METHODS AND APPARATUS FOR INCREASED SECURITY IN ISSUING APPLICATION TOKENS

In some embodiments, an apparatus includes an authorization module implemented in at least one of a memory or a processing device. The authorization module receives at a first time and from a first mobile application, a request for an access token associated with a second mobile application that includes an identifier associated with the second mobile application and a first random verification identifier. The authorization module provides to the first mobile application a signal representing an authorization code associated with the access token. The authorization module receives from the second mobile application at a second time a signal representing the authorization code, the identifier associated with the second mobile application, and a second random verification identifier. The authorization module provides a signal representing the access token to the second mobile application based in part on the first random verification identifier being equal to the second random verification identifier.
METHODS AND APPARATUS FOR INCREASED SECURITY IN ISSUING APPLICATION TOKENS

Related Application

[1001] This application is a continuation of and claims priority to U.S. Patent Application No. 13/737,525, filed January 9, 2013, entitled "METHODS AND APPARATUS FOR INCREASED SECURITY IN ISSUING APPLICATION TOKENS," the disclosure of which is incorporated herein by reference in its entirety.

Background

[1002] Some embodiments described herein relate generally to methods and apparatus for an authorization server to authenticate an application installed on a mobile communication device before issuing a token for the application. In particular, but not by way of limitation, some embodiments described herein relate to methods and apparatus of authenticating an application by using a randomly-generated verification identifier unique to the application before issuing a token for the application.

[1003] Open Authorization (OAuth) is an open standard protocol for authorization, and allows a user, such as an enterprise employee, to grant a third party application access to information associated with that user stored at a given location (e.g., on a given website), without sharing that user's account credentials (e.g., password) or the full extent of that user's data. Some known systems use OAuth tokens to authenticate applications for users of a variety of communication devices (e.g., a laptop computer, a personal digital assistant (PDA), a standard mobile telephone, a tablet personal computer (PC), etc.). If the OAuth Authorization Server is unable to authenticate an application before issuing a token, malicious software may be able to obtain the token issued for a valid application and use that token in making application programming interface (API) calls and impersonate the valid application.
Accordingly, a need exists for methods and apparatus for an OAuth Authorization Server to authenticate an application installed on a mobile communication device before issuing a token to that application.

**Summary**

In some embodiments, an apparatus includes an authorization module implemented in at least one of a memory or a processing device. The authorization module receives at a first time and from a first mobile application, a request for an access token associated with a second mobile application that includes an identifier associated with the second mobile application and a first random verification identifier. The authorization module provides to the first mobile application a signal representing an authorization code associated with the access token. The authorization module receives from the second mobile application at a second time a signal representing the authorization code, the identifier associated with the second mobile application, and a second random verification identifier. The authorization module provides a signal representing the access token to the second mobile application based in part on the first random verification identifier being equal to the second random verification identifier.

**Brief Description of the Drawings**

FIG. 1 is a schematic illustration of an access token issuing system, according to an embodiment.

FIG. 2 is a message flow diagram illustrating a method for approving a user of an application on a device and issuing the application an access token to access application data, according to an embodiment.

FIG. 3 is a flow chart illustrating a method for issuing an access token for an application by an authorization module, according to an embodiment.

**Detailed Description**

In some embodiments, an apparatus includes an authorization module implemented in at least one of a memory or a processing device. The authorization module is configured to
receive at a first time and from a first mobile application, a request for an access token associated with a second mobile application. The request includes an identifier associated with the second mobile application and a first random verification identifier. The authorization module is configured to provide to the first mobile application and in response to the request a signal representing an authorization code associated with the access token. The authorization module is configured to receive from the second mobile application and at a second time after the first time a signal representing the authorization code, the identifier associated with the second mobile application, and a second random verification identifier. The authorization module is configured to provide a signal representing the access token to the second mobile application based at least in part on the first random verification identifier being equal to the second random verification identifier.

[1010] In some embodiments, a non-transitory processor-readable medium includes code to cause a processor to receive at a first time and from a first mobile application a request for an access token associated with a second mobile application. The request includes an identifier associated with the second mobile application and a verification identifier generated by the second mobile application. The code causes the processor to provide to the first mobile application a signal representing an authorization code associated with the access token in response to the request. The code also causes the processor to receive from the second mobile application and at a second time after the first time a signal representing the authorization code, the identifier associated with the second mobile application, and the verification identifier. The code further causes the processor to provide a signal representing the access token to the second mobile application when the authorization code, the identifier associated with the second mobile application, and the verification identifier are verified.

[1011] In some embodiments, an apparatus includes a first mobile application implemented in at least one of a memory or a processing device. The first mobile application is configured to generate a random verification identifier and initiate a second mobile application such that the second mobile application sends a signal representing the random verification identifier and an identifier associated with the first mobile application to an authorization module at a first time. The first mobile application is configured to receive a signal representing an authorization code provided by the authorization module based on the authorization code being within an address associated with the first mobile application at a second time after the first time. The first mobile
application is configured to provide a signal representing the random verification identifier and the authorization code to the authorization module. The first mobile application is configured to receive a signal representing an access token associated with the first mobile application based at least in part on the authorization module validating the random verification identifier and the authorization code.

