US009270703B1

(12) **United States Patent**
Clough et al.

(10) **Patent No.: US 9,270,703 B1**
(45) **Date of Patent: Feb. 23, 2016**

(54) **ENHANCED CONTROL-PLANE SECURITY FOR NETWORK-ACCESSIBLE SERVICES**

(71) Applicant: **Amazon Technologies, Inc.**, Reno, NV (US)

(72) Inventors: **Duncan Matthew Clough**, Western Cape (ZA); **Andries Petrus Johannes Dippenaar**, Western Cape (ZA); **Marcin Piotr Kowalski**, Western Cape (ZA)

(73) Assignee: **Amazon Technologies, Inc.**, Reno, NV (US)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 151 days.

(21) Appl. No.: **14/060,511**

(22) Filed: **Oct. 22, 2013**

(51) **Int. Cl.**
H04L 9/00 (2006.01)
H04L 29/06 (2006.01)

(52) **U.S. Cl.**
CPC ***H04L 63/20*** (2013.01)

(58) **Field of Classification Search**
CPC H04L 41/0803
See application file for complete search history.

(56) **References Cited**

U.S. PATENT DOCUMENTS

7,724,513	B2	5/2010	Coglitore et al.	
8,122,282	B2	2/2012	Betzler et al.	
8,218,322	B2	7/2012	Clidaras et al.	
8,250,215	B2	8/2012	Stienhans et al.	
8,261,295	B1	9/2012	Risbood et al.	
8,271,536	B2	9/2012	Amradkar et al.	
8,271,653	B2	9/2012	Dehaan	
8,310,829	B2	11/2012	Monk et al.	
2002/0046311	A1 *	4/2002	Kageyama	710/105
2005/0182966	A1 *	8/2005	Pham et al.	713/201
2010/0064033	A1	3/2010	Travostino et al.	
2011/0055399	A1	3/2011	Tung et al.	
2011/0231525	A1	9/2011	Balani et al.	
2012/0072597	A1	3/2012	Teather et al.	
2012/0124211	A1	5/2012	Kampas et al.	
2012/0226789	A1	9/2012	Ganesan et al.	
2012/0239739	A1	9/2012	Manglik et al.	
2012/0303790	A1 *	11/2012	Singh et al.	709/224
2013/0083476	A1	4/2013	Clidaras et al.	
2013/0231038	A1	9/2013	Chang	

OTHER PUBLICATIONS

U.S. Appl. No. 13/747,176, filed Jan 22, 2013, Marcin Piotr Kowalski.
U.S. Appl. No. 13/747,190, filed Jan 22, 2013, Marcin Piotr Kowalski.
U.S. Appl. No. 14/133,533, filed Dec. 18, 2013, Andries Petrus Johannes Dippenaar.

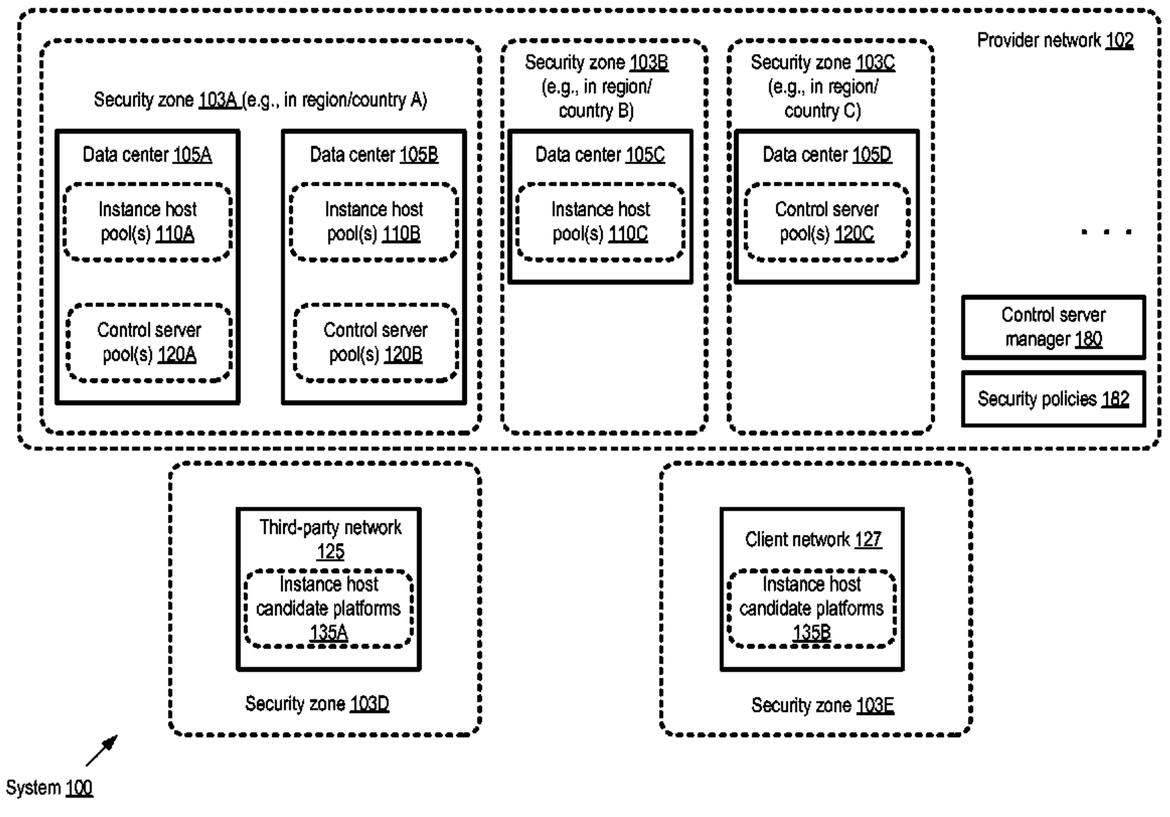
* cited by examiner

Primary Examiner — Brandon Hoffman
Assistant Examiner — Helai Salehi
(74) *Attorney, Agent, or Firm* — Robert C. Kowert; Meyertons, Hood, Kivlin, Kowert & Goetzel, P.C.

(57) **ABSTRACT**

Methods and apparatus for enhancing control-plane security of a network-accessible service are described. In accordance with a security policy, one or more control servers are selected to perform administrative operations associated with configuration of a service instance at a particular instance host of a network-accessible service. The control servers may differ in security properties from the instance host. In response to a configuration request directed at the instance host, administrative operations are implemented at the selected control servers. A low-level command is issued for execution to the instance host from a control server. A result of the low-level command is obtained at the control server and is used to determine a response to the configuration request.

22 Claims, 10 Drawing Sheets



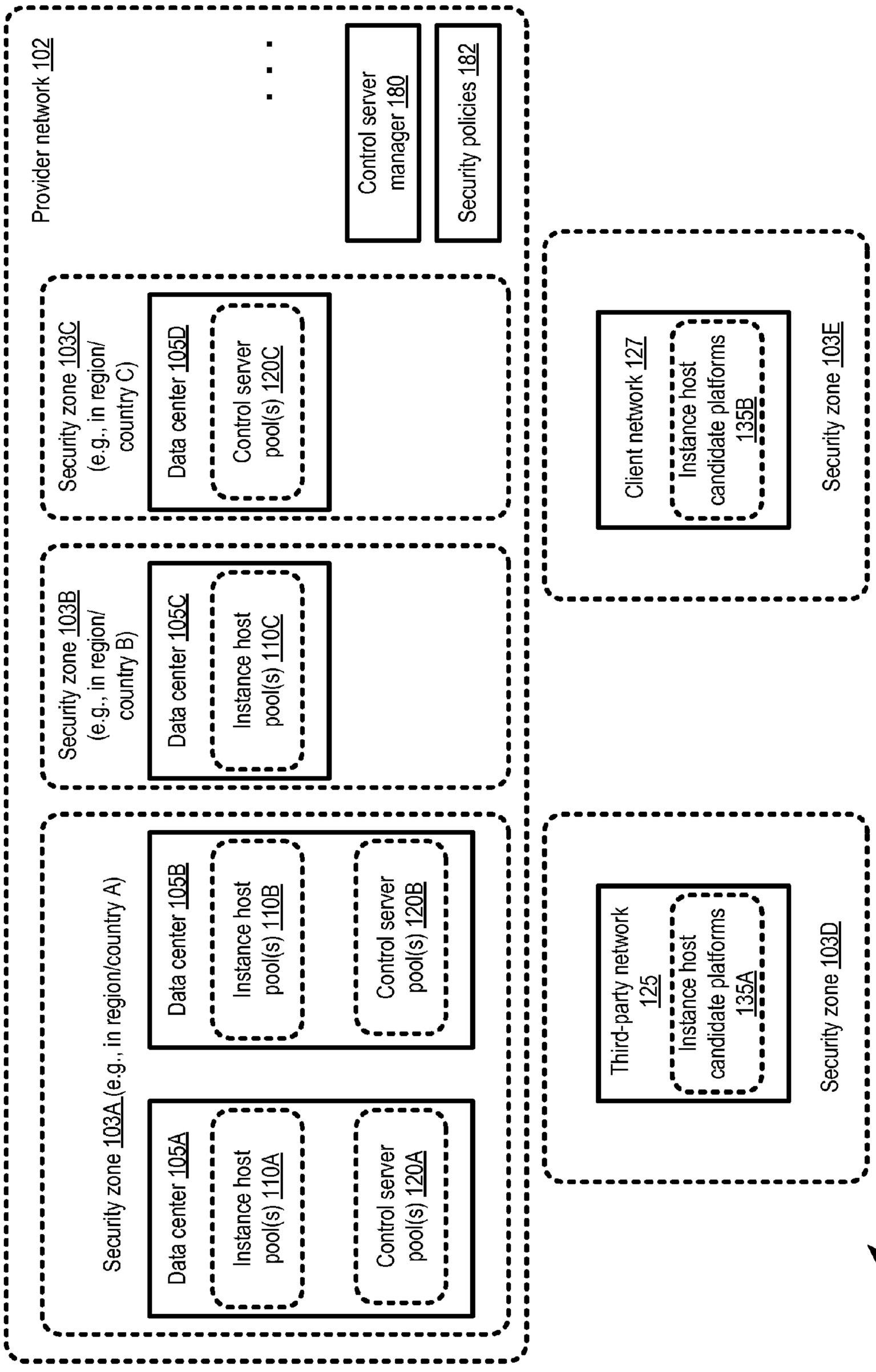

Provider network 102
Control server manager 180
Security policies 182
Security zone 103C (e.g., in region/country C)
Data center 105D
Control server pool(s) 120C
Security zone 103B (e.g., in region/country B)
Data center 105C
Instance host pool(s) 110C
Client network 127
Instance host candidate platforms 135B
Security zone 103E
Security zone 103A (e.g., in region/country A)
Data center 105B
Instance host pool(s) 110B
Control server pool(s) 120B
Data center 105A
Instance host pool(s) 110A
Control server pool(s) 120A
Third-party network 125
Instance host candidate platforms 135A
Security zone 103D
System 100

FIG. 1

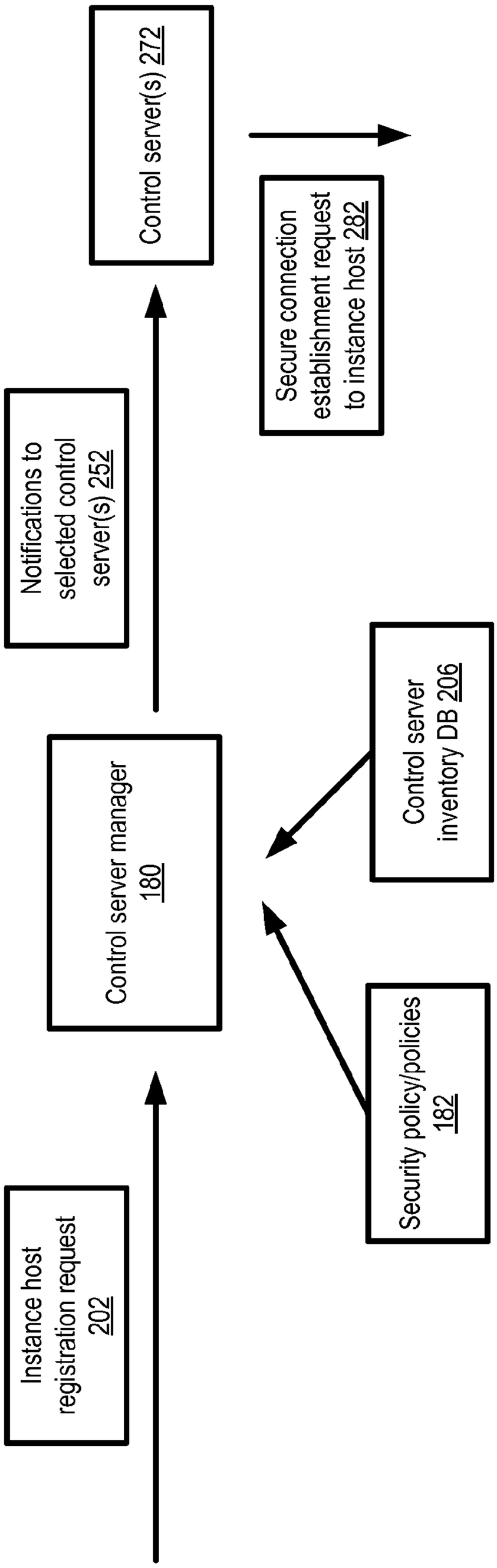

Instance host registration request 202
Notifications to selected control server(s) 252
Control server(s) 272
Secure connection establishment request to instance host 282
Control server manager 180
Security policy/policies 182
Control server inventory DB 206

