

SECURE AND EFFICIENT ONE-TIME PASSWORD SYSTEM AND METHOD

CROSS-REFERENCE TO RELATED APPLICATIONS

[0001] None.

5

BACKGROUND

[0002] Passwords are often used for authentication in computer systems and web services. In order to access a service or resource, a user can provide a password along with a username to a computer system. The computer system can verify the correctness of the password corresponding to that username in order to
10 authenticate the user. If the user is successfully authenticated, the user can be granted access to that service or resource. For example, a username and password can be used to access an email account.

[0003] One-time passwords (also referred to as “OTPs” or “OTP codes”) are a type of password. While a conventional password may be valid for a long period of
15 time (e.g., six months) or even indefinitely, a one-time password may be valid for, e.g., a single use. However, the term “one-time password” is often used to refer to more general limited use passwords, rather than passwords that are valid for exactly one use. As such, a one-time password may refer to a password that is valid for e.g., five uses, or may be valid for a short period of time (e.g., 15 minutes), or may
20 be valid up until a certain date in the near future.

[0004] One-time passwords can be used as a form of single-factor authentication. As an example, a user can attempt to log into a computer system by providing an email address. The computer system can generate a one-time password and transmit it to that email address. The user can then access their
25 emails, read the one-time password, and provide the one-time password to the computer system. The computer system can verify the one-time password and authenticate the user provided the one-time password is correct.

[0005] One-time passwords can also be used for multi-factor authentication, e.g., in conjunction with traditional passwords. For example, a user can attempt to

log into a computer system by providing a username and password. The computer system can verify that the password is correct, and (provided the password is correct) can generate a one-time password, which can be sent in a text message to a phone number associated with the user. The computer system can then prompt the user to enter the one-time password. The user can read the one-time password off their phone and enter it into the computer system, at which point the computer system can verify the one-time password and authenticate the user. In this way, the computer system can determine that the user both knows their username and traditional password, and is in possession of the phone associated with the user. As such, the use of one-time passwords can enable multi-factor authentication techniques that are more secure than traditional password based authentication.

[0006] In a conventional one-time password authentication system, after a one-time password is generated, it is typically stored in some data structure (such as a “one-time password table”), enabling a computer system to later retrieve the one-time password in order to authenticate the user, e.g., upon receiving a one-time password from the user. Typically, the one-time password is stored in association with an identifier. When generating the one-time password, the computer system may also generate the identifier and provide it to the user. When a user (or a user device operated by the user) provides a one-time password to the computer system, the user or user device may also provide the identifier to the computer system. The computer system can use the identifier to look-up the corresponding one-time password in the table, and verify that the one-time password matches the provided one-time password.

[0007] This conventional one-time password authentication system has several security risks. If a cybercriminal or other malicious entity acquires the one-time password table, they can use the identifiers stored in the one-time password table to impersonate any legitimate user corresponding to those identifiers. A cybercriminal could potentially request a “legitimate” one-time password from the computer system by impersonating the user using the identifier, potentially enabling the cybercriminal to access services or resources that the cybercriminal should not be able to access.

[0008] There are some techniques that can be used to secure such conventional one-time password systems. For example, one-time passwords and identifiers can be stored in encrypted form. In this way, even if a cybercriminal acquires an encrypted one-time password table or encrypted identifiers, they may be
5 unable to decrypt and determine the one-time passwords or identifiers. In such a case, the identifiers and one-time passwords must be encrypted during one-time password generation and decrypted during one-time password authentication. This can be a problem because encryption and decryption operations are often time consuming. For high volume one-time password authentication systems (e.g.,
10 associated with popular webservices), it may not be feasible to perform such encryption and decryption operations and still generate and authenticate one-time passwords for large volumes of users in a timely manner.

[0009] Embodiments address these and other problems, individually and collectively.

15

SUMMARY

[0010] Embodiments of the present disclosure are directed to novel methods and systems for generating, provisioning, and storing one-time passwords, as well as methods and systems for authenticating users using said one-time passwords. Services can be provided to users (or “client devices” operated by those users) upon
20 successful authentication of those users.

[0011] A client device can transmit a service request to a server computer, which can contain one or more data elements, which can include a user identifier (e.g., data that identifies the user) and an intent identifier (e.g., data that identifies the user’s intent, e.g., identifying a particular service that the user intends to access).
25 The server computer can combine the one or more data elements in some way (e.g., via concatenation) and hash the resulting combination using a hash function, thereby generating a hash. The server computer can generate a one-time password and store it in a data structure in association with the hash, such that the hash can later be used to “look-up” or otherwise identify the one-time password.

30 **[0012]** The server computer can provide the one-time password to the user, e.g., by transmitting it to the client device or by transmitting it to another client device associated with the user. For example, if the user is attempting to log into a

webservice (such as a video on-demand streaming service) using their laptop (a “first client device”), the server computer can transmit the one-time password to the laptop, or could, e.g., transmit the one-time password to a smartphone (a “second client device”) associated with the user, e.g., via a text message.

5 **[0013]** To complete the authentication process, the user device can transmit the one or more data elements along with the one-time password to the server computer. The server computer can combine these elements of data and hash the resulting combination, generating a second hash. The server computer can then
10 look up the one-time password in the data structure using the second hash and compare it to the one-time password received from the user. If the one-time passwords match, the computer system can authenticate the user. The server computer can then provide a service to the user, e.g., via the client device. This service can correspond to the intent identifier provided by the user. For example, if the user’s intent was to change the password on their online banking account, the
15 server computer could enable the user to do so via the client device.

[0014] In more detail, one embodiment is directed to a method performed by a server computer. The server computer can receive a service request comprising one or more first data elements from a client device. The one or more first data elements can include a first user identifier. The server computer can hash the one or
20 more first data elements to form a first hash using a hash function. The server computer can generate a first one-time password. The server computer can store the first one-time password in association with the first hash. The server computer can transmit the first one-time password to the first client device or a second client device. The server computer can receive a second one-time password and one or
25 more second data elements from the first client device or the second client device. The one or more second data elements can include a second user identifier. The server computer can hash the one or more second data elements to form a second hash using the hash function. The server computer can determine the first one-time password using the second hash. The server computer can compare the first one-
30 time password to the second one-time password. Based on this comparison, the server computer can provide a service associated with the service request to the first client device.

[0015] Another embodiment is directed to a method performed by a client device. The client device can transmit a service request comprising one or more data elements. The one or more data elements can include a user identifier. The server computer can form a first hash using a hash function and the one or more data elements. The server computer can further generate a one-time password, and can store the one-time password in association with the first hash. The client device can receive the one-time password from the server computer. The client device can transmit the one-time password and the one or more data elements to the server computer. The server computer can form a second hash using the hash function and the one or more data elements. The server computer can determine a stored one-time password using the second hash. The server computer can compare the stored one-time password to the one-time password. The client device can receive a service provided by the server computer and corresponding to the service request. The server computer can provide the service to the client device based on comparing the stored one-time password to the one-time password.

[0016] Some other embodiments are directed to systems configured to perform these methods and other methods. For example, an embodiment is directed to a server computer comprising a processor and a non-transitory computer readable medium coupled to the processor, the non-transitory computer readable medium comprising code or instructions, executable by the processor for performing one of the methods described above.

[0017] These and other embodiments of the present disclosure are described in the detailed description below.

TERMS

[0018] A “server computer” may refer to a computer or cluster of computers. A server computer may be a powerful computing system, such as a large mainframe. Server computers can also include minicomputer clusters or a group of servers functioning as a unit. In one example, a server computer can include a database server coupled to a web server. A server computer may comprise one or more computational apparatuses and may use any of a variety of computing structures, arrangements, and compilations for servicing requests from one or more client devices.

- [0019]** A “client device” may comprise a device that accesses service(s) made available by a server computer. The server computer may be on another computer system, and as such, the client device may access the service by way of a network, such as the Internet. A client device may comprise a “client computer,” and the term
- 5 “client” may refer to either a program or computer hardware or software that accesses a service made available by a server computer, or may alternatively refer to an entity, such as a “user,” that owns and/or operates the client device. For example, a client can comprise a web browser operating on a client device. A client device may comprise a “user device.”
- 10 **[0020]** A “user” may refer to an entity that uses something for some purpose. An example of a user is a person who uses a “user device” or a “mobile device.” A user device may refer to any device operated by a user, such as a smartphone, smartcard, wearable device, laptop, tablet, desktop computer, etc.
- [0021]** A “memory” may refer to any suitable device or devices that may store
- 15 electronic data. A suitable memory may comprise a non-transitory computer readable medium that stores instructions that can be executed by a processor to implement a desired method. Examples of memories include one or more memory chips, disk drives, etc. Such memories may operate using any suitable electrical, optical, and/or magnetic mode of operation.
- 20 **[0022]** A “processor” may refer to any suitable data computation device or devices. A processor may comprise one or more microprocessors working together to accomplish a desired function. The processor may include a CPU that comprises at least one high-speed data processor adequate to execute program components for executing user and/or system generated requests. The CPU may be a
- 25 microprocessor such as AMD’s Athlon, Duron and/or Opteron; IBM and/or Motorola’s PowerPC; IBM’s and Sony’s Cell processor; Intel’s Celeron, Itanium, Pentium, Xenon, and/or Xscale; and/or the like processors.
- [0023]** An “identifier” may refer to data that can be used to identify something. For example, a “device identifier” may refer to data (such as an alphanumeric code)
- 30 that can be used to identify a device, and a “user identifier” may refer to an identifier (such as a given name) that can be used to identify a user.

[0024] A “one-time password” may refer to passwords that have some restrictions limiting their user. For example, as implied by the name, a one-time password can be a password that is valid for exactly one use. However, one-time passwords may also refer to passwords that can be used, e.g., three times, five times, etc., or may refer to passwords that are valid for a certain period of time (e.g., 15 minutes, one day). As such, the qualifier “one-time” should not necessarily be interpreted literally.

