical changes are not required, since all the machines are interconnected. Only the logical connection between nodes must be updated, which can be performed quickly. Like the stateful hierarchical firewall, the initial grid model will use a central machine to coordinate node configuration and communication. This central machine will play an important role in the system (balancing processing demand per node) and may constrain the efficiency of the firewall grid. Future versions of the firewall grid will attempt to place more intelligence in the nodes and allow them to interact independently and adapt dynamically under the guidance of a general security policy.

Figures 11A and 11B illustrate an example of a firewall grid according to an embodiment of the subject matter described herein. Referring to Figure 11A, a plurality of firewall nodes **1100**-**1114** are connected via a control node **1116**. Each firewall node **1100**-**1114** may be configured to implement different portions of a firewall rule set, as controlled by control node **1116**. Each of the firewall nodes is physically connected to the other firewall nodes via physical

IPR2023-01353 Page 00813

connections **1118**. Physical connections **1118** may be local area network connections, such as Ethernet connections. As such, control node **1116** may be an Ethernet switch.

In Figure 11B, firewall nodes **1100-1114** are connected via logical connections **1120**. Logical connections **1120** may be implemented by control node **1116** without changing physical connections **1118**. For example, control node **1116** may configure the output port to which firewall **1106** is connected to receive frames from firewalls **1100**, **1102**, and **1104**. Similar internal configurations can be implemented for the remaining firewalls illustrated in Figure 11B. In Figure 11B, logical connections **1120** between the firewalls implement a hierarchical, function-parallel firewall design, such as that illustrated in Figure 1. In addition, the design illustrated in Figure 11B is also data-parallel at each level. For example, each firewall node at each level may implement a common rule or rule set. For example, the rule set implemented by the firewall nodes in level 1 is illustrated in Figure 11B as R1. The firewall nodes in level 2 implement a rule set represented by R2, and the firewall nodes in level 3 implement a rule set represented by R3. If any of the firewall nodes in a given level fails, the remaining firewall nodes in that level may take over operation for the failed firewall. Alternatively, as described above, if a failure or a bottleneck occurs at one level, a less-used firewall at another level may be logically configured to operate at the level where the failure or congestion occurred. Packet distributors **1122** may be configured to distribute the firewall rules between the firewalls at each level. Thus, as illustrated by Figures 11A and 11B, logical connections between firewall nodes and a firewall grid may be dynamically altered (i.e., changed on the fly without ceasing packet filtering operation) by a controller node without changing the physical connections between the firewalls.

Firewall Implementation

In addition to the analytical models and simulations described above, an implementation of a hierarchical firewall to measure performance will now be described. One exemplary implementation may utilize Linux-based PCs and Ethernet connections. Each PC will contain multiple Ethernet cards and be

IPR2023-01353 Page 00814

loaded with the Linux 2.4 kernel. The 2.4 kernel has support for various packet queueing disciplines, traffic control methods, and packet processing (e.g., iproute2, tc, and iptable). Collectively, these functions are required for advanced packet filtering. For example, iproute2 and iptable can route packets based on information other than the destination address, while tc can mark and shape traffic. PCs will serve as hierarchical nodes, sources and receivers of network traffic, and hierarchy controllers.

Results from an initial implementation of a three-node hierarchical firewall indicate that these open-source components are suitable for high-speed networks. The Computer Science Department at Wake Forest University has 10 PCs available for this project, where each has a 600 Mhz processor and multiple 100 Mbps Ethernet cards. These machines will serve as an initial testbed for system development; however, faster machines and high-speed network connections are required for actual system testing. For this reason, this project requests the purchase of 6 additional PCs (2.4 Ghz processors), each with multiple Gigabit Ethernet cards (3 cards per machine). These new machines will be configured as the firewall, while the current machines will serve as traffic generators and receivers. For comparison purposes, parallel and single-machine firewalls will also be implemented using this equipment. Any software required will be developed and provided by the Computer Science Department.

Firewall Rule Distribution for Function-Parallel Firewall Design

As stated above, one aspect with creating an optimal function-parallel firewall design includes distributing rules among firewalls in the firewall rule set. Firewall rules are preferably distributed among firewall nodes in a manner that preserves firewall policy integrity. The following sections describe an exemplary firewall rule set and different options for distributing the firewall rule set among firewall nodes in a function-parallel firewall design.

