US 20220294699A1

(19) **United States**

(12) **Patent Application Publication**

Yang et al.

(10) **Pub. No.: US 2022/0294699 A1**

(43) **Pub. Date: Sep. 15, 2022**

(54) **NETWORK REACHABILITY IMPACT ANALYSIS**

(71) Applicant: **Google LLC**, Mountain View, CA (US)

(72) Inventors: **Hongkun Yang**, San Jose, CA (US); **Hui Liu**, San Ramon, CA (US); **Gargi Adhav**, San Jose, CA (US); **Alan Tang**, Los Angeles, CA (US)

(73) Assignee: **Google LLC**, Mountain View, CA (US)

(21) Appl. No.: **17/804,389**

(22) Filed: **May 27, 2022**

Related U.S. Application Data

(63) Continuation of application No. 17/117,376, filed on Dec. 10, 2020, now Pat. No. 11,381,460.

Publication Classification

(51) **Int. Cl.**

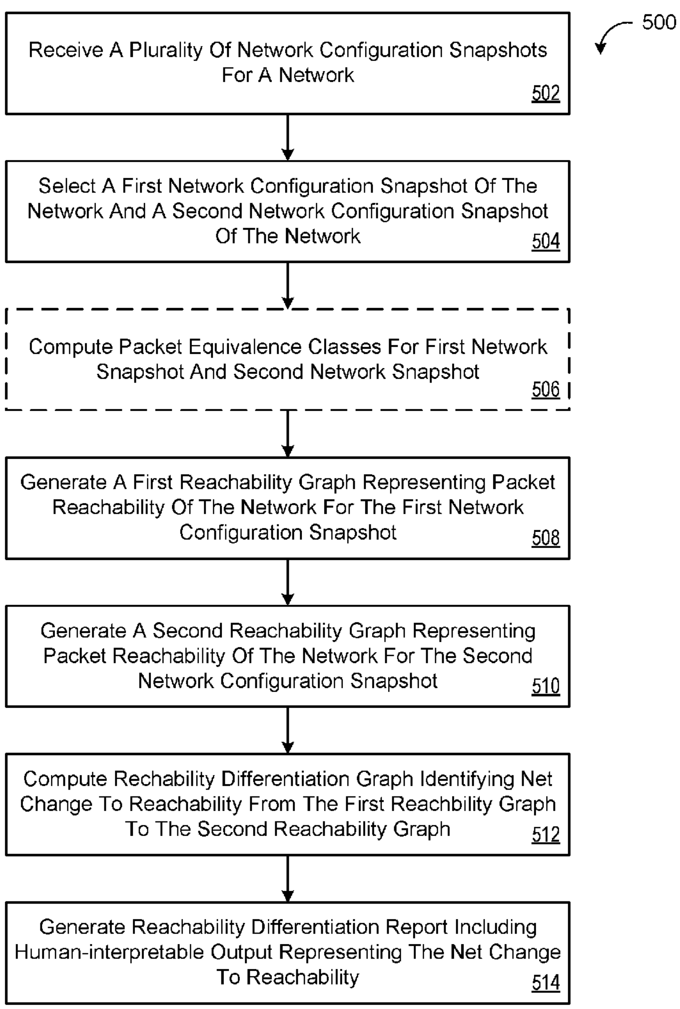
H04L 41/12 (2006.01)
H04L 43/045 (2006.01)
H04L 41/084 (2006.01)
H04L 41/082 (2006.01)

(52) **U.S. Cl.**

CPC ***H04L 41/12*** (2013.01); ***H04L 43/045*** (2013.01); ***H04L 41/0846*** (2013.01); ***H04L 41/082*** (2013.01)

(57) **ABSTRACT**

A method of network reachability impact analysis includes receiving a plurality of network configuration snapshots for a network. The method also includes selecting a first network configuration snapshot of the network and a second network configuration snapshot of the network. The method further includes generating a first reachability graph representing packet reachability of the network for the first network configuration snapshot. The method also includes generating a second reachability graph representing packet reachability of the network for the second network configuration snapshot. The method also includes computing a reachability differentiation graph identifying a net change to reachability from the first reachability graph to the second reachability graph. The method further includes generating a reachability differentiation report including a human-interpretable output of the net change to reachability.



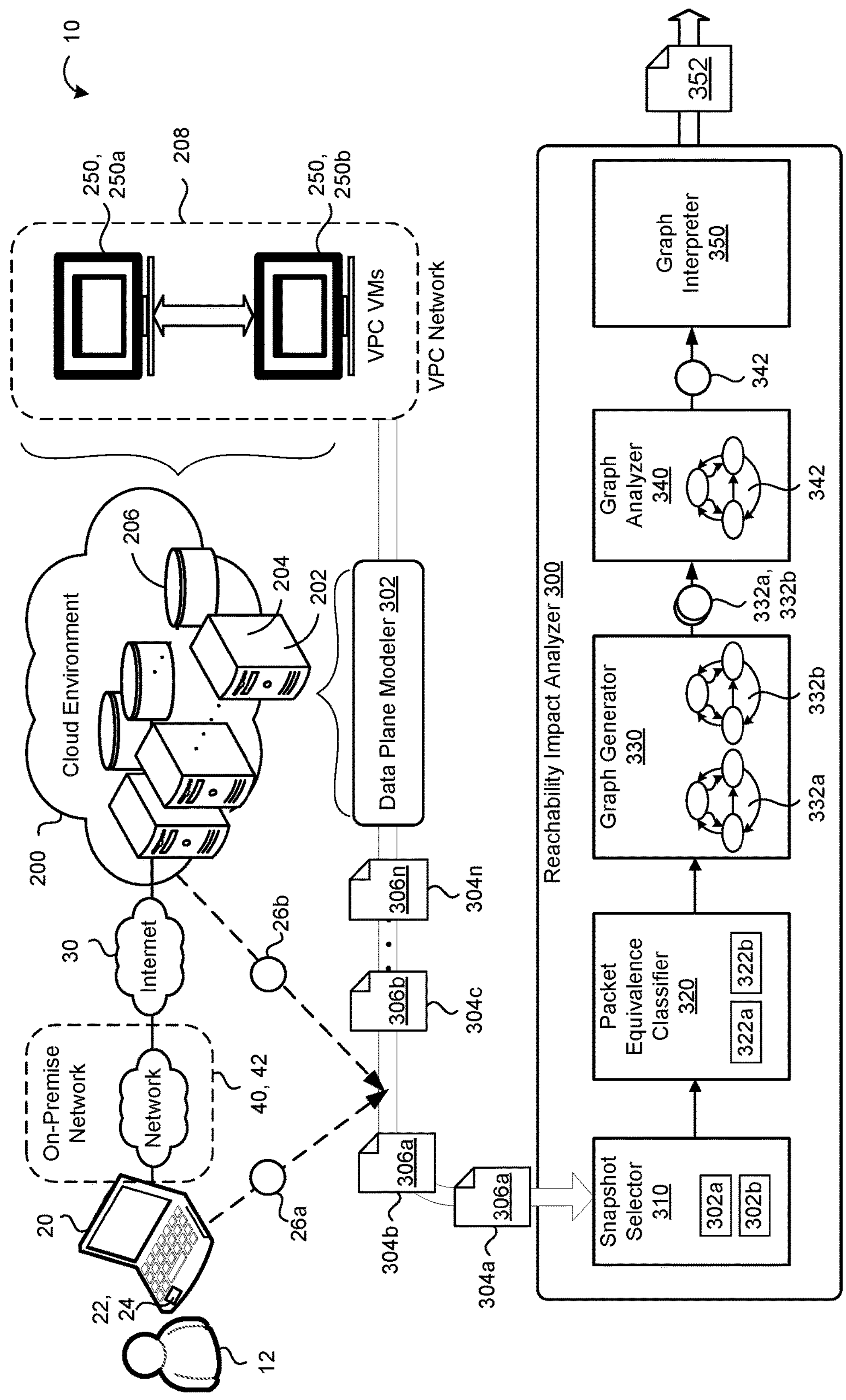

10
12
20
22, 24
26a
26b
On-Premise Network
Network
40, 42
Internet
30
200
Cloud Environment
202
204
206
208
VPC VMs
VPC Network
250, 250a
250, 250b
Data Plane Modeler 302
306a
304a
304b
306b
304c
306n
304n
Reachability Impact Analyzer 300
Snapshot Selector 310
302a
302b
Packet Equivalence Classifier 320
322a
322b
Graph Generator 330
332a
332b
332a, 332b
Graph Analyzer 340
342
Graph Interpreter 350
352

