

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

NETSKOPE, INC.,

Petitioner,

v.

FORTINET, INC.,

Patent Owner.

PTAB Case No. IPR2026-00473

U.S. Patent No. 11,290,527

**PETITION FOR *INTER PARTES* REVIEW
OF U.S. PATENT NO. 11,290,527**

TABLE OF CONTENTS

EXHIBIT LIST	iv
I. INTRODUCTION	1
II. MANDATORY NOTICES	2
A. Real party in interest.....	2
B. Related matters	2
C. Counsel and service information.....	2
D. Fees.....	3
III. IPR REQUIREMENTS	3
A. Standing.....	3
B. Challenged claims and relief requested.....	3
1. Prior Art	3
2. Grounds for challenge and challenged claims	4
IV. NO DISCRETIONARY DENIAL FACTORS ARE PRESENT.....	4
V. TECHNOLOGY DESCRIPTION.....	5
A. Overview	5
1. Cloud Resource Tagging.....	5
2. Tags to Govern and Secure Cloud Resources.....	7
VI. The Challenged Patent.....	8
A. Alleged Invention.....	8
B. Prosecution History	13

VII. LEVEL OF ORDINARY SKILL	13
VIII. CLAIM CONSTRUCTION	13
IX. SPECIFIC GROUNDS.....	14
A. Ground 1: Foskett renders claims 1, 3, 4, 6, 7, 8, 10, 11, 13, 14, 15, 17, and 20 obvious.	14
1. Claims 1, 8, and 15.....	19
2. Claims 3, 10, and 17: “... wherein the cloud environment comprises a hybrid multi-cloud environment including a private cloud and a plurality of public clouds.”	39
3. Claims 4 and 11: “...wherein said assigning further enables visibility and control for the plurality of cloud resources by the security manager.”	41
4. Claims 6, 13, and 20: “... wherein the plurality of cloud resources comprise virtual machine instances.”	43
5. Claims 7 and 14: “... wherein the security manager comprises a cloud based security management appliance.”	43
B. Ground 2: Combination of Foskett and Whitlock renders claims 2, 5, 9, 12, 16, and 19 obvious.	44
1. Motivations to combine	46
2. Claim 2, 9, and 16: “... facilitating, by the security manager, application of a security policy expressed with reference to one or more unified tags of the plurality of unified tags by retrieving information associated with a particular cloud resource of the plurality of cloud resources from a storage database.”	47
3. Claims 5, 12, and 19: “... wherein the pre-defined global tagging policy identifies the unified tag based on one or more of the information regarding the cloud resource, a	

	location of the resource, and a function of the cloud resource.”	49
C.	Ground 3: Combination of Foskett and Juniper renders claim 18 obvious.	51
1.	Motivations to combine	52
2.	Claim 18: “wherein said assigning enables visibility and control of the plurality of cloud resources via a single pane of glass user interface.”	53
D.	Ground 4: Combination of Kung and Serhiienko renders claims 1, 2, 3, 4, 6, 7, 8, 9, 10, 11, 13, 14, 15, 16, 17, and 20 obvious.	54
1.	Motivations to combine	58
2.	Claims 1, 8, and 15.....	59
3.	Claims 2, 9, and 16:	71
4.	Claims 3, 10, and 17:	73
5.	Claims 4 and 11:	74
6.	Claims 6, 13, and 20:	75
7.	Claims 7 and 14:	76
E.	Ground 5: Combination of Kung, Serhiienko, and Whitlock renders claims 5, 12, and 19 obvious.	77
1.	Motivations to combine	77
2.	Claims 5, 12, and 19:.....	79
F.	Ground 6: Combination of Kung, Serhiienko, and Juniper renders claim 18 obvious.....	81
1.	Motivations to combine	81
2.	Claim 18:.....	82

X. CONCLUSION.....83

EXHIBIT LIST

Exhibit No.	Description
1001	U.S. Patent No. 11,290,527
1002	Supporting Declaration of Dr. Paul Martin
1003	Prosecution History of U.S. Patent No. 11,290,527 (the “527 Prosecution History”)
1004	U.S. Patent Application Publication 2017/0063720 A1 (“Foskett”)
1005	U.S. Patent Application Publication 2018/0276254 A1 (“Whitlock”)
1006	U.S. Patent Application Publication 2020/0059370 A1 (“Juniper”)
1007	U.S. Patent Application Publication US 2018/0234459 A1 (“Kung”)
1008	IEEE Article “Extensible Declarative Management of Cloud Resources across Providers” by O. Serhienko et al. (“Serhienko”)
1009	U.S. Patent 9,444,819 (“Muppidi”)
1010	U.S. Patent 11,171,994 (“Ramachandran”)
1011	“New Amazon EC2 Feature: Resource Tagging” (https://aws.amazon.com/blogs/aws/new-amazon-ec2-feature-resource-tagging/)
1012	“Tagging Best Practices Implement an Effective AWS Resource Tagging Strategy” (https://d1.awsstatic.com/whitepapers/aws-tagging-best-practices.pdf)

Exhibit No.	Description
1013	“Multi-Cloud Tagging Strategies For the Win” (https://securityboulevard.com/2019/10/multi-cloud-tagging-strategies-for-the-win-2/)
1014	“Tag Editor Document History” (https://docs.aws.amazon.com/tag-editor/latest/userguide/doc-history.html)
1015	“Microsoft.Resources tags 2019-10-01” (https://learn.microsoft.com/en-us/azure/templates/microsoft.resources/2019-10-01/tags)
1016	“The NIST Definition of Cloud Computing” (https://nvlpubs.nist.gov/NISTPUBS/LEGACY/SP/NISTSPECIALPUBLICATION800-145.PDF)
1017	“AWS vs Azure vs Google: Cloud Wars” (https://www.cbtnuggets.com/blog/certifications/microsoft/aws-vs-azure-vs-google-cloud-wars-2019)
1018	“Extensible Declarative Management of Cloud Resources across Providers” (https://ieeexplore.ieee.org/document/8752784)

I. INTRODUCTION

Cloud computing took off in the mid-2000s when companies like Amazon and Google launched on-demand cloud services. Over the next fifteen years the field exploded: new providers entered the market, new services proliferated, and an ecosystem of management tools matured around them. Among the earliest and most fundamental of those tools was tagging—assigning metadata to a cloud resource so that service providers and customers could organize, track, and govern deployments. As enterprises moved to multi-cloud environments, tagging naturally extended across providers, and by 2020 cross-provider tag management was standard industry practice.

U.S. Patent No. 11,290,527 (“the ’527 Patent”)—filed in June 2020 and issued in 2022, well after all of this was commonplace—claims nothing more than this familiar concept. Yet the patent sailed through prosecution without a single office action and without any genuine examination of the prior art. None of the references discussed in this Petition were before the Examiner. As the Board will see upon reviewing those references, every element of the alleged invention was well known throughout the industry long before the ’527 Patent.

Accordingly, Netskope, Inc. (“Petitioner”) requests *inter partes* review of claims 1-20 of the ’527 Patent. For the reasons detailed below, the Board should institute IPR and cancel the challenged claims.

II. MANDATORY NOTICES

A. Real party in interest

Netskope, Inc. is the sole real party-in-interest.

B. Related matters

Patentee originally asserted the '527 Patent against Petitioner in the United States District Court for the Northern District of California, Case No. 4-25-cv-02360-HSG. That claim has now been severed into Case No. 4:26-cv-08886-HSG.

C. Counsel and service information

Lead Counsel: Thomas N. Millikan (Reg. No. 72,316).

Back-up counsel: Joseph P. Reid (Reg. No. 72,317), Andrew Klein (*pro hac vice* forthcoming), Usayd Siddiqi (Reg. No. 80,999).

These attorneys can be reached by mail at Sheppard, Mullin, Richter & Hampton LLP, 12275 El Camino Real, Suite 100, San Diego, California, 92130, by phone at (858) 720-8900, and by fax at (858) 509-3691.

Petitioner consents to electronic service. All services and communications to the attorneys listed above may be sent to: LegalTm-NetskopeFortinetII-IPRs@sheppard.com.

A Power of Attorney for Petitioner has been filed.

D. Fees

Petitioner paid the required fees. The Office is authorized to charge any fee deficiency, or credit any overpayment, to Deposit Account No. 50-4561.

III. IPR REQUIREMENTS

A. Standing

The '527 Patent is available for IPR and Petitioner is not barred or estopped from challenging the claims on the grounds herein.

B. Challenged claims and relief requested

Petitioner respectfully requests cancellation of the challenged claims under §103.

1. Prior Art

Petitioner relies on the references listed in the Table of Exhibits, including:

- U.S. Patent Application Publication 2017/0063720 A1 (“Foskett”)
- U.S. Patent Application Publication 2018/0276254 A1 (“Whitlock”)
- U.S. Patent Application Publication 2020/0059370 A1 (“Juniper”)
- U.S. Patent Application Publication US 2018/0234459 A1 (“Kung”)
- Serhiienko, Gkikopoulos and Spillner, “Extensible Declarative Management of Cloud Resources across Providers,” 2019 19th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGRID), Larnaca, Cyprus, 2019, pp. 678-683 (“Serhiienko”)

Petitioner also relies on the declaration of Dr. Paul Martin, the Chief Scientist at Harbor Experts and a part-time lecturer in the Department of Computer Science at Johns Hopkins University.

2. Grounds for challenge and challenged claims

Petitioner requests *inter partes* review and cancelation of the challenged claims on the following grounds:

Ground	Basis	Reference(s)	Challenged Claims
1	§103	Foskett	1, 3, 4, 6, 7, 8, 10, 11, 13, 14, 15, 17, and 20
2	§103	Foskett and Whitlock	2, 5, 9, 12, 16, and 19
3	§103	Foskett and Juniper	18
4	§103	Kung and Serhienko	1, 2, 3, 4, 6, 7, 8, 9, 10, 11, 13, 14, 15, 16, 17, and 20
5	§103	Kung, Serhienko, and Whitlock	5, 12, and 19
6	§103	Kung, Serhienko, and Juniper	18

IV. NO DISCRETIONARY DENIAL FACTORS ARE PRESENT

Petitioner submits that *inter partes* review is warranted here because, in addition to the claimed invention being known before the '527 Patent, there are no discretionary factors that weigh against review. For example:

- 1) None of the prior art discussed herein was before the Examiner during prosecution.
- 2) There are no settled expectations because the '527 Patent issued only four years ago.
- 3) No schedule is set, including no set trial date, in the parallel district court proceeding.
- 4) Petitioner has not taken inconsistent positions on claim construction—indeed, claim construction proceedings have not begun in the parallel district court proceeding.
- 5) Netskope is a U.S.-based cloud security company.

V. TECHNOLOGY DESCRIPTION

A. Overview

The challenged claims merely combine the following well-known concepts into a commonly understood configuration for managing cloud resources across providers.

1. Cloud Resource Tagging

Cloud computing refers to on-demand access to a shared pool of configurable computing resources and has been widely understood since at least

2009, when the National Institute of Standards and Technology published its foundational definition. (Ex. 1009, 7:13-8:38.)¹

Tagging emerged alongside cloud computing as the practice of attaching key-value metadata labels to cloud resources for classification and identification. Tagging allowed resources to be identified and organized in various ways and assisted in administration, monitoring, automation, and access control. Tagging was already commonly established before the '527 Patent: “known tagging solutions have been used to facilitate asset management,” and tags could be applied “manually (through a configuration tool), via an automated set of tag rules (rule-based tagging), statically or dynamically, or programmatically.” (Ex. 1009, 10:66–11:7.)

As early as 2010, Amazon Web Services (“AWS”) introduced resource-tagging for Amazon EC2, letting customers attach up to ten key-value pairs to resources and filter and organize them through an API or management console.² In 2018, AWS released a whitepaper to help users develop a tagging strategy to manage AWS resources.³

¹ See <https://nvlpubs.nist.gov/NISTPUBS/LEGACY/SP/NISTSPECIALPUBLICATION800-145.PDF> (Ex. 1016)

² <https://aws.amazon.com/blogs/aws/new-amazon-ec2-feature-resource-tagging/> (Ex. 1011)

³ <https://d1.awsstatic.com/whitepapers/aws-tagging-best-practices.pdf> (Ex. 1012)

By 2019, cloud resources and services had become ubiquitous and organizations managed disparate resources and services “spread across multiple cloud providers and accounts/subscriptions” using a “global tagging policy” to “identify and describe the resources running across one more cloud environments.”⁴ By 2020, resource tagging was standard practice, with AWS and Azure providing dedicated mechanisms for assigning metadata and identifying resources by environment, ownership, purpose, and function.⁵ Tagging thus sits within the well-established field of infrastructure and configuration management. (Ex. 1002, ¶¶ 43-50.)

2. Tags to Govern and Secure Cloud Resources

Using tags for security policy enforcement was well-established before the ’527 Patent. Technologies such as network functions virtualization (‘NFV’) enabled operators to deploy services on off-the-shelf hardware. While virtualization may provide flexibility to network operators and other entities, “several challenges pose difficulties in migrating services to virtualized networks.” (Ex. 1010, 1:37-38.) Among those challenges was the need to apply security

⁴ <https://securityboulevard.com/2019/10/multi-cloud-tagging-strategies-for-the-win-2/> (Ex. 1013)

⁵ See, e.g., <https://docs.aws.amazon.com/tag-editor/latest/userguide/doc-history.html> (Ex. 1014); <https://learn.microsoft.com/en-us/azure/templates/microsoft.resources/2019-10-01/tags> (Ex. 1015)

policies automatically as resources are instantiated, scaled, and updated. To fulfill this need, methods were developed for tag-based security policy creation.

Compliance-driven security management using cloud resource metadata was likewise standard practice with audit logs “made available in relational databases to allow easy querying” to meet well-known regulatory requirements. (Ex. 1009, 1:14-27.) Retrieving audit logs allowed organizations to “obtain visibility into the cloud resources they are managing” (*Id.*, 2:13-16; Ex. 1002, ¶¶ 51-52.)

VI. The Challenged Patent

A. Alleged Invention

The '527 Patent is directed to these well-known concepts. The patent is directed to “[s]ystems and methods for automatically tagging cloud resources that are spread across multiple cloud platforms.” (Ex. 1001, Abstract.)

Figure 1 in the patent (below) depicts a network architecture that includes a security manager (102) with an orchestrator (120) that is connected to private cloud (110), public cloud (112), hybrid cloud (114), and private network (114). (*Id.*, 6:1–21.)

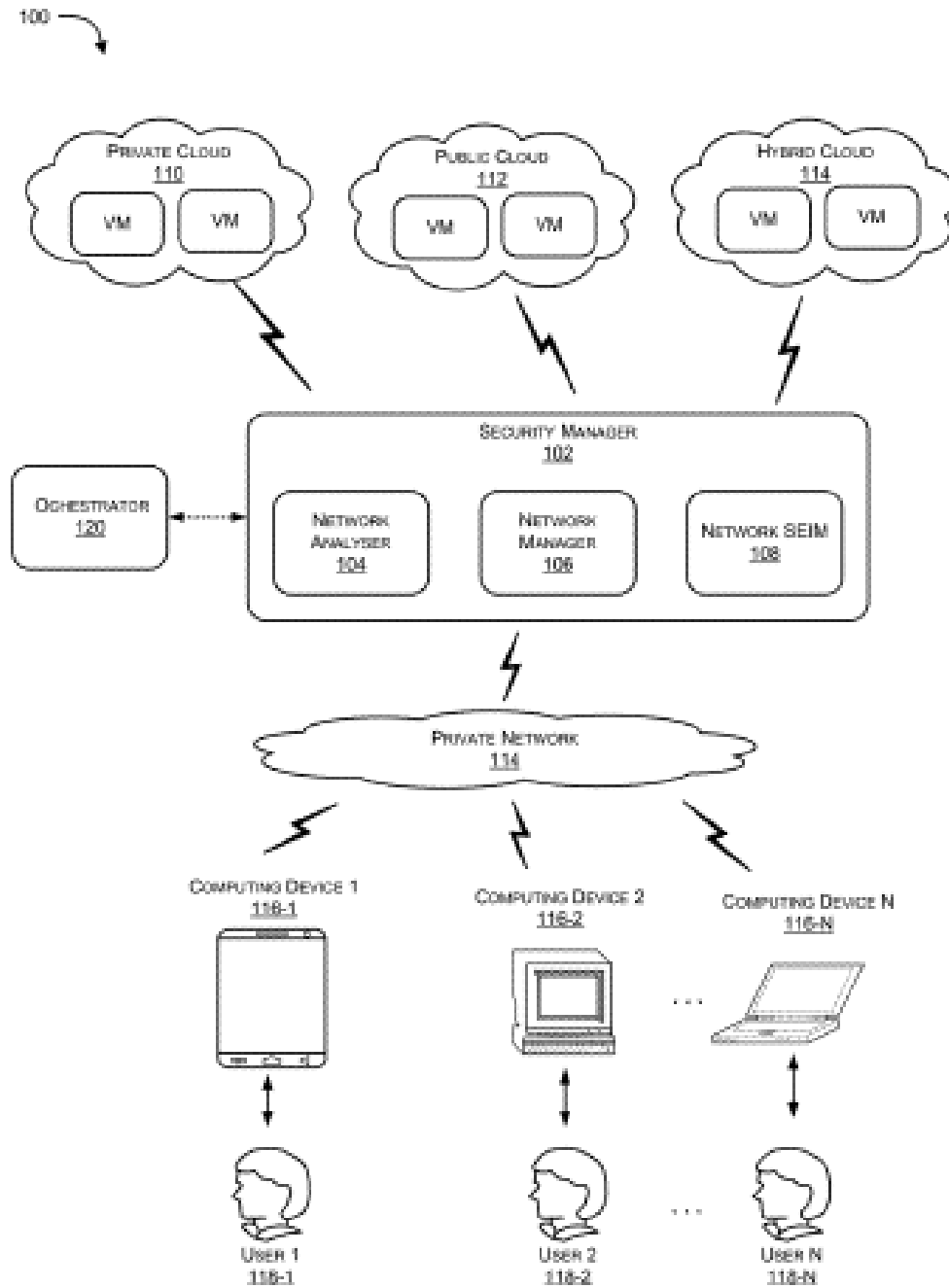


FIG. 1

(*Id.*, Fig. 1.)