FIG. 2

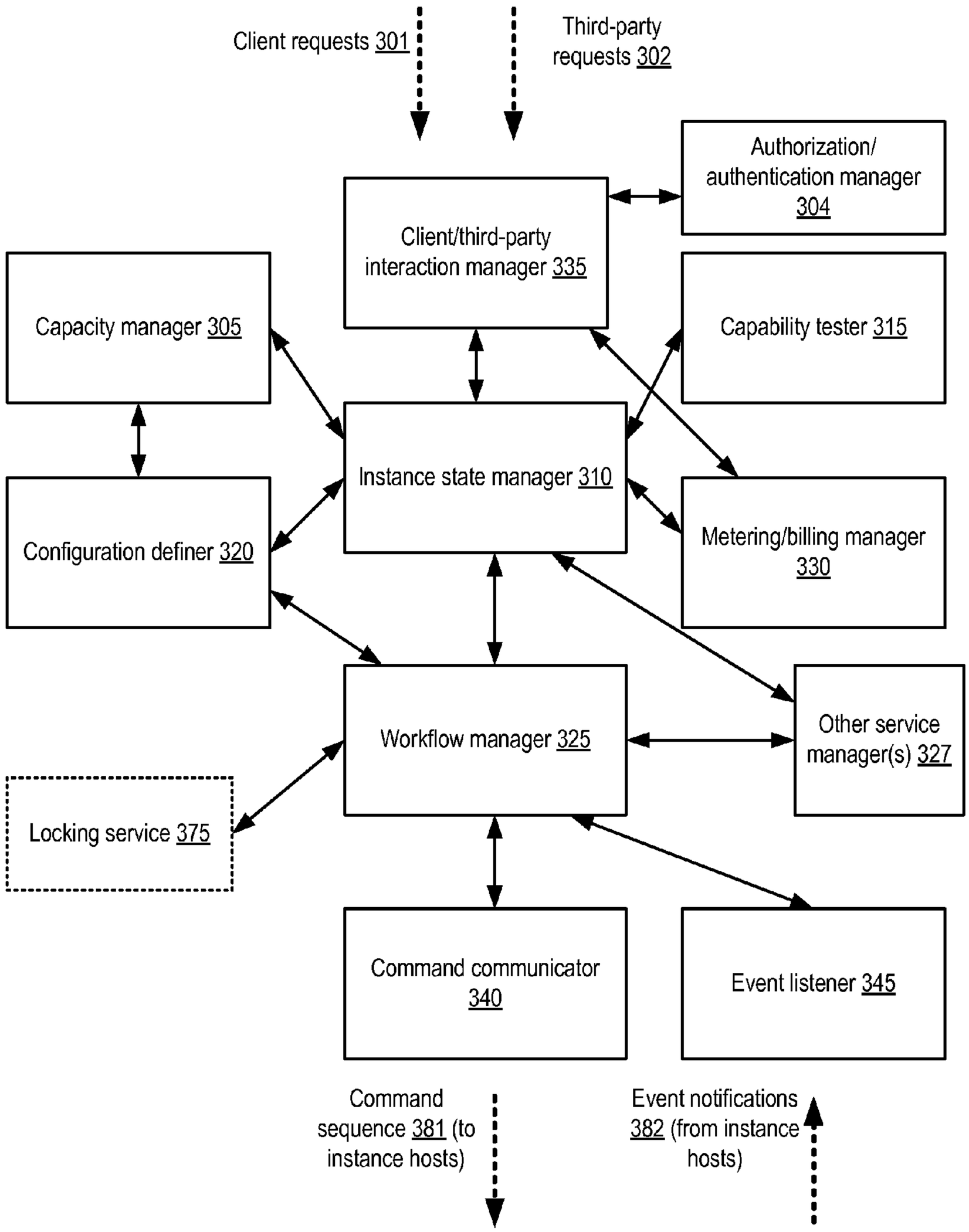

Client requests 301
Third-party requests 302
Client/third-party interaction manager 335
Authorization/ authentication manager 304
Capacity manager 305
Capability tester 315
Instance state manager 310
Configuration definer 320
Metering/billing manager 330
Workflow manager 325
Other service manager(s) 327
Locking service 375
Command communicator 340
Event listener 345
Command sequence 381 (to instance hosts)
Event notifications 382 (from instance hosts)

FIG. 3

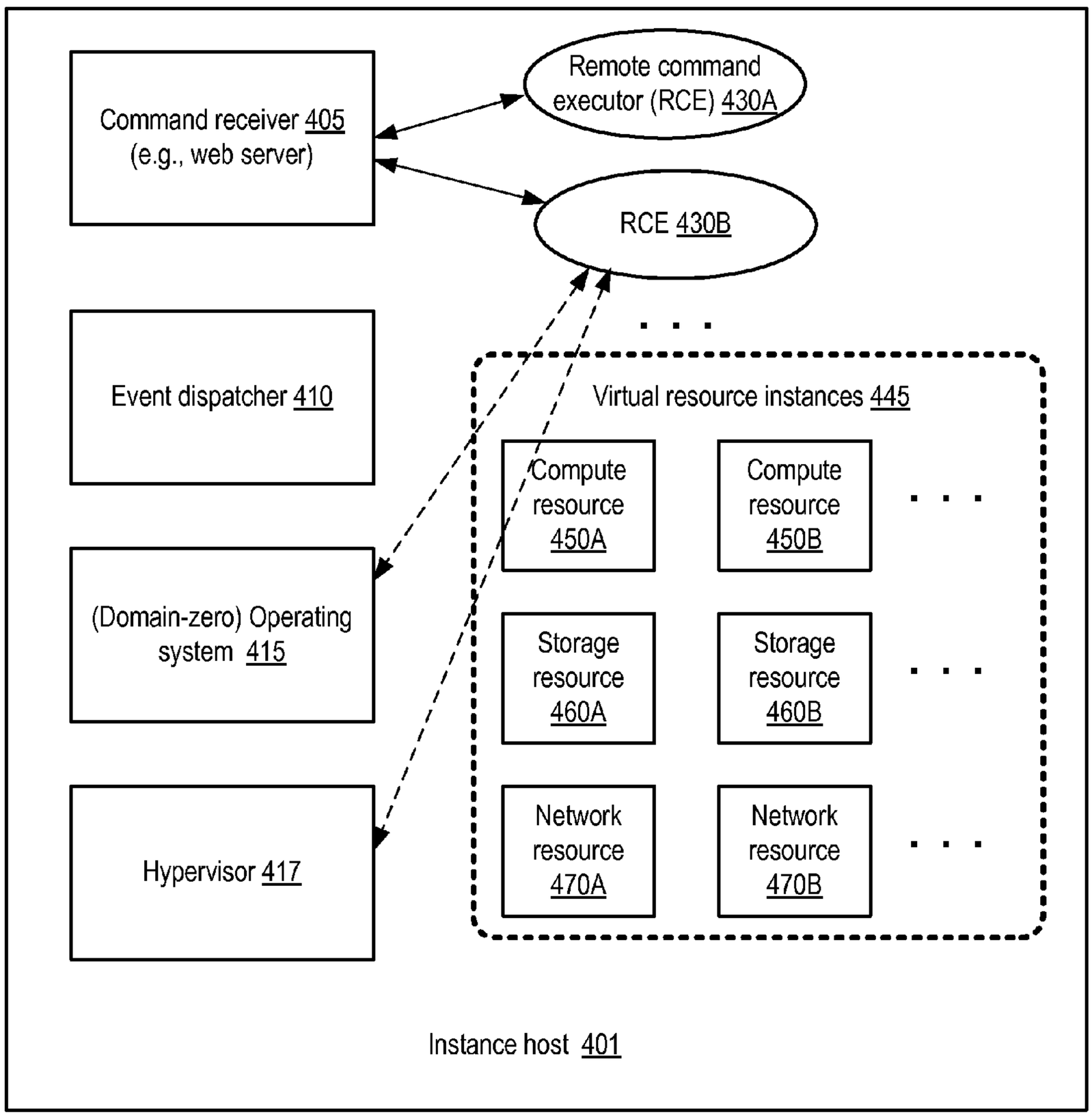

Command receiver 405
(e.g., web server)
Remote command executor (RCE) 430A
RCE 430B
Event dispatcher 410
Virtual resource instances 445
Compute resource 450A
Compute resource 450B
(Domain-zero) Operating system 415
Storage resource 460A
Storage resource 460B
Hypervisor 417
Network resource 470A
Network resource 470B
Instance host 401

FIG. 4

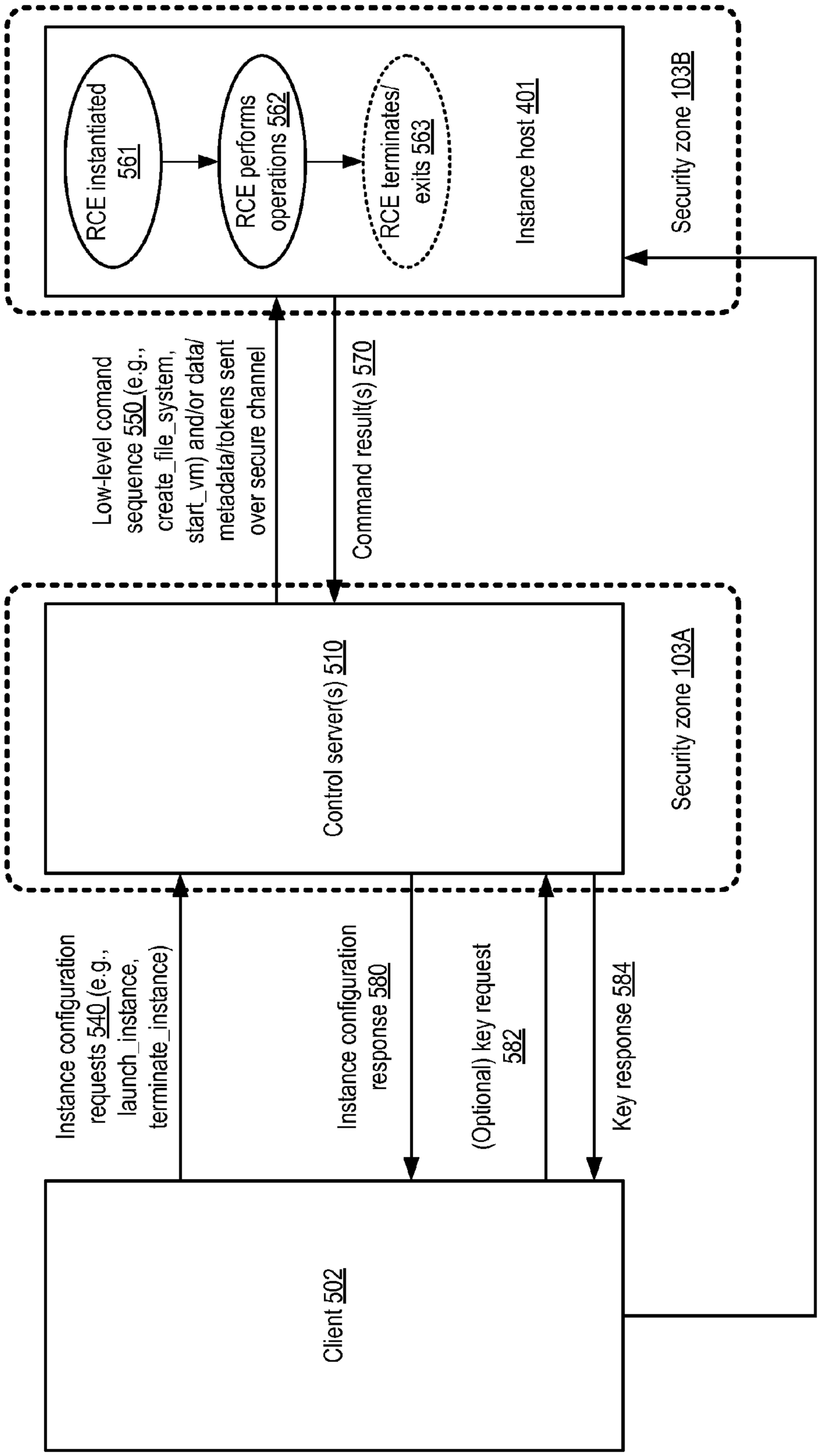

Client 502
Control server(s) 510
Instance host 401
RCE instantiated 561
RCE performs operations 562
RCE terminates/ exits 563
Security zone 103B
Security zone 103A
Instance configuration requests 540 (e.g., launch_instance, terminate_instance)
Low-level comand sequence 550 (e.g., create_file_system, start_vm) and/or data/ metadata/tokens sent over secure channel
Instance configuration response 580
Command result(s) 570
(Optional) key request 582
Key response 584
(Optional) key 588