Firewall Policy

In this section and the following sections, a rule r is modeled as an ordered tuple of sets, $r = (r[1], r[2], ..., r[k])$. Order is necessary among the tuples since comparing rules and packets requires the comparison of

IPR2023-01353 Page 00815

corresponding tuples. Each tuple $r[l]$ is a set that can be fully specified, specify a range, or contain wildcards '*' in standard prefix format. For the Internet, security rules are commonly represented as a 5-tuple consisting of: protocol type, IP source address, source port number, IP destination address, and destination port number [21, 22]. For example, the prefix 192.* would represent any IP address that has 192 as the first dotted-decimal number. Given this model, the ordered tuples can be supersets and subsets of each other, which forms the basis of precedence relationships. In addition to the prefixes, each filter rule has an action, which is to accept or deny. However, the action will not be considered when comparing packets and rules. Similar to a rule, a packet (IP datagram) d can be viewed as an ordered k-tuple $d = (d[1], d[2], ..., d[k])$; however, ranges and wildcards are not possible for any packet tuple.

Using the previous rule definition, a standard security policy can be modeled as an ordered set (list) of n rules, denoted as $R = \{r_1, r_2, ..., r_n\}$. A packet d is sequentially compared against each rule r_i starting with the first, until a match is found ($d \Rightarrow r_i$) then the associated action is performed. A match is found between a packet and rule when every tuple of the packet is a subset of the corresponding tuple in the rule.

Definition Packet d matches r_i if

$$d \Rightarrow r_i \text{ iff } d[l] \subseteq r_i[l],\ l = 1, ..., k$$

The rule list R is comprehensive if for every possible legal packet d a match is found using R. Furthermore, two rule lists R and R' are equivalent if for every possible legal packet d the same action is performed by the two rule lists. If R and R' are different (e.g. a reorder) and the lists are equivalent, then the policy integrity is maintained.

As described above, a rule list has an implied precedence relationship where certain rules must appear before others if the integrity of the policy is to be maintained. Table 2 shown below illustrates an exemplary rule set that will be used herein to illustrate precedence relationships and rule assignment according to an embodiment of the subject matter described herein.

No.	Proto.	Source IP	Source Port	Destination IP	Destination Port	Action	Prob.
1	UDP	1.1.*	*	*	80	deny	0.01
2	TCP	1.*	*	1.*	90	accept	0.02
3	TCP	2.*	*	2.*	20	accept	0.25
4	UDP	1.*	*	*	*	accept	0.22
5	*	*	*	*	*	deny	0.50

Table 2: Example Security Policy Consisting of Multiple Ordered Rules

In Table 2, rule r_1 must appear before rule r_4, likewise rule r_5 must be the last rule in the policy. If for example, rule r_4 was moved to the beginning of the policy, then it will shadow [23] the original rule r_1. Shadowing is an anomaly that occurs when a rule r_j matches a preceding rule r_i, where $i < j$. As a result of the relative order r_j will never be utilized. However, there is no precedence relationship between rules r_1, r_2, or r_3 given in Table 2. Therefore, the relative order of these three rules will not impact the policy integrity and can be changed to improve firewall performance.

As described in [28], the precedence relationship between rules in a policy can be modeled as a Policy Directed Acyclical Graph (DAG). Let $G = (R,E)$ be a *policy DAG* for a rule list R, where vertices are rules and edges E are the precedence relationships (constraint). A precedence relationship, or edge, exists between rules r_i and r_j, if $i < j$ and the rules intersect [28].

Definition The intersection of rule r_i and r_j, denoted as $r_i \cap r_j$ is

$$r_i \cap r_j = (r_i[l] \cap r_j[l]),\ l = 1, \ldots, k$$

Therefore, the intersection of two rules results in an ordered set of tuples that collectively describes the packets that match both rules. The rules r_i and r_j intersect if every tuple of the resulting operation is non-empty. In contrast, the rules r_i and r_j do not intersect, denoted as $r_i \not\cap r_j$, if at least one tuple is the empty set. It should be noted that the intersection operation is symmetric; therefore, if r_i intersects r_j, then r_j will intersect r_i. The same is true for rules that do not intersect.

For example, for the rules given in Table 2, the intersection of r_1 and r_4 yields (UDP, 1.1.*, *, *, 80). Again, the rule actions are not considered in the

IPR2023-01353 Page 00817

intersection or match operation. Since these two rules intersect, a packet can match both rules for example d = (UDP, 1.1.1.1, 80, 2.2.2.2, 80). The relative order must be maintained between these two rules and an edge drawn from r_1 to r_4 must be present in the DAG, as seen in Figure 12. In Figure 12, the policy DAG represents the rules in Table 2. Vertices **1200** indicate rules while edges **1202** indicate precedence relationships. Rules r_2 and r_3 in Table 2 provide an example of rules that can be re-ordered without changing the firewall policy. These two rules do not intersect due to the fifth tuple (destination port). A packet cannot match both rules indicating the relative order can change; therefore, an edge will not exist between them.