[0025] A “hash function” may refer to a function that takes data of arbitrary size or length as an input and produces data of fixed size or length as an output. The output of a hash function may be referred to as a “hash value” or a “hash.” A hash function can be “collision resistant” if it has a reduced likelihood of a “hash collision”, which can occur when two different inputs result in identical hash values. A hash function can be “secure” or “cryptographically secure” if it is difficult or impossible to determine the input of the hash function based on its output.

[0026] A “data structure” may refer to any structure that can be used to store data. “Databases”, “data lakes”, and “data tables” are examples of data structures. Typically, the term data structure can refer to virtual structures to store data, as opposed to physical structures. However, the term data structure may also refer to physical hardware used to store data. In some cases, terms such as “database” may refer to either physical hardware or a data structure implemented using that hardware.

[0027] A “database” may refer to a structured set of data held or stored in a computer or other device. Alternative, a database may refer to a device which hold such a structured set of data. A “data record” may refer to a unit of data stored in a database. Data stored in a database may be in the form of a “data table”, an arrangement of data in rows, columns, or more complex structures. Data tables may comprise “key value” pairings, in which “values” (e.g., numerical values, strings, or any other data) are associated with corresponding “keys” (e.g., labels corresponding to those values). Databases can be parsed (e.g., searched or queried) or sorted based on the values corresponding to keys, e.g., using query languages such as SQL.

[0028] A “hash table” may refer to a type of data table that uses hashes. In a hash table, the keys in key-value pairs may comprise hash values. As an example, for a phonebook hash table, instead of comprising names, keys could comprise hash values derived from those names using a hash function. Values (e.g., phone numbers) can be retrieved from a hash table by generating a hash based on some data (e.g., a name), then parsing the hash table using the generated hash in order to find a corresponding record, then retrieving the values corresponding to that key and that record.

[0029] “Authentication” may refer to the process or action of proving or showing something to be true, genuine, or valid. Authenticating a user may refer to the process of verifying the identity of the user, e.g., verifying that the user is who they claim to be.

[0030] A “cookie”, “HTTP cookie”, “web cookie”, “browser cookie,” etc., may refer to data produced by a webpage and stored on a user device, usually in association with a browser application operating on that device.

BRIEF DESCRIPTION OF THE DRAWINGS

[0031] FIG. 1 shows an exemplary service systems, which can be used to provide services to users of client devices.

[0032] FIG. 2A shows an issuance phase of an exemplary method for authenticating a user as part of a service request according to some embodiments.

[0033] FIG. 2B shows an authentication phase of an exemplary method for authenticating a user as part of a service request according to some embodiments.

[0034] FIG. 3 shows an exemplary client device according to some embodiments.

[0035] FIG. 4 shows an exemplary server computer according to some embodiments.

DETAILED DESCRIPTION

[0036] As summarized above, embodiments of the present disclosure are directed to novel methods and systems for one-time password authentication. In

broad terms, an authentication method can involve both an issuance phase and an authentication phase. During the issuance phase, a server computer (which may also be referred to as an “authentication computer” or a “one-time password management computer”) can combine data elements received from a client device
5 (such as a user identifier and an intent identifier) and use the resulting combination to generate a hash. The server computer can generate a one-time password and store it in a data structure (such as a hash table) using the hash as a key. The one-time password can be issued to the client device (or another client device).

[0037] Later, during the authentication phase, the client device can provide
10 the data elements (e.g., the user identifier and the intent identifier) along with the one-time password to the server computer. The server computer can again combine the data elements and hash the resulting combination to generate a hash, then use the hash to identify the stored one-time password. The server computer can then compare the stored one-time password to the received one-time password in order
15 to authenticate the user. Both combining multiple elements of data to derive the hash values, and the use of hash values rather than traditional keys or identifiers provide advantages over conventional one-time password authentication systems, as described in more detail below. Upon authenticating the user, the server computer can provide a service to the user, either directly or by enabling the user to
20 receive the service from another server computer (e.g., from a “second server computer”), for example, by providing access data that can enable the user to access the service provided by the other server computer.

[0038] Prior to describing embodiments of the present disclosure in more detail, some features and characteristics one-time passwords, user identifiers, and
25 intent identifiers according to embodiments are described below in order to facilitate a better understanding of methods and systems according to embodiments.

[0039] In some embodiments, one-time passwords may be associated with one or more “one-time password types”, which may be labels, classifications, or other elements of data that in some way defines a “type” corresponding to a given
30 one-time password. For example, a one-time password type may indicate that a corresponding one-time password is a single use password and is only valid for a single authentication attempt. A different one-time password type may indicate that

a corresponding one-time password is a "timed access" one-time password, which is valid only during a specified time period. As another example, a one-time password type may indicate that a one-time password is an "elevated" or "privileged" one-time password, indicating that the corresponding one-time password can be used to
5 access "privileged" services (described in more detail with reference to FIG. 1) that would generally be inaccessible with an "unelevated" or "unprivileged" one-time password.

[0040] A one-time password type may comprise data, such as alphanumeric codes used to identify the one-time password type. For example, a code such as
10 "000:1" may indicate a single use password, a code such as "000:3" may indicate a one-time password valid for three usages, a code such as "012:5" may indicate a one-time password valid for five minutes, a code such as "012:15" may indicate a one-time password valid for fifteen minutes, and a code such as "145:1" may indicate an elevated one-time password valid for a single use. A one-time password may
15 have multiple corresponding one-time password types, and may therefore correspond to multiple elements of data or alphanumeric codes indicating those types. For example, a one-time password with types "145:1 / 012:15" may indicate that the one-time password is an elevated one-time password valid for a single use within the next fifteen minutes. It should be understood that the alphanumeric codes
20 above are provided only as examples and that a wide variety of data can be used as one-time password types, including a wide variety of codes conforming to any appropriate format.

[0041] In some embodiments, one-time passwords can be stored in data structures in association with their respective one-time password type(s) along with
25 any other data that may facilitate one-time password based authentication. For example, if a user attempts to use a one-time password to authenticate and access a privileged service, an authenticating computer (e.g., a server computer) can verify that a corresponding one-time password type indicates that the one-time password is valid for accessing privileged services.

30 **[0042]** Examples of other data that may be stored in association with one-time passwords include dynamic data, such as counters. A counter may comprise a number or other data that indicates the number of times that a one-time password

has been used. During authentication, an authentication computer can check the counter against, e.g., a one-time password type to verify that the one-time password is still valid. After each authentication attempt (or each successful authentication attempt), the computer system can increment the counter so that it accurately
5 reflects the total number of authentication attempts or total number of authentications.

[0043] Another example of data that may be associated with (or stored in association with) a one-time password are timestamps. A timestamp can indicate the time at which a one-time password was generated or issued and/or transmitted
10 to a user device. During authentication, the computer system can use the timestamp and the current time to evaluate how much time has elapsed since the one-time password was generated (or issued). The computer system can compare this elapsed time against, e.g., a time indicated by the one-time password type in order to determine if the one-time password is still valid. As an alternative, the timestamp
15 could correspond to the time at which the one-time password expires. During authentication, the timestamp can be compared against the current time to determine if the one-time password is still valid.

[0044] It should be understood that these examples of other data that may be associated with, or stored in association with one-time passwords are intended only
20 as non-limiting examples, and that other types of data may become apparent upon reading the present disclosure.

[0045] Some methods according to embodiments can involve a server computer combining data elements, such as user identifiers and device identifiers, and using the resulting combination to generate hash values as part of a one-time
25 password based authentication process. As such, user identifiers and intent identifiers according to embodiments are described in more detail below.

[0046] As defined above in the Terms Section, a user identifier can comprise data that can be used to identify a user. As an example, a user identifier can comprise a username such as "JohnDoe85" which may be produced by the
30 corresponding user or generated and assigned to the user by a computer system, such as a server computer, or another entity. Another example of a user identifier is an account number such as a 16 digit payment account number associated with a

credit card. Yet another example of a user identifier is a user email address, such as "JohnDoe@example.com". Other examples of user identifiers include phone numbers and biometric data, such as a biometric produced by the user (e.g., a thumb scan or iris scan, produced, e.g., using a client device) or a biometric template
5 derived from such biometric data. Another example of a user identifier is a cookie, which may be stored on a user's browser and which may identify that user. There are wide variety of user identifiers and those described above are intended only as non-limiting examples. In some embodiments, a user identifier can refer to one or more user identifiers, e.g., a "first user identifier" could comprise both a username
10 and a phone number.

[0047] An "intent identifier" can comprise an element of data, such as an alphanumeric string or code, that in some way identifies a user's intent with regard to a service or service system. For example an intent identifier such as "intent:ResetPassword" can indicate that a user is attempting to access a service in
15 order to reset their password. Examples of intent identifiers include "user registration intent identifiers" (e.g., indicating that a user intends to register as a user of a service or service system), "password recovery intent identifiers" (e.g., indicating that a user intends to reset or recover their password), "contact update intent identifiers" (e.g., indicating that a user wants to update their contact information stored in association
20 with a service or a service system), login intent identifiers (e.g., indicating that a user wants to log into a service system for the purpose of accessing general services provided by that service system, e.g., in order to watch movies or television shows provided by an on-demand video streaming service), or privileged service intent identifiers (e.g., indicating that a user intends to access one or more privileged
25 services). In some embodiments, a contact update intent identifier can comprise an email update intent identifier (indicating, e.g., that a user wants to update an email address associated with their account) and/or a phone number intent identifier (indicating, e.g., that a user wants to update the phone number associated with their account). In some embodiments, the privileged service intent identifier can comprise
30 a key management intent identifier, which can indicate that a user wants to remove, add, or modify cryptographic keys used by a server computer or service system to perform its functions. For example, an administrator user may access a secure messaging service in order to update the cryptographic keys used to encrypt and

decrypt messages, if, for example, those cryptographic keys are leaked or are otherwise no longer secure. There are a wide variety of intent identifiers corresponding to a wide variety of possible intents or services, and those described above are intended only as non-limiting examples. In some embodiments, an intent
5 identifier can comprise one or more intent identifiers described above. For example, an intent identifier can include both a login intent identifier and a phone number update intent identifier, indicating that a user intends to login to their account and change the phone number associated with their account.