[1012] As used in this specification, the singular forms "a," "an" and "the" include plural referents unless the context clearly dictates otherwise. Thus, for example, the term "a device" is intended to mean a single device or a combination of devices.

[1013] FIG. 1 is a schematic illustration of an access token issuing system 100, according to an embodiment. The access token issuing system 100 includes a device 110, a network 120, an authorization server 130 and a resource server 150. The network 120 can be any type of network (e.g., a local area network (LAN), a wide area network (WAN), a virtual network, a telecommunications network) implemented as a wired network and/or wireless network. As described in further detail herein, in some embodiments, for example, the device 110 can be connected to the authorization sen' er 130 and the resource server 150 via an intranet, an Internet Service Provider (ISP) and the Internet, a cellular network (e.g., network 120), and/or the like.

[1014] The device 110 can be any mobile communication or computing device such as a laptop computer, a personal digital assistant (PDA), a standard mobile telephone, a tablet personal computer (PC), and/or so forth. The device 110 includes a memory 112 and a processor 111. The device 110 can be configured to generate random verification identifiers, and request and receive access tokens from the authorization server 130. The device 110 can also be configured to request application data from the resource server 150 (e.g., by sending access tokens to the resource sen' er 150) and to receive application data from the resource server 150 via the network 120 as described in further detail herein.

[1015] The memory 112 can be, for example, a random access memory (RAM), a memory buffer, a hard drive, a database, an erasable programmable read-only memory (EPROM), an electrically erasable read-only memory (EEPROM), a read-only memory (ROM) and/or so forth. In some embodiments, the memory 112 stores instructions to cause the processor to execute modules, processes and/or functions associated with such an access token issuing system 100.
The processor 111 can be a general purpose processor, a Field Programmable Gate Array (FPGA), an Application Specific Integrated Circuit (ASIC), a Digital Signal Processor (DSP), and/or the like. The processor 111 can be configured to run and/or execute applications and/or other modules, processes and/or functions associated with an access token issuing system 100. The processor 111 includes a browser 113 and an application 114.

The application 114 can be any native mobile application installed on the device 110 or a web-browser-based application. The application 114, for example, can be configured to receive access tokens from the authorization module 140 and cause the processor 111 to execute specific operations associated with particular functions in an enterprise. For example, the application 114 can be an accounting application, a sales application, a payroll application and/or the like. In other configurations, application 114 can be any other enterprise or third party application configured to be run and/or executed at the device 110.

A mobile application 114 can be configured to generate a random verification identifier (e.g., a string of random or pseudo-random numbers such as a cryptographic nonce) and initiate another mobile application such as, for example, the browser 113 such that this other mobile application can send a signal representing the random verification identifier and an identifier associated with the application 114 to an authorization module 140 at a first time. Note that the browser 113 is a specific example of a native mobile application in the specific instance described herein. In other instances, the browser 113 need not be considered as a native mobile application but instead can be considered as and functions as a browser independent of application 114. The application 114 can be configured to receive a signal representing an authorization code provided by the authorization module 140 based on the authorization code being within an address associated with the application 114 at a second time after the first time. The application 114 can provide a signal representing the random verification identifier and the authorization code to the authorization module 140. The application 114 is also configured to receive a signal representing an access token associated with the application 114 based in part on the authorization module 140 validating the random verification identifier and the authorization code.

The browser 113 can be any native mobile browser installed on the device 110 such as, for example, the Android browser, the Blackberry browser, Safari, Internet Explorer Mobile
browser, Firefox for mobile browser, and/or the like. The browser 113 can connect to the Internet via, for example, a cellular network and/or a wireless local area network (LAN), using standard Hypertext Transfer Protocol (HTTP) over Transmission Control Protocol/Internet Protocol (TCP/IP), and can display web pages written in, for example, HyperText Markup Language (HTML), Extensible HyperText Markup Language (XHTML Mobile Profile WAP 2.0), Wireless Markup Language (WML), and/or the like.

[1020] The browser 113 can be configured to be initiated by the application 114 and send a request signal for an access token associated with the application 114 to an authorization module 140 at a first time. The request signal can include the random verification identifier generated by the application 114 and an identifier associated with the application 114. The browser 113 can also be configured to send to the authorization module 140 a signal representing approval to provide an access token to the application 114 before the authorization module 140 provides an authorization code associated with the access token to the browser 113.