Figure 4 in the patent (below) discloses a process flow for tagging cloud resources across providers: the cloud-tagging orchestrator service (“CTOS”) receives information about each cloud provider (402), creates a unified tag for each cloud resource based on a pre-defined global tagging policy (404), and associates the tag with cloud resource in a storage database, (406). The security manager then retrieves tags from the database so the appropriate security policy can be applied (408).

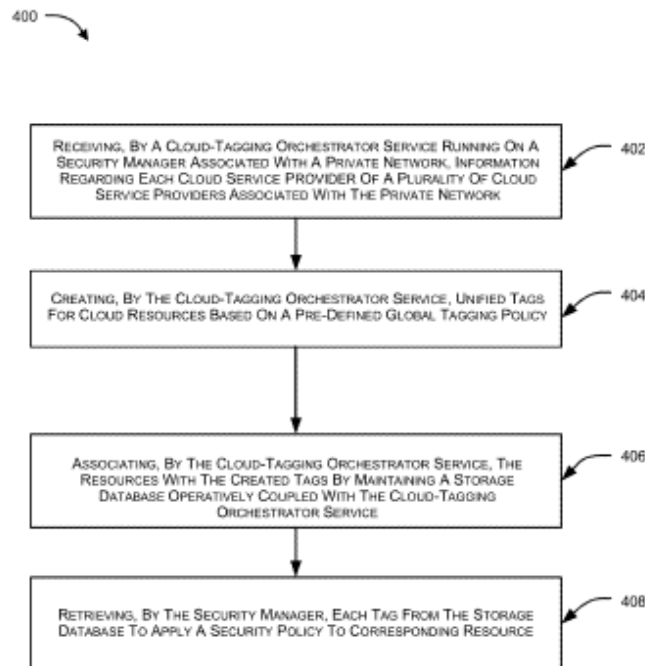


FIG. 4

(Ex. 1001, Fig. 4.)

The '527 Patent describes the security manager (102) to include a range of security-related implementations, encompassing physical, on-premises hardware, software-based systems, and cloud-hosted implementations. In one embodiment, it is an “on-premises management appliance” associated with the network that manages computing devices associated with the same network. (*Id.*, 6:22–25.) In another embodiment, it is “a cloud based security manager that can be implemented in the form of a cloud service.” (*Id.*, 6:25–28.) In yet another embodiment, the CTOS acts as a means for communication for the security manager such that all cloud environments and computing devices interact with the CTOS in order to interact with the security manager, while maintaining connectivity the network. (*Id.*, 6:28–35.) The '527 Patent also describes that “orchestrator 120 may be part of security manager 102 or may be an independent service used by security manager 102.” (*Id.*, 7:6–8.)

Figure 2 in the patent (below), illustrates various components of the security manager (102), including processor(s) (202), memory (204), interface(s) (206), a processing engine (208) comprising an information receiving engine (212), a tagging engine (214), a remediation engine (216), other unit(s) (218), and a database (210). (*Id.*, Fig. 2; 8:48–57; 9:15-42.)

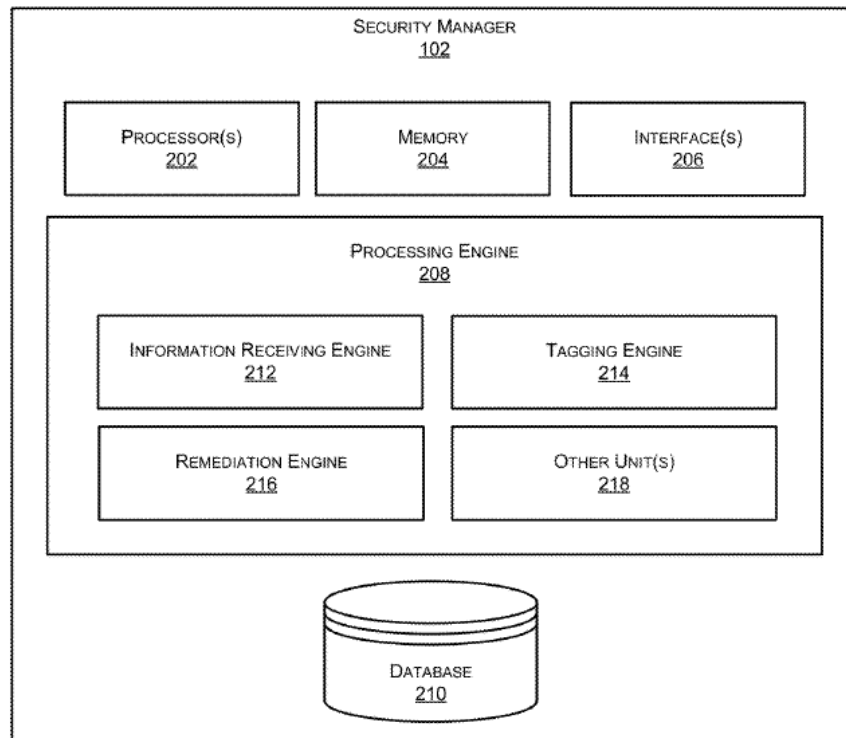


FIG. 2

(Ex. 1001, Fig. 2.)

The information receiving engine (212) and tagging engine (214)—working in conjunction with other components, such as database (210) and other unit(s) (218)—implement the claimed CTOS. In addition, the security manager may also embody components serving broader security functions, such as remediation engine (216), which retrieves tags from database (210) to apply a security policy to the resource (408). (Ex. 1002, ¶¶ 53-58.)

B. Prosecution History

The '527 Patent's filing history contains no office actions and no amendments. The Examiner noted that Pargaonkar (US 2021/0176191) disclosed all limitations of the independent claims except for the “via an application programming interface (API) of a cloud provider of the plurality of cloud providers hosting the resource.” (Ex. 1003 at 0015.)

VII. LEVEL OF ORDINARY SKILL

A POSITA in June 2020 would have had a bachelor's degree in computer science, electrical engineering, or a related field, and at least two years of practical experience with network and data security, including the then-current technologies such as cloud infrastructure management, cloud resource organization, and/or virtual server or resource administration. (Ex. 1002, ¶¶ 33-37.)

VIII. CLAIM CONSTRUCTION

The challenged claims should be construed “in accordance with the ordinary and customary meaning of such claim as understood by one of ordinary skill in the art and the prosecution history pertaining to the patent.” 37 C.F.R. § 42.100(b); *see Phillips v. AWH Corp.*, 415 F.3d 1303 (Fed. Cir. 2005). For this proceeding, Petitioner does not believe any construction is needed. (Ex. 1002, ¶¶ 60-61)

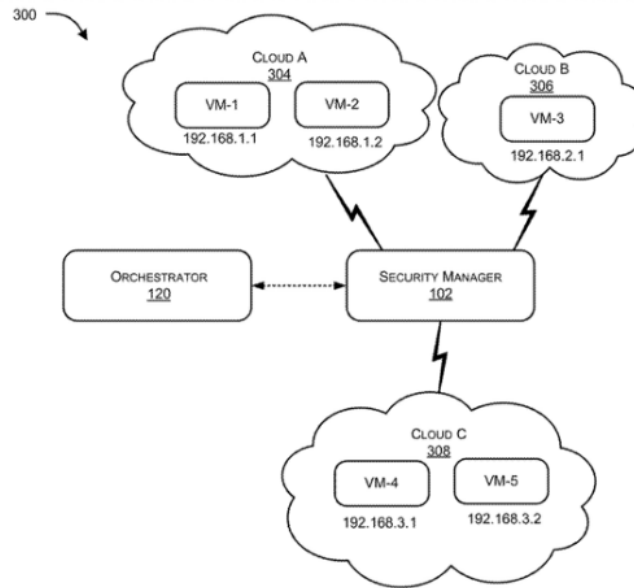
IX. SPECIFIC GROUNDS

A. Ground 1: Foskett renders claims 1, 3, 4, 6, 7, 8, 10, 11, 13, 14, 15, 17, and 20 obvious.

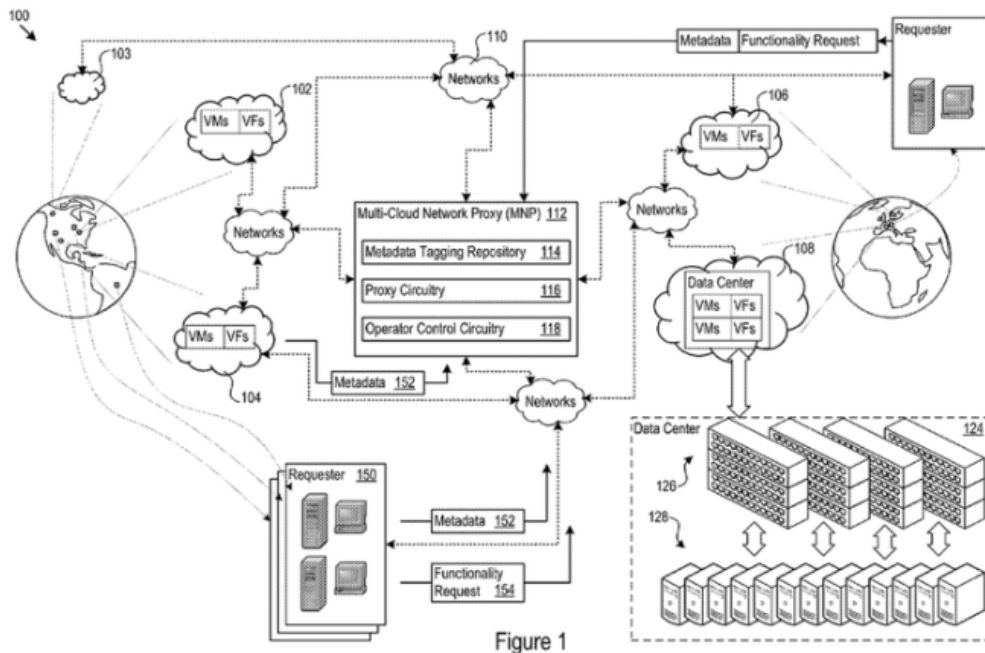
Foskett: Five years before the '527 Patent, Foskett disclosed a system to apply uniform tags across multiple cloud service providers. (Ex. 1004, ¶ 0015.) Foskett disclosed that “[p]rovisioning resources into the cloud is a constantly increasing technical challenge as more cloud service providers emerge, each offering disparate computing platforms, services, assets, supported technical components, and other features. A multi-cloud network proxy controls and normalizes the tagging data applicable across the multiple cloud service providers.” (*Id.*, Abstract.)

Foskett centered on a Multi-Cloud Network Proxy (“MNP”) positioned between cloud service providers and a requester, with a metadata tagging repository (114) and proxy circuitry (116) storing tag identifiers and values across providers with and without native tagging repositories. (*Id.*, ¶¶ 0015, 0031, 0033–0034, 0040.)

Figure 3A from the patent (below) and Figure 1 from Foskett (below) are architecturally similar: a security manager associated with an orchestrator sits in between multiple clouds and manages tagging, analogous to the MNP sitting between multiple networks in Foskett.



(Ex. 1001, Fig. 3A.)



(Ex. 1004 Fig. 1.)

Figure 3B from the '527 Patent (below on left), compared with Figure 6 from Foskett (below on right), show the same pattern: “Universal tags” are organized by “IP Address” and “Cloud Services,” while Foskett’s tag content values are organized by provider and tag identifier. In Foskett, the provider is analogous to the cloud services column, and the tag identifier is a more generalized form of the IP Address column.

IP ADDRESS	CLOUD SERVICES	UNIVERSAL TAG
192.168.1.1	A	CDA11
192.168.1.2	A	CDA12
192.168.2.1	B	CDB21
192.168.3.1	C	CDC31
192.168.3.2	C	CDC32

FIG. 3B

The screenshot shows a web application titled "Resource Metadata Tagging Control". It features three dropdown menus: "Resource Type" (set to "Virtual Machine"), "Client" (set to "Client X"), and "Project" (set to "Laser Array"). Below these is a section titled "Metadata Tags" containing a table with three columns: "Provider", "Tag Identifier", and "Content Value". The table lists five entries: "Red" (Class: Name_001), "Red" (Environment: Power Output: 7), "Red" (Remarks: Location_Secret), "Red" (Specifications: 455 MBps), and "Blue" (Power: Power Output: 9). Navigation controls at the bottom show "1 - 5 of 5 Tags" and "Page 1 of 1".

Provider	Tag Identifier	Content Value
Red	Class	Name_001
Red	Environment	Power Output: 7
Red	Remarks	Location_Secret
Red	Specifications	455 MBps
Blue	Power	Power Output: 9

Figure 6

(Ex. 1001, Fig. 3B; Ex. 1004 Fig. 6.)

Process-wise, the MNP discovered resources and tags by connecting to service provider APIs and sending “discovery requests” to each “target hosting platform” (i.e., the specific cloud platform offered by a given service provider such as AWS, Google Compute Engine, or Microsoft Azure) to obtain identifiers of

deployed resources. (Ex. 1004, ¶ 0018.) Figure 4 in Foskett (below) illustrates this process.

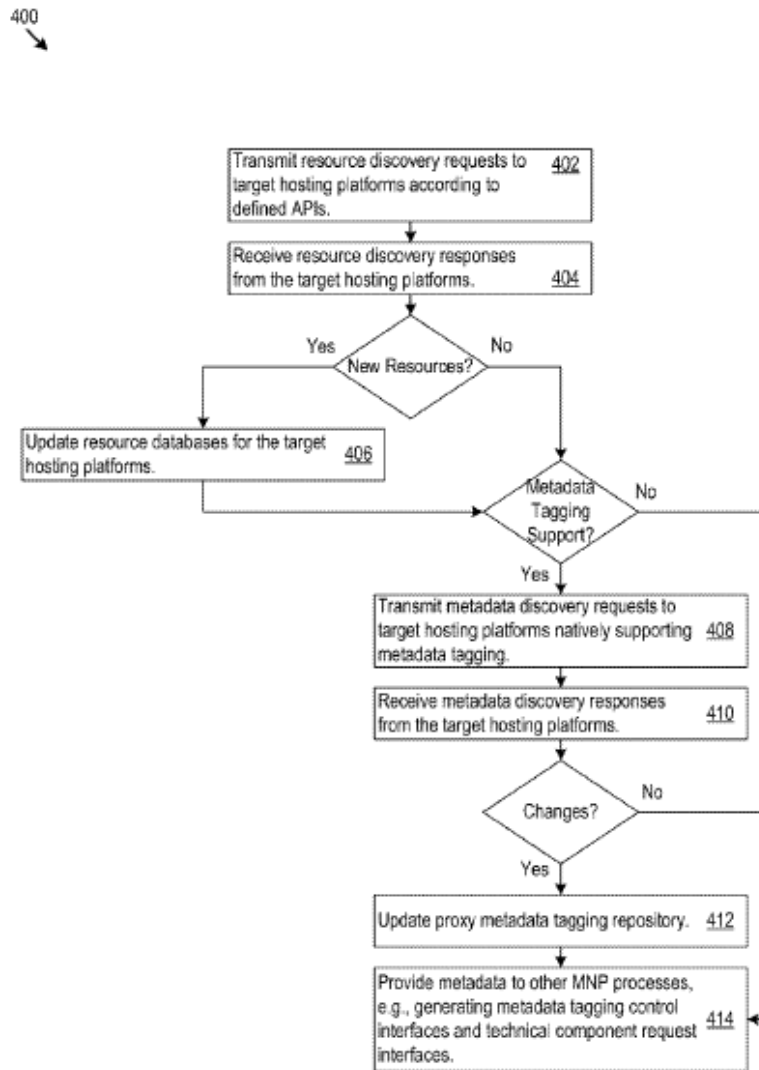


Figure 4

(*Id.*, Fig. 4.)

