PARAMETER BASED KEY DERIVATION

CROSS-REFERENCE TO RELATED APPLICATIONS

[0001] This application claims priority to U.S. Patent Application Nos. 13/248,962, filed on September 29, 2011, entitled "PARAMETER BASED KEY DERIVATION" (Attorney Docket No. 90204-813889 (029400PC)); 13/248,953, filed on September 29, 2011, entitled "TECHNIQUES FOR CLIENT CONSTRUCTED SESSIONS" (Attorney Docket No. 90204-818478 (032300US)); and 13/248,973, filed on September 29, 2011, entitled "KEY DERIVATION TECHNIQUES" (Attorney Docket No. 90204-813890 (029500US)), the full disclosures of which are incorporated herein by reference.

BACKGROUND

[0002] Computing environments take many forms. As an example, organizations often utilize networks of computing devices to provide a robust set of services to their users. Networks often span multiple geographic boundaries and often connect with other networks. An organization, for example, may support its operations using both internal networks of computing resources and computing resources managed by others. Computers of the organization, for instance, may communicate with computers of other organizations to access and/or provide data while using services of another organization. In many instances, organizations configure and operate remote networks using hardware managed by other organizations, thereby reducing infrastructure costs and achieving other advantages.

[0003] While diverse computing environments have proven useful for a wide variety of applications, such environments present many challenges. For example, configuring computer resources in furtherance of one organizational goal can adversely affect furtherance of another organizational goal. For example, effective management of computing resource security can often come at the cost of efficient access to data and services. Balancing the goals of security and efficiency can be quite challenging, often requiring significant effort and resources.

BRIEF DESCRIPTION OF THE DRAWINGS

[0004] Figure 1 shows an illustrative example of a computing environment that can be used to implement various aspects of the present disclosure in accordance with at least one embodiment;

[0005] Figure 2 shows an illustrative example of an environment that includes a computing resource provider that manages multiple fault zones in accordance with at least one embodiment;

[0006] Figure 3 shows an illustrative example of an environment inside of a fault zone of Figure 2, in accordance with at least one embodiment;

[0007] Figure 4 shows an illustrative example of a computing resource configuration that can be used to support an environment such as the environment shown in Figure 3, in accordance with at least one embodiment;

[0008] Figure 5 is a diagram that illustrates an example manner in which various elements participating in a computing environment may be allocated different scopes of authority in accordance with at least one embodiment;

[0009] Figure 6 is a diagram illustrating an example manner in which information may be communicated among participants in a message signature verification process in accordance with at least one embodiment;

[0010] Figure 7 is a flowchart showing an illustrative example of a process for signing messages in accordance with an embodiment;

[0011] Figure 8 is a flowchart showing an illustrative example of a process for signature verification in accordance with at least one embodiment;

[0012] Figure 9 is a diagram illustrating an example manner of distributing keys in accordance with at least one embodiment;

[0013] Figure 10 is a diagram illustrating an example manner of distributing keys in a manner that provides various scopes of authority in accordance with at least one embodiment;

[0014] Figure 11 is a flowchart showing an illustrative example of a process of key derivation in accordance with at least one embodiment;

[0015] Figure 12 is a diagram illustrating multiple-restriction key derivations in accordance with at least one embodiment;

[0016] Figure 13 is an illustrative example of a function for deriving a signature, in accordance with at least one embodiment;

[0017] Figure 14 is an illustrative example of how multiple key derivation may be performed and used in accordance with at least one embodiment;

[0018] Figure 15 is a diagram illustrating an example manner in which keys may be derived, in accordance with at least one embodiment;

[0019] Figure 16 is a diagram illustrating another example manner in which keys may be derived, in accordance with at least one embodiment;

[0020] Figure 17 is a diagram illustrating yet another example manner in which keys may be derived, in accordance with at least one embodiment;

[0021] Figure 18 is a flowchart showing an illustrative example of a process for initiating a session, in accordance with at least one embodiment;

[0022] Figure 19 is a flowchart showing an illustrative example of a process for generating a session key, in accordance with at least one embodiment.

[0023] Figure 20 is a flowchart showing an illustrative example of a process for obtaining access to one or more computing resources during a session in accordance with at least one embodiment;

[0024] Figure 21 is a flowchart showing an illustrative example of a process for determining whether to grant requested access to one or more computing resources in accordance with at least one embodiment;

[0025] Figure 22 is a flowchart showing an illustrative example of a process for delegating authority in accordance with at least one embodiment;

[0026] Figure 23 is a diagram representing an illustrative example of multiple delegations of authority in accordance with at least one embodiment; and

[0027] Figure 24 is a diagram representing an illustrative example of a manner in which keys may be derived using keys from multiple authorities.

DETAILED DESCRIPTION

[0028] In the following description, various embodiments will be described. For purposes of explanation, specific configurations and details are set forth in order to provide a thorough understanding of the embodiments. However, it will also be apparent to one skilled in the art that the embodiments may be practiced without the specific details. Furthermore, well-known features may be omitted or simplified in order not to obscure the embodiment being described.

[0029] Techniques described and suggested herein include systems and methods for key generation, in accordance with various embodiments. The keys may be used for various purposes, such as authentication and participation in message signing schemes. In an embodiment, a computing resource provider provides computing services to customers based at least in part on electronic requests received from user devices of the services. The services may be any suitable service that may be offered including, but not limited to, access to data, access to computing resources to perform operations, access to data storage services, and the like.

[0030] To ensure that services are provided in a secure manner, various embodiments of the present disclosure utilize techniques to authenticate requests (also referred to as "messages") to ensure that the requests are legitimate. In an embodiment, requests are authenticated using a Hash Message Authentication Code (HMAC) algorithm or other suitable algorithm, as discussed in more detail below.

[0031] In an embodiment, both the authenticating party (e.g., user of services or party acting on behalf of the user) and the authenticator (e.g., provider of services or party acting on behalf of the provider) share a secret credential, which may be referred to as a key. An authenticator may store shared secret credentials for multiple users. As part of a transaction, the authenticating party may sign requests using the shared secret credential, thereby forming a signature. The signature may be provided to the authenticator with the requests. The authenticator may use its own copy of the shared secret credential to generate a signature for the received requests and, by comparing if the generated signature matches the received signature (for example by being identical to the received signature), determine whether the requests were signed using the shared secret credential. If determined that the requests were signed using the shared secret credential, the requests may be considered authentic and, therefore, it may be determined that the requests should be fulfilled.

[0032] Because the interaction above is symmetric (i.e., both utilize common information when performing their roles), the shared secret credentials that an authenticator keeps can be used to both authenticate authenticating parties or to act on their behalf. As a result, a high degree of security is desirable to protect these credentials. Maintaining high degrees of security may have negative performance and availability consequences. For example, maintaining a high degree of security may include maintaining a centralized system for key storage. Such centralized systems, however, may cause a scaling bottleneck since the addition of users and/or services causes a greater burden to the centralized system. If such a centralized system fails, it may be difficult or impossible to authenticate requests. Thus, centralization provides both advantages for security and disadvantages for scaling and availability of services.

[0033] In an embodiment, negative impacts of such systems (and other systems) are reduced by utilizing a signing protocol that derives from shared secret credentials artifacts that may be used to prove that an authenticating party has a shared secret credential and, therefore, is likely authorized to obtain access specified in requests signed with the artifacts. In an embodiment, such artifacts are obtained by configuring authenticator computer systems to accept as a signature a value that is based at least in part on a derivation of a shared credential, instead of the shared credential itself. The derivation of the shared credential may be such that, as described more fully below, the derivation does not allow for practical determination of the shared credential.

[0034] For example, in an embodiment, authenticating parties are able to sign signatures with

HMAC(M, HMAC(X, credential)),

where M is a message, and HMAC(X, credential) is an artifact derived from a shared secret credential. The value for X may be some value that is known both by the authenticating party and the authenticator, and may be publicly available. For example, X may be a current date, encoded in a predetermined manner to ensure that HMAC(X, credential) is computed consistently by the authenticating party and the authenticator. As another example, X may be an identifier of a service with which the artifact is usable. As yet another example, X may encode multiple semantic meanings and be provided in a manner such that both the authenticating party and the authenticator consistently compute the artifact. The semantic meaning may be a restriction on use of the key, including meaning that indicates that no further derivations form the key should be used. Combining previous examples of the present paragraph, X may be encoded

as "20110825/DDS" where the string left of the slash represents a date and the string right of the slash represents the name of a service with which an artifact computed with X is usable. Generally, X may be any value or set of values encoded consistently for both the authenticating party and the authenticator. It should be noted that other suitable functions other than HMAC functions may be used, as discussed below.

[0035] Returning to the example utilizing HMACs, in an embodiment, values for X are chosen to provide additional advantages. As noted, X may (but does not necessarily) correspond to one or more semantic meanings. Semantic meanings such as time stamps, service names, regional names, and the like are used, in an embodiment, to provide a system where artifacts created in accordance with techniques of the present disclosure provide corresponding restrictions on use of keys derived from X. In this manner, even though compromise of keys generated may allow authentication by undesired parties, restrictions used to encode keys allow for the adverse effects to be minimized when keys are compromised. As an example, time restrictions used to derive keys provide an efficient way for a system to check if a submitted signature was signed with a key that was valid at the time of signature submission. As a concrete example, if a current date is used to derive a key and an authenticator system only accepts signatures submitted on the current date, the authenticator system will determine that signatures generated using keys derived with different dates are invalid. Similarly, a key derived with an identifier of a particular service would be invalid for use with another service. Other examples are provided below.

[0036] As noted, various techniques of the present disclosure allow for multiple parameters to be used to derive keys. In an embodiment, keys are derived from multiple parameters through multiple use of an HMAC function. For example, a key may be computed as follows:

$$K_S=HMAC(...HMAC(HMAC(HMAC(K, P_1), P_2), P_3)..., P_N),$$

where K is a shared secret credential and the P_i are parameters. The key, K_S, may be used to generate a signature, such as:

$$S=HMAC(K_S, M),$$

where M is a message, which may be canonicalized. In this manner, the key is derived in a layered manner, allowing for partial derivations of the key to be passed to various components of a distributed system. For example, $K_{P1}=HMAC(K, P_1)$ may be computed and passed on to one or more components of a distributed system. The components that receive K_{P1} may compute

K_{P2}=HMAC(K_{P1}, P_2), where P_2 may be the same for each component or different for some or all components. The values for K_{P2} calculated by the various components may pass the calculations to other components of the distributed systems which may compute K_{P3}=HMAC(K_{P2}, P_3). Each component may cache the results it calculates, and possible results computed and calculated by other components. In this manner, more security may be provided around a data store that stores shared secret keys because computations of derived keys may be performed by other components of the distributed system.