[1021] The authorization server 130 can be, for example, a web server, an application server, a proxy server, a telnet server, a file transfer protocol (FTP) server, a mail server, a list server, a collaboration server and/or the like. The authorization server 130 can receive from a first mobile application such as, for example, the browser 113 in the device 110, a signal requesting an access token associated with a second mobile application such as, for example, the application 114 in the device 110. The authorization server 130 can send a signal representing an authorization code associated with the access token to the first mobile application in response to the request signal via the network 120. The authorization server 130 can also execute modules, processes and/or functions associated with such an access token issuing system 100. The authorization server 130 can be, for example, an OAuth 2.0 authorization server. OAuth 2.0 defines a standardized messaging protocol by which a native application installed on a mobile communication device can obtain security tokens such as, for example, access tokens from an authorization server. OAuth 2.0 is described in detail in "The OAuth 2.0 Authorization Framework", dated October 2012, which is incorporated herein by reference in its entirety. These security tokens can then be included by the application on its application programming interface (API) calls to a resource server such as resource server 150 to authenticate before obtaining application data.
The authorization server 130 includes a processor 134 and a memory 132. The memory 132 can be, for example, a random access memory (RAM), a memory buffer, a hard drive, a database, an erasable programmable read-only memory (EPROM), an electrically erasable read-only memory (EEPROM), a read-only memory (ROM) and/or so forth. The memory 132 can store instructions to cause the processor 134 to execute modules, processes and/or functions associated with such an access token issuing system 100 (e.g., authorization module 140).

The processor 134 can be a general purpose processor, a Field Programmable Gate Array (FPGA), an Application Specific Integrated Circuit (ASIC), a Digital Signal Processor (DSP), and/or the like. In some configurations, processor 134 can be configured to run and/or execute application modules, processes and/or functions associated with such an access token issuing system 100. For example, processor 134 can be configured to run and/or execute modules, processes, and/or functions associated with or in response to access token requests (e.g., authorization module 140). As shown the processor 134 includes an authorization module 140.

The authorization module 140 can be a hardware and/or software module (stored in memory 132 and/or executed in a processor 134) that can receive at a first time and from a first mobile application such as, for example, the browser 113 in the device 110, a signal requesting an access token associated with a second mobile application such as, for example, the application 114 in the device 110. In some instances, the browser 113 can part of another native mobile application. In the context of the access token issuing system 100 described in FIG. 1, however, the browser 113 is considered as a specific example of a native mobile application. The request signal can include an identifier associated with the second mobile application, and a first random verification identifier. Upon receiving the request signal, the authorization module 140 can send a signal via the network 120 to the browser 113 to provide user approval (or authentication) information. Upon receiving the user approval information from the browser 113, the authorization module 140 can authenticate the user as described in greater detail herein. Upon successful authentication of the user, the authorization module 140 can send a signal representing an authorization code associated with the access token to the first mobile application (browser 113) in response to the request signal via the network 120. As described in further detail herein, the first mobile application (browser 113) can send the signal representing the authorization code
to the second mobile application (application 114) via, for example, the operating system of the
device 110. The authorization module 140 can receive, from the second mobile application and
at a second time after the first time, a signal representing the authorization code, the identifier
associated with the second mobile application, and a second random verification identifier. The
authorization module 140 can be configured to generate and/or define access tokens for the
application 114 on the device 110. The authorization module 140 can provide a signal
representing the access token to the second mobile application based in part on the first random
verification identifier being equal to the second random verification identifier, as described in
further detail herein. The authorization module 140 can also be configured to not provide the
first random verification identifier to the first mobile application or the second mobile
application for maintaining security.

[1025] The resource server 150 can either be a server associated with the enterprise (such as
a company) that is associated with the authorization server 130 or a third party server distinct
from the enterprise that is associated with the authorization server 130. The resource server 150
can be, for example, a Software as a Service (SaaS) host that is a web server, a proxy server, a
telnet server, a file transfer protocol (FTP) server, a mail server, a list server, a collaboration
server, and/or the like.

[1026] The resource server 150 includes a processor 154 and a memory 152. The memory
152 can be, for example, a random access memory (RAM), a memory buffer, a hard drive, a
database, an erasable programmable read-only memory (EPROM), an electrically erasable read-
only memory (EEPROM), a read-only memory (ROM) and/or so forth. In some configurations,
the memory 152 can store instructions to cause the processor 154 to execute modules, processes
and/or functions associated with the application server 150 within the access token issuing
system 100 (e.g., resource module 156).

[1027] The resource module 156 can be a hardware and/or software module (stored in
memory 152 and/or executed in a processor 154) configured to provide application data to
applications 114 via the network 120. For example, the application data can include results of
external market analysis for a product of the enterprise and/or the like. This application data
can be used by the application 114, for example, to set the retail price of the product, to calculate the
expected profit levels for the product, to determine locations for the best retail vendors available
for the product, and/or the like. In other words, the application data can be related to the type of application: sales data for a sales-related application, financial data for a financial application, etc. The resource module 156 can receive and process an access token from the application 114 of the device 110 prior to providing application data to the application 114. Such application data can enable the application 114 to execute modules, processes and/or functions associated with the device 110.