Foskett clarified that service providers may also be on-premises. “Service providers may additionally or alternatively provide computing resources in ‘on-premises’ hosting platforms, which typically refers to a location with increased privacy and security compared to public cloud resources.” (*Id.*, ¶ 0018.)

The MNP could also transmit a metadata discovery request to the provider’s metadata tagging repository to obtain metadata tagging information (e.g., tag identifiers and tag content). (*Id.*, ¶¶ 0032-0035.) The MNP stored the response in its resource database, adding newly discovered resources and removing deleted ones based on the current state provided. (*Id.*, ¶ 0034.) When updating tags, the MNP established an API connection and transmitted the tags and content from its proxy repository to the platform for synchronization, effectively transmitting the uniform tag back to each provider that natively supports tagging. (*Id.*, ¶¶ 0051, 0055–0063.)

Because cloud providers used inconsistent tag naming conventions and value formats, the MNP maintained a proxy metadata tagging repository as the authoritative, provider-neutral baseline for all tags across every connected provider. (*Id.*, ¶ 0015.) A uniform tag comprised a tag identifier and content value standardized to defined types and constraints, which the MNP validated before

storing and then pushed back to each platform via API to synchronize provider repositories to that baseline. (*Id.*, ¶¶ 0047–0055; Fig. 9, 916–918.)

Foskett disclosed multiple figures representing various aspects and variations of the same invention. If needed, a POSITA would have been motivated to combine features across implementations because, for example, Foskett described Figures 1 and 2 as merely “one of many possible different implementation contexts,” expressly noting its solutions “are not limited” to those figures, and confirmed that “many other implementations are also possible” beyond those “specifically described.” (*Id.*, ¶¶ 0016, 0069; Ex. 1002, ¶¶ 63-70.) Because Foskett’s implementations are predictable variations of the same underlying MNP architecture, combining them does not require a leap of inventiveness. *See Bos. Sci. Scimed, Inc. v. Cordis Corp.*, 554 F.3d 982, 991 (Fed. Cir. 2009).⁶

1. Claims 1, 8, and 15

1[pre]: “A method comprising”

⁶ Where the disputed claim elements are disclosed together in a single embodiment of a single reference, no separate showing of motivation to combine or reasonable expectation of success is required. *See Gen. Elec. Co. v. Raytheon Techs. Corp.*, 983 F.3d 1334, 1352 (Fed. Cir. 2020); *see also Guardant Health, Inc. v. Univ. of Wash.*, No. 2024-1129, 2026 WL 184334, at *1, *4–5 (Fed. Cir. Jan. 23, 2026) (nonprecedential).

Foskett disclosed a method for centrally managing and normalizing metadata tags across multiple cloud service providers. The MNP “provides a central control mechanism for discovering, monitoring, and updating complex tagging data applied to the technical components provisioned with the cloud service providers, regardless of whether any given service provider natively supports metadata tagging.” (Ex. 1004, ¶ 0015; Ex. 1002, ¶ 71.)

8[pre]: “A non-transitory computer-readable storage medium embodying a set of instructions, which when executed by a processing resource of a security manager of a private network, causes the processing resource to perform a method comprising”

Foskett could be implemented on a computer. Specifically, Foskett disclosed that its methods “may be embodied in machine-executable instructions” stored on “a tangible storage medium that is other than a transitory signal, such as a flash memory, a Random Access Memory (RAM), a Read Only Memory (ROM), an Erasable Programmable Read Only Memory (EPROM),” or similar non-transitory media, executed by “an instruction processor.” (Ex. 1004, ¶¶ 0066–0067.) It would have been obvious to a POSITA that the MNP has CRSM, instructions, and other typical computing components because it performs various steps in a computer, as shown in claims 1/8/15 below.

Foskett disclosed that the MNP's infrastructure “may include any combination of hardware, software, firmware, or other logic,” and its implementation “may be distributed,” spanning “multiple distinct system components, such as multiple processors and memories,” and “multiple distributed processing systems.” (*Id.*, ¶¶ 0026, 0068.) Foskett further disclosed that its “system circuitry 204 may include one or more instruction processors 218 and memories 220” where the “memory 220 stores, for example, control instructions 222 and an operating system.” (*Id.*, ¶ 0026; Fig. 2 (below).)

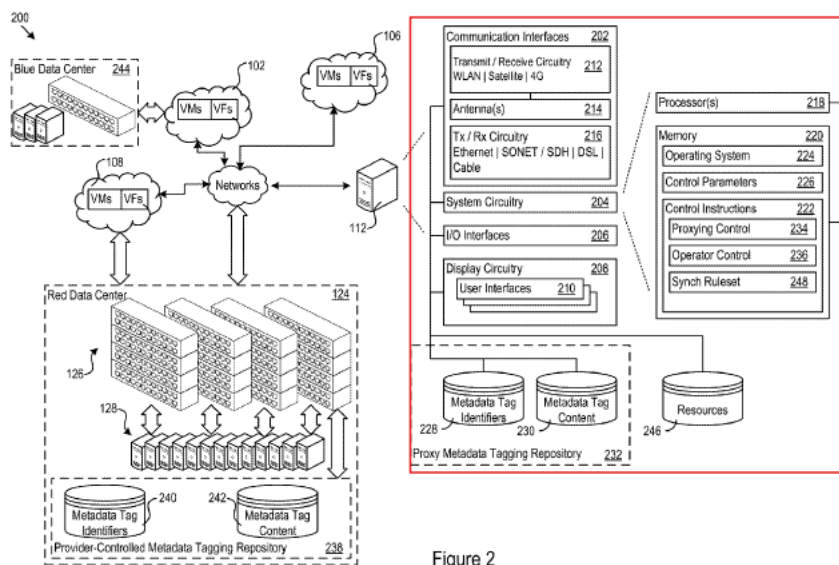


Figure 2

(*Id.*, Fig. 2.)

Foskett disclosed the “processing resource” for the reasons discussed in claims 1[a]/8[a]/15[a] below, because the processing resource, just like the MNP,

is what performs the actual method steps. Foskett disclosed “the security manager of a private network” for the same reasons provided in claims 1[a]/8[a]/15[a] below. (Ex. 1002, ¶¶ 129-133.)

15[pre]: “A multi-tag orchestration system comprising: a processing resource; and a non-transitory computer-readable medium, coupled to the processing resource, having stored therein instructions that when executed by the processing resource cause the processing resource to perform a method comprising:”

Foskett disclosed or rendered obvious a multi-tag orchestration system: “[a] multi-cloud network proxy [that] controls and normalizes the tagging data applicable across multiple cloud service providers,” implemented using the MNP’s processor(s) 218 (a processing resource) coupled to memory 220 (a non-transitory computer-readable medium) storing control instructions 222 that, when executed, cause the processing resource to perform the claimed method. (Ex. 1004, Abstract, ¶ 0026.) For the remaining elements, Foskett disclosed those elements for the same reasons discussed in element 1[Preamble] and element 8[Preamble]. (Ex. 1002, ¶¶ 143-144.)

1[a]/8[a]/15[a]: “receiving, by a cloud-tagging orchestrator service running on a security manager associated with a private network, information regarding each cloud provider of a plurality of cloud providers associated with a cloud environment used by the private network”

Foskett disclosed or rendered obvious this element because the MNP's tagging software (CTOS) runs on the MNP's infrastructure (security manager), which is tied to a resource requester (associated with a private network). That software connects to each of several named cloud providers (plurality of cloud providers) through that provider's own API and receives back (receiving) a discovery response enumerating that provider's resources (information regarding each cloud provider). Those providers, taken together, make up the set of platforms the MNP manages for the requester (a cloud environment used by the private network).

Figure 2 in Foskett (below) depicted the internal architecture of the MNP (Ex. 1004, Fig. 2, ¶¶ 0024–0026.) Foskett explained that the system circuitry (204)—a component of the MNP—“may include any combination of hardware, software, firmware, or other logic.” (*Id.*, ¶ 0026.) This infrastructure of the MNP's system circuitry 204, communication interfaces 202, and I/O interfaces 206, together with the associated supporting software that operates and manages that infrastructure, such as the operating system 224 and any hypervisor, database infrastructure, or other system-level software used to manage and control access to the underlying hardware resources that corresponds to the claimed security manager. In other words, the infrastructure and its supporting system-level

software that together house and operate everything the MNP does are just like the '527 Patent's security manager. (Ex. 1001, 6:22–28.) (Ex. 1002, ¶¶ 73, 75.)

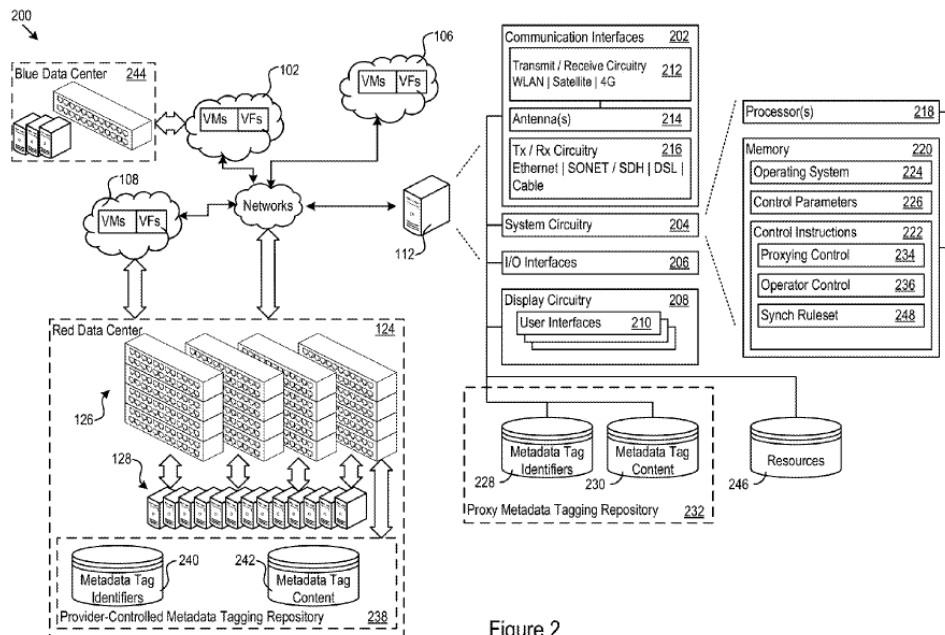


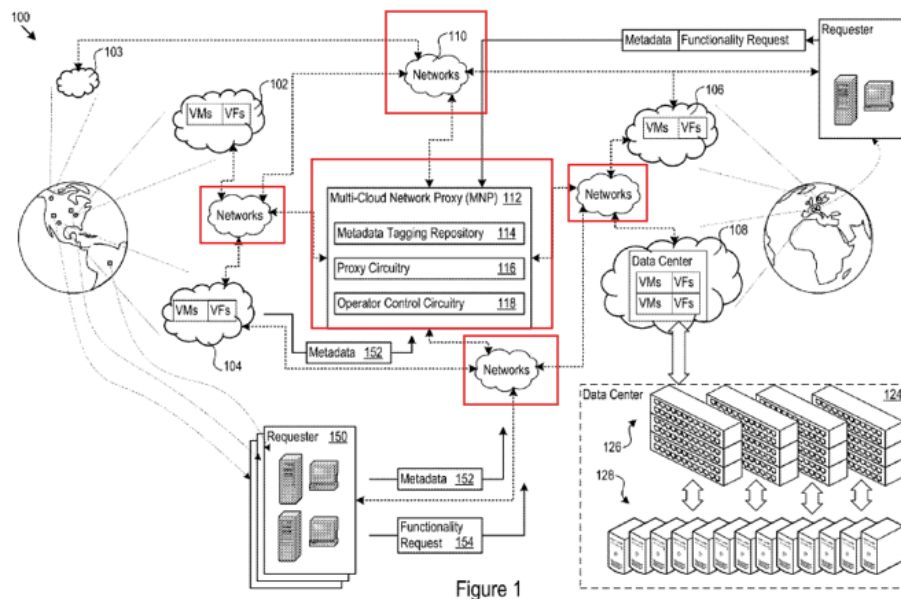
Figure 2

(Ex. 1004, Fig. 2.)

The MNP's tagging-specific software—like its control instructions 222 and central proxying control 234—corresponds to the CTOS. Foskett explained that “[t]he system circuitry 204 may implement the proxy circuitry 116 and the operator control circuitry 118, e.g., as control instructions 222 executed by the processor 218,” and that “[t]he control instructions 222 implement a central proxying control 234 for discovering, monitoring, updating, and proxying complex tagging data applied to the technical components provisioned with the cloud service providers,” as well as “an operator control 236 that dynamically generates

technical component request interfaces that include specifically prioritized metadata tags.” (Ex. 1004, ¶ 0028.) This tagging software *runs* on the MNP’s infrastructure and carries out the discovering, tagging, and proxying functions discussed throughout Foskett’s specification. (Ex. 1002, ¶¶ 74-75.)

The MNP is (or it would have been obvious that it is) “associated with a private network” because it sits between a resource requester and the cloud providers hosting the requester’s resources, acting as the requester’s centralized point of control. (Ex. 1004, Fig. 1.) Figure 1 in Foskett (below) illustrates this.



(*Id.*, Fig. 1.)

Discovery by the MNP can include “obtaining access credentials 326, e.g. logon username/password from an account owner at the resource requester 150,”

after which “[t]he MNP 112 submits the access credentials to the corresponding service provider to access and request the resources and attached metadata (if any) for the resources.” (*Id.*, ¶ 0033.) A POSITA would have understood the “requester” to be a client-side entity requesting a resource from a provider, like an enterprise user of the MNP’s tagging and provisioning services. The MNP then obtained access credentials from the resource requester to submit to the service provider, effectively allowing the MNP to make provisioning requests on the requester’s behalf and for its private network. The MNP is thus associated with the private network of that enterprise, organization, or other end-user. (Ex. 1002, ¶¶ 76-77.)

The MNP receives information about each cloud provider from each one. First, the MNP reaches out to each cloud provider through that provider’s own API: “the MNP 112 may, for instance, connect through service provider application programming interfaces (APIs) 302, 304, and 306 to transmit discovery request messages 308, 310, 312 to the target virtual hosting platforms 314, 316, and 318 of the service providers 102, 106, and 108 respectively.” (Ex. 1004, ¶ 0032.) Second, each provider responds back to the MNP: “[t]he MNP 112 receives discovery responses 320, 322, and 324 enumerating the resources as a result of transmitting the discovery request messages.” (*Id.*) Figure 3 in Foskett (below) illustrated this exchange. (Ex. 1002, ¶ 78.)

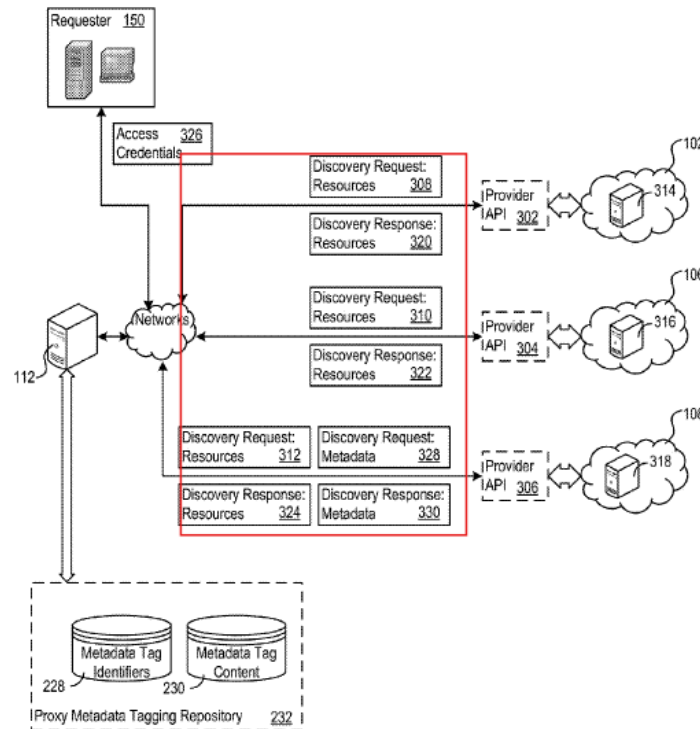


Figure 3

(Ex. 1004, Fig. 3.)