[0037] Techniques of the present disclosure also provide for the initiation of sessions. For example, as discussed, a shared secret credential and one or more parameters may be used to derive a key. Accordingly, parameters for a session may be used to generate a credential that may be used during the session. The credential may be used by the user that requested or, in some embodiments, by a user to whom the credential was passed and to whom access to one or more computing resources has been delegated. In such instances, because a delagatee of such access uses a key derived from a shared secret credential, but not the shared secret credential itself, a higher level of security is maintained and there is no need to rotate the shared secret credential to prevent future use by the delegatee. As discussed in more detail below, delegatees may also become delegators using techniques of the present disclosure, many of which are described in more detail below.

[0038] Figure 1 illustrates aspects of an example environment 100 for implementing aspects of the present disclosure in accordance with various embodiments. As will be appreciated, although a Web-based environment is used for purposes of explanation, different environments may be used, as appropriate, to implement various embodiments. The environment includes an electronic client device 102, which can include any appropriate device operable to send and receive requests, messages, or information over an appropriate network 104 and convey information back to a user of the device. Examples of such client devices include personal computers, cell phones, handheld messaging devices, laptop computers, set-top boxes, personal data assistants, electronic book readers, and the like. The network can include any appropriate network, including an intranet, the Internet, a cellular network, a local area network, or any other such network or combination thereof. Components used for such a system can depend at least in part upon the type of network and/or environment selected. Protocols and components for communicating via such a network are well known and will not be discussed herein in detail. Communication over the network can be enabled by wired or wireless connections, and

combinations thereof. In this example, the network includes the Internet, as the environment includes a Web server 106 for receiving requests and serving content in response thereto, although for other networks an alternative device serving a similar purpose could be used as would be apparent to one of ordinary skill in the art.

[0039] The illustrative environment includes at least one application server 108 and a data store 110. It should be understood that there can be several application servers, layers, or other elements, processes, or components, which may be chained or otherwise configured, which can interact to perform tasks such as obtaining data from an appropriate data store. As used herein the term "data store" refers to any device or combination of devices capable of storing, accessing, and retrieving data, which may include any combination and number of data servers, databases, data storage devices, and data storage media, in any standard, distributed, or clustered environment. The application server can include any appropriate hardware and software for integrating with the data store as needed to execute aspects of one or more applications for the client device, handling a majority of the data access and business logic for an application. The application server provides access control services in cooperation with the data store, and is able to generate content such as text, graphics, audio, and/or video to be transferred to the user, which may be served to the user by the Web server in the form of HTML, XML, or another appropriate structured language in this example. The handling of all requests and responses, as well as the delivery of content between the client device 102 and the application server 108, can be handled by the Web server. It should be understood that the Web and application servers are not required and are merely example components, as structured code discussed herein can be executed on any appropriate device or host machine as discussed elsewhere herein.

[0040] The data store 110 can include several separate data tables, databases, or other data storage mechanisms and media for storing data relating to a particular aspect. For example, the data store illustrated includes mechanisms for storing production data 112 and user information 116, which can be used to serve content for the production side. The data store also is shown to include a mechanism for storing log data 114, which can be used for reporting, analysis, or other such purposes. It should be understood that there can be many other aspects that may need to be stored in the data store, such as for page image information and to access right information, which can be stored in any of the above listed mechanisms as appropriate or in additional mechanisms in the data store 110. The data store 110 is operable, through logic associated therewith, to receive instructions from the application server 108 and obtain, update, or otherwise

process data in response thereto. In one example, a user might submit a search request for a certain type of item. In this case, the data store might access the user information to verify the identity of the user, and can access the catalog detail information to obtain information about items of that type. The information then can be returned to the user, such as in a results listing on a Web page that the user is able to view via a browser on the user device 102. Information for a particular item of interest can be viewed in a dedicated page or window of the browser.

[0041] Each server typically will include an operating system that provides executable program instructions for the general administration and operation of that server, and typically will include a computer-readable storage medium (e.g., a hard disk, random access memory, read only memory, etc.) storing instructions that, when executed by a processor of the server, allow the server to perform its intended functions. Suitable implementations for the operating system and general functionality of the servers are known or commercially available, and are readily implemented by persons having ordinary skill in the art, particularly in light of the disclosure herein.

[0042] The environment in one embodiment is a distributed computing environment utilizing several computer systems and components that are interconnected via communication links, using one or more computer networks or direct connections. However, it will be appreciated by those of ordinary skill in the art that such a system could operate equally well in a system having fewer or a greater number of components than are illustrated in Figure 1. Thus, the depiction of the system 100 in Figure 1 should be taken as being illustrative in nature, and not limiting to the scope of the disclosure.

[0043] Figure 2 shows an illustrative example of an environment 200 that includes a computing resource provider 202 that manages multiple fault zones 204 in accordance with at least one embodiment. A computing resource provider, in an embodiment, is an organization that operates computer hardware on behalf of one or more customers 206. The computing resource provider may provide computing resources in various ways. For example, in an embodiment, the computing resource provider 202 manages hardware that is configured for use by customers 206. The computing resource provider 202 provides an interface that allows the customers 206 to programmatically configure computing resources using the hardware. For example, the computing resource provider may maintain hardware servers that execute virtual computer systems that are programmatically controlled by the customer. As another example,

the computing resource provider 202 may manage various data stores to provide remote data storage solutions, such as high-durability data storage and block-level data storage.

[0044] A fault zone, in an embodiment, is a collection of computing resources that are separated by one or more fault boundaries such that each fault zone is tolerant to a fault of another fault zone. As an example, each fault zone 204 may be a separate data center. Thus, if one data center ceases being operational, perhaps due to a power outage or other disruptive event, other data centers may continue to operate. The fault zones may be each located in different geographic locations and some or all of the fault zones may be separated by geopolitical boundaries. For example, two or more of the fault zones may be in different countries. It should be noted that, for the purpose of illustration, the present disclosure provides numerous examples where fault zones are data centers. However, fault zones can be defined in numerous other ways. For example, separate rooms in the same data center may be considered separate fault zones in accordance with various embodiments. As another example, computing resources in the same location, but supported by different backup power generators and/or supported by different network resources, may be considered different fault zones. As yet another example, data centers may be clustered such that each cluster of data centers may be considered a fault zone. Further, there may be many reasons a fault zone may fail, including reasons relating to power grid operation, public network operation, political assertions of power, and other reasons.

[0045] In an embodiment, customers 206 communicate with the computing resource provider 202 over a network 208, such as the Internet. The customers 206 may have resources configured in one or more of the fault zones 204 and may communicate with the resources by sending electronic messages, such as messages invoking a web-service application programming interface (API) of the computing resource provider in order to configure and operate the resources. Customers may utilize resources in multiple fault zones in order to decrease the effects of potential failures that impact the customers' resources. A customer who utilizes resources of the computing resource provider 202 to operate a publicly accessible website may, for example, maintain web and other servers in separate fault zones so that, if servers in one fault zone fail, the public may still access the web site by accessing servers in another fault zone.

[0046] Figure 3 shows an illustrative example of an environment 300 inside of a fault zone 302, which may be a fault zone of a computing resource provider as illustrated in Figure 2. The fault zone 302, in an embodiment, includes computing resources that are used to provide various

services on behalf of customers. For example, as illustrated in Figure 3, the fault zone 302 includes computing resources that are used to provide a durable data storage service, which may cheaply and redundantly store relatively large amounts of data on behalf of customers. Such a service may be used when large amounts of data storage and/or security of the data storage is required, but when input/output performance is not high priority. The fault zone 306 may also include a block data storage service 306 which provides the use of block-level storage devices, physical devices and/or virtual, to customers. The customers may, for example, attach block-level storage devices to computer systems also utilized by the customers. Also illustrated is a virtual computer system service 308 which may provide computing services for customers. In an embodiment, the virtual computer system service 308 provides computing services by implementing virtual computer systems for the customers on physical servers maintained by the computing resource provider, although variations are possible, such as where physical computer systems are allocated to customers for customer use. In an embodiment related to virtual computer systems, the customers may programmatically manage the virtual computer systems according to their needs. For example, as illustrated in Figure 3, customers may configure virtual computer systems of the virtual computer system service 308 to server customers of the customers of the virtual computing service provider. The virtual computer systems may be, for instance, configured to operate a publicly accessible website. Both the customers of the virtual computing resource provider and the customers' customers may, in various embodiments, access the various services operated in the fault zone 302 by communicating with the services over a network 310, which may be the network 208 described above in connection with Figure 2.

[0047] It should be noted that the various embodiments illustrated in Figure 3, as with all illustrative embodiments shown in the Figures and described herein, are illustrative in nature and that variations are considered as being within the scope of the present disclosure. For example, other services different from those illustrated may be provided in the fault zone 302 in addition to or instead of the services illustrated. As illustrated by the ellipses in Figure 3, for instance, additional services may be operated in the fault zone 302. In addition, some services may utilize other services. For example, multiple services (such as a block-level data storage service 306 and a virtual computer system service 308) may be utilized together to provide other services, such as a relational database service, an electronic mail service, and, generally, any type of computing service that can be provided using resources of a computing resource provider.

[0048] As illustrated in Figure 3, each of the services of the computing resource provider may include a separate verifier 312. The verifier may be a computing device, collection of computing devices, application module, or other resource that verifies various attestations made by customers and possibly by other computer systems. In an embodiment, each of the verifiers 312 verifies message signatures that are produced in accordance with the various embodiments herein and then provided by customers in connection with requests to access computing resources, as described in more detail below. Keys and other relevant information may be propagated to the verifiers from a central key authority to enable the verifiers to verify information. It should be noted that each service having a verifier is an illustrative example of a particular embodiment, but that other arrangements are within the scope of the present disclosure. For example, a single verifier may support multiple services, even all services and may even support multiple fault zones.

[0049] Figure 4 shows an illustrative example of a computing resource configuration that can be used to support an environment such as the environment shown in Figure 3, in accordance with at least one embodiment. Figure 4 specifically shows a specific example where the fault zone in Figure 3 is a data center. Accordingly, returning to Figure 4, a data center 402 may include multiple server racks 404-406. The data center 402 is an example of one or more data centers that may be used in various embodiments of the present disclosure, such as data centers shown in Figure 4. The ellipsis between the server rack 404 and the server rack 406 indicates that the data center 402 may include any suitable number of server racks although, for clarity, only two are shown in Figure 4. Each server rack 404-406 may participate in maintaining services such as electric power and data communications to multiple server computers 408-414 and 416-422. Again, the ellipses indicate that the server racks 404-406 may include any suitable number of server computers. For example, the server computers 408-422 may include one or more virtual computer system (VCS) servers and/or one or more data store servers. Each server 408-422 may correspond to an implementation resource dedication unit.