As described above, traditional firewall implementations consist of a single, dedicated machine, similar to a router, that applies the rule set to each arriving packet. However, packet filtering represents a significantly higher processing load than routing decisions [15, 18, 22]. For example, a firewall that interconnects two 100 Mbps networks would have to process over 300,000 packets per second [21]. Successfully handling these traffic loads becomes more difficult as rule sets become more complex [4, 14, 22]. Furthermore, firewalls must be capable of processing even more packets as interface speeds increase. In a high-speed environment (e.g. Gigabit Ethernet), a single firewall can easily become a bottleneck and is susceptible to DoS attacks [4, 7, 10, 11]. Building a faster single firewall is possible [8, 15, 18, 19, 20]; however, the benefits are temporary (traffic loads and interface speeds are increasing); it is not scalable; it is a single point of failure; and it is generally not cost-effective for all installations.

Parallelization has been demonstrated to offer a scalable approach for improving the performance of network firewalls. As seen in Figure 13A, parallel designs consist of multiple firewalls, referred to as firewall nodes, connected in an array. In Figure 13A, firewall nodes **1300** each implement a common rule set. A packet distributor **1302** load shares packets among firewall nodes **1300**. In Figure 13B, a function-parallel design is illustrated where firewall nodes **1304** implement different portions of a firewall rule set. A packet duplicator **1306** duplicates each packet to each firewall node. A gate **1308** controls packets entering the network based on control information received from firewall nodes

1304. The operation of the system illustrated in Figure 13B will be described in more detail below.

The two designs depicted in Figures 13A and 13B are different based on what is distributed: packets or rules. For example, the design developed by Benecke et al consisted of multiple identical firewall nodes connected in parallel, as shown in Figure 13A. Each firewall node implements the complete security policy *R*, and arriving packets are distributed across the firewall nodes for processing (one packet is sent to one firewall node) [4]; therefore different packets are processed in parallel. Using terminology developed for parallel computing, this design is considered data-parallel since the data (packets) is distributed across the firewall nodes [25]. How the load-balancing algorithm distributes the packets is vital to the data-parallel system and typically implemented as a specialized high-speed switch [11, 12]. Although data-parallel firewalls have been shown to achieve higher throughput than traditional firewalls and have a redundant design [4], they suffer from two major disadvantages. First, these designs have difficulty maintaining QoS across networks. For example, legitimate traffic can encounter delays if it is queued behind traffic that requires more processing. Under these circumstances, users notice poor network performance, which is a growing concern as more network applications require QoS assurances. Second, stateful inspection requires all traffic from a certain connection or exchange to traverse the same firewall node, which is difficult to perform at high speeds using the data-parallel approach [4, 14].

Rule Distribution and Function-Parallel Firewall

As described above, new scalable firewall designs are needed to cope with increasing traffic loads, network speeds, and security threats. To address these issues, one implementation of the subject matter described herein includes a firewall design that consists of multiple firewalls connected in parallel and a gate machine, as shown in Figure 13B. Again in this design, the individual firewalls will be referred to as firewall nodes. However unlike the data-parallel design that distributes the packets, the function-parallel design distributes the policy rules across the firewall nodes. The distribution is done such that policy integrity is maintained. Therefore, a single traditional firewall

IPR2023-01353 Page 00819

and a function-parallel firewall will always give the same result for any packet. The rules distributed to a firewall node will be referred to as the local rule set or local policy. Using parallel computing terminology this design will be referred to as function-parallel, since the rules are distributed across the firewall nodes [25].

The general operation of one exemplary function-parallel system can be described as follows. When a packet arrives to the function-parallel system, the packet is forwarded to every firewall node. Each firewall node processes the packet using its local rule list. The firewall node then signals the gate indicating either no match was found or provides the rule number and action if a match was found. Therefore, no match is a valid response and is required for this basic function-parallel design. The gate keeps track of the results and determines the final action to perform on the packet. For example, a first match policy can be implemented by applying the action of the lowest numbered matching rule to the packet.