1028 FIG. 2 is a message flow diagram illustrating a method 200 for approving a user of an application 114 on a device 110 and issuing the application 114 an access token to access application data in an access token issuing system 100, according to an embodiment. This embodiment includes the application 114 generating a unique verification ID and adding the verification ID to both (1) the request for the authorization code associated with the access token and (2) when exchanging the authorization code for the access token. At 210, the application 114 generates a unique verification ID at the beginning of a communication session. The verification ID is used by the application 114 to authenticate itself to the authorization server 140 before obtaining the access token from the authorization server 140. The verification ID can be, for example, a string of arbitrary random or pseudo-random numbers (e.g., a cryptographic nonce) issued in step 210 to ensure that this verification ID is not reused in any replay attacks. To ensure that the verification ID is used only once, the verification ID can be made to be time-variant by including a suitably fine-grained timestamp in its value or can be generated with enough random bits to ensure a probabilistically insignificant chance of repeating a previously generated value.

1029 At 212, the application 114 initiates the browser 113 by launching the browser 113 and sends a signal representing the verification ID and a client ID associated with the application 114 to the browser 113. Upon launching, the application 114 can send a signal representing a URL associated with the address of the authorization module 140 to the operating system of the mobile device 110. The operating system can then launch the browser 113 to fetch the URL associated with the address of the authorization module 140.

1030 The client ID can include an encrypted or an unencrypted form of an application identity number that has been assigned to the application 114 (by the authorization module 140). In some instances, the client ID can also include, for example, additional encrypted or
unencrypted application credentials such as the version of the application 114, the features of the application 114 that are accessible to the user of the device 110, a device 110 identifier, an identifier associated with the user of the device 110, and/or the like.

At 214, the browser sends a signal requesting the access token for the application 114 to the authorization module 140 via the network 120. The request signal can include the verification ID and the client ID associated with the application 114. Upon receiving the request signal, the authorization module 140 can store the association between the first verification ID and the client ID. Upon receiving the request signal, the authorization module 140 can also send a signal via the network 120 to the browser 113 to request user approval (or authentication) information, at 216. Such user approval request signal can, for example, redirect the browser 113 to an OAuth authorization server 140 login page where the user can enter the information used for authentication of the user with respect to the OAuth authorization server 140.

At 218, the browser 113 sends a signal that contains user approval information to the authorization module 140 via the network 120. The user approval (or authentication) information can include, for example, an identifier unique to each user of the device 110 using the OAuth 2.0 protocol. The identifier associated with each user can include, for example, a user login, a user password, an employee ID number, a personal identification number (PIN), biometric information (e.g., data associated with a fingerprint scan, a retinal scan, etc.), an employee security/access level, an enterprise division code, and/or the like. Note that in some instances, the identifier can be unique to the user and the application 114. For example, a specific user can have different passwords for different applications installed on the device 110.

The authorization module 140 receives the user approval (or authentication) signal from the browser 113 via the network 120 and authenticates the user, at 220. In some instances, the authorization module 140 can authenticate the user by matching the user authentication information to a specific entry in a user authentication database or a look-up table stored in the memory 132 of the authorization server 130. The user authentication database can include, for example, a list of employees and their corresponding employment positions and security/access level, social security numbers, employee personal identification numbers (PIN), employee passwords, and/or the like. In other configurations, the user authentication database can be stored in the memory of another device (e.g., a server not shown in FIGS. 1 and 2) operatively coupled to the network 120. The authorization module 140 can authenticate the user using any
suitable authentication protocol such as, for example, Secure Sockets Layer (SSL), Secure Shell (SSH), Kerberos, and/or the like.