[0050] In Figure 4, each server rack 404-406 is depicted as including a rack switch 424-426. The rack switches 424 and 426 may be responsible for switching packets of digital data to and from their respective sets of server computers 408-414 and 416-422. The rack switches 424-426 may be communicatively linked to a data center switching fabric 428 and then to a set of edge routers 430 that connects the data center 402 to one or more other computer networks including the Internet. The switching fabric may include any suitable set of networking components

including multiple interconnected switches 432-438 (for clarity, only four are shown in Figure 4) of one or more switch types arranged in one or more switching layers, as well as routers, gateways, bridges, hubs, repeaters, firewalls, computers, and suitable combinations thereof. In at least one embodiment, the rack switches 424-426 and the edge routers 430 are considered part of the switching fabric 428. The rack switches 424-426, the edge routers 430, and the components of the switching fabric 428 are examples of the network hardware 224 of Figure 2.

[0051] As noted above, various embodiments of the present disclosure allow for various levels of authority to be given for different reasons. Figure 5 is a diagram that illustrates an example way of a manner in which various elements participating in a computing environment may be allocated different scopes of authority in accordance with at least one embodiment. In Figure 5, a computing resource provider 502 is illustrated. In an embodiment, the computing resource provider 502 has authority over its resources and, as illustrated in Figure 5, is able to apportion that authority among various participants in the resources' use. It should be noted that, for the purpose of illustration consistent with other illustrations and descriptions therein, Figure 5 shows a computing resource provider 502 having authority over a domain. However, embodiments of the present disclosure are also applicable to other masters of authority domains. For instance, a master of authority may be a government or governmental organization, a sub-organization of another organization or, generally, any entity with authority over some domain.

[0052] Returning to the illustrative example of Figure 5, the computing resource provider 502 manages its authority by allowing different sub-entities to have authority over different sub-domains. For example, as shown in the Figure, each of a number of fault zones 504 of the computing resource provider are provided a corresponding sub-domain of the computing resource provider's 502 domain. Thus, each fault zone may have authority over its own resources, but not resources of another fault zone (although, in some instances authority over some sub-domains may be shared). Thus, in accordance with an embodiment, a fault zone may provide user access to computing resources in the fault zone, but not access to computing resources of another fault zone.

[0053] As noted above, each fault zone may include one or more services 506. Accordingly, as illustrated in Figure 5, each service may be responsible for a sub-domain of the domain of the corresponding fault zone 506. Thus, a service, in an embodiment, can provide access to resources accessible by the service, but not to other services. Each service may serve one or

more customers 508 and, therefore, each customer may be responsible for a sub-domain of authority of a corresponding service 506. Thus, in an embodiment, a customer may provide access to its own resources involved with a corresponding service, but not to another customer's service. As a concrete illustrative example, if the service is a virtual computing resource service, a customer may provide access (such as public access) to its own virtual computer systems, but not, without permission, to virtual computer systems of other customers.

[0054] As noted, the particular allocation of authority as illustrated in Figure 5 is for the purpose of illustration and numerous variations are considered as being within the scope of the present disclosure. As noted, embodiments of the present disclosure are applicable to domains of authority outside of domains managed by computing resource providers and sub-domains may be determined according to particular needs and circumstances. Further, Figure 5 shows customers of a virtual resource provider having the smallest sub-domains of authority. However, techniques of the present disclosure may allow customer domains to be divided into one or more sub-domains.

[0055] Various embodiments of the present disclosure relate to message signatures. Figure 6 is a diagram 600 illustrating an example manner in which information may be communicated among participants in a message signature verification process in accordance with at least one embodiment. In an embodiment, a key source 602 provides a key to both a message submitter 604 and a signature verifier 606. The key source may be a computer system configured to provide keys to at least the message submitter 604 and the signature verifier 606. The key source may also generate the keys using various techniques, including various embodiments described herein or may obtain generated keys from another source. The message submitter 604 may be a computer system configured to submit a message and a signature to the signature verifier 606 or other component that operates in connection with the signature verifier 606. The computer system of the message submitter 604 may be a computer system of a customer of a computing resource provider, for example. The signature verifier 606 may be a computer system configured to receive messages and signatures and analyze the signature to verify that the message is authentic, as discussed below. Briefly, the signature verifier 606 may analyze a received signature and message to determine if the signature was generated using the correct key K. It should be noted that, while Figure 6 shows a key source 602 separate from the message submitter 604 and signature verifier 606, either of the message submitter or signature verifier could also be a key source. For example, customers of a computing resource provider may

provide their own keys. Customer keys may then be provided to the signature verifier for verification of signatures. In addition, the message submitter 604 and signature verifier 606 may each receive different keys from the key source 602. For example, the message submitter 604 may receive a key and the signature verifier 606 may receive a key that is derived, using the various embodiments of the present disclosure, from the key received by the message submitter 604.

[0056] As illustrated in Figure 6, the signature verifier 606 receives messages and corresponding signatures from the message submitter 604. The messages may be, for example, electronic requests for access to a computing service 608. The messages may, for instance, encode API calls to a web service. If analysis of the signature and message indicates that the messages are authentic, then the signature verifier notifies the service (or a component controlling access to the service) that the message submitter can have the requested access. For example, the signature verifier may pass the received message to the service to enable the service to fulfill the request. Accordingly, the service may be a computer system operable to fulfill requests, such as the various services described above. It should be noted that, while various descriptions of various components of Figure 6 and other components describe the components as possibly being implemented as computer systems configured to perform certain actions, components may also comprise multiple computing devices, such as networks of computing devices, that are collectively configured to perform the actions.

[0057] Figure 7 is a flowchart showing an illustrative example of a process 700 for signing messages in accordance with an embodiment. Some or all of the process 700 (or any other processes described herein, or variations and/or combinations thereof) may be performed under the control of one or more computer systems configured with executable instructions and may be implemented as code (e.g., executable instructions, one or more computer programs, or one or more applications) executing collectively on one or more processors, by hardware, or combinations thereof. The code may be stored on a computer-readable storage medium, for example, in the form of a computer program comprising a plurality of instructions executable by one or more processors. The computer-readable storage medium may be non-transitory.

[0058] In an embodiment, the process 700 includes obtaining 701 a key K. The key can be obtained in any suitable manner. For example, the key may be generated by a computer system performing the process 700. The key may be electronically received by a computer system

performing the process 700. Generally, obtaining the key may be performed in any suitable manner. The key may be any suitable key for a particular signature algorithm being utilized. For example, if a hash-based message authentication code (HMAC) scheme is being used with a secure hash algorithm (SHA)-256 cryptographic hash function, the key may be a sequence of bytes, such as a sequence of 64 or fewer bytes. Different cryptographic hash functions, such as SHA-224, SHA-384, and SHA-512 may also be used.

[0059] In an embodiment, the process also includes canonicalizing a message M to form a canonicalized message M_c. Canonicalizing a message may include arranging information in the message in a format that allows a verifier to verify whether a signature of the message is valid. Generally, many information communication protocols transform the bits that comprise a message while leaving the message semantically identical. As a result, two semantically identical messages may comprise different sets of bits and, therefore, may result in different signatures. Accordingly, canonicalization allows for a straightforward way of ensuring that a signature can be verified. It should be noted, however, that some embodiments of the present disclosure do not require message canonicalization. For example, if various protocols being utilized do not result in semantically identical messages comprising different sets of bits, canonicalization may not be necessary and may be omitted. Generally, canonicalization may be omitted in any instance where signature verification is able to proceed successfully without manipulation of a signed message.

[0060] In an embodiment, a signature is generated by computing HMAC(K, M_c), where HMAC() is an HMAC function, such as described above. HMAC functions have several properties that make them particularly useful for various embodiments of the present disclosure. For example, HMAC functions can be computed efficiently by a computer system, thereby leaving computing resources available for other tasks. In addition, HMAC functions are preimage resistant (non-invertable). For instance, given a signature S=HMAC(K, M) with K a key and M a message, essentially no information is gained about the key K. For example, from S it would be computationally impossible or at least impractical to determine K from S. HMAC functions are also second preimage resistant. In other words, given S=HMAC(K, M) and M, it is impossible or at least computationally impractical to determine a message M' different from M such that S=HMAC(K,M'). In addition, HMAC functions are forgery-resistant. For instance, given an oracle for S=HMAC(K, M), querying the oracle N times (N a positive integer) allows for the production of at most N signature-message pairs. In other words, given a set of signature-

message pairs, it is impossible or computationally impractical to determine the key or determine a function that will produce a correct signature for a message not in the set.

[0061] While HMAC functions are particularly useful for various embodiments, other functions can be used. For example, any function with the above properties of HMAC functions may be used. In addition, other functions that do not necessarily have all (or any) of the above properties can be used, such as in circumstances where security is not of primary concern and/or where security is a concern, but is maintained through other mechanisms. It should be noted that various illustrations of various embodiments show specific inputs into HMAC functions, but that variations are possible. For example, the inputs to an HMAC function (or other function) may be different. As described above, for instance, one input is a key. However, this input may be derived from a key or otherwise based at least in part on a key. As an illustrative example, input may comprise a key with information, such as a signature scheme identifier (perhaps a version identifier), that is added to the key as a suffix, prefix, or otherwise. As another example, input may be information that is obtained by use of a mapping of the key to the information, which may be another key. Similarly an input shown as a message may be derived from a message. As another example variation considered as being within the scope of the present disclosure, the signature may not be the output of an HMAC function, but one or more values that are derived from the output of a HMAC function (or other suitable function). In some embodiments, the key and the message may be passed into the function in the reverse order.

[0062] Returning to the description of Figure 7, once the signature is generated by computing HMAC(K,Mc), the signature and message M are provided 708 to a receiver, which may be a computing device that verifies signatures or another computing device involved in a signature verification process, such as a computing device providing an interface for communication of messages and signatures. As with all embodiments explicitly described herein, variations are considered as being within the scope of the present disclosure. For example, the canonicalized message M_C may be provided to the receiver instead of or in addition to the message M. In addition, providing the message M and the signature to the receiver may also include providing other information, such as a key identifier that may be used to identify, in a data store that associates keys with key identifiers. Further, other information, such as parameters that encode policy, as discussed below, may be provided with the message M and signature.