FIG. 5

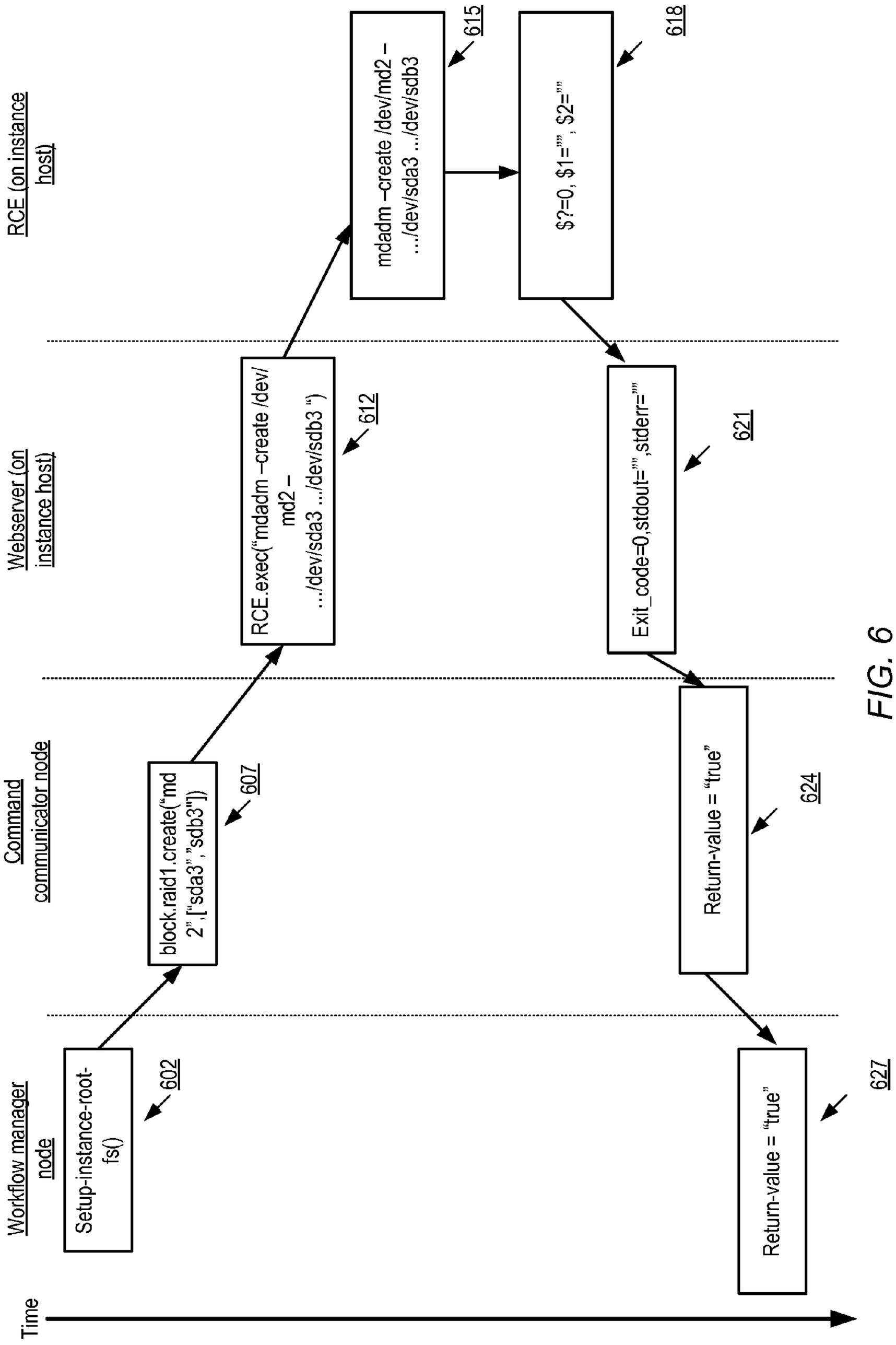

Workflow manager node
Setup-instance-root-fs()
602
Command communicator node
block.raid1.create("md 2",["sda3","sdb3"])
607
Webserver (on instance host)
RCE.exec("mdadm –create /dev/ md2 – .../dev/sda3 .../dev/sdb3 ")
612
RCE (on instance host)
mdadm –create /dev/md2 – .../dev/sda3 .../dev/sdb3
615
$?=0, $1="", $2=""
618
Exit_code=0,stdout="",stderr=""
621
Return-value = "true"
624
Return-value = "true"
627
Time

FIG. 6

720

```
...
Content-MD5: <Body digest/hash>
...
PUT /cgi-bin/cmd/<CHANGE-ID>/<UUID>/<subsystemID>/...
...
```

721

Request header 702

```
...
{
“commands":["cmd1 -f FILE1 FILE2", "cmd2", “cmd3 -f FILE2”],
"@FILE1":"<fileContents1>”,
“@FILE2”:”<fileContents2>”,
...
"uuid":"<UUID>"
}
...
```

Request body 706

```
...
{
"stdout":"Some output on stdout",
"stderr":"Some output on stderr",
"exit-code":"3",
"user-cpu-time":"2000432",
"system-cpu-time":"30432",
"wallclock-time":"2030432",
"command-number":1
}
...
```

Response body 710

FIG. 7

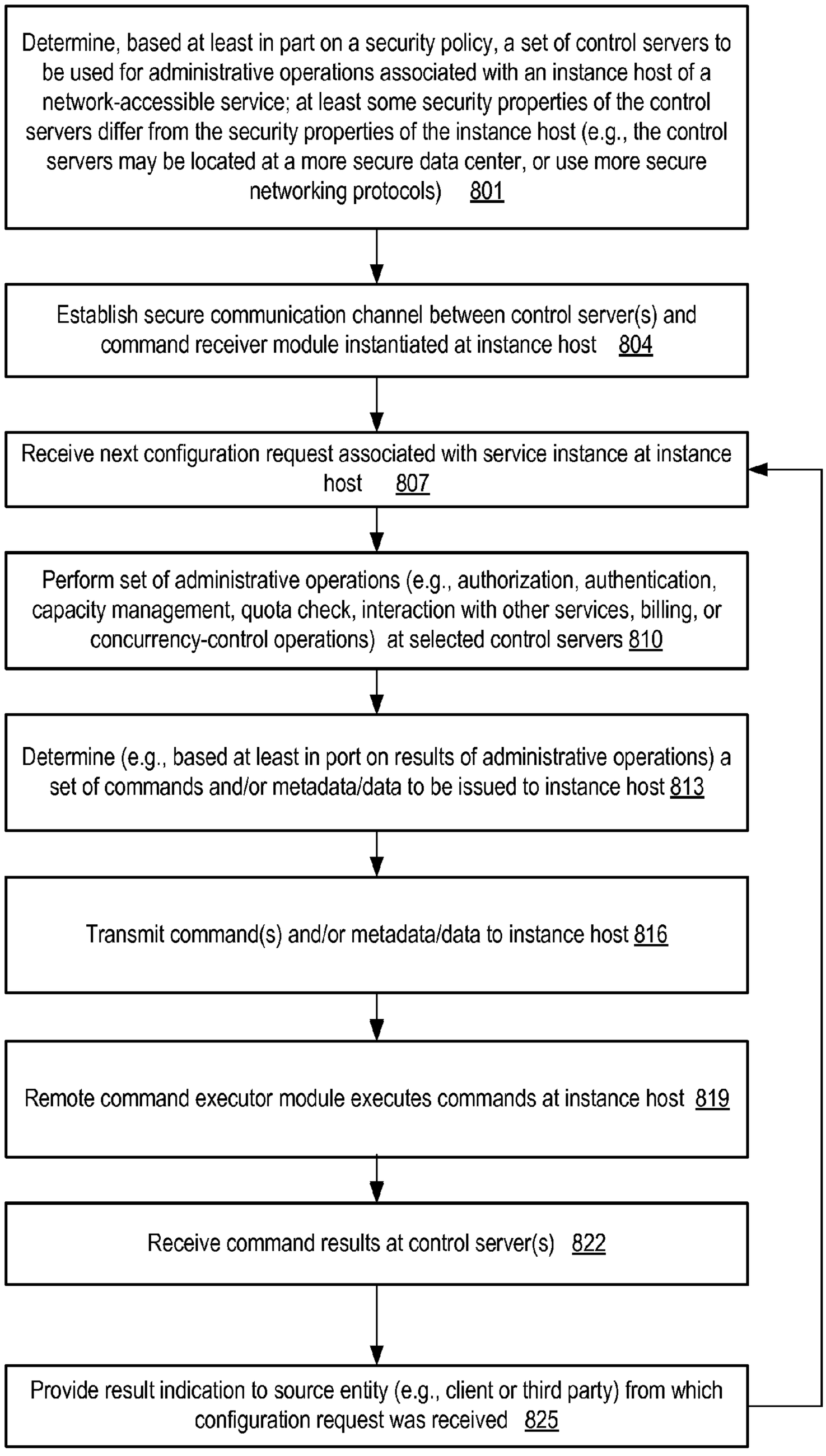

Determine, based at least in part on a security policy, a set of control servers to be used for administrative operations associated with an instance host of a network-accessible service; at least some security properties of the control servers differ from the security properties of the instance host (e.g., the control servers may be located at a more secure data center, or use more secure networking protocols) 801
Establish secure communication channel between control server(s) and command receiver module instantiated at instance host 804
Receive next configuration request associated with service instance at instance host 807
Perform set of administrative operations (e.g., authorization, authentication, capacity management, quota check, interaction with other services, billing, or concurrency-control operations) at selected control servers 810
Determine (e.g., based at least in port on results of administrative operations) a set of commands and/or metadata/data to be issued to instance host 813
Transmit command(s) and/or metadata/data to instance host 816
Remote command executor module executes commands at instance host 819
Receive command results at control server(s) 822
Provide result indication to source entity (e.g., client or third party) from which configuration request was received 825

FIG. 8

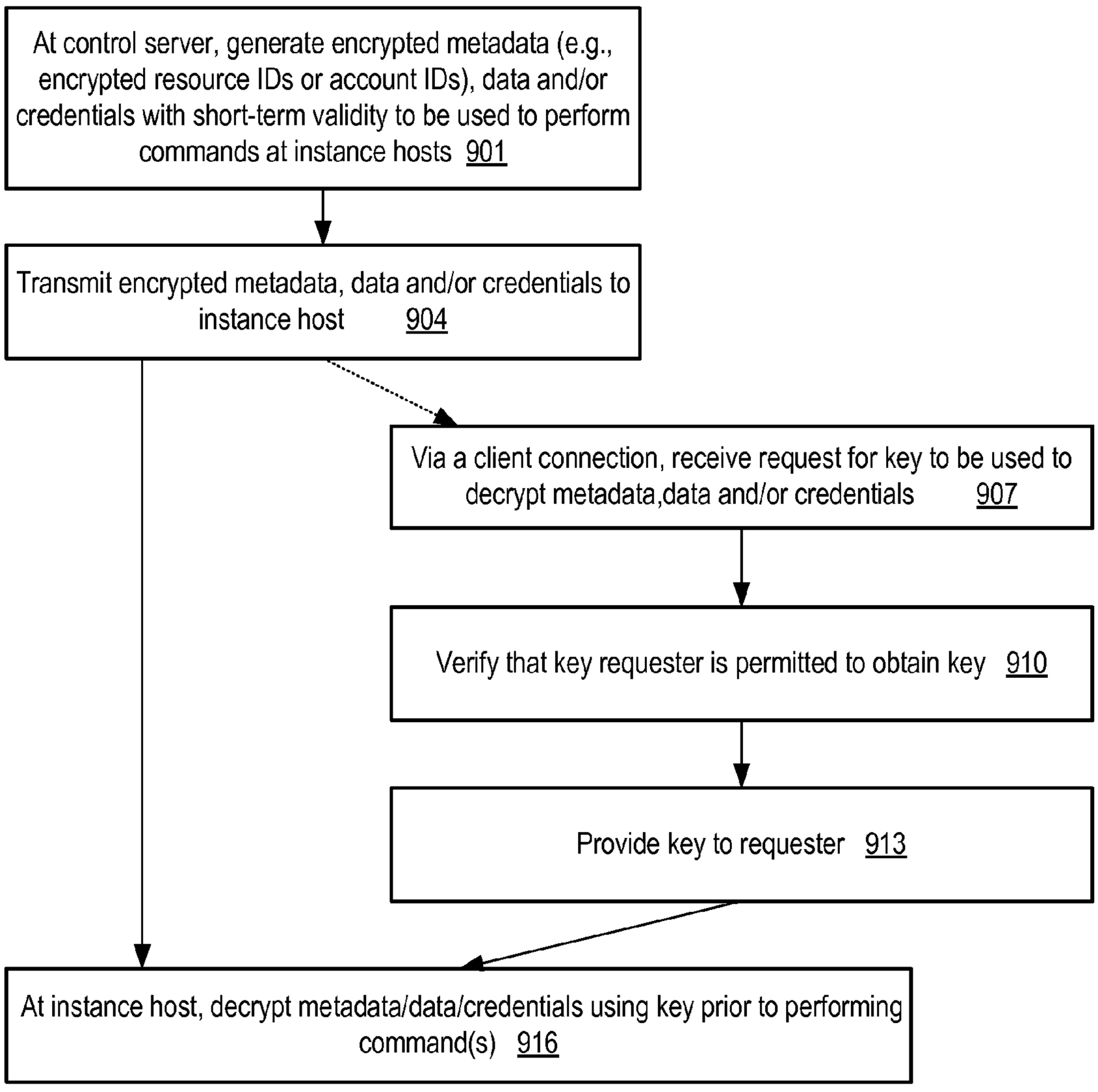

At control server, generate encrypted metadata (e.g., encrypted resource IDs or account IDs), data and/or credentials with short-term validity to be used to perform commands at instance hosts 901
Transmit encrypted metadata, data and/or credentials to instance host 904
Via a client connection, receive request for key to be used to decrypt metadata,data and/or credentials 907
Verify that key requester is permitted to obtain key 910
Provide key to requester 913
At instance host, decrypt metadata/data/credentials using key prior to performing command(s) 916