FIG. 1

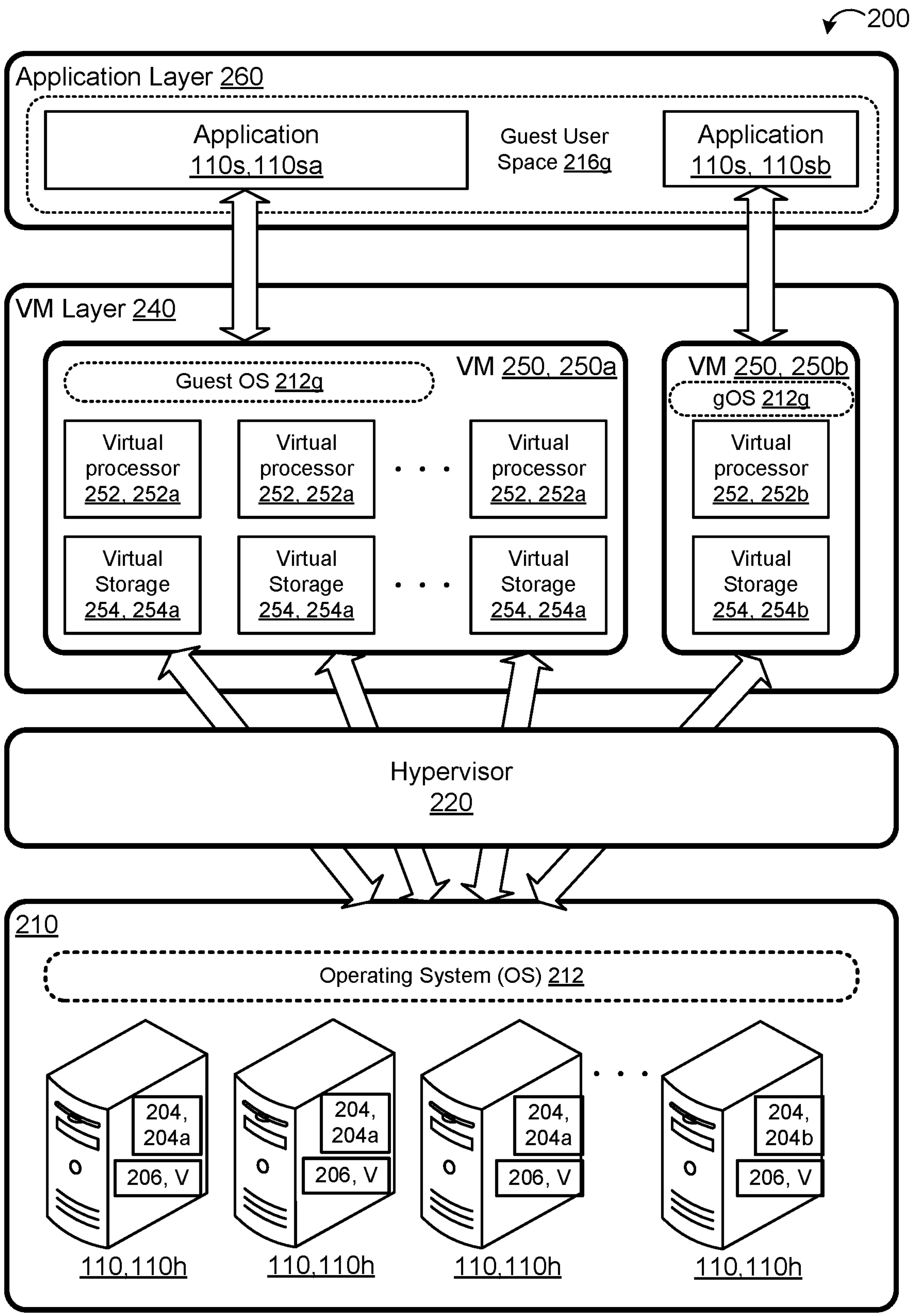

200
Application Layer 260
Application
110s,110sa
Guest User
Space 216g
Application
110s, 110sb
VM Layer 240
Guest OS 212g
VM 250, 250a
VM 250, 250b
gOS 212g
Virtual
processor
252, 252a
Virtual
processor
252, 252a
Virtual
processor
252, 252a
Virtual
processor
252, 252b
Virtual
Storage
254, 254a
Virtual
Storage
254, 254a
Virtual
Storage
254, 254a
Virtual
Storage
254, 254b
Hypervisor
220
210
Operating System (OS) 212
204,
204a
206, V
204,
204a
206, V
204,
204a
206, V
204,
204b
206, V
110,110h
110,110h
110,110h
110,110h

FIG. 2

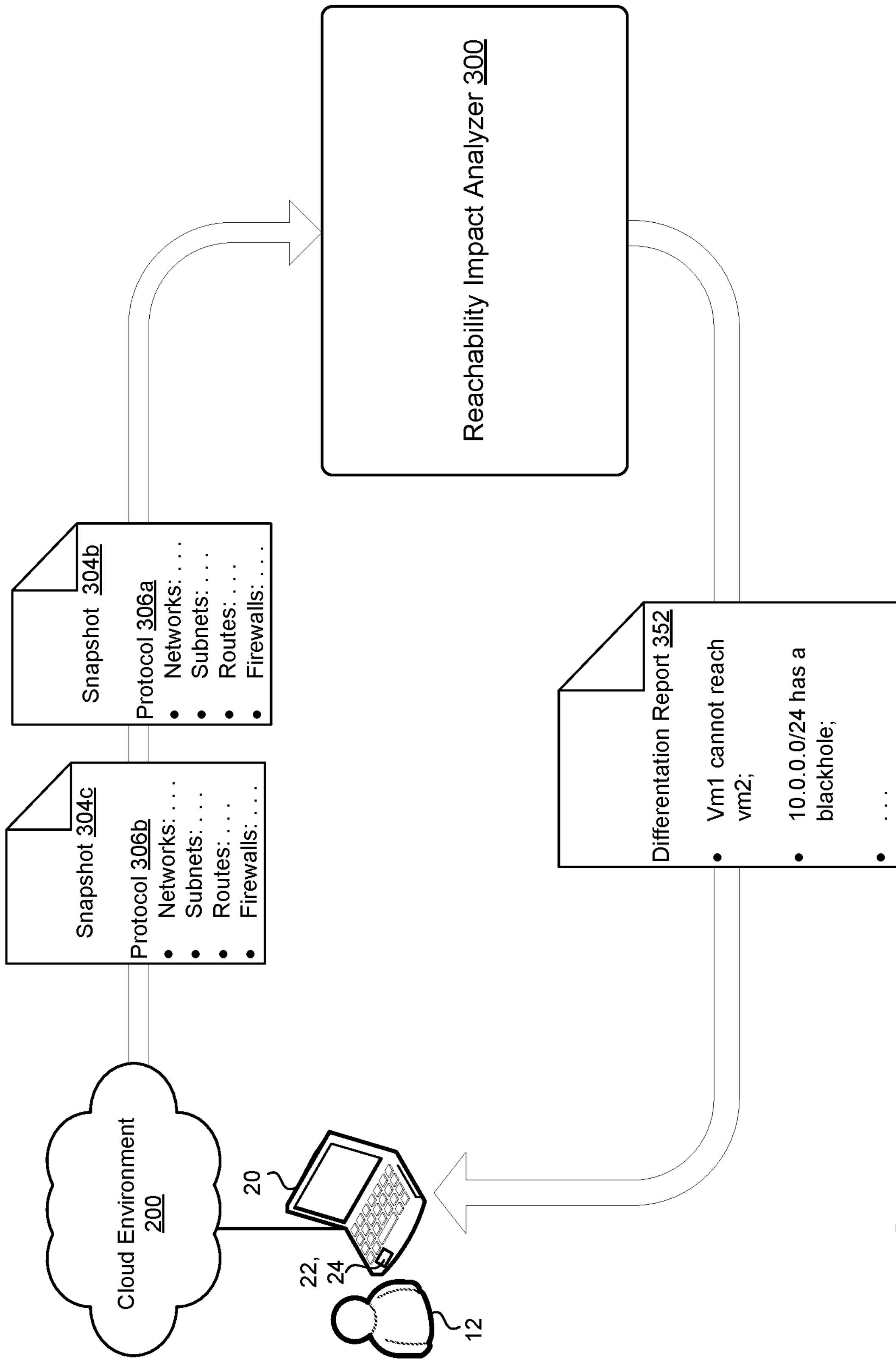

Reachability Impact Analyzer 300
Snapshot 304b
Protocol 306a
Networks: . . .
Subnets: . . .
Routes: . . .
Firewalls: . . .
Snapshot 304c
Protocol 306b
Networks: . . .
Subnets: . . .
Routes: . . .
Firewalls: . . .
Differentation Report 352
Vm1 cannot reach vm2;
10.0.0.0/24 has a blackhole;
. . .
Cloud Environment 200
20
22, 24
12