[0048] Having described one-time passwords, associated data, user
10 identifiers, and intent identifiers according to embodiments, it may now be useful to describe systems according to embodiments. FIG. 1 shows an exemplary service system 108. FIG. 1 is intended to describe, generally, how authentication, particularly one-time password based authentication can be used to grant a user (e.g., user 102) access to one or more services (e.g., unprivileged service(s) 116
15 and/or privileged services 118) provided by the service system 108. In the description below, FIG. 1 is used to describe both service systems according to embodiments of the present disclosure as well as conventional service systems, in order to contrast embodiments of the present disclosure with conventional systems, thereby highlighting some advantages provided by embodiments of the present
20 disclosure.

[0049] A service system 108 can comprise a network of computers, servers, and other devices associated with one or more services. Some of these services may comprise "primary services," e.g., services that the service system 108 is primarily associated with. For example, for a service system 108 associated with a
25 video-on-demand streaming service, the primary service may comprise streaming videos to client devices (FIG. 1 shows two client devices, first client device 104 and second client device 106, owned and/or operated by a user 102, however, it should be understood that a user such as user 102 can own and/or operate any number of client devices). As another example, for a service system 108 associated with online
30 banking, these primary services may comprise services associated with online banking, e.g., viewing checking and savings account balances, paying credit card bills, etc. However, although a service system 108 may be associated with a relatively small set of primary services, many service systems provide a variety of

services in addition to their primary services. For example, a video streaming service system may provide services enabling users to change their password or update their contact information, in addition to a primary service enabling users to stream television shows and movies to their client devices.

5 **[0050]** Services provided by service systems such as service system 108 may be divided into unprivileged service(s) 116 and privileged service(s) 118. An unprivileged service 116 may comprise a service that can be accessed by a user (such as user 102) without any “privileges” or with a small or minimal set of privileges. For example, a “contact information update service” may enable a user
10 102 to updated the contact information associated with their account (which may be stored, e.g., in a user repository 120). It should be understood that generally, even “unprivileged” services 116 may require some set of privileges to access; for example, a user 102 may need to log into their account in order to request a change to their contact information, and as such, that user 102 may be “privileged” in the
15 sense that they are an existing account holder, but may be “unprivileged” in that the user 102 can access the contact information update service without needing to be an administrator or other privileged user.

[0051] A privileged service 118 may comprise a service for which a user requires some stronger set of privileges to access, such privileged services 118 may
20 be inaccessible to many of a service system’s 108 users. For example, a video streaming service system may have a privileged “content control” service that enables an administrator (or other privileged user) to remove videos from the service, e.g., if the service loses the rights to stream those videos, or if those videos do not comply with the service’s content standards. As another example, service
25 system 108 may have a privileged “user record access” service, which may enable an administrator (or other privileged user) to access stored personally identifiable information (stored, e.g., in user repository 120) corresponding to a particular user. Such a service may be privileged in order to protect the privacy of users.

[0052] As such, many service systems such as exemplary service system 108
30 may implement some form of access control beyond password or one-time password based authentication. In some cases, a user (such as user 102) might have a “privilege level” or “command level” that can indicate which services that user is

allowed to access. For example, a user 102 may be able to access services that have a privilege level less than or equal to the privilege level of the user 102, and may not be able to access services with a higher privilege level. The privilege level of a user 102 can be associated with a user record, stored, e.g., in a user repository
5 such as user repository 120. When a user 102 attempts to access a particular service, an exemplary service system 108 (or a computer component of such a service system 108) can access the user record corresponding to that user 102 in the user repository 120 in order to determine if the user 102 is permitted to access the service. In some cases, accessing “more privileged” services may require more
10 forms of authentication and/or different forms of authentication. For example, accessing a highly-privileged service may require three-factor authentication, whereas accessing a less privileged service may only require single-factor authentication.

[0053] Services provided by the service system 108 can be provided by one
15 or more service provider computers (which may be more generally referred to as “server computers”) 112 associated with the service system 108. In order to access such services, however, a user 102 may need to be authenticated by the service system 108. To do so, the user 102 can communicate with an authentication computer 114 (which may also be more generically referred to as a “server
20 computer”) via a client device, such as first client device 104 and/or second client device 106. In some embodiments, the client device(s) and the authentication computer 114 can perform a one-time password based authentication process in order to authenticate the user 102 and enable the user 102 to access a service provided by the service system 108.

[0054] Often in larger systems, communications between computers and
25 devices outside the system (e.g., the first client device 104 and second client device 106) are mediated by a gateway, firewall, load balancer, reverse proxy, etc. (e.g., 110) or a combination thereof. Such systems can direct network traffic to their intended recipient, filter out unwanted or malicious service requests, balance the
30 load between servers (such as service provider computer(s) 112), and/or perform other similar functions. However, in some service systems, client devices may communicate directly with authentication computers. As such, it should be understood that a phrase such as “the first client device 104 transmits a service

request to an authentication computer 114” is intended to describe both “direct” transmission between the first client device 104 and the authentication computer 114, and “semi-direct” transmission (e.g., via one or more internet relays and/or a gateway 110), or some combination thereof.

5 **[0055]** In general terms, in a one-time password based authentication, a service system 108 (or an element of the service system such as an authentication computer 114) can generate a one-time password and transmit it to a client device operated by the user 102. The authentication computer 114 can store the one-time password in a data structure, such as one-time password database 122. Later, the
10 user 102 can then provide that one-time password back to the service system 108 as a credential in order to complete authentication and access a service or services provided by the service system. The authentication computer 114 can retrieve a stored one-time password and compare it to a one-time password received from a client device. If the one-time passwords match, the authentication computer 114 can
15 authenticate the user 102 and enable the user to access a service, e.g., provided by service provider computer(s) 112.

20 **[0056]** For service systems 108 in which one-time passwords are used as a form of multi-factor authentication, a one-time password can be transmitted to a client device other than the client device that made the authentication request or service request. For example, if a user 102 initiates a service request using first client device 104, the authentication computer 114 can transmit a one-time password to second client device 106. In this way, if the user 102 is able to provide the one-time password back to the service system (e.g., via first client device 104), it demonstrates that the user 102 is in possession of the second client device 106. If
25 the second client device 106 is known to be owned or operated by the user 102, possession of the second client device 106 can indicate that the user 102 is more likely to be who they purport to be, and therefore transmitting the one-time password to the second client device 106, rather than the first client device 104 can provide a stronger form of authentication.

30 **[0057]** As described above in the background, in conventional one-time password authentication systems, an identifier is often generated in addition to a one-time password. The one-time password can be stored in a one-time password

database, such as one-time password database 122 in association with this identifier, allowing the one-time password to be later identified by the authentication computer 114, e.g., when comparing the stored one-time password to a received one-time password when authenticating the user 102. This one-time password can
5 be provided to a client device. When the client device transmits a one-time password back to the service system to complete authentication, the client device can also transmit the identifier. Using the received identifier, the authentication computer 114 can retrieve the corresponding one-time password from the one-time password database 122 and compare it to the received password in order to
10 authenticate the user 102.

[0058] This use of identifiers can pose a security risk, as it requires the authentication computer 114 to securely store the identifier and securely transmit the identifier to a client device, and further requires the client device to securely store the identifier. If a cybercriminal gains access to an identifier, e.g., by stealing it from the
15 first client device 104, the second client device 106, or the authentication computer 114, or intercepting it during communication between the service system 108 and either client device, the cybercriminal can potentially impersonate the user 102 or otherwise misuse the identifier.

[0059] For example, if a cybercriminal acquires a valid identifier, they may be
20 able to use that identifier to request or otherwise acquire a valid one-time password, enabling the cybercriminal to use that one-time password in order to authenticate themselves as the user 102, and potentially gain access to resources, services, or data intended for the user 102. If a cybercriminal is able to generate valid one-time passwords (in addition to having access to an illicitly acquired identifier), the
25 cybercriminal can potentially trick the service system 108 to, e.g., accept a one-time password generated for one purpose for a different purpose. In this way, a cybercriminal could, for example, use a stolen identifier (corresponding to e.g., an administrator) and a one-time password generated for an unprivileged service (e.g., changing contact information) in order to impersonate the administrator and access a
30 privileged service, such as gaining access to personally identifiable information from the user repository 120. As such, in conventional one-time password authentication systems, sensitive information needs to be protected when stored or transmitted to user devices. This can create additional security overhead, increase the complexity

of one-time password authentication, and reduce the speed and efficiency of one-time password generation and authentication.

[0060] By contrast, in some embodiments of the present disclosure, no additional information (such as a new identifier generated by the authentication computer 114) needs to be securely transmitted to the first client device 104 or second client device 106 and securely stored. Instead, as described below, a hash can be derived from data received from a client device in an authentication or service request (such as a user identifier and an intent identifier). This information may comprise non-private information, or information that is already known by the first client device 104, second client device 106, and/or the authentication computer 114. Such hashes can be derived both when a one-time password is issued to a client device, and when the client device uses that one-time password during authentication, and therefore may not need to be stored in the interim. This is advantageous because it reduces security risk and overhead. A cybercriminal cannot acquire a hash and use it to impersonate the user 102 or otherwise defeat the service system 108.

