

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

Orca Security Ltd,
Petitioner

v.

Wiz, Inc.,
Patent Owner

Inter Partes Review Case No. IPR2025-01084
U.S. Patent No. 11,929,896

**PETITION FOR *INTER PARTES* REVIEW
OF U.S. PATENT NO. 11,929,896**

TABLE OF CONTENTS

I.	INTRODUCTION.....	1
II.	SUMMARY OF THE '896 PATENT.....	1
A.	DESCRIPTION OF THE '896 PATENT	1
B.	SUMMARY OF THE PROSECUTION HISTORY	4
C.	LEVEL OF SKILL OF A PERSON HAVING ORDINARY SKILL IN THE ART.....	5
III.	REQUIREMENTS UNDER 37 C.F.R. § 42.104	6
A.	GROUND FOR STANDING UNDER 37 C.F.R. § 42.104(A).....	6
B.	IDENTIFICATION OF CHALLENGE UNDER 37 C.F.R. § 42.104(B)	6
C.	CLAIM CONSTRUCTION UNDER 37 C.F.R. § 42.104(B)(3).....	6
IV.	SHOWING OF ANALOGOUS, PRIOR ART	7
A.	U.S. PATENT NO. 11,863,580 (ROSS ET AL.).....	7
B.	U.S. PATENT NO. 9,800,470 (AGARWAL ET AL.).....	8
C.	U.S. PATENT NO. 10,447,553 (BIRAN ET AL.)	9
V.	GROUND 1: CLAIMS 1-29 ARE OBVIOUS OVER ROSS, AGARWAL, AND BIRAN	9
A.	CLAIM 1	9
1.	<i>Claim 1[pre]: “A method for generation of unified graph models for network entities, comprising:”</i>	<i>9</i>
2.	<i>Claim 1[a]: “collecting, for each network entity of a plurality of network entities, network entity data, wherein the network entity data collected for a network entity includes at least a network entity property, wherein the plurality of network entities are deployed in a plurality of cloud computing platforms;”</i>	<i>11</i>
3.	<i>Claim 1[b]: “genericizing each of the network entities based on the respective collected network entity data to generate a plurality of generic network entities, wherein a generic network entity includes a generic representation of respective network entities from different cloud computing platforms of the plurality of cloud computing platforms;”</i>	<i>16</i>
4.	<i>Claim 1[c]: “generating at least a network graph, wherein the generated network graph is a multi-</i>	

	<i>dimensional data structure providing a representation of the plurality of generic network entities and relations between the generic network entities of the plurality of network entities; and”</i>	22
5.	<i>Claim 1[d]: “creating at least one imputed entity, wherein the at least one imputed entity is a generic network entity representing an executed platform functionality, and wherein the executed platform functionality is different than a network entity; and”</i>	28
	a) Discovering host-less components	29
	b) Creating imputed entities based on host-less components.....	32
6.	<i>Claim 1[e]: “storing the generated network graph.”</i>	34
B.	CLAIM 2: “THE METHOD OF CLAIM 1, WHEREIN THE NETWORK ENTITY PROPERTY RELATES TO AT LEAST ONE OF: A NETWORK ENTITY TYPE, A NETWORK ENTITY CLASS, A NETWORK ENTITY CATEGORY, AND A NETWORK ENTITY CONFIGURATION.”	36
C.	CLAIM 3: “THE METHOD OF CLAIM 1, WHEREIN GENERICIZING EACH OF THE NETWORK ENTITIES FURTHER COMPRISES: GENERATING A NEW GENERIC NETWORK ENTITY, WHEREIN THE NEW GENERIC NETWORK ENTITY INCLUDES AT LEAST A NETWORK ENTITY PROPERTY OF NETWORK ENTITIES.”	38
D.	CLAIM 4: “THE METHOD OF CLAIM 1, WHEREIN A NEW GENERIC NETWORK ENTITY INCLUDES AT LEAST A NETWORK ENTITY PROPERTY OF AT LEAST ONE NETWORK ENTITY.”	39
E.	CLAIM 5: “THE METHOD OF CLAIM 1, WHEREIN CREATING AT LEAST ONE IMPUTED ENTITY COMPRISES: IDENTIFYING AT LEAST ONE CLOUD COMPUTING PLATFORM OF THE PLURALITY OF CLOUD COMPUTING PLATFORMS OR ENVIRONMENT FUNCTIONALITY WHICH CORRESPONDS WITH A FUNCTIONALITY OF A GENERIC NETWORK ENTITY.”	39
F.	CLAIM 6: “THE METHOD OF CLAIM 5, WHEREIN THE GENERATED NETWORK GRAPH IS A UNIFIED REPRESENTATION OF THE PLURALITY OF NETWORK ENTITIES INCLUDING THE IMPUTED NETWORK ENTITIES AND THE GENERIC NETWORK ENTITIES.”	40
G.	CLAIM 7: “THE METHOD OF CLAIM 1, WHEREIN THE GENERATED NETWORK GRAPH INCLUDES AT LEAST ONE GRAPH VERTEX, WHEREIN THE AT LEAST ONE GRAPH VERTEX REPRESENTS AT LEAST ONE OF: A NETWORK ENTITY, A GENERIC ENTITY, AND AN IMPUTED ENTITY.”	41

H.	CLAIM 8: “THE METHOD OF CLAIM 7, WHEREIN THE AT LEAST ONE GRAPH VERTEX INCLUDES AT LEAST A PROPERTY LABEL, WHEREIN THE AT LEAST A PROPERTY LABEL INCLUDES AT LEAST A DESCRIPTION OF A PROPERTY OF THE AT LEAST ONE GRAPH VERTEX.”	42
I.	CLAIM 9: “THE METHOD OF CLAIM 7, WHEREIN THE GENERATED AT LEAST A NETWORK GRAPH INCLUDES AT LEAST ONE GRAPH EDGE, WHEREIN THE AT LEAST ONE GRAPH EDGE IS A CONNECTION BETWEEN TWO GRAPH VERTICES, AND WHEREIN THE AT LEAST ONE GRAPH EDGE REPRESENTS A RELATIONSHIP BETWEEN TWO CONNECTED ENTITIES, WHEREIN A CONNECTED ENTITY IS AT LEAST ONE OF: A NETWORK ENTITY, A GENERIC ENTITY, AND AN IMPUTED ENTITY.”	44
J.	CLAIM 10: “THE METHOD OF CLAIM 9, WHEREIN THE AT LEAST ONE GRAPH EDGE INCLUDES AT LEAST ONE OF: A PROPERTY LABEL, AND A DIRECTIONALITY INDICATOR, WHEREIN A PROPERTY LABEL INCLUDES AT LEAST A DESCRIPTION OF A PROPERTY OF THE AT LEAST ONE GRAPH EDGE, AND WHEREIN A DIRECTIONALITY INDICATOR INCLUDES AT LEAST A DESCRIPTION OF A DIRECTION OF THE AT LEAST ONE GRAPH EDGE.”	45
K.	CLAIM 11: “THE METHOD OF CLAIM 1, WHEREIN STORING THE GENERATED NETWORK GRAPH FURTHER COMPRISES: RETURNING AT LEAST A VISUALIZATION OF THE GENERATED AT LEAST ONE NETWORK GRAPH.”	46
L.	CLAIM 12: “THE METHOD OF CLAIM 1, WHEREIN STORING THE GENERATED AT NETWORK GRAPH FURTHER COMPRISES: STORING THE GENERATED NETWORK GRAPH IN A GRAPH DATABASE.”	46
M.	CLAIM 13: “THE METHOD OF CLAIM 1, WHEREIN A NETWORK ENTITY OF THE PLURALITY OF NETWORK ENTITIES INCLUDES ANY ONE OF: A PRIVATE ENDPOINT, A TRANSIT GATEWAY, A TAG-BASED RULESET, AN ENTITY CONFIGURED TO IMPLEMENT A TAG-BASED RULESET, A CONTAINER-MANAGEMENT SERVICE, A CONTAINER-MANAGEMENT APPLICATION, A THIRD-PARTY CONTAINER, A THIRD-PARTY IMAGE, A WEB-ACCESS FIREWALL, A FIREWALL IMPLEMENTATION, A MULTI-ENTITY CONNECTION, A CROSS-ENTITY CONNECTION, A CONTAINER MANAGER, AND A CONTAINER MANAGER CONNECTION.”	47
N.	CLAIM 14	48
	1. <i>Claim 14[pre]: “A non-transitory computer readable medium having stored thereon instructions for causing a</i>	

	<i>processing circuitry to execute a process for generation of unified graph models for network entities, the process comprising: ”</i>	48
2.	<i>Claim 14[a]: “collecting, for each network entity of a plurality of network entities, network entity data, wherein the network entity data collected for a network entity includes at least a network entity property, wherein the plurality of network entities are deployed in a plurality of cloud computing platforms; ”</i>	49
3.	<i>Claim 14[b]: “genericizing each of the network entities based on the respective collected network entity data to generate a plurality of generic network entities, wherein a generic network entity includes a generic representation of respective network entities from different cloud computing platforms of the plurality of cloud computing platforms; ”</i>	49
4.	<i>Claim 14[c]: “generating at least a network graph, wherein the generated network graph is a multi-dimensional data structure providing a representation of the plurality of generic network entities and relations between the network entities of the plurality of network entities; ”</i>	49
5.	<i>Claim 14[d]: “creating at least one imputed entity, wherein the at least one imputed entity is a generic network entity representing an executed platform functionality, and wherein the executed platform functionality is different than a network entity; and ”</i>	50
6.	<i>Claim 14[e]: “storing the generated network graph. ”</i>	50
O.	CLAIM 15	50
1.	<i>Claim 15[pre]: “A system for generating unified graph models for network entities, comprising: ”</i>	50
2.	<i>Claim 15[a]: “a processing circuitry; and ”</i>	50
3.	<i>Claim 15[b]: “a memory, the memory containing instructions that, when executed by the processing circuitry, configure the system to: ”</i>	51
4.	<i>Claim 15[c]: “collect, for each network entity of a plurality of network entities, network entity data, wherein the network entity data collected for a network entity includes at least a network entity property, wherein the</i>	

	<i>plurality of network entities are deployed in a plurality of cloud computing platforms;”</i>	51
5.	<i>Claim 15[d]: “genericize each of the network entities based on the respective collected network entity data to generate a plurality of generic network entities, wherein a generic network entity includes a generic representation of respective network entities from different cloud computing platforms of the plurality of cloud computing platforms;”</i>	51
6.	<i>Claim 15[e]: “generate at least a network graph, wherein the generated network graph is a multi-dimensional data structure providing a representation of the plurality of network entities and relations between the network entities of the plurality of network entities;”</i>	51
7.	<i>Claim 15[f]: “creating at least one imputed entity, wherein the at least one imputed entity is a generic network entity representing an executed platform functionality, and wherein the executed platform functionality is different than a network entity; and”</i>	52
8.	<i>Claim 15[g]: “store the generated network graph.”</i>	52
P.	CLAIM 16: “THE SYSTEM OF CLAIM 15, WHEREIN THE NETWORK ENTITY PROPERTY RELATES TO AT LEAST ONE OF: A NETWORK ENTITY TYPE, A NETWORK ENTITY CLASS, A NETWORK ENTITY CATEGORY, AND A NETWORK ENTITY CONFIGURATION.”	52
Q.	CLAIM 17: “THE SYSTEM OF CLAIM 15, WHEREIN THE SYSTEM IS FURTHER CONFIGURED TO: GENERATE A NEW GENERIC NETWORK ENTITY, WHEREIN THE NEW GENERIC NETWORK ENTITY INCLUDES AT LEAST A NETWORK ENTITY PROPERTY OF NETWORK ENTITIES.”	52
R.	CLAIM 18: “THE SYSTEM OF CLAIM 15, WHEREIN A NEW GENERIC NETWORK ENTITY INCLUDES AT LEAST A NETWORK ENTITY PROPERTY OF AT LEAST ONE NETWORK ENTITY.”	52
S.	CLAIM 19: “THE SYSTEM OF CLAIM 15, WHEREIN THE SYSTEM IS FURTHER CONFIGURED TO: IDENTIFY AT LEAST ONE CLOUD COMPUTING PLATFORM OF THE PLURALITY OF CLOUD COMPUTING PLATFORMS OR ENVIRONMENT FUNCTIONALITY WHICH CORRESPONDS WITH A FUNCTIONALITY OF A GENERIC NETWORK ENTITY.”	52
T.	CLAIM 20: “THE SYSTEM OF CLAIM 19, WHEREIN THE GENERATED NETWORK GRAPH IS A UNIFIED REPRESENTATION OF THE	

	PLURALITY OF NETWORK ENTITIES INCLUDING THE IMPUTED NETWORK ENTITIES AND THE GENERIC NETWORK ENTITIES.”	53
U.	CLAIM 21: “THE SYSTEM OF CLAIM 15, WHEREIN THE GENERATED AT LEAST A NETWORK GRAPH INCLUDES AT LEAST ONE GRAPH VERTEX, WHEREIN THE AT LEAST ONE GRAPH VERTEX REPRESENTS AT LEAST ONE OF: A NETWORK ENTITY, A GENERIC ENTITY, AND AN IMPUTED ENTITY.”	53
V.	CLAIM 22: “THE SYSTEM OF CLAIM 21, WHEREIN THE AT LEAST ONE GRAPH VERTEX INCLUDES AT LEAST A PROPERTY LABEL, WHEREIN THE AT LEAST A PROPERTY LABEL INCLUDES AT LEAST A DESCRIPTION OF A PROPERTY OF THE AT LEAST ONE GRAPH VERTEX.”	53
W.	CLAIM 23: “THE SYSTEM OF CLAIM 21, WHEREIN THE GENERATED AT LEAST A NETWORK GRAPH INCLUDES AT LEAST ONE GRAPH EDGE, WHEREIN THE AT LEAST ONE GRAPH EDGE IS A CONNECTION BETWEEN TWO GRAPH VERTICES, AND WHEREIN THE AT LEAST ONE GRAPH EDGE REPRESENTS A RELATIONSHIP BETWEEN TWO CONNECTED ENTITIES, WHEREIN A CONNECTED ENTITY IS AT LEAST ONE OF: A NETWORK ENTITY, A GENERIC ENTITY, AND AN IMPUTED ENTITY.”	53
X.	CLAIM 24: “THE SYSTEM OF CLAIM 23, WHEREIN THE AT LEAST ONE GRAPH EDGE INCLUDES AT LEAST ONE OF: A PROPERTY LABEL, AND A DIRECTIONALITY INDICATOR, WHEREIN A PROPERTY LABEL INCLUDES AT LEAST A DESCRIPTION OF A PROPERTY OF THE AT LEAST ONE GRAPH EDGE, AND WHEREIN A DIRECTIONALITY INDICATOR INCLUDES AT LEAST A DESCRIPTION OF A DIRECTION OF THE AT LEAST ONE GRAPH EDGE.”	53
Y.	CLAIM 25: “THE SYSTEM OF CLAIM 15, WHEREIN THE SYSTEM IS FURTHER CONFIGURED TO: RETURN AT LEAST A VISUALIZATION OF THE NETWORK GRAPH.”	54
Z.	CLAIM 26: “THE SYSTEM OF CLAIM 15, WHEREIN THE SYSTEM IS FURTHER CONFIGURED TO: STORE THE GENERATED NETWORK GRAPH IN A GRAPH DATABASE.”	54
AA.	CLAIM 27: “THE SYSTEM OF CLAIM 15, WHEREIN A NETWORK ENTITY OF THE PLURALITY OF NETWORK ENTITIES INCLUDES ANY ONE OF: A PRIVATE ENDPOINT, A TRANSIT GATEWAY, A TAG- BASED RULESET, AN ENTITY CONFIGURED TO IMPLEMENT A TAG- BASED RULESET, A CONTAINER-MANAGEMENT SERVICE, A CONTAINER-MANAGEMENT APPLICATION, A THIRD-PARTY CONTAINER, A THIRD-PARTY IMAGE, A WEB-ACCESS FIREWALL, A	