FIG. 3

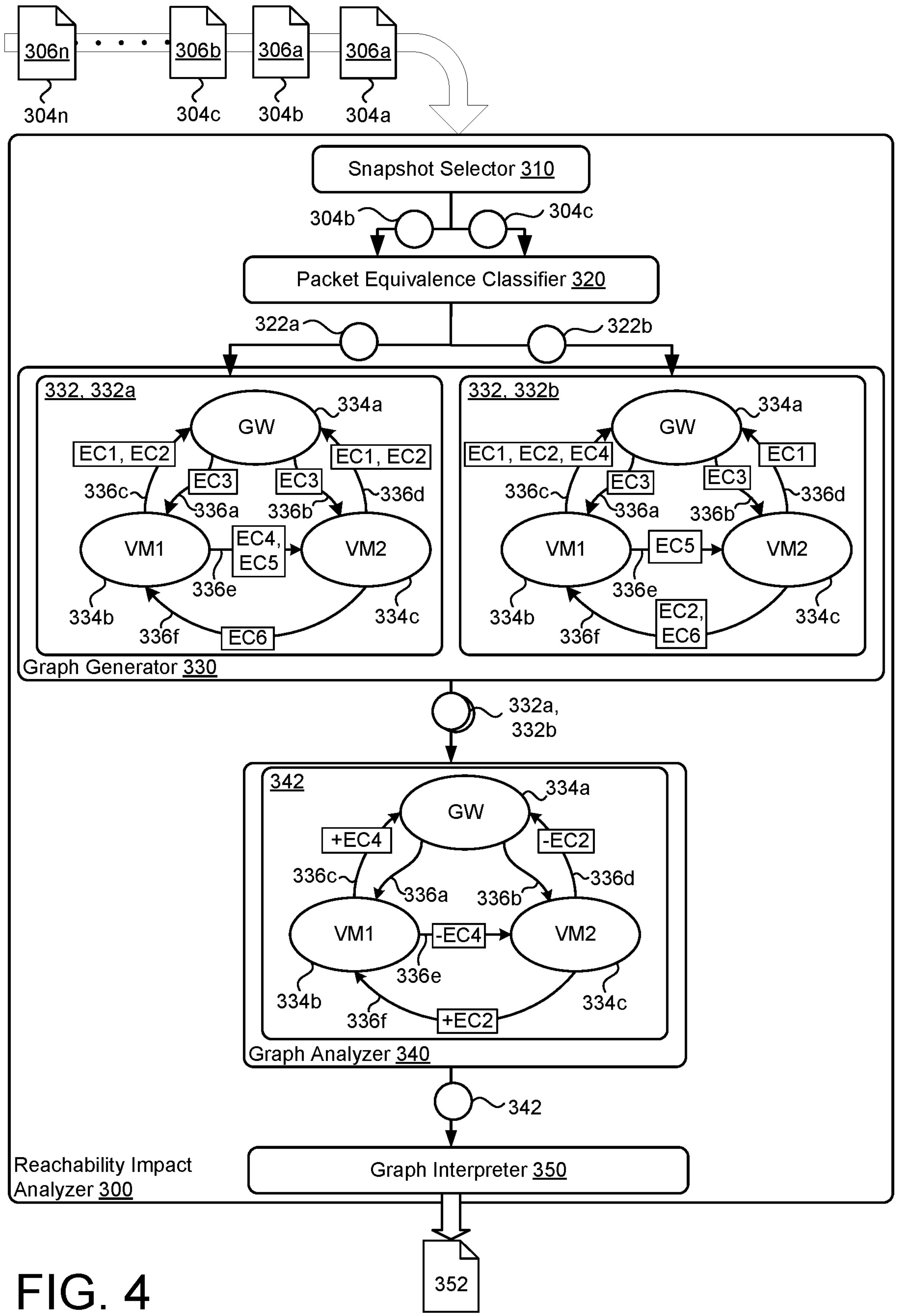

306n
306b
306a
306a
304n
304c
304b
304a
Snapshot Selector 310
304b
304c
Packet Equivalence Classifier 320
322a
322b
332, 332a
GW
334a
EC1, EC2
EC1, EC2
EC3
EC3
336c
336a
336b
336d
VM1
EC4, EC5
VM2
336e
334b
336f
EC6
334c
332, 332b
GW
334a
EC1, EC2, EC4
EC1
EC3
EC3
336c
336a
336b
336d
VM1
EC5
VM2
336e
334b
336f
EC2, EC6
334c
Graph Generator 330
332a, 332b
342
GW
334a
+EC4
-EC2
336c
336a
336b
336d
VM1
-EC4
VM2
336e
334b
336f
+EC2
334c
Graph Analyzer 340
342
Reachability Impact Analyzer 300
Graph Interpreter 350
352

FIG. 4

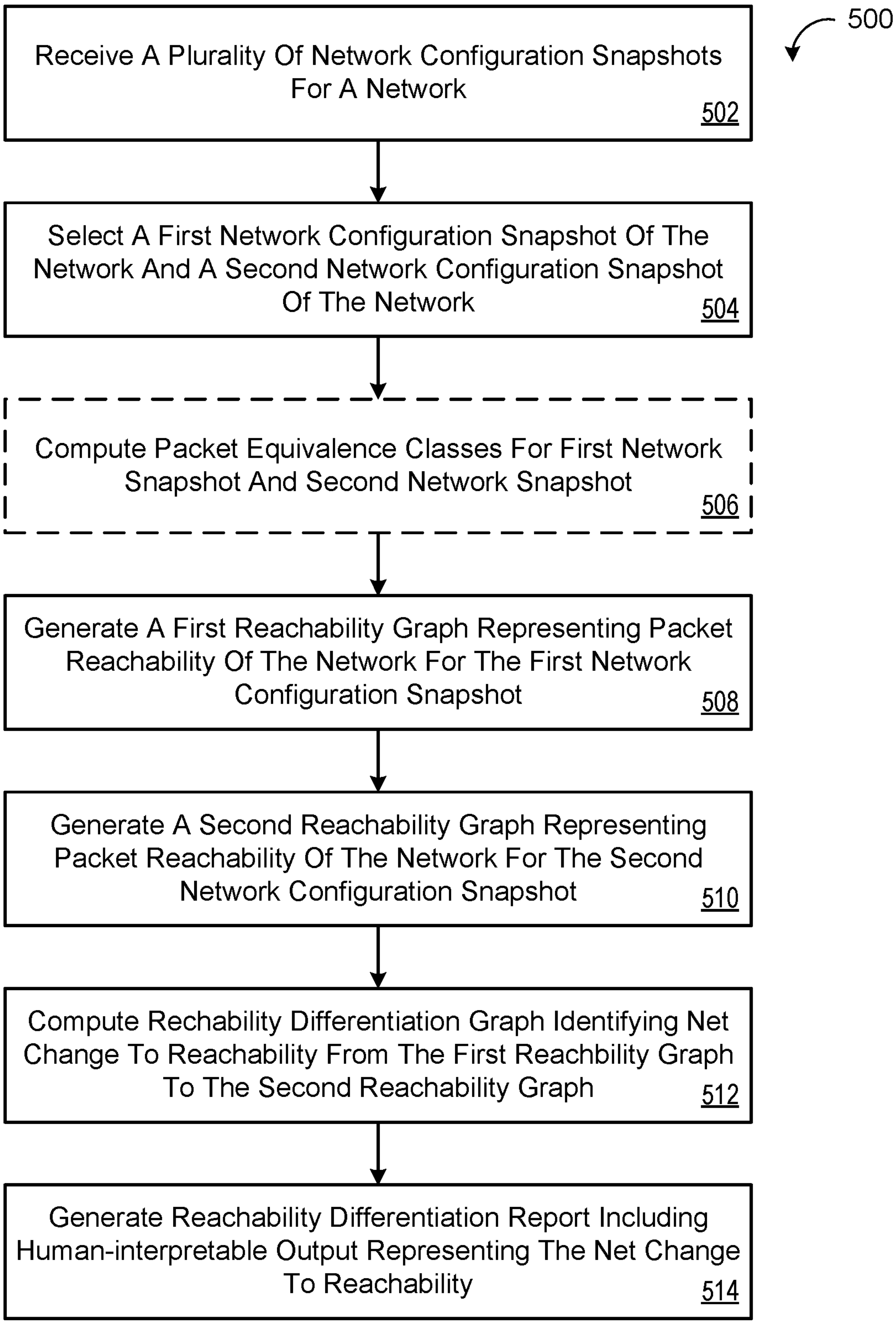

500
Receive A Plurality Of Network Configuration Snapshots For A Network
502
Select A First Network Configuration Snapshot Of The Network And A Second Network Configuration Snapshot Of The Network
504
Compute Packet Equivalence Classes For First Network Snapshot And Second Network Snapshot
506
Generate A First Reachability Graph Representing Packet Reachability Of The Network For The First Network Configuration Snapshot
508
Generate A Second Reachability Graph Representing Packet Reachability Of The Network For The Second Network Configuration Snapshot
510
Compute Rechability Differentiation Graph Identifying Net Change To Reachability From The First Reachbility Graph To The Second Reachability Graph
512
Generate Reachability Differentiation Report Including Human-interpretable Output Representing The Net Change To Reachability
514

