(12) UK Patent (19) GB (11) 2599273 (13) B

(45) Date of B Publication 01.11.2023

(54) Title of the Invention: Fine-grained token based access control

(51) INT CL: ***H04L 9/40*** (2022.01)

(21) Application No: **2117302.6**

(22) Date of Filing: **06.04.2020**

Date Lodged: **30.11.2021**

(30) Priority Data:
(31) **19172926** (32) **07.05.2019** (33) **EP**

(86) International Application Data:
PCT/IB2020/053254 En 06.04.2020

(87) International Publication Data:
WO2020/225616 En 12.11.2020

(43) Date of Reproduction by UK Office **30.03.2022**

(72) Inventor(s):
Martin Smolny
Thomas Duerr
Michael Beck
Juergen Schaeck

(73) Proprietor(s):
International Business Machines Corporation
New Orchard Road, Armonk, New York 10504,
United States of America

(74) Agent and/or Address for Service:
IBM United Kingdom Limited
Intellectual Property Law, Hursley Park, Hursley,
WINCHESTER, Hampshire, SO21 2JN,
United Kingdom

(56) Documents Cited:
WO 2019/036012 A1
CN 109309683 A
US 20190068575 A1
US 20170034172 A1
US 20160028737 A1

(58) Field of Search:
As for published application 2599273 A viz:
INT CL **G06F, H04L, H04W**
Other: **CNABS,CNTXT,CNKI.DWPI,SIPOABS**
updated as appropriate