Foskett also applied to a “plurality of cloud providers,” disclosing that the MNP is designed from the outset to interact with more than one cloud provider at a time, and naming several real-world examples: “[e]xamples of service providers include Amazon, Google, Microsoft, and Accenture, who offer different target hosting platforms, e.g., Amazon Web Services (AWS), Google Compute Engine (GCE), Microsoft Azure (Azure), Accenture Cloud Platform (ACP), and Windows

Azure Pack (WAP) for on-premise cloud implementations.” (*Id.*, ¶ 0018; 0019 (“A multi-cloud network proxy (MNP) 112 makes complex cloud architectural provisioning and execution decisions across multiple cloud services, taking into account the global network architecture 100, the various service provider locations and capabilities, and other factors.”).) Foskett referred to each of these providers’ own hosting environments as a “target hosting platform.” Figure 3 in Foskett (below) disclosed that the MNP contacted several different providers simultaneously. (*Id.*, ¶ 0032; Ex. 1002, ¶¶ 79-80.)

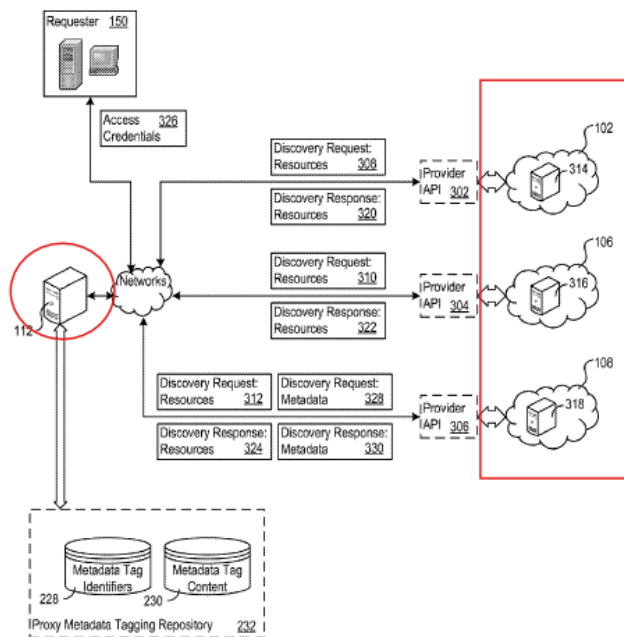
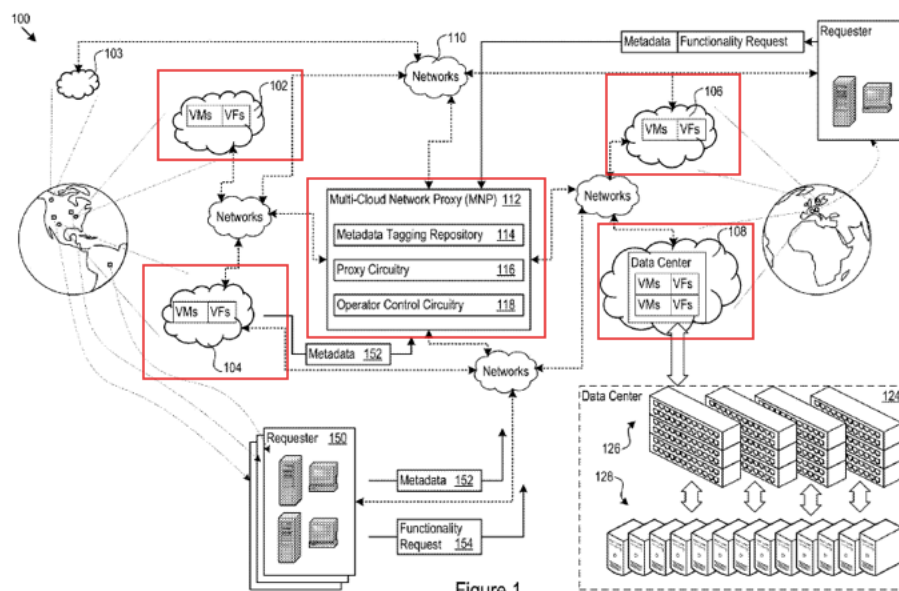


Figure 3

(Ex. 1004, Fig. 3.)

These providers were “associated with a cloud environment used by the private network.” Foscett disclosed “cloud environment” as the collection of target hosting platforms that the MNP centrally manages on behalf of a given requester. Figure 1 in Foscett (below) depicted the MNP (112) between a requester (150) and multiple service providers (102, 104, 106, and 108), serving as the requester’s unified point of contact across that collection of providers. (*Id.*, ¶¶ 0019–0022, Fig. 1.) The requester provides credentials for each provider to the MNP so it can interface on its behalf. (*Id.*, ¶ 0033.) A POSITA would have understood or found obvious that the MNP was “associated with a cloud environment used by the private network” because of the composite set of target hosting platforms. (Ex. 1002, ¶ 81.)



(Ex. 1004, Fig. 1.)

Accordingly, Foskett disclosed or rendered obvious all elements of limitation 1[a]/8[a]/15[a]. (Ex. 1002, ¶¶ 72, 82, 134, 145.)

1[b][i]/8[b[i]]/15[b][i]: “based on the received information, for each cloud resource of a plurality of cloud resources hosted by the plurality of cloud providers on behalf of the private network”

Foskett disclosed or rendered obvious this element because, using the information it already gathered about each provider (based on the received information), the MNP tracked the virtual machines and virtual functions (cloud resources) provisioned at each provider’s platform (hosted by the plurality of cloud providers), all for the benefit of the same requester whose credentials and provisioning requests the MNP handled (on behalf of the private network).

Cloud resources include provider-hosted items such as VMs provisioned into a provider’s target hosting platform. Foskett disclosed that “[t]he service providers may provide computing resources via hosting platforms that are generally publicly available.” (Ex. 1004, ¶ 0018.) Examples of such service providers are identified in the prior element. (*Id.*) Figure 1 in Foskett (below) depicted VMs and VFs hosted within each of the illustrated service providers (102, 104, 106, 108), confirming that resources of this kind sit inside the target hosting platforms.

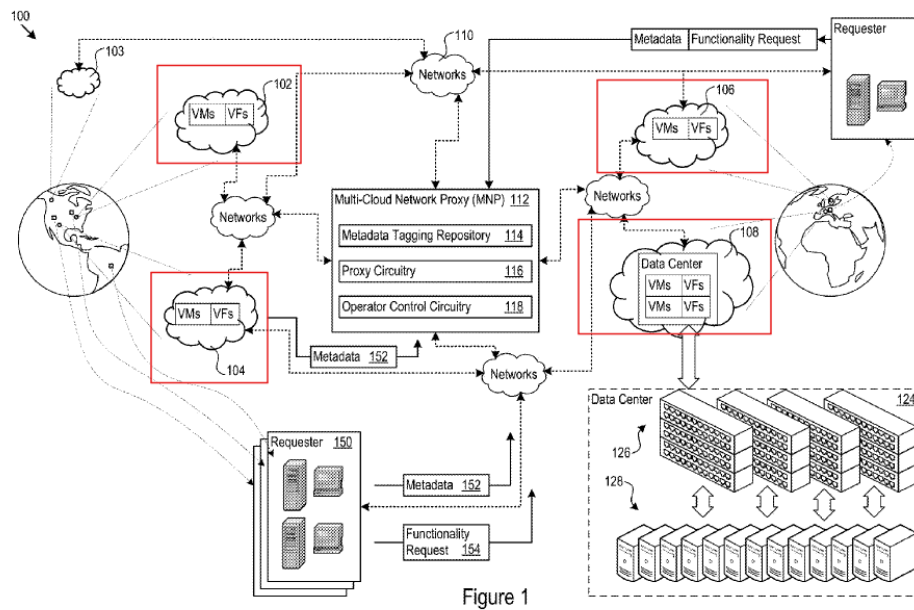


Figure 1

(Ex. 1004, Fig. 1.)

The “requester” is an end-user on whose behalf the MNP operates. Foskett confirmed that resources are provisioned at the requester’s own initiative and for the requester’s own benefit: “[t]he resource requester 150 interacts with the request interface 1000 to submit a request for a new resource to provision to a target hosting environment,” and the MNP acts “[i]n response to a resource requester 150 requesting provisioning of a new resource.” (*Id.*, ¶¶ 0058–0059.) Because cloud-based resources (like these VM and VF resources) are provisioned into a target hosting platform in order to serve that same requester, Foskett disclosed or rendered obvious resources “hosted by the plurality of cloud providers on behalf of the private network.”

Accordingly, Foskett disclosed or rendered obvious all elements of limitation 1[b][i]/8[b][i]/15[b][i]. (Ex. 1002, ¶¶ 83-88, 135, 146.)

1[b][ii]/8[b][ii]/15[b][ii]: “retrieving, by the cloud-tagging orchestrator service, information associated with the cloud resource”

Foskett disclosed or rendered obvious this element because the MNP’s tagging software (CTOS) separately queried the provider’s own metadata tagging repository and obtained (retrieving) that resource’s existing tag identifiers and content (information associated with the cloud resource).

“The discovery messages [sent by the MNP] specify a request for the target hosting platform to return identifiers of each resource deployed in the target virtual hosting platform,” after which the MNP “receives discovery responses ... enumerating the resources” and “updates its resource database.” (Ex. 1004, ¶ 0032.)

The MNP then retrieves additional information about each resource: “the MNP 112 also engages in metadata discovery for the resources provisioned in the target hosting platforms,” transmitting “metadata discovery requests 328 to the target hosting platforms” and receiving “metadata discovery responses 330” containing “metadata tag identifiers and metadata tag content stored remotely in the provider-controlled metadata tagging repository.” (*Id.*, ¶ 0035.) This discovery request is distinct request from the initial resource discovery request. Figures 3 and

4 in Foskett (below) also disclosed the MNP retrieving information associated with each cloud resource (e.g., identifier or metadata).

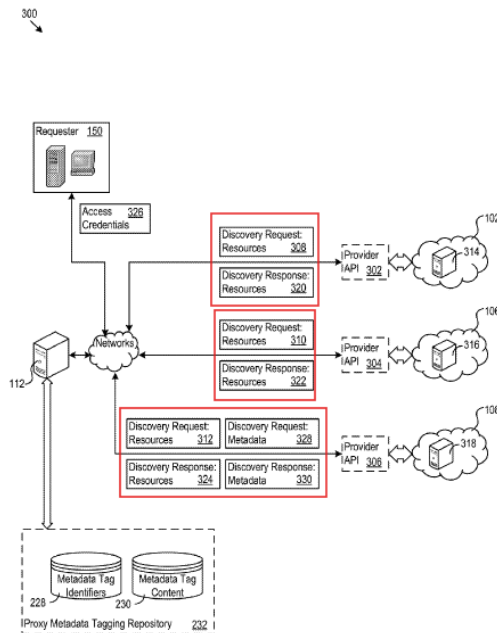


Figure 3

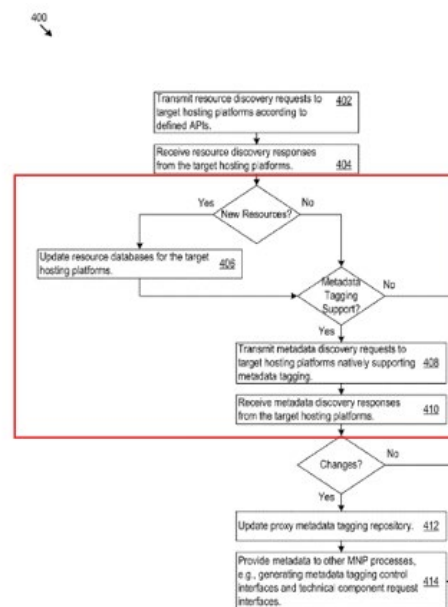


Figure 4

(Ex. 1004, Fig. 3, Fig. 4.)

Accordingly, Foskett disclosed or rendered obvious all elements of limitation 1[b][ii]/8[b][ii]/15[b][ii]. (Ex. 1002, ¶¶ 89-93, 136, 147.)

1[c]/8[c]/15[c]: “identifying, by the cloud-tagging orchestrator service, a unified tag of a plurality of unified tags for the cloud resource based on a pre-defined global tagging policy”

Foskett disclosed or rendered obvious this element because the MNP's tagging software (CTOS) funneled every provider's resources into one shared, provider-neutral repository and identified (identifying) the tag for a given resource (a unified tag of a plurality of unified tags) using definitions and value constraints an administrator set up in advance (a pre-defined global tagging policy), applied the same way regardless which provider hosted the resource.

Foskett aimed to solve the same problem as the '527 Patent: inconsistent tagging across providers preventing cross-provider policies. Foskett explained that its architecture "provides a technical solution to establishing a normalized data model for cloud resources, given the existing wide disparity and lack of consistency in how metadata tagging has been approached in the past." (Ex. 1004, Abstract; ¶ 0015.)

Instead of letting each provider's own tagging rules control, the MNP funnels every provider's resources through a single, shared repository it defines and maintains: "[t]he MNP 112 controls and facilitates normalization of the metadata tag identifie[r]s and metadata tag content attached to the technical components hosted in the various target hosting platforms by the service providers," regardless of whether any particular platform natively supports metadata tagging.

(*Id.*, ¶ 0031.) That “proxy metadata tagging repository” is described as “a centralized source of metadata tagging information.” (*Id.*)

Figure 2 in Foskett (below) showed this: one provider’s data center (Red Data Center) keeps its own native tag repository (238), another provider’s data center (Blue Data Center) has no native tagging capability at all (244), and both nonetheless connect into the same single proxy repository (232) housed in the MNP. (*Id.*, Fig. 2; ¶¶ 0027, 0029.) Every provider’s resources are thus tagged using tags within that shared repository, so the same tag carries a consistent meaning across cloud service providers—corresponding to the claimed “unified tag.”

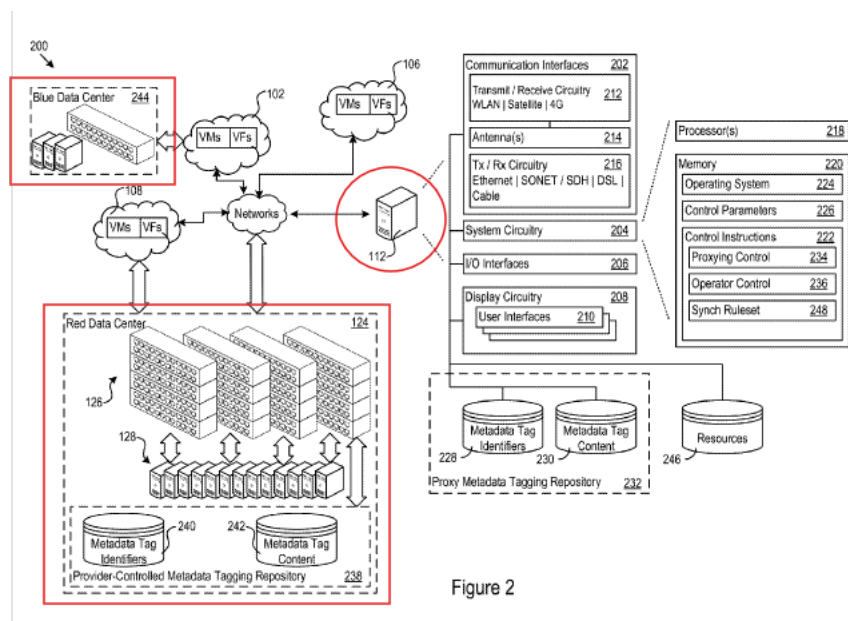


Figure 2

(Ex. 1004, Fig. 2.)

These tags are set according to a pre-defined global tagging policy. The tag assigned to a resource comes from a policy an administrator configures ahead of

time, through control interfaces. (*Id.*, Fig. 5, 6, and 12.) An administrator defines available metadata tag identifiers, marks certain tags as “prioritized” for particular resource types and providers and attaches “metadata tag type definitions” constraining what content each tag can hold. (*Id.*, ¶¶ 0042–0043, 0046–0048.)