FIG. 5

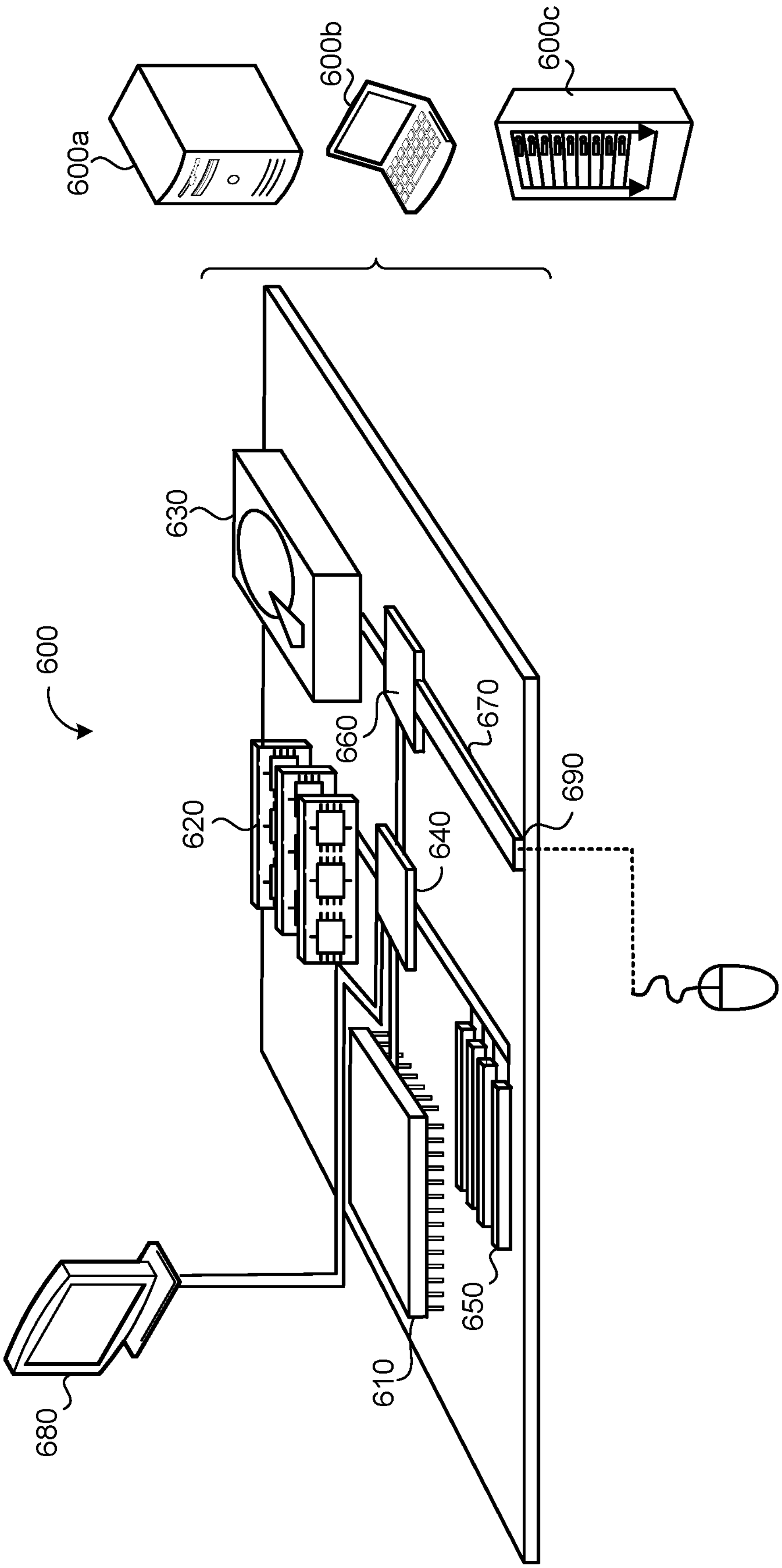


FIG. 6

NETWORK REACHABILITY IMPACT ANALYSIS

CROSS REFERENCE TO RELATED APPLICATIONS

[0001] This U.S. patent application is a continuation of, and claims priority under 35 U.S.C. § 120 from, U.S. patent application Ser. No. 17/117,376, filed on Dec. 10, 2020. The disclosure of this prior application is considered part of the disclosure of this application and is hereby incorporated by reference in its entirety.

TECHNICAL FIELD

[0002] This disclosure relates to reachability impact analysis of a cloud network.

BACKGROUND

[0003] A virtual private cloud (VPC) is an on-demand configurable pool of shared computing resources allocated within a public cloud environment. The VPC provides isolation for a user from other cloud users. The VPC may execute one or more virtual machines (VMs) which may communication with the user's on-premises network or other remote resources via a virtual private network (VPN). Due to the potential scale and complexity of the VPC, which may include any number of VMs, network gateways, load balancers, etc., significant network configuration is often necessary to operate and maintain the VPC.

SUMMARY

[0004] One aspect of the disclosure provides a method of network reachability impact analysis. The method includes receiving, at data processing hardware, a plurality of network configuration snapshots for a network. The method also includes selecting, by the data processing hardware, a first network configuration snapshot of the network and a second network configuration snapshot of the network. The method further includes generating, by the data processing hardware, a first reachability graph representing packet reachability of the network for the first network configuration snapshot. The method also includes generating, by the data processing hardware, a second reachability graph representing packet reachability of the network for the second network configuration snapshot. The method also includes computing, by the data processing hardware, a reachability differentiation graph identifying a net change to reachability from the first reachability graph to the second reachability graph. The method further includes generating, by the data processing hardware, a reachability differentiation report including a human-interpretable output of the net change to reachability.

[0005] Implementations of the disclosure may include one or more of the following optional features. In some implementations, generating the reachability differentiation report further includes translating, by the data processing hardware the reachability differentiation graph from a computer-interpretable format to the human-interpretable output. In some examples, each of the plurality of network configuration snapshots includes a data plane model protocol taken at a respective time instance. Here, selecting the first network configuration snapshot and the second network configuration snapshot may include comparing, by the data processing hardware, the data plane model protocols of consecutive network configuration snapshots Optionally, selecting the first network configuration snapshot and the second network configuration snapshot includes, when a first data plane model protocol of a first one of the consecutive network configuration snapshots is different than a second data plane model protocol of a second one of the consecutive network configuration snapshots, selecting, by the data processing hardware, the first one of one of the consecutive network configuration snapshots as the first network configuration snapshot and the second one of the consecutive network configuration snapshots as the second network configuration snapshot.

[0006] In some implementations, the method further includes computing, by the data processing hardware, one or more packet equivalence classes for the first network configuration snapshot and the second network configuration snapshot, each of the one or more packet equivalence classes including a set of packets having the same forwarding behavior. Optionally, the method further includes assigning, by the data processing hardware, the one or more packet equivalence classes to the first reachability graph, and assigning, by the data processing hardware, the one or more packet equivalence classes to the second reachability graph. In some examples, computing the reachability differentiation graph includes identifying a net change to network equivalence classes from the first reachability graph to the second reachability graph.

[0007] In some configurations, the method includes generating the first reachability graph and the second reachability graph includes generating, by the data processing hardware, a directed graph including two or more nodes and one or more edges connecting each of the two or more nodes. Here, generating the directed graph may include associating, by the data processing hardware, each of the two or more nodes to a network endpoint and associating each of the one or more edges to a network forwarding route from one network endpoint to another network endpoint.

[0008] Another aspect of the disclosure provides a system. The system includes data processing hardware and memory hardware in communication with the data processing hardware. The memory hardware stores instructions that when executed on the data processing hardware cause the data processing hardware to perform operations. One operation includes receiving a plurality of network configuration snapshots for a network. Another operation includes selecting a first network configuration snapshot of the network and a second network configuration snapshot of the network. The operations further include generating a first reachability graph representing packet reachability of the network for the first network configuration snapshot. Another operation includes generating a second reachability graph representing packet reachability of the network for the second network configuration snapshot. The operations further include computing a reachability differentiation graph identifying a net change to reachability from the first reachability graph to the second reachability graph, and generating a reachability differentiation report including a human-interpretable output of the net change to reachability.

[0009] This aspect of the disclosure may include one or more of the following optional features. In some examples, generating the reachability differentiation report further includes translating the reachability differentiation graph from a computer-interpretable format to the human-interpretable output. In some examples, each of the plurality of network configuration snapshots includes a data plane model protocol taken at a respective time instance. Here, selecting the first network configuration snapshot and the second network configuration snapshot may further include comparing the data plane model protocols of consecutive network configuration snapshots. Optionally, selecting the first network configuration snapshot and the second network configuration snapshot includes, when a first data plane model protocol of a first one of the consecutive network configuration snapshots is different than a second data plane model protocol of a second one of the consecutive network configuration snapshots, selecting the first one of one of the consecutive network configuration snapshots as the first network configuration snapshot and the second one of the consecutive network configuration snapshots as the second network configuration snapshot.

[0010] In some examples, the operations further include computing one or more packet equivalence classes for the first network configuration snapshot and the second network configuration snapshot, each of the one or more packet equivalence classes including a set of packets having the same forwarding behavior. Here, the operations further include assigning the one or more packet equivalence classes to the first reachability graph and assigning the one or more packet equivalence classes to the second reachability graph. Optionally, computing the reachability differentiation graph includes identifying a net change to network equivalence classes from the first reachability graph to the second reachability graph.