	FIREWALL IMPLEMENTATION, A MULTI-ENTITY CONNECTION, A CROSS-ENTITY CONNECTION, A CONTAINER MANAGER, AND A CONTAINER MANAGER CONNECTION.”	54
BB.	CLAIM 28: “THE METHOD OF CLAIM 3, WHEREIN THE GENERIC NETWORK ENTITY REPRESENTS AT LEAST AN ENTITY CONFIGURED TO APPLY SECURITY RULES.”	54
CC.	CLAIM 29: “THE SYSTEM OF CLAIM 17, WHEREIN THE GENERIC NETWORK ENTITY REPRESENTS AT LEAST AN ENTITY CONFIGURED TO APPLY SECURITY RULES.”	55
VI.	CONCLUSION	55
VII.	MANDATORY NOTICES UNDER 37 C.F.R. § 42.8(A)(1)	56
A.	REAL PARTY-IN-INTEREST	56
B.	RELATED MATTERS	56
C.	LEAD AND BACK-UP COUNSEL	56
D.	37 C.F.R. § 42.8(B)(4) – SERVICE INFORMATION	57

TABLE OF AUTHORITIES

Cases:

<i>IMS Tech., Inc. v. Haas Automation, Inc.</i> , (Fed. Cir. 2000).....	10
<i>Nidec Motor Corp. v. Zhongshan Broad Ocean Motor Co., Ltd.</i> , Cir. 2017)	7
<i>Pitney Bowes, Inc. v. Hewlett Packard Co.</i> , (Fed.Cir.1999)	10
<i>Rowe v. Dror</i> , (Fed. Cir. 1997).....	10

Statutes:

35 U.S.C. § 102(a)(1)	8, 9
35 U.S.C. § 102(a)(2)	7
35 U.S.C. § 103	6

Regulations:

37 C.F.R. § 42.104(b)(3)	6
37 C.F.R. § 42.8(a)(1)	55
37 C.F.R. § 42.8(b)(1)	55
37 C.F.R. § 42.8(b)(3)	55
37 C.F.R. § 42.8(b)(4)	56

I. INTRODUCTION

Petitioner Orca Security Ltd. (“Petitioner”) requests *Inter Partes* Review of Claims 1-29, (the “Challenged Claims”) of U.S. Pat. No. 11,929,896 (the “’896 Patent”) assigned to Wiz, Inc. (“Patent Owner”).

II. SUMMARY OF THE ’896 PATENT

A. Description of the ’896 Patent

The ’896 Patent identifies a problem with which the inventors were involved as providing simple, unified understandings of technologies included in complex and large-scale computing systems implemented through cloud service providers. ’896 *Patent*, 1:13-2:30; Ex. 1003, ¶56. For example, some solutions required separate analysis and representation of similar network components, thus failing to provide simple, unified views. *Id.* Other solutions failed to provide for an integrated representation and analysis of both “explicit” and “implicit” objects (e.g., traffic management devices incorporated into a cloud platform host system). *Id.*

The ’896 Patent describes the “technical field” as generally relating to “cybersecurity,” and, more particularly, to “systems and methods for automated generation of unified graph models for network entities.” ’896 *Patent*, 1:5-9. Figure 2 depicts an exemplary method:

200
)

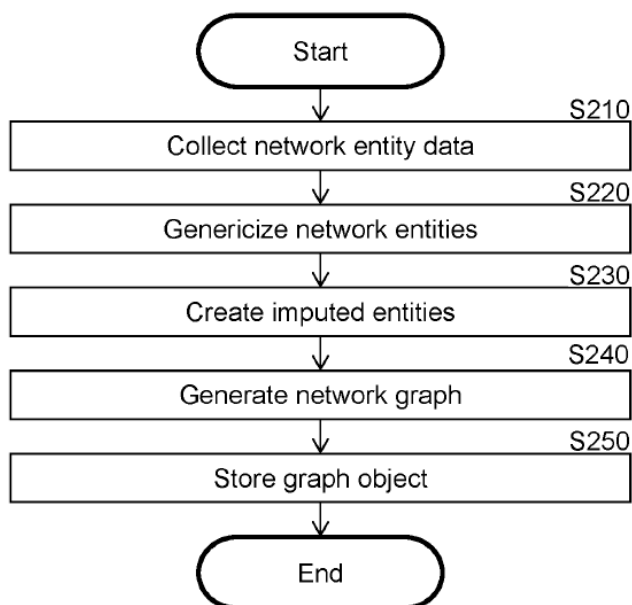


FIG. 2

'896 Patent, Fig. 2, 3:37-39.

The '896 Patent proposes genericizing network entities to create “[g]eneric network entities,” which are network entities “configured to include one or more standard properties.” '896 Patent, 9:18-22. The “[g]enericization” process may include “converting specific objects into generic objects, or generating new generic objects based on specific objects” ('896 Patent, 9:57-60) and is depicted in Figure 4:

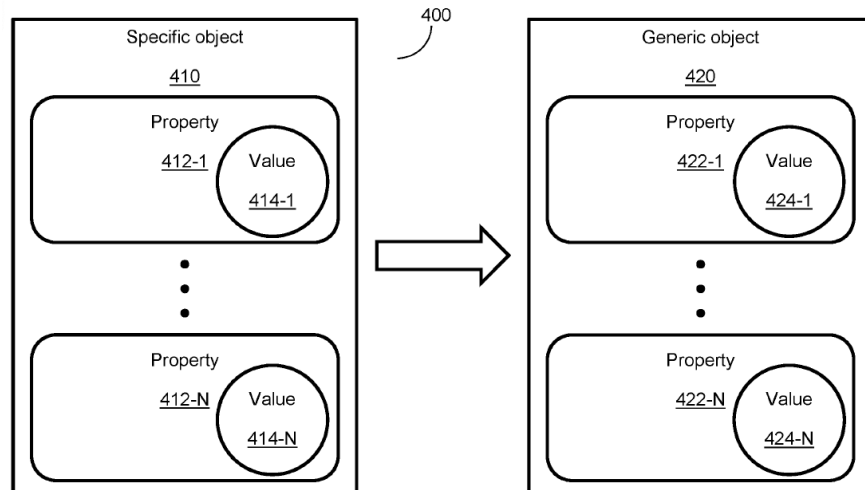


FIG. 4

'896 Patent, Fig. 4; *see also, Id.*, 9:29-31, 3:42-43.

The '896 Patent also proposes creating “imputed entities” to “provide representation of network entities which are integrated into host platforms, or network entities which are shielded from, or not otherwise exposed to, a system configured to execute network analysis processes and methods[.]” *'896 Patent*, 10:58-67. An example of such an implicit network entity is load balancing functionality provided by a cloud platform provider. *'896 Patent*, 11:29-41.

The '896 Patent proposes generating a network graph including generic entities and imputed entities and the connections between such entities. *'896 Patent*, 11:54-64. Figure 3 provides an example network graph visualization:

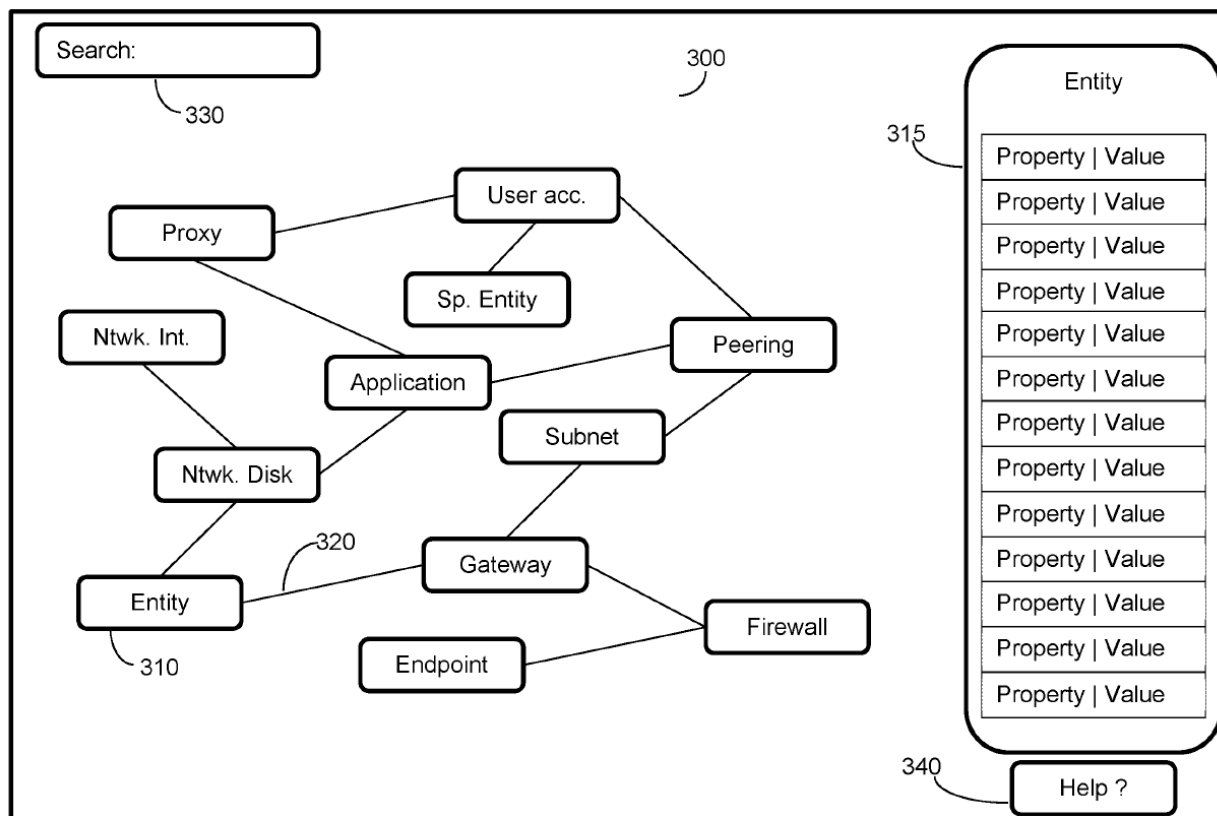


FIG. 3

'896 Patent, Fig. 3; see also, *Id.*, 14:41-49.

B. Summary of the Prosecution History

The '896 Patent issued on March 12, 2024, from an application that was filed January 28, 2021. '896 Patent at (22), (45). During prosecution, the claims were repeatedly rejected. After a final rejection on March 7, 2023, Applicant filed an appeal brief. Ex. 1002 at 134, 100-125. In its appeal brief, Applicant argued Examiner's primary reference, Prabhu (U.S. Patent Publication No. 2019/0258973), failed to teach collecting data on, genericizing, or generating network graphs modeling "network entities." Ex. 1002 at 106-108. Applicant argued that Prabhu is

directed to “business entity modeling,” where the “business entities” are not “network entities.” Ex. 1002 at 106-109. Applicant also argued that the Examiner’s secondary reference, Manghirmalani (U.S. Patent No. 5,819,028), did not teach generic network entities representative of respective network entities, instead teaching that generic objects provided only a level of the network’s functioning (e.g., health, load, performance). Ex. 1002 at 109-110. Applicant also argued that neither reference represented entities across cloud computing platforms. Ex. 1002 at 106-114.

Following the appeal brief, an Examiner’s amendment was entered for the independent claims to include the “creating at least one imputed entity” limitation previously present in dependent claim 5 (Ex. 1002 at 51, 43-49) and a Notice of Allowance issued on November 2, 2023 (Ex. 1002 at 39).

C. Level of Skill of a Person Having Ordinary Skill in the Art

A POSITA at the time of the ’896 Patent would have had a bachelor’s degree in an academic area emphasizing computer science (i.e., computer science, computer information systems, computer engineering, etc.) or cybersecurity and (2) at least two years of work experience with cybersecurity and cloud computing. A POSITA would also have had some training in graph theory. Additional education or experience might substitute for the above requirements. Ex. 1003, ¶¶53-54.

III. REQUIREMENTS UNDER 37 C.F.R. § 42.104

A. Grounds for Standing Under 37 C.F.R. § 42.104(a)

Petitioner certifies the '896 Patent is eligible for IPR.

B. Identification of Challenge Under 37 C.F.R. § 42.104(b)

Petitioner requests the Challenged Claims be found unpatentable on the following grounds.

Proposed Grounds of Unpatentability	Exhibits
Ground 1: Claims 1-29 are obvious under 35 U.S.C. § 103 over Ross in view of Agarwal and Biran.	1004, 1005, 1006

C. Claim Construction Under 37 C.F.R. § 42.104(b)(3)

Petitioner notes that the parties have agreed to the following construction for one term of the '896 Patent in the related—and stayed¹— district court litigation:

Claim Term	Joint Proposed Construction
“imputed entity”	“Generic entity representing a network entity that is integrated into one or more host platforms, or a network entity that is shielded from, or not otherwise exposed to, a system configured to execute network analysis processes and methods”

¹ The district court litigation was stayed “through final written decision in each of the Wiz IPRs.” Ex. 1012, 2. The “Wiz IPRs” refer to the instituted IPRs filed by Patent Owner against Petitioner’s patents. Ex. 1012, 1.

Ex. 1053, Amended Joint CC Chart, 3. The district court’s *Markman* hearing “is CONTINUED” and has not been rescheduled in view of the instituted Wiz IPRs and related stay. *Orca Security Ltd. v. Wiz, Inc.*, 1:23-cv-00758-JLH-SFR, Dkt. 223, Oral Order (Dec. 17, 2024).

Because the proposed grounds below render the Challenged Claims obvious under the parties’ agreed construction and the plain/ordinary meaning, Petitioner does not contend that any claim construction is necessary. *Nidec Motor Corp. v. Zhongshan Broad Ocean Motor Co., Ltd.*, 868 F.3d 1013, 1017 (Fed. Cir. 2017)

IV. SHOWING OF ANALOGOUS, PRIOR ART

A. U.S. Patent No. 11,863,580 (Ross et al.)

Ross was filed December 23, 2020, qualifying as prior art to the ’896 Patent under at least 35 U.S.C. § 102(a)(2).² *Ross*, (22), (45). Ross is analogous prior art to the ’896 Patent because Ross is in the same field of endeavor as the ’896 Patent and is reasonably pertinent to a problem with which the inventors of the ’896 Patent were involved. Ex. 1003, ¶¶21-24. Ross, like the ’896 Patent, is directed to cybersecurity, including the generation of graph models for network entities. *Compare Ross*, 1:24-26, Abstract, 3:3-6 *with ’896 Patent*, 1:7-9, Abstract. Further, like the ’896 Patent, Ross addresses problems related to providing a simple, unified network graph of technologies included in multiple cloud computing platforms without requiring

² All references to Title 35 are to AIA unless otherwise stated.

separate representation of similar network components. *Compare Ross*, 2:54-63, 5:67-6:48, 9:20-41, Figs. 2, 4A *with '896 Patent*, 1:13-28, 1:37-62, Fig. 3. Ross was neither cited nor considered during the prosecution of the '896 Patent. *See generally* Ex. 1002.