Foskett confirmed this configuration precedes any specific tagging decision: “the MNP 112 obtains designations of prioritized metadata tags” as a standing configuration, and only afterward, “[i]n response to a resource requester 150 requesting provisioning of a new resource,” does the MNP consult those pre-existing designations to generate the request interface. (*Id.*, ¶ 0059.) This advance configuration makes the resulting policy “pre-defined.”

As for “global,” the tags are applied uniformly: policies are configured then applied to every applicable resource the MNP manages, irrespective of provider. Because the MNP maintains one proxy repository serving every connected provider, a tag identifier and its content rules carry the same meaning no matter which provider hosts the resource. (*Id.*, ¶¶ 0027, 0031, 0046–0048.) At minimum, configuring that shared repository so a given tag carries one consistent meaning across every synchronized provider would have been an obvious application of Foskett’s own design. (*Id.*, ¶¶ 0015, 0037, 0040.)

Foskett also disclosed a plurality of unified tags because the MNP applies its pre-defined tagging policy across the full population of resources it manages. The MNP's operator control circuitry defines multiple distinct metadata tag identifiers. (*Id.*, ¶¶ 0042-0043, 0046-0048.) Because the MNP's proxy repository stores and normalizes tag identifiers and content across this entire set of resources and providers, and different resource types are assigned different unified tags under the same pre-defined policy, Foskett disclosed a "plurality of unified tags." (*Id.*, ¶¶ 0027, 0031, 0040.)

Accordingly, Foskett disclosed or rendered obvious all elements of limitation 1[c]/8[c]/15[c]. (Ex. 1002, ¶¶ 94-102, 137, 148.)

1[d]/8[d]/15[d]: "assigning, by the cloud-tagging orchestrator service, the unified tag to the cloud resource via an application programming interface (API) of a cloud provider of the plurality of cloud providers hosting the resource"

Foskett disclosed or rendered obvious this element because the MNP's tagging software (CTOS) pushed (assigning) that same tag (the unified tag) out to the resource's hosting provider by calling that provider's API (via an API of a cloud provider hosting the resource).

Having identified the unified tag for a resource under a pre-defined global tagging policy, the MNP assigned the tag to the appropriate resource via the API of

the specific cloud provider hosting it. This occurs when an administrator updates an existing resource's tags and synchronizes the change back to the provider: “[i]n response to the Update option, the MNP 112 establishes a data connection to the target hosting platform (916) and transmits the metadata tags and metadata content currently in the proxy metadata tagging repository 232 to the specified target hosting platform to perform synchronization (918),” and “[t]o push the data, the MNP 112 may call, e.g., APIs provided by the target hosting platform.” (Ex. 1004, ¶ 0055.) Figure 9 in Foskett (below) depicted this synchronization logic.

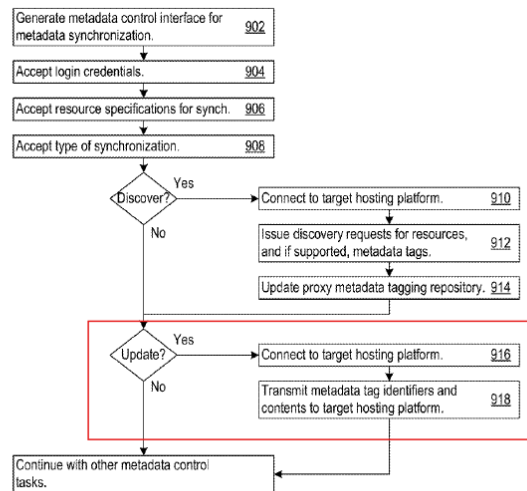


Figure 9

(*Id.*, Fig. 9.)

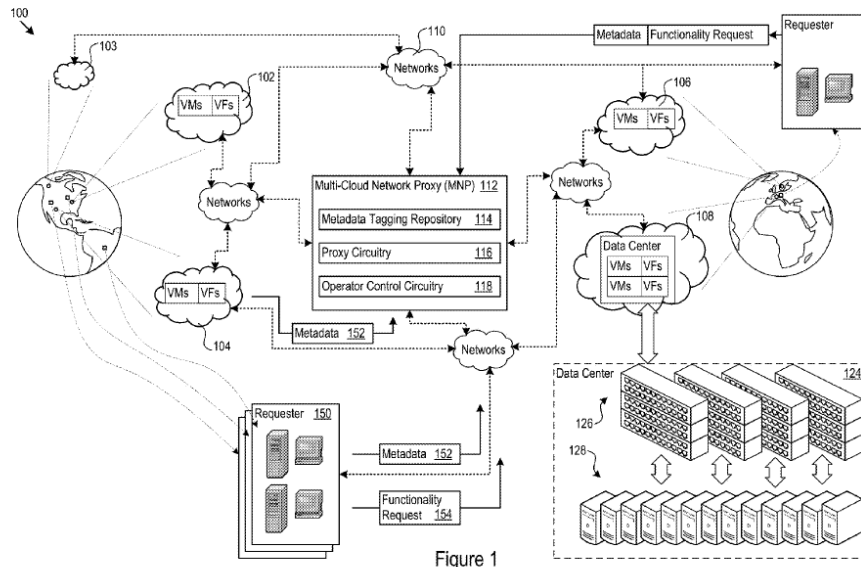
Foskett further disclosed that when the requester selects the synchronization option for a new resource, “the MNP 112 initiates the provisioning, also including the dynamic request content,” and “provides the dynamic request content with provisioning by transmitting (e.g., via API calls), to target hosting platform, the metadata tag identifiers and metadata tag contents entered into the request interface.” (*Id.*, Fig. 11; ¶ 0065.)

In both scenarios, the MNP calls the provider’s API to push the tag to the resource, and the tag content pushed is the same unified tag content the proxy repository holds under the pre-defined global tagging policy discussed in Element 1[c]. Accordingly, Foskett disclosed or rendered obvious all elements of limitation 1[d]/8[d]/15[d]. Thus, Foskett disclosed or rendered obvious claims 1/8/15. (Ex. 1002, ¶¶ 103-109, 138, 149.)

2. Claims 3, 10, and 17: “... wherein the cloud environment comprises a hybrid multi-cloud environment including a private cloud and a plurality of public clouds.”

Foskett disclosed or rendered obvious this claim because the MNP concurrently managed an on-premise platform (a private cloud) alongside multiple named public cloud platforms (a plurality of public clouds) together, forming the kind of mixed deployment the claim covers (a hybrid multi-cloud environment).

Figure 1 (below) depicted various cloud service providers, both public and private.



(Ex. 1004, Fig. 1.)

On the public side, Foscett named multiple public cloud platforms that the MNP manages: “Amazon Web Services (AWS), Google Compute Engine (GCE), Microsoft Azure (Azure).” (*Id.*, ¶ 0018.) On the private side, Foscett explained that “[s]ervice providers may additionally or alternatively provide computing resources in ‘on-premises’ hosting platforms, which typically refers to a location with increased privacy and security compared to public cloud resources,” and that “[a]n on-premise location may be within a secure facility owned by a resource requester

which has moved computing functionality to a cloud based implementation.” (*Id.*, ¶ 0018.)

It would have been obvious to a POSITA that a cloud environment would be a hybrid multi-cloud environment including a private cloud and a plurality of public clouds. Before the '527 Patent, enterprises typically paired private clouds—which housed sensitive assets but were costlier and less scalable—with cheaper, more flexible public cloud services for routine operations, often using multiple public providers to avoid single-provider dependence and capture the best cost and services of each.⁷ Thus, Foskett disclosed or rendered obvious claims 3/10/17. (Ex. 1002, ¶¶ 110-116, 139, 150.)

3. Claims 4 and 11: “...wherein said assigning further enables visibility and control for the plurality of cloud resources by the security manager.”

Foskett disclosed or rendered obvious this claim because the MNP’s assigning (discussed in element 1[d]) gives it an observable window into the current tag state of resources at every connected provider (visibility) and the means to determine and carry that tag state into effect at each provider (control) over the plurality of cloud resources hosted across the different connected providers.

⁷ E.g. <https://www.cbttuggets.com/blog/certifications/microsoft/aws-vs-azure-vs-google-cloud-wars-2019> (Ex. 1017)

Foskett disclosed display circuitry that generated user interfaces locally or for remote display. (Ex. 1004, ¶ 0024.) Foskett further disclosed that the metadata tag identifiers and content flowing into the proxy metadata tagging repository as a result of assigning “drive other processes within the MNP,” including “the generation of interactive metadata tagging control interfaces.” (*Id.*, ¶ 0036.) Because those interfaces reflect the same tag data the MNP assigned, an operator using them observes the actual, current tag state of resources at every connected provider—enabling visibility.

Foskett further disclosed that the MNP’s assigning also gives it control: the MNP “controls and facilitates normalization of the metadata tag identifie[r]s and metadata tag content attached to the technical components hosted in the various target hosting platforms by the service providers ... regardless of whether any particular target hosting platform natively supports metadata tagging.” (*Id.*, ¶ 0031.)

It is the MNP’s assigning step that gives it both an observable window into, and an operative hand on, the tag state of the plurality of cloud resources across every connected provider. Thus, Foskett disclosed or rendered obvious claims 4/11. (Ex. 1002, ¶¶ 117-121, 140.)

4. Claims 6, 13, and 20: “... wherein the plurality of cloud resources comprise virtual machine instances.”

Foskett disclosed or rendered obvious this claim because the resources the MNP tagged and managed (the plurality of cloud resources) included virtual machines (virtual machine instances) running at a provider’s data center.

Foskett explained that “FIG. 1 shows a target hosting platform implemented by a data center 124 for the service provider 108,” and that “[t]he data center 124 supports running many different virtual machines (VMs), each potentially running many different virtual functions (VFs).” (Ex. 1004, ¶ 0022.) Foskett’s Figure 10 process further confirmed that the MNP’s resource-request interface is used, for example, “when the resource requester 150 is requesting a new VM hosted by the ... service provider,” at which point “the MNP 112 may add to the request interface the metadata tags previously designated as prioritized for VMs for the ... service provider.” (*Id.*, ¶ 0061.) Thus, Foskett disclosed or rendered obvious claims 6/13/20. (Ex. 1002, ¶¶ 122-124, 141, 151.)

5. Claims 7 and 14: “... wherein the security manager comprises a cloud based security management appliance.”

Foskett disclosed or rendered obvious this claim because the MNP’s infrastructure (security manager) was described as capable of distributed implementation across multiple processors, memories, and systems (comprises),

communicating with cloud providers over both private and public networks, consistent with a cloud-based security management appliance.

Foskett disclosed that the MNP's infrastructure "may include any combination of hardware, software, firmware, or other logic," and its implementation "may be distributed," spanning "multiple distinct system components, such as multiple processors and memories," and "multiple distributed processing systems." (Ex. 1004, ¶¶ 0026, 0068.) The MNP's infrastructure communicates with the service providers' hosting platforms over networks that "may include private and public networks," consistent with a cloud-based, network-accessible appliance. (*Id.*, ¶¶ 0018–0019.) Thus, Foskett disclosed or rendered obvious claims 7/14. (Ex. 1002, ¶¶ 125-128, 142.)

B. Ground 2: Combination of Foskett and Whitlock renders claims 2, 5, 9, 12, 16, and 19 obvious.

Whitlock: Whitlock disclosed systems and methods for classifying and organizing IT assets using tags, including automatically generated tags based on asset attributes. (Ex. 1005, Abstract, ¶ 0002.) Whitlock further disclosed that "certain tag sets are automatically generated" through a "harvesting" process in which "one or more items of data concerning the configuration of the node are automatically stored." (Ex. 1005, ¶ 0059.) This harvested "data can then be used by the tag-based monitoring and management tool ... to automatically assemble

system tag sets without user intervention,” using “information about the assets that is available from (or harvested from) the assets,” including “information about the asset’s operating system, database version, IP address, and/or other such[] configuration information.” (*Id.*, ¶ 0041.) Whitlock’s tagging profile interface further allowed an administrator to define automatically-applied “conditions” that determine which tags are assigned to a node. (*Id.*, ¶¶ 0037, 0070, 0080.)

Whitlock depicted a tag set management interface where “tag set display 810 is named ‘location’ and refers to tags that are interrelated to each other because the tags each indicate an asset’s location” and also listed “Primary Function” as a tag set category. (*Id.*, Fig. 8 (below); ¶¶ 0072–0073.)

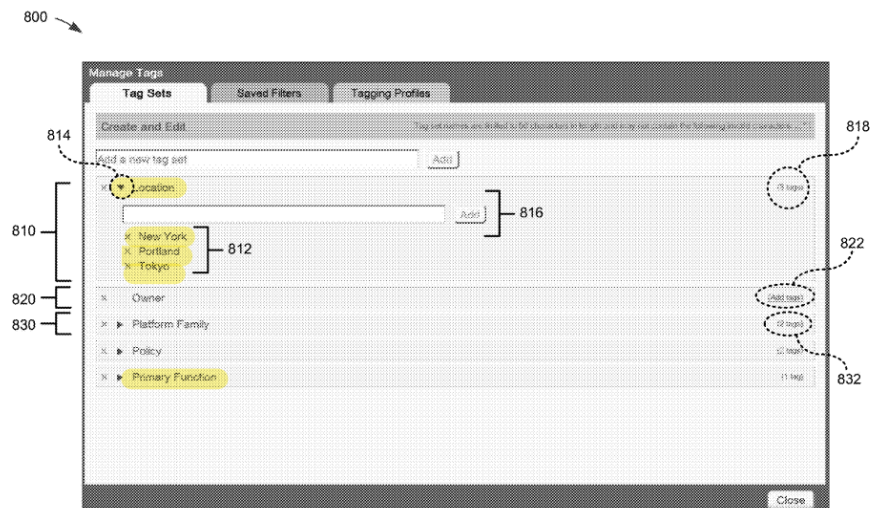


FIG. 8

(*Id.*, Fig. 8.)

Whitlock further disclosed that applying a tagging profile generates “a tagged set of nodes” and only afterward, as a separate step, is that set used by “a compliance and configuration control tool” to apply a rule or other policy. (*Id.*, ¶¶ 0069–0070.) (Ex. 1002, ¶¶ 152-155)

1. Motivations to combine

A POSITA would have been motivated to apply Whitlock’s tag-set teachings—location, function, and the separation between tag assignment and policy application—within Foscett’s MNP architecture with a reasonable expectation of success.

First, Foscett and Whitlock addressed the same problem from complementary angles. Foscett’s goal was normalizing tags across disparate providers (Ex. 1004, ¶ 0015), but it did not catalog location or function as tag bases, or separate a tagged state from later policy application. Whitlock fills both gaps: IT asset tags are commonly organized around location and function (Ex. 1005, Fig. 8; ¶¶ 0072–0073), and tag assignment and policy application are distinct steps. (*Id.*, ¶¶ 0069–0070). A POSITA would naturally have looked to these conventions, improving Foscett’s system without detracting from it.

Second, incorporating location and function would have furthered Foscett’s objective of avoiding “inconsistent and wildly variable metadata tagging” (Ex.

1004, ¶ 0063), since these are among the most basic attributes of any cloud resource.

Third, combining Whitlock’s categorization and attributes with Foskett’s MNP amounted to applying known techniques to a known device to yield the predictable result of a more complete policy—the kind of combination *KSR International Co. v. Teleflex Inc.*, 550 U.S. 398 (2007) confirmed is obvious. (Ex. 1002, ¶¶ 156-160.)

2. **Claim 2, 9, and 16: “... facilitating, by the security manager, application of a security policy expressed with reference to one or more unified tags of the plurality of unified tags by retrieving information associated with a particular cloud resource of the plurality of cloud resources from a storage database.”**

Foskett in view of Whitlock disclosed or rendered obvious this claim because Foskett’s proxy metadata tagging repository (storage database) from which the MNP retrieves an already-assigned unified tag, and Whitlock confirmed that a security policy is applied by reference to that tag only after the tag itself has already been assigned.

Foskett disclosed that the MNP maintains a storage database in the form of its “proxy metadata tagging repository” that stores the unified tag identifiers and tag content applicable to each cloud resource. (Ex. 1004, ¶¶ 0027, 0031.) This repository drives further processes, including “the generation of interactive

metadata tagging control interfaces and driving dynamic adaptation to technical component request interfaces,” under which the MNP “add[s] specifically identified metadata tag identifiers and tag content fields to technical component request interfaces” based on retrieved tags. (*Id.*, ¶ 0036.) Certain tags carry mandatory-value and content-format requirements: a type definition may specify that “a value must be provided for the metadata tag,” and the MNP “constrains the input for the metadata tag content according to the type definition,” rejecting nonconforming content. (*Id.*, ¶¶ 0049–0051.) These requirements are part of assigning a unified tag.