[0063] Figure 8 is a flowchart showing an illustrative example of a process 800 for signature verification in accordance with at least one embodiment. The process 800 shown in Figure 8 may be performed by a verifier, such as described in Figure 2. Further, the process 800 may be performed in response to receipt of a signature and a message, such as in response to another computer system having performed the process 700 of Figure 7. In an embodiment, the process 800 includes obtaining 802 a key K, such as described above. Obtaining a key K may also include other actions in various embodiments. For instance, if the process 800 is used by a computer system that verifies signatures generated from multiple keys (such as from multiple customers of a computing resource provider), obtaining the key K may include selecting the key from multiple keys in a data store. The data store may associate various keys with those that submit signatures for verification. For instance, each customer of a computing resource provider may have a key identifier (or several key identifiers) that is used to reference a data store and identify an appropriate key. The key identifier may be submitted in connection with submission of the message and its signature or may be otherwise determined, such as upon submission of login credentials. A recipient of a key identifier (e.g. a message verifier) may reference a data store to determine whether a key corresponding to the key identifier is in the data store and, if not, may then generate the key itself, such as by using the techniques described herein to derive the key directly or indirectly from a shared secret credential. To enable this, the recipient may have access to a key derivation path which, in an embodiment, is information that encodes information necessary to derive the key from the information the recipient already has (e.g. a key derived from a shared secret credential). This information may be provided to the recipient form a submitter of a message with a signature or otherwise may be made available to the recipient. For instance, the recipient may be programmed to automatically generate keys using its assigned region and a code for the current date. Generally, any method of obtaining the key that was used to generate the signature (or another key that can be used to verify the signature, in some embodiments) can be used. The receiver might also enforce policy regarding permissible and impermissible key derivation paths with respect to the request at hand or some other property known to the receiver.

[0064] In an embodiment, a signature S and message M are received 804. The signature S and message M may be received electronically from a submitter, such as a computing device that performed the process 700 of Figure 7. The message M is then canonicalized 806 to determine M_c, in accordance with an embodiment. Canonicalization of the message M, in various

embodiments, ensures that the signature S can be verified. Accordingly, in an embodiment, the process 800 includes generating 808 a signature S' by computing HMAC(K, M_c). In an embodiment, S' is equal to HMAC(K, M_c), although S' can be derived from HMAC(K, Mc), in various embodiments. For the purpose of illustration, the remainder of the process 800 will be described with the assumption that S'=HMAC(K, M_c), but that numerous variations are within the scope of the present disclosure.

[0065] Accordingly, in an embodiment, a determination is made 810 whether S' is equal to the received signature S. In other words, a determination is made whether the received signature is sufficient, for example, because it is a signature that was generated using the key K. Thus, in an embodiment, if it is determined 810 that S' and S are not equal, then the signature is 812 unverified. However, if the S' is equal to S, then the signature is 814 verified. Depending on whether the signature is verified, appropriate action may be taken. For instance, if the message was a request for access to a computing resource, the requested access may be denied (at least temporarily). Similarly, if the message was a request for access to the computing resource and the signature was verified, the requested access may be granted. It should be noted, however, that the appropriate action to be taken can vary widely in various embodiments depending on the reason(s) signatures are received and verified.

[0066] As noted above, various embodiments of the present disclosure apply to numerous environments. In many environments, it is useful to have centralized management of various aspects of security maintenance. Figure 9, for example, is a diagram 900 illustrating an example manner of distributing keys in accordance with at least one embodiment. In figure 9, a central key authority maintains one or more data stores (collectively referred to as a "data store") that contain various keys utilized by an organization. The keys may correspond, for example, to users of computing devices of the organization. Each user of a set of users may, for instance, be assigned one or more keys. In an embodiment, at least some keys correspond to customers (and/or users of the customers) of the organization. For instance, in an embodiment, the organization is a computing resource provider and each customer of the computing resource provider corresponds to one or more keys that allow the customers' users to access computing resources maintained by the computing resource provider. Other adaptations of the process 800 of Figure 8 in accordance with the variations described above with Figure 7 are also within the scope of the present disclosure.

[0067] As illustrated in Figure 9, the key authority 902 propagates keys to a plurality of key zones 904. A key zone may be a domain of the organization in which a received key is valid. For example, referring to Figure 2, each key zone 904 may correspond to a fault zone, such as a data center. Key zones may be, but are not necessarily, geographically defined. For example, each key zone may correspond to a country, region, or other geographically defined region. Key zones may also be defined in other ways. For example, each key zone may correspond to a service provided by a computing resource provider, to a customer of an organization, and the like. While not illustrated as such, key zones may have sub-zones. For example, a key zone may correspond to a country. Inside the country may be multiple regions, each corresponding to sub-zones of the key zone. Keys may be propagated to sub-zones in such embodiments.

[0068] As illustrated in Figure 9, key zones 904 may propagate keys to one or more verifiers 906 for the key zone. For instance, if a key zone corresponds to a data center, a computing device of the data center may propagate keys to verifiers for each of a plurality of services supported by computing resources in the data center. In this manner, the verifiers can be used to verify signatures submitted in connection with various requests. This relieves the computing resources of the key authority themselves from verifying signatures and also reduces latency and bandwidth requirements, especially in instances where the key authority 902 is geographically distant from services to which requests are made.

[0069] Key propagation may be made in various ways. In an embodiment, keys are distributed over secure channels to various recipients. In some embodiments, the key authority propagates the same keys to each key zone. Also, some keys may be usable in multiple key zones. The key authority 902 may propagate keys usable in multiple key zones to those multiple key zones while refraining from propagating those keys to key zones where the keys cannot be used. Thus, in the example of a computing resource provider, the key authority 902 may propagate a key for a customer only to those key zones where the customer is able to use the key, such as data centers used to maintain computing resources of the customer.

[0070] Various embodiments of the present disclosure also provide for key propagation in manners providing for numerous advantages. Figure 10 is a diagram 1000 illustrating an example manner of distributing keys in a manner that provides various scopes of authority in accordance with at least one embodiment. As with Figure 10, the diagram 1000 includes a key authority 1002 with a key K that propagates keys, directly or indirectly, to various key zones

1004 and verifiers 1006, such as in accordance with the above description in connection with Figure 9. While, for the purpose of illustration, the diagram 1000 is described in connection with a single key K, and keys derived from K, the embodiments described herein apply when the key authority performs such actions for numerous keys.

[0071] As illustrated in Figure 10, the key K is used as a basis for other keys derived from K. For example, from K, a key K_1 is derived and propagated to a first key zone (Key $Zone_1$). As such, the key K_1 (or keys derived from the key K_1) is usable in the first key zone, but not in other key zones that do not have K_1 (or a key derived from the key K_1). Similarly, each of a number of other key zones receive corresponding different keys derived from the key K. It should be noted that, while Figure 10 shows keys derived from the key K being propagated from the key authority 1002 to corresponding key zones, variations are possible. For example, the key K may be propagated to the key zones and each key zone that receives the key K may use the key K to derive one or more corresponding keys. For example, the key zone 1004 labeled "Key $Zone_1$" may receive the key K and derive K_1. Generally, various tasks involved in key derivation and propagation may be performed differently than illustrated in various embodiments.

[0072] As shown in the illustrative example of Figure 10, the keys received by the key zones 1004 are used to derive keys that are propagated further. For example, referring to the key zone 1004 labeled "Key $Zone_2$," a key K_2 that is derived from the key K is used to derive additional keys K_2' and K_2''. The keys K_2' and K_2'' are propagated to corresponding verifiers 1006 for use by the verifiers 1006 in verifying signatures. Thus, a verifier that receives K_2' would, in an embodiment, be able to verify a signature generated using K_2', whereas a verifier that did not receive K_2' would not be able to verify the signature. By propagating the keys in the manner illustrated in Figures 9 and 10 (or variations thereof) advantages are achieved. For instance, by propagating the keys to numerous verifiers in multiple locations instead of one or more centralized verifiers, lower latency is achieved. In addition, referring to Figure 10, by propagating derived keys to other devices that, in turn, derive additional keys, it is possible to spread computations over multiple devices over multiple locations, thereby allowing for faster key derivation and increasing fault tolerance.

[0073] Derivations of keys may be performed in numerous ways. Figure 11 is a flowchart showing an illustrative example of a process 1100 of key derivation in accordance with at least one embodiment. In an embodiment, the process 100 includes obtaining 1002 a key K_i, such as

in a manner described above. The key K_i may be any suitable key, such as described above. In addition, the key K_i may be, but is not necessarily, derived from another key, such as by performance of the process 1100 or another process. Upon obtaining the key K_i, a new key is derived from K_i. In the illustrative example of Figure 11, a new key K K_{i+1} is computed as (or based at least in part on) HMAC(K_i, R_{i+1}), where R_{i+1} is information identifying one or more restrictions on the key K_{i+1}. R_{i+1} may be, for instance, a sequence of bits that encodes information indicating where the key K_{i+1} is usable. For example, R_{i+1} may encode a key zone where the key K_{i+1} may be used. Restrictions may be based at least in part on geography, time, user identity, service, and the like. Example restrictions are provided in the description below.

[0074] Further, as discussed more below, the process 1100 may be used multiple times to derive a key. For example, a key generated using the process 1100 (or a variation thereof) may be used to generate another key, using the same or another restriction. Using the terminology in the figure, R_{i+1} may be, for instance, a sequence of bits that encodes information indicating where the key K_{i+1} could be used. K_{i+1} would become the key K_i for a next iteration of the process. For example, if the process 1100 was used to generate a key based on a geographic restriction, the key generated may be used to generate a key with a date-based restriction. Such a process may be utilized multiple times to use multiple restrictions to derive a key. As discussed more fully below, by using multiple restrictions to derive a key, one or more verifiers can enforce policy while verifying signatures. As a brief illustrative example, as part of a signature verification process, a verifier may determine an expected signature using a restriction, such as an encoding of a current date. If a signature was provided that was generated on a different date, then verification of the signature would fail, in accordance with an embodiment. Generally, if use of a signature does not comply with a restriction used to derive a key, signature verification may fail in accordance with various embodiments.

[0075] Figure 12 is a diagram 1200 showing an illustrative example of a derivation of a key using multiple restrictions in accordance with at least one embodiment. In figure 12, a key is derived using multiple restrictions. In this example, a key and a date restriction are used to determine a date key (Kdate, in the figure). In the figure, the date is encoded as 20110715, corresponding to July 15, 2011, although dates may be encoded differently and, generally, information may be encoded differently than illustrated in the figures. The date key is used with a regional restriction to derive a regional key, Kregion. In this example, the region is encoded with a regional identifier "USA-zone-1," which may correspond one of several regions in the

United States. The key Kregion is used with a service restriction to derive a service key, Kservice. In this example, the service is a virtual computer system service, encoded by its acronym VCS. The key Kservice is used with a request identifier to derive a signing key, that is, a key used to sign requests to a service. In this example, the request identifier is "vcs_request" which may correspond to a particular type of request that may be submitted to the VCS service. For instance, "vcs_request" may correspond to a request to provision, stop, or otherwise modify a virtual computer system. The signing key is used to generate a signature that may be submitted with requests. The signature may be generated in any suitable manner, such as described above.