B. U.S. Patent No. 9,800,470 (Agarwal et al.)

Agarwal was filed November 10, 2014, and published October 24, 2017, qualifying as prior art to the '896 Patent under at least 35 U.S.C. § 102(a)(1). *Agarwal*, (22), (45). Agarwal is analogous prior art to the '896 Patent because Agarwal is in the same field of endeavor as the '896 Patent and is reasonably pertinent to a problem with which the inventors of the '896 Patent were involved. Ex. 1003, ¶¶25-27. Agarwal, like the '896 Patent, is directed to the generation of graph models for network entities. *Compare Agarwal*, 1:20-25, 2:35-38, 7:54-62, 9:20-30 *with '896 Patent*, 1:7-9, Abstract. Further, like the '896 Patent, Agarwal addresses problems related to providing a simple, unified network graph of technologies included in a large cloud computing environment without requiring separate representation of similar network components and including automatically discovered cloud resources such as virtual load balancers. *Compare Agarwal*, 1:20-25, 1:60-2:7, 2:47-62, 5:6-19, 8:48-52, 13:21-32 *with '896 Patent*, 1:13-28, 1:37-2:28, Fig. 3. Agarwal was neither cited nor considered during prosecution of the '896 Patent. *See generally* Ex. 1002.

C. U.S. Patent No. 10,447,553 (Biran et al.)

Biran was filed May 5, 2017, and published October 15, 2019, qualifying as prior art to the '896 Patent under at least 35 U.S.C. § 102(a)(1). *Biran*, (22), (45). Biran is analogous prior art to the '896 Patent because Biran is in the same field of endeavor as the '896 Patent and is reasonably pertinent to a problem with which the inventors of the '896 Patent were involved. Ex. 1003, ¶¶28-32. Biran, like the '896 Patent, is directed to the generation of a unified, service-aware view of network entities. *Compare Biran*, 1:33-39, 1:59-2:16 with '896 Patent, 1:7-9, Abstract. Further, like the '896 Patent, Biran addresses problems related to providing a simple, unified view of technologies included in a large cloud computing environment, including automatically discovered transparent assets incorporated into the cloud platform host system, such as AWS or Azure load balancers. *Compare Biran*, 1:16-41, 1:52-2:16, 4:31-41 with '896 Patent, 1:13-28, 1:37-2:28, Fig. 3. Biran was neither cited nor considered during the prosecution of the '896 Patent. *See generally* Ex. 1002.

V. GROUND 1: CLAIMS 1-29 ARE OBVIOUS OVER ROSS, AGARWAL, AND BIRAN

A. Claim 1

1. Claim 1[pre]: “A method for generation of unified graph models for network entities, comprising:”

Petitioner contends that Claim 1[pre] is not limiting because the body of the claim sets out the complete invention, and the preamble is not necessary to give “life,

meaning, and vitality” to the claim. *Pitney Bowes, Inc. v. Hewlett Packard Co.*, 182 F.3d 1298, 1305 (Fed.Cir.1999); *See also Rowe v. Dror*, 112 F.3d 473, 478 (Fed. Cir. 1997) (“[W]here a patentee defines a structurally complete invention in the claim body and uses the preamble only to state a purpose or intended use for the invention, the preamble is not a claim limitation.”); *IMS Tech., Inc. v. Haas Automation, Inc.*, 206 F.3d 1422, 1434 (Fed. Cir. 2000) (“If the preamble adds no limitations to those in the body of the claim, the preamble is not itself a claim limitation and is irrelevant to proper construction of the claim.”)

To the extent the preamble is deemed limiting, the combined Ross-Agarwal-Biran system teaches a method for the generation of unified graph models for network entities. *See* Claims 1[a]-1[e]. Namely, the '896 Patent provides “a flowchart depicting a method for generating unified graph models for network entities, according to an embodiment” in Figure 2, and the Figure 2 steps correspond (at a high level) to each of Claims 1[a]-1[e]. '896 Patent, 3:37-39, Fig. 2, Claim 1. Because the Ross-Agarwal-Biran combination teaches Claims 1[a]-1[e], the combination also renders obvious Claim 1[pre].

2. ***Claim 1[a]: “collecting, for each network entity of a plurality of network entities, network entity data, wherein the network entity data collected for a network entity includes at least a network entity property, wherein the plurality of network entities are deployed in a plurality of cloud computing platforms;”***

Ross alone or in view of Agarwal discloses limitation 1[a]. Ross teaches collecting *network entity data*³ including *at least a network entity property* (e.g., IP address, geographic location/zone, database name, service name, etc., Ross at 4:10-30, 5:36-45, 3:35-46) for each *network entity* (e.g., workload) of a *plurality of network entities* (e.g., workloads 110_{1,1} – 110_{X,Y}, Ross at Fig. 1) that are *deployed in a plurality of cloud computing platforms* (e.g., cloud computing environment 100, which “can be hosted by more than one cloud service, such as those provided by Amazon, Microsoft, and Google,” Ross at 2:54-3:6, 4:10-16, Fig. 2).

First, Ross’s workloads are *network entities*. As described in Ross, a “workload” can be “a unit of computing resource, such as a physical computing system (also referred to as a bare metal server), virtual machine, container, pod, and combinations thereof.” Ross, 3:3-6. Similarly, the ’896 Patent provides example “network entities” including “containers,” “virtual machines,” “virtual entities or instances of systems, devices or components,” “third-party containers and images,” “virtual entities, devices, and the like, to process layer-7 (application layer)

³Claim language indicated by italics unless stated otherwise.

traffic[.]” ’896 Patent, 5:44-6:14, Claim 13. Workloads, represented as “nodes” in Ross’s graph, perform network activities and are entities configured to process data, send data, or receive data. *Compare, e.g., Ross*, 6:36-43 (describing how constituent workloads of a community “each perform different functions, such as web server, database server, application server, and the like[.]” “communicate with each other often and in high volume[.]” “act collectively to perform an application, service, and/or business function”), 13:32-33 and Fig. 7 (“Node 710 may send out information consumed by Node 730”), 4:25-30 (describing data packets exchanged between workloads), 3:47-67 *with* ’896 Patent, 5:29-37 (“network entities” include “entities configured to process data, send data, or receive data”), 5:37-43 (“network entities” connect with other network and/or external entities for “sending data” and “receiving data”). A POSITA would have understood Ross’s workloads to be network entities. Ex. 1003, ¶¶74-77; Ex. 1052 (Woolward), 6:42-43 (“Each of VMs 260₁-260_v (FIG. 2) and containers 340₁-340_z can be referred to as workloads and/or endpoints.”).

Second, Ross’s plurality of workloads 110_{1,1} – 110_{x,y} *are deployed in a plurality of cloud computing platforms*. For example, Ross discloses “cloud computing environment 100 including workloads 110_{1,1} – 110_{x,y}” “can physically reside in one or more data centers and/or be physically distributed over multiple locations” and that environment 100 “can be **hosted by more than one cloud**

service, such as those provided by Amazon, Microsoft, and Google.” *Ross*, 2:54-63 (emphasis added⁴); *see also, Id.*, 8:4-9 (contemplating “one” cloud computing environment of “cloud services 260₁-260_Z (e.g., Amazon Web Services (AWS), Microsoft Azure, and Google Cloud Platform (GCP))”), 8:57-62 (describing a container or pod (i.e., workload, *see supra*) that “is deployed” in the cloud computing environment “hosted by cloud services 260₁-260_Z”), 8:11-14 (“Cloud drivers 250₁-250_Z maintain an inventory and topology (i.e., current state) of the workloads in the cloud computing environments hosted by cloud services 260₁-260_Z, respectively.”), 2:63-3:2. Cloud computing platforms, also known as “cloud platforms,” generally refer to the underlying infrastructure, and a “cloud service” generally refers to specific services (e.g., Infrastructure as a Service, Platform as a Service, Software as a Service) built on top of that platform. Ex. 1003, ¶85 (citing Ex. 1007 (US2019/0190778), [0001], [0009]); *'896 Patent*, 5:1-5 (“A cloud platform 104 may be a commercially available cloud system, provided on a service basis, such as, as examples and without limitation, Amazon AWS®, Microsoft Azure®, and the like.”). *Ross*’s disclosure that a cloud environment can be hosted by multiple cloud services (e.g., Amazon AWS, Microsoft Azure, and Google GCP) is an express teaching that workloads of such an environment are deployed in a plurality of cloud platforms. Ex. 1003, ¶¶84-86 (citing Ex. 1007 (US2019/0190778), [0001], [0009]).

⁴ Bold emphasis added unless stated otherwise.

Finally, Ross teaches collecting *network entity data* including *at least a network entity property* from each of the plurality of workloads. For example, Ross teaches: “Exemplary methods include: **gathering data about workloads and applications in the cloud computing environment ...**” *Ross*, Abstract. Ross teaches collecting “streaming telemetry 275 from network logs 270 (also referred to herein as logs or network and middleware system logs), events 285 from cloud control plane 280, and inventory 295 from configuration management database (CMDB) 290.” *Ross*, 4:3-9, 9:10-19, Figs. 2, 5. A POSITA would have understood that each such source of information provides data, including network entity data (e.g., data describing one or more entities in the cloud environment 100). *Compare, e.g., '896 Patent*, 8:64-67 (“Network entity data [] is data describing one or more entities included in a network, cloud, environment, or other, like, deployment.”) *with Ross*, 4:45-50 (“Events 285 can include information about a container (and/or a pod) being created” including “**information about the workload such as a service name, image deployed**, and the like can be received in events 285.”), 4:53-58 (CMDB 290 “can be a database of **information about the hardware and software components (also known as assets)** used in [] cloud computing environment 100 [] and relationships between those components and business functions”), 4:10-30 (streaming telemetry 275 provides information about data exchanges between

workloads including the 5-tuple (**source IP address**/port number, **destination IP address**/port number, and protocol in use for the exchange)); Ex. 1003, ¶¶78-83.

The '896 Patent provides exemplary “properties” including “entity names, addresses, types, configurations, edge direction, and the like[.]” *'896 Patent*, 13:34-40; *see also, Id.*, 8:64-9:5. Ross expressly teaches that collected information includes IP addresses. *Ross*, 4:10-30, Fig. 4A. More broadly, a POSITA would have understood a “network entity property” to be an attribute or characteristic of the network entity (i.e., Ross’s workload). Ex. 1003, ¶83 (citing Ex. 1042 and Ex. 1051). Ross teaches that “[u]sing streaming telemetry 275, events 285, and inventory 295, graph 220 (FIG. 2) can determine information 335 about node 330” (i.e., one of the workloads) including application or service name, function (e.g., application, service, business function), “business organization (e.g., division within a company),” “realm (e.g., production system, development system, and the like),” “(geographic) location/zone” and the “name of the database[.]” among other attributes. *Ross*, 5:36-45, 5:30-31, Fig. 3. A POSITA would have understood that such attributes are collected network entity properties of each workload. Ex. 1003, ¶¶78-83. Additionally, a POSITA would have understood that where one or more workloads is a container, the collected “service name” is a property or attribute of the container and therefore constitutes a network entity property. *Ross*, 3:35-46, 4:45-50; *see also '896 Patent*, Claim 13; Ex. 1003, ¶¶82-83.

Additionally, a POSITA would have found it obvious and been motivated to implement the combined Ross-Agarwal system to collect network entity property information including “security groups,” “instance tags,” “auto scaling groups,” “key pair,” process names, port numbers, identifier, instance name, region, etc. *See* Claim 1[b], Claim 2, *infra*; Ex. 1003, ¶¶87-91, 104-107.

3. ***Claim 1[b]: “genericizing each of the network entities based on the respective collected network entity data to generate a plurality of generic network entities, wherein a generic network entity includes a generic representation of respective network entities from different cloud computing platforms of the plurality of cloud computing platforms;”***

Ross alone or in view of Agarwal discloses limitation 1[b]. Ross teaches or suggests, *genericizing* (e.g., classifying into groups, clustering together, and/or forming into communities, *Ross* at 5:64-6:48) each of the *network entities* (e.g., workloads 110_{1,1} – 110_{X,Y}, *see* Claim 1[a]) based on collected *network entity data* (e.g., streaming telemetry 275, events 285, inventory 295, *see* Claim 1[a]). For example, Ross discloses classifying each workload node in graph database 225 into groups by analyzing its “relationships” to other workload nodes, which are created “as streaming telemetry 275, events 285, and inventory 295 are processed” (i.e., based on collected *network entity data*). *Ross*, 5:64-6:2; *see also id.*, 6:2-6 (“if the node accepts database client connections from systems that are known to be application servers, then graph 220 may classify the node as a database management system (i.e., a certain group)”); Ex. 1003, ¶¶92, 95. The classification criteria for

membership in a group can include heuristic rules or machine learning algorithms. *Ross*, 6:6-14.

Additionally, or alternatively, *Ross* discloses clustering together workload nodes that behave similarly, for example, workloads that “run the same application, are associated with the same organization, and the like[,]” including workloads that fulfill the same function. *Ross*, 6:21-32. *Ross* further discloses identifying “communities” consisting of workloads that “act collectively to perform an application, service, and/or business function.” *Ross*, 6:33-48. Application and service name, application and service function, business organization, etc., are collected streaming telemetry 275, events 285, and inventory 295 data, and thus clustered workload nodes and identified communities are based on collected network entity data. *Ross*, 5:36-45; Ex. 1003, ¶¶93-95.

Ross’s cloud computing environment 100 includes *network entities* (e.g., workloads 110_{1,1} – 110_{X,Y}, *Ross* at Fig. 1) that are *deployed in a plurality of cloud computing platforms* (e.g., cloud computing environment 100, which “can be hosted by more than one cloud service, such as those provided by Amazon, Microsoft, and Google[,]” *Ross* at 2:54-3:6, Fig. 2 (Cloud SVC 260₁ ... Cloud SVC 260_z)). *See* Claim 1[a]. A POSITA would have understood or found obvious that *Ross*’s disclosed classification/clustering/community grouping is across all workloads from which data is collected, and thus workloads of different cloud service providers are

classified/clustered/grouped together so long as they, for example, run the same application or perform the same function. Ex. 1003, ¶96.

A POSITA reading Ross's disclosure of 1) classifying workload nodes into groups; 2) clustering together workload nodes; and/or 3) identifying "communities" of workloads would have understood Ross to teach, or at the very least suggest, genericizing workloads based on collected network entity data, including from different cloud computing platforms. Ex. 1003, ¶¶96-97. However, Ross does not disclose the specifics of any generic network entity that results from such genericizing. Ex. 1003, ¶97.

In related art, Agarwal expressly discloses genericizing *network entities* (e.g., cloud resources, such as virtual machines or instances (servers), *Agarwal*, 5:5-14, 5:35-43, 11:59-63, 15:11-13) based on collected *network entity data* (e.g., security group, instance tags, process name, identifier, instance name, region, etc., *Agarwal*, 15:45-51, 18:55-19:5, 25:34-39) to generate *generic network entities* (e.g., node groups), where each group includes *a generic representation of respective network entities* (e.g., a group object, e.g., *Agarwal*, 24:23-46, that shares the common properties of the specific workload nodes in the group). Namely, Agarwal teaches "collect[ing] data associated with a user's collection of cloud resources[,]" such as through deployed sensors or a cloud provider's API, where "[t]he collected data can reveal a user's nodes and the connection between nodes in the cloud environment."