Whitlock confirmed tag assignment and policy application are distinct, sequential steps where the tags are applied to a node, the nodes are assigned to groups, and then policies are applied automatically to these groups. (Ex. 1005, ¶ 0060). Specifically, Whitlock disclosed generating “a tagged set of nodes” by assigning tags to nodes satisfying a profile’s conditions, structuring that set into a data set usable by a downstream tool, and only afterward using a “compliance and configuration control tool ... to apply a rule (or other policy) to the tagged set of nodes in the data set.” (*Id.*, ¶¶ 0069–0071.)

Whitlock’s compliance tool operates by “capturing a baseline of server file systems, desktop file system, directory servers, databases, virtual systems

middleware applications and/or network device configurations in a known good state,” with “[o]ngoing integrity checks then compar[ing] the current states against these baselines to detect changes.” (*Id.*, ¶ 0037-0038.) Detecting drift from a known-good state is a well-recognized security control. Whitlock confirmed this rule applies at the tagged-group level, not the tagging step itself: a “rule group” (e.g., “Critical Change Audit Rules”) can be selected “for application to a node group” created from tags, scoping “tasks and policies” to those groups. (*Id.*, ¶ 0094.)

Consistent with the mapping in Element 1[a], the MNP retrieves the unified tag already assigned from the proxy metadata tagging repository and, consistent with the tag-then-policy sequence Whitlock disclosed, applies a security policy to that already-tagged resource. Accordingly, Foskett in view of Whitlock disclosed or rendered obvious all elements of claims 2/9/16. (Ex. 1002, ¶¶ 161-168, 175, 177.)

3. Claims 5, 12, and 19: “... wherein the pre-defined global tagging policy identifies the unified tag based on one or more of the information regarding the cloud resource, a location of the resource, and a function of the cloud resource.”

Foskett disclosed or rendered obvious this claim because Foskett’s own tag type definitions already key on information about the resource, its location, and its

function, and Whitlock independently confirmed location and function as established, named bases for identifying a tag to the extent Foskett's disclosure of those two bases is found insufficient on its own.

Foskett disclosed that the pre-defined global tagging policy identifies the unified tag based "on one or more of the information regarding the cloud resource." Foskett's operator control circuitry defines metadata tag type definitions (e.g., datatype, mandatory-value, and valid-value-list constraints) based on information regarding the resource being tagged, such as its resource type and provider. (Ex. 1004, ¶¶ 0042–0048.)

Foskett alone also disclosed identifying the tag based "on the location of the resource": resources are "provisioned into a region hosted by a resource provider," a "unit of hosting capacity in a particular geographic region." (*Id.*, ¶ 0020.) The MNP's synchronization interface includes "input fields 706 for provider, location, and resource type," allowing the operator "to specify which types of resources to synchronize at which provider and which provider location." (*Id.*, ¶ 0053.) Foskett's tag type definitions—configurable to key on any field, including resource type and provider—would just as readily key on this location field. (*Id.*, ¶¶ 0042–0048) Foskett alone also disclosed identifying the tag based on "a function of the cloud resource": its "Environment" tag is restricted to a list such as "Development,"

“Test,” and “Production.” (*Id.*, ¶¶ 0047–0048.) These identify a resource’s functional role.

To the extent Foskett alone does not expressly identify location or function as separate bases, Whitlock disclosed both: a “Location” tag set with example tags for New York, Portland, and Tokyo, and a “Primary Function” tag set. (Ex. 1005, Fig. 8; ¶¶ 0072–0073.) A POSITA would have configured Foskett’s policy to identify the tag by location or function using that same mechanism. (Ex. 1004, ¶¶ 0042–0048.) Accordingly, Foskett in view of Whitlock disclosed or rendered obvious all elements of claims 5/12/19. (Ex. 1002, ¶¶ 169-174, 176, 178.)

C. Ground 3: Combination of Foskett and Juniper renders claim 18 obvious.

Juniper: Juniper disclosed a single point of management for a multi-cloud environment, including route propagation, security, and application deployment. (Ex. 1006, Title.) Juniper explained that “[i]n some existing techniques, each VCE administrator would need to setup policies to establish communication tunnels between the different VCEs,” resulting in “delays or errors due to the complications associated with ensuring proper route propagation, security, and application deployment.” (*Id.*, ¶ 0010.) Juniper’s solution is “[a] centralized controller ... that allows one administrator, with a single pane of glass, to control the interconnections and security within” a multi-cloud network, so “the network

administrator may interface with only [the] controller ... via a single pane of glass to establish the communication links” across every managed cloud. (*Id.*, ¶¶ 0032, 0035.) (Ex. 1002, ¶ 179.)

1. Motivations to combine

A POSITA would have been motivated to look to Juniper and present Foskett’s centralized visibility and control through a single pane of glass user interface with a reasonable expectation of success.

First, Foskett already disclosed centralized visibility and control over tagged resources across multiple providers, but its control interfaces (Ex. 1004, Figs. 5, 6, and 12) address tag management for individual resource types and providers, without a single consolidated view. Juniper fills this gap, disclosing that “an administrator may be able to configure communication between different VCEs ... rather than establish communication links on a VCE-by-VCE or data center-by-data center basis.” (Ex. 1006, ¶ 0074.)

Second, both references address the same administrative burden of managing resources across heterogeneous environments. Juniper’s purpose was to spare an administrator from interfacing separately with each cloud’s controller (*id.*, ¶¶ 0010, 0035), the same problem Foskett addresses through tag normalization;

pairing the two would further both references' shared goal of unified administration.

Third, combining Juniper's presentation with Foskett's repository amounted to applying a known technique to a known device to achieve a predictable, unified administrative view—the kind of combination *KSR* confirmed is obvious. (Ex. 1002, ¶¶ 180-184.)

2. Claim 18: “wherein said assigning enables visibility and control of the plurality of cloud resources via a single pane of glass user interface.”

Foskett and Juniper disclosed or rendered obvious this claim because Foskett's assigning step already gave its MNP centralized visibility and control over resources at every connected provider and Juniper disclosed presenting that same kind of centralized, cross-provider control to an administrator through a single, unified interface (a single pane of glass user interface).

Juniper disclosed “a centralized controller (e.g., controller 14) that allows one administrator, with a single pane of glass, to control the interconnections and security within” a multi-cloud network, such that “the network administrator may interface with only [the] controller... via a single pane of glass to establish the communication links” across every managed cloud. (Ex. 1006, ¶¶ 0032, 0035.) A POSITA combining Foskett with Juniper would have presented Foskett's

centralized, cross-provider visibility and control through Juniper's single pane of glass interface. Accordingly, Foskett in view of Juniper disclosed or rendered obvious claim 18. (Ex. 1002, ¶¶ 185-188.)

D. Ground 4: Combination of Kung and Serhiienko renders claims 1, 2, 3, 4, 6, 7, 8, 9, 10, 11, 13, 14, 15, 16, 17, and 20 obvious.

Kung: Kung disclosed a contextual security platform that automatically configures native, network-level security mechanisms across an organization's computer network using security policies specified at an application level. (Ex. 1007, Abstract, ¶ 0009.) Kung's platform runs as software on one or more computers in communication with a hybrid computer network infrastructure "having both traditional hardware resources and virtual resources provided by private and public cloud infrastructure services." (*Id.*, ¶¶ 0009, 0046.)

Kung's platform functions as an engine reading and writing a database storing application-level security policies, logical groups, infrastructure resource information, and network flow information. (*Id.*, ¶ 0050.) Kung's platform communicates with each provider's API, requesting and receiving information on network resources and traffic flows, and using the API to configure security mechanisms native to those providers. (*Id.*, ¶ 0051.)

Kung's platform associates key/value attributes with infrastructure resources using several mechanisms, including reading "infrastructure tags" that providers

such as AWS, Azure, GCP, and OpenStack allow customers to associate with their resources, which “are read using the API of the service provider during the process of discovering infrastructure resources and used to automatically map discovered infrastructure resources to logical groups.” (*Id.*, ¶ 0141.) Logical groups can be created automatically based on provider-assigned properties (e.g., AWS’s VPC, region, and account properties, Azure’s Resource Group and virtual network properties, and OpenStack’s project and network properties). (*Id.*, ¶¶ 0145–0148.) Kung’s platform selects a resource into logical groups using these attributes and computes a security configuration using the security policies together with resources belonging to the applicable groups. (*Id.*, ¶ 0085.)

Kung disclosed that its platform can provide “an effective global security policy” and enforce policies for an enterprise’s full set of resources by combining native security mechanisms at multiple compute environments in a coordinated way. (*Id.*, ¶¶ 0073, 0077, 0132.) Kung’s attributes are populated by reading each provider’s own native tags and resource properties. (*Id.*, ¶ 0141.) (Ex. 1002, ¶¶ 189-193.)

Serhiienko: Serhiienko disclosed an extensible tag management middleware based on cloud functions, called the Universal Connector (“UC”), that turns tags into a powerful declarative means of cloud management, demonstrated by a

deployment administration scenario involving both AWS and Google Cloud Platform services. (Ex. 1008, Abstract.) Serhienko explained that “[o]ne of the biggest challenges in multi-cloud tag-based management is that each provider implements tagging in a different way,” and that a “universal tagging system can thus aid this process by implementing a ruleset for tags that meets the constraints of all providers simultaneously, and by providing a single interface through which to manage tagging across all providers.” (*Id.*, § III.A.) Serhienko’s implementation achieves this “by providing a common API tagging method that abstracts the provider-specific details and provides a uniform interface, and by providing the ability to apply tags automatically.” (*Id.*)

The UC’s core functions include managing tags “in a secure, audible and cost-effective manner” and performing “best practices cloud management tasks driven by regulated tags,” (*Id.*, § I; § III.) Serhienko’s ruleset defines, for each resource type and provider filter, a rule composed of conditions plus tag key-value pairs to apply when a resource’s metadata satisfies those conditions, depending on retrieving the resource’s full metadata from its provider so that “rules [can] be created for any field in the metadata.” (*Id.*, § III.B.) When satisfied, “the method then calls the appropriate tagResource function and applies the tag to the resource.” (*Id.*)

Serhiienko's tagging algorithm iterates over every provider, calling a function to get each resource's metadata and a function to get the list of resources, checking each resource's metadata against the rules, and calling a function to tag the matching resources. (*Id.*, § III.C, Listing 1.) Serhiienko explained that “[i]n order to satisfy the need for consistent tagging we employ the auto-tagging functionality that allows us to define a set of tagging rules in a YAML file,” and that “[t]he rules can apply to any metadata field of each individual resource, across all providers,” implemented through an “autoGenerate API method” that “composes several cloud functions from all providers in order to retrieve all resources that satisfy the ruleset and to apply the necessary tags to them,” such that “just one invocation of this function can tag every resource across all managed projects with the right tag.” (*Id.*, § IV.A.)

Serhiienko was presented at an IEEE conference and published by IEEE in 2019—IEEE being one of the two main computer-science professional organizations that publish peer-reviewed research and widely-applied industry standards, which a POSITA in cloud infrastructure would have known to review.⁸ The IEEE's website showing Serhiienko was published in “2019 19th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGRID),”

⁸ <https://ieeexplore.ieee.org/document/8752784> (Ex. 1018)

presented at an IEEE conference in May 2019, and added to IEEE *Xplore* in July 2019 is more than ample evidence of its date of publication and public accessibility. *See Acceleration Bay, LLC v. Activision Blizzard, Inc.*, 908 F.3d 765 (Fed. Cir. 2018) (affirming Board’s determinations of unpatentability based on IEEE article); *Ericsson Inc. v. Intellectual Ventures*, IPR2014-00527, Paper 41 (P.T.A.B. May 18, 2015) (accepting publication information on the IEEE copyright line as evidence of reference’s date of publication and public accessibility). (Ex. 1002, ¶¶ 194-197.)

1. Motivations to combine

A POSITA would have been motivated to combine Kung’s platform with Serhiienko’s UC with a reasonable expectation of success.

First, the references address complementary halves of the same problem. Kung disclosed a platform that computes security policy from attributes read from each provider’s own native tags and resource properties (Ex. 1007, ¶¶ 0145–0148), but did not disclose normalizing those differently formatted, provider-native values into a single, consistent tag. (Ex. 1007, ¶¶ 0085, 0141.) Serhiienko disclosed exactly this normalization: a universal ruleset and connector that “meets the constraints of all providers simultaneously,” applying “a single interface through which to manage tagging across all providers.” (Ex. 1008, § III.A.) A POSITA implementing Kung’s platform across the same multi-cloud deployments it already

contemplated (Ex. 1007, ¶ 0046) would naturally have looked to Serhiienko's ruleset to normalize those tags into a single, provider-independent tag before selecting resources into logical groups.

Second, both references relied on tags as the trigger for automated management action. Kung disclosed operating across hybrid infrastructures spanning AWS, Azure, GCP, OpenStack, and VMware vCenter (Ex. 1007, ¶¶ 0009–0010, 0046), and a POSITA implementing it across such a heterogeneous deployment would have recognized the need Serhiienko identified: “each provider implements tagging in a different way” (Ex. 1008, § III.A), such that a normalization mechanism would benefit consistent, cross-provider policy.

Third, combining Serhiienko's ruleset with Kung's platform amounted to combining known elements according to known methods to yield the predictable result of consistent, normalized tags before computing security policy—the type of combination *KSR* confirmed is obvious. (Ex. 1002, ¶¶ 198-202.)

2. Claims 1, 8, and 15

1[pre]:

Kung in view of Serhiienko disclosed a method for managing cloud resource tags across multiple cloud providers. The UC in Serhiienko “manages

tags in a secure, audible and cost-effective manner” and “performs best practices cloud management tasks driven by regulated tags.” (Ex. 1008, § I; Ex. 1002, ¶ 203.)

8[pre]:

Kung disclosed that its processes may be embodied in machine-executable instructions stored on a “non-transitory machine-readable medium,” including semiconductor memory devices such as EPROM and EEPROM, magnetic disks, magneto-optical disks, and CD-ROM/DVD-ROM disks, that are executed by a processor to cause the processor to perform the disclosed processes. (Ex. 1007, ¶ 0042.) Kung further disclosed “[c]omputer 100 includes a processor 102,” which “communicates with a main or working memory 104 and a storage memory 106 over one or more buses.” (*Id.*, ¶ 0037.) Consistent with the mapping established in Element 1[a], Kung’s processor 102, on which Serhiienko’s UC runs to execute the instructions stored on Kung’s non-transitory machine-readable medium, forms the processing resource. Kung further disclosed “a security manager of a private network” for the same reasons provided in Element 1[a] below. (*Id.*, ¶ 0011; Ex. 1002, ¶¶ 259-260.)

15[pre]:

Kung disclosed or rendered obvious a multi-tag orchestration system, comprising a computer, its component parts, and the software processes running on

it that together discover, tag, and apply security policy to cloud resources across multiple providers. Kung disclosed a computer comprising a processor and memory that communicates with a main or working memory over a bus, where the memory is capable of storing instructions being executed by the processor, which may be stored on a non-transitory machine-readable medium that, when executed, causes the processor to perform the disclosed processes. (Ex. 1007, ¶¶ 0037–0038, 0042.) For the remaining elements, Kung in view of Serhiienko disclosed those elements for the same reasons discussed in claims 1 and 8. (Ex. 1002, ¶¶ 271-272.)

1[a]/8[a]/15[a]:

Kung in view of Serhiienko disclosed or rendered obvious this element because Kung’s contextual security platform (security manager) runs on an organization’s network infrastructure (associated with a private network) and communicates with various cloud APIs (each cloud provider of a plurality of cloud providers), and Serhiienko’s UC (CTOS) receives each provider’s list of hosted resources and metadata information across that same multi-provider deployment (receiving ... information regarding each cloud provider of a plurality of cloud providers associated with a cloud environment used by the private network).

Kung disclosed its platform is “implemented as software running on a computer server in communication with various security mechanisms native to an

organization's or enterprise's data network infrastructure.” (Ex. 1007, ¶ 0011.) Kung explained that “[t]o prevent un-authorized accesses to data and resources available in workloads on an organization's or enterprise's computer network,” its platform enforces “network security policies using native network security mechanisms” within “hybrid computer network infrastructures having both traditional hardware resources and virtual resources provided by private and public cloud infrastructure services.” (*Id.*, Abstract, ¶ 0009.) A POSITA would have understood Kung's platform to disclosed the claimed “security manager associated with a private network.” (*Id.*, ¶ 0011; Ex. 1002, ¶ 205.)