[1034] Upon successful authentication of the user, the authorization module 140 can generate an authorization code associated with the access token. The authorization code can be, for example, a string of arbitrary random or pseudo-random numbers generated by the authorization module 140 that is mapped to the access token for a given application. The authorization module 140 can send to the browser 113 a signal that redirects the browser 113 to a device or resource within network 120 and that includes the authorization code via a uniform resource locator (URL) represented by the signal, at 222. The URL provided is formatted in a scheme that the application 114 is configured to handle (the scheme of the URL provided is not recognized by the browser 113). When an application 114 is installed on a mobile device, the application 114 registers itself with the operating system of the mobile device as a handler of a uniform resource locator (URL) of a given scheme such as, for example, URLs of the form 'nativeapp_a://resi_of_url'. The scheme of the URL represented by the signal sent by the authorization module 140 is only recognizable by the application 114 (and not the browser) because the valid application 114 (and not any other application) is designed to be able to access the authorization code that can be exchanged for the access token. If the scheme of the URL represented by the signal sent by the authorization module 140 was otherwise recognizable by the browser, a possibility exists that the browser (and not the valid application) might obtain the authorization code - a situation that can be avoided by the application 114 registering itself with the operating system. The authorization code can be random or pseudo-random string of numbers that is generated at the authorization module 140 that is uniquely associated with the access token for an application 114 installed on the device 110 for a user. In some configurations, after the authorization module 140 generates the authorization code for the first time for an application, the authorization module 140 can store the authorization code along with the access token and the verification ID for an application 114 in a database in the memory 132 of the authorization server 130 for access in the future. In other configurations, the new authorization code can be generated by the authorization module 140 each time an access token is requested by an application 114 and the authorization code can be made time variant by including a suitably fine-grained timestamp in its value.
[103S] Because the URL scheme used by the authorization module 140 to format the signal is not recognizable by the browser 113, the browser 113 sends the URL to the operating system. The signal or portion of the signal representing the same URL scheme is registered at the authorization module 140 and the operating system of the mobile device 110 as being associated with the particular application 114. For example, a database or lookup table can exist in the memory of the authorization server 132 and the memory of the device 112, and this database or lookup table can contain entries associated with different applications and the URL’s each application can handle. On receiving the URL from the browser 113, the operating system of device 110 can search the database or lookup table to detect or determine whether the specific URL scheme associated with the URL is supported by the device 110 and if so, determine what applications 114 have used the URL scheme to register itself on the device 110. On detecting the application 114 that can handle that URL scheme, the operating system sends the signal or portion of the signal representing the URL with the authorization code to the application 114, at 224.

[1036] The application 114 can access the authorization code contained in the URL and generate a second verification ID. In some instances, the application 114 can retrieve the original verification ID from the memory of the device instead of generating a second verification ID. The application 114 can send a signal representing the authorization code, the verification ID and the client ID to the authorization module 140 without going through the browser 113 via the network 120, at 226.

[1037] At 228, the authorization module 140 can receive via the network 120, the signal representing the authorization code, the verification ID and the client ID from the application 114, and can assess the validity of and/or verify the attributes of the authorization code, the verification ID and the client ID. In some instances, the authorization module 140 can perform this verification by querying a database or a look-up table stored in the memory of the authorization server 130 for an entry that corresponds to the authorization code and/or the verification ID and/or the client ID. In other instances, the database or look-up table can be stored in the memory of a different server operatively coupled to the network 120. For example, the authorization module 140 can acknowledge if an authorization code for an application 114 of an authorized user of the communication device 110 has been received. In another example, the authorization module 140 can use the authorization code, the verification ID and the client ID to
query a second database or lookup table stored the memory of the authorization server that can contain entries that list the approved functionalities of each application for an authenticated user. The authorization module 140 can acknowledge if the approved functionalities of the application 114 has been activated for the user. In yet another example, the authorization module 140 can acknowledge if the time stamp on the verification ID and/or the client ID is valid, and/or the like.

[1038] Upon verifying the authorization code, the verification ID and the client ID, the authorization module 140 can generate and/or define an access token for the requested application 114 installed on the device 110. Note that the access token can be generated at this point when the verification ID sent by the application 114 (after obtaining the authorization code - step 226) is equal to the original verification ID (stored in the memory of the authorization server) and will not be generated when they differ. The two verification ID’s differing is an indication that the application that generated the signal requesting the access token is not the same application that initiated the process of user authentication and access token generation. Accordingly, the verification ID can be used to verify that the application that generated the signal requesting the access token is not malicious code posing as the application that initiated the process of user authentication and access token generation.

[1039] In some instances, the access token can include an encrypted or unencrypted form of a user personal identity number (PIN), a user security/access level, a device identification number, an internet protocol (IP) and/or media access control (MAC) address of the device, an application identity number, an address of the authorization module 140, and/or the like. In other instances, the access token can include additional encrypted or unencrypted application credentials such as the specific features and/or resources of the application 114 that has been approved for the user, a duration for which the access token is valid, and/or the like. In some instances, the access token can be, for example, an Open Authorization (OAuth) token, an OpenID token, a SecurID token, and/or the like. Upon generating the access token, the authorization module 140 can send the access token to the application 114 via the network, at 230.

[1040] The application 114 can send the access token to the resource module 156 directly via the network (i.e. not through the browser), at 232. Upon successfully receiving the access token,
the resource module 156 can, at 234, send application data via the network 120 to the application
114 in a data package (e.g., a JavaScript Object Notation (JSON) package) for use during
execution of the application 114 on the device 110. For example, if the application 114 is a sales
application, the application data can include a representation of the price of a unit of a product,
the number of units sold in a month, the number of units in pending orders, the revenue
generated from all sales in a month, the profits earned in a month, and/or the like. In some
instances, the application 114 can use the application data to populate a table for display to the
user. In another example, if the application 114 is a human resource (HR) application, the
application data can include a user employee number, a user social security number, an employee
status code, an enterprise code for paid time off (PTO), and/or the like. In some instances, the
application 114 can allow the user to use this data to calculate the total number of hours worked
that can be charged to the enterprise in a pay cycle, and/or the like.