[0011] In some implementations, generating the first reachability graph and the second reachability graph includes generating a directed graph including two or more nodes and one or more edges connecting each of the two or more nodes. Here, generating the directed graph includes associating each of the two or more nodes to a network endpoint and associating each of the one or more edges to a network forwarding route from one network endpoint to another network endpoint.

[0012] The details of one or more implementations of the disclosure are set forth in the accompanying drawings and the description below. Other aspects, features, and advantages will be apparent from the description and drawings, and from the claims.

DESCRIPTION OF DRAWINGS

[0013] FIG. **1** is a schematic view of an example system for performing network reachability impact analysis.

[0014] FIG. **2** is a schematic view of exemplary components of a virtual machine of the system of FIG. **1**.

[0015] FIG. **3** is a schematic view of an example system for performing network reachability impact analysis.

[0016] FIG. **4** is a schematic view of an example system for performing network reachability impact analysis.

[0017] FIG. **5** is a flowchart of an example arrangement of operations for a method of performing cloud network reachability analysis.

[0018] FIG. **6** is a schematic view of an example computing device that may be used to implement the systems and methods described herein.

[0019] Like reference symbols in the various drawings indicate like elements.

DETAILED DESCRIPTION

[0020] A virtual private cloud (VPC) is an on-demand configurable pool of shared computing resources allocated within a public cloud environment to provide isolation for a user from other cloud users. This isolation may occur through allocation of private Internet Protocol (IP) subnets and/or virtual communication constructs. The VPC may execute one or more virtual machines (VMs) which may communication with the user's on-premises network or other remote resources via a virtual private network (VPN) to ensure secure access to the VPC environment. Because some VPC environments are very complex with a very large scale (i.e., include a number of VMs, network gateways, load balancers, etc.), significant network configuration is often necessary to operate and maintain the VPC.

[0021] Implementations herein are directed toward a cloud reachability impact analyzer that allows a user understand the impact that changes to the configuration of the network will have on packet reachability within the network. The cloud reachability impact analyzer generates directed graphs representing network reachability for two network configuration snapshots. The cloud reachability impact analyzer then performs a reachability analysis on the graphs to identify changes to reachability caused by the network configuration changes between the two network configuration snapshots. Thus, the cloud reachability impact analyzer allows the user to verify how a network configuration change will affect packet reachability relative to a previous network configuration.

[0022] Referring to FIG. **1**, in some implementations, an example system **10** includes a user device **20** associated with a respective user **12** and in communication with a cloud network **200** via a network **30** (e.g., the Internet) and an on-premises network **40** (i.e., the local network that the user device **20** uses to connect to the network **30**). The on-premises network **40** includes a network gateway **42** (e.g., a router) that serves as the forwarding host for the on-premises network **40**. The user device **20** may correspond to any computing device, such as a desktop workstation, a laptop workstation, or a mobile device (e.g., a smart phone or tablet). The user device **20** includes computing resources **22** (e.g., data processing hardware) and/or storage resources **24** (e.g., memory hardware).

[0023] The cloud network **200** may be a single computer, multiple computers, or a distributed system (e.g., a cloud environment) having scalable/elastic resources **202** including computing resources **204** (e.g., data processing hardware) and/or storage resources **206** (e.g., memory hardware). A data store (i.e., a remote storage device) may be overlain on the storage resources **206** to allow scalable use of the storage resources **206** by one or more of the client or computing resources **204**. The cloud network **200** is configured to implement and execute one or more virtual machines (VMs) **250**, **250***a*—n. One or more of the VMs execute securely in a virtual private cloud (VPC) environment or VPC **208** associated with or operated by the user **12**. The VPC **208** may include a variety of other network elements, such as load balancers, gateways, front ends, and back ends.

[0024] In the example shown in FIG. **2**, the distributed system **200** includes a collection **210** of resources **110** (e.g., hardware resources **110***h*), a virtual machine monitor (VMM) **220**, a VM layer **240** executing one or more of the VMs **250**, and an application layer **260**. Each hardware resource **110***h* may include one or more physical central processing units (pCPU) **204** ("physical processor **204**") and memory hardware **206**. While each hardware resource **110***h* is shown having a single physical processor **204**, any hardware resource **110***h* may include multiple physical processors **204**. An operating system **212** may execute on the collection **210** of resources **110**.

[0025] In some examples, the VMM **220** corresponds to a hypervisor **220** (e.g., a Compute Engine) that includes at least one of software, firmware, or hardware configured to create and execute the VMs **250**. A computer (i.e., data processing hardware **204**) associated with the VMM **220** that executes the one or more VMs **250** may be referred to as a host machine, while each VM **250** may be referred to as a guest machine. Here, the VMM **220** or hypervisor is configured to provide each VM **250** a corresponding guest operating system (OS) **212***g* having a virtual operating platform and manage execution of the corresponding guest OS **212***g* on the VM **250**. As used herein, each VM **250** may be referred to as an "instance" or a "VM instance". In some examples, multiple instances of a variety of operating systems may share virtualized resources. For instance, a first VM **250** of the Linux® operating system, a second VM **250** of the Windows® operating system, and a third VM **250** of the OS X® operating system may all run on a single physical x86 machine.

[0026] The VM layer **240** includes one or more virtual machines **250**. The distributed system **200** enables the user **12** to launch VMs **250** on demand. A VM **250** emulates a real computer system and operates based on the computer architecture and functions of the real computer system or a hypothetical computer system, which may involve specialized hardware, software, or a combination thereof. In some examples, the distributed system **200** authorizes and authenticates the user **12** before launching the one or more VMs **250**. An instance of software, or simply an instance, refers to a VM **250** hosted on (executing on) the data processing hardware **204** of the distributed system **200**.

[0027] Each VM **250** may include one or more virtual central processing units (vCPUs) **252** ("virtual processor"). In the example shown, a first virtual machine **250***a* includes a first set **252***a* of one or more virtual processors **252** and a second virtual machine **250***b* includes a second set **252***b* of one or more virtual processors **252**. While the second set **252***b* is shown as only including one virtual processor **252**, any number of virtual processors **252** is possible. Each virtual processor **252** emulates one or more physical processors **204**. For example, the first set **252***a* of the one or more virtual processors **252** emulates a first set **204***a* of one or more physical processors **204**, and the second set **252***b* of the one or more virtual processors **252** emulates a second set **204***b* of one or more physical processors **204**. The application layer **260** includes software resources **110***s*, **110***sa*, **110***sb* (software applications) that may execute on the virtual machine(s) **250**.

[0028] Typically, each instance of software (e.g., a virtual machine **250**) includes at least one virtual storage device **254** that provides volatile and non-volatile storage capacity for the service on the physical memory hardware **206**. For instance, the storage capacity on the physical memory hardware **206** can include persistent disks (PD) that store data for the user **12** across several physical disks (e.g., memory regions **620** (FIG. **6**) of the memory hardware **206** or random access memory (RAM) to provide volatile memory. More specifically, each virtual storage device **254** of a corresponding VM **250** moves data in sequences of bytes or bits (blocks) to an associated physical block storage volume V on the memory hardware **206** to provide non-volatile storage. Accordingly, a virtual storage device **254** of a corresponding VM instance **250** provides a storage capacity that maps to corresponding physical block storage volumes V on the memory hardware **206**. In some examples, the virtual storage devices **254** support random access to the data on the memory hardware **206** and generally use buffered I/O. Examples include hard disks, CD-ROM drives, and flash drives. Similarly, portions of volatile memory (e.g., RAM) of physical memory hardware **206** may be divided across the virtual storage devices **254**.

[0029] Within the guest operating system **212***g* resides a guest kernel **214***g*. A kernel is a computer program that is the core of the operating system with full access and control over the OS. That is, the kernel is an intermediary between applications **110***s* and the hardware resources **110***h* of the host machine. Most modern computing systems segregate virtual memory into protected kernel space and user space **216***g*. The kernel typically remains in volatile memory within the protected kernel space and is isolated from user space **216***g*. To increase safety and reliability, applications **110***s* and other software services typically execute in the guest user space **216***g* and lack the privileges necessary to interact with the protected kernel space.

[0030] Referring to FIGS. **1** and **3**, the cloud network **200** executes a cloud reachability impact analyzer **300** for analyzing network configuration snapshots **304**, **304***a*-**304***n* of the cloud network **200** to determine differences in packet reachability between two consecutive network configuration snapshots **304**, **304***a*-**304***n*. The cloud reachability impact analyzer **300** then generates a human-interpretable differentiation report **352** identifying the differences in packet reachability between the analyzed configuration snapshots **304** and presents the differentiation report **352** to the user **12** via the user device **20**. In some examples, the analyzer **300** determines whether changes to the network **208** result in a policy violation that affects reachability, and localize a configuration stanza responsible for the policy violation.