[0061] In more detail, In some embodiments, a client device (such as first client device 104) can send a service request comprising one or more data elements to a server computer (e.g., authentication computer 114). The one or more data elements can include both a user identifier (e.g., a username or an email address associated with the user 102) and an intent identifier. The intent identifier may specify the user's 102 intent, and may indicate which service of unprivileged service(s) 116 and privileged service(s) 118 that the user 102 is attempting to access. In addition to generating a one-time password, the authentication computer 114 can combine and hash the user identifier and optionally the intent identifier to form a hash. The server computer can store the one-time password in association with the hash, e.g., in one-time password database 122. The server computer can store the one-time password in any appropriate data structure, e.g., a hash table stored in one-time password database 122.

[0062] This is in contrast to conventional one-time password authentication systems, which do not store one-time passwords in hash tables or other hash-based data structures. Unlike conventional one-time password authentication systems that

store one-time passwords in plaintext, it is not possible for cybercriminals to determine potentially sensitive information (e.g., user identifiers and intent identifiers) from their corresponding hash values (even if a cybercriminal is capable of acquiring such a hash value from the one-time password database 1122), as those hash
5 values cannot be reversed to reconstruct user identifiers and intent identifiers.

[0063] Further, hashing is generally faster and more efficient than encryption and decryption, providing speed and efficiency improvements over conventional one-time password authentication systems that use encryption or decryption. Additionally, in a conventional system, if the cryptographic keys used to encrypt the
10 identifiers and one-time passwords are stolen by cybercriminals, those cybercriminals can decrypt and acquire the identifiers and one-time passwords. By contrast, in embodiments, provided the underlying hash function is secure, a cybercriminal cannot determine any data elements (e.g., the user identifiers and intent identifiers) used to generate the hash values.

15 **[0064]** After generating and storing the one-time password in the one-time password database 122, the authentication computer (server computer) 114 can provide the one-time password to either the first client device 104 or the second client device 106. For example, the authentication computer 114 could send the one-time password to an email address associated with the user 102, which the user
20 102 could access via the first client device 104 or the second client device 106.

[0065] In some embodiments, upon receiving the one-time password (or sometime thereafter), the user 102 can then transmit the one-time password back to a server computer (e.g., authentication computer 114) along with the one or more data elements (e.g., a user identifier and an intent identifier). The server computer
25 can combine and hash the one or more data elements to form the hash, then use the hash to look up the one-time password in one-time password database 122. The server computer can then compare the received one-time password to the stored one-time password. If the received one-time password and the stored one-time password match, the server computer can authenticate the user 102, enabling the
30 user 102 to access the requested services via the first client device 104 and/or the second client device 106.

[0066] How a user 102 is granted access to a service can depend on how the service system 108 is designed. In smaller service systems 108, which may comprise a single server performing both authentication functions and providing services, a server computer (such as authentication computer 114) may also be a service provider computer 112, and itself may provide the requested service. In larger systems, a server computer may transmit access data such as a cookie or a session key, which may enable the first client device 104 or second client device 106 to communicate with a second server computer (e.g., a service provider computer 112) in order to receive the requested service. As such, it should be understood that a phrase such as “the server computer provides a service associated with a service request to the first client device 104” is intended to cover both the server computer directly providing that service and the server computer performing actions that enable that service to be provided to the first client device 104, e.g., by transmitting access data to the first client device 104, enabling the first client device 104 to receive that service from another server computer.

[0067] In many service systems, such as exemplary service system 108, it is generally preferable to control access to services, in order to prevent users from accessing privileged services 118 that the users may not be permitted to access. In general, it is preferable that a one-time password generated to access a particular service cannot be used to access other services (even other services of similar privilege levels), in order to control access to services. As described above, other access control methods and systems (beyond one-time password based authentication) may be implemented, such as assigning privilege levels to users and services. Implementing such systems can increase the security overhead and consequently increase one-time password implementation difficulty and the time required to authenticate user using one-time passwords.

[0068] However, by using intent identifiers, embodiments of the present disclosure can provide an efficient, “built-in” form of access control. As described above, in some embodiments a server computer (e.g., authentication computer 114) can use an intent identifier received from the user 102 (e.g., in combination with a user identifier) to generate a hash, which can be stored in association with a generated one-time password. Such an intent identifier can correspond to a service that the user is attempting to access, e.g., an (unprivileged) password recovery

service may have a different intent identifier than a (privileged) user repository access service.

[0069] Using intent identifiers in this manner is an efficient and built-in form of access control, as it prevents one-time passwords corresponding to one service from
5 being used to access other services. A user 102 can be issued a valid one-time password corresponding to a particular service request, but cannot use that one-time password to access other services, as the intent identifiers used when the password is issued and the user is authenticated would be different and result in different hashes, preventing authentication of the user 102 if the user 102 attempts to switch
10 intent identifiers during authentication.

[0070] This is particularly useful if a cybercriminal has managed to impersonate an administrator or other privileged user, but has not necessarily acquired the full set of credentials needed to access a privileged service. In such a case, the cybercriminal may attempt a workaround, in which they request a one-time
15 password for an unprivileged service, and then attempt to use that one-time password to access the privileged service. However, using methods according to embodiments, the cybercriminal will be unable to access the privileged service using the one-time password generated for the unprivileged service, as the hashes derived from the intent identifiers may not match, even with the same user identifier. By
20 contrast, in a conventional system, a further form of access control logic would be needed to prevent the cybercriminal from using the one-time password for any service.

[0071] Further, embodiments of the present disclosure generally promote system independence, as the hash value can be derived (solely or partially) from
25 information provided by the user 102, e.g., a user identifier and an intent identifier. As such, the authentication computer 114 does not need to access other information to derive the hash value, such as information stored in a user repository 120. As such, the user repository 120 and one-time password database 122 can be “loosely coupled” or even “uncoupled,” i.e., information stored in one does not impact (or
30 minimally impacts) the function or security of the other.

[0072] This loose coupling can provide further security benefits. If either a one-time password data structure (stored, e.g., in one-time password database 122)

or the user repository 120 is breached by a cybercriminal, the cybercriminal is unable to access or determine any data stored in the other data structure. Further, system independence makes individual subsystems of the service system 108 easier to implement, maintain, or modify. Changes made to the user repository 120 (or other
5 subsystem) do not require changes to the one-time password database 122 or the authentication computer 114 and consequently it is easier to modify or upgrade single elements of the service system 108. A user repository 120 or other subsystem could be completely removed and replaced without disrupting the function of the authentication computer 114.

10 **[0073]** Having described both conventional service systems and service systems according to embodiments of the present disclosure, a method for authenticating a user 202 using one-time passwords is described with reference to the diagrams of FIGs. 2A and 2B. This authentication process can be associated with a “service request” associated with the user 202, e.g., a request to access a
15 service provided by a service system (such as the exemplary service system 108 from FIG. 1). Upon successful one-time password based authentication, the user 202 may be able to access this service.

[0074] The one-time password authentication method can generally be divided into an issuance phase (described with reference to FIG. 2A) and an
20 authentication phase (described with reference to FIG. 2B). The issuance phase can generally involve a server computer 208 generating a first one-time password 218 and issuing that first one-time password 218 to the user 202. The authentication phase can generally involve the server computer 208 receiving a second one-time password 222 from the user 202, authenticating the user 202 by e.g., verifying that
25 the second one-time password 222 matches the first one-time password 218. The server computer 208 can optionally provide a service to the user 202. In some cases, the server computer 208 providing the service to the user 202 may involve enabling the user 202 to access the service, rather than providing the service directly, e.g., by enabling the user 202 to receive the service from a second server
30 computer 210 (which could comprise, e.g., a service provider computer as depicted in FIG. 1).

[0075] Referring to FIG. 2A, at step S230, a first client device 204 (e.g., a user device operated by the user, such as a laptop, tablet, smartphone, desktop computer, videogame console, smart tv, etc.) can collect or receive any data necessary to form or otherwise generate a service request (e.g., at step S232).

5 Such a service request can include one or more first data elements 212, which can include a first user identifier and a first intent identifier. As such, at step S230, the first client device 204 can collect or receive some or all of these one or more first data elements 212.

[0076] In some embodiments, the first client device 204 can receive the first user identifier via user inputs produced by a user 202 of the first client device 204. For example, using a touchscreen or keyboard on a smartphone first client device 204, the user 202 could enter a first user identifier, such as an email address. In some other embodiments, the first client device 204 can access the first user identifier via a browser or another application operating on the first client device 204.

15 For example, if the user 202 is using a browser operating on the first client device 204 to attempt to access a service, the browser may securely store the user's 202 email address (or another suitable first user identifier), and the first client device 204 may be able to access that email address from the browser. There are a wide variety of possible first user identifiers, including the examples described further

20 above. In some embodiments, the first user identifier can comprise one or more of a username, an account number, a user email address, a user phone number, a biometric, or a cookie.