FIG. 9

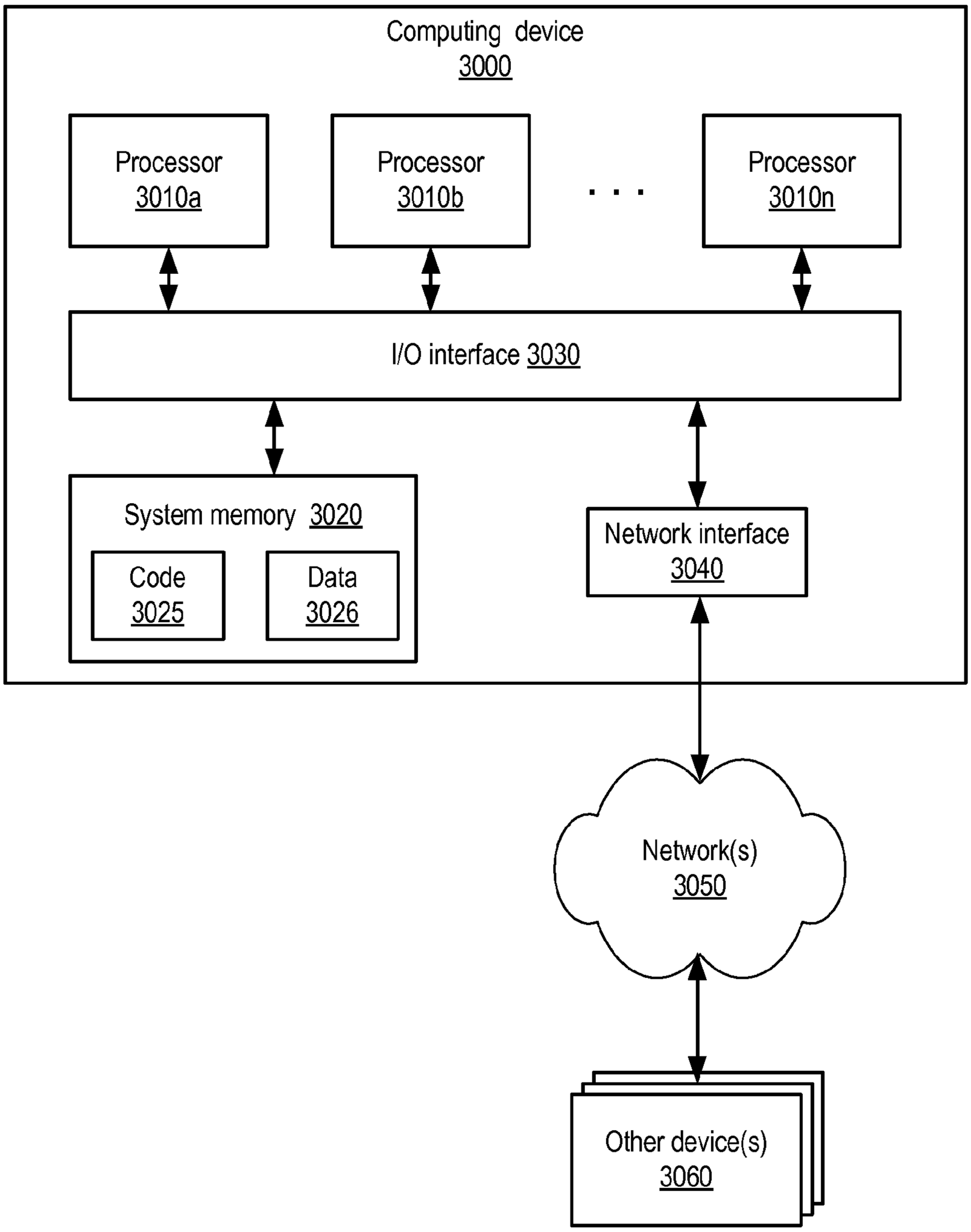

Computing device
3000
Processor
3010a
Processor
3010b
Processor
3010n
I/O interface 3030
System memory 3020
Code
3025
Data
3026
Network interface
3040
Network(s)
3050
Other device(s)
3060

FIG. 10

ENHANCED CONTROL-PLANE SECURITY FOR NETWORK-ACCESSIBLE SERVICES

BACKGROUND

Many companies and other organizations operate computer networks that interconnect numerous computing systems to support their operations, such as with the computing systems being co-located (e.g., as part of a local network) or instead located in multiple distinct geographical locations (e.g., connected via one or more private or public intermediate networks). For example, data centers housing significant numbers of interconnected computing systems have become commonplace, such as private data centers that are operated by and on behalf of a single organization, and public data centers that are operated by entities as businesses to provide computing resources to customers. Some public data center operators provide network access, power, and secure installation facilities for hardware owned by various customers, while other public data center operators provide "full service" facilities that also include hardware resources made available for use by their customers. However, as the scale and scope of typical data centers has increased, the tasks of provisioning, administering, and managing the physical computing resources have become increasingly complicated.

The advent of virtualization technologies for commodity hardware has provided benefits with respect to managing large-scale computing resources for many customers with diverse needs, allowing various computing resources to be efficiently and securely shared by multiple customers. For example, virtualization technologies may allow a single physical computing machine to be shared among multiple users by providing each user with one or more virtual machines hosted by the single physical computing machine, with each such virtual machine being a software simulation acting as a distinct logical computing system that provides users with the illusion that they are the sole operators and administrators of a given hardware computing resource, while also providing application isolation and security among the various virtual machines. Furthermore, some virtualization technologies are capable of providing virtual resources that span two or more physical resources, such as a single virtual machine with multiple virtual processors that spans multiple distinct physical computing systems.

As the functionality and features supported by providers of virtualized compute, storage and networking resources grows, and as the fleet of hardware platforms that are used by large-scale providers grows, the implementation of administrative control operations such as configuration changes on the platforms can itself become fairly complex. Accordingly, the providers may implement sophisticated algorithms and/or workflows to manage various types of control operations. Such workflows or algorithms may include business logic that provides competitive advantages to the providers, and thus may need to be protected from attackers that could potentially access and reverse-engineer the logic. As the provider networks grow to include a variety of data centers with different levels of physical and/or network security, the vulnerability of business logic implemented at the data center hosts is only likely to increase further.

BRIEF DESCRIPTION OF DRAWINGS

FIG. **1** illustrates an example system environment, according to at least some embodiments.

FIG. **2** illustrates examples of operations that may be performed to select control servers for instance hosts, according to at least some embodiments.

FIG. **3** illustrates example components of control servers configured for remote configuration of instance hosts, according to at least some embodiments.

FIG. **4** illustrates example components of instance hosts, according to at least some embodiments.

FIG. **5** illustrates example interactions between clients, control servers, and instance hosts, according to at least some embodiments.

FIG. **6** illustrates an example of command flow starting from a workflow manager node at a control server, according to at least some embodiments.

FIG. **7** illustrates example elements of command requests issued to an instance host from a control server, according to at least some embodiments.

FIG. **8** is a flow diagram illustrating aspects of operations that may be performed to enhance control-plane security for network-accessible services by selecting and using appropriate control servers for an instance host, according to at least some embodiments.

FIG. **9** is a flow diagram illustrating aspects of operations associated with the encryption and decryption of data and/or metadata that may be used for configuration operations at instance hosts, according to at least some embodiments.

FIG. **10** is a block diagram illustrating an example computing device that may be used in at least some embodiments.

While embodiments are described herein by way of example for several embodiments and illustrative drawings, those skilled in the art will recognize that embodiments are not limited to the embodiments or drawings described. It should be understood, that the drawings and detailed description thereto are not intended to limit embodiments to the particular form disclosed, but on the contrary, the intention is to cover all modifications, equivalents and alternatives falling within the spirit and scope as defined by the appended claims. The headings used herein are for organizational purposes only and are not meant to be used to limit the scope of the description or the claims. As used throughout this application, the word "may" is used in a permissive sense (i.e., meaning having the potential to), rather than the mandatory sense (i.e., meaning must). Similarly, the words "include," "including," and "includes" mean including, but not limited to.

DETAILED DESCRIPTION

Various embodiments of methods and apparatus for enhancing control-plane security for network-accessible services of a provider network are described. Networks set up by an entity such as a company or a public sector organization to provide one or more services (such as various types of multi-tenant and/or single-tenant cloud-based computing or storage services) accessible via the Internet and/or other networks to a distributed set of clients may be termed provider networks in this document. The term "multi-tenant" may be used herein to refer to a service that is designed to implement application and/or data virtualization in such a manner that different client entities are provided respective customizable, isolated views of the service, such that one client to whom portions of the service functionality are being provided using a given set of underlying resources may not be aware that the set of resources is also being used for other clients. A provider network may support single-tenant services (such as for private cloud implementations) in some embodiments, either in addition to, or instead of, multi-tenant services. A given provider network may include numerous data centers hosting

various resource pools, such as collections of physical and/or virtualized computer servers, storage devices, networking equipment and the like, needed to implement, configure and distribute the infrastructure and services offered by the provider. Within large provider networks, some data centers may be located in different cities, states or countries than others, potentially with different levels of physical security, network security or even legal protections (e.g., different sets of laws may apply to different data centers, and the data centers may be governed by different legal jurisdictions).

The administrative or control-plane architecture for at least some of the services of a provider network may accordingly be implemented in a modular manner in some embodiments, so that at least some aspects of the logic involved in configuring various client-accessible service resources can be executed at locations and/or devices that are less vulnerable to attacks than the client-accessible service resources themselves. The term "control-plane" may be used herein to distinguish administrative or configuration-related operations from "data-plane" operations that involve manipulation of client application data. Control plane operations may include, among other types of operations, resource lifecycle management operations (such as state changes resulting from client requests), anti-entropy processes, and the like. In one example scenario, the client-accessible service resources may include compute instances of a virtualized computing service implemented at a set of instance hosts located in a particular data center DC1 of the provider network. In accordance with a security policy established for the virtualized computing service in some embodiments, a determination may be made that at least some control-plane decisions for the set of instance hosts should be made at a more secure location than DC1. Any of a variety of different types of control-plane operations may be designated as requiring higher security in different embodiments, including for example authorization/authentication operations associated with configuration requests, capacity management operations, quota checks, interactions with other network-accessible services of the provider network (e.g., the acquisition of storage volumes from a storage service), billing account related operations associated with client requests, or concurrency control operations to manage concurrent updates to internal data structures of the service.

The modular control-plane architecture may allow the service to select one or more control servers located in a different data center DC2, whose security characteristics are deemed more suitable than the devices available at DC1, to implement the higher-security control-plane operations for the instance hosts at DC1 in such embodiments. Such an approach may also be referred to herein as "remote configuration" of instance hosts. Secure network channels or connections may be established between the selected secure control servers in DC2 and the instance hosts in DC1 in some embodiments for the transmission of low-level control commands to the instance hosts, and for receiving the results of such commands at the control servers. When configuration requests associated with the instance hosts (such as requests to launch/stop instances, attach/detach storage devices, and the like) at DC1 are received from clients of the network-accessible service, at least some of the corresponding control-plane operations may be performed at the more-secure control servers at DC2. Only relatively simple low-level commands and/or associated metadata or data, determined at least in part based on the control-plane operations, may be transmitted to the instance hosts for local execution at DC1 in at least some embodiments. Results of the low-level commands may be transmitted back to the control servers at DC2 in such embodiments, and corresponding responses to the client configuration requests may be provided from the control servers. As a result of separating the execution location of the control-plane logic from the hosts at which the client-accessible resources are located, an extra layer of protection for the provider network's business logic assets may be provided. Even in the unlikely event than an intruder that is able to penetrate the security of the instance hosts at DC1, such an intruder may still be unable to access, examine, or reverse-engineer much or all of the control-plane logic used for configuring the instance hosts. In some embodiments and for certain types of configuration operations, additional security-enhancing techniques such as the use of encrypted credentials and/or other metadata/data for the local operations at the instance hosts may also be used, as described below in further detail.

A number of different types of network-accessible services may implement a modular control-plane architecture in various embodiments, such as the aforementioned virtual computing service, various storage-related services, database services, specialized parallel computing services, scientific computing services, and the like. A subset of the resources of a given service of the provider network may in some embodiments be offered for reservation by (and allocation to) clients in units called "instances," such as virtual or physical compute instances, storage instances, or network resource instances. The term "service instances" may also be used to refer to these types of service units herein. A virtual compute instance may, for example, comprise one or more servers with a specified computational capacity (which may be specified by indicating the type and number of CPUs, the main memory size, storage device number and size, and so on) and a specified software stack (e.g., a particular version of an operating system, which may in turn run on top of a hypervisor). Resource instances of various kinds, including virtual compute instances, storage resource instances or network resource instances, may be instantiated on systems termed "instance host platforms" or "instance hosts" herein. In some embodiments, an instance host capable of instantiating N different virtual compute instances of a particular type may, for example, comprise a hardware server with a selected set of relatively low-level software components initially installed, such as virtualization software and/or operating system software typically utilizing a small fraction of the hardware server's compute capabilities. As more virtual compute instances are launched, a larger portion of the server's compute capabilities may get used, e.g., for client applications running on the different virtual compute instances. A number of different types of computing devices may be used singly or in combination to implement the resources of the provider network in different embodiments, including general purpose or special purpose computer servers, storage devices, network devices and the like. As described below, a subset of the provider network resources may be dedicated for administrative control and configuration purposes (e.g., for launching, monitoring and terminating resource instances on instance hosts in response to client requests) in some embodiments. Such dedicated control resources may be termed "control plane resources", "control plane servers", or "control servers" herein. In at least some embodiments, in addition to being used to configure resource instances on instance hosts within the provider network, at least some control servers of a given provider network may also be able to remotely configure instances hosted at platforms external to the provider network, e.g., in third party data centers or facilities, or at point-of-presence locations or similar facilities, as described below in further detail.