Serhiienko's UC disclosed the CTOS, performing the receiving, identifying, and assigning work the claim requires: its middleware “manages tags in a secure, audible and cost-effective manner” and “performs best practices cloud management tasks driven by regulated tags” across multiple cloud providers. (Ex. 1008, § I.) A POSITA implementing Kung's platform across the multi-cloud deployment Kung already contemplates would have run Serhiienko's UC to orchestrate its tagging functions. (Ex. 1002, ¶ 206.)

Serhiienko's UC receives information regarding each cloud provider by calling a function “to get list of resources” for every provider in turn, returning the set of resources that provider hosts. (*Id.*, § III.C, Listing 1.) Kung's own disclosure

confirmed the same basic mechanism at the platform level: Kung’s platform “communicates with the application programming interface of one or more cloud service providers ... using the native application programming interfaces provided by the services,” and “request[s] and receive[s] [information] from the API of each of the cloud service provider[s],” including “information on network resources provided by the service.” (Ex. 1007, ¶ 0051; Ex. 1002, ¶ 207.)

Kung and Serhiienko both disclosed “a plurality of cloud providers.” Kung was explicit that its platform “can support any number of cloud service providers.” (Ex. 1007, ¶ 0051.) Figure 2 in Kung (below) depicted the platform communicating with multiple provider APIs concurrently. (*Id.*) Serhiienko confirmed its UC likewise operates with a multi-provider system simultaneously: its tagging algorithm iterates “all resources” within a project or account across multiple providers, and its evaluation scenario applied its ruleset across AWS and GCP concurrently. (Ex. 1008, § III.C, Listing 1; § IV.A.) A POSITA would have understood Kung’s platform, running Serhiienko’s UC, is genuinely capable of receiving information regarding a plurality of cloud providers at once. (Ex. 1002, ¶¶ 208-209.)

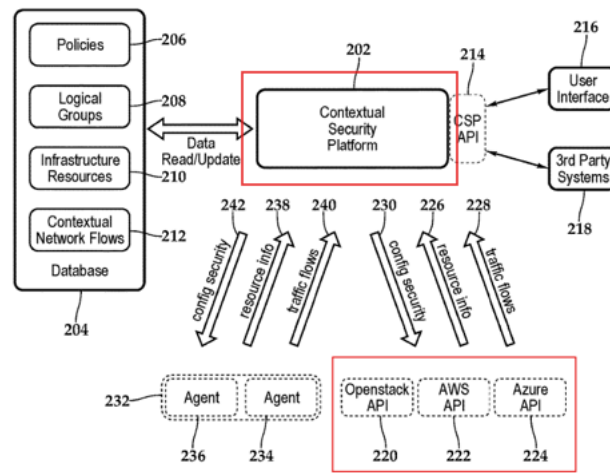


Fig.2

(Ex. 1007, Fig. 2.)

Kung and Serhiienko further disclosed that the providers are “associated with a cloud environment used by the private network”: Kung disclosed a “cloud environment” as the collection of providers that its platform centrally manages on behalf of a given organization’s network, and Serhiienko likewise disclosed that its UC manages resources “across all providers” through a single connector instance for that deployment. (Ex. 1007, ¶¶ 0049, 0119; Ex. 1008, § III.A; Ex. 1002, ¶¶ 210-211.)

Accordingly, Kung in view of Serhiienko disclosed or rendered obvious all elements of limitation 1[a]/8[a]/15[a]. (Ex. 1002, ¶¶ 204, 212, 261, 273.)

1[b][i]/8[b][i]/15[b][i]:

Kung in view of Serhiienko disclosed or rendered obvious this element because Serhiienko’s tagging algorithm evaluated, using the metadata each

provider returned under Element 1[a] (based on the received information), every resource satisfying its ruleset across every connected provider (for each cloud resource of a plurality of cloud resources), and Kung confirmed those resources are created and hosted on the providers' own infrastructure for the particular owner organization whose network Kung's platform serves (hosted by the plurality of cloud providers on behalf of the private network).

Serhiienko disclosed "based on the received information" by disclosing that, based on the resource and metadata information the UC received from each provider under Element 1[a], its tagging algorithm iterated " $\forall$ metadata $\in$ resources_metadata" for every provider evaluating each individual resource in turn against the ruleset's conditions. (Ex. 1008, § III.C, Listing 1).

Serhiienko disclosed "a plurality of cloud resources" by disclosing a ruleset defining conditions that determine which resources are tagged and managed, after which the tagging algorithm assembles a "resources_to_tag" set spanning every resource, across every connected provider, to satisfy those conditions. (*Id.*, §§ III.B-C, Listing 1.) Serhiienko's evaluation scenario applied this ruleset to resources hosted across AWS and GCP on behalf of a single organization implementing the connector, retrieving each provider's own resource list. (*Id.*, § IV.A.)

Kung confirmed that this kind of resource is created using, and hosted by, the cloud service provider itself. Kung disclosed that infrastructure service providers allow tags to be associated with “infrastructure resources created using, and hosted by, the service,” read “by communicating with the cloud service provider’s API.” (Ex. 1007, ¶ 0229) Each such resource is associated with an “owner organization”—the same organization whose network Kung’s platform serves. (*Id.*, Fig. 10, ¶¶ 0011, 0217.) A POSITA would have understood these resources to be “hosted by the plurality of cloud providers on behalf of the private network.” Accordingly, Kung in view of Serhienko disclosed or rendered obvious all elements of limitation 1[b][i]/8[b][i]/15[b][i]. (Ex. 1002, ¶¶ 213-218, 262, 274.)

1[b][ii]/8[b][ii]/15[b][ii]:

Kung in view of Serhienko disclosed or rendered obvious this element because Serhienko’s UC (CTOS) called a function to obtain each resource’s full metadata from its provider (retrieving information associated with the cloud resource).

Serhienko disclosed that the UC’s tagging algorithm iterates for every provider to retrieve that provider’s resource metadata and resource list and calls a function to “get resource metadata” for each provider. (Ex. 1008, § III.C, Listing 1). Serhienko explained that “by retrieving the entirety of the resource’s metadata

from its provider, it allows for rules to be created for any field in the metadata.” (*Id.*, § III.B.) Accordingly, Kung in view of Serhiienko disclosed or rendered obvious all elements of limitation 1[b][ii]/8[b][ii]/15[b][ii]. (Ex. 1002, ¶¶ 219-221, 263, 275.)

1[c]/8[c]/15[c]:

Kung in view of Serhiienko disclosed or rendered obvious this element because Serhiienko’s UC (CTOS) matched a resource’s metadata against an administrator-authored, cross-provider ruleset (identifying ... a unified tag ... based on a pre-defined global tagging policy) to determine, from among several rule-specified tags, the one that resource’s metadata satisfied (a unified tag of a plurality of unified tags).

Serhiienko’s tagging algorithm evaluates each resource’s metadata against the ruleset and, when a rule’s conditions are met, determines the specific tag that rule assigns to that resource before any tag is pushed to it: “when rule applied to metadata: resources_to_tag.add the resource from resources_list with the tags from rule.” (Ex. 1008, § III.C, Listing 1.) This determination is the identifying step, matching a resource’s metadata against the ruleset’s conditions to determine which tag key-value pair applies, using a defined logical structure: “the different conditions have a logical OR relationship, so in order for the rule to be satisfied, at

least one condition must be met,” while “[t]he statements within a condition have a logical AND relationship, meaning all statements within a condition must be met by a resource simultaneously for it to be compliant with the rule.” (Ex. 1008, § III.B.)

The resulting tag is a “unified tag”: the ruleset “defines per resource type and provider filter a rule composed of an arbitrary number of conditions plus a set of tag key-value pairs to be applied in case of a match,” so the resulting tag carries a consistent, provider-independent meaning regardless of which provider hosts it. (*Id.*, §§ III.A-C, IV.B.)

The policy is “pre-defined.” Before any resource is evaluated, “a set of consistency enforcement rules need to be employed by the system’s administrator” (*id.*, § III.B). Serhiienko disclosed that it employs “the auto-tagging functionality that allows us to define a set of tagging rules in a YAML file,” subsequently applied by “the autoGenerate API method” against resources retrieved at runtime. (*Id.*, § IV.A-B.) This advance configuration makes the policy pre-defined.

The policy is also “global,” applied consistently “across all providers.” Serhiienko disclosed the ruleset “meets the constraints of all providers simultaneously,” providing “a single interface through which to manage tagging across all providers” via “a common API tagging method that abstracts the

provider-specific details.” (*Id.*, § III.A, § IV.A, Title.) “The rules can apply to any metadata field of each individual resource, across all providers,” and the autoGenerate method “composes several cloud functions from all providers ... to apply the necessary tags to them,” such that “just one invocation of this function can tag every resource across all managed projects with the right tag.” (*Id.*, § IV.A.)

Serhiienko disclosed identifying a unified tag of a “plurality” of unified tags because a single ruleset invocation applies across the UC’s entire population of managed resources and rules, with different rules specifying different tag key-value pairs. (*Id.*, § IV.A-B; § III.B.) Accordingly, Kung in view of Serhiienko disclosed or rendered obvious all elements of limitation 1[c]/8[c]/15[c]. (Ex. 1002, ¶¶ 222-229, 264, 276.)

1[d]/8[d]/15[d]:

Kung in view of Serhiienko disclosed or rendered obvious this element because Serhiienko’s UC (CTOS) called the matched tag’s tagResource function (assigning the unified tag to the cloud resource), executed through the specific hosting provider’s own native cloud function service (via an API of a cloud provider ... hosting the resource).

Once a resource is matched to a tag, Serhiienko’s “method then calls the appropriate tagResource function and applies the tag to the resource.” (Ex. 1008, §

III.B.) A POSITA would have understood this application of the tag to be equivalent to “assigning.” This tagResource call is carried out through each provider’s own API: the UC’s tagging functions are “implemented for both AWS Lambda and Google Cloud Functions (GCF),” each deployed into, and executed using, the specific hosting provider’s own native cloud function service. (*Id.*, § IV.A-B.)

Figure 1 in Serhiienko (below) depicted this concretely: functions on “AWS Lambda” act on resources hosted at AWS, while functions on “GCP Cloud Functions” act on resources hosted at GCP, with “[t]he functions deployed to the various providers ... registered to an event gateway” so that “requests made to the connector reach the functions via the gateway” for the applicable provider. (*Id.*, § IV.A, Fig. 1.) The tagResource call for a given resource thus necessarily executes through the API of the specific provider hosting it.

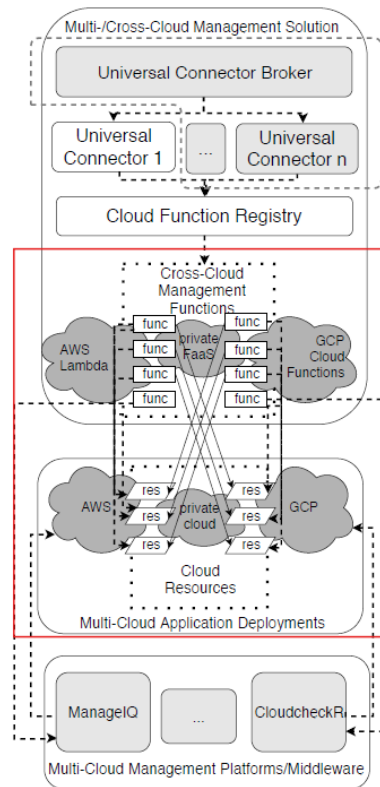


Fig. 1. Universal connector architecture outlining extensibility through FaaS and interoperability with existing management platforms

(*Id.*, Fig. 1.)

Accordingly, Kung in view of Serhienko disclosed or rendered obvious all elements of limitation 1[d]/8[d]/15[d]. (Ex. 1002, ¶¶ 230-236, 265, 277.) Thus, Kung in view of Serhienko disclosed or rendered obvious claims 1/8/15.

3. Claims 2, 9, and 16:

Kung in view of Serhienko disclosed or rendered obvious this claim because Serhienko's UC maintained a repository of generated tags retrievable on demand (a storage database), and Kung's platform retrieved tags from that repository to trigger creation of concrete firewall and security-group rules keyed to

those tags (facilitating ... application of a security policy expressed with reference to one or more unified tags).

Serhiienko disclosed the claimed storage database: the UC's ruleset and generated tags are maintained and applied through its own configuration and cloud function deployment, from which resources and their tags can be retrieved on demand. (Ex. 1008, § IV.A-B.)

Kung disclosed the platform's role in retrieving tags and applying an actual security policy expressed with reference to them: "infrastructure tags" are read "using the API of the service provider during the process of discovering infrastructure resources and used to automatically map discovered infrastructure resources to logical groups." (Ex. 1007, ¶ 0141.) "[S]ecurity policies are represented by logical groups," such that a resource's tag-driven group membership determines which policy applies: "a logical group PCI defines a selection attribute ... This ensures that all resources for application app1 are assigned policies defined for [the] PCI logical group." (*Id.*, ¶¶ 0137, 0144.) Kung's platform computes a security configuration using those policies and resources, then deploys enforcement: "[u]sing infrastructure provider APIs or agents, the communication between logical groups are use[d] to create security rules in AWS Security groups, OpenStack Security groups, Azure Security Groups, Host

Firewalls, etc.” (*Id.*, ¶¶ 0085, 0154-0155.) A POSITA would have configured Kung’s platform to retrieve the unified tags Serhienko’s UC generated for logical-group membership, facilitating application of these policies to the corresponding resource. Thus, Kung in view of Serhienko disclosed or rendered obvious claims 2/9/16. (Ex. 1002, ¶¶ 237-241, 266, 278.)

4. Claims 3, 10, and 17:

Kung in view of Serhienko disclosed or rendered obvious this claim because Kung’s platform concurrently managed an on-premise data center (a private cloud) alongside named public cloud service providers (a plurality of public clouds), together forming a hybrid multi-cloud environment.

Because a hybrid multi-cloud environment, by definition, includes both private and public clouds, Kung’s disclosure of those components necessarily disclosed a “hybrid multi-cloud environment.” Kung’s platform operates across environments including “Amazon AWS, Microsoft Azure and Google GCP” as example public cloud providers and “OpenStack and VMware vCenter” as example private cloud environments. (Ex. 1007, ¶ 0046.) Kung’s Figure 5 embodiment confirmed that a single platform concurrently manages both an “on premise data center 502 of an organization or enterprise” and “a cloud service provider environment 504 that provides virtualized infrastructure resources for use

by the organization,” noting that “any number of environments can be controlled by the contextual security platform.” (*Id.*, ¶ 0074.)

Serhiienko’s UC disclosed managing resources across multiple such providers concurrently through a single ruleset and connector instance, applying its ruleset to resources hosted across AWS and GCP on behalf of a single organization implementing the connector. (Ex. 1008, § IV.A.) A POSITA implementing Serhiienko’s UC on Kung’s platform would have done so across the same hybrid public/private cloud environment Kung already disclosed. Thus, Kung in view of Serhiienko disclosed or rendered obvious claims 3/10/17. (Ex. 1002, ¶¶ 242-245, 267, 279.)

5. Claims 4 and 11:

Kung in view of Serhiienko disclosed or rendered obvious this claim because Serhiienko’s UC’s tag assignment made every connected provider’s resources queryable by tag, giving Kung’s platform (security manager) an observable window into resources’ current tag state (visibility for the plurality of cloud resources) and, through logical-group selection and security policy computation from that same tag data, a means to act on those resources at their respective providers (control for the plurality of cloud resources). It is Serhiienko’s

UC that performs this assigning, and Kung’s platform that obtains visibility and control as a result.

Once tags are applied, “[r]etrieval of resources based on their tags ... can retrieve information on resources of any type on any provider, based on the tags applied to the resource.” (Ex. 1008, § IV.B.) Because that retrieval reflects the same tag data the UC assigned, an operator querying resources by tag observes the actual, current tag state of resources at every connected provider—enabling visibility. Serhiienko’s assigning likewise gives control: “the capability to manage resources by tag” allows the connector to “invoke a number of resource management cloud functions from both providers” that “act on all resources with the specified tag.” (*Id.*) Kung’s platform relies on this same tag data to select resources into logical groups and compute security policy configurations, then deploys that configuration back to each resource’s provider. (Ex. 1007, ¶ 0085.) Thus, Kung in view of Serhiienko disclosed or rendered obvious claims 4/11. (Ex. 1002, ¶¶ 246-251, 268.)