[0076] As illustrated in Figure 12, the request may be canonicalized to form a message, M_c, which is as input into an HMAC function to generate the signature. Of course, variations, including variations where canonicalization is not necessary and where functions other than HMAC functions are used, may be utilized in accordance with various embodiments. In addition, Figure 12 shows a particular example derivation of a signature in accordance with an embodiment. However, more or fewer restrictions may be used in deriving the signature and restrictions may be used in a different order than illustrated. In addition, while Figure 12 shows derivation of a signature, the techniques may be applied to derive other objects that may not be considered signatures in all applications. For example, the techniques illustrated in Figure 12 (and elsewhere) may be used generally to derive keys.

[0077] Figure 13 is an illustrative example of a function 1300 for deriving a signature, in accordance with at least one embodiment. As illustrated in Figure 13, the signature is calculated as:

HMAC(HMAC(HMAC(HMAC(HMAC(K, date), region), service), protocol), Mc).

In this example, K is a key, "date" is an encoding of a date, "region" is an encoding of an identifier of a region, "service" is an encoding of an identifier of a service, "protocol" corresponds to a particular message encoding protocol, and M_c is a canonicalized message. Thus, as illustrated in Figure 13, the signature is computed by computing the same HMAC function multiple times, each time with a different restriction as an input to the HMAC function. The signing key, in this example, is:

HMAC(HMAC(HMAC(HMAC(K, date), region), service), protocol)

which itself is derived by use of the HMAC function multiple times, each time with a different restriction.

[0078] In the example of Figure 13, the various restrictions each define a domain and the intersection of the defined domains defines the manner in which the signature generated with the signing key would be valid. In this specific example, a signature generated with the signing key illustrated in Figure 13 would be valid on the date specified, in the region specified, and for the service specified using the protocol specified. Thus, if a request is signed using the signing key, but on a different date than specified by the input to the signing key, the signature to the request may be considered unverified, even if the request was made for the specified service and in the specified region.

[0079] As with other embodiments described herein, variations are considered as being within the scope of the present disclosure. For instance, Figure 13 shows repeated use of an HMAC function. Multiple functions may be used to derive a signature and, in some embodiments, HMAC functions are not used in every part of the derivation. Also, as noted, different restrictions and different numbers of restrictions may also be used in various embodiments.

[0080] Key derivation may be performed in numerous ways in accordance with various embodiments. For instance, a single computing device could compute a signing key, in accordance with some embodiments. In accordance with other embodiments, multiple computing devices may collectively compute a signing key. As a specific illustrative example, referring to Figure 13, one computer may compute

Kregion=HMAC(HMAC(K, date), region)

and another computer may compute

Signing Key=HMAC(Kregion, Service).

[0081] As another example, a separate computer system may perform a different layer in the computation of the signing key. Referring to the example in the previous paragraph, instead of a single computer computing Kregion, one computer may compute

Kdate=HMAC(K, date)

and another computer may compute

Kregion=HMAC(Kdate, region).

Figure 14 is an illustrative example of how multiple key derivation may be performed and used in accordance with at least one embodiment. In particular, Figure 14 shows an example diagram 1500 illustrating members of a distributed set of computer systems collectively computing a signing key (or other key, in various other embodiments). As shown in Figure 14, each member of the set is a key provider computer system 1402 that generates a key and provides the generated key to another computer system. For example, a key provider labeled Key Provider$_1$ obtains a key K (from another source, or by generating the key itself), and uses the key and a restriction, labeled R_1 to generate a key K_1. Key Provider$_1$ passes the key K_1 to Key Provider$_2$, which uses K_2 and another restriction, R_2, to generate another key K_2. Key Provider$_2$ passes the key K_2 to Key Provider$_3$, which uses K_3 and another restriction, R_3, to generate another key K_3. Depending on how many key providers there are in a particular embodiment, this process may continue until Key Provider$_{N-1}$ passes a key K_{N-1} to Key Provider$_N$, which uses K_{N-1} and another restriction, R_N, to generate another a signing Key, K_N. The key K_N is then passed to a verifier computer system 1404. The key K or any key(s) derived from K (generally referred to as K_i in the figure) may also be passed to a signer computer system 1406, such as through a secure key-exchange algorithm.

[0082] The signer computer system 1406 may also, in various embodiments, generate K_N on its own if, for example, the restrictions R_1-R_N are made available to the signer and/or made publicly available. In addition, the signer computer system 1406 may perform only part of the process for deriving K_N on its own in various embodiments. For instance, the signer may obtain (perhaps from an appropriate key provider computer system) K_i, for some integer i that is less than N and restrictions R_{i+1} through R_N. The signer may then use K_i and restrictions R_{i+1} through R_N to generate the signing key, K_N. Other variations are also considered as being within the scope of the present disclosure.

[0083] The signer computer system 1406 may use the key K_N to sign messages to be verified by the verifier 1404. For instance, as illustrated, the signer 1406 computes the signature $S=HMAC(K_N, M_C)$, where M_C is a canonicalized version of a message M, also sent to the verifier. Because the verifier has K_N, the verifier can independently canonicalize the message M and compute $HMAC(K_N, M_C)$ to determine if the result of the computation matches the received signature S.

[0084] It should be noted that variations of the process illustrated in Figure 14, and other processes described herein, while shown as involving multiple use of HMAC functions, multiple different functions may be used to derive keys. For instance, different types of message authentication code (MAC) functions may be used at different times in deriving a key. For example, the output of one type of MAC function may be used as the basis for input into another type of MAC function. Generally, other types of functions may be used instead of and/or in addition to HMAC functions in a key derivation process and, in various embodiments, it is not necessary to use the same type of function multiple times to derive a key, but different functions may be used each time a function is required.

[0085] Figure 15 is a diagram 1500 illustrating an example manner in which keys may be derived using multiple restrictions, in accordance with at least one embodiment. The example shown in Figure 15 refers to customers, such as customers of a computing resource provider. However, as noted, the techniques described herein, including techniques described in connection with Figure 15, may be used in numerous other contexts.

[0086] As shown, a customer key, K_{cust}, is part of a set of customer long term keys, each of which may be keys used by a customer for a period of time, such as until the customer updates the key, is assigned a new key, or otherwise changes the key. The keys may also be used indefinitely by one or more customers. The customer key, K_{cust}, is used to derive one or more region keys, such as in a manner illustrated above. For example, as illustrated in Figure 15, two region keys may be generated, such as by computing HMAC(K_{cust}, USA-E-1) and HMAC(K_{cust}, USA-N-1), where USA-E-1 and USA-N-1 are identifiers of respective regions. Similarly, the region keys may be used to derive date keys whose validity may be restricted by the date used to encode the date keys. Each of the date keys may be used to derive service keys, such as in a manner described above.

[0087] In this manner, in various embodiments, the service keys may be used with respective services only on the date and in the regions used to encode the keys. New date keys may be generated for each day, whereas region keys and customer long term keys may be generated less often. Multiple restriction key derivation such as illustrated in Figure 15 and elsewhere in the present disclosure provides numerous advantages. For example, by deriving the key in the manner described in connection with Figure 15 if a signing key is compromised (e.g. maliciously obtained by a third party), the security breach is limited to a particular region, on a particular

day, and in connection with a particular service. Other services would remain unaffected. Similar advantages are applicable in other ways that keys may be derived.

[0088] Figure 16, for example, is a diagram 1600 illustrating another example manner in which keys may be derived, in accordance with at least one embodiment. Figure 16 illustrates concepts in a manner similar to that of Figure 16. In Figure 16, however, customer long term keys are used to derive date keys. The date keys are used to derive region keys. The region keys are used to derive service keys. Derivation may be accomplished in accordance with the various embodiments described herein.

[0089] Figure 17 is a diagram 1700 illustrating yet another example manner in which keys may be derived, in accordance with at least one embodiment. In Figure 17, customer long term keys are used to derive month keys. The month keys are used to derive regional keys. The regional keys are used to derive date keys. The date keys are used to define service keys. The derivation of the various keys may be done in a manner consistent with the above description.

[0090] As discussed, various techniques of the present disclosure allow for a novel way of generating sessions. A session may be a period of time for which a set of one or more actions are allowed, where expiration (or other termination) of the session causes the set of one or more actions to be disallowed. Figure 18 is a flowchart showing an illustrative example of a process 1800 for initiating a session, in accordance with at least one embodiment. The process 1800 may be performed by any suitable computing device or collectively by any suitable collection of computing devices. For example, the process 1800 may be performed by a client device of a customer of a computing resources provider. As another example, in another embodiment, referring to Figure 3, one of the services of a fault zone may be a session service and one or more computing devices participating in providing the service may perform the process 1800.

[0091] Returning to Figure 18, in an embodiment, the process 1800 includes obtaining 1802 a key, K. The key K may be any suitable key, such as a key derived using other keys, such as in a manner described above. For example, the key K may have been propagated to a computing device participating in performance of the process 1800. At some point (such as upon obtaining the key K, as illustrated in the Figure), in an embodiment, a request to initiate a session may be received 1804. The request may be an electronic request, such as described above. In addition, the request, in an embodiment, is signed and verified using various techniques of the present disclosure. Also, the request may be a different request depending on a particular environment

used to implement the process 1800. For example, if the process 1800 is performed by a client device (such as a customer device of a customer of a computing resources provider) to generate a session, the request to initiate the session may be received by a module of the client device.

[0092] In an embodiment, session parameters for the session are determined 1806. The session parameters may be information that indicates one or more restrictions on the session being generated. Example parameters include, but are not limited to, duration, identifiers of acceptable users of a session key to be generated, one or more services with which the session key to be generated is usable, restrictions on actions that may be performed using the session key, any of the restrictions described above, and others. The parameters may be encoded electronically according to predefined formatting requirements to ensure that computations involving a session key to be generated are consistent. For example, dates may be required to be encoded in the format YYYYMMDD. Other parameters may have their own formatting requirements. In addition, determining the session parameters may be performed in various ways. For example, the parameters may be default parameters for a session, such that a session key is only usable for a range of actions permitted to the requestor of the session initiation and for a predefined period of time (such as a twenty-four hour period). As another example, parameters may be provided as part of or otherwise in connection with the received request. For example, the parameters may be generated according to user input from the requestor and encoded according to a predefined schema.

[0093] In an embodiment, once the parameters are determined, the parameters are used to compute 1808 a session key, K_S. Computing the session key K_S may be performed in numerous ways. For example, in one embodiment, the session key K_S may be computed as (or otherwise based at least in part on)

HMAC(K, Session_Parameters)

where Session_Parameters is an encoding of the parameters that were determined 1806. Session_Parameters may be encoded in a predefined manner that ensures for computational consistency. The session key K_S may also be computed in other ways, such as in a manner described below in connection with Figure 19.