[0031] The cloud reachability impact analyzer **300** continuously receives or obtains the network configuration snapshots **304**, **304***a*-**304***n* from the cloud network **200**. The network configuration snapshots **304** are provided by the cloud network **200** in a format of a data plane model protocol **306** including network configuration information. Optionally, the cloud network **200** may execute a data plane modeler **302** that obtains the network configuration information from network components of the VPC **208** and includes, for example, routes between network resources (e.g., VMs, load balances, network gateways, etc.) of the VPC **208**, subnets, firewall rules, and/or ports or interfaces for directing a data packet within the VPC **208** and/or between the VPC **208** and other networks (e.g., other VPCs and/or the on-premises network **40**).

[0032] In FIG. **1**, the network configuration snapshots **304** include a first network configuration snapshot **304***a* including a first data plane model protocol **306***a* of the VPC **208** at a first time instance, a second network configuration snapshot **304***b* including the first data plane model protocol **306***a* of the VPC **208** at a second time instance immediately subsequent to the first time instance, a third network configuration snapshot **304***c* including a second data plane model protocol **306***b* of the VPC **208** at a third time instance immediately subsequent to the second time instance, and a plurality of subsequent configuration snapshots **304***n* including data plane model protocols **306***n* taken at time instances following the third time instance.

[0033] Consecutive ones of the network configuration snapshots **304** may include the same data plane model protocol **306** when the network configuration is not changed from one time instance to the next. For example, in the illustrated example, the first network configuration snapshot **304***a* and the second network configuration snapshot **304***b* include the same first data plane model protocol **306***a* associated with an unchanged network configuration at the first and second time instances. Alternatively, subsequent ones of the network configuration snapshots **304** may include different data plane model protocols **306** when the network configuration is changed between time instances. For example, in FIG. **1**, a change **26***a*, **26***b* to a configuration or state of the VPC **208** is incorporated between the second network configuration snapshot **304***b* associated with the second time instance and the third network configuration snapshot **304***c* associated with the third time instance. Thus, the third network configuration snapshot **304***c* has a different data plane model protocol **306***b* than the data plane model protocol **306***a* of the immediately preceding network configuration snapshot **304***b*.

[0034] Examples of network changes **26***a*, **26***b* include a user change **26***a* implemented by the user **12** via the user device **20** or a system change **26***b* caused by the cloud network **200**. User changes **26***a* may include pending changes proposed by the user **12** or changes that have already been deployed. System changes **26***b* may include automated configuration changes incorporated by network monitoring applications and/or network state changes associated with involuntary changes in the VPC **208** (e.g., operation states down). For clarity, the changes **26***a*, **26***b* are shown as being incorporated between the illustrated second and third network configuration snapshots **304***b*, **304***c* in FIG. **1**. However, the actual changes **26***a*, **26***b* are implemented within the network **208**, upstream up the data plane modeler **302**. Thus, the second network configuration snapshot **304***b* includes the pre-change first data plane model protocol **306***a* and the third network configuration snapshot **304***c* includes the post-change second data plane model protocol **306***b*.

[0035] As generally illustrated in FIG. **3**, the cloud reachability impact analyzer receives the network configuration snapshots **304***b*, **304***c* including the pre-change first data plane model protocol **306***a* and the post-change second data plane model protocol **306***b* and generates the human-interpretable differentiation report **352** identifying changes to forwarding behavior between the two network configuration snapshots **304***b*, **304***c*. Human-interpretable format includes providing descriptions of the changes to the forwarding behavior using written representations of the differences in network topologies. For example, the differentiation report **352** may include text describing that one of the VMs **250** has become unreachable in the third network configuration snapshot **304***c*, or that prefix 10.0.0.0/24 goes to a VPN tunnel in the second network configuration snapshot **304***b* and to a subnet in the third network configuration snapshot **304***c*.

[0036] With continued reference to FIG. **1**, the cloud reachability impact analyzer **300** includes an optional snapshot selector **310** configured to extract consecutive network configuration snapshots **304** from the data plane modeler **302**. The cloud reachability impact analyzer **300** also includes a packet equivalence classifier **320** that computes packet equivalences for each of the selected consecutive network configuration snapshots **304**. The cloud reachability impact analyzer **300** further includes a graph generator **330** that creates a reachability graph **332***a*, **332***b* corresponding to each of the selected network configuration snapshots **304**. A graph analyzer **340** receives and compares the reachability graphs **332***a*, **332***b* to generate a differentiation graph **342**. The cloud reachability impact analyzer **300** further includes a graph interpreter **350** that evaluates the differentiation graph **342** and generates the human-interpretable differentiation report **352**.

[0037] FIG. **4** shows a more detailed schematic illustrating the configuration and operation of the cloud reachability impact analyzer **300**. As previously discussed, the cloud reachability impact analyzer **300** receives or obtains a continuous feed of the network configuration snapshots **304** from the cloud network **200**. More specifically, when included, the snapshot selector **310** continuously receives and compares the network configuration snapshots **304** to determine when one or more changes **26***a*, **26***b* have been implemented in the VPC **208**. Thus, in the example shown, the snapshot selector **310** executes a first comparison between the first network configuration snapshot **304***a* and the second network configuration snapshot **304***b* and does not determine that any changes **26***a*, **26***b* have been implemented, as both snapshots **304***a*, **304***b* include the same data plane model protocol **306***a*. Subsequently, the snapshot selector **310** executes a second comparison between the second network configuration snapshot **304***b* and the third network configuration snapshot **304***c* and identifies that the one or more changes **26***a*, **26***b* have been implemented in the third network configuration snapshot **304***c* where the second network configuration snapshot **304***b* includes a different data plane model protocol **306***a* than the data plan model protocol **306***b* of the third network configuration snapshot **304***c*. The snapshot selector **310** then selects the second network configuration snapshot **304***b* immediately preceding the change **26***a*, **26***b* and the third network configuration snapshot **304***c* including the change **26***a*, **26***b*.

[0038] While the snapshot selector **310** may automatically select the network configuration snapshots **304**, as described here, in other examples the snapshot selector **310** may receive instructions for selecting the network configuration snapshots **304***b*, **304***c* from the user device **20** or the cloud network **200** in conjunction with one of the changes **26***a*, **26***b* being implemented. For instance, the user **12** or the cloud network **200** may provide instructions to the snapshot selector **310** including information identifying the consecutive network configuration snapshots **304***b*, **304***c* and/or the time instances associated with the change **26***a*, **26***b*.

[0039] The packet equivalence classifier **320** receives the selected network configuration snapshots **304***b*, **304***c* from the snapshot selector **310** and computes packet equivalence classes for each of the network configuration snapshots **304***b*, **304***c*. A packet equivalence class EC, EC**1**-EC**6** represents a set of packets that have the same forwarding behavior with respect to all configuration rules of both of the network configuration snapshots **304***b*, **304***c*. For example, the illustrated examples of the network configuration snapshots **304***b*, **304***c* include a total of six of the equivalence classes EC1-EC6. Optionally, the packet equivalence classifier 320 may compile the computed equivalence classes EC1-EC6 into a first subgroup 322*a* associated with the second network configuration snapshot 304*b* and a second subgroup 322*b* associated with the third network configuration snapshot 304*c*.

[0040] The graph generator 330 receives the equivalence class subgroups 322*a*, 322*b* including the computed equivalence classes EC1-EC6 and builds reachability graphs 332*a*, 332*b* representing routing of the equivalence classes EC1-EC6 relative to the network topologies associated with each of the respective network configuration snapshots 304*b*, 304*c*. As shown, each of the reachability graphs 332*a*, 332*b* includes a directed graph 332 having a plurality of nodes 334, 334*a*-334*c* and edges 336, 336*a*-336*f*. Each of the nodes 334 represents a network endpoint associated with a network resource, such as a VM 250 or a network gateway 42. Each of the edges 336 represents a forwarding route from one of the nodes 334 to another one of the nodes 334. In the illustrated example, the network reachability graphs 332*a*, 332*b* each represent a network topology including a gateway node 334*a*, a first VM node 334*b*, and a second VM node 334*c*.

[0041] Each node 334*a*-334*c* is connected each other node 334*a*-334*c* by an edge 336*a*-336*f* representing a forwarding route from one endpoint to another. A first edge 336*a* represents a forwarding route from the gateway node 334*a* to the first VM node 334*b*, a second edge 336*b* represents a forwarding route from the gateway node 334*a* to the second VM node 334*c*, a third edge 336*c* represents a forwarding route from the first VM node 334*b* to the gateway node 334*a*, a fourth edge 336*d* represents a forwarding route from the second VM node 334*c* to the gateway node 334*a*, a fifth edge 336*e* represents a forwarding route from the first VM node 334*b* to the second VM node 334*c*, and a sixth edge 336*f* represents a forwarding route from the second VM node 334*c* to the first VM node 334*b*.

[0042] The graph generator 330 uses the reachability graphs 332*a*, 332*b* to model which of the equivalence classes EC1-EC6 are allowed to travel through each edge for each reachability graph 332*a*, 332*b*. Thus, as shown, the graph generator 330 assigns each of the equivalence classes EC1-EC6 to respective ones of the edges 336*a*-336*f* that the equivalence class EC1-EC6 is allowed to travel along. Here, assignments are illustrated by labeling each edge 336*a*-336*c* with the corresponding equivalence classes EC1-EC6 that are allowed to travel along the edge 336*a*-336*c*.