[0077] Similarly, in some embodiments, the first client device 204 can also acquire a first intent identifier, from e.g., input produced by the user 202 of the first client device 204 or from data stored by the first client device 204, e.g., data associated with a browser or another application operating on the first client device 204 (such as an application associated with the requested service, e.g., a video on-demand streaming application, a mobile banking application, etc.). As an example, a user 202 could use a mouse connected to a laptop first client device 204 to click on a

25 "forgot my password" button on a webpage associated with a service system. From this, the first client device 204 (or the browser operating on the first client device 204) can surmise that the user's intent is a password reset, and can determine a corresponding first intent identifier, which can take the form of, e.g., a string such as

30

“ResetPassword” or another appropriate alphanumeric string, code or other data. As another example, a user 202 could use the touchscreen of a smartphone first client device 204 to tap a “user registration button” on a video-on-demand streaming application operating on the first client device 204. From this, the first client device 5 204 (or the application operating on the first client device 204) can determine a first intent identifier, such as “userRegistration” or any other appropriate data that can be used to identify that the user 202 intends to register as a user of the service. There are a wide variety of possible intent identifiers, such as the examples provided above. In some embodiments, the first intent identifier can comprise one or more of a “user registration intent identifier,” a password recovery intent identifier,” a “contact 10 update intent identifier”, a “login intent identifier”, or a “privileged service intent identifier.” In some embodiments, a contact update intent identifier can comprise an email update intent identifier and/or a phone number update intent identifier. In some embodiments, a privileged service intent identifier can comprise a key management intent identifier, indicating, e.g., that the user 202 wants to remove, 15 add, or modify cryptographic keys used by the server computer 208 (or an associated service system). In FIG. 2A, the exemplary one or more first data elements 212 comprise a first user identifier “JohnDoe@example.com” and a first intent identifier “userRegistration”.

20 **[0078]** In some embodiments, the one or more first data elements 212 can additionally include one or more first dynamic data elements, which can include, e.g., a first counter. Dynamic data elements such as the first counter can provide extra security, and can be useful in contexts where the first client device 204 is periodically or repeatedly authenticating with the server computer 208. For example, the server 25 computer 208 may (as a security feature) track the number of times that the first client device 204 requests issuance of one-time passwords, and dynamic data such as the first counter may reflect the number of times that the first client device 204 has requested one-time passwords from the server computer 208. As an alternative, “one-time passwords” generated by server computer 208 may comprise limited use 30 passwords, valid for e.g., three usages. The first counter may indicate the number of times that a one-time password (such as one-time password 218) has been used. Later, during authentication, the server computer 208 can evaluate the first counter (e.g., to verify it matches a counter stored on the server computer 208) and

increment the first counter. If the first counter exceeds a predetermined number of usages (e.g., three) then the server computer 208 may fail to authenticate the user 202.

[0079] At step S232, the first client device 204 can generate or otherwise form
5 a service request comprising the one or more first data elements 212. The service request can take any form that is capable of transmitting or otherwise communicating the one or more first data elements 212 to the server computer 208. The form of the service request can be determined by, e.g., an application or browser operating on the first client device 204, and can be formatted in any appropriate manner, e.g., as
10 a list of first data elements, such as an ordered list of one or more first user identifiers, one or more first intent identifiers, and one or more first dynamic data elements.

[0080] At step S234, the first client device 204 can transmit the service request to the server computer 208, which may comprise, e.g., an authentication
15 computer, such as authentication computer 114 from FIG. 1 (also sometimes referred to as a “one-time password management computer”). The first client device 204 can transmit the service request comprising the one or more first data elements 212 to the server computer 208 using any appropriate means. For example, if the first client device 204 is operating a browser application, HTTP “POST” or “GET”
20 methods can be used to transmit the service request to the server computer 208. As another example, if the first client device 204 is operating a secure online banking application, the secure online banking application may establish a secure and encrypted communication channel with the server computer 208, and the first client device 204 can transmit the service request to the server computer 208 via the
25 secure and encrypted communication channel. The first client device 204 can transmit the service request directly or via any number of relays, routers, firewalls, gateways, reverse proxies, etc.

[0081] Upon the server computer 208 receiving the service request from the first client device 204, at step S236 the server computer 208 can combine (e.g.,
30 concatenate) the one or more first data elements 212, thereby generating a first combination 214. This first combination 214 can depend on the number of first data elements included in the one or more first data elements 212. For example, in some

embodiments, the one or more first data elements 212 may comprise a single first user identifier, and as such the first combination 214 may comprise the single first user identifier. In other embodiments, the one or more first data elements 212 may comprise a first user identifier and a first intent identifier, and as such, the first
5 combination 214 may comprise a combination of the first user identifier and the first intent identifier. In other embodiments, the one or more first data elements 212 may comprise the first user identifier, the first intent identifier, and one or more first dynamic data elements including a first counter, and as such, the first combination
10 214 may comprise a combination of the first user identifier, the first intent identifier, the first counter, and one or more other first dynamic data elements.

[0082] As depicted in FIG. 2A, the exemplary first combination 214 comprises a concatenation of the user identifier "JohnDoe@example.com" and the intent identifier "userRegistration", i.e., "JohnDoe@example.com+userRegistration". However, there are a wide variety of ways that the server computer 208 can
15 generate the first combination 214 at step S236, and a brief list of examples is provided below. As described above, the server computer 208 can generate the first combination 214 by concatenating the one or more first data elements 212. As another example, the one or more first data elements 212 could be combined by interleaving alphanumeric characters defining each of the first data elements 212.
20 As yet another example, the one or more first data elements 212 can be combined by performing a bitwise exclusive-or (XOR) between the bits defining each first data element of the one or more first data elements 212. Other combination techniques may become apparent to one of ordinary skill in the art upon reading this disclosure.

[0083] At step S238, the server computer 208 can generate a first hash 216
25 by hashing the one or more first data elements 212. In some embodiments, hashing the one or more first data elements 212 can comprise hashing the first combination 214. The server computer 208 can use any appropriate hash function to hash the one or more first data elements 212 or the first combination 214. In some
30 embodiments, the hash function can comprise an MD hash function, an SHA hash function, a BLAKE hash function, an MAC based hash function, or a CRC based hash function. Later (e.g., in step S242), the server computer 208 can store a first one-time password 218 in a one-time password data structure 220 (e.g., a hash table) in association with the first hash 216.

[0084] At step S240, the server computer 208 can generate a first one-time password 218 (or more generically, a “one-time password” 218). The one time-password 218 may be referred to as a first one-time password 218, in order to distinguish it from a one-time password received from the first client device 204 or second client device 206 (e.g., at step S250 in FIG. 2B, which may be referred to as a “second one-time password” 222). Often, one-time passwords such as first one-time password 218 comprise six digit numbers (such as 123456, as shown in FIG. 2A). However, in embodiments of the present disclosure, one-time passwords such as first one-time password 218 can comprise numbers, alphanumeric strings, or any other appropriate data (e.g., a QR code). The server computer 208 can generate the first one-time password 218 at step S240 using any appropriate means. For example, the server computer 208 can randomly generate a number or an alphanumeric string and use the random number or random alphanumeric string as the first one-time password 218. Generating one-time passwords randomly may make it more difficult for cybercriminals to generate their own illegitimate one-time passwords.

[0085] Generating the first one-time password 218 can also involve generating supplemental data associated with the first one-time password 218. For example, the server computer 208 can generate a timestamp associated with the first one-time password 218. This timestamp can indicate the time at which the server computer 208 generated the first one-time password 218, which can be useful when later authenticating the user 202 using a received one-time password. For example, if the first one-time password 218 is only valid for fifteen minutes, then upon receiving the second one-time password 222 (which may be the same as the first one-time password 218) from the first client device 204, the server computer 208 could check the current time against the timestamp in order to determine if the second one-time password 222 is still valid. Other examples of data that can be generated along with the first one-time password 218 include a first one-time password type, or data that indicates the first one-time password type, such as the alphanumeric codes described above.

[0086] In some embodiments, the server computer 208 can generate the first one-time password 218 based on an intent corresponding to a first intent identifier received by the server computer 208 in the service request received at step S232.

In some cases, it may be preferable to have different authentication standards when attempting to access different services (e.g., privileged services versus unprivileged services). As such, the server computer 208 can determine an intent (e.g., indicating which service the user 202 intends to access) based on the first intent identifier and
5 generate the first one-time password 218 based on the intent. For example, if the server computer 208 determines that the user 202 intends to access a low-security or low-privilege service, the server computer 208 can generate a first one-time password 218 with a corresponding low-security one-time password type. The first one-time password 218 may comprise, e.g., a shorter alphanumeric string, and may
10 be valid for a longer period of time and/or a higher number of uses. By contrast, if the server computer 208 determines that the user 202 intends to access a high-security or high-privilege service, the server computer 208 can generate a first one-time password 218 with a corresponding high-security one-time password type. This first one-time password 218 may comprise, e.g., a longer alphanumeric string, and
15 may be valid for a shorter period of time and/or a single usage.

[0087] At step S242, the server computer 208 can store the first one-time password 218 in association with the first hash 216 in a one-time password data structure 220, which may be stored, e.g., in a one-time password database, such as one-time password database 122 from FIG. 1. In some embodiments, the server
20 computer 208 can store a one-time password record in a key-value table, wherein the first one-time password 218 comprises a value associated with the one-time password record and the first hash 216 comprises a key associated with the one-time password record. In some embodiments, the one-time password data structure 220 can comprise a hash table. Storing the first one-time password 218 in the one-
25 time password data structure 220 may enable the server computer 208 to later retrieve the one-time password 218 and compare it to a received second one-time password 222, in order to authenticate the user 202.

[0088] In some embodiments, the first one-time password 218 may be stored in addition to other data, such as a one-time password type, a timestamp, an
30 expiration limit (e.g., indicating the duration for which the one-time password is valid), a maximum usage count (e.g., indicating the total number of time that the one-time password 218 can be used before it expires), or any other appropriate data.

When authenticating the user 202, the server computer 208 can use this data in order to determine whether a received one-time password is still valid.

[0089] As described above with reference to the service system 108 of FIG. 1, storing one-time passwords in hash tables is more secure than plaintext storage
5 used in many conventional one-time password authentication systems, as non-reversible hashes can prevent cybercriminals or other malicious entities from gaining access to potentially sensitive data such as user identifiers or intent identifiers. Additionally, storing one-time passwords in hash tables is more efficient than encrypted storage techniques used in some conventional one-time password
10 authentication systems, as it eliminates the need to perform computationally expensive encryption and decryption operations. Further, the use of hashing enables methods according to embodiments to be used with a wide variety of user identifiers, intent identifiers, and other data.