The modular approach to control-plane architecture may be beneficial not just in view of the security considerations indicated above in at least some embodiments, but may also have significant performance benefits. In such embodiments, control software for managing instances may generally be implemented so as to minimize the administrative overhead imposed on the instance hosts. Much of the configuration-related processing may be offloaded from the instance hosts in such an embodiment, so that high-level decisions and metadata manipulation may be implemented at the control servers, while only simple low-level (and typically idempotent and stateless) configuration-related commands may have to be executed at the instance hosts themselves. Details about instance states and instance type definitions may not be required to be understood at the instance hosts in such embodiments. For example, in one such embodiment, a layered control software architecture may be employed at the control servers, in which an instance state manager responds to a client's instance configuration request by invoking a workflow manager component. In some implementations, components of the control-plane may be configured to perform authentication and/or authorization checks associated with client requests, e.g., by communicating with an identity management service implemented in the provider network. Other components may be involved in communicating with other network-accessible services, such as storage services or networking-related services whose resources may be needed to implement the desired configuration operations (e.g., attaching a storage volume, or activating a network interface) at the instance hosts. The workflow manager may translate a higher-level configuration decision (reached by the instance state manager in response to the client's instance configuration request), in the context of an instance configuration definition provided by a configuration definer component of the control software, into one or more lower-level workflow operations specific to that configuration definition. The workflow manager may in turn transmit the workflow operations to a command communicator component of the control software at the control server. The command communicator may securely submit one or more low-level commands (such as operating system commands or virtualization software commands), corresponding to a given workflow operation, to a particular instance host over a network, in accordance with a command protocol. In some implementations and/or for some types of commands, associated data, metadata and/or credentials (e.g., in the form of tokens for which a short-term validity period is determined at the control servers) may also be transmitted to the instance host.

At the instance host, a command receiver (such as a simple web server) may respond to a given command from the communicator by instantiating a remote command executor (RCE) in some embodiments. An RCE, which may comprise a single thread of execution (or a software process) spawned by the command receiver on demand, may at least in some embodiments only remain active long enough to issue one or more operations, typically directed to a virtualization software component, an operating system component, monitoring software or workflow software at the instance host. The RCE may exit or terminate after the operations have been initiated in such embodiments. The command receiver may provide, to the command communicator, return codes, standard output or error output generated by the RCE's operations. In some implementations, one or more metrics associated with the commands executed by the RCE may also be supplied to the command receiver, such as user/system/kernel runtime, resources used for the commands, or a list of the commands. The supplied results and/or additional information may be interpreted at the control server to determine the success or failure of the requested commands, and a response to the client's instance configuration request may be formulated accordingly in some embodiments. Thus, the instance configuration overhead at the instance hosts may be limited largely to the instantiation of the RCEs and the operations requested by the RCEs in such embodiments, thereby reducing the likelihood of attackers being able to access the control-plane algorithms or code, and also retaining the vast majority of the instance host resources for the use of the client-requested resource instances themselves. In some implementations, the encapsulation of configuration responsibilities at different layers of control server software may be efficient enough to allow hundreds or thousands of instance hosts to be remotely configured from a single control server or a few control servers. Such encapsulation may further enhance control-plane security, as only a few control servers in secure locations may be required to manage large numbers of instance hosts, thus reducing the number of servers that can be targeted for attack.

In at least some embodiments, instantiating an RCE may comprise instantiating at least one thread of execution in accordance with the Common Gateway Interface (CGI), e.g., by a web server. An efficient and well-known protocol such as HTTPS (a secure version of HTTP, the HyperText Transfer Protocol) may be used for command transmissions to instance hosts, and/or to receive results from instance hosts in some implementations. The commands themselves may be formatted in an industry-standard format or notation such as some variant of JSON (JavaScript Object Notation) or XML (Extended Markup Language) in some embodiments. In other embodiments, private or proprietary protocols and/or formats may be used. The command protocol used may support a plurality of command types, of which at least a subset are designed to be idempotent—e.g., if a particular idempotent command "cmd1" with a given set of parameters is issued more than once, the net effect of the multiple "cmd1" issuances is the same as the effect of a single issuance of "cmd1", and the second issuance and any later issuances of the command have no negative effects.

In some embodiments the provider network may be organized into a plurality of geographical regions, and each region may include one or more availability containers, which may also be termed "availability zones" herein. An availability container in turn may comprise one or more distinct locations or data centers, engineered in such a way that the resources in a given availability container are insulated from failures in other availability containers. That is, a failure in one availability container may not be expected to result in a failure in any other availability container; thus, the availability profile of a resource instance or control server is intended to be independent of the availability profile of resource instances or control servers in a different availability container. Clients may be able to protect their applications from failures at a single location by launching multiple application instances in respective availability containers. At the same time, in some implementations, inexpensive and low latency network connectivity may be provided between resource instances that reside within the same geographical region (and network transmissions between resources of the same availability container may be even faster). Some clients may wish to specify the locations at which their resources are reserved and/or instantiated, e.g., at either the region level, the availability container level, or a data center level, to maintain a desired degree of control of exactly where various components of their applications are run. Other clients may be less interested in the exact location where their resources are reserved or

instantiated, as long as the resources meet the client requirements, e.g., for performance, high availability, supported software levels, and so on. Control servers located in one availability container (or data center) may be able to remotely configure resource instances at instance hosts in other availability containers (or other data centers) in some embodiments—that is, a particular availability container or data center may not need to have local control servers to manage the local resource instances. In some embodiments, at least some availability containers may have different security-related properties than others, so that different availability containers may also represent respective security zones or security domains. The terms "security zone" or "security domain", as used herein, may refer to a group of resources that have similar properties with respect to physical and/or network security. In other embodiments, the security characteristics of different data centers within a given availability container may differ, or the security characteristics of different parts of a given data center may differ, so that security zone boundaries may differ from the boundaries of data centers or availability containers.

One of the design goals for the modular control software architecture may be to ensure that recovery from certain types of large scale failure events can be accomplished within an acceptable timeframe. For example, even though data centers and availability zones may be implemented with various levels of redundancy at critical components to reduce data-center-wide or availability-zone-wide failures, it may be very hard to prevent such large scale failures with a 100% guarantee. Since many of the clients of the provider network may rely upon its resource instances for mission-critical functions, a reasonably quick recovery from such rare failure events may be desired. Accordingly, in at least some embodiments, the resources dedicated to control servers may be determined based on (among other factors such as security policies) target recovery times for large scale failures. A rate at which instance recovery configuration operations may be required in the event of a large-scale failure may be estimated. A parameterized model may be generated that includes, for example, representations of the sizes of the failures to be managed (e.g., the number of simultaneous or near-simultaneous failures for which contingency plans are to be drawn up) as well as the potential mapping of those instances to different data centers, the sequences of recovery related configuration operations that would need to be performed to fully re-instantiate the instances, and the number of such operations that a recovery server with a certain level of computing and network capability may be able to orchestrate per unit time. Security policies may also be factored into the model in at least some embodiments. Using various parameters of the model, including the security considerations and the required recovery operations rate to meet a recovery time target, the number of control servers of a particular capability level may be determined, and a pool of control servers of the appropriate type may be established at locations selected in accordance with the security policies in effect.

In at least some embodiments, several or all of the components of the control servers, such as the workflow manager and the command communicator, may be implemented as nodes of a cluster whose size can be increased dynamically as needed. For example, there may be W workflow manager nodes and C command communicator nodes instantiated at a given point in time, and the number of nodes for each component may be increased or decreased as desired. A given hardware device may be used for one or more nodes of a given type of control server component in some implementations—e.g., it may be possible to allocate S control servers to host W workflow manager nodes and C command communicator nodes, where S<=(W+C).

As noted above, a given instance host platform may be capable of supporting multiple resource instances in some embodiments. Flexible mappings between the resource instances on a given instance host and the control servers that manage them may be implemented in some such embodiments—e.g., one resource instance RI-X on a host H1 may be managed by a control server CS1, while another resource instance RI-Y on H1 may be managed by a different control server CS2, as long as compliance with any associated security policies is maintained. In at least some embodiments, a concurrency control mechanism may be implemented to prevent conflicting operations (e.g., two different commands to create a software storage device such as a file system with the same name or with conflicting names) from being attempted. For example, the number of concurrent configuration operations on a given instance host platform may be limited using locks in one implementation. A lock manager may be implemented in some embodiments, from which an exclusive lock (or a shared lock with restrictions on the number of sharers and/or the types of instance host operations allowed while holding the shared lock) has to be obtained prior to performing configuration operations on a given instance host. Concurrency control operations and interactions may also typically be restricted to secure control servers in at least some embodiments.

In some embodiments, the provider network's control software architecture may support the instantiation of resource instances using equipment at locations outside the provider network, e.g., at data centers or other facilities owned/managed by third parties or by clients of the provider network, or at access points between the provider network and other networks. For example, a third party provider (or even a client of the network-accessible service) may wish to capitalize on underused hardware at a data center by deploying the hardware for resource instances that are to be managed using control servers of the provider network. In another example, hosts at one or more Internet point-of-presence (POP) locations associated with the provider network may be utilized for remote instances using control servers in some embodiments. In some such POP locations, at least some of the hosts may be configured to support a service (such as content distribution) of the provider network, and such hosts may in some cases use a stripped-down version of the software stack typically installed on most of the instance hosts used for instantiating resource instances within the provider network. Such stripped-down hosts may be used to instantiate resource instances by control servers.

A given control server may be able to manage third party platforms, as well as, or instead of, the provider network's own instance hosts in some embodiments. The provider network operator may be willing to support such scenarios as it may increase the overall pool of resources that are accessible by clients, and also may lead to a better geographical distribution, enhanced system-wide risk management, and increases in revenue. In one such embodiment, a third party vendor (or a client, or a POP location operator) may submit a platform approval request (e.g., via a programmatic interface supported by a control server component) indicating candidate platforms located at remote facilities, that can be used for hosting virtualized resources in a manner similar to the way the provider network's own instance hosts are used. In response, a control server component responsible for verifying platform capabilities may perform one or more tests on the candidate platforms. Such tests, which may be termed

"capability determination operations" herein, may include a variety of different components, including installed software stack checks, performance tests, security-related checks, checks to verify that the remote command executor (RCE) mechanism can be used successfully on the third party platform, and so on. If a particular candidate platform passes the tests, it may be designated as an "approved" platform on which resource instances can be configured by the provider network's control servers. (Similar capability testing may be performed on the provider network's own hardware platforms in some embodiments, prior to their use for instances.)

Example System Environment

FIG. **1** illustrates an example system environment, according to at least some embodiments. As shown, system **100** comprises a provider network **102**, including a plurality of data centers **105** such as data centers **105**A, **105**B, **105**C and **105**D whose resources are used to implement one or more network-accessible services such as a virtual computing service or a storage service. Based on differences in their security-related characteristics, the data centers **105** may be logically grouped into a plurality of security zones **103**, such as security zone **103**A, **103**B and **103**C, with the security characteristics of one security zone differing in at least some properties from those of other security zones. Some data centers may have different physical security rules or protocols, for example, governing who is allowed physical entry into various parts of the data centers. Data centers may also differ with respect to network security—e.g., from the provider network operator's perspective, some network pathways in and out of a given data center may be owned and managed by the operator, and may thus be deemed more secure than pathways in and out of a different data center **105** over which the operator has less control. In some embodiments, the security levels at different data centers may correlate at least to some extent with geographical location—e.g., data centers located in one country or region may be more secure than those located in another country or region. The legal environment, such as the applicable laws and/or the efficiency of the legal system in implementing the applicable laws, may also differ from one data center to another in some embodiments and may also play a role in determining where sensitive operations should preferably be conducted. It is noted that although each data center **105** is shown as belonging to a single security zone **103** in FIG. **1**, in at least some embodiments multiple security zones may exist within a single data center.

In the embodiment shown in FIG. **1**, the resources of the provider network may comprise two broad categories—control plane resources used primarily for administrative operations of one or more network-accessible services of the provider network, and instance hosts used primarily to implement client-accessible and client-modifiable resources of the network-accessible services. Control-plane resources may typically not be modifiable by clients in some embodiments. In at least some embodiments, pools of control-plane servers (such as control server pools **120**A, **120**B and **120**C) and/or pools of instance hosts (such as instance host pools **110**A, **110**B and **110**C) may be established at various data centers **105**. Some data centers, such as **105**A and **105**B, may include both control server pools **120** and instance host pools **110**; others may comprise only instance host pools or only control server pools. For example, data center **105**C includes instance host pool **110**C but does not include control servers, while data center **105**D include control server pool **120**C but does not include instance hosts.

Based on security policies **182** and/or on other considerations such as performance goals or load-balancing requirements, one or more control servers may be selected from pools **120** to implement at least some control plane operations associated with configuration of client-accessible service instances (such as compute instances of a virtual computing service) at a given instance host in the embodiment shown in FIG. **1**. The selection of the specific control servers to be used for a given instance host (based for example on differences in security properties of the instance host and the control servers) may be performed by a control server manager **180** in some embodiments. Such a control server manager **180** may itself comprise one or more hardware and/or software components. Although the control server manager **180** is depicted as a single entity in FIG. **1**, in some embodiments the functionality of the control server manager **180** may be performed by a collection of geographically or logically distributed hardware and/or software. In one embodiment in which several different network-accessible services of provider network **102** implement a similar modular or layered control-plane architecture for security and/or performance reasons, each such service may have its own control server manager **180**. In other embodiments, a single control server manager **180** may be configured to implement security policies **182** for several different network-accessible services.

After a control server has been selected by the control server manager **180** for implementing some set of administrative operations associated with an instance host, a secure network connection may be established between the control server and the instance host in the depicted embodiment. Subsequently, when a client of the network-accessible service (e.g., a client on whose behalf a compute instance is to be launched at the instance host) submits a configuration request to the network-accessible service (e.g., a request to launch or stop a compute instance, or a request to attach a storage device or a network device to an instance), at least a subset of control-plane operations required for the configuration request may be performed at the selected control server. Based on results of the control-plane operations, one or more low-level commands may be transmitted over the secure network channel to the instance host and executed there, for example by a remote command executor. Results of the low-level commands may be provided back to the control server, and a corresponding response to the client's configuration request may be generated by the control server.