[1041] The access token issuing system as described in FIG. 2 involves the generation of a
first and a second verification ID by the (valid) application 114 (or the generation and the
retrieval of a valid verification ID by the application 114) during the request for the
authorization code (step 214) and request for the access token (step 226). An access token is
generated by the authorization module 140 when the two verification ID’s are equal but not
when they differ (or when the verification ID retrieved by the application 114 is equal to the
verification ID stored by the authorization module 140). Hence, the generation of the unique
verification ID (e.g., a nonce) by the (valid) application 114 can reduce the risk of a malicious
software to claim the access token as the malicious software does not generate the verification ID
(that is unique only to the valid application 114). Known operating systems on mobile devices
110 cannot prevent a malicious software (e.g., spyware, virus, etc.) from claiming the URL
scheme associated with a valid application 114. Hence in known operating systems, even though
a valid application 114 may start the process of obtaining the access token as shown in FIG. 2, if
a malicious application has been able to claim the URL scheme, the operating system on the
(mobile) device 110 can pass the URL with the authorization code to the malicious application
instead of the valid application 114. The malicious software can then exchange the authorization
code for the access token and use it on making API calls. Although by implementing the access
token issuing system described in FIGS. 1 and 2 a malicious software may still be able to
intercept the authorization code, the malicious application, however, will not be able to exchange
the authorization code with the access token as the malicious software cannot provide a valid verification ID. Only a valid application 114 can provide the valid verification ID and hence only a valid application 114 can obtain the access token.

FIG. 3 is a flow chart illustrating a method for issuing access tokens for an application by an authorization module, according to an embodiment. The method 300 includes receiving at, for example, an authorization module at a first time and from a first mobile application, a request for an access token (1) associated with a second mobile application and (2) including an identifier associated with the second mobile application and a verification identifier generated by the second mobile application, at 302. The first mobile application can be, for example, any native mobile browser installed on a mobile device that can connect to the Internet via, for example, a cellular network and/or a wireless local area network (IAN), using standard Hypertext Transfer Protocol (HTTP) over Transmission Control Protocol/Internet Protocol (TCP/IP) and can display web pages written in, for example, HyperText Markup Language (HTML), Extensible HyperText Markup Language (XHTML), Mobile Profile WAP 2.0, Wireless Markup Language (WML), and/or the like. Examples of browsers include the Android browser, the Blackberry browser, Safari, Internet Explorer Mobile browser, Firefox for mobile browser, Polaris browser, and/or the like. The second mobile application can be, for example, any native mobile application installed on a mobile device or a web-browser-based application that is configured to receive access tokens from the authorization module and cause the processor to execute specific operations associated with particular functions in an enterprise.

At 304, a signal representing an authorization code associated with the access token is provided to the first mobile application and in response to the request. The signal representing the authorization code is sent from, for example, the authorization module. As described above, the authorization module can send to the first mobile application (browser) a signal that redirects the browser to a network device or resource associated with a uniform resource locator (URL) that includes the authorization code and a scheme the application is configured to handle (not recognized by the browser). The authorization code can be a random or pseudo-random siring of numbers that is generated at the authorization module that is associated with an application installed on the device for a user.
At 306, a signal representing the authorization code, the identifier associated with the second mobile application, and the verification identifier is received at, for example, the authorization module, at a second time after the first time. As described above, the verification identifier (or verification ID) can be, for example, a string of arbitrary random or pseudo-random numbers (e.g., a cryptographic nonce) issued so that this verification ID is not reused in any replay attacks. To ensure that the verification ID is used only once, the verification ID can be made to be time-variant, for example, by including a suitably fine-grained timestamp in the value or generating the verification ID with a sufficient number of random bits to ensure a probabilistically insignificant chance of repeating a previously-generated value.

At 308, a signal representing the access token is provided by, for example, the authorization module to the second mobile application when the authorization code, the identifier associated with the second mobile application, and the verification identifier are verified. In some instances, the authorization module can perform this verification by querying a database or a look-up table stored in the memory of the authorization server for an entry that corresponds to the authorization code, verification ID and/or client ID. In other instances, the database or look-up table can be stored in the memory of a different server operatively coupled to the network. As described above, the access token is generated and sent from the authorization module when the first verification identifier (generated in step 302) is equal to the second verification identifier (generated in step 306) and not when they differ. The generation of the unique verification ID (e.g., a nonce) by the (valid) application can reduce the risk of a malicious software to claim the access token because the malicious software does not generate the verification ID (that is unique only to the valid application).