Agarwal, 14:8-24; *See also* 14:10-20, Fig. 7 (“Collect data using cloud resource API”).

The Agarwal system uses one or more collected node characteristics (e.g., security groups, instance tags, load balancer, key pair, auto scaling groups, process names, port numbers, identifier, instance name, region, etc.) to generate node groups. *Agarwal*, 15:45-51, 18:55-19:5, 25:34-39. The specific characteristics used to generate the node groups can be selected/recommended by system logic, and the user may manually select or modify the metadata attributes used. *Id.* A POSITA would have understood that Agarwal’s characteristics (also referred to as “parameters” or “metadata attributes”) are collected *network entity data*, such as through deployed sensors or a cloud provider’s API. *Compare, e.g., ’896 Patent*, 8:64-67 (“Network entity data [] is data describing one or more entities included in a network, cloud, environment, or other, like, deployment.”) *with Agarwal*, 15:14-17 (referring to security groups, instance tags, load balancer, key and auto scaling groups as “characteristics of nodes”), 14:21-22 (“collected data can reveal a user’s nodes and the connection between nodes in the cloud environment”), 18:34-35 (“An entity can be described by metadata.”), 14:8-17, 18:55-56; Ex. 1003, ¶¶87-90, 99-100. Like Ross, Agarwal teaches using clustering techniques to generate initial node groups. *Agarwal*, 19:6-49.

A POSITA would have understood that each of Agarwal’s node groups is *a generic representation of respective network entities*. Ex. 1003, ¶¶98-102. Like the *generic network entities* of the ’896 Patent, Agarwal’s node groups include properties that are common to the specific nodes that are part of the group. *Compare, e.g., ’896 Patent* at 15:56-16:60, Fig. 4, *with Agarwal* at 23:65-25:20; Ex. 1003, ¶¶100-101. Namely, in Agarwal’s “Test Case 1,” collected instance characteristics include:

node ID #		Tag			SG	KP	LB	ASG
1	Key	Name	tag.a	tag.b	sg.x	key.x	lb.x	asg.x
	Value	node.x.1	tag.a.x	tag.b.x				
2	Key	Name	tag.a	tag.b	sg.x	key.x	lb.x	asg.x
	Value	node.x.2	tag.a.x	tag.b.x				
3	Key	Name	tag.a	tag.b	sg.x	key.x	lb.x	asg.x
	Value	node.y.1	tag.a.x	tag.b.x				
4	Key	Name	tag.a	tag.b	sg.x	key.x	lb.x	asg.x
	Value	node.y.2	tag.a.x	tag.b.x				

Agarwal, 24:6-22 (annotated); Ex. 1003, ¶101. Note that node 1 and node 2 have similar names of node.x.#, while node 3 and node 4 have similar names of node.y.#. *Id.* Following Test Case 1, the two generated groups include properties that correspond to the properties shared among the nodes of the group:

Group Name	Nodes ID #	Group Details							
tag:Name:name.x	1, 2			Tag		SG	KP	LB	ASG
		Key	Name	tag.a	tag.b	sg.x	key.x	lb.x	asg.x
tag:Name:name.y	3, 4	Value	node.x	tag.a.x	tag.b.x				
				Tag		SG	KP	LB	ASG
		Key	Name	tag.a	tag.b	sg.x	key.x	lb.x	asg.x
		Value	node.y	tag.a.x	tag.b.x				

Agarwal, 24:23-46; Ex. 1003, ¶¶101-102.

A POSITA would have found it obvious and been motivated to implement the Ross system to genericize network entities as taught by *Agarwal*. Ex. 1003, ¶¶103-108. *Agarwal* provides an express motivation for the combination: “The grouping of nodes can be helpful when attempting to understand system properties in a cloud environment, like availability, security and performance, where the system includes a large number of nodes (e.g., more than twenty single nodes). . . .” *Agarwal*, 4:13-26; *see also*, 13:21-27; Ex. 1003, ¶103. Additionally, the Ross-*Agarwal* combination would have merely involved use of known techniques to improve similar devices in the same way. Ex. 1003, ¶106. A POSITA would have recognized that both references expressly contemplate grouping network entities based on collected network entity data and also using clustering techniques and machine learning algorithms to group network entities. Ex. 1003, ¶107; *Ross*, 5:64-6:48; *Agarwal*, 13:21-46, 18:27-29, 19:6-49, 25:34-39. A POSITA would have been motivated to implement the genericizing teachings of *Agarwal* into *Ross* for ease of visual

representation to a user. Ex. 1003, ¶107. Implementing the combined system to group network entities based on other collected network entity data (e.g., security group, instance tags, instance name, identifier, geographic location/region, etc.) would have been obvious at least because these other properties are the type of data that is typically stored in a CMDB from which Ross collects data. Ex. 1003, ¶105.

It also would have been obvious to a POSITA to implement the combined Ross-Agarwal system, such that the *generic network entity* (e.g., workload node group) includes a generic representation of respective network entities *from the different cloud computing platforms* (e.g., Amazon, Microsoft, Google, *see* Claim 1[a]). Ex. 1003, ¶¶104-106. A POSITA would have been motivated to group similar workload nodes across cloud service providers (i.e., from different cloud computing platforms) to simplify the visualization, management, and security of complex cloud environments. Ex. 1003, ¶104.

4. Claim 1[c]: “generating at least a network graph, wherein the generated network graph is a multi-dimensional data structure providing a representation of the plurality of generic network entities and relations between the generic network entities of the plurality of network entities; and”

Ross in view of Agarwal discloses limitation 1[c]. First, Ross teaches generating *at least a network graph* (e.g., graph database), which a POSITA would have understood to be a multi-dimensional data structure. Ex. 1003, ¶¶109-110. Namely, Ross teaches graph 220 uses collected data (e.g., streaming telemetry 275,

events 285, and inventory 295) “to create and update graph (database) [225].” *Ross*, 5:16-20, Fig. 2. “Graph database 225 uses a graph data model comprised of nodes (also referred to as vertices), which is an entity such as a workload[,],” and “edges, which represent the relationship between two nodes.” *Ross*, 4:66-5:13; *see also, Id.* 5:7-13. A POSITA would have understood that a “graph database,” as described in *Ross*, is a specific type of database that stores data as nodes (entities) and edges (relationships), instead of in tables and documents. Ex. 1003, ¶¶109-110. Accordingly, a POSITA would have understood that creating and updating a graph database, as described in *Ross*, constitutes generating a network graph that is a multi-dimensional data structure. Ex. 1003, ¶¶109-110.

Second, a POSITA would have found *Ross* to teach or suggest that the graph database provides a *representation of the plurality of generic network entities* (e.g., workload node groups, *see* Claim 1[b], visually represented as a single shape, *Ross*, 6:21-48) in addition to the *relations* (e.g., double-sided arrows) between such groups. *Ross*, Figure 4A; Ex. 1003, ¶¶111-114. Namely, *Ross* teaches “visually organiz[ing] information from graph database 225.” *Ross*, 6:15-48, Fig. 4A. In *Ross*, a group of specific workload entities (e.g., as a result of being clustered together or formed into a community) can be visually presented as a single shape (e.g., circle, rectangle, and the like) instead of a shape for each individual specific entity. *Ross*, 6:21-48. *Ross* discloses that a single shape of the network graph can denote “the

application performed, the number of constituent workloads, and the like.” *Ross*, 6:43-48. For example, Figure 4A includes a group of 51 workloads labeled “Auto-Group 1” (annotated in red in Fig. 4A below). Ex. 1003, ¶113. In view of these disclosures, a POSITA would have found *Ross* to teach, or at least suggest, that the visualized graph of Figure 4A (and, thus the contents of the graph database) provides a representation of groups of specific entities in addition to the relations between such group entities (i.e., depicted as double-sided arrows). Ex. 1003, ¶113. For example, a POSITA would have understood Figure 4A to show or at least suggest (particularly in view of *Agarwal* as discussed below), that a group of 4 workloads running “SWIFT_NetLink” communicate with 8 workloads running “SWIFT_GUI” (annotated in green in Fig. 4A below). Ex. 1003, ¶113.

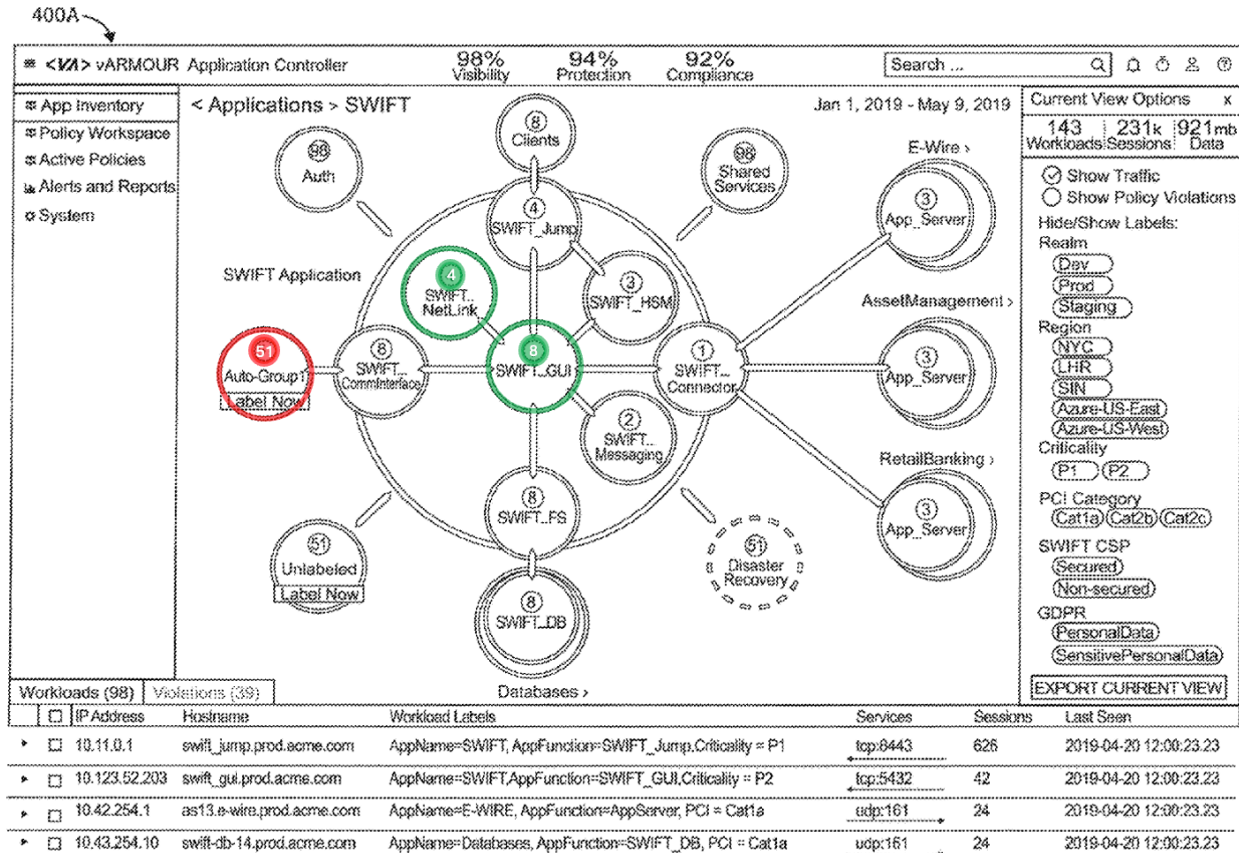


FIG. 4A

Ross, Fig. 4A (annotated).

To the extent Patent Owner argues that the disclosures of Ross alone are insufficient to render this limitation obvious, Agarwal is directed to “a system and a user interface associated with grouping entities and visualizing the relationship [...] of grouped entities in the context of virtual networks and clouds.” *Agarwal*, 1:20-25. In addition to disclosing generating *generic network entities* (e.g., node groups, *see above* Claim 1[b]), Agarwal teaches generating flows between node groups, where a collection of flows can form “flow groups.” *Agarwal*, 14:49-51, 14:62-67, 15:37-44. Similar to Ross’s network graph, Agarwal teaches generating a “flow

map” that “represents a particular grouping of resources and the connections between the resources.” *Agarwal*, 8:11-12. “FIG. 23H shows a flow map for the groups of Family A,” including groups of nodes and the relationships between the groups, as shown below:

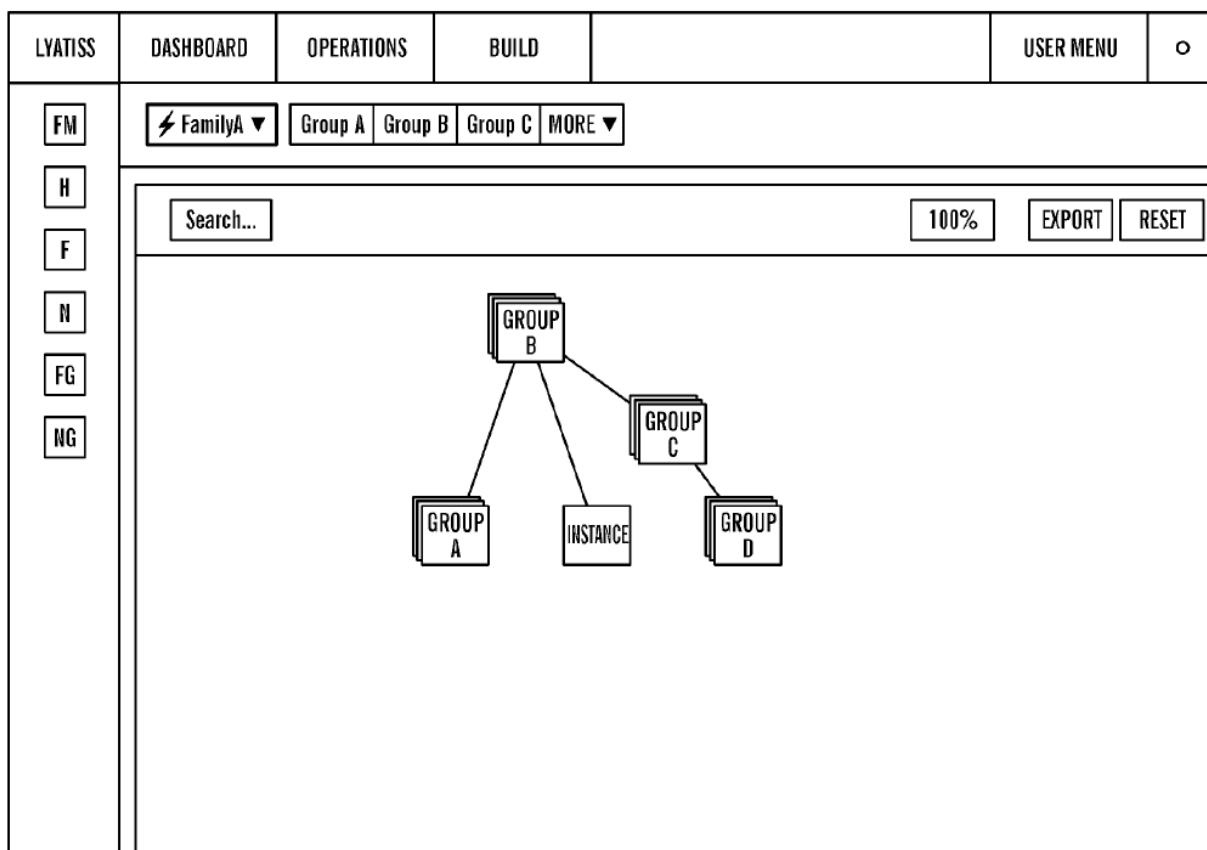


FIG. 23H

Agarwal, 25:67-26:1, Fig. 23H.