6. Claims 6, 13, and 20:

Kung in view of Serhiienko disclosed or rendered obvious this claim because Serhiienko’s ruleset tagged virtual machine resources hosted at AWS and

GCP and Kung’s contextual security platform likewise managed virtual machine instances (the plurality of cloud resources comprise virtual machine instances).

Serhiienko disclosed that its evaluation scenario applied the UC’s ruleset to VM resources hosted across AWS and Google Cloud Platform. (Ex. 1008, § IV.A.) Kung confirmed its infrastructure resources include virtual machine instances specifically: “[a]n environment managed by the contextual security platform can host nodes with different operating systems that provide different network security mechanisms,” and “[w]hen nodes are virtual machines or containers, their traffic can be enforced by mechanisms running in the hypervisor or operating system hosting the virtual nodes.” (Ex. 1007, ¶ 0088.) In the cloud context specifically, “Amazon AWS offer[s] security groups that can enforce security rules for individual virtual machine instances.” (*Id.*, ¶ 0089.) Thus, Kung in view of Serhiienko disclosed or rendered obvious claims 6/13/20. (Ex. 1002, ¶¶ 252-255, 269, 280.)

7. Claims 7 and 14:

Kung in view of Serhiienko disclosed or rendered obvious this claim because Kung’s contextual security platform (security manager) runs as server software that configures native security mechanisms across named public and

private cloud infrastructure services (a cloud based security management appliance).

Kung disclosed that its platform is “implemented as software running on a computer server in communication with various security mechanisms native to an organization’s or enterprise’s data network infrastructure” (Ex. 1007, ¶ 0011), and that this infrastructure includes public and private cloud services. (*Id.*, ¶ 0046.) Kung’s Figure 5 embodiment further confirmed the platform concurrently manages resources hosted directly within a cloud service provider environment. (*Id.*, ¶¶ 0074, 0076.) A POSITA would have understood a security manager implemented as software running on a server in communication with, and configuring security mechanisms native to, such infrastructure services is a “cloud-based security management appliance.” Thus, Kung in view of Serhienko disclosed or rendered obvious claims 7/14. (Ex. 1002, ¶¶ 256-258, 270.)

E. Ground 5: Combination of Kung, Serhienko, and Whitlock renders claims 5, 12, and 19 obvious.

1. Motivations to combine

A POSITA would have been motivated to combine Whitlock’s location and function tag-set teachings with the Kung/Serhienko combination with a reasonable expectation of success.

First, Kung and Serhienko already identified a unified tag using information, location-related, and function-related resource attributes. (Ex. 1007, ¶¶ 0145–0148; Ex. 1008, § IV.A.) Whitlock disclosed these same categories under express, named tag-set headings: a “Location” tag set with example tags for New York, Portland, and Tokyo, and a “Primary Function” tag set. (Ex. 1005, ¶¶ 0072–0073, Fig. 8.) A POSITA would have looked to Whitlock’s named categories as a further model for organizing attributes Kung and Serhienko already read and evaluate.

Second, both Whitlock and the combined platform address the same problem of classifying resources using attributes beyond a bare identifier. Whitlock’s techniques allow a user to “easily and quickly select and perform actions on groups of one or more assets” based on “node properties or user-selected criteria or conditions” (Ex. 1005, Abstract, ¶ 0040), the same basic function the Kung/Serhienko combination already performs. (Ex. 1007, ¶¶ 0145–0148; Ex. 1008, § IV.A.)

Third, applying Whitlock’s categories to the combination amounted to applying a known technique to a known combination to yield the predictable result of a platform that identifies a unified tag using those categories—the kind of combination *KSR* confirmed is obvious. (Ex. 1002, ¶¶ 281-285.)

2. Claims 5, 12, and 19:

Kung in view of Serhiienko disclosed or rendered obvious this claim because Serhiienko's ruleset identified the unified tag by keying on metadata fields of the cloud resource, which a POSITA would have understood to include the resource's information, location, and function. To the extent needed, Whitlock independently disclosed or rendered obvious this claim because Whitlock tagged assets using harvested configuration information (the information regarding the cloud resource), a named "Location" tag set (a location of the resource), and a named "Primary Function" tag set (a function of the cloud resource).

Serhiienko disclosed that its ruleset conditions can key on "any metadata field of each individual resource, across all providers" (Ex. 1008, § IV.A.) Serhiienko's example ruleset conditions key specifically on resource type and provider (*id.*, § III.B), with its retrieval function obtaining "the entirety of the resource's metadata from its provider" (*Id.*, § III.B). A POSITA would have understood a resource's information, location or function as fields commonly returned by a provider's resource metadata API, keyed on using this same mechanism.

To the extent that Serhiienko does not itself disclose these three bases, Whitlock disclosed each directly and by name. As to information regarding the

resource, Whitlock's tag-based classification tool tags assets using information "about the asset's operating system, database version, IP address, and/or other such[] configuration information," obtained by harvesting data directly from the assets. (Ex. 1005, ¶ 0041.) As to location, Whitlock disclosed a "Location" tag set, illustrating that "tag set display 810 is named 'location' and refers to tags that are interrelated to each other because the tags each indicate an asset's location," with "specific tags for 'New York', 'Portland', and 'Tokyo,'" indicating that a tagging profile can be configured to automatically apply a location tag to a node meeting specified conditions. (*Id.*, ¶¶ 0067–0068, 0073, Fig. 13.) As to function, Whitlock disclosed that a tag set may reflect an asset's functional role, listing "Primary Function" as a tag set category alongside "Location," "Owner," and "Platform Family." (*Id.*, Fig. 13.) A POSITA would have understood that Whitlock's "Primary Function" tag set identifies a resource's functional role in the same manner that Whitlock applied to location tagging.

Thus, Serhienko disclosed or rendered obvious claims 5/12/19, and Whitlock provides further, independent support for these claims in the proposed combination of Kung, Serhienko, and Whitlock. (Ex. 1002, ¶¶ 286-295.)

F. Ground 6: Combination of Kung, Serhienko, and Juniper renders claim 18 obvious.

1. Motivations to combine

A POSITA would have been motivated to look to Juniper and present the centralized visibility and control in Kung/Serhienko through a single pane of glass user interface with a reasonable expectation of success.

First, Kung in view of Serhienko already disclosed centralized visibility and control over tagged resources across multiple providers but addresses tag management for individual resources without a consolidated view spanning all providers. Juniper fills this gap, disclosing that a multi-cloud platform can present centralized, cross-provider information through such a consolidated interface. (Ex. 1006, ¶¶ 0032, 0035.)

Second, both references address the same administrative burden of managing resources across heterogeneous providers. Juniper's purpose was to spare an administrator from interfacing separately with each cloud's controller (*id.*, ¶¶ 0010, 0035), the same goal furthered by pairing the combined platform's centralized tag data with Juniper's single pane of glass.

Third, combining Juniper's presentation with the combined platform's tag data amounted to applying a known technique to a known platform to achieve a

predictable, unified administrative view—the type of combination *KSR* confirmed is obvious. (Ex. 1002, ¶¶ 296-300.)

2. Claim 18:

Kung, Serhienko, and Juniper disclosed or rendered obvious this claim because Serhienko’s assigning step already gave its UC centralized visibility and control over resources at every connected provider and Juniper disclosed presenting that same kind of centralized, cross-provider control to an administrator through a single, unified interface (a single pane of glass user interface).

Kung in view of Serhienko already disclosed that the UC’s assigning gives the combined system centralized visibility and control over tagged and untagged resources across every connected provider. (Ex. 1007, ¶ 0085; Ex. 1008, § IV.A-B.) Juniper disclosed presenting exactly this kind of centralized, cross-provider information through a single pane of glass: “a centralized controller ... that allows one administrator, with a single pane of glass, to control the interconnections and security within” a multi-cloud network, such that “the network administrator may interface with only [the] controller ... via a single pane of glass to establish the communication links” across every managed cloud. (Ex. 1006, ¶¶ 0032, 0035.) A POSITA combining Kung and Serhienko with Juniper would have presented the platform’s centralized, cross-provider visibility and control through this interface.

Thus, Kung in view of Serhienko, in further view of Juniper, disclosed or rendered obvious claim 18. (Ex. 1002, ¶¶ 301-304.)

X. CONCLUSION

Accordingly, the Board should institute IPR and cancel the challenged claims.

Dated: September 15, 2026

12275 El Camino Real, Ste 100
San Diego, California, 92130
Phone: (858) 720-8900
Fax: (858) 509-3691

Respectfully submitted,
SHEPPARD, MULLIN,
RICHTER & HAMPTON LLP

/s/ Thomas Millikan

Thomas Millikan (Reg. No. 72,316)

Back-up Counsel
Joseph P. Reid (Reg. No. 72,317)
Andrew Klein (*PHV* forthcoming)
Usayd Siddiqi (Reg. No. 80,999)

Attorneys for Netskope, Inc.

LISTING OF CHALLENGED CLAIMS

1[pre]: A method comprising:

1[a]: receiving, by a cloud-tagging orchestrator service running on a security manager associated with a private network, information regarding each cloud provider of a plurality of cloud providers associated with a cloud environment used by the private network; and

1[b][i]: based on the received information, for each cloud resource of a plurality of cloud resources hosted by the plurality of cloud providers on behalf of the private network:

1[b][ii]: retrieving, by the cloud-tagging orchestrator service, information associated with the cloud resource;

1[c]: identifying, by the cloud-tagging orchestrator service, a unified tag of a plurality of unified tags for the cloud resource based on a pre-defined global tagging policy; and

1[d]: assigning, by the cloud-tagging orchestrator service, the unified tag to the cloud resource via an application programming interface (API) of a cloud provider of the plurality of cloud providers hosting the resource.

2: The method of claim 1, further comprising facilitating, by the security manager, application of a security policy expressed with reference

to one or more unified tags of the plurality of unified tags by retrieving information associated with a particular cloud resource of the plurality of cloud resources from a storage database.

3: The method of claim 1, wherein the cloud environment comprises a hybrid multi-cloud environment including a private cloud and a plurality of public clouds.

4: The method of claim 1, wherein said assigning further enables visibility and control for the plurality of cloud resources by the security manager.

5: The method of claim 1, wherein the pre-defined global tagging policy identifies the unified tag based on one or more of the information regarding the cloud resource, a location of the resource, and a function of the cloud resource.

6: The method of claim 1, wherein the plurality of cloud resources comprise virtual machine instances.

7: The method of claim 1, wherein the security manager comprises a cloud based security management appliance.

8[pre]: A non-transitory computer-readable storage medium embodying a set of instructions, which when executed by a processing

resource of a security manager of a private network, causes the processing resource to perform a method comprising:

8[a]: receiving information regarding each cloud provider of a plurality of cloud providers associated with a cloud environment used by the private network; and

8[b][i]: based on the received information, for each cloud resource of a plurality of cloud resources hosted by the plurality of cloud providers on behalf of the private network:

8[b][ii]: retrieving information associated with the cloud resource;

8[c]: identifying a unified tag of a plurality of unified tags for the cloud resource based on a pre-defined global tagging policy; and

8[d]: assigning the unified tag to the cloud resource via an application programming interface (API) of a cloud provider of the plurality of cloud providers hosting the resource.

9: The non-transitory computer-readable storage medium of claim 8, wherein the method further comprises facilitating application of a security policy expressed with reference to one or more unified tags of the plurality of unified tags by retrieving information associated with a particular cloud resource of the plurality of cloud resources from a storage database.

10: The non-transitory computer-readable storage medium of claim 8, wherein the cloud environment comprises a hybrid multi-cloud environment including a private cloud and a plurality of public clouds.

11: The non-transitory computer-readable storage medium of claim 8, wherein said assigning further enables visibility and control for the plurality of cloud resources by the security manager.

12: The non-transitory computer-readable storage medium of claim 8, wherein the pre-defined global tagging policy identifies the unified tag based on one or more of the information regarding the cloud resource, a location of the resource, and a function of the cloud resource.

13: The non-transitory computer-readable storage medium of claim 8, wherein the plurality of cloud resources comprise virtual machine instances.

14: The non-transitory computer-readable storage medium of claim 8, wherein the security manager comprises a cloud based security management appliance.

15[pre]: A multi-tag orchestration system comprising: a processing resource; and a non-transitory computer-readable medium, coupled to the processing resource, having stored therein instructions that when executed

by the processing resource cause the processing resource to perform a method comprising:

15[a]: receiving information regarding each cloud provider of a plurality of cloud providers associated with a cloud environment used by a private network; and

15[b][i]: based on the received information, for each cloud resource of a plurality of cloud resources hosted by the plurality of cloud providers on behalf of the private network:

15[b][ii]: retrieving information associated with the cloud resource;

15[c]: identifying a unified tag of a plurality of unified tags for the cloud resource based on a pre-defined global tagging policy; and

15[d]: assigning the unified tag to the cloud resource via an application programming interface (API) of a cloud provider of the plurality of cloud providers hosting the resource.

16: The system of claim 15, wherein the method further comprises facilitating application of a security policy expressed with reference to one or more unified tags of the plurality of unified tags by retrieving information associated with a particular cloud resource of the plurality of cloud resources from a storage database.

17: The system of claim 15, wherein the cloud environment comprises a hybrid multi-cloud environment including a private cloud and a plurality of public clouds.

18: The system of claim 15, wherein said assigning enables visibility and control of the plurality of cloud resources via a single pane of glass user interface.

19: The system of claim 15, wherein the pre-defined global tagging policy identifies the unified tag based on one or more of the information regarding the cloud resource, a location of the resource, and a function of the cloud resource.

20: The system of claim 15, wherein the plurality of cloud resources comprise virtual machine instances.

CERTIFICATE OF WORD COUNT UNDER 37 CFR §42.24(D)

Under 37 C.F.R. §42.24(d), the undersigned certifies that the word count for this Petition for *Inter Partes* Review totals 13,567 words, excluding the parts exempted by 37 C.F.R. §42.24(a). The word count was made using the built-in word count function in the Microsoft® Word 365 software used to prepare this document.

Dated: September 15, 2026

12275 El Camino Real, Ste 100
San Diego, California, 92130
Phone: (858) 720-8900
Fax: (858) 509-3691

Respectfully submitted,
SHEPPARD, MULLIN,
RICHTER & HAMPTON LLP

/s/ Thomas Millikan

Thomas Millikan (Reg. No. 72,316)

Back-up Counsel
Joseph P. Reid (Reg. No. 72,317)
Andrew Klein (*PHV* forthcoming)
Usayd Siddiqi (Reg. No. 80,999)

Attorneys for Netskope, Inc.

CERTIFICATE OF SERVICE

The undersigned hereby certifies that true copies of this PETITION FOR *INTER PARTES* REVIEW and supporting materials (Exhibits 1001-1018 and Power of Attorney) have been served via USPS Express Mail and electronically on counsel for patent owner as follows:

GISH PLLC
50 California Street, Suite 1500
San Francisco, CA 94111
UNITED STATES
ryan@gishpllc.com

In addition, the undersigned certifies that a true copy of the petition and supporting materials have been served on Patent Owner via email to its legal counsel:

David A. Nelson – davidnelson@quinnemanuel.com
Athena D. Dalton – athenadalton@quinnemanuel.com
Harrison Rose – harrisonrose@quinnemanuel.com
Luiz Miranda – luizmiranda@quinnemanuel.com
Alexander Rudis – alexanderrudis@quinnemanuel.com
Jeff Gerchick – jeffgerchick@quinnemanuel.com
Andrew M. Holmes – drewholmes@quinnemanuel.com
Ognjen Zivojnovic – ogizivojnovic@quinnemanuel.com
Justin Sorenson – justinsorensen@quinnemanuel.com

Andrew D. Gish – andrew@gishpllc.com
Christopher Gerson – chris.gerson@gishpllc.com
Josef Schenker – josef.schenker@gishpllc.com

Dated: September 15, 2026

12275 El Camino Real, Ste 100
San Diego, California, 92130
Phone: (858) 720-8900
Fax: (858) 509-3691

Respectfully submitted,
SHEPPARD, MULLIN,
RICHTER & HAMPTON LLP

/s/ Thomas Millikan

Thomas Millikan (Reg. No. 72,316)

Back-up Counsel
Joseph P. Reid (Reg. No. 72,317)
Andrew Klein (*PHV* forthcoming)
Usayd Siddiqi (Reg. No. 80,999)

Attorneys for Netskope, Inc.