The performance of this basic design can be improved by allowing the firewall nodes to pipeline packets [25]. For example, once a firewall node has completed processing a packet, it can immediately begin processing the next packet. This allows the firewall nodes to work independently, thus increasing the overall system performance. However, pipelining requires the gate to keep track of multiple packets simultaneously. Performance can also be increased if the gate can signal the firewall nodes that further processing of a certain packet is no longer needed (similar to short-circuit evaluation). This occurs when the appropriate match has been found by a firewall node before the other firewall nodes have completed processing the same packet. Short-circuit evaluation requires the gate to know how the rules are distributed as well as the dependencies, which is discussed in detail below.

The function-parallel design has several significant advantages over traditional and data-parallel firewalls. First, the function-parallel design results in the faster processing since every firewall node is utilized to process a single packet. Reducing the processing time, instead of the arrival rate (the data-parallel paradigm), yields better performance that will be proven theoretically and demonstrated experimentally. Second, unlike the data-parallel design, function-parallel can maintain state information about existing connections.

IPR2023-01353 Page 00820

Maintaining state can be viewed as the addition of a new rule corresponding to a requested connection. Unlike the data-parallel design, the new rule can be placed in any firewall node since a packet will be processed by every firewall node. Third, the function-parallel design has the unique ability to provide service differentiation between classes of traffic; therefore, the design can provide QoS requirements. Finally, the function-parallel system can be implemented using any current firewall system (as discussed below, the gate can be removed), without any specialized hardware. The disadvantage of the system is the limited scalability, since the system cannot have more firewall nodes than rules, unless data-redundancy is provided. However, given the size of most firewall policies range in the thousands to hundred of thousands of rules, which is expected to increase, the scalability limit is not an important concern.

The operation and performance of the function-parallel design depends on the correct distribution of policy rules. Rule distribution techniques that maintain integrity, increase processing speed, allow redundancy and robustness, and eliminate the gate are described in the following sections.

Rule Distribution and Policy Integrity

As described above, the function-parallel design distributes the security policy rules across an array of firewall nodes. Using this design, all the firewall nodes process an arriving packet in parallel using its local rule set. The rules must be distributed such that the integrity of the policy is maintained. Again, it will be assumed in the examples described herein that a first-match policy is desired. Therefore, the distribution of the rules must not introduce any anomalies into the original rule list, such as shadowing.

Let m be the number of firewall nodes in the system and j be a firewall node $j = 1...m$. Each firewall node has a local rule set that is a portion of the security policy. Denote the local rule list for firewall node j as R_j that has n_j rules. The firewall node will process an arriving packet using the local rule list in a top-down fashion to find the first match. It should be noted that a software implementation of the firewall node according to the subject matter described herein is not required to have a rule list. The function-parallel design is independent of software implementation. In this scenario integrity is maintained

IPR2023-01353 Page 00821

under two conditions: every rule is assigned to at least one firewall node, and a policy DAG edge never traverses upward within a firewall node. The first condition assures every rule in the original policy exists in the function-parallel system, while the second condition prevents rule shadowing within a firewall node. Consider a two-firewall node system and the five-rule policy given in Table 2. Possible rule distributions are depicted in Figures 14A-14C. The distribution illustrated in Figure 14A will be referred to as vertical, while the distribution illustrated in Figure 14B will be referred to as horizontal. In both cases the five rules are present and all policy DAG edges are downward; thus integrity is maintained. In contrast, Figure 14C is an illegal distribution because rule r_4 shadows r_1 (upward policy DAG edge).

The gate can also utilize the policy DAG to determine the first match, and if a short-circuit evaluation can occur. For example consider the distribution illustrated in Figure 14A. Assume a packet matches r_3 in the second firewall node and this result is reported to the gate. Using the policy DAG, the gate can determine this is the first match, since it is impossible for a packet that matches r_3 to also match any rules that preceded it in the original policy (rules r_1 or r_2). The gate can inform the remaining firewall nodes to stop processing the packet.

Optimal Rule Distribution

The previous section described the distribution requirements for maintaining policy integrity. However it was proven in [27] that any linear arrangement of a policy DAG maintains integrity, which also holds for a local policy DAG. Furthermore as evident in the previous section, many different distributions are possible. Given the possible rule distributions for the function-parallel design, a distribution is sought that minimizes the average number of comparisons required per packet [27].