Agarwal also teaches “recommend[ing] a group structure for a set of ungrouped nodes” and that “[t]he ability of the system to generate an initial group structure for an existing set of nodes and recommend a placement of new nodes

within an existing group structure can also simplify the management of a cloud system.” *Agarwal*, 18:15-26. For example, when using a machine learning algorithm such as clustering to create groups, Agarwal teaches storing new groups in a “Groups DB.” *Agarwal*, 3:51-54, Fig. 20 (illustrating the result of “Add nodes to Closest Cluster” and “create new Cluster” is to “store in DB,” namely the “Groups DB”). Agarwal also teaches using a Grouping Manager service to create groups of nodes (*see* limitation 1[b]) and flows, and a TGraph Server service to represent a cloud infrastructure as a graph, where the output of the Grouping Manager is to the TGraph Server whose output is to “OrientDB.” *Agarwal*, 23:51-64, Fig. 21. A POSITA would have understood that “Orient DB,” initially released in 2010, is an open-source NoSQL database management system that supports graph models and where relationships are managed in the same way as graph databases with direct connections between records. Ex. 1003, ¶118. Based on these disclosures, a POSITA would have understood Agarwal to teach a multi-dimensional data structure (e.g., Orient DB) providing a representation of generic network entities (e.g., node groups) and relations between them (e.g., flow groups). Ex. 1003, ¶¶115-120.

A POSITA would have found it obvious and been motivated to implement the Ross-Agarwal system to generate at least a network graph that is a multi-dimensional structure (e.g., a graph database or OrientDB) providing a representation of the plurality of generic network entities (e.g., workload node groups, *see* Claim 1[b])

and relations between them (e.g., flow groups), such that, for example, the workload node groups themselves are stored as individual nodes/records in the database with direct connections between the records representing the flow groups. Ex. 1003, ¶¶121-125. As discussed above, Ross and Agarwal each teach, or at least suggest, that when the contents of a graph database or a similar database are visualized, groups of specific entities and the relationships between those groups are displayed. Implementing the combined Ross-Agarwal network graph structure to provide the generated workload node groups and the relationships between them amounts to use of a known technique to improve similar devices in the same way. Ex. 1003, ¶122. Additionally, a POSITA would have been motivated to generate a network graph in this way to better visualize complex cloud computing environments. Ex. 1003, ¶123. Use of a network graph structure in this way was commonplace and disclosed by numerous references, giving a POSITA a reasonable expectation of success in the modification. Ex. 1003, ¶¶124, 36-48.

5. ***Claim 1[d]: “creating at least one imputed entity, wherein the at least one imputed entity is a generic network entity representing an executed platform functionality, and wherein the executed platform functionality is different than a network entity; and”***

Ross in view of Agarwal and Biran discloses the limitations of Claim 1[d].

a) Discovering host-less components

First, Ross teaches that collected information includes information from configuration management database (CMDB) 290, including known “assets” used in a cloud computing environment. *Ross*, 4:3-9, 4:53-65; *see also*, Claim 1[a]. A POSITA would have found it obvious and been motivated to include host-less component information in Ross’s CMDB and to collect it in view of the express teachings of Biran. Ex. 1003, ¶¶127-136.

Biran teaches “service mapping logic that generates an accurate, service-aware view (e.g., a ‘service mapping’) of the system infrastructure[.]” *Biran*, 1:59-61. Biran’s service mapping logic enables discovery of logical Platform as a Service (PaaS) services, such as “Amazon Web Services (AWS) elastic load balancing (ELB) and relational database services (RDS) and/or Azure Load Balancer (LB),” and stores information regarding the discovered services in the CMDB. *Biran*, 2:4-10, 4:9-10, 4:31-41.

Particularly, Biran teaches that “[t]he CMDB 108 is a series of tables containing information about all of the assets and enterprise services controlled by a client 102 and the configurations of these assets and services. The assets and services include configuration items (CIs) 110 that may be computers, other devices on a network 112 (or group of networks), software contracts and/or licenses, or enterprise services.” *Biran*, 5:41-47; *see also*, *Id.*, 5:47-6:5. According to Biran, PaaS service

discovery is “more complex than discovering typical hosts” because they are “host-less components (e.g., where the host or hosts are transparent to a service consuming the service)[.]” *Biran*, 17:46-65. These host-less components are “discovered by infrastructure patterns and/or application patterns[.]” *Id.*; *see generally*, *Biran*, 17:66-18:46. A POSITA would have understood that host-less components such as an AWS ELB and Azure LB (among other host-less components) are network entities that are integrated into one or more host platforms. Ex. 1003, ¶¶128-131. A POSITA would also have understood that host-less components include components that are transparent (i.e., shielded from or not otherwise exposed) to a system configured to execute network analysis processes and methods. *Id.*

Biran further teaches a “service mapping timeline” that is used to “display an accurate, up-to-date view of infrastructure of one or more networks[.]” including a graphical representation of discovered CIs. *Biran*, 16:38-49. An exemplary service mapping timeline, including a graphical representation of a discovered elastic load balancer 804 (annotated in red below), appears in Figure 12:

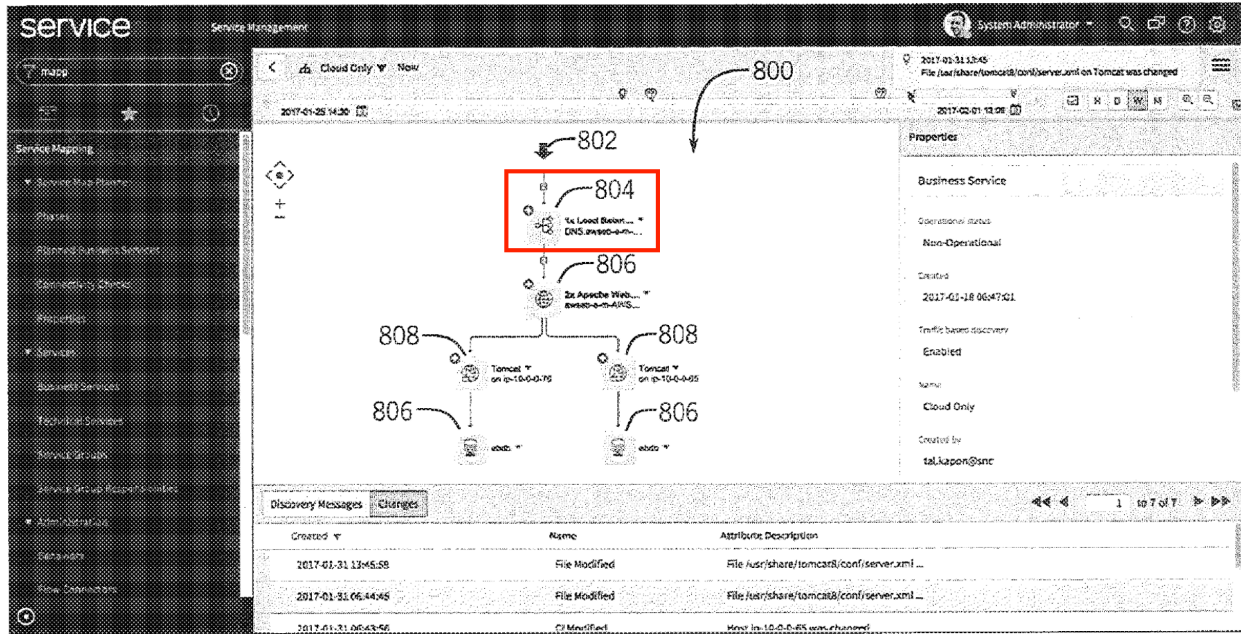


FIG. 12

Biran, Fig. 12 (annotated); *see also*, *Biran*, 18:41-46; Ex. 1003, ¶133. *Biran* expressly discloses that each CI in the CMDB includes a “class” and “type” and that the “type” field indicates that the entity is a particular type of component (e.g., “Cloud Database”) and that the “class” field may indicate, for example, a host-less CI. *Biran*, 18:27-40, Fig. 11.

A POSITA would have understood that the discovered “host-less components” in *Biran* each constitute a network entity that is integrated into one or more host platforms, or a network entity that is shielded from, or not otherwise exposed to, a system configured to execute network analysis processes and methods. Ex. 1003, ¶¶128-131.

Biran provides express motivation to implement Ross to gather information on host-less components (in addition to hosted components) – to generate an accurate, service-aware view of the system infrastructure. *Biran*, 1:59-61; Ex. 1003, ¶134. Moreover, the combination represents the simple addition of one known element (e.g., discovering and tracking host-less components) to another known element (e.g., CMDB) to obtain a predictable result (e.g., a CMDB that tracks both hosted and host-less components). Ex. 1003, ¶135. Moreover, this modification would have been straight forward and well within the skill of a POSITA, giving a POSITA a reasonable expectation of success in the modification. Ex. 1003, ¶136.

b) Creating imputed entities based on host-less components

Second, when implemented to gather information on host-less components, a POSITA would have found it obvious and been motivated to create at least one group of similar host-less components, thereby creating at least one imputed entity. Ex. 1003, ¶¶137-144.

As discussed above with respect to limitation 1[b], the combined Ross-Agarwal system allows for classification/clustering/community formation based on collected data (e.g., application or service name, function, security group, instance tags, instance name, identifier, geographic location/region, etc.) to generate groups of workload nodes that include a generic representation of respective workload nodes (e.g., share common properties or characteristics). *Supra*, limitation 1[b].

Agarwal also broadly teaches that node groups can be created for automatically discovered cloud resources (e.g., *Agarwal*, 4:7-12, 8:44-52, 11:59-12:7, 13:47-50, 14:8-22), including for resources such as “a virtual load balancer, a virtual router, a virtual switch[,]” or other “network service” (e.g., *Agarwal*, 5:12-13, 5:35-46).

When implementing Ross to collect information on automatically discovered host-less components, including elastic load balancers and other transparent assets as taught in Biran, a POSITA would have found it obvious and been motivated to create a group of such transparent assets, particularly in view of Agarwal’s express suggestion to create a group of similar virtual load balancers and other network service resources. Ex. 1003, ¶¶137-138. A POSITA would have understood that a group of similar transparent assets (e.g., elastic load balancers) constitutes an “imputed entity” under the agreed construction of “generic entity representing a network entity that is integrated into one or more host platforms, or a network entity that is shielded from, or not otherwise exposed to, a system configured to execute network analysis processes and methods.” Ex. 1003, ¶¶139-141. A POSITA would also have understood that a group of one or more similar transparent assets represents an executed platform functionality (e.g., AWS elastic load balancing (ELB), Azure Load Balancer (LB)), and that the executed platform functionality is different than a network entity (e.g., workloads, *see* Claim 1[a]). Ex. 1003, ¶139-141.

In addition to providing an accurate, service-aware view of the cloud infrastructure, Agarwal provides an additional express motivation for implementing the combined system in this way: “The grouping of nodes can be helpful when attempting to understand system properties in a cloud environment, like availability, security and performance, where the system includes a large number of nodes (e.g., more than twenty single nodes).” *Agarwal*, 4:13-17; *See also*, *Agarwal*, 13:21-26; Ex. 1003, ¶142. Creating generic entities for transparent assets, such as ones representing an executed platform functionality, was known and specifically suggested by Agarwal, giving a POSITA a reasonable expectation of success in implementing the system in this way. Ex. 1003, ¶143.

6. *Claim 1[e]: “storing the generated network graph.”*

Ross and Agarwal each teach storing generated network graphs in a graph database or a similar database. Ex. 1003, ¶¶145-152. Ross discloses storing the generated graph in “graph database 225.” *Ross*, 4:67-5:3, 9:20-30 (“...stored in graph database 225.... present a graph using data in graph database 225.”). Figure 2 shows Ross’s system for cloud security management, including graph database 225:

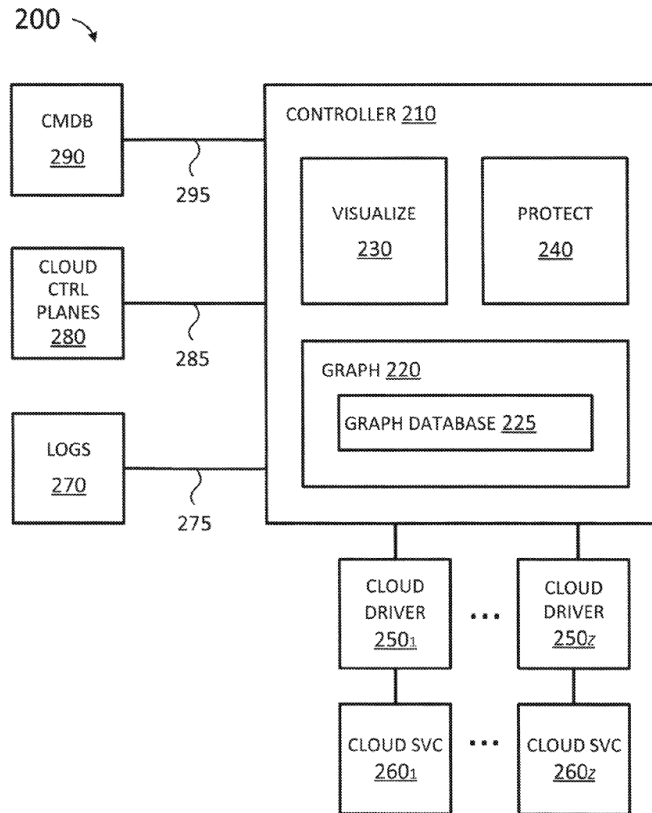


FIG. 2

Ross, Fig. 2; *see also Id.*, 5:16-19, Fig. 5

Additionally, as discussed in Claim 1[c], Agarwal teaches the use of a Grouping Manager service to group nodes and a TGraph Server service to represent a cloud infrastructure as a graph, where the output of the TGraph Server is to “OrientDB.” *Agarwal*, 23:51-64, Fig. 21; *Supra*, Claim 1[c]. A POSITA would have understood that “OrientDB,” initially released in 2010, is an open-source NoSQL database management system that supports graph models and where relationships are managed in the same way as graph databases, with direct connections between records. Ex. 1003, ¶150. Accordingly, a POSITA would have understood Agarwal

to teach storing a network graph in OrientDB, which is similar to a graph database.
Id.

When implementing the Ross-Agarwal-Biran combination, a POSITA would have found it obvious and been motivated to store the generated network graph in a graph database or similar database, including because Ross and Agarwal expressly teach doing so. Ex. 1003, ¶151. A POSITA would have had a reasonable expectation of success in implementing the system to store the generated network graph because storage of network graphs in graph databases or other similar databases was commonplace and taught by Ross and Agarwal. Ex. 1003, ¶151.

B. Claim 2: “The method of claim 1, wherein the network entity property relates to at least one of: a network entity type, a network entity class, a network entity category, and a network entity configuration.”

The '896 Patent does not distinguish between a “network entity type,” a “network entity class,” and a “network entity category,” instead repeatedly referring to them as “like” properties that are “common to specific network entities configured to achieve the same or similar functionalities” including “properties relevant to the execution or provision of such functionalities.” ’896 Patent, 9:32-38; *see also id.*, 9:60-10:2. Exemplary property “configurations” include “pre-defined properties” and “user-defined properties,” among others. ’896 Patent, 9:39-56.