[0090] Further, as described above, storing a one-time password in
15 association with a hash value generated using an intent identifier, in addition to a user identifier and other data, provides an efficient and built-in form of access control. A user such as user 202 cannot use a one-time password for multiple different services (with corresponding different intent identifiers) as it may result in a hash mismatch, leading to authentication failure. This is in contrast to some
20 conventional one-time password authentication systems, in which one-time passwords may be associated with singular identifiers, and which may require further forms of access control to prevent a user from using a one-time password generated for one service to access other services. Further, the use of hash functions eliminates the need to generate, issue, store, and protect potentially sensitive
25 information, as hashes can be generated during both an issuance phase (e.g., as described above) and during the authentication phase (described further below), eliminating the need for the server computer 208 to store potentially sensitive information such as user identifiers or intent identifiers in the interim.

[0091] At step S242, the server computer 208 can transmit the first one-time
30 password 218 to the first client device 204 or the second client device 206. Often, in one-time password authentication, a one-time password such as first one-time password 218 can be sent to a client device other than the client device that initiated

the service request or authentication request. For example, if a user 202 is attempting to log into a video on-demand streaming service using their laptop (e.g., a first client device 204), the video on-demand streaming service may send a text message containing the one-time password to the user's 202 smartphone (e.g., a second client device 206). In this way, the video on-demand streaming service can verify that the user 202 is in possession of two client devices associated with the user 202. However, the first one-time password 218 can also be transmitted directly back to the first client device 204. The first one-time password 218 may be transmitted in encrypted form, if necessary.

10 **[0092]** Later, during the authentication phase (described below with reference to FIG. 2B), the user 202 may provide a second one-time password 222 back to the server computer 208. If the user is legitimately attempting to authenticate with the server computer 208, this second one-time password 222 can be the first one-time password 218 received from the server computer 208 at step S244. The server
15 computer 208 may compare the second one-time password 222 against the first one-time password 218 stored in the one-time password data structure 220 to authenticate the user 202. The user 202 may be authenticated if the second one-time password 222 matches the first one-time password 218.

[0093] At any time during the issuance phase, the server computer 208 may
20 access the one-time password data structure 220 and manage (e.g., evaluate, add, or remove) one-time passwords or one-time password data records stored therein. For example, the server computer 208 can periodically evaluate the one-time passwords in the one-time time password data structure 220 in order to determine if those one-time passwords have expired or are no longer valid, e.g., by evaluating an
25 associated timestamp, associated expiration limit, a current time, etc. If the server computer 208 determines that a one-time password has expired, the server computer 208 can remove that one-time password (and any associated data, such as a timestamp and expiration limit) from the one-time password data structure 220. This may be useful because it may free up memory and one-time passwords,
30 enabling their use for other users.

[0094] Having issued the first one-time password 218 to the user 202 via either the first client device 204 or the second client device 206, the server computer

208 and one or more of the client devices can perform an authentication phase described below with reference to FIG. 2B. If the user 202 is successfully authenticated during the authentication phase, the server computer 208 (which may, in some embodiments comprise a first server computer 208) and/or a second server computer 210 can provide a service to the first client device 204 and/or the second client device 206 (e.g., at step S260).

[0095] If necessary, at step S246, the user 202 can acquire a second one-time password 222 from the second client device 206. This step may occur if, for example, the server computer 208 transmitted the first one-time password 218 to the second client device 206, rather than the first client device 204. For example, if the server computer transmitted the first one-time password 218 in a text message to a smartphone second client device 206, then the user 202 can acquire the second one-time password 222 from the second client device 206 by reading the second one-time password 222 off a display of the second client device 206. Ideally (i.e., in terms of successful authentication of the user 202, and not for methods according to embodiments generally), the second one-time password 222 comprises the first one-time password 218. However, for whatever reason, the second one-time password 222 may not match the first one-time password 218 (e.g., transmission error, user misinput, etc.). The terms first one-time password 218 and second one-time password 222 are generally intended to differentiate the one-time password generated by the server computer 208 and stored in a one-time password data structure 220 (i.e., the first one-time password 218) and the one-time password received from the first client device 204 (or second client device 206) and compared to the first one-time password 218 for the purpose of authenticating the user 202 (i.e., the second one-time password 222).

[0096] Similar to step S230, at step S248, the first client device 204 can collect or receive one or more second data elements 224, which can later be provided to the server computer 208 along with the second one-time password 222, in order to authenticate the user 202. The one or more second data elements 224 can comprise a second user identifier and a second intent identifier associated with the service request, which can be collected by the first client device 204 at step S248. Further, in some embodiments, the one or more second data elements 224 can include one or more second dynamic data elements including a second counter.

[0097] In some embodiments, the second user identifier can comprise one or more of a username, an account identifier, a user email address, a user phone number, a cookie, or a biometric. In some embodiments the first client device 204 can receive the second user identifier via inputs produced by the user 202 of the first client device 204, e.g., via a touch screen, keyboard, or other input device. Alternatively, the first client device 204 can retrieve the second user identifier from an application such as a browser operating on the first client device 204. In some embodiments, the second intent identifier can comprise one or more of a user registration intent identifier, a password recovery intent identifier, a contact update intent identifier, a login intent identifier, or a privileged service intent identifier. In some embodiments, the contact update intent identifier can comprise an email update intent identifier and/or a phone number update intent identifier. In some embodiments, the privileged service intent identifier can comprise a key management intent identifier. The first client device 204 can receive the second intent identifier via inputs produced by the user 202 of the first client device 204, or e.g., via a browser or other application running on the first client device 204.

[0098] Ideally (in the context of successful one-time password based authentication, and not for embodiments as a whole), the one or more second data elements 224 comprise the one or more first data elements 212, the second user identifier comprises the first user identifier, and the second intent identifier comprises the first intent identifier. If this is the case, the server computer 208 can generate a second combination 226 (e.g., at step S252) that is the same as the first combination 214 generated during the issuance phase (e.g., at step S236), which may enable the server computer 208 to identify a relevant first one-time password 218 stored in a one-time password data structure 220, enabling the server computer 208 to successfully authenticate the user 202 by comparing the first one-time password 218 to the second one-time password 222.

[0099] At step S250, a client device (e.g., the first client device 204 or the second client device 206) can transmit the second one-time password 222 and the one or more second data elements 224 to the server computer 208. As described above, the one or more second data elements 224 can comprise a second user identifier and a second intent identifier. The first client device 204 or second client device 206 can transmit the one or more second data elements 224 and the second

one-time password 222 to the server computer 208 using any appropriate means, e.g., via a browser or via another application operating on the first client device 204 or second client device 206.

[0100] Upon receiving the one or more second data elements 224 and the
5 second one-time password 222 from the first client device 204 (or the second client device 206), the server computer 208 can combine (e.g., concatenate) the one or more second data elements 224 at step S252, thereby generating a second combination 226. The second combination 226 can depend on the number of second data elements included in the one or more second data elements 224. For
10 example, the one or more second data elements 224 may comprise a single second user identifier, and as such, the second combination 226 may comprise the single second user identifier. In other embodiments, the one or more second data elements 224 may comprise a second user identifier and a second intent identifier, and as such, the second combination 226 may comprise a combination of the
15 second user identifier and the second intent identifier. In other embodiments, the one or more second data elements 224 may comprise the second user identifier, the second intent identifier, and one or more second dynamic data elements including a second counter, and as such, the second combination 226 may comprise a combination of the second user identifier, the second intent identifier, the second
20 counter, and one or more other second dynamic data elements. In some embodiments, combining the one or more second data elements 224 can comprise concatenating the one or more second data elements 224. However, other combination methods, such as those described above with reference to FIG. 2A, are also possible.

25 **[0101]** At step S254, the server computer 208 can generate a second hash 228 by hashing the one or more second data elements 224. In some embodiments, hashing the one or more second data elements 224 can comprise hashing the second combination 226. The server computer 208 can use any appropriate hash function to hash the second combination 226 or the one or more second data
30 elements 224. In some embodiments, the hash function can comprise an MD hash function, an SHA hash function, a BLAKE hash function, an MAC based hash function, or a CRC based hash function. In some embodiments, the second hash 228 may comprise the first hash 216, e.g., if the second combination 226 is the same

as the first combination 214, which may occur if, e.g., the one or more second data elements 224 are the same as the one or more first data elements 212.

[0102] At step S256, the server computer 208 can determine the first one-time password 218 (also referred to as a “stored one-time password”) using the second hash 228. In some embodiments, the server computer 208 can identify the a one-time password record stored in the one-time password data structure 220 using the second hash 228 as a key. The server computer 208 can then determine the first one-time password 218 using the one-time password data record. The server computer 208 can also determine or retrieve any other data that may be used in authenticating the user 202, such as a counter, expiration limit, timestamp, etc., as described above.

[0103] In some embodiments, if the server computer 208 does not identify a first one-time password 218 stored in the one-time password data structure 220, the server computer 208 can terminate the authentication phase. In some embodiments, the server computer 208 can transmit an error message to the first client device 204 or the second client device 206 indicating that the authentication phase failed.

[0104] At step S258, the server computer 208 can attempt to authenticate the user 202, e.g., by comparing the first one-time password 218 (also referred to as a “stored one-time password”) to the second one-time password 222 (also referred to as a “received one-time password” or more generically as a “one-time password”). If the one-time passwords match, the server computer 208 may successfully authenticate the user 202. During step S258, the server computer 208 may also perform other verification operations in order to authenticate the user 202. For example, if the first one-time password 218 is only valid for a certain number of uses (e.g., three), the server computer 208 can compare the value of a counter (e.g., the second counter or the first counter) against a total number of uses. If the value of the counter exceeds the total number of uses, the server computer 208 may deny the authentication attempt and transmit (to the first client device 204 or the second client device 206) an error message, error code, or other indicator to this effect, e.g., stating or indicating that the provided one-time password has expired.