As indicated in FIG. **1**, in at least some embodiments, control servers of the provider network pools **120** may also be configurable to manage or administer resource instances instantiated at third-party networks **125** and/or client networks **127**. For example, a third party entity or a client may select some set of hardware at their data centers as candidates for hosting resource instances of a network-accessible service of the provider network. Pools **135** (e.g., pool **135**A in third-party network **125** or pool **135**B of client network **127**) of such candidate instance hosts may be identified in some embodiments by the third parties and/or the clients. In some implementations, control-plane components of the network-accessible service may execute some set of capability tests at the candidate instance hosts, e.g., to verify that the candidates are capable of the desired functionality and/or have adequate security safeguards, before the candidates are accepted as instance hosts of the service. From the perspective of the control server manager, in the depicted embodiment, each candidate pool **135** may be deemed to be within its own security zone—for example, third-party candidate instance host pool **135**A is shown in security zone **103**D in FIG. **1**, while client candidate instance host pool **135**B is shown in security zone **103**E. In some embodiments, such external security zones may by default be deemed to have lower secu-

rity levels than the security zones located within the provider network, and as a result the control plane manager **180** may be configured to select highly secure control servers for the external instance hosts.

Control Server Manager Interactions

FIG. **2** illustrates examples of operations that may be performed to select control servers for instance hosts, according to at least some embodiments. In the depicted embodiment, when a new instance host is to be brought online for a given network-accessible service, a procedure called instance host registration may be implemented. As part of such a procedure, a registration request **202** providing various details regarding the new instance host (such as its hardware and/or software specifications, network address(es) of management software modules such as a hypervisor installed on the instance host, its location, and various security-related properties that apply to the instance host) may be transmitted to the control server manager **180**.

The control server manager **180** may consult the security policies **182** and/or a control server inventory database **206** to determine which specific control servers should be selected for the new instance host in the depicted embodiment. For some instance hosts, a control server located within the same security zone or data center may suffice—e.g., if the new instance host is itself deemed to have appropriate physical and/or network security protocols in place. For other instance hosts, a control server (or a set of control servers) may be selected in a different security zone or a different data center than the instance host, typically with a higher level of security in place than the instance host.

The control server manager **180** may send notifications (e.g., including the network addresses of the instance host's management software) **252** to the selected control server(s) **272**. The control servers **272** may then establish secure network channels or connections, e.g., by issuing connection establishment requests **282**, to the instance hosts. The channels may be used for subsequent low-level commands submitted by the control servers **272** after control plane operations corresponding to client-submitted configuration requests are performed at the control servers, as described below. The creation of the network channel may also serve as a response to the registration request **202** in some embodiments, indicating to the instance host that it has been successfully registered with the network-accessible service. In some embodiments, appropriate pre-instantiated control servers may not be found for a new instance host, e.g., because the resource utilization levels of the pre-existing control servers are above a desired threshold. In such cases, the control server manager **180** may instantiate new control servers for the new instance host.

Control Server Components

FIG. **3** illustrates example components of control servers configured for remote configuration of instance hosts, according to at least some embodiments. The mapping between the illustrated components, and hardware/software servers on which the components are implemented, may vary over time and in different embodiments, and may in at least some cases be determined by control server managers **180** (not shown in FIG. **3**). For example, in some implementations, it may be possible to instantiate each of the illustrated components on a single computing device, while in other embodiments, one or more computing devices may be used for instances or nodes of a particular component (e.g., multiple workflow manager nodes may be instantiated, with one or more workflow manager nodes incorporated at a given computing device). In one implementation, each of the depicted components may be implemented using at least a respective operating system process.

A client and third party interaction manager component **335** may be responsible for receiving incoming client requests **301** and/or third party requests **302**, such as instance launch or configuration requests, or approval requests for third party or client-owned platforms in the depicted embodiment. Is some embodiments, one or more programmatic interfaces (such as web pages, web sites, APIs, graphical user interfaces or command-line tools) may be implemented to support the client interactions and/or third party interactions. Instance state manager **310** may be responsible for orchestrating configuration operations in response to client or third-party requests, for responding to outages or unexpected instance shutdowns, and/or for registering new instance hosts in the depicted embodiment. For example, in response to an instance launch request from a client, the instance state and recovery manager **310** may identify (with the help of capacity manager **305**) exactly which instance host is to be used for the launch, and may then issue a launch command to the workflow manager **325**, to be translated into lower-level commands for eventual execution at the selected instance host. Authorization/authentication manager **304** may be responsible for verifying the identity and/or permissions of the clients and/or third parties whose configuration requests are received. In at least some embodiments, the authorization/authentication manager **304** may also be responsible for generating credentials to be used at the instance hosts to implement some types of configuration operations. In one implementation, to further enhance the security of the system, some such credentials may have associated validity expiration times, and short validity periods (e.g., of a few minutes) may be selected so that even if the credentials are intercepted or obtained by attackers, they cannot be used for very long and therefore cannot lead to much damage or data loss. In some embodiments, an identity management service may be implemented at the provider network, and the authorization/authentication manager **304** may interact with (or comprise an element of) the identity management service.

Capacity manager **305** may be configured in the depicted embodiment to ensure that instance host pools **110** are adequately sized for the expected demand, and/or to move resources between pools if needed. Capability tester **315** may be configured to run tests (such as performance tests, security-related tests, software stack confirmations, and the like) to help with the decision to approve third party candidate platforms and/or to verify that instance hosts within the provider network are adequately provisioned. Metering/billing manager **330** may be configured to determine, based for example on metrics such as network request counts, measured traffic, I/O counts, CPU utilization and the like, how much a given client is to be charged for using a particular resource instance over a billing period, in accordance with the particular pricing plan in effect for the client.

Configuration definer **320** may be responsible in the depicted embodiment for generating, for a particular instance type to be launched, details of a specific configuration layout (e.g., names of various file systems and software devices to be set up, parameter values for various tunable settings, and the like) to be implemented at a particular instance host. Workflow manager **325** may be responsible for receiving the high-level command issued by the instance state manager **310** and configuration layout details from the configuration definer **320**, and translating the command into a workflow that includes one or more lower-level commands. Workflow manager **325** may then hand off the workflow commands to the command communicator **340**, which may transmit the corre-

sponding command sequence **381** (e.g., formatted in JSON or XML) to a selected instance host (e.g., via HTTPS) for execution via RCEs. In at least one embodiment, some of the configuration operations to be performed at the instance hosts may require the acquisition and/or configuration of resources at other network-accessible services of the provider network—e.g., a storage device implemented by a storage service may need to be obtained or configured for a compute instance at an instance host, or a virtual network interface managed by a networking service may need to be configured for a compute instance. In such an embodiment, the workflow manager **325** and/or the instance state manager **310** may communicate with administrative components of such other services, e.g., with service managers **327**, to obtain and/or configure the resources at the other services before the low-level commands are sent to the instance hosts.

In some embodiments, a locking service **375** may be used by the workflow manager **325** (or by other components illustrated in FIG. **3**) to ensure that an instance host configuration does not get corrupted due to conflicting or overlapping modification requests—e.g., an exclusive lock on an instance host may be required before a configuration change of a particular type is allowed. Logical processes responsible for making ownership/assignment decisions for various components and/or tasks of the system may also require concurrency control. In some embodiments, the locking service **375** may be a component of the control servers, while in other embodiments it may be an external entity. For example, a pre-existing locking service used for other purposes in the provider network may be used, or a new locking service may be used. A control server may also include an event listener **345** in some embodiments, configured to receive notifications when certain types of events (such as unexpected shutdowns, hardware or software errors or failures that may affect resource instances) occur at instance hosts. The event listener **345** may transmit the information about events to the instance state manager in some embodiments, which may interpret them appropriately to determine, for example, whether instance state information needs to be updated. In at least some embodiments, command communicator **340** may also submit low level commands to the instance hosts to collect performance or other metrics from the instance hosts, e.g., on behalf of metering manager **330**; in such embodiments, the set of commands issued by the command communicator may include non-modifying commands for metrics collection, as well as modifying commands to implement configuration changes.

It is noted that while instance state manager **310**, as indicated by its name, may be aware of the state of various resource instances, lower-level components such as workflow manager **325**, command communicator **340**, and/or event listener **345** may be stateless, at least in the sense that knowledge of, or details about, instance state may not be needed by such lower-level components to perform their functions in the depicted embodiment. By restricting information about instance states to a limited set of components, the implementation of stateless components such as the workflow manager and the command communicator may be substantially simplified in such embodiments. It is also noted that while the double arrows of FIG. **3** indicate examples of some of the types of interactions that may occur between the various control server components illustrated, additional types of interactions may also be supported between the components in at least some embodiments—e.g., any one of the components may be able to communicate with any other component in some embodiments. In various embodiments, a subset (or all) of the control server components shown in FIG. **3** may be implemented at a different security zone than the instance host being controlled.

Instance Host Components

FIG. **4** illustrates example components of an instance host **401**, according to at least some embodiments. As shown, the instance host may include a command receiver component **405**, such as a web server, configured to receive the sequence of commands generated by the command communicator **340** of the control server. The instance host may also include a hypervisor **417** providing the virtualization functionality on top of the bare hardware of the host. The hypervisor **417** may organize the resources of the instance host platform into a plurality of domains in the depicted embodiment, with one domain (which may be called domain zero) being used for administration, and the other domains being used for resource instances. An instance of an operating system **415** may be set up in domain zero. In response to each received command, or to a sequence of commands, the command receiver **405** may instantiate a remote command executor (RCE) **430**, such as **430**A or **430**B. The RCE **430** may then issue a request for an operation, e.g., an operation directed to the hypervisor **417** or a set of system calls directed to the domain-zero operating system **415**. Depending on the specific commands being implemented, in some embodiments credentials or metadata sent from the control server may also be used for the commands; in some cases, encrypted data (e.g., client application data) may also be used. In some embodiments RCE **430***s* may be considered, or implemented as, components of the domain-zero operating system **415** or the hypervisor **417**. After issuing its operation request(s), and receiving the results (including for example return codes, error output or standard output), a given RCE may terminate or exit in the illustrated embodiment. The RCE may exit or terminate of its own accord in some implementations, while in other implementations an RCE may be terminated by the command receiver **405** (e.g., using a "kill" signal or some other mechanism). In other embodiments, RCEs may remain in existence for longer time periods than needed just to initiate a given operation—e.g., a pool of RCEs may be maintained. In at least one implementation, each RCE may represent a CGI process or thread of execution. In some embodiments, an RCE may start a long-running operation and exit, and the results of the long-running operation (which may continue after the RCE exits) may be obtained asynchronously by the command receiver.

The operations initiated by the RCEs may (if the operations succeed) eventually result in the implementation of the configuration commands from the workflow manager **325**, resulting for example in the instantiation of (or configuration modifications of) various virtualized resource instances **445**, such as compute resources **450**A or **450**B, storage resources **460**A or **460**B, or network resources **470**A or **470**B. The RCEs and the command receiver may also be stateless with respect to instance state, in the sense that they may be unaware of what state a particular instance is in at a given time, in the depicted embodiment. In some embodiments where the instance host is organized into domains by the hypervisor, each virtual resource instance may correspond to a respective domain. The instance host may also comprise an event dispatcher **410** in the depicted embodiment. The event dispatcher may subscribe to one or more event monitors (e.g., monitors implemented within the hypervisor **417** or the domain-zero operating system **415**). The event monitor(s) may notify the event dispatcher if and when certain types of events occur at the instance host, and the event dispatcher may notify the event

listener **445** at a control server about the events, either directly or via the command receiver in various embodiments.

Example Request/Response Interactions

FIG. **5** illustrates example interactions between clients, control servers, and instance host platforms, according to at least some embodiments. As shown, a client **502** of a network-accessible service may submit an instance configuration request **540** to a control server **510** that has been selected, for security and/or performance reasons as described above, for an instance host of the network-accessible service in the depicted embodiment. The client request may be transmitted via a programmatic interface such as a web page or an API implemented by an interaction manager component **335** of a control server **410** in some embodiments. A number of components of the control server architecture (such as the instance state manager **310**, the workflow manager **325** and/or the command communicator **340**) may cooperate to translate the instance configuration request into a sequence of low-level commands **550** that are transmitted to the instance host **401**. In some embodiments, for certain types of commands **550**, associated data, metadata and/or credentials in the form of tokens may also be sent to the instance host **401**. As shown, the control server **510** selected for the instance host **401** may be in a different security zone **103**A than the security zone **103**B of the instance host.

The low-level command may be translated into RCE operations in the depicted embodiment at the instance host platform **401**. As shown, an RCE may be instantiated (element **561** of FIG. **5**), e.g., by spawning a new process or thread, the RCE may issue or perform one or more operations (element **562**), and then exit or terminate (element **563**). The results **570** of the commands may be sent back to the control server **510**. Based on the results **570**, an instance configuration response **580** may be sent back to the requesting client **502**. Command sequence **550** and/or results **570** may be transmitted using any appropriate secure networking protocol, such as HTTPS, in various embodiments. The commands and results may be formatted in accordance with a variant of JSON or XML in some embodiments. The command protocol used may support at least some idempotent operations in various embodiments. In some embodiments, the command protocol may support a variety of other command types and functions including performance metrics collections, log record collection and the like—e.g., in order to determine billing amounts for a client that owns one or more resource instances at the instance host platform **401**, low-level commands may be issued by the control server **510** to determine how many operations of various kinds the client issued to the instances, or how much network traffic the client incurred at the instance host platform. In some implementations, a mechanism other than RCEs may be used for certain types of control server-requested operations such as metrics or log record collection, while in other implementations RCEs may be used for both configuration modification and metrics/log collection.