Ross alone or in view of Agarwal and Biran discloses Claim 2. Ross discloses that the collected network entity properties (*see* limitation 1[a]) include application

or service name, function (e.g., application, service, business function), “business organization (e.g., division within a company),” “realm (e.g., production system, development system, and the like),” “(geographic) location/zone” (among others). *Ross*, 5:36-45, 4:45-65, 6:41-43; *Supra*, limitation 1[a]. A POSITA would have considered a workload’s application or service name, function, business organization, realm, and location/zone to each be a network entity “type,” “class,” and/or “category” because they are key characteristics or attributes of the workload that enable classifying the workload together with other similar workloads and are therefore common among workloads that are configured to achieve the same or similar functionality. *Ross*, 6:15-48; Ex. 1003, ¶¶153-155.

Additionally, a POSITA would have found it obvious and been motivated to implement the combined *Ross*-*Agarwal* system to collect network entity properties, such as “security group,” “instance tag,” “auto scaling group,” “key pair,” process names, port numbers, identifier, instance name, region, etc., in order to create groups based on such information as discussed in Claims 1[a] and 1[b]. *Agarwal*, 14:8-24, 15:11-28, 18:55-19:5, Fig. 7 (“step 702”); *Supra*, Claims 1[a] and 1[b]; Ex. 1003, ¶156. A POSITA would have considered a security group, instance tag, autoscaling group, and key pair to be a network entity “configuration” at least because each such property is a “pre-defined property” and/or “user-defined property.” Ex. 1003, ¶157.

Additionally, Biran expressly discloses that each configuration item in the CMDB includes a “class” and “type” and that the “type” field indicates that the entity is a particular type of component (e.g., “Cloud Database”) and that the “class” field may indicate, for example, a host-less CI. *Biran*, 18:27-40, Fig. 11. A POSITA would have found it obvious and been motivated to implement the combined Ross-Agarwal-Biran system to collect class and type information from hosted and host-less components for the reasons discussed in Claim 1[d]. Ex. 1003, ¶158. *See* Section V.A.5.a).

C. Claim 3: “The method of claim 1, wherein genericizing each of the network entities further comprises: generating a new generic network entity, wherein the new generic network entity includes at least a network entity property of network entities.”

In the Ross-Agarwal-Biran combination, a POSITA would have found it obvious and been motivated to *genericize each of the network entities* (*see* Claim 1[b]) such that the genericizing further includes generating a *new generic network entity* (e.g., new workload node group) that includes at least a *network entity property of network entities* (e.g., a property that is common to the specific network entities that the node group represents). Ex. 1003, ¶¶160-167. As discussed above with respect to limitation 1[b], Agarwal expressly teaches genericizing network entities in this way. *Agarwal*, 14:34-40 (“In 710, the system can generate a node grouping model based upon the input parameters and generate initial node groupings so that each node group is populated with one or more nodes.”), 4:29-31 (“the system

can be configured to automatically generate a node grouping”), 24:6-46 (showing that two generated groups include network entity properties (e.g., security group, instance tag, auto scaling group, key pair) that correspond to the properties common to the specific nodes of the group), Fig. 7 (“step 710”); *See also, Agarwal*, 15:45-17:17, 24:47-25:19; *see* limitation 1[b]. For the reasons discussed above with respect to limitation 1[b], a POSITA would have found it obvious and been motivated to implement the Ross system to genericize network entities as taught by Agarwal. Ex. 1003, ¶¶161-166.

D. Claim 4: “The method of claim 1, wherein a new generic network entity includes at least a network entity property of at least one network entity.”

In the Ross-Agarwal-Biran combination, a *new generic network entity* (e.g., a new workload node group) includes at least a *network entity property of at least one network entity* (e.g., a property that is common to the specific network entities that the node group represents). *Supra*, Claim 3; Ex. 1003, ¶168.

E. Claim 5: “The method of claim 1, wherein creating at least one imputed entity comprises: identifying at least one cloud computing platform of the plurality of cloud computing platforms or environment functionality which corresponds with a functionality of a generic network entity.”

In the Ross-Agarwal-Biran combination, creating at least one imputed entity (*see* Claim 1[d]), comprises *identifying at least one cloud platform or environment functionality* (e.g., AWS elastic load balancing (ELB), Azure Load Balancer (LB)),

which *corresponds with a functionality of a generic network entity* (e.g., an automatically generated group of host-less components). *Supra*, Claim 1[d]; Ex. 1003, ¶170.

F. Claim 6: “The method of claim 5, wherein the generated network graph is a unified representation of the plurality of network entities including the imputed network entities and the generic network entities.”

A POSITA would have found it obvious and been motivated to implement the combined Ross-Agarwal-Biran system to include the created *imputed network entities* (e.g., groups of host-less components, *see* Claim 1[d]) in addition to the *generic network entities* (e.g., groups of hosted components, *see* Claims 1[b] and 1[c]) in the network graph, thereby generating a network graph that is a *unified representation* of the network entities. Ex. 1003, ¶¶172-176.

Biran expressly teaches displaying host-less components together with hosted components in a service-mapping timeline. *Biran*, 18:41-46, Fig. 12; *See also supra*, limitation 1[d]. Agarwal broadly teaches that node groups can be created for automatically discovered cloud resources, including resources such as “a virtual load balancer, a virtual router, a virtual switch,” or other “network service.” *Agarwal*, 4:7-12, 8:44-52, 11:59-12:7, 13:47-50, 14:8-22, 5:12-13, 5:35-46. Agarwal also teaches that created groups are displayed in a flow map. *Supra*, limitation 1[c].

When implementing the Ross-Agarwal-Biran system to automatically discover and group host-less components (e.g., AWS elastic load balancing (ELB),

Azure Load Balancer (LB), or other transparent assets), a POSITA would have found it obvious and been motivated to include the group(s) of host-less components in the generated network graph based on the express suggestion of Biran and Agarwal to do so. Ex. 1003, ¶¶177-180. Additionally, implementing the system in this way would have provided an accurate, service-aware view of the multi-platform cloud environment. Ex. 1003, ¶181. Providing a unified representation of hosted and host-less assets together, such as in a network graph, was known and disclosed by Agarwal and Biran, giving a POSITA a reasonable expectation of success in implementing the system in this way. Ex. 1003, ¶182.

G. Claim 7: “The method of claim 1, wherein the generated network graph includes at least one graph vertex, wherein the at least one graph vertex represents at least one of: a network entity, a generic entity, and an imputed entity.”

In Ross, the generated network graph includes at least one graph vertex. *Ross*, 5:3-7 (“Graph database 225 uses a graph data model comprised of nodes (also referred to as vertices), which is an entity such as a workload” and “edges, which represent the relationship between two nodes.”); *see also* limitation 1[c].

In the Ross-Agarwal-Biran combination, the graph vertices represent at least one network entity (e.g., a specific network entity, *see, e.g., Agarwal*, Fig 23H (Node labeled “Instance”)), at least one generic entity (e.g., a group of similar workload nodes, *supra*, Claim 1[c]), and an imputed entity (e.g., a group of similar host-less components, *supra*, limitation 1[d] and Claim 6). A POSITA would have found it

obvious and been motivated to include each such entity in the generated network graph based on the express suggestion of Biran and Agarwal to do so. Ex. 1003, ¶¶184-1890. Additionally, implementing the system in this way would have provided an accurate, service-aware view of the multi-platform cloud environment. Ex. 1003, ¶188. Providing a unified representation of hosted and host-less assets together, such as in a network graph, was known and disclosed by Agarwal and Biran, giving a POSITA a reasonable expectation of success in implementing the system in this way. Ex. 1003, ¶189.

H. Claim 8: “The method of claim 7, wherein the at least one graph vertex includes at least a property label, wherein the at least a property label includes at least a description of a property of the at least one graph vertex.”

Ross teaches at least one graph vertex includes a label, annotated in green in Figure 4A below (e.g., “SWIFT_CommInterface,” “SWIFT_NetLink,” “SWIFT_DB,” etc.):

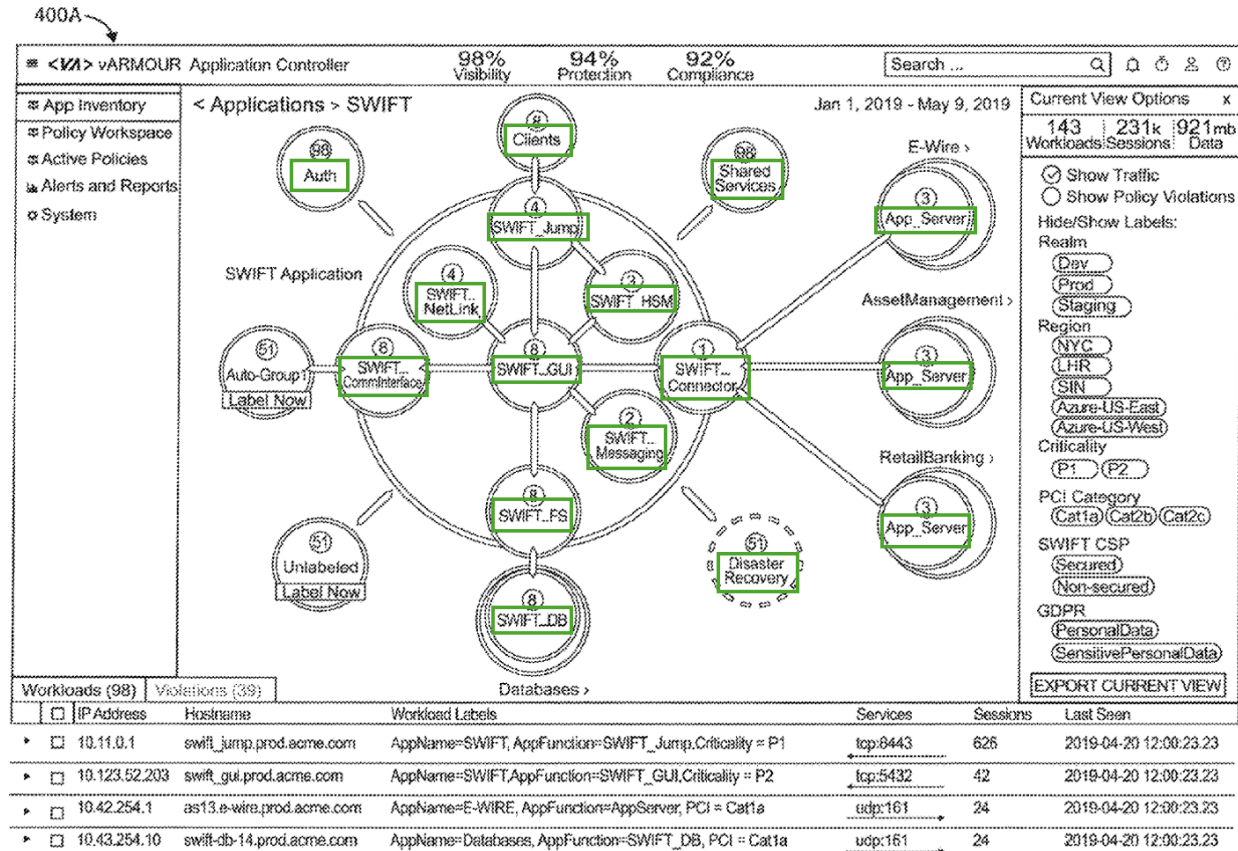


FIG. 4A

Ross, Fig. 4A (annotated); Ex. 1003, ¶191. A POSITA would have understood the above labels to be a description of a property of the vertex (e.g., the application function of the underlying workload(s)). Ex. 1003, ¶¶191-192; *see also* limitation 1[a].

Additionally, or alternatively, a POSITA would have found it obvious and been motivated to implement the combined Ross-Agarwal-Biran system such that graph nodes/vertices are labeled with a description of a property of the node/vertex. Ex. 1003, ¶193. Labels for use in graphs are well-known and commonplace, and a POSITA would have found it obvious and been motivated to include vertex labels

to assist users in understanding the resulting visualization and the underlying network entity/entities that the vertex represents. *Id.*

- I. Claim 9: “The method of claim 7, wherein the generated at least a network graph includes at least one graph edge, wherein the at least one graph edge is a connection between two graph vertices, and wherein the at least one graph edge represents a relationship between two connected entities, wherein a connected entity is at least one of: a network entity, a generic entity, and an imputed entity.”**

Ross teaches a network graph that includes at least one graph edge that is a connection between two graph nodes (i.e., vertices) and that the edge “represent[s] the relationship between two nodes.” *Ross*, 5:3-9. In the Ross-Agarwal-Biran combination, the graph vertices represent a network entity, a generic entity, and an imputed entity. *Supra*, Claim 7. A POSITA would have found it obvious and been motivated to include at least one graph edge in the Ross-Agarwal-Biran network graph to represent relationships between connected entity vertices based on Ross’s express suggestion to do so. Ex. 1003, ¶¶195-201. Use of edges to represent relationships between connected entities was ubiquitous, giving a POSITA a reasonable expectation of success in implementing the system in this way. Ex. 1003, ¶200.

- J. Claim 10: “The method of claim 9, wherein the at least one graph edge includes at least one of: a property label, and a directionality indicator, wherein a property label includes at least a description of a property of the at least one graph edge, and wherein a directionality indicator includes at least a description of a direction of the at least one graph edge.”**

Ross specifically teaches that edges of the network graph “can have a start node, end node, type, and direction,” where “an edge can describe parent-child relationships, actions, ownership, and the like.” *Ross*, 4:67-5:13. A POSITA would have understood that an edge having a start node, end node, and direction is an edge having a “directionality indicator” that “includes at least a description of a direction of the at least one graph edge.” Ex. 1003, ¶202.

Additionally, or alternatively, a POSITA would have found it obvious and been motivated to implement the Ross-Agarwal-Biran combination such that the graph edges include such a directionality indicator. Ex. 1003, ¶203. Directionality indicators for use in graphs were well-known and commonplace, giving a POSITA a reasonable expectation of success in the modification. Ex. 1003, ¶204. A POSITA would have been motivated to include directionality indicators to assist users in understanding the resulting visualization and the underlying relationships of the network entities that the edge connects. Ex. 1003, ¶204.

A POSITA would also have found it obvious and been motivated to implement the combined Ross-Agarwal-Biran system such that graph edges are labeled with a description of a property of the edge. Ex. 1003, ¶205. Labels for use

in graphs are well-known and commonplace, and a POSITA would have found it obvious and been motivated to include edge labels to assist users in understanding the resulting visualization and the underlying relationship that the edge represents. Ex. 1003, ¶205.

K. Claim 11: “The method of claim 1, wherein storing the generated network graph further comprises: returning at least a visualization of the generated at least one network graph.”

Ross, Agarwal, and Biran each teach returning a visualization of a generated graph or map. *Ross*, 9:20-30, Fig. 4A; *Agarwal*, 25:66-26:1, Fig. 23H; *Biran*, 18:41-46, Fig. 12. Accordingly, in implementing the Ross-Agarwal-Biran combination, a POSITA would have found it obvious and been motivated to return a visualization of the generated network graph, including in furtherance to storing the graph. Ex. 1003, ¶¶207-212; *See also, supra*, Claim 1[e].

L. Claim 12: “The method of claim 1, wherein storing the generated at network graph further comprises: storing the generated network graph in a graph database.”