[0094] Once the session key K_S is computed 1808, in an embodiment, the session key K_S is provided for use. Providing the session key may be performed in numerous ways in various embodiments. For example, the session key may be provided to a module of the requestor to

enable the requestor to sign messages with the session key. The session key may also be provided over a network to another device to enable the other device to sign messages with the session key. For example, the session key may also be provided to a delagatee for which the session is initiated. For instance, the requestor may have specified a delegatee in or otherwise in connection with the request to initiate the session. The session key may be provided electronically according to information provided by the requestor (i.e. delegator), such as an electronic mail or other electronic address.

[0095] As noted, Figure 19 shows an illustrative example of a process 1900 that may be used to generate a signature, in accordance with an embodiment. The process 1900 may be performed by one or more computing devices, such as one or more computing devices performing the process 1800 described above in connection with Figure 18. The process 1900, as illustrated in Figure 19, includes receiving session parameters, such as described above. With the session parameters having been obtained, in an embodiment, an intermediate key, K_{i+1} is computed 1904 as:

$$K_{i+1}=HMAC(K_i, P_i),$$

where K_i may be the key K in the description of Figure 18 for the first computation of K_{i+1}, and P_i is the i^{th} parameter of the session parameters. The session parameters may be ordered according to a predetermined ordering to ensure computational consistency of the key signature.

[0096] In an embodiment, a determination is made 1906 whether there are additional parameters to be used in generating the session key. If there are additional parameters, in an embodiment, the index i is increased 1908 by one and K_{i+1} is again computed 1904. If, however, it is determined that there are no additional parameters, then K_S is set 1910 to the value of K_{i+1}.

[0097] Figure 20 is a flowchart showing an illustrative example of a process 2000 for obtaining access to one or more computing resources during a session in accordance with at least one embodiment. It should be noted that, while Figure 20 presents a process 2000 for obtaining access to one or more computing resources, as with other processes described herein, the process 2000 may be modified for any situation where signature processes are used. The process 2000 may be performed by a computer system of a user requesting access to one or more computing resources, such as a client computer system illustrated in Figure 1 and/or a customer computer system described elsewhere herein. In an embodiment, the process 2000 includes obtaining a session key K_S. The session key may be obtained in any suitable manner, such as in an

electronic message. The session key may be obtained from a computer system of a delegator of access to the one or more computing resources or another computer system, such as a computer system operating in connection with the one or more computer systems that performed a process for generating K_S.

[0098] In an embodiment, a request R is generated 2004. The request R may be a message, such as described above. The request R is then canonicalized 2006, in an embodiment, and a signature is computed 2008 from the canonicalized message, such as by computing the signature as (or otherwise based at least in part on) HMAC(K_S, R_C). Upon generation of the signature, the signature S and the request R are provided 2010. For example, as discussed above, the signature S and request R may be provided electronically to an interface of a computer system that participates in managing requests and verifying signatures. The signature S and request R, as with signatures and messages in general, may be provided together in a single communication, in separate communications, or collectively by multiple communications. Other information may also be provided in connection with the signature S and request R. For instance, identification information may be provided to enable a verifier to select a proper key for generating a signature with which to verify the received signature. The identification may be, for instance, an identifier of a key that should be used in generating a signature for comparison. Other information may also be provided and used, as appropriate in the various embodiments.

[0099] Figure 21 is a flowchart showing an illustrative example of a process 2100 for determining whether to grant requested access to one or more computing resources in accordance with at least one embodiment. As illustrated in Figure 12, the process 2100 includes obtaining 2102 a signing key K_S. As with other recitations herein of obtaining a signing key, the signing key can be obtained in various ways, such as by receiving the signing key from another source, retrieving the signing key from memory, computing the signing key from information available, and the like.

[0100] In an embodiment, the received request R is canonicalized to form R_C, such as in a manner described above. It should be noted that, as with other processes described herein, variations are possible. For example, a computer system performing a variation of the process 2100 (or another process) may simply receive the canonicalized message and canonicalization may be performed by another computing device. Returning to the description of Figure 21, a signature S' is computed as (or otherwise based at least in part on) HMAC(K_S, R_C). The

computed signing key S' is compared 2110 with the received signature S to determine whether the two signatures are equivalent. If the two signatures are determined to not be equivalent, the session is determined 2112 to be unvalidated and appropriate action, such as denial of the request, may be taken. If the two signatures are determined to be equivalent, the session is validated 2114 and appropriate action, such as granting access to one or more computing resources, may be taken.

[0101] Techniques of the present disclosure, as mentioned, may be used to allow delegation of authority. Figure 22 is a flowchart showing an illustrative example of a process 2200 for delegating authority in accordance with at least one embodiment. The process 2200 may be performed by a computing device, such as a computing device of a user attempting to delegate access to one or more computing resources, or a computing device of a computing resource provider, or any suitable computing device. As illustrated in the figure, the process 2200 includes obtaining 2202 a session key K_{si}. The obtained session key K_{si} may be obtained in any suitable way, such as a manner in which keys described above are described as being obtained. Further, the session key may be a key that was generated as part of a process to delegate access to one or more computing resources. For example, the session key may have been generated by performing the process 2200, or a variation thereof.

[0102] In an embodiment, session parameters are determined 2004. The session parameters may be determined in any suitable way, such as described above in connection with Figure 18. With the session parameters determined 2004, a new session key $K_{S(i+1)}$ may be generated, such as described above, including as described above in connection with Figure 19. Once generated, the new session key may be provided to a delegatee. For example, the session key may be sent in an electronic message to the delegatee. The session key may be provided directly or indirectly to the delegatee. For example, the session key may be given to the delegator and the delegator may be responsible for providing the session key to one or more delegatees. Other information may also be provided to the delegatee. For instance, the session parameters may be provided to the delegatee to enable the delegatee to provide the session parameters with signatures, thereby enabling a recipient (e.g. verifier) of the session parameters to generate expected signatures to verify if the provided signatures are valid. For instance, the recipient may use the parameters to generate a session key from a secret credential or a key derived therefrom and use that session key to generate a signature for a canonicalized version of a corresponding signed message. Generally, the parameters may be made available to the recipient of a signature in any suitable

manner to enable the recipient to verify message signatures and the delegatee does not necessarily need access to the parameters if the recipient has access to the parameters independent of the delegatee.

[0103] Figure 23, for example, shows a diagram 2300 illustrating how privileges may be delegated multiple times. A delegator 2302 may wish to grant one or more access privileges to a delegatee 2304. The delegatee 2304, however, in this example, may wish to provide one or more privileges to another delegatee 2306. Thus, in this example, the delegatee 2304 may become a delegator. Similarly, the delegatee 2306 may wish to provide access to another delegatee and that delegatee may wish to grant access to another delegatee and so on until finally one or more privileges are granted to yet another delegatee 2308.

[0104] Thus, in this example, the original delegator 2302 submits a delegation request to a session-based authentication service 2310 which may be a service of a fault zone, as described above. In response, in an embodiment, the session-based authentication service generates and provides a session key to the delegator 2302, such as described above in connection with Figure 22. The delegator 2302 then, in an embodiment, provides the session key that it received from the session-based authentication service 2310 to the delegatee 2304. The delegatee 2304 may provide the session key to another delegatee 2306. In this manner the delegatee 2306 would receive the scope of privileges received by the delegatee 2304 that would be the same as the scope of the privileges provided to the delegatee 2306.

[0105] However, also illustrated in Figure 23, the delegatee 2304 may submit a delegation request to the session-based authentication service 2310 and receive a different session key that had been generated by the session-based authentication service 2310 in response to the delegation request. The delegatee 2304 may provide this new session key to the next delegatee 2306. The next delegatee 2306 may provide the session key to yet another delegatee, or as described above may also submit a delegation request to the session-based authentication service 2310 which would then generate a session key and provide the session key to the delegatee 2306 that submitted the delegation request. As indicated in Figure 23, this may continue and one or more of the delegatees may attempt to use a session key that he or she has received.

[0106] In this particular example, a delegatee 2308 provides the session key to a computing resource 2312 in connection with a request. As above, the request may include the session key although the session key may be provided separately from the request. The computing resource

2312 may be any of the computing resources described above or, generally, any computing resource. A policy management service 2314 may include a verifier, such as described above, and may, upon request of the computing resource, validate requests. The computing resource 2312 and policy management service 2314 may also be a single component, although illustrated separately in Figure 23. Further, while Figure 23 shows a single session-based authentication service 2310 being used to generate session keys, various embodiments may utilize different session-based authentication services.

[0107] As noted above, numerous variations in addition to the illustrative examples provided herein are considered as being within the scope of the present disclosure. Figure 24 shows a diagram 2400 representing an illustrative example of a manner in which keys may be derived using keys from multiple authorities, in accordance with an embodiment. In Figure 23, a customer key, K_{cust}, is from a set of customer keys maintained by a computing resource provider. As with embodiments described above, while Figure 23 discussed an illustrative example in connection with a computing resource provider, other variations are considered as being within the scope of the present disclosure.

[0108] In Figure 24, a set of authority keys are maintained, where each authority key corresponds to a different domain of authority. Each authority key derived from the customer key K_{cust} may be, for example, propagated to different fault zones, such as described above. The fault zones may be, for instance, data centers in different political jurisdictions. It should be noted, however, that while Figure 24 shows each divided authority key having been derived from a single customer key, K_{cust}, variations are possible. For example, the divided authority keys may be independently derived. As another example, one or more divided authority keys may be derived from a common key, one or more others may be derived from another common key, and the like.

[0109] In an embodiment, multiple authorities are able to combine authority to allow access to one or more computing resources. For example, as illustrated in Figure 24, subsets of divided authority keys may be used to derive other keys. For example, as illustrated in Figure 23, two authority keys, labeled Auth1 and Auth2, are used to derive a merged authority key. To derive the merged authority key, in an embodiment, a value of HMAC(f(Auth1, Auth2), R) is computed, where R is some restriction, such as described above. In this example, f is a function of divided authority keys, and may be more than two-dimensional. For instance, the three

divided authority keys, Auth1, Auth2, and Auth3 are used, as illustrated in Figure 23, in a function f(Auth1, Auth2, Auth3) to compute the merged authority key as (or otherwise based at least in part on) HMAC(f(Auth1, Auth2, Auth3), R).