According to at least one embodiment, encrypted metadata or encrypted credentials may be sent to the instance host **401** by the control server **510** to be used for the low-level commands. For example, in one implementation, a component (such as the authorization/authentication manager **304**) of the control server **510** may determine a short deadline before which a particular low-level command is to be completed at the instance host. A corresponding authentication security token with a short validity period may then be generated, encrypted, and transmitted to the instance host, e.g., via the command communicator. In at least some embodiments, the encrypted token or metadata may have to be decrypted at the instance host before the corresponding command may be executed, and a mechanism or pathway may be made available for the requesting client **502** to obtain a key to be used for the decryption. In one such embodiment, for example, the client **502** may transmit an optional key request **582** to the control server **510**, and receive the key to be used for decrypting the metadata or tokens in a corresponding response **584**. The client may then provide the key **588** to the instance host to be used for decrypting the metadata. The use of encrypted metadata/tokens and the alternate pathway for the key to be used for decryption may further decrease the likelihood that an intruder or attacker is able to misuse the instance host for unauthorized purposes in such embodiments. Various different kinds of encrypted metadata may be used in different implementations for enhancing security—for example, in some embodiments, the identifiers or names of some resources such as storage volumes, network interfaces and the like that are to be used in the low-level commands may be obfuscated via encryption, or client account names may be obfuscated. It is noted that in some embodiments and.or for certain types of low-level commands, encrypted metadata, credentials or tokens may not be used.

FIG. **6** illustrates an example of command flow starting from a workflow manager node at a control server, according to at least some embodiments. The illustrated example deals with the creation of a software RAID (redundant array of independent disks) device at an instance host, which may represent part of the configuration required to set up a new virtual compute instance. The example is provided here to illustrate, using a concrete example, the level of command detail at which different components of the control server and the instance host may operate in one embodiment; many other types of configuration operations, unrelated to RAID devices, may be implemented using commands of similar granularity in various embodiments. Elapsed time increases from the top to the bottom of FIG. **6**.

The workflow manager **325** may receive a high-level request to set up a root file system for a compute instance (element **602** of FIG. **6**) in the depicted embodiment, e.g., from instance state manager **310** in response to a client's request for a new compute instance. The workflow manager **325** may submit, to the command controller **340**, a command "block.raid1.create" directed to a block device subsystem (element **607**), requesting creation of a RAID1 device with specified parameter values (e.g., for software device names such as "md2", "sda3" and the like). The workflow manager **325** may have determined the parameter values based at least in part on a configuration definition or layout obtained from the configuration definer **320** for the new instance to be created.

In response to the "block.raid1.create" command, the command communicator **340** may submit an "RCE.exec" command to the instance host's command receiver **405** (element **612**). The command receiver **405** may in turn instantiate an RCE process or thread that executes the requested operation, in this case an invocation of an "mdadm" (multiple device administration) command at the domain-zero operating system layer (element **615**). The RCE process or thread may obtain the return value or exit code from the invocation (the "$?" value in element **618**), the standard output from the invoked operation (the "$1" value in element **618**), and the standard error from the invoked operation (the "$2" value in element **618**). These results may be transmitted by the command receiver back to the command communicator **340** (element **621**). The command controller **340** may in turn translate the results into a return value (e.g., "true", indicating success in this example) for "block.raid1.create" command it had

received, and transmit the return value back up to the workflow manager **325** (element **624**). The workflow manager **325** may similarly determine a return value for the "setup-instance-root-fs" command it had received, and provide this return value (also "true" in this example) to the instance state manager (element **627**). It is noted that the various components whose interactions are illustrated in FIG. **6** may not be aware of instance state information, which may be maintained by the instance state manager; instead, each of the depicted layers may simply perform lower level operations as needed, the accumulated results of which may contribute to a change in instance state (e.g., to a launch, a reconfiguration, or a termination of an instance).

HTTPS Command Requests and Responses

In at least some embodiments, as noted earlier, communications between the control servers and the instance hosts may be implemented using a secure protocol such as HTTPS. FIG. **7** illustrates example elements of command requests issued to an instance host from a control server, according to at least some embodiments. The HTTPS requests and responses formats used may comprise a plurality of headers and body elements, of which only a few examples are provided in FIG. **7**. As shown in element **702**, a request header used for a command sent to the instance host's command receiver from a control server's command communicator may include a digest or hash value **720** determined from the body of the request, so that the integrity of the request body can be verified at the instance host. The request header may specify the HTTP "PUT" verb or request method, as shown in element **721**, with a resource name that includes a "CHANGE-ID", a "UUID", and a "subsystemID". The CHANGE-ID may represent the specific client request that led to the command being issued; the CHANGE-ID corresponding to a given client request may be assigned for example by the client interaction manager **335** in some embodiments, and may be passed as a parameter in the command and response flow between the different components of the control server such as that illustrated in FIG. **6**. A universally unique identifier or UUID may be generated for the specific command request in the depicted embodiment, e.g., the command communicator **340** may generate a distinct UUID for each command request it sends to the instance host. The subsystem identifier may indicate the specific subsystem at the domain-zero operating system or hypervisor layer that is to be used to perform the requested operation in the depicted embodiment. In at least some embodiments, log records may be generated when a command request is sent, received, or when the corresponding operation is executed at the instance host, and the log records may include some or all of the CHANGE-ID, the UUID, and the subsystem ID, allowing for easier debugging or correlation analysis.

The body **706** of the HTTPS request may include a sequence of commands in accordance with a defined command protocol, specified using a JSON-like syntax in the depicted example of FIG. **7**. In some embodiments, the command protocol may allow the specification of file contents within the request body **706**, where the file contents may serve as parameters of some or all of the commands. For example, in FIG. **7**, the contents (e.g., in URL-encoded hexadecimal form) of two files with labels @FILE1 and @FILE2 may be included in the request body. As show, the keyword "commands" may indicate the sequence of commands included in the request. Three commands—"cmd1", "cmd2" and "cmd3" are shown in the sequence. "cmd1" has two file parameters FILE1 and FILE2, whose respective contents are indicated by @FILE1 and @FILE2. "cmd2" does not have any file parameters, while "cmd3" has a single file parameter FILE2. According to the command protocol in use, when an operation corresponding to "cmd1" is executed at the instance host via an RCE, the contents of @FILE1 and @FILE2 would be provided as parameters for the operation in the depicted embodiment. Similarly, when an RCE performs an operation corresponding to "cmd3", the contents of @FILE2 would be provided as a parameter. The specification of files in the request body in the manner shown in FIG. **7** may represent a convenience function in the depicted embodiment; other approaches, such as separate messages containing the file contents, may be used in other embodiments. In some embodiments, the command protocol may require that the commands be executed at the instance host in the order in which they appear in the request body; in other embodiments, such an ordering may be not be required. In one implementation, a maximum limit may be imposed on the number of commands that can be transmitted in a single request. In other implementations, no limit on the number of commands may be imposed. The UUID of the request header may be included in the body, as shown in FIG. **7**, in some embodiments. Different formats than the JSON-like format shown in FIG. **7**, such as XML, may be used to indicate the command sequence in other embodiments.

In some embodiments, the reply to the command request may include separate clauses or elements for each of the commands of the sequence. The response clause for the first command in the command sequence of request body **706** ("cmd1—F FILE1 FILE2") is shown in response body **710** for one embodiment. The "command-number" value ("1" in the depicted example) indicates that the clause is for the first command of the sequence. The standard output produced by the execution of the first command is indicated in the "stdout" field. The standard error output is indicated in the "stderr" field. The exit-code of the command (e.g., a value returned by the operating system or hypervisor component used) is indicated in the "exit-code" field. In addition, the response clause contains metrics for the wall-clock time (the elapsed time taken to complete the command on the instance host), as well as system and user CPU times indicating resource usage taken for the command at the instance host, expressed in units such as microseconds or milliseconds. Other formats than those shown in FIG. **7** may be used for commands and/or for command responses in various embodiments.

Methods of Using Remote Configuration to Enhance Control-Plane Security

FIG. **8** is a flow diagram illustrating aspects of operations that may be performed to enhance control-plane security for network-accessible services by selecting and using appropriate control servers for an instance host, according to at least some embodiments. As shown in element **801**, a set of one or more control servers may be selected, based at least in part on a security policy, for performing administrative operations associated with an instance host. The control servers may be selected because they differ from the instance host in an least one security-related property—e.g., the control servers may be located in a different data center (or a different room of a data center) with a more secure physical security protocol in effect, or the control servers may use a different network security protocol. In embodiments in which the provider network is logically subdivided into security zones, the control servers for some instance hosts may be co-located in the same security zone as the instance hosts based on security considerations, while the control servers for other instance hosts may be selected from different security zones than the instance hosts. In some cases the control servers may share a physical location with instance hosts, but might be isolated from the instance hosts by way of separately cabled networks

or separate logical networks such as VLANs (virtual local area networks). In some embodiments, the control servers and the instance host may be in different legal jurisdictions, so that different laws and authorities may apply to the instance host than those that apply to the control servers. In some implementations the control servers may be selected as part of, or in conjunction with, an instance host registration process implemented by one or more network-accessible services of a provider network. A secure network communication channel may be established between the control server(s) and the instance host in at least some embodiments, e.g., using TLS (Transport Layer Security), SSL (secure sockets layer), or any other appropriate protocol (element **804**).

After control servers have been assigned to the instance host and a communication channel has been established, at least a subset of administrative operations corresponding to subsequent configuration requests directed at the instance host may be performed at the control servers. When the next configuration request for a service instance (e.g., a compute instance or a storage device) at the instance host is received (element **807**), associated administrative operations such as authorization, authentication, capacity management, quota checks, interactions with other network-accessible services of the provider network to obtain or configure additional resources, billing-related operations, and/or concurrency control operations may be performed at the selected control servers (element **810**). A set of low-level commands (e.g. commands at the level of system calls) to be run at the instance hosts may then be determined, based at least in part on results of the administrative operations (element **813**). In some implementations, for certain types of commands, a set of metadata, credentials, or security tokens may also be determined, such as short-validity-duration security tokens that may have to be used to implement the commands successfully at the instance hosts.

The low-level commands and/or associated data/metadata may be transmitted to the instance host (element **816**) over the secure communication channels. In some embodiments, the communication channels may be established on an as-needed basis rather than in advance. At the instance host, a remote command executor (RCE) module may be responsible for executing the commands (element **819**). Results of the low-level commands may be transmitted back to the control server(s) (element **822**). In turn, a response to the configuration request, based on the result indicators, may be provided to the source of the configuration request (e.g., a client of the provider network or a third-party entity) (element **825**). The next configuration request received may then be dealt with similarly, e.g., by repeating operations corresponding to elements **807** onwards. Operations similar to those illustrated in FIG. **8** may be performed at or for a number of different network-accessible services in various embodiments, such as virtualized computing services, storage-related services, database services, or networking-related services.

FIG. **9** is a flow diagram illustrating aspects of operations associated with the encryption and decryption of metadata or data that may be used for configuration operations at instance hosts, according to at least some embodiments. As shown in element **901**, various types of encrypted data (e.g., application-level data objects or resource identifiers), metadata (such as system-level resource identifiers or account identifiers), and/or security tokens or credentials may be generated at a control server assigned to perform administrative operations for a given instance host. In some implementations, some or all of the data, metadata or the security tokens/credentials may have validity periods associated with them, and a short-term validity period such as a few minutes may be selected, so that the data, metadata or credentials may not be usable for long even if they are obtained by an intruder or attacker. The encrypted data, metadata and/or credentials may be transmitted to the instance host (element **904**), e.g., using the same network channels as are used for the low-level commands.

Via a different pathway such as a connection established between the client (on whose behalf the configuration operations are to be performed) and the control server, a request may be received at the control server for a key to be used to decrypt the data, metadata or credentials (element **907**). The requester's identity and permissions may be verified to determine whether the key should be supplied in response (element **910**), and if the verification succeeds, the key may be provided to the requester (element **913**). The requesting client may then transmit the key to the instance host, and the key may be used at the instance host to decrypt the data, metadata and/or credentials to perform the commands corresponding to the client's configuration requests (element **916**). In some embodiments, different keys may have to be used for different configuration requests, while in other embodiments, a single key or a common set of keys may be used for data, metadata or security tokens associated with a number of different configuration requests.

It is noted that in various embodiments, operations other than those illustrated in the flow diagrams of FIGS. **8** and **9** may be implemented to support various techniques for enhancing control-plane security, and that some of the operations shown may not be implemented, or may be implemented in a different order or in parallel rather than sequentially. For example, multiple client configuration requests may be handled in parallel in some embodiments instead of using the sequential approach shown in FIG. **8**.

Use Cases

The techniques described above, of implementing an efficient, modular architecture for control-plane operations of various network-accessible services, may be beneficial in various types of environments in which large numbers of platforms are to be used for hosting virtualized resources. They may be particularly useful in environments where different data centers or geographical regions of a large provider network have different security protocols (e.g., for physical security or network security) in place. Such security differences may be even more likely in environments in which third-party business partners, or even clients, wish to utilize their on-premise resources for implementing various service instances. In addition to the security benefits, performance benefits may also be achieved by such a modular approach, as it may become possible to devote a greater fraction of the hardware resources to service instances rather than to control-plane functions. The provider network operator could also safeguard itself and its intellectual property from motions of discovery that might be filed in the instance hosts' jurisdiction if the control-plane services are located in a different jurisdiction.

Illustrative Computer System

In at least some embodiments, a server that implements a portion or all of one or more of the technologies described herein, including the techniques to implement the functionality of the control server managers, the various control server components and/or the instance hosts, may include a general-purpose computer system that includes or is configured to access one or more computer-accessible media. FIG. **10** illustrates such a general-purpose computing device **3000**. In the illustrated embodiment, computing device **3000** includes one or more processors **3010** coupled to a system memory **3020** (which may comprise both non-volatile and volatile memory modules) via an input/output (I/O) interface **3030**. Comput-

ing device **3000** further includes a network interface **3040** coupled to I/O interface **3030**.