[0043] The reachability graphs 332*a*, 332*b* created by the graph generator 330 are forwarded to the graph analyzer 340, which evaluates the reachability graphs 332*a*, 332*b* to determine an impact to reachability between the two reachability graphs 332*a*, 332*b*. In other words, the graph analyzer 340 compares the reachability graphs 332*a*, 332*b* to identify differences in reachability between each of the nodes 334*a*-334*c* caused by implementing the changes 26*a*, 26*b*. The graph analyzer 340 models the reachability impact as a differentiation graph 342 including the same nodes 334*a*-334*c* and edges 336*a*-336*f* as the reachability graphs 332*a*, 332*b*. The reachability impact analyzer 340 then computes a net change (e.g., addition/removal of equivalence classes) for each edge 336*a*-336*f* to determine the impact to reachability from the second network configuration snapshot 304*b* to the third network configuration snapshot 304*c*.

[0044] In the illustrated example, the differentiation graph 342 shows that reachability along the first and second edges 336*a*, 336*b* corresponding to the forwarding paths from the gateway node 334*a* to each of the VM nodes 334*b*, 334*c* is unchanged between the first reachability graph 332*a* and the second reachability graph 332*b*. However, the differentiation graph 342 shows that the remaining edges 336*c*-336*f* each include changes corresponding to added or removed allowances of equivalence classes EC1-EC6. Particularly, the fourth equivalence class EC4 is added +EC4 to the third edge 336*c* and removed −EC4 from the fifth edge 336*e*, representing that the fourth equivalence class EC4 can now travel to (i.e., reach) the gateway node 334*a* from the first VM node 334*b*, but cannot travel to (i.e., reach) the second VM node 336*c* from the first VM node 334*b*. Additionally, the second equivalence class EC2 is added +EC2 to the sixth edge 336*f* and deleted from the fourth edge 336*d*, representing that the second equivalence class EC2 is allowed to travel to (i.e., reach) the first VM node 334*b* from the second VM node 334*c* and cannot traveling to (i.e., reach) the gateway node 334*a* from the second VM node 334*c*.

[0045] In some examples, the graph interpreter 350 receives the differentiation graph 342 from the graph analyzer 340 and translates the graphical representation of the reachability changes +/−EC2, +/−EC4 into the human-interpretable differentiation report 352. For example, the graph interpreter 350 translates the reachability changes +/−EC2, +/−EC4 of the directed graph into a text-based differentiation report 352 identifying the impact of the changes 26*a*, 26*b* on reachability. In some examples, the graph interpreter 350 analyzes the reachability changes +/−EC2, +/−EC4 relative to one or more network intentions identified by the user 12. Here, the graph interpreter 350 may highlight or identify specific ones of the reachability changes +/−EC2, +/−EC4 that will impact the user intention for the network.

[0046] Using the differentiation report 352, a network user 12 can determine whether changes 26*a*, 26*b* to a network topology of a VPC 208 should be implemented. Where the impact on reachability is unintended and/or unacceptable, the user 12 may decline the changes 26*a*, 26*b* or revert the network configuration to a state prior to the change 26*a*, 26*b*. In some instances, the reachability impact analyzer 300 may be implemented in conjunction with a network change simulator to model and analyze reachability impact prior to the change 26*a*, 26*b* being implemented on the production VPC 208.

[0047] FIG. 5 is a flowchart of an exemplary arrangement of operations for a method 500 of performing cloud network reachability impact analysis. The method 500 includes, at operation 502, receiving, at data processing hardware 202, a plurality of network configuration snapshots 304, 304*a*-304*n* for a network 208. Optionally, the network configuration snapshots 304, 304*a*-304*n* include data plane model protocols 306, 306*a*-306*n*. At operation 504, the method 500 includes selecting a first network configuration snapshot 304*b* and a second network configuration snapshot 304*c* of the network 208. The method 500 may also include computing, at operation 506, packet equivalence classes for each of the first network snapshot and the second network snapshot. The method 500 also includes, at operation 508, generating a first reachability graph 332*a* representing packet reachability of the network 208 for the first network configuration snapshot 304*b*. At operation 510, the method 500 includes generating a second reachability graph 332*b* representing packet reachability of the network **208** for the second network configuration snapshot **304***c*. The method **500** also includes, at operation **512**, computing a reachability differentiation graph **342** that identifies net change to reachability from the first reachability graph **332***a* to the second reachability graph **332***b*. At operation **514**, the method **500** includes generating a differentiation report **352** including human-interpretable output representing a net change to reachability.

[0048] FIG. **6** is schematic view of an example computing device **600** that may be used to implement the systems and methods described in this document. The computing device **600** is intended to represent various forms of digital computers, such as laptops, desktops, workstations, personal digital assistants, servers, blade servers, mainframes, and other appropriate computers. The components shown here, their connections and relationships, and their functions, are meant to be exemplary only, and are not meant to limit implementations of the inventions described and/or claimed in this document.

[0049] The computing device **600** includes a processor **610**, memory **620**, a storage device **630**, a high-speed interface/controller **640** connecting to the memory **620** and high-speed expansion ports **650**, and a low speed interface/controller **660** connecting to a low speed bus **670** and a storage device **630**. Each of the components **610**, **620**, **630**, **640**, **650**, and **660**, are interconnected using various busses, and may be mounted on a common motherboard or in other manners as appropriate. The processor **610** can process instructions for execution within the computing device **600**, including instructions stored in the memory **620** or on the storage device **630** to display graphical information for a graphical user interface (GUI) on an external input/output device, such as display **680** coupled to high speed interface **640**. In other implementations, multiple processors and/or multiple buses may be used, as appropriate, along with multiple memories and types of memory. Also, multiple computing devices **600** may be connected, with each device providing portions of the necessary operations (e.g., as a server bank, a group of blade servers, or a multi-processor system).

[0050] The memory **620** stores information non-transitorily within the computing device **600**. The memory **620** may be a computer-readable medium, a volatile memory unit(s), or non-volatile memory unit(s). The non-transitory memory **620** may be physical devices used to store programs (e.g., sequences of instructions) or data (e.g., program state information) on a temporary or permanent basis for use by the computing device **600**. Examples of non-volatile memory include, but are not limited to, flash memory and read-only memory (ROM)/programmable read-only memory (PROM)/erasable programmable read-only memory (EPROM)/electronically erasable programmable read-only memory (EEPROM) (e.g., typically used for firmware, such as boot programs). Examples of volatile memory include, but are not limited to, random access memory (RAM), dynamic random access memory (DRAM), static random access memory (SRAM), phase change memory (PCM) as well as disks or tapes.

[0051] The storage device **630** is capable of providing mass storage for the computing device **600**. In some implementations, the storage device **630** is a computer-readable medium. In various different implementations, the storage device **630** may be a floppy disk device, a hard disk device, an optical disk device, or a tape device, a flash memory or other similar solid state memory device, or an array of devices, including devices in a storage area network or other configurations. In additional implementations, a computer program product is tangibly embodied in an information carrier. The computer program product contains instructions that, when executed, perform one or more methods, such as those described above. The information carrier is a computer- or machine-readable medium, such as the memory **620**, the storage device **630**, or memory on processor **610**.

[0052] The high speed controller **640** manages bandwidth-intensive operations for the computing device **600**, while the low speed controller **660** manages lower bandwidth-intensive operations. Such allocation of duties is exemplary only. In some implementations, the high-speed controller **640** is coupled to the memory **620**, the display **680** (e.g., through a graphics processor or accelerator), and to the high-speed expansion ports **650**, which may accept various expansion cards (not shown). In some implementations, the low-speed controller **660** is coupled to the storage device **630** and a low-speed expansion port **690**. The low-speed expansion port **690**, which may include various communication ports (e.g., USB, Bluetooth, Ethernet, wireless Ethernet), may be coupled to one or more input/output devices, such as a keyboard, a pointing device, a scanner, or a networking device such as a switch or router, e.g., through a network adapter.

[0053] The computing device **600** may be implemented in a number of different forms, as shown in the figure. For example, it may be implemented as a standard server **600***a* or multiple times in a group of such servers **600***a*, as a laptop computer **600***b*, or as part of a rack server system **600***c*.

[0054] Various implementations of the systems and techniques described herein can be realized in digital electronic and/or optical circuitry, integrated circuitry, specially designed ASICs (application specific integrated circuits), computer hardware, firmware, software, and/or combinations thereof. These various implementations can include implementation in one or more computer programs that are executable and/or interpretable on a programmable system including at least one programmable processor, which may be special or general purpose, coupled to receive data and instructions from, and to transmit data and instructions to, a storage system, at least one input device, and at least one output device.

[0055] A software application (i.e., a software resource) may refer to computer software that causes a computing device to perform a task. In some examples, a software application may be referred to as an "application," an "app," or a "program." Example applications include, but are not limited to, system diagnostic applications, system management applications, system maintenance applications, word processing applications, spreadsheet applications, messaging applications, media streaming applications, social networking applications, and gaming applications.