Some embodiments described herein relate to a computer storage product with a non-transitory computer-readable medium (also can be referred to as a non-transitory processor-readable medium) having instructions or computer code thereon for performing various computer-implemented operations. The computer-readable medium (or processor-readable medium) is non-transitory in the sense that it does not include transitory propagating signals per se (e.g., a propagating electromagnetic wave carrying information on a transmission medium such as space or a cable). The media and computer code (also can be referred to as code) may be those designed and constructed for the specific purpose or purposes. Examples of non-transitory computer-readable media include, but are not limited to: magnetic storage media such as hard
disks, floppy disks, and magnetic tape: optical storage media such as Compact Disc/Digital Video Discs (CD/DVDs), Compact Disc-Read Only Memories (CD-ROMs), and holographic devices; magneto-optical storage media such as optical disks; carrier wave signal processing modules; and hardware devices that are specially configured to store and execute program code, such as Application-Specific Integrated Circuits (ASICs), Programmable Logic Devices (PLDs), Read-Only Memory (ROM) and Random-Access Memory (RAM) devices. Other embodiments described herein relate to a computer program product, which can include, for example, the instructions and/or computer code discussed herein.

[1047] Examples of computer code include, but are not limited to, micro-code or micro-instructions, machine instructions, such as produced by a compiler, code used to produce a web service, and files containing higher-level instructions that are executed by a computer using an interpreter. For example, embodiments may be implemented using imperative programming languages (e.g., C, Fortran, etc.), functional programming languages (Haskell, Erlang, etc.), logical programming languages (e.g., Prolog), object-oriented programming languages (e.g., Java, C++, etc.) or other suitable programming languages and/or development tools. Additional examples of computer code include, but are not limited to, control signals, encrypted code, and compressed code.

[1048] While various embodiments have been described above, it should be understood that they have been presented by way of example only, and not limitation. Where methods described above indicate certain events occurring in certain order, the ordering of certain events may be modified. Additionally, certain of the events may be performed concurrently in a parallel process when possible, as well as performed sequentially as described above. For example, in some instances, the browser can include the user approval information along with the verification ID and the client ID in the signal sent to the authorization module that requests an access token for the application. In such instances, step 218 can be performed concurrently with step 214 in FIG. 2 and step 216 can be avoided.
What is claimed is:

1. An apparatus, comprising:

   an authorization module implemented in at least one of a memory or a processing device, the authorization module configured to receive, at a first time and from a first mobile application, a request for an access token associated with a second mobile application, the request including an identifier associated with the second mobile application and a first random verification identifier, the authorization module configured to provide, to the first mobile application and in response to the request, a signal representing an authorization code associated with the access token,

   the authorization module configured to receive, from the second mobile application and at a second time after the first time, a signal representing the authorization code, the identifier associated with the second mobile application, and a second random verification identifier, the authorization module configured to provide a signal representing the access token to the second mobile application based at least in part on the first random verification identifier being equal to the second random verification identifier.

2. The apparatus of claim 1, wherein the request includes a uniform resource locator (URL) associated with an address of the authorization module, the URL including the identifier associated with the second mobile application and the first random verification identifier, the authorization module configured to store an association between the first random verification identifier and the identifier associated with the second mobile application.

3. The apparatus of claim 1, wherein the first random verification identifier is generated by the second mobile application.

4. The apparatus of claim 1, wherein the first mobile application is a web browser application.

5. The apparatus of claim 1, wherein the authorization module is configured to not provide the first random verification identifier to the first mobile application or the second mobile application.
6. The apparatus of claim 1, wherein the access token is an Open Authorization (OAuth) access token.

7. The apparatus of claim 1, wherein the authorization module is configured to receive, from a user of the first mobile application, a signal representing approval to provide the access token to the second mobile application prior to providing the authorization code associated with the access token to the first mobile application.

8. The apparatus of claim 1, wherein the authorization module is configured to provide the authorization code associated with the access token to the first mobile application by providing, to the first mobile application, a signal representing a uniform resource locator (URL) that (1) includes the authorization code, and (2) the second mobile application is configured to handle.

9. The apparatus of claim 1, wherein the second mobile application is a native mobile application on a mobile device.

10. A non-transitory processor-readable medium storing code representing instructions to be executed by a processor, the code comprising code to cause the processor to:
    receive, at a first time and from a first mobile application, a request for an access token associated with a second mobile application, the request including an identifier associated with the second mobile application and a verification identifier generated by the second mobile application;
    provide, to the first mobile application and in response to the request, a signal representing an authorization code associated with the access token;
    receive, from the second mobile application and at a second time after the first time, a signal representing the authorization code, the identifier associated with the second mobile application, and the verification identifier; and
    provide a signal representing the access token to the second mobile application when the authorization code, the identifier associated with the second mobile application, and the verification identifier are verified.
11. The non-transitory processor-readable medium of claim 10, wherein the first mobile application is a web browser application, the code to cause the processor to provide includes code to cause the processor to provide, to the web browser application, a uniform resource locator (URL) (1) unrecognizable by the web browser and (2) including the authorization code.