In various embodiments, computing device **3000** may be a uniprocessor system including one processor **3010**, or a multiprocessor system including several processors **3010** (e.g., two, four, eight, or another suitable number). Processors **3010** may be any suitable processors capable of executing instructions. For example, in various embodiments, processors **3010** may be general-purpose or embedded processors implementing any of a variety of instruction set architectures (ISAs), such as the x86, PowerPC, SPARC, or MIPS ISAs, or any other suitable ISA. In multiprocessor systems, each of processors **3010** may commonly, but not necessarily, implement the same ISA. In some implementations, graphics processing units (GPUs) may be used instead of, or in addition to, conventional processors.

System memory **3020** may be configured to store instructions and data accessible by processor(s) **3010**. In various embodiments, the volatile portion of system memory **3020** may be implemented using any suitable memory technology, such as static random access memory (SRAM), synchronous dynamic RAM or any other type of memory. For the non-volatile portion of system memory (which may comprise one or more NVDIMMs, for example), in some embodiments flash-based memory devices, including NAND-flash devices, may be used. In at least some embodiments, the non-volatile portion of the system memory may include a power source, such as a supercapacitor or other power storage device (e.g., a battery). In various embodiments, memristor based resistive random access memory (ReRAM), three-dimensional NAND technologies, Ferroelectric RAM, magnetoresistive RAM (MRAM), or any of various types of phase change memory (PCM) may be used at least for the non-volatile portion of system memory. In the illustrated embodiment, program instructions and data implementing one or more desired functions, such as those methods, techniques, and data described above, are shown stored within system memory **3020** as code **3025** and data **3026**.

In one embodiment, I/O interface **3030** may be configured to coordinate I/O traffic between processor **3010**, system memory **3020**, and any peripheral devices in the device, including network interface **3040** or other peripheral interfaces such as various types of persistent and/or volatile storage devices used to store physical replicas of data object partitions. In some embodiments, I/O interface **3030** may perform any necessary protocol, timing or other data transformations to convert data signals from one component (e.g., system memory **3020**) into a format suitable for use by another component (e.g., processor **3010**). In some embodiments, I/O interface **3030** may include support for devices attached through various types of peripheral buses, such as a variant of the Peripheral Component Interconnect (PCI) bus standard or the Universal Serial Bus (USB) standard, for example. In some embodiments, the function of I/O interface **3030** may be split into two or more separate components, such as a north bridge and a south bridge, for example. Also, in some embodiments some or all of the functionality of I/O interface **3030**, such as an interface to system memory **3020**, may be incorporated directly into processor **3010**.

Network interface **3040** may be configured to allow data to be exchanged between computing device **3000** and other devices **3060** attached to a network or networks **3050**, such as other computer systems or devices as illustrated in FIG. **1** through FIG. **9**, for example. In various embodiments, network interface **3040** may support communication via any suitable wired or wireless general data networks, such as types of Ethernet network, for example. Additionally, network interface **3040** may support communication via telecommunications/telephony networks such as analog voice networks or digital fiber communications networks, via storage area networks such as Fibre Channel SANs, or via any other suitable type of network and/or protocol.

In some embodiments, system memory **3020** may be one embodiment of a computer-accessible medium configured to store program instructions and data as described above for FIG. **1** through FIG. **9** for implementing embodiments of the corresponding methods and apparatus. However, in other embodiments, program instructions and/or data may be received, sent or stored upon different types of computer-accessible media. Generally speaking, a computer-accessible medium may include non-transitory storage media or memory media such as magnetic or optical media, e.g., disk or DVD/CD coupled to computing device **3000** via I/O interface **3030**. A non-transitory computer-accessible storage medium may also include any volatile or non-volatile media such as RAM (e.g. SDRAM, DDR SDRAM, RDRAM, SRAM, etc.), ROM, etc., that may be included in some embodiments of computing device **3000** as system memory **3020** or another type of memory. Further, a computer-accessible medium may include transmission media or signals such as electrical, electromagnetic, or digital signals, conveyed via a communication medium such as a network and/or a wireless link, such as may be implemented via network interface **3040**. Portions or all of multiple computing devices such as that illustrated in FIG. **10** may be used to implement the described functionality in various embodiments; for example, software components running on a variety of different devices and servers may collaborate to provide the functionality. In some embodiments, portions of the described functionality may be implemented using storage devices, network devices, or special-purpose computer systems, in addition to or instead of being implemented using general-purpose computer systems. The term "computing device", as used herein, refers to at least all these types of devices, and is not limited to these types of devices.

CONCLUSION

Various embodiments may further include receiving, sending or storing instructions and/or data implemented in accordance with the foregoing description upon a computer-accessible medium. Generally speaking, a computer-accessible medium may include storage media or memory media such as magnetic or optical media, e.g., disk or DVD/CD-ROM, volatile or non-volatile media such as RAM (e.g. SDRAM, DDR, RDRAM, SRAM, etc.), ROM, etc., as well as transmission media or signals such as electrical, electromagnetic, or digital signals, conveyed via a communication medium such as network and/or a wireless link.

The various methods as illustrated in the Figures and described herein represent exemplary embodiments of methods. The methods may be implemented in software, hardware, or a combination thereof. The order of method may be changed, and various elements may be added, reordered, combined, omitted, modified, etc.

Various modifications and changes may be made as would be obvious to a person skilled in the art having the benefit of this disclosure. It is intended to embrace all such modifications and changes and, accordingly, the above description to be regarded in an illustrative rather than a restrictive sense.

What is claimed is:

1. A system, comprising:

a plurality of computing devices comprising one or more hardware processors and memory, wherein the plurality of computing devices are configured to:

determine, by at least one of the computing devices and in accordance with a security policy of a network-accessible service implemented at a provider network, that at least a subset of administrative operations associated with configuration of a service instance at a particular instance host are to be performed at one or more control servers of the plurality of computing devices, wherein:

at least one control server of the one or more control servers differs from the particular instance host in at least one security-related property indicated in the security policy; and

the at least one security-related property of the at least one control server comprises a higher level of security than the particular instance host;

establish, by at least one of the computing devices, a secure communication channel between a particular control server of the one or more control servers and the particular instance host;

in response to an indication of a configuration request directed at the service instance,

perform one or more administrative operations associated with the configuration request at the one or more control servers;

transmit, by one of the computing devices implementing the particular control server, at least one command via the secure communication channel to a command receiver instantiated at the particular instance host, wherein the at least one command is determined based at least in part on a result of an administrative operation of the one or more administrative operations;

receive, by one of the computing devices implementing the particular control server, a command result from the particular instance host via the secure communication channel; and

provide, by at least one of the computing devices and based at least in part on the command result, a response to the configuration request.

2. The system as recited in claim **1**, wherein the particular instance host is located at a first data center, and wherein the one or more control servers are located at a second data center, wherein the first data center differs from the second data center in at least one of: (a) a physical security protocol governing access to data center assets, (b) a network security protocol, (c) country of location, or (d) legal authorities granted jurisdiction.

3. The system as recited in claim **1**, wherein the one or more administrative operations associated with the configuration request include one of: (a) an authorization operation, (b) an authentication operation, (c) a capacity management operation, (d) a quota check, (e) an interaction with a different network-accessible service of the provider network to obtain access to a resource of the different network-accessible service, (f) a billing account check, or (g) a concurrency control operation associated with managing concurrent updates to internal data structures of the network-accessible service.

4. The system as recited in claim **1**, wherein the plurality of computing devices are further configured to:

transmit, to the particular instance host via the secure channel, at least one of (a) a security token to be used to execute the at least one command, wherein the security token is configured with a validity period determined at the one or more control servers, or (b) encrypted data to be used to execute the at least one command.

5. The system as recited in claim **1**, wherein the network-accessible service comprises one of: a virtual computing service, a storage service, or a database service.

6. A method, comprising:

performing, by a plurality of computing devices comprising one or more hardware processors and memory:

determining, by at least one of the computing devices and in accordance with a security policy of a network-accessible service implemented at a provider network, that at least a subset of administrative operations associated with configuration of a service instance at a particular instance host are to be performed at one or more control servers of the plurality of computing devices, wherein:

at least one control server of the one or more control servers differs from the particular instance host in at least one security-related property; and

the at least one security-related property of the at least one control server comprises a higher level of security than the particular instance host; and

in response to an indication of a configuration request directed at the service instance,

implementing one or more administrative operations associated with the configuration request at the one or more control servers;

transmitting, by one of the computing devices implementing a particular control server, at least one command via a network connection from the particular control server of the one or more control servers to the particular instance host;

receiving, by one of the computing devices implementing the particular control server, a command result from the particular instance host via the network connection; and

providing, by at least one of the computing devices and based at least in part on the command result, a response to the configuration request.

7. The method as recited in claim **6**, wherein the particular instance host is located at a first data center, and wherein the one or more control servers are located at a second data center.

8. The method as recited in claim **6**, wherein the security-related property comprises an indication of one of: (a) a physical security protocol governing access to data center assets, (b) a network security protocol, (c) a country of location, or (d) a legal authority granted jurisdiction.

9. The method as recited in claim **6**, wherein the one or more administrative operations associated with the configuration request include one of: (a) an authorization operation, (b) an authentication operation, (c) a capacity management operation, (d) a quota limit check, (e) an interaction with a different network-accessible service of the provider network to obtain access to a resource of the different network-accessible service, (f) a billing account check, or (g) a concurrency control operation associated with managing concurrent updates to internal data structures of the network-accessible service.

10. The method as recited in claim **6**, further comprising performing, by the plurality of computing devices:

generating a security credential with a validity period to be used to execute the at least one command at the particular instance host, and

transmitting the security credential to the particular instance host.

11. The method as recited in claim 6, further comprising performing, by the plurality of computing devices:

generating encrypted data to be used to execute the at least one command at the particular instance host, and

transmitting the encrypted data to the particular instance host.

12. The method as recited in claim 6, further comprising performing, by the plurality of computing devices:

invoking, by a stateless command executor module at the instance host, one or more system calls to implement the at least one command.

13. The method as recited in claim 6, further comprising performing, by the plurality of computing devices:

instantiating a secure communication channel between at least one control server and the particular instance host;

wherein said transmitting comprises utilizing the secure communication channel.

14. The method as recited in claim 6, wherein the network-accessible service comprises one of: a virtual computing service, a storage service, or a database service.

15. The method as recited in claim 11, wherein the encrypted data comprises one of: (a) an encrypted identifier of a resource, or (b) encrypted application-level data.

16. The method as recited in claim 11, further comprising performing, by the plurality of computing devices:

providing, via a programmatic interface, a key usable to decrypt the data at the instance host.

17. A non-transitory computer-accessible storage medium storing program instructions that when executed on one or more computing devices

determine, in accordance with a security policy of a network-accessible service implemented at a provider network, that at least a subset of administrative operations associated with configuration of a service instance at a first instance host located within a first security zone of the provider network are to be performed at a first control server of the plurality of computing devices located within a different second security zone, wherein:

the first and second security zone differ in at least one security property; and

the second security zone of the first control server comprises a higher level of security than first security zone of the first instance host;

determine, in accordance with the security policy, that at least a subset of administrative operations associated with configuration of a service instance at a second instance host located within a second security zone are to be performed at a second control server located within the second security zone;

provide identification information pertaining to the first instance host to the first control server, enabling the first control server to establish a first network channel to be used for transmission of configuration commands from the first control server to the first instance host; and

provide identification information pertaining to the second instance host to the second control server, enabling the second control server to establish a second network channel to be used for transmission of configuration commands from the second control server to the second instance host.

18. The non-transitory computer-accessible storage medium as recited in claim 17, wherein the first security zone differs from the second security zone in at least one of: (a) a physical security protocol governing access to data center assets, (b) a network security protocol, (c) a country of location, or (d) a legal authority granted jurisdiction.

19. The non-transitory computer-accessible storage medium as recited in claim 17, wherein the subset of administrative operations associated with the configuration of the service instance at the first instance host located within a first security zone include one of: (a) an authorization operation, (b) an authentication operation, (c) a capacity management operation, (d) a quota limit check, (e) an interaction with a different network-accessible service of the provider network to obtain access to a resource of the different network-accessible service, (f) a billing account check, or (g) a concurrency control operation associated with managing concurrent updates to internal data structures of the network-accessible service.

20. A non-transitory computer-accessible storage medium storing program instructions that when executed on one or more computing devices:

receive, at a control server selected based at least in part on a security policy to perform a set of administrative operations associated with a service unit of a network-accessible service of a provider network, a service configuration request,

implement, at the control server, one or more administrative operations associated with the service configuration request;

transmit at least one command via a secure network connection to a target host associated with the service unit, wherein at least one security-related property of the control server comprises a plurality of security-related properties of the target host;

receive a command result from the target host via the secure network connection; and

provide, based at least in part on the command result, a response to the service configuration request.

21. The non-transitory computer-accessible storage medium storing program instructions as recited in claim 20, wherein the control server is located at a first data center, and wherein the target host is located at a second data center.

22. The non-transitory computer-accessible storage medium storing program instructions as recited in claim 20, wherein the one or more administrative operations associated with the service configuration request include one of: (a) an authorization operation, (b) an authentication operation, (c) a capacity management operation, (d) a quota limit check, (e) an interaction with a different network-accessible service of the provider network to obtain access to a resource of the different network-accessible service, (f) a billing account check, or (g) a concurrency control operation associated with managing concurrent updates to internal data structures of the network-accessible service.

* * * * *