[0056] These computer programs (also known as programs, software, software applications or code) include machine instructions for a programmable processor, and can be implemented in a high-level procedural and/or object-oriented programming language, and/or in assembly/machine language. As used herein, the terms "machine-readable medium" and "computer-readable medium" refer to any computer program product, non-transitory computer readable medium, apparatus and/or device (e.g., magnetic discs, optical disks, memory, Programmable Logic Devices (PLDs)) used to provide machine instructions and/or data to a programmable processor, including a machine-readable medium that receives machine instructions as a machine-readable signal. The term "machine-readable signal" refers to any signal used to provide machine instructions and/or data to a programmable processor.

[0057] The processes and logic flows described in this specification can be performed by one or more programmable processors, also referred to as data processing hardware, executing one or more computer programs to perform functions by operating on input data and generating output. The processes and logic flows can also be performed by special purpose logic circuitry, e.g., an FPGA (field programmable gate array) or an ASIC (application specific integrated circuit). Processors suitable for the execution of a computer program include, by way of example, both general and special purpose microprocessors, and any one or more processors of any kind of digital computer. Generally, a processor will receive instructions and data from a read only memory or a random access memory or both. The essential elements of a computer are a processor for performing instructions and one or more memory devices for storing instructions and data. Generally, a computer will also include, or be operatively coupled to receive data from or transfer data to, or both, one or more mass storage devices for storing data, e.g., magnetic, magneto optical disks, or optical disks. However, a computer need not have such devices. Computer readable media suitable for storing computer program instructions and data include all forms of non-volatile memory, media and memory devices, including by way of example semiconductor memory devices, e.g., EPROM, EEPROM, and flash memory devices; magnetic disks, e.g., internal hard disks or removable disks; magneto optical disks; and CD ROM and DVD-ROM disks. The processor and the memory can be supplemented by, or incorporated in, special purpose logic circuitry.

[0058] To provide for interaction with a user, one or more aspects of the disclosure can be implemented on a computer having a display device, e.g., a CRT (cathode ray tube), LCD (liquid crystal display) monitor, or touch screen for displaying information to the user and optionally a keyboard and a pointing device, e.g., a mouse or a trackball, by which the user can provide input to the computer. Other kinds of devices can be used to provide interaction with a user as well; for example, feedback provided to the user can be any form of sensory feedback, e.g., visual feedback, auditory feedback, or tactile feedback; and input from the user can be received in any form, including acoustic, speech, or tactile input. In addition, a computer can interact with a user by sending documents to and receiving documents from a device that is used by the user; for example, by sending web pages to a web browser on a user's client device in response to requests received from the web browser.

[0059] A number of implementations have been described. Nevertheless, it will be understood that various modifications may be made without departing from the spirit and scope of the disclosure. Accordingly, other implementations are within the scope of the following claims.

What is claimed is:

1. A computer-implemented method executed by data processing hardware that causes the data processing hardware to perform operations comprising:

obtaining a stream of consecutive network configuration snapshots for a network, each network configuration snapshot of the stream of consecutive network configuration snapshots comprising network configuration information;

determining that first network configuration information of a first network configuration snapshot of the network from the stream of consecutive network configuration snapshots for the network is not the same as second network configuration information of a second network configuration snapshot of the network from the stream of consecutive network configuration snapshots for the network, the second network configuration snapshot of the network consecutive with the first network configuration snapshot in the stream of consecutive network configuration snapshots for the network;

determining, based on a difference in packet reachability of the network during the first network configuration snapshot and packet reachability of the network during the second network configuration snapshot, a net change in reachability for the network;

determining, based on the net change in reachability for the network, a policy violation affecting reachability of the network present in the second network configuration snapshot; and

generating a reachability differentiation report including the policy violation.

2. The method of claim **1**, wherein generating the reachability differentiation report comprises translating the policy violation from a computer-interpretable format to a human-interpretable output.

3. The method of claim **1**, wherein each network configuration snapshot of the stream of consecutive network configuration snapshots comprises a data plane model protocol taken at a respective time instance.

4. The method of claim **3**, wherein determining that first network configuration information of the first network configuration snapshot of the network is not the same as second network configuration information of the second network configuration snapshot of the network comprises determining that the data plane model protocol of the first network configuration snapshot of the network is not the same as the data plane model protocol of the second network configuration snapshot of the network.

5. The method of claim **1**, wherein the operations further comprise:

determining that third network configuration information of a third network configuration snapshot of the network from the stream of consecutive network configuration snapshots for the network is the same as fourth network configuration information of a fourth network configuration snapshot of the network from the stream of consecutive network configuration snapshots for the network, the fourth network configuration snapshot of the network consecutive with the third network configuration snapshot in the stream of consecutive network configuration snapshots for the network; and

determining, based on determining that the third network configuration information of the third network configuration snapshot of the network is the same as the fourth network configuration information of the fourth network configuration snapshot of the network, that the network is unchanged between the third network configuration snapshot and the fourth network configuration snapshot.

6. The method of claim **1**, wherein the operations further comprise determining a first one or more packet equivalence classes for the first network configuration snapshot and a second one or more packet equivalence classes for the second network configuration snapshot, each packet equivalence class comprising a set of packets having the same forwarding behavior.

7. The method of claim **6**, wherein the operations further comprise:

assigning the first one or more packet equivalence classes for the first network configuration snapshot to the first network configuration information; and

assigning the second one or more packet equivalence classes for the second network configuration snapshot to the second network configuration information.

8. The method of claim **7**, wherein determining the net change in reachability for the network includes identifying a net change to network equivalence classes between the first network configuration information and the second network configuration information.

9. The method of claim **1**, wherein the operations further comprise localizing a configuration stanza responsible for the policy violation.

10. The method of claim **9**, wherein the reachability differentiation report includes the localized configuration stanza responsible for the policy violation.

11. A system comprising:

data processing hardware; and

memory hardware in communication with the data processing hardware, the memory hardware storing instructions that are executed on the data processing hardware and cause the data processing hardware to perform operations comprising:

obtaining a stream of consecutive network configuration snapshots for a network, each network configuration snapshot of the stream of consecutive network configuration snapshots comprising network configuration information;

determining that first network configuration information of a first network configuration snapshot of the network from the stream of consecutive network configuration snapshots for the network is not the same as second network configuration information of a second network configuration snapshot of the network from the stream of consecutive network configuration snapshots for the network, the second network configuration snapshot of the network consecutive with the first network configuration snapshot in the stream of consecutive network configuration snapshots for the network;

determining, based on a difference in packet reachability of the network during the first network configuration snapshot and packet reachability of the network during the second network configuration snapshot, a net change in reachability for the network;

determining, based on the net change in reachability for the network, a policy violation affecting reachability of the network present in the second network configuration snapshot; and

generating a reachability differentiation report including the policy violation.

12. The system of claim **11**, wherein generating the reachability differentiation report comprises translating the policy violation from a computer-interpretable format to a human-interpretable output.

13. The system of claim **11**, wherein each network configuration snapshot of the stream of consecutive network configuration snapshots comprises a data plane model protocol taken at a respective time instance.

14. The system of claim **13**, wherein determining that first network configuration information of the first network configuration snapshot of the network is not the same as second network configuration information of the second network configuration snapshot of the network comprises determining that the data plane model protocol of the first network configuration snapshot of the network is not the same as the data plane model protocol of the second network configuration snapshot of the network.

15. The system of claim **11**, wherein the operations further comprise:

determining that third network configuration information of a third network configuration snapshot of the network from the stream of consecutive network configuration snapshots for the network is the same as fourth network configuration information of a fourth network configuration snapshot of the network from the stream of consecutive network configuration snapshots for the network, the fourth network configuration snapshot of the network consecutive with the third network configuration snapshot in the stream of consecutive network configuration snapshots for the network; and

determining, based on determining that the third network configuration information of the third network configuration snapshot of the network is the same as the fourth network configuration information of the fourth network configuration snapshot of the network, that the network is unchanged between the third network configuration snapshot and the fourth network configuration snapshot.

16. The system of claim **11**, wherein the operations further comprise determining a first one or more packet equivalence classes for the first network configuration snapshot and a second one or more packet equivalence classes for the second network configuration snapshot, each packet equivalence class comprising a set of packets having the same forwarding behavior.

17. The system of claim **16**, wherein the operations further comprise:

assigning the first one or more packet equivalence classes for the first network configuration snapshot to the first network configuration information; and

assigning the second one or more packet equivalence classes for the second network configuration snapshot to the second network configuration information.

18. The system of claim **17**, wherein determining the net change in reachability for the network includes identifying a net change to network equivalence classes between the first network configuration information and the second network configuration information.

19. The system of claim **11**, wherein the operations further comprise localizing a configuration stanza responsible for the policy violation.

20. The system of claim **19**, wherein the reachability differentiation report includes the localized configuration stanza responsible for the policy violation.

* * * * *