Ross discloses storing the generated network graph in graph database 225:

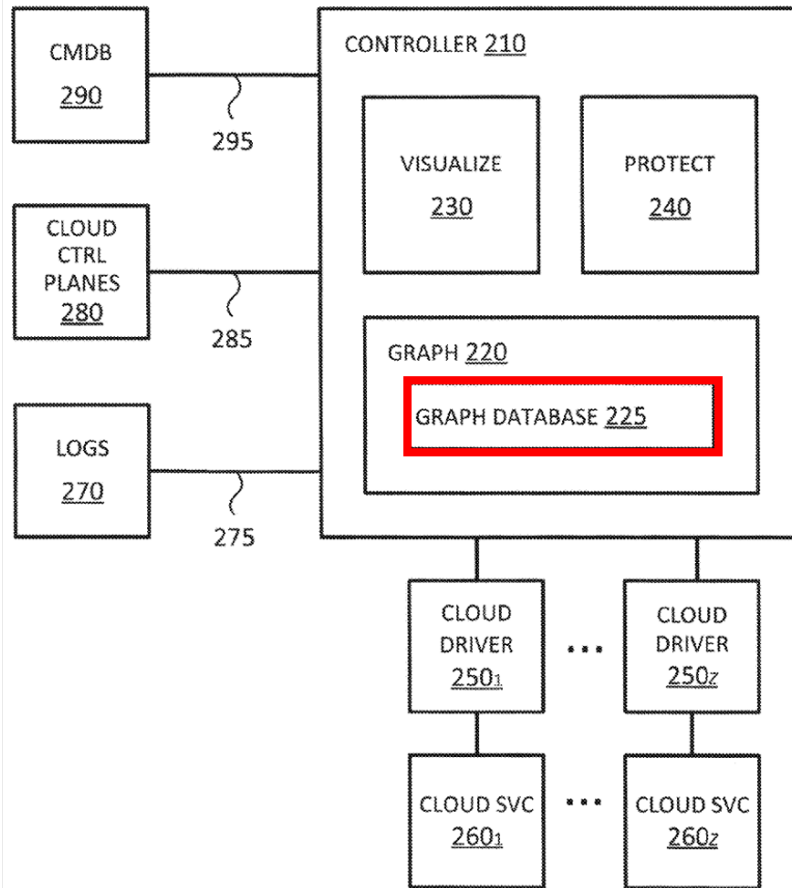


FIG. 2

Ross, Fig. 2 (annotated); *See also*, Ross, 4:67-5:3; Claim 1[c]; Ex. 1003, ¶¶213-217.

- M. Claim 13:** “The method of claim 1, wherein a network entity of the plurality of network entities includes any one of: a private endpoint, a transit gateway, a tag-based ruleset, an entity configured to implement a tag-based ruleset, a container-management service, a container-management application, a third-party container, a third-party image, a web-access firewall, a firewall implementation, a multi-entity connection, a cross-entity connection, a container manager, and a container manager connection.”

Ross teaches that a *network entity* (e.g., workload) can be a “container,” where a “container is an instance of an image.” Ross, 3:3-6, 3:35; *see also id.*, 3:3-46, 4:34-

52. A POSITA would have known that Docker was a popular and well-known third-party container and that NGINX and Redis (among others) were popular and well-known third-party images running in containers as of the relevant time. Ex. 1003, ¶¶218-223 (citing Ex. 1008 (US 11,159,366), 7:43-8:29; Ex. 1009 (US 11,175,939), 2:54-3:25 Ex. 1010 (<https://www.datadoghq.com/container-report-2020/>); Ex. 1011 (<https://www.datadoghq.com/docker-adoption/>)). A POSITA would have found it obvious and been motivated to utilize a third-party container (e.g., Docker) and/or third-party image (e.g., NGINX, Redis), as doing so is nothing more than combining prior art elements according to known methods to yield predictable results. Ex. 1003, ¶¶222-225. It also would have been obvious to try any of the finite number of third-party containers readily available to a POSITA at the time (i.e., Docker, CoreOS Rkt, LXC, OpenVZ, FreeBSD jail, AIX workload partitions, Solaris), as container technology was extremely well-known and commonplace, providing a POSITA with a reasonable expectation of success in doing so. *Id.*

N. Claim 14

1. ***Claim 14[pre]: “A non-transitory computer readable medium having stored thereon instructions for causing a processing circuitry to execute a process for generation of unified graph models for network entities, the process comprising:”***

To the extent the preamble is limiting, because the Ross-Agarwal-Biran combination teaches limitations 14[a]-14[e], the combination also renders obvious “a process for generation of unified graph models for network entities.” *See* Claims

14[a]-14[e]; '896 *Patent*, 3:37-39, Fig. 2, Claim 14; *see also* Claim 1[pre]. Additionally, Ross discloses that computer program instructions, including the software for implementing Ross's disclosed system and methods, may be stored in a "computer readable medium", such as exemplary "mass data storage 630", which is a "non-volatile storage device" for storing data and instructions. *Ross*, 10:7-35, 14:64-15:20; Ex. 1003, ¶¶227-230. Agarwal and Biran's disclosed systems and methods are also implemented in software. *See, e.g., Agarwal*, Abstract, 27:36-59; *Biran*, 9:1-10:4, Figs. 1-2.

2. ***Claim 14[a]: "collecting, for each network entity of a plurality of network entities, network entity data, wherein the network entity data collected for a network entity includes at least a network entity property, wherein the plurality of network entities are deployed in a plurality of cloud computing platforms;"***

Supra, limitation 1[a].

3. ***Claim 14[b]: "genericizing each of the network entities based on the respective collected network entity data to generate a plurality of generic network entities, wherein a generic network entity includes a generic representation of respective network entities from different cloud computing platforms of the plurality of cloud computing platforms;"***

Supra, limitation 1[b].

4. ***Claim 14[c]: "generating at least a network graph, wherein the generated network graph is a multi-dimensional data structure providing a representation of the plurality of generic network entities and relations between the network entities of the plurality of network entities;"***

Supra, limitation 1[c].

5. ***Claim 14[d]: “creating at least one imputed entity, wherein the at least one imputed entity is a generic network entity representing an executed platform functionality, and wherein the executed platform functionality is different than a network entity; and”***

Supra, limitation 1[d].

6. ***Claim 14[e]: “storing the generated network graph.”***

Supra, limitation 1[e].

O. Claim 15

1. ***Claim 15[pre]: “A system for generating unified graph models for network entities, comprising:”***

To the extent the preamble is limiting, because the Ross-Agarwal-Biran combination teaches limitations 15[a]-15[g], the combination also renders obvious *a system for generation of unified graph models for network entities. See Claims 15[a]-15[g]; '896 Patent, 3:37-39, Fig. 2, Claim 15; see also limitation 1[pre].*

2. ***Claim 15[a]: “a processing circuitry; and”***

Ross discloses that the system includes “one or more processor unit(s) 610” (i.e., processing circuitry) connected to other system components via a local microprocessor bus. *Ross*, 10:7-13, 10:21-28, Fig. 6; Ex. 1003, ¶328-330.

3. ***Claim 15[b]: “a memory, the memory containing instructions that, when executed by the processing circuitry, configure the system to:”***

Ross discloses a “main memory 620” containing instructions “for execution by processor unit(s) 610.” *Ross*, 10:13-16, Fig. 6; Ex. 1003, ¶¶333-335. *See also* limitations 15[c]-15[g].

4. ***Claim 15[c]: “collect, for each network entity of a plurality of network entities, network entity data, wherein the network entity data collected for a network entity includes at least a network entity property, wherein the plurality of network entities are deployed in a plurality of cloud computing platforms;”***

Supra, limitation 1[a].

5. ***Claim 15[d]: “genericize each of the network entities based on the respective collected network entity data to generate a plurality of generic network entities, wherein a generic network entity includes a generic representation of respective network entities from different cloud computing platforms of the plurality of cloud computing platforms;”***

Supra, limitation 1[b].

6. ***Claim 15[e]: “generate at least a network graph, wherein the generated network graph is a multi-dimensional data structure providing a representation of the plurality of network entities and relations between the network entities of the plurality of network entities;”***

Supra, limitation 1[c].

7. ***Claim 15[ff]: “creating at least one imputed entity, wherein the at least one imputed entity is a generic network entity representing an executed platform functionality, and wherein the executed platform functionality is different than a network entity; and”***

Supra, limitation 1[d].

8. ***Claim 15[g]: “store the generated network graph.”***

Supra, limitation 1[e].

- P. **Claim 16: “The system of claim 15, wherein the network entity property relates to at least one of: a network entity type, a network entity class, a network entity category, and a network entity configuration.”**

Supra, Claim 2.

- Q. **Claim 17: “The system of claim 15, wherein the system is further configured to: generate a new generic network entity, wherein the new generic network entity includes at least a network entity property of network entities.”**

Supra, Claim 3.

- R. **Claim 18: “The system of claim 15, wherein a new generic network entity includes at least a network entity property of at least one network entity.”**

Supra, Claim 4.

- S. **Claim 19: “The system of claim 15, wherein the system is further configured to: identify at least one cloud computing platform of the plurality of cloud computing platforms or environment functionality which corresponds with a functionality of a generic network entity.”**

Supra, Claim 5.

- T. Claim 20: “The system of claim 19, wherein the generated network graph is a unified representation of the plurality of network entities including the imputed network entities and the generic network entities.”**

Supra, Claim 6.

- U. Claim 21: “The system of claim 15, wherein the generated at least a network graph includes at least one graph vertex, wherein the at least one graph vertex represents at least one of: a network entity, a generic entity, and an imputed entity.”**

Supra, Claim 7.

- V. Claim 22: “The system of claim 21, wherein the at least one graph vertex includes at least a property label, wherein the at least a property label includes at least a description of a property of the at least one graph vertex.”**

Supra, Claim 8.

- W. Claim 23: “The system of claim 21, wherein the generated at least a network graph includes at least one graph edge, wherein the at least one graph edge is a connection between two graph vertices, and wherein the at least one graph edge represents a relationship between two connected entities, wherein a connected entity is at least one of: a network entity, a generic entity, and an imputed entity.”**

Supra, Claim 9.

- X. Claim 24: “The system of claim 23, wherein the at least one graph edge includes at least one of: a property label, and a directionality indicator, wherein a property label includes at least a description of a property of the at least one graph edge, and wherein a directionality indicator includes at least a description of a direction of the at least one graph edge.”**

Supra, Claim 10.

- Y. Claim 25: “The system of claim 15, wherein the system is further configured to: return at least a visualization of the network graph.”**

Supra, Claim 11.

- Z. Claim 26: “The system of claim 15, wherein the system is further configured to: store the generated network graph in a graph database.”**

Supra, Claim 12.

- AA. Claim 27: “The system of claim 15, wherein a network entity of the plurality of network entities includes any one of: a private endpoint, a transit gateway, a tag-based ruleset, an entity configured to implement a tag-based ruleset, a container-management service, a container-management application, a third-party container, a third-party image, a web-access firewall, a firewall implementation, a multi-entity connection, a cross-entity connection, a container manager, and a container manager connection.”**

Supra, Claim 13.

- BB. Claim 28: “The method of claim 3, wherein the generic network entity represents at least an entity configured to apply security rules.”**

Ross teaches that a security policy is deployed in the cloud computing environment. *Ross*, 8:4-18, 8:49-56, 9:55-62. Agarwal teaches that entities can be grouped based on the “security group” to which they belong. *Supra*, limitation 1[b]. A POSITA would have understood that a grouping of entities that share the same security group is a generic entity that represents an entity configured to apply security rules. Ex. 1003, ¶¶510-513, 99-102. A POSITA would have found it obvious and been motivated to implement the Ross-Agarwal combination to group

entities based on the security group. *Supra*, limitation 1[b]. Ex. 1003, ¶¶514, 103-107.

CC. Claim 29: “The system of claim 17, wherein the generic network entity represents at least an entity configured to apply security rules.”

Supra, Claim 28.

VI. CONCLUSION

For the foregoing reasons, Petitioner respectfully requests institution of *inter partes* review of the Challenged Claims.

Respectfully submitted,

ERISE IP, P.A.

BY: /s/ Robin A. Snader
Robin A. Snader, Reg. No. 66,085
robin.snader@eriseip.com
7015 College Boulevard, Suite 700
Overland Park, Kansas 66211
(913) 777-5600 Telephone
(913) 777-5601 Facsimile

COUNSEL FOR PETITIONER

VII. MANDATORY NOTICES UNDER 37 C.F.R. § 42.8(A)(1)

A. Real Party-In-Interest

Petitioner is the real party-in-interest. 37 C.F.R. § 42.8(b)(1).

B. Related Matters

The '896 Patent is presently asserted in the stayed district court litigation: Orca Security Ltd. v. Wiz, Inc., No. 1:23-CV-00758 (D. Del.). The '896 Patent is not currently and has not been subject to any other district court litigation, inter partes review, reexamination, or other post-grant proceeding.

C. Lead and Back-Up Counsel

Petitioner provides the following designation and service information for lead and back-up counsel. 37 C.F.R. § 42.8(b)(3) and (b)(4).

Lead Counsel	Back-Up Counsel
Robin A. Snader (Reg. No. 66,085) robin.snader@eriseip.com PTAB@eriseip.com	Eric A. Buresh (Reg. No. 50,394) eric.buresh@eriseip.com
<u>Postal and Hand-Delivery Address:</u> ERISE IP, P.A. 7015 College Blvd., Suite 700 Overland Park, Kansas 66211 Telephone: (913) 777-5600 Fax: (913) 777-5601	Kathleen Fitterling (Reg. No 62,950) kathleen.fitterling@eriseip.com
	Christopher Schmidt (Reg. No. 63,982) chris.schmidt@eriseip.com
	<u>Postal and Hand-Delivery Address:</u> ERISE IP, P.A. 7015 College Blvd., Suite 700 Overland Park, Kansas 66211 Telephone: (913) 777-5600 Fax: (913) 777-5601

D. 37 C.F.R. § 42.8(b)(4) – Service Information

Please address all correspondence to the lead and back-up counsel as shown above. Petitioner consents to electronic service by e-mail at the e-mail addresses provided above.

CLAIM LISTING APPENDIX
U.S. Patent No. 11,929,896 for Claims 1-29

Claim Designation	Claim Language
Claim 1[P]	A method for generation of unified graph models for network entities, comprising:
Claim 1[a]	collecting, for each network entity of a plurality of network entities, network entity data, wherein the network entity data collected for a network entity includes at least a network entity property, wherein the plurality of network entities are deployed in a plurality of cloud computing platforms;
Claim 1[b]	genericizing each of the network entities based on the respective collected network entity data to generate a plurality of generic network entities, wherein a generic network entity includes a generic representation of respective network entities from different cloud computing platforms of the plurality of cloud computing platforms;
Claim 1[c]	generating at least a network graph, wherein the generated network graph is a multi-dimensional data structure providing a representation of the plurality of generic network entities and relations between the generic network entities of the plurality of network entities; and
Claim 1[d]	creating at least one imputed entity, wherein the at least one imputed entity is a generic network entity representing an executed platform functionality, and wherein the executed platform functionality is different than a network entity; and
Claim 1[e]	storing the generated network graph.
Claim 2	The method of claim 1, wherein the network entity property relates to at least one of: a network entity type, a network entity class, a network entity category, and a network entity configuration.
Claim 3[P]	The method of claim 1, wherein genericizing each of the network entities further comprises:
Claim 3[a]	generating a new generic network entity, wherein the new generic network entity includes at least a network entity property of network entities.
Claim 4[P]	The method of claim 1,

Claim Designation	Claim Language
Claim 4[a]	wherein a new generic network entity includes at least a network entity property of at least one network entity.
Claim 5[P]	The method of claim 1, wherein creating at least one imputed entity comprises:
Claim 5[a]	identifying at least one cloud computing platform of the plurality of cloud computing platforms or environment functionality which corresponds with a functionality of a generic network entity.
Claim 6	The method of claim 5, wherein the generated network graph is a unified representation of the plurality of network entities including the imputed network entities and the generic network entities.
Claim 7	The method of claim 1, wherein the generated network graph includes at least one graph vertex, wherein the at least one graph vertex represents at least one of: a network entity, a generic entity, and an imputed entity.
Claim 8	The method of claim 7, wherein the at least one graph vertex includes at least a property label, wherein the at least a property label includes at least a description of a property of the at least one graph vertex.
Claim 9	The method of claim 7, wherein the generated at least a network graph includes at least one graph edge, wherein the at least one graph edge is a connection between two graph vertices, and wherein the at least one graph edge represents a relationship between two connected entities, wherein a connected entity is at least one of: a network entity, a generic entity, and an imputed entity.
Claim 10	The method of claim 9, wherein the at least one graph edge includes at least one of: a property label, and a directionality indicator, wherein a property label includes at least a description of a property of the at least one graph edge, and wherein a directionality indicator includes at least a description of a direction of the at least one graph edge.
Claim 11[P]	The method of claim 1, wherein storing the generated network graph further comprises:
Claim 11[a]	returning at least a visualization of the generated at least one network graph.