12. The non-transitory processor-readable medium of claim 10, further comprising code to cause the processor to:
   - receive, from a user of the first mobile application, approval to provide the access token to the second mobile application prior to providing the authorization code associated with the access token to the first mobile application.

13. The non-transitory processor-readable medium of claim 10, wherein the access token is an Open Authorization (OAuth) access token.

14. The non-transitory processor-readable medium of claim 10, wherein the request includes a uniform resource locator (URL) associated with an address of an authorization module.

15. The non-transitory processor-readable medium of claim 10, further comprising code to cause the processor to:
   - store an association between the verification identifier and the identifier associated with the second mobile application.

16. An apparatus, comprising:
   - a first mobile application implemented in at least one of a memory or a processing device, the first mobile application configured to generate a random verification identifier, the first mobile application configured to initiate a second mobile application such that the second mobile application sends a signal representing the random verification identifier and an identifier associated with the first mobile application to an authorization module at a first time,
the first mobile application configured to receive a signal representing an authorization code provided by the authorization module based on the authorization code being within an address associated with the first mobile application at a second time after the first time,

the first mobile application configured to provide a signal representing the random verification identifier and the authorization code to the authorization module, the first mobile application configured to receive a signal representing an access token associated with the first mobile application based at least in part on the authorization module validating the random verification identifier and the authorization code.

17. The apparatus of claim 16, wherein the second mobile application is a web browser application.

18. The apparatus of claim 16, wherein the address associated with the first mobile application is a uniform resource locator (URL) (1) that includes the authorization code, and (2) from which the second mobile application is configured to extract the authorization code.

19. The apparatus of claim 16, wherein the access token is an Open Authorization (OAuth) access token.

20. The apparatus of claim 16, wherein the first mobile application is configured to provide a signal representing an address associated with the authorization module to the second mobile application when initiating the second mobile application.

21. The apparatus of claim 16, wherein the first mobile application is configured to provide the access token to an application module associated with the first mobile application in response to receiving the signal representing the access token.
FIG. 2

1. Generate verification ID

2. Initiate browser, and send verification ID and client ID

3. Request token including verification ID and client ID

4. Request user approval

5. Provide user approval

6. Authenticate user

7. Redirect browser to local URL including code

8. Provide code, verification ID and client ID

9. Verify code, verification ID and client ID

10. Access Token

11. Access Token

12. Application data (AD)
Receive, at a first time and from a first mobile application, a request for an access token (1) associated with a second mobile application and (2) including an identifier associated with the second mobile application and a verification identifier generated by the second mobile application.

Provide, to the first mobile application and in response to the request, a signal representing an authorization code associated with the access token.

Receive, from the second mobile application and at a second time after the first time, a signal representing the authorization code, the identifier associated with the second mobile application, and the verification identifier.

Provide a signal representing the access token to the second mobile application when the authorization code, the identifier associated with the second mobile application, and the verification identifier are verified.

FIG. 3
INTERNATIONAL SEARCH REPORT

A. CLASSIFICATION OF SUBJECT MATTER
IPC: G06F 21/30 (2013.01)

USPC: 7264

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
U.S. : 7264;9,26-29;713/172

C. DOCUMENTS CONSIDERED TO BE RELEVANT

<table>
<thead>
<tr>
<th>Category</th>
<th>Citation of document, with indication, where appropriate, of the relevant passages</th>
<th>Relevant to claim No.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Y</td>
<td>US 2012/0291 114 A1 (POLISHANKO ET AL) 15 November 2012 (15.1 1 2012)</td>
<td>1-21</td>
</tr>
<tr>
<td>Y,P</td>
<td>US 2013/0086679 A1 (BEITER) 04 April 2013 (04.04.2013)</td>
<td>1-21</td>
</tr>
<tr>
<td>Y</td>
<td>US 2011/0321131 A1 (AUSTEL ET AL) 29 December 2011 (29.12.2011)</td>
<td>2,4,6,8,11,13,14 and</td>
</tr>
<tr>
<td>Y</td>
<td>US 2012/0110646 A1 (AITOMI ET AL) 03 May 2012 (03.05.2012)</td>
<td>7 and 12</td>
</tr>
</tbody>
</table>

Further documents are listed in the continuation of Box C.

See patent family annex.

Date of the actual completion of the international search: 23 January 2014 (23.01.2014)

Date of mailing of this international search report: 2 January 2014 (02.01.2014)

Authorized: Kim Huynh

Telephone: No. 571-2724147

Form PCT/ISA/210 (second sheet) (April 2007)
Continuation of B. FIELDS SEARCHED Item 3:

Search Text: US=PGPUB;USPAT;USOCR;FPRS;EPO;JPPO;DERWENT;IBM_TDB

Search Terms: identity, token, application, ping, nonce, device identification, random, different, verification, sequence, browser