[0105] At step S260, based on comparing the first one-time password 218 to the second one-time password 222, the server computer 208 can provide a service associated with the service request to the first client device 204 or the second client device 206. In this way, a client device (and by extension the user 202) can receive
5 the service provided by the server computer. In some embodiments, the server computer 208 can comprise a first server computer 208, which may not directly provide the service to the first client device 204 or second client device 206. Instead, the first server computer 208 may transmit access data enabling a client device to access the service provided by the second server computer 210. The second server
10 computer 210 may provide the service responsive to receiving the access data from the first client device 204 or the second client device 206.

[0106] At any time during the authentication phase, the server computer 208 may access the one-time password data structure 220 and manage (e.g., evaluate, add, or remove) one-time passwords or one-time password data records stored
15 therein. For example, once the first one-time password 218 has been compared to the second one-time password 222, the server computer 208 may evaluate whether the first one-time password 218 is still valid, and if no, the server computer 208 can delete the first one-time password 218 from the one-time password data structure 220.

[0107] A client device may be better understood with reference to FIG. 3 which shows an exemplary client device 300 comprising a processor 302, a communications interface 304, a user interface 306, and a computer readable medium 308. The computer readable medium 308 may be non-transitory and coupled to the processor 302. The computer readable medium 308 may contain
25 data, code, and/or software modules, which may be used by the client device 300 to implement some methods according to embodiments. These data, code, and/or software modules may include a communications module 310, a browser 312, a service application 314, user identifier(s) 316, intent identifier(s) 318, dynamic data element(s) 320, and one-time password 322. It should be understood that the
30 particular software modules were chosen primarily for the purpose of explaining some methods, steps, or operations according to embodiments, and that FIG. 3 shows only one of a large number of valid client device 300 configurations.

[0108] Processor 302 may comprise any suitable data computation device or devices. Processor 302 may be able to interpret code and carry out instructions stored on computer readable medium 308. Processor 302 may comprise a Central Processing Unit (CPU) operating on a reduced instructional set, and may comprise a
5 single or multi-core processor. Processor 302 may also include an Arithmetic Logic Unit (ALU) and a cache memory.

[0109] Communications interface 304 may comprise any interface by which client device 300 can communicate with other devices or computers, such as a server computer. Examples of communications interfaces include wired interfaces,
10 such as USB, Ethernet, or FireWire, as well as wireless interfaces such as Bluetooth or Wi-Fi receivers. Client device 300 may possess multiple communications interfaces 304. As an example, a client device 300 may communicate through a wireless cellular interface as well as a USB port.

[0110] The client device 300 can also comprise a user interface 306, which
15 can comprise an interface through which the client device 300 can receive user input. For example, the user interface can comprise a touch screen or keyboard enabling the user to input e.g., user identifier(s) 316, intent identifier(s) 318, or a one-time password 322 to the client device 300.

[0111] Communications module 310 may comprise code, software or
20 instructions that may be interpreted and executed by processor 302. This software may be used by the client device 300 to communicate with other computers, devices, and entities, such as a server computer or other computers and devices in, e.g., a service system such as service system 108 in FIG. 1. Particularly, the client device 300 can use communications module 310 to transmit service requests to a server
25 computer (comprising, e.g., user identifier(s) 316, intent identifier(s) 318, and dynamic data element(s) 320), receive one-time passwords from the server computer, and transmit one-time passwords to the server computer.

[0112] Browser 312 may comprise a web-browser, an application that can be used to access websites. Browser 312 may be used by client device 300 in order to
30 access web-based services provided by a server computer, and may be used by the client device 300 to communicate with the server computer (e.g., in conjunction with communications module 310).

[0113] Service application 314 may comprise an application used by the client device 300 to communicate with a server computer (e.g., in conjunction with communications module 310 and/or browser 312). For example, for a server computer associated with an online banking service, service application 314 could
5 comprise an online banking application, used by a user operating client device 300 in order to access that service.

[0114] User identifier(s) 316 may be provided by the client device 300 to a server computer during a one-time password based authentication method, e.g., as described above. These user identifier(s) 316 may be stored on the computer
10 readable medium 308. In some cases, these user identifier(s) 316 may be stored in association with the browser 312 or service application 314, e.g., as cookies associated with browser 312. Likewise, intent identifier(s) 318 may be provided by the client device 300 to a server computer during authentication, and may be stored on the computer readable medium 308, e.g., in association with either the browser
15 312 or service application 314. Similarly, dynamic data elements 320 may comprise data elements, such as counters or timestamps, that may be provided to a server computer during authentication, which may be stored on the computer readable medium 308, e.g., in association with browser 312 or service application 314. The computer readable medium 308 can likewise store a one-time password 322, which
20 can be provided to a server computer in order to authenticate a user of the client device 300, and which may be stored in association with the browser 312 or service application 314. The user identifier(s) 316, intent identifier(s) 318, dynamic data element(s) 320, and one-time password 322 may be stored in secure memory, if necessary.

[0115] A server computer may be better understood with reference to FIG. 4
25 which shows an exemplary server computer 400 comprising a processor 402, a communications interface 404, and a computer readable medium 406. The computer readable medium 406 may be non-transitory and coupled to the processor 402. The computer readable medium 406 may contain data, code, and/or software
30 modules, which may be used by the server computer 400 to implement some methods according to embodiments. These data, code, and/or software modules may include a communications module 408, a combination module 410, a hashing module 412, one-time password generation module 414, one-time password data

structure 416, data structure management module 418, one-time password comparison module 420, and service module 422. It should be understood that the particular software modules were chosen primarily for the purpose of explaining some methods, steps, or operations according to embodiments, and that FIG. 4
5 shows only one of a large number of valid server computer 400 configurations.

[0116] Processor 402 may comprise any suitable data computation device or devices. Processor 402 may be able to interpret code and carry out instructions stored on computer readable medium 406. Processor 402 may comprise a Central Processing Unit (CPU) operating on a reduced instructional set, and may comprise a
10 single or multi-core processor. Processor 402 may also include an Arithmetic Logic Unit (ALU) and a cache memory.

[0117] Communications interface 404 may comprise any interface by which server computer 400 can communicate with other devices or computers, such as a client device. Examples of communications interfaces include wired interfaces, such
15 as USB, Ethernet, or FireWire, as well as wireless interfaces such as Bluetooth or Wi-Fi receivers. Server computer 400 may possess multiple communications interfaces 404. As an example, a server computer 400 may communicate through an ethernet interface as well as a USB port.

[0118] Communications module 408 may comprise code, software or
20 instructions that may be interpreted and executed by processor 402. This software may be used by the server computer 400 to communicate with other computers, devices, and entities, such as a client device or other computers and devices in, e.g., a service system such as service system 108 in FIG. 1. Particularly, the server computer 400 can use communications module 408 to receive service requests from
25 a client device (comprising one or more data elements such as user identifier(s), intent identifier(s), and dynamic data element(s)). Further, the server computer 400 can use communications module 408 to transmit one-time passwords to client devices as well as receive one-time passwords from client devices.

[0119] Combination module 410 may comprise code, software, or instructions
30 that may be interpreted and executed by processor 402. This software may be used by the server computer 400 to combine data elements received from client devices, such as user identifiers and intent identifiers using techniques such as

concatenation. Likewise, hashing module 412 can comprise code, software, or instructions that may be interpreted or executed by processor 402 for generating hashes using a hash function and combined data elements (which may be generated using combination module 410). One-time password generation module 414 may
5 comprise code or instructions, executable by the processor 402 for generating one-time passwords, which may include software routines or methods for generating random numbers.

[0120] One-time password data structure 416 may comprise a data structure (such as a database, data table, or hash table) used to store generated one-time
10 passwords in association with their respective hashes. The server computer 400 may use a data structure management module 418 in order to manage one-time password data structure 416, e.g., in order to identify, add or remove one-time passwords from one-time password data structure 416. Although one-time password data structure 416 is showed stored on the computer readable medium
15 406, in some embodiments, the one-time password data structure 416 can be implemented outside the computer readable medium 406, e.g., in an external one-time password database, such as one-time password database 122 in FIG. 1. The server computer 400 can use one-time password comparison module 420 to compare received one-time passwords to stored one-time passwords in the one-time
20 password data structure 416, e.g., for the purpose of authenticating users of client devices. In some cases, the server computer 400 can use a service module 422 in order to provide a service to a user via their client device. For example, server computer 400 could use service module 422 in order to stream video data to a client device as part of a video on-demand streaming service.

25 **[0121]** Any of the computer systems mentioned herein may utilize any suitable number of subsystems. In some embodiments, a computer system includes a single computer apparatus, where the subsystems can be components of the computer apparatus. In other embodiments, a computer system can include multiple computer apparatuses, each being a subsystem, with internal components.

30 **[0122]** A computer system can include a plurality of the components or subsystems, e.g., connected together by external interface or by an internal interface. In some embodiments, computer systems, subsystems, or apparatuses

can communicate over a network. In such instances, one computer can be considered a client and another computer a server, where each can be part of a same computer system. A client and a server can each include multiple systems, subsystems, or components.

5 **[0123]** It should be understood that any of the embodiments of the present invention can be implemented in the form of control logic using hardware (e.g., an application specific integrated circuit or field programmable gate array) and/or using computer software with a generally programmable processor in a modular or integrated manner. As used herein a processor includes a single-core processor,
10 multi-core processor on a same integrated chip, or multiple processing units on a single circuit board or networked. Based on the disclosure and teachings provided herein, a person of ordinary skill in the art will know and appreciate other ways and/or methods to implement embodiments of the present invention using hardware and a combination of hardware and software.