Claim Designation	Claim Language
Claim 12[P]	The method of claim 1, wherein storing the generated at network graph further comprises:
Claim 12[a]	storing the generated network graph in a graph database.
Claim 13	The method of claim 1, wherein a network entity of the plurality of network entities includes any one of: a private endpoint, a transit gateway, a tag-based ruleset, an entity configured to implement a tag-based ruleset, a container-management service, a container-management application, a third-party container, a third-party image, a web-access firewall, a firewall implementation, a multi-entity connection, a cross-entity connection, a container manager, and a container manager connection.
Claim 14[P]	A non-transitory computer readable medium having stored thereon instructions for causing a processing circuitry to execute a process for generation of unified graph models for network entities, the process comprising:
Claim 14[a]	collecting, for each network entity of a plurality of network entities, network entity data, wherein the network entity data collected for a network entity includes at least a network entity property, wherein the plurality of network entities are deployed in a plurality of cloud computing platforms;
Claim 14[b]	genericizing each of the network entities based on the respective collected network entity data to generate a plurality of generic network entities, wherein a generic network entity includes a generic representation of respective network entities from different cloud computing platforms of the plurality of cloud computing platforms;
Claim 14[c]	generating at least a network graph, wherein the generated network graph is a multi-dimensional data structure providing a representation of the plurality of generic network entities and relations between the network entities of the plurality of network entities;
Claim 14[d]	creating at least one imputed entity, wherein the at least one imputed entity is a generic network entity representing an executed platform functionality, and wherein the executed platform functionality is different than a network entity; and

Claim Designation	Claim Language
Claim 14[e]	storing the generated network graph.
Claim 15[P]	A system for generating unified graph models for network entities, comprising:
Claim 15[a]	a processing circuitry; and
Claim 15[b]	a memory, the memory containing instructions that, when executed by the processing circuitry, configure the system to:
Claim 15[c]	collect, for each network entity of a plurality of network entities, network entity data, wherein the network entity data collected for a network entity includes at least a network entity property, wherein the plurality of network entities are deployed in a plurality of cloud computing platforms;
Claim 15[d]	genericize each of the network entities based on the respective collected network entity data to generate a plurality of generic network entities, wherein a generic network entity includes a generic representation of respective network entities from different cloud computing platforms of the plurality of cloud computing platforms;
Claim 15[e]	generate at least a network graph, wherein the generated network graph is a multi-dimensional data structure providing a representation of the plurality of network entities and relations between the network entities of the plurality of network entities;
Claim 15[f]	creating at least one imputed entity, wherein the at least one imputed entity is a generic network entity representing an executed platform functionality, and wherein the executed platform functionality is different than a network entity; and
Claim 15[g]	store the generated network graph.
Claim 16	The system of claim 15, wherein the network entity property relates to at least one of: a network entity type, a network entity class, a network entity category, and a network entity configuration.
Claim 17[P]	The system of claim 15, wherein the system is further configured to:

Claim Designation	Claim Language
Claim 17[a]	generate a new generic network entity, wherein the new generic network entity includes at least a network entity property of network entities.
Claim 18[P]	The system of claim 15,
Claim 18[a]	wherein a new generic network entity includes at least a network entity property of at least one network entity.
Claim 19[P]	The system of claim 15, wherein the system is further configured to:
Claim 19[a]	identify at least one cloud computing platform of the plurality of cloud computing platforms or environment functionality which corresponds with a functionality of a generic network entity.
Claim 20	The system of claim 19, wherein the generated network graph is a unified representation of the plurality of network entities including the imputed network entities and the generic network entities.
Claim 21	The system of claim 15, wherein the generated at least a network graph includes at least one graph vertex, wherein the at least one graph vertex represents at least one of: a network entity, a generic entity, and an imputed entity.
Claim 22	The system of claim 21, wherein the at least one graph vertex includes at least a property label, wherein the at least a property label includes at least a description of a property of the at least one graph vertex.
Claim 23	The system of claim 21, wherein the generated at least a network graph includes at least one graph edge, wherein the at least one graph edge is a connection between two graph vertices, and wherein the at least one graph edge represents a relationship between two connected entities, wherein a connected entity is at least one of: a network entity, a generic entity, and an imputed entity.
Claim 24	The system of claim 23, wherein the at least one graph edge includes at least one of: a property label, and a directionality indicator, wherein a property label includes at least a description of a property of the at least one graph edge, and wherein a directionality indicator includes at least a description of a direction of the at least one graph edge.

Claim Designation	Claim Language
Claim 25[P]	The system of claim 15, wherein the system is further configured to:
Claim 25[a]	return at least a visualization of the network graph/
Claim 26[P]	The system of claim 15, wherein the system is further configured to:
Claim 26[a]	store the generated network graph in a graph database.
Claim 27	The system of claim 15, wherein a network entity of the plurality of network entities includes any one of: a private endpoint, a transit gateway, a tag-based ruleset, an entity configured to implement a tag-based ruleset, a container-management service, a container-management application, a third-party container, a third-party image, a web-access firewall, a firewall implementation, a multi-entity connection, a cross-entity connection, a container manager, and a container manager connection.
Claim 28	The method of claim 3, wherein the generic network entity represents at least an entity configured to apply security rules.
Claim 29	The system of claim 17, wherein the generic network entity represents at least an entity configured to apply security rules.

APPENDIX OF EXHIBITS

Exhibit 1001	U.S. Patent No. 11,929,896 (“ <i>The ’896 Patent</i> ”)
Exhibit 1002	File History of the ’896 Patent
Exhibit 1003	Declaration of Dr. William C. Easttom II, CV attached as Appendix
Exhibit 1004	U.S. Patent No. 11,863,580 to Ross et al. (“ <i>Ross</i> ”)
Exhibit 1005	U.S. Patent No. 9,800,470 to Agarwal et al. (“ <i>Agarwal</i> ”)
Exhibit 1006	U.S. Patent No. 10,447,553 to Biran et al. (“ <i>Biran</i> ”)
Exhibit 1007	U.S. Patent Publication 2019/0190778 to Easterling et al.
Exhibit 1008	U.S. Patent No. 11,159,366 to Gawade et al.
Exhibit 1009	U.S. Patent No. 11,175,939 to Kumatagi et al.
Exhibit 1010	https://www.datadoghq.com/container-report-2020/ , retrieved June 3, 2025
Exhibit 1011	https://www.datadoghq.com/docker-adoption/ , retrieved June 3, 2025
Exhibit 1012	Stipulation and Order for Stay Pending <i>Inter Partes</i> Review, Case No. 23-00758-JLH-SRF, Dkt 233
Exhibit 1013	https://online.eou.edu/resources/article/understanding-cloud-computing-basics/ , retrieved May 29, 2025
Exhibit 1014	https://web2.qatar.cmu.edu/~msakr/15319-s10/lectures/lecture02.pdf , January 2010
Exhibit 1015	https://ui.adsabs.harvard.edu/abs/2009LNCS.5931..626Q/abstract , retrieved May 29, 2025
Exhibit 1016	The Introduction of Big Data in Cloud Computing, https://red.mnstate.edu/cgi/viewcontent.cgi?article=1021&context=sac , retrieved May 29, 2025
Exhibit 1017	https://www.webopedia.com/reference/cloud-computing-guide/ , retrieved May 29, 2025
Exhibit 1018	https://www.cnet.com/culture/study-cloud-computing-to-brighten-future-of-data-centers/ , retrieved May 29, 2025
Exhibit 1019	https://www.microsoft.com/en-us/research/publication/energy-aware-consolidation-for-cloud-computing/ , retrieved May 29, 2025
Exhibit 1020	https://www.cs.princeton.edu/courses/archive/fall10/cos561/papers/AboveClouds09.pdf , February 10, 2009
Exhibit 1021	https://www.pewresearch.org/internet/2008/09/12/use-of-cloud-computing-applications-and-services/ , retrieved May 29, 2025

Exhibit 1022	https://nvlpubs.nist.gov/nistpubs/legacy/sp/nistspecialpublication800-145.pdf , September 2011
Exhibit 1023	https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.500-291r2.pdf , July 2013
Exhibit 1024	Deo, N. (2016). Graph theory with applications to engineering and computer science. Courier Dover Publications.
Exhibit 1025	Bondy, J. A., & Murty, U. S. R. (1976). Graph theory with applications (Vol. 290). London: Macmillan.
Exhibit 1026	Bollobás, B. (1998). Modern graph theory (Vol. 184). Springer Science & Business Media.
Exhibit 1027	Bondy, J. A., & Murty, U. S. R. (2008). Graph theory. Springer Publishing Company, Incorporated.
Exhibit 1028	Gross, J. L., Yellen, J., & Anderson, M. (2018). Graph theory and its applications. Chapman and Hall/CRC.
Exhibit 1029	Sadavare, A. B., & Kulkarni, R. V. (2012). A review of application of graph theory for network. International Journal of Computer science and Information technologies, 3(6), 5296-5300.
Exhibit 1030	Hayes, B. (2000). Computing science: Graph theory in practice: Part I. American Scientist, 88(1), 9-13.
Exhibit 1031	Postel, J. B., & Farber, D. J. (1976). Graph modeling of computer communications protocols.
Exhibit 1032	McQuillan, J. M. (1977). Graph theory applied to optimal connectivity in computer networks. ACM SIGCOMM Computer Communication Review, 7(2), 13-41.
Exhibit 1033	Uhr, L. (1981). Compounding denser (d, k) graph architectures for computer networks. University of Wisconsin-Madison Department of Computer Sciences.
Exhibit 1034	Boesch, F. T. (1988, June). A survey and introduction to network reliability theory. In IEEE International Conference on Communications,-Spanning the Universe. (pp. 678-682). IEEE.
Exhibit 1035	https://www.metrostate.edu/academics/courses/math-215 , retrieved May 29, 2025
Exhibit 1036	https://sites.ohio.edu/catalog-archive/97-98/colleges/comsci.htm , retrieved May 29, 2025
Exhibit 1037	https://www.utrgv.edu/_files/documents/academics/legacy-catalogs/utpa%201998-2000%20undergraduate%20catalog.pdf , 2000

Exhibit 1038	https://www.csuohio.edu/sites/default/files/ugradcatalog2000-2002.pdf , 2002
Exhibit 1039	https://cvw.cac.cornell.edu/parallel/hpc/nodes , retrieved May 29, 2025
Exhibit 1040	http://users.ece.northwestern.edu/~rberry/ECE333/Lectures/lec1.pdf , 2002
Exhibit 1041	https://www.ou.edu/research/electron/internet/zen-glos.htm , retrieved May 29, 2025
Exhibit 1042	Microsoft Press. (2002). Microsoft computer dictionary. Microsoft Press.
Exhibit 1043	https://hpcdocs.hpc.arizona.edu/software/containers/what_are_containers/ , retrieved May 29, 2025
Exhibit 1044	https://rc-docs.northeastern.edu/en/latest/containers/index.html , retrieved May 29, 2025
Exhibit 1045	https://docs.rc.uab.edu/workflow_solutions/getting_containers/ , retrieved May 29, 2025
Exhibit 1046	https://insights.sei.cmu.edu/blog/11-leading-practices-when-implementing-a-container-strategy/ , retrieved May 29, 2025
Exhibit 1047	https://www.chpc.utah.edu/documentation/software/containers.php , retrieved May 29, 2025
Exhibit 1048	https://www.docker.com/blog/docker-11-year-anniversary/ , retrieved May 29, 2025
Exhibit 1049	https://kubernetes.io/blog/2024/06/06/10-years-of-kubernetes/ , retrieved May 29, 2025
Exhibit 1050	https://d2iq.com/blog/brief-history-containers , retrieved May 29, 2025
Exhibit 1051	Webster's New World Dictionary and Thesaurus Second Edition (2002).
Exhibit 1052	U.S. Patent Publication 2020/0382560A1 to Woolward et al.
Exhibit 1053	Amended Joint Claim Construction Chart, Case No. 23-00758, Dkt 222.

CERTIFICATION OF WORD COUNT

The undersigned certifies pursuant to 37 C.F.R. §42.24 that the foregoing Petition for *Inter Partes* Review, excluding any table of contents, mandatory notices under 37 C.F.R. §42.8, certificates of service or word count, or appendix of exhibits, contains 10,586 words according to the word-processing program used to prepare this document (Microsoft Word).

Dated: June 4, 2025

ERISE IP, P.A.

BY: /s/ Robin A. Snader
Robin A. Snader, Reg. No. 66,085
robin.snader@eriseip.com
7015 College Blvd., Suite 700
Overland Park, KS 66211
P: (913) 777-5600
F: (913) 777-5601

COUNSEL FOR PETITIONER

**CERTIFICATE OF SERVICE ON PATENT OWNER
UNDER 37 C.F.R. § 42.105**

Pursuant to 37 C.F.R. §§ 42.6(e) and 42.105(b), the undersigned certifies that on June 4, 2024, a complete and entire copy of this Petition for *Inter Partes* Review and Exhibits were provided via Federal Express to the Patent Owner by serving the correspondence address of record for the '896 Patent:

M&B IP Analysts, LLC
150 Morristown Road
Suite 205
Bernardsville, NJ 07924-2626

Further, a courtesy copy of this Petition for *Inter Partes* Review was sent via email to Patent Owner's litigation counsel:

Frederick L. Cottrell, III (cottrell@rlf.com)
Kelly E. Farnan (farnan@rlf.com)
Christine Dealy Haynes (haynes@rlf.com)
Alex Miller (alex.miller@wsgr.com)
Callie Davidson (ccdavidson@wsgr.com)
Jordan R. Jaffe (jjaffe@wsgr.com)
Lisa D. Zang (lzang@wsgr.com)
Praatika Prasad (pprasad@wsgr.com)

Respectfully submitted,

ERISE IP, P.A.

BY: /s/ Robin A. Snader
Robin A. Snader, Reg. No. 66,085
robin.snader@eriseip.com
7015 College Blvd., Suite 700
Overland Park, KS 66211

P: (913) 777-5600
F: (913) 777-5601

COUNSEL FOR PETITIONER