[0110] Numerous variations of constructing keys from different authorities are considered as being within the scope of the present disclosure. For example, an authority may generate (or have generated) a key (K_{spec}) using various embodiments of the present disclosure. Each authority K_{spec} may correspond to a partial key seed, which may be a publicly available encoding (or encoding otherwise available to a message signor and signature verifier) of restrictions used to generate its K_{spec}. For example, a partial key seed may be `(K1/20110810/usa-east-1/DDS,K2/20110810/org_name/jpl/DDS)`, where each string between slashes is a restriction. Such an encoding of information may be referred to as a key path. As a more general example, a partial key seed may be $X_1/\ldots/X_n$, where each X_i (for i between 1 and n) corresponds to a parameter, such as a parameter described above. The partial key seeds from the applicable authorities may be encoded as an n-tuple, referred to as a key seed. An n-tuple for the example immediately above may be ($spec_1$, $spec_2$,…,$spec_n$), where each entry is a key path for a corresponding K_{spec}. It should be noted that the key seed (and/or key path) encodes the precise key use (full restriction among all authorized keys) that the key holder is authorizing by producing a signature/key. Further, with partial key seeds available to both message signors and signature verifiers, arbitrary ordering of the parameters used to generate keys and signatures is possible since, for example, a message signor has information that specifies the order the parameters were used to generate a signing key and can, therefore, generate the signing key and message accordingly.

[0111] A value for HMAC(Kspec, key-seed) may then be obtained or computed for each of the applicable authorities, that is, the authorities for which a key is to be generated. This value may be computed by a client obtaining a signing key to sign messages or may be computed by another device and subsequently provided to the client, in various embodiments. Each of these values may be referred to as partial keys, for the purpose of the following discussion. The semantics of each of these partial keys, in an embodiment, are that they are valid only when combined with the below construction (and certain variations of the below construction) and, when combined, form the intersection of specializations encoded in the key seeds.

[0112] To generate a signing key to sign a message, a value for

$$K_S = HMAC(partial_key_1+...+partial_key_n, key\text{-}seed)$$

where "+" may refer to some associative operation on partial keys that surround the symbol in the formula. The "+" symbol may be, for example, an exclusive OR (XOR) operation on bits comprising the partial keys. The "+" symbol may also refer to some other suitable operation or function.

[0113] To verify a signature used to sign a message, a verifier may obtain each partial key, combine the partial keys as above to form a signing key, sign a received message and compare the result with an expected result to verify the signature, such as discussed above.

[0114] Example embodiments of the disclosure can be described in view of the following clauses:

Clause 1. A computer-implemented method for providing services, comprising:

under the control of one or more computer systems configured with executable instructions,

receiving, from an authenticating party, electronic information encoding a message, a signature for the message, and a set of one or more restrictions on keys derived from a secret credential shared with the authenticating party, the signature being determinable by applying a hash-based message authentication code function to the message, the secret credential, and the set of one or more restrictions, but also being undeterminable having only the hash-based message authentication code function but without having the set of one or more restrictions;

obtaining a key generated at least in part using at least a subset of the set of one or more restrictions;

calculating, by the one or more computer systems, a value of a hash-based message authentication code function by at least inputting into the hash-based message authentication code function:

first input based at least in part on the obtained key; and

second input based at least in part on the set of one or more restrictions;

determining, by the one or more computer systems and based at least in part on the calculated value, whether the signature is valid; and

providing access to one or more computing resources when determined that the signature is valid.

Clause 2. The computer-implemented method of clause 1, wherein:

the message comprises a request for access to the one or more computing resources;

the method further comprises determining whether the set of one or more restrictions indicate that the request should be fulfilled; and

providing access to the one or more computing resources is contingent on determining that restrictions indicate that the request should be fulfilled.

Clause 3. The computer-implemented method of claim 2, wherein the information encoding the set of one or more restrictions is encoded by a document and wherein determining whether the set of restrictions indicate that the request should be fulfilled includes evaluating the document against a context in which the request was received.

Clause 4. The computer-implemented method of clause 1, wherein:

the message comprises a request to access a computing resource of the one or more computing resources;

the information encoding the set of one or more restrictions includes information specifying the computing resource; and

providing access to the one or more computing resources includes providing access to the computing resource when the computing resource matches the specified computing resource.

Clause 5. The computer-implemented method of clause 1, wherein:

the information encoding the set of one or more restrictions corresponds to a time period for which the message is valid; and

determining whether the signature is valid is based at least in part on whether the message was submitted during the corresponding time period.

Clause 6. The computer-implemented method of clause 1, wherein:

the information encoding the set of one or more restrictions corresponds to a restriction based at least in part on a location; and

determining whether the signature is valid is based at least in part on whether a location of at least one of the one or more computer systems matches the corresponding location.

Clause 7. A computer-implemented method for providing services, comprising:

under the control of one or more computer systems configured with executable instructions,

obtaining electronic information encoding (i) a message, (ii) a first signature for the message, and (iii) a set of one or more parameters, the first signature having been generated based at least in part on (i) the message, (ii) a secret credential, and (iii) the set of one or more parameters, the first signature further being undeterminable having only the message and the secret credential but without the set of one or more parameters;

deriving a second credential based at least in part on the secret credential and at least a subset of the set of one or more parameters;

generating, based at least in part on the derived second credential, a second signature;

determining whether the first signature matches the second signature; and

providing access to one or more computing resources when the generated second signature matches the first signature.

Clause 8. The computer-implemented method of clause 7, wherein deriving the second credential includes inputting, into a function, the secret credential and the at least a subset of the set of one or more parameters.

Clause 9. The computer-implemented method of clause 8, wherein the function is a symmetric message authentication function.

Clause 10. The computer-implemented method of clause 9, wherein the symmetric message authentication function is a hash function.

Clause 11. The computer-implemented method of clause 9, wherein inputting, into the function, the secret credential and the at least a subset of the one or more parameters is performed as part of Hash-Based Message Authentication Code (HMAC).

Clause 12. The computer-implemented method of clause 8, wherein generating the second signature includes inputting to the function both an output of the function and a parameter from the set of one or more parameters.

Clause 13. The computer-implemented method of clause 7, wherein the information encoding the one or more parameters comprises an electronic document encoding the set of one or more parameters.

Clause 14. The computer-implemented method of clause 8, wherein:

generating the second signature is based at least in part on a key;

the set of one or more parameters include one or more restrictions on use of the key; and

providing access to the one or more computing resources is performed in accordance with the one or more restrictions.

Clause 15. The computer-implemented method of clause 14, wherein the key is based at least in part on a result of inputting the secret credential into a function.

Clause 16. A non-transitory computer-readable storage medium having stored thereon instructions that, when executed by a computer system, cause the computer system to at least:

obtain an intermediate key that is derived from at least a secret credential and one or more parameters for use of the intermediate key;

apply, based at least in part on the obtained intermediate key, at least a portion of a signature generation process that results in a signature for a message, the signature generation process configured such that the signature is undeterminable, by the signature generation process, to a computing device having the message, the secret credential, and the signature but lacking the one or more restrictions; and

provide the message, the signature, and the one or more parameters to another computer system that is configured to analyze, based at least in part on the one or more parameters and the message, the signature to determine whether the signature is valid.

Clause 17. The non-transitory computer-readable storage medium of clause 16, wherein the one or more parameters encode one or more restrictions on use of the intermediate key that are enforced, at least in part, by said another computer system.

Clause 18. The non-transitory computer-readable storage medium of clause 16, wherein the one or more restrictions correspond to at least one of a time period within which the intermediate key is usable, a location in which the intermediate key is usable, and one or more services for which the intermediate key is usable to gain access.

Clause 19. The non-transitory computer-readable storage medium of clause 16, wherein the instructions, when executed by the computer system, enable the computer system to generate the signature without the computer system having access to the secret credential.

Clause 20. The non-transitory computer-readable storage medium of clause 19, wherein, having the set of one or more parameters, the signature is determinable by the signature generation process using either the shared secret credential or the intermediate key.

Clause 21. The non-transitory computer-readable storage medium of clause 19, wherein obtaining the intermediate key includes performing an algorithm wherein at least one output of a hash function is input, with at least one of the parameters, into the hash function.

Clause 22. A computer system, comprising:

one or more processors; and

memory including instructions that, when executed by one or more processors of a computer system, cause the computer system to at least:

receive one or more electronic communications that collectively encode a message, a signature for the message, and one or more parameters, the signature being generated based at least in part on the secret credential and the one or more parameters;

analyze, based at least in part on the one or more parameters, an intermediate credential derived from at least a portion of the one or more parameters and the secret credential, but without the secret credential, the message and signature to determine whether the signature is valid; and

take one or more actions contingent on determining that the signature is valid.

Clause 23. The computer system of clause 22, wherein:

the memory and one or more processors are part of a first server system in a first geographic location;

the computer system comprises a second server system in a second geographic location, the second server system being configured to generate, based at least in part on the secret credential, a different signature;

the first server system and second server system both lack the secret credential;

analyzing the message and signature includes inputting into a function the at least a portion of the one or more parameters and the intermediate credential; and

the first server system and second server system each lack information from which a same signature could be generated, using the function, from the message.

Clause 24. The computer system of clause 22, wherein:

the computer system corresponds to a service; and

the one or more actions include providing access to the service.

Clause 25. The computer system of clause 24, wherein the one or more parameters limit use of the intermediate credential to use in accessing the service.

Clause 26. The computer system of clause 22, wherein:

analyzing the message and signature includes applying a hash function to the intermediate credential;

the one or more parameters include multiple restrictions on use of the intermediate credential; and

wherein the computer system is configured to enforce the restrictions.

Clause 27. The computer system of clause 22, wherein:

analyzing the message and signature includes applying a hash function to a key that is derived from the secret credential; and

the instructions, when executed by the one or more processors of the computer system, cause the computer system to further receive the derived key from a key authority computer system.

Clause 28. The computer system of clause 27, wherein the instructions that cause the computer system to further receive the derived key from the key authority computer system cause the computer system to receive the derived key from the key authority computer system prior to receipt of the message.

Clause 29. The computer system of clause 22, wherein the intermediate credential is determined by another computer system different from the computer system.

[0115] The various embodiments further can be implemented in a wide variety of operating environments, which in some cases can include one or more user computers, computing devices, or processing devices which can be used to operate any of a number of applications. User or client devices can include any of a number of general purpose personal computers, such as desktop or laptop computers running a standard operating system, as well as cellular, wireless, and handheld devices running mobile software and capable of supporting a number of networking and messaging protocols. Such a system also can include a number of workstations running any of a variety of commercially-available operating systems and other known applications for purposes such as development and database management. These devices also can include other electronic devices, such as dummy terminals, thin-clients, gaming systems, and other devices capable of communicating via a network.