Achieving this goal will require information not present in the firewall rule list. Certain firewall rules have a higher probability of matching a packet than others. As a result, it is possible to develop a policy profile over time that indicates frequency of rule matches (similar to cache hit ratio). Let $P = \{p_1, p_2, \ldots, p_n\}$ be the policy profile, where p_i be the probability that a packet will match rule r_i (first match in the policy). Furthermore, assume a packet will always find

IPR2023-01353 Page 00822

a match, $\sum_{i=1}^{n} p_i = 1$; therefore R is comprehensive. Using this information, the average number of rule comparisons required for the policy R using a standard single machine firewall is

$$E(R) = \sum_{i=1}^{n} i \cdot p_i \quad (4)$$

This equation also applies to firewall nodes. However, the rules of a local policy include rules of the original policy; therefore, a mapping between the local and original policy rules is needed. Let $r_{i,j}$ represent the ith rule of firewall node j, that corresponds a certain rule in the original rule list R. Consider the local rule list for the second firewall node depicted in Figure 14A. The local policy is $R_2 = \{r_4, r_5\}$, where $r_{1,2} = r_4$ and $r_{2,2} = r_5$. Similarly, let $p_{i,j}$ be the probability of matching the ith rule of firewall node j. Therefore the average number of comparisons required for firewall node j is

$$E(R_j) = \sum_{i=1}^{n_j} i \cdot p_{i,j} \quad (5)$$

Using the previous equation, the average number of comparisons required for the function-parallel system is

$$E_f(R) = \sum_{j=1}^{m} E(R_j) = \sum_{j=1}^{m} \sum_{i=1}^{n_j} i \cdot p_{i,j} \quad (6)$$

For example, the average number of comparisons required for the distribution given in Figure 14A is 2.24, while the average number of comparisons for the distribution given in Figure 14B is 2.45.

Given a policy DAG $G = (R,E)$ and policy profile P, a distribution is sought that minimizes Equation 6. In the absence of precedence relationships, the average number of comparisons is minimized if the rules are sorted in non-increasing order according to the probabilities [28], which is also referred to as Smith's algorithm [29]. The sorted rules are then distributed in a horizontal fashion, as depicted in Figure 14B. Precedence constraints cause the problem

IPR2023-01353 Page 00823

to be more realistic; however, precedence constraints also make determining the optimal permutation more problematic. In [27] determining the optimal rule order for a single machine firewall is proven to be *NP*-complete; therefore, so is determining the optimal function-parallel rule distribution. However, [27] presents a simple sorting algorithm to improve the optimal order, that can be modified to apply to function-parallel rule distribution. This modified algorithm sorts the policy rules using the method described in [27], then distributes the rules in a horizontal fashion.

Redundancy

Similar to the traditional and data-parallel firewalls, a disadvantage of the current function-parallel design is its inability to withstand a single firewall node failure. If a single firewall node fails in the function-parallel system, then part of the policy is lost and integrity is not maintained. A simple solution commonly used for traditional firewalls is to duplicate the entire system; however, this solution is cost prohibitive, not efficient, and difficult to manage.

Redundancy can be provided in the function-parallel design by replicating rules, instead of firewall nodes. Consider the function-parallel system depicted in Figure 15A, that includes four firewall nodes and an eight-rule policy distributed in a horizontal manner. If the local rule list of the right neighbor is added to the end of every firewall node, as illustrated in Figure 15B, then any non-continuous firewall nodes can fail and integrity will be maintained. It should be noted that the criteria for maintaining integrity still apply: every rule must be implemented at a node and no upward edges in the local policy DAG. Furthermore, the redundant design requires short-circuit evaluations to prevent multiple firewall nodes from evaluating the same rule.

Gateless Design

The gate plays an important role in the function-parallel design. It accepts information from the firewall nodes to determine the correct action to apply to a packet. Unfortunately, the gate is a special device that must coordinate with the individual firewall nodes in the system, which may be problematic to implement. However, it is possible to eliminate the gate and allow the firewall nodes to operate independently without any coordination. The

IPR2023-01353 Page 00824

significant advantage is that the gateless function-parallel design can be implemented with any firewall system (from open source to proprietary).

Eliminating the gate requires a rule distribution such that for any given packet, only one machine will ever accept and the remaining machines will always deny. As described above, rule actions are either accept or deny. This can be seen in the policy DAG by denoting deny rules with a square and accept rules with a circle. To assure only one machine will accept a packet, an accept rule in a firewall node cannot intersect with another accept rule in any other firewall node. As described above, the intersection of two rules describes the set of packets that can match both rules. If an edge spans the local rules of firewall nodes, then the set of packets would be accepted by both firewall nodes; thus, duplicate packets would be accepted. In addition, every drop rule must be present in each firewall node to assure the remaining firewall nodes will drop the packet. Given these two requirements a policy DAG edge will not span firewall nodes, as illustrated in Figures 16B and 16C.