15 **[0124]** Any of the software components or functions described in this application may be implemented as software code to be executed by a processor using any suitable computer language such as, for example, Java, C, C++, C#, Objective-C, Swift, or scripting language such as Perl or Python using, for example, conventional or object-oriented techniques. The software code may be stored as a
20 series of instructions or commands on a computer readable medium for storage and/or transmission, suitable media include random access memory (RAM), a read only memory (ROM), a magnetic medium such as a hard-drive or a floppy disk, or an optical medium such as a compact disk (CD) or DVD (digital versatile disk), flash memory, and the like. The computer readable medium may be any combination of
25 such storage or transmission devices.

[0125] Such programs may also be encoded and transmitted using carrier signals adapted for transmission via wired, optical, and/or wireless networks conforming to a variety of protocols, including the Internet. As such, a computer readable medium according to an embodiment of the present invention may be
30 created using a data signal encoded with such programs. Computer readable media encoded with the program code may be packaged with a compatible device or provided separately from other devices (e.g., via Internet download). Any such

computer readable medium may reside on or within a single computer product (e.g. a hard drive, a CD, or an entire computer system), and may be present on or within different computer products within a system or network. A computer system may include a monitor, printer or other suitable display for providing any of the results
5 mentioned herein to a user.

[0126] Any of the methods described herein may be totally or partially performed with a computer system including one or more processors, which can be configured to perform the steps. Thus, embodiments can be involve computer systems configured to perform the steps of any of the methods described herein,
10 potentially with different components performing a respective steps or a respective group of steps. Although presented as numbered steps, steps of methods herein can be performed at a same time or in a different order. Additionally, portions of these steps may be used with portions of other steps from other methods. Also, all or portions of a step may be optional. Additionally, and of the steps of any of the
15 methods can be performed with modules, circuits, or other means for performing these steps.

[0127] The specific details of particular embodiments may be combined in any suitable manner without departing from the spirit and scope of embodiments of the invention. However, other embodiments of the invention may be involve specific
20 embodiments relating to each individual aspect, or specific combinations of these individual aspects. The above description of exemplary embodiments of the invention has been presented for the purpose of illustration and description. It is not intended to be exhaustive or to limit the invention to the precise form described, and many modifications and variations are possible in light of the teaching above. The
25 embodiments were chosen and described in order to best explain the principles of the invention and its practical applications to thereby enable others skilled in the art to best utilize the invention in various embodiments and with various modifications as are suited to the particular use contemplated.

[0128] The above description is illustrative and is not restrictive. Many
30 variations of the invention will become apparent to those skilled in the art upon review of the disclosure. The scope of the invention should, therefore, be determined not with reference to the above description, but instead should be

determined with reference to the pending claims along with their full scope or equivalents.

[0129] One or more features from any embodiment may be combined with one or more features of any other embodiment without departing from the scope of the invention.

[0130] A recitation of “a”, “an” or “the” is intended to mean “one or more” unless specifically indicated to the contrary. The use of “or” is intended to mean an “inclusive or,” and not an “exclusive or” unless specifically indicated to the contrary.

[0131] All patents, patent applications, publications and description mentioned herein are incorporated by reference in their entirety for all purposes. None is admitted to be prior art.

WHAT IS CLAIMED IS:

1 1. A method comprising:
2 receiving, by a server computer from a first client device, a service
3 request comprising one or more first data elements, wherein the one or more first
4 data elements include a first user identifier;
5 hashing, by the server computer, the one or more first data elements to
6 form a first hash using a hash function;
7 generating, by the server computer, a first one-time password;
8 storing, by the server computer, the first one-time password in
9 association with the first hash;
10 transmitting, by the server computer, the first one-time password to the
11 first client device or a second client device;
12 receiving, by the server computer from the first client device or the
13 second client device, a second one-time password and one or more second data
14 elements, wherein the one or more second data elements include a second user
15 identifier;
16 hashing, by the server computer, the one or more second data
17 elements to form a second hash using the hash function;
18 determining, by the server computer, the first one-time password using
19 the second hash;
20 comparing, by the server computer, the first one-time password to the
21 second one-time password; and
22 based on comparing the first one-time password to the second one-
23 time password, providing, by the server computer, a service associated with the
24 service request to the first client device.

1 2. The method of claim 1, wherein the one or more first data
2 elements includes one or more first dynamic data elements including a first counter,
3 and wherein the one or more second data elements includes one or more second
4 dynamic data elements including a second counter.

1 3. The method of claim 1, wherein the hash function comprises an
2 MD hash function, an SHA hash function, a BLAKE hash function, an MAC based
3 hash function, or a CRC based hash function.

1 4. The method of claim 1, wherein the first one-time password
2 comprises an alphanumeric string, and wherein generating the first one-time
3 password comprises randomly generating, by the server computer, the alphanumeric
4 string.

1 5. The method of claim 1, wherein:
2 prior to hashing the one or more first data elements, the method further
3 comprises combining, by the server computer, the one or more first data elements,
4 thereby generating a first combination, wherein hashing the one or more first data
5 elements comprises hashing the first combination; and
6 prior to hashing the one or more second data elements, the method
7 further comprises combining, by the server computer, the one or more second data
8 elements, thereby generating a second combination, wherein hashing the one or
9 more second data elements comprises hashing the second combination.

1 6. The method of claim 5, wherein:
2 combining the one or more first data elements comprises
3 concatenating, by the server computer, the one or more first data elements; and
4 combining the one or more second data elements comprises
5 concatenating, by the server computer, the one or more second data elements.

1 7. The method of claim 1, wherein storing the first one-time
2 password in association with the first hash comprises storing, by the server
3 computer, a one-time password record in a key-value table, wherein the first one-
4 time password comprises a value associated with the one-time password record and
5 the first hash comprises a key associated with the one-time password record.

1 8. The method of claim 7, wherein determining the first one-time
2 password using the second hash comprises:
3 identifying, by the server computer, the one-time password record
4 using the second hash as a key; and
5 determining, by the server computer, the first one-time password using
6 the one-time password record.

1 9. The method of claim 1, wherein the one or more first data
2 elements include a first intent identifier associated with the service request and
3 wherein the one or more second data elements include a second intent identifier
4 associated with the service request.

1 10. The method of claim 9, wherein the method further comprises
2 determining, by the server computer, an intent based on the first intent identifier, and
3 wherein the server computer generates the first one-time password based on the
4 first intent identifier.

1 11. The method of claim 9, wherein the one or more first data
2 elements and the one or more second data elements are the same.

1 12. The method of claim 9, wherein the first user identifier and/or the
2 second user identifier comprises one or more of the following:
3 a username;
4 an account identifier;
5 a user email address;
6 a user phone number;
7 a cookie; or
8 a biometric.

1 13. The method of claim 9, wherein the first intent identifier and/or
2 the second intent identifier comprises one or more of the following:
3 a user registration intent identifier;
4 a password recovery intent identifier;
5 a contact update intent identifier;
6 a login intent identifier; or
7 a privileged service intent identifier.

1 14. The method of claim 13, wherein the one-time password is
2 transmitted to the second client device.

1 15. The method of claim 14, wherein the first client device is a
2 laptop computer and the second client device is a mobile phone.

1 16. A server computer comprising:
2 a processor; and
3 a non-transitory computer readable medium coupled to the processor,
4 the non-transitory computer readable medium comprising code or instructions,
5 executable by the processor for performing a method comprising:
6 receiving from a first client device, a service request comprising one or
7 more first data elements, wherein the one or more first data elements include a first
8 user identifier;
9 hashing the one or more first data elements to form a first hash using a
10 hash function;
11 generating a first one-time password;
12 storing the first one-time password in association with the first hash;
13 transmitting the first one-time password to the first client device or a
14 second client device;
15 receiving, from the first client device or the second client device, a
16 second one-time password and one or more second data elements, wherein the one
17 or more second data elements include a second user identifier;
18 hashing the one or more second data elements to form a second hash
19 using the hash function;
20 determining the first one-time password using the second hash;
21 comparing the first one-time password to the second one-time
22 password; and
23 based on comparing the first one-time password to the second one-
24 time password, providing a service associated with the service request to the first
25 client device.

1 17. A method comprising:
2 transmitting, by a client device to a server computer, a service request
3 comprising one or more data elements, wherein the one or more data elements
4 include a user identifier, wherein the server computer forms a first hash using a hash
5 function and the one or more data elements, wherein the server computer generates
6 a one-time password, and wherein the server computer stores the one-time
7 password in association with the first hash;

8 receiving, by the client device from the server computer, the one-time
9 password;

10 transmitting, by the client device to the server computer, the one-time
11 password and the one or more data elements, wherein the server computer forms a
12 second hash using the hash function and the one or more data elements, wherein
13 the server computer determines a stored one-time password using the second hash,
14 and wherein the server computer compares the stored one-time password to the
15 one-time password; and

16 receiving, by the client device, a service provided by the server
17 computer and corresponding to the service request, wherein the server computer
18 provides the service to the client device based on comparing the stored one-time
19 password to the one-time password.

1 18. The method of claim 17, wherein the server computer comprises
2 a first server computer, and wherein receiving the service provided by the server
3 computer comprises:

4 receiving, by the client device from the first server computer, access
5 data enabling the client device to access the service; and

6 transmitting, by the client device, the access data to a second server
7 computer in order to access the service.

1 19. The method of claim 17, further comprising, prior to transmitting
2 the service request to the server computer:

3 accessing, by the client device, the user identifier via a browser or
4 another application operating on the client device.

1 20. The method of claim 17, further comprising, prior to transmitting
2 the service request to the server computer:

3 receiving, by the client device, the user identifier via user input
4 produced by a user of the client device.

1

1