[0116] Most embodiments utilize at least one network that would be familiar to those skilled in the art for supporting communications using any of a variety of commercially-available protocols, such as TCP/IP, OSI, FTP, UPnP, NFS, CIFS, and AppleTalk. The network can be, for example, a local area network, a wide-area network, a virtual private network, the Internet, an intranet, an extranet, a public switched telephone network, an infrared network, a wireless network, and any combination thereof.

[0117] In embodiments utilizing a Web server, the Web server can run any of a variety of server or mid-tier applications, including HTTP servers, FTP servers, CGI servers, data servers, Java servers, and business application servers. The server(s) also may be capable of executing programs or scripts in response requests from user devices, such as by executing one or more Web applications that may be implemented as one or more scripts or programs written in any programming language, such as Java®, C, C# or C++, or any scripting language, such as Perl, Python, or TCL, as well as combinations thereof. The server(s) may also include database servers, including without limitation those commercially available from Oracle®, Microsoft®, Sybase®, and IBM®.

[0118] The environment can include a variety of data stores and other memory and storage media as discussed above. These can reside in a variety of locations, such as on a storage medium local to (and/or resident in) one or more of the computers or remote from any or all of the computers across the network. In a particular set of embodiments, the information may reside in a storage-area network ("SAN") familiar to those skilled in the art. Similarly, any

necessary files for performing the functions attributed to the computers, servers, or other network devices may be stored locally and/or remotely, as appropriate. Where a system includes computerized devices, each such device can include hardware elements that may be electrically coupled via a bus, the elements including, for example, at least one central processing unit (CPU), at least one input device (*e.g.*, a mouse, keyboard, controller, touch screen, or keypad), and at least one output device (*e.g.*, a display device, printer, or speaker). Such a system may also include one or more storage devices, such as disk drives, optical storage devices, and solid-state storage devices such as random access memory ("RAM") or read-only memory ("ROM"), as well as removable media devices, memory cards, flash cards, etc.

[0119] Such devices also can include a computer-readable storage media reader, a communications device (*e.g.*, a modem, a network card (wireless or wired), an infrared communication device, etc.), and working memory as described above. The computer-readable storage media reader can be connected with, or configured to receive, a computer-readable storage medium, representing remote, local, fixed, and/or removable storage devices as well as storage media for temporarily and/or more permanently containing, storing, transmitting, and retrieving computer-readable information. The system and various devices also typically will include a number of software applications, modules, services, or other elements located within at least one working memory device, including an operating system and application programs, such as a client application or Web browser. It should be appreciated that alternate embodiments may have numerous variations from that described above. For example, customized hardware might also be used and/or particular elements might be implemented in hardware, software (including portable software, such as applets), or both. Further, connection to other computing devices such as network input/output devices may be employed.

[0120] Storage media and computer readable media for containing code, or portions of code, can include any appropriate media known or used in the art, including storage media and communication media, such as but not limited to volatile and non-volatile, removable and non-removable media implemented in any method or technology for storage and/or transmission of information such as computer readable instructions, data structures, program modules, or other data, including RAM, ROM, EEPROM, flash memory or other memory technology, CD-ROM, digital versatile disk (DVD) or other optical storage, magnetic cassettes, magnetic tape, magnetic disk storage or other magnetic storage devices, or any other medium which can be used to store the desired information and which can be accessed by the a system device. Based on the

disclosure and teachings provided herein, a person of ordinary skill in the art will appreciate other ways and/or methods to implement the various embodiments.

[0121] The specification and drawings are, accordingly, to be regarded in an illustrative rather than a restrictive sense. It will, however, be evident that various modifications and changes may be made thereunto without departing from the broader spirit and scope of the invention as set forth in the claims.

[0122] Other variations are within the spirit of the present disclosure. Thus, while the disclosed techniques are susceptible to various modifications and alternative constructions, certain illustrated embodiments thereof are shown in the drawings and have been described above in detail. It should be understood, however, that there is no intention to limit the invention to the specific form or forms disclosed, but on the contrary, the intention is to cover all modifications, alternative constructions, and equivalents falling within the spirit and scope of the invention, as defined in the appended claims.

[0123] The use of the terms "a" and "an" and "the" and similar referents in the context of describing the disclosed embodiments (especially in the context of the following claims) are to be construed to cover both the singular and the plural, unless otherwise indicated herein or clearly contradicted by context. The terms "comprising," "having," "including," and "containing" are to be construed as open-ended terms (i.e., meaning "including, but not limited to,") unless otherwise noted. The term "connected" is to be construed as partly or wholly contained within, attached to, or joined together, even if there is something intervening. Recitation of ranges of values herein are merely intended to serve as a shorthand method of referring individually to each separate value falling within the range, unless otherwise indicated herein, and each separate value is incorporated into the specification as if it were individually recited herein. All methods described herein can be performed in any suitable order unless otherwise indicated herein or otherwise clearly contradicted by context. The use of any and all examples, or exemplary language (*e.g.*, "such as") provided herein, is intended merely to better illuminate embodiments of the invention and does not pose a limitation on the scope of the invention unless otherwise claimed. No language in the specification should be construed as indicating any non-claimed element as essential to the practice of the invention.

[0124] Preferred embodiments of this disclosure are described herein, including the best mode known to the inventors for carrying out the invention. Variations of those preferred

embodiments may become apparent to those of ordinary skill in the art upon reading the foregoing description. The inventors expect skilled artisans to employ such variations as appropriate, and the inventors intend for the invention to be practiced otherwise than as specifically described herein. Accordingly, this invention includes all modifications and equivalents of the subject matter recited in the claims appended hereto as permitted by applicable law. Moreover, any combination of the above-described elements in all possible variations thereof is encompassed by the invention unless otherwise indicated herein or otherwise clearly contradicted by context.

[0125] All references, including publications, patent applications, and patents, cited herein are hereby incorporated by reference to the same extent as if each reference were individually and specifically indicated to be incorporated by reference and were set forth in its entirety herein.

WHAT IS CLAIMED IS:

1. A computer-implemented method, comprising:

receiving a delegation request from a first entity, fulfilment of which involves granting a second entity an access privilege to a computing resource;

generating a session key based at least in part on a restriction and a secret credential shared with the first entity, the restriction indicating a limitation on access to be granted to the second entity;

providing the session key to the first entity;

receiving, from the second entity, an access request to access the computing resource, the access request including information that indicates access to the session key;

validating the access request based at least in part on the session key; and

granting, to the second entity, access to the computing resource if the access request is validated.

2. The computer-implemented method of claim 1, wherein the restriction corresponds to an identity of a key zone of a plurality of key zones.

3. The computer-implemented method of claim 1, wherein:

the delegation request includes an identity of the entity for which the session key is to be generated; and

the restriction is based at least in part on the identity of the entity.

4. The computer-implemented method of claim 1, wherein the restriction corresponds to a restriction on a permitted action.

5. The computer-implemented method of claim 1, wherein:

the access request further includes the restriction provided by the first entity; and

validating the access request includes validating that the session key was generated based at least in part on the restriction included in the access request.

6. The computer-implemented method of claim 1, wherein the restriction corresponds to a limit on an amount of time the session key is valid.

7. A system, comprising:

one or more processors; and

memory including instructions that, when executed by the one or more processors, cause the system to:

receive a delegation request from a first entity, fulfilment of which involving granting a second entity an access privilege to a computing resource;

in response to receipt of the delegation request:

generate a session key based at least in part on passing a secret credential, shared between the first entity and the one or more computer systems, and a session restriction through a cryptographic hash algorithm, the session restriction indicating a limitation on access to be granted to the second entity; and

provide the session key to the first entity;

receive an access request from the second entity to access the computing resource, the access request associated with the session key; and

in response to receipt of the access request:

validate the access request based at least in part on the session key; and

grant, to the second entity, access to the computing resource if the access request is validated.

8. The system of claim 7, wherein the instructions that validate the access request include instructions that cause the system to validate whether the request complies with the session restriction.

9. The system of claim 7, wherein the access request is a first access request and the instructions include further instructions that cause the system to:

receive a second access request from a third entity to access the computing resource, the second access request associated with the session key; and

in response to receipt of the second access request:

validate the second access request based at least in part on the session key; and

grant, to the third entity, access to the computing resource.

10. The system of claim 7, further comprising the first entity, wherein the first entity comprises one or more computer systems configured with first instructions that, when executed by the one or more computer systems, causes the one or more computer systems to, as a result of receiving the session key, provide the session key to the second entity without providing the secret credential to the second entity.

11. The system of claim 10, wherein the first instructions that provide the session key to the second entity include instructions that cause the one or more computer systems to provide information, usable by the second entity to obtain the session key, to an electronic destination accessible to the second entity.

12. The system of claim 11, wherein the electronic destination is an e-mail address.

13. The system of claim 7, wherein the instructions that cause the system to validate the access request, include instructions that cause the system to apply the cryptographic hash algorithm both to a first set of inputs and a second set of inputs, wherein:

the first set of inputs include the secret credential, the access request, and the session restriction; and

the second set of inputs include the session key and the access request.

14. The system of claim 13, wherein the cryptographic hash algorithm is a hash-based message authentication code function.

15. The system of claim 7, wherein the access request is associated with the session key by providing a digital signature with the access request generated using the session key.

16. The system of claim 15, wherein the instructions that cause the system to validate the access request, include instructions that cause the system to:

apply the cryptographic hash algorithm to the secret credential, the access request, and the session restriction to obtain a hash result; and

compare the hash result to the digital signature.

17. A non-transitory computer-readable storage medium having stored thereon executable instructions that, when executed by one or more processors of a computer system, cause the computer system to at least:

receive a first request from a first entity, fulfilment of which involving granting a second entity an access privilege to a computing resource;

generate a session key based at least in part on a restriction and a secret credential shared between the first entity and the computer system, the restriction indicating a limitation on access to be granted to the second entity;

provide the session key, usable at least in part to prove possession of an access privilege to a computing resource, to the first entity;

receive a second request to access the computing resource, fulfilment of which involves providing a second entity access to a computing resource, the second request associated with the session key;

validate the second request based at least in part on the session key; and

fulfill the second request, if validated, by providing access to the computing resource.

18. The non-transitory computer-readable storage medium of claim 17, wherein the executable instructions that cause the computer system to fulfill the second request include executable instructions that cause the computer system to fulfill the second request without providing the second entity access to the secret credential.

19. The non-transitory computer-readable storage medium of claim 17, wherein the executable instructions further include executable instructions that cause the computer system to:

receive a third request to access the computing resource, fulfilment of which involves providing a third entity access to the computing resource, the third request associated with the session key;

validate the third request based at least in part on the session key; and

fulfill the third request by providing access to the computing resource depending at least in part on validation of the session key.

20. The non-transitory computer-readable storage medium of claim 17, wherein:

the second request includes a digital signature generated using the session key; and

validating the second request further includes validating authenticity of the digital signature.