For example, consider a two-firewall node function-parallel system that implements the five-rule policy given in Table 2. The policy DAG is given in Figure 16A, where the first and last rules are deny. An initial gateless rule distribution is given in Figure 16B which has no edges spanning the firewall nodes. Given the local policy DAG, deny rules can be removed if: no inbound edges exist for the deny rule; and all out-going edges for the deny rule connect to a superset deny rule. Using these constraints, the first rule in the second firewall node depicted in Figure 16B can be removed (it is a subset of r_5 and has no other edges). This gives the final gateless rule distribution shown in Figure 16C. As a result, the first firewall node in this distribution will only process UDP traffic, while the second machine only processes TCP. The rules can be divided with respect to other tuples, and can be done to provide QoS guarantees.

Theoretical Results

Assuming arrivals and service times are exponentially distributed, a firewall system can be considered an open network of M/M/1 queues (Jackson network) [5, 17]. Probabilities can be assigned to each link to indicate the likelihood of moving to the next node, which are given from the policy profile (hit

IPR2023-01353 Page 00825

ratio) described above. The average end-to-end delay for q cascading machines (traversal path) can be computed as

$$E(T) = \sum_{i=1}^{q} \frac{1}{\mu_i - \lambda_i} \tag{7}$$

where $1/\mu_i$ is the service time (processing and transmission) and λi is the arrival rate to node i. As a result, we have a theoretical model for the average delay across a firewall system. Consider the data-parallel and functional-parallel firewall designs given in Figures 13A and 13B. Assume each system consists of m machines and implements the same n rule security policy. Let the total arrival rate to each system be λ packets per unit time and each node performs x rules per unit time.

For the data-parallel firewall, traffic arrives at the packet distributor, which evenly distributes the traffic. As a result the arrival rate to each machine is λ m. The service rate is x n, since each node implements the complete rule set. The end-to-end delay across any machine in the parallel firewall is

$$E_d(T) = \frac{1}{\frac{x}{n} - \frac{\lambda}{m}} \tag{8}$$

In contrast, rules are distributed across each machine, in the function-parallel firewall. Furthermore, all traffic arriving to the system is forwarded to each machine. If we assume the rules in each machine are independent (no policy DAG edges spanning machines), then the end-to-end delay across any machine in the functional-parallel firewall is

$$E_f(T) = \frac{1}{\frac{m \cdot x}{n} - \lambda} \tag{9}$$

The reduction in delay compared to a data-parallel firewall system is 1 m; therefore, the functional-parallel system has the potential to be m times faster than a (data) parallel system. However, the scalability of the functional-parallel system is dependent on the rule set.

IPR2023-01353 Page 00826

Figures 17A and 17B compare the performance of function-parallel, data-parallel, and single firewall designs. More particularly, Figure 17A compares the average among the different firewall designs as the arrival rate λ increases. As can be seen from Figure 17A, the function-parallel design achieves lower average delay as the arrival rate increases than the data-parallel design. Figure 17B illustrates the average delay as arrival rate and the number of rules increase. In Figure 17B, the function-parallel design again achieves better simulated performance than the data-parallel design.

Experimental Results

The previous section described models that provide insight to the average delay across different firewall designs. This information is beneficial in guiding initial design; however, we are also interested in the maximum delay encountered. For this reason, a discrete event simulator has been developed to model the different systems.

The data-parallel firewall and function-parallel firewall designs were simulated, where both systems implemented a 50-rule policy and each consisted of 5 firewall nodes. Three classes of traffic were modeled, first, second, and background. The first and second classes represent legitimate traffic that was eventually accepted by the firewall system, while background traffic was typically dropped. The two designs were simulated under normal conditions, where each traffic type had the same volume. Figures 18A and 18B shows the simulation results. As seen in Figure 18A and predicted by the theoretical models, the average delay encountered by packets in the functional-parallel design was lower than the data-parallel designs. The function-parallel design also provided lower maximum delays than the data-parallel, as seen in Figure 18B. In summary, the theoretical models and simulation results indicate function-parallel is a better design than a traditional single firewall system or a data-parallel system; yet, function-parallel requires no more computing resources than data-parallel.

Summary

Thus, as described above, the subject matter described herein includes methods, systems, and computer program products for providing a function-

IPR2023-01353 Page 00827