Additional Fields
Other: **None**

GB 2599273 B

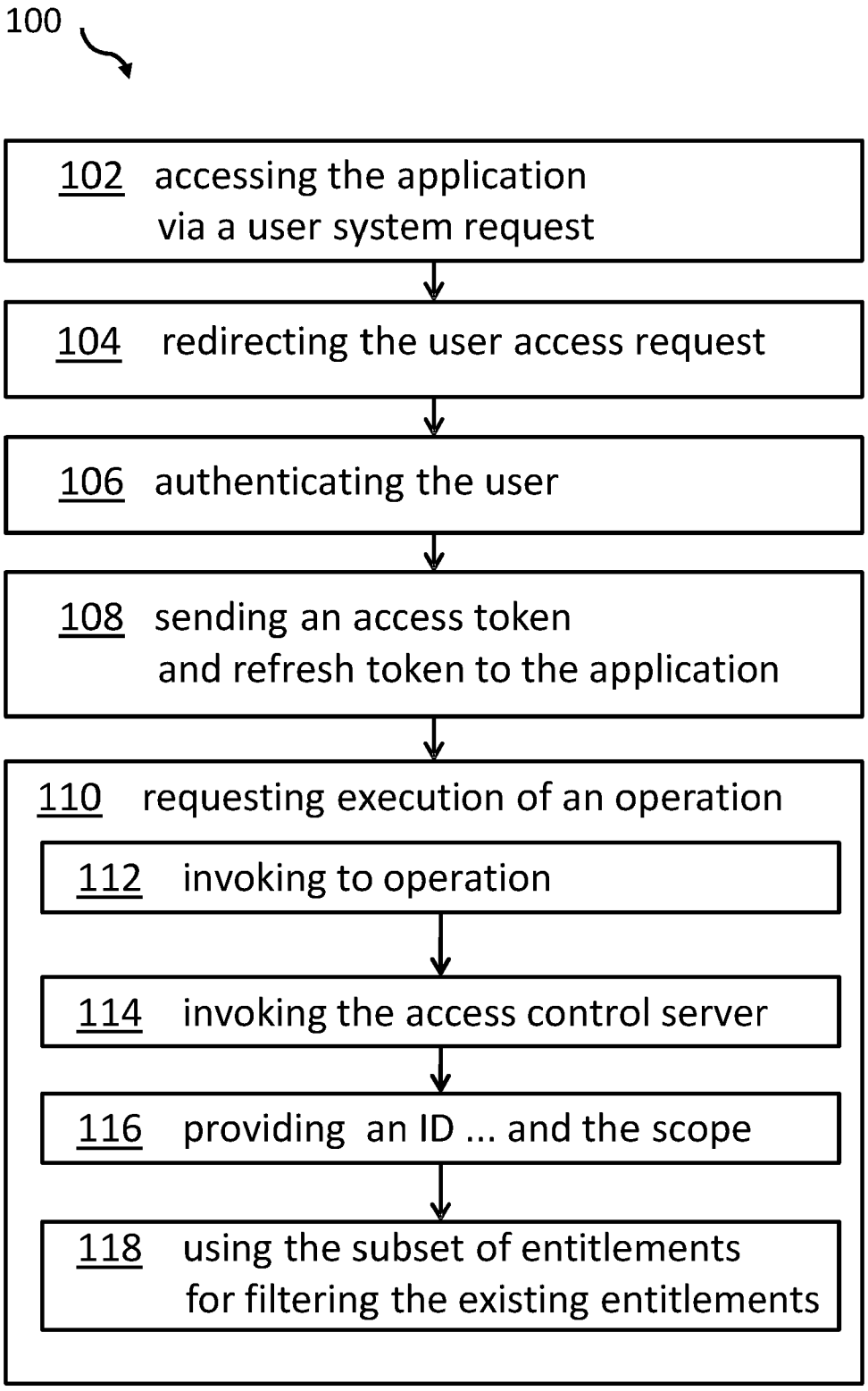

100
102 accessing the application via a user system request
104 redirecting the user access request
106 authenticating the user
108 sending an access token and refresh token to the application
110 requesting execution of an operation
112 invoking to operation
114 invoking the access control server
116 providing an ID ... and the scope
118 using the subset of entitlements for filtering the existing entitlements

FIG. 1

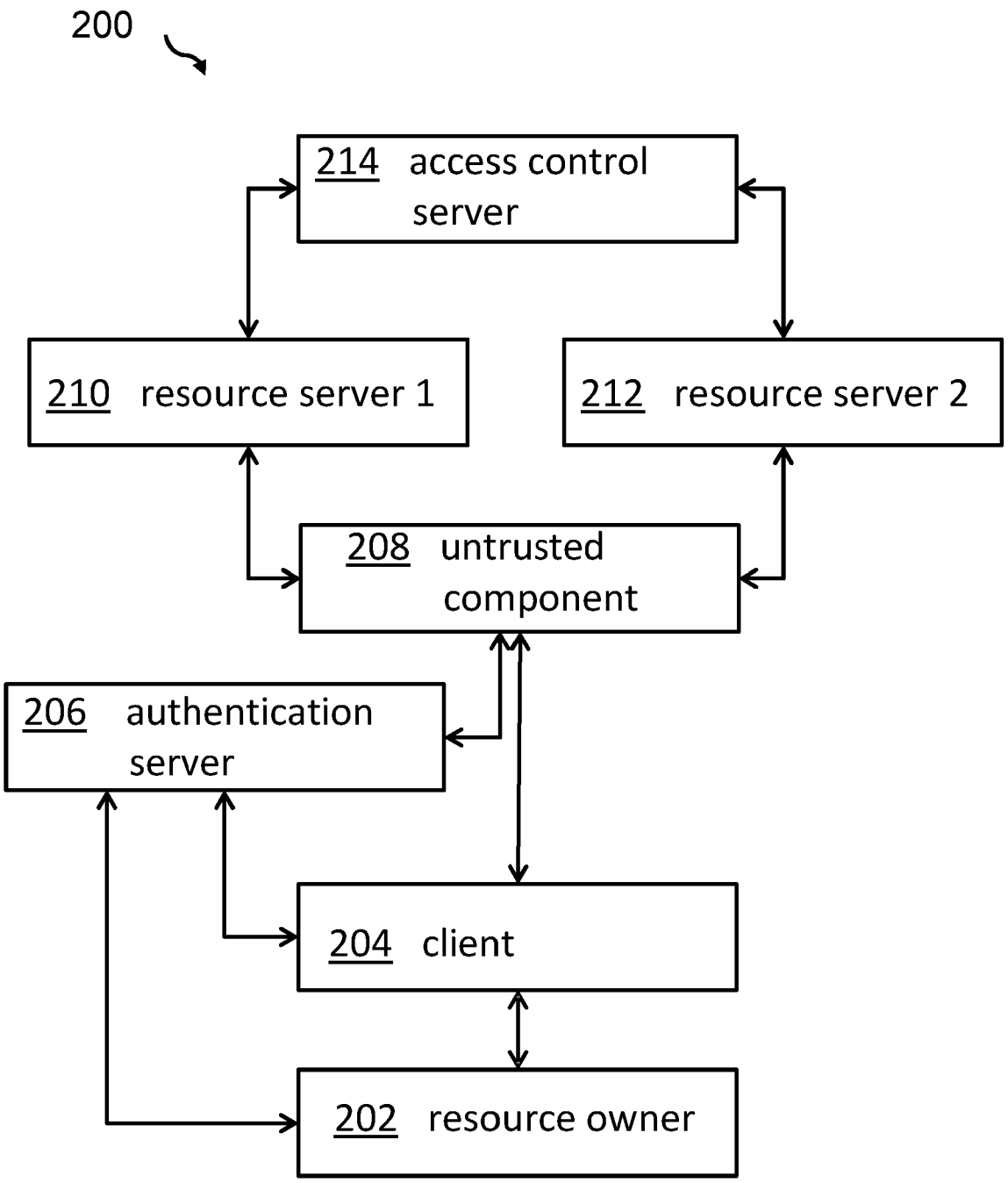

200
214 access control server
210 resource server 1
212 resource server 2
208 untrusted component
206 authentication server
204 client
202 resource owner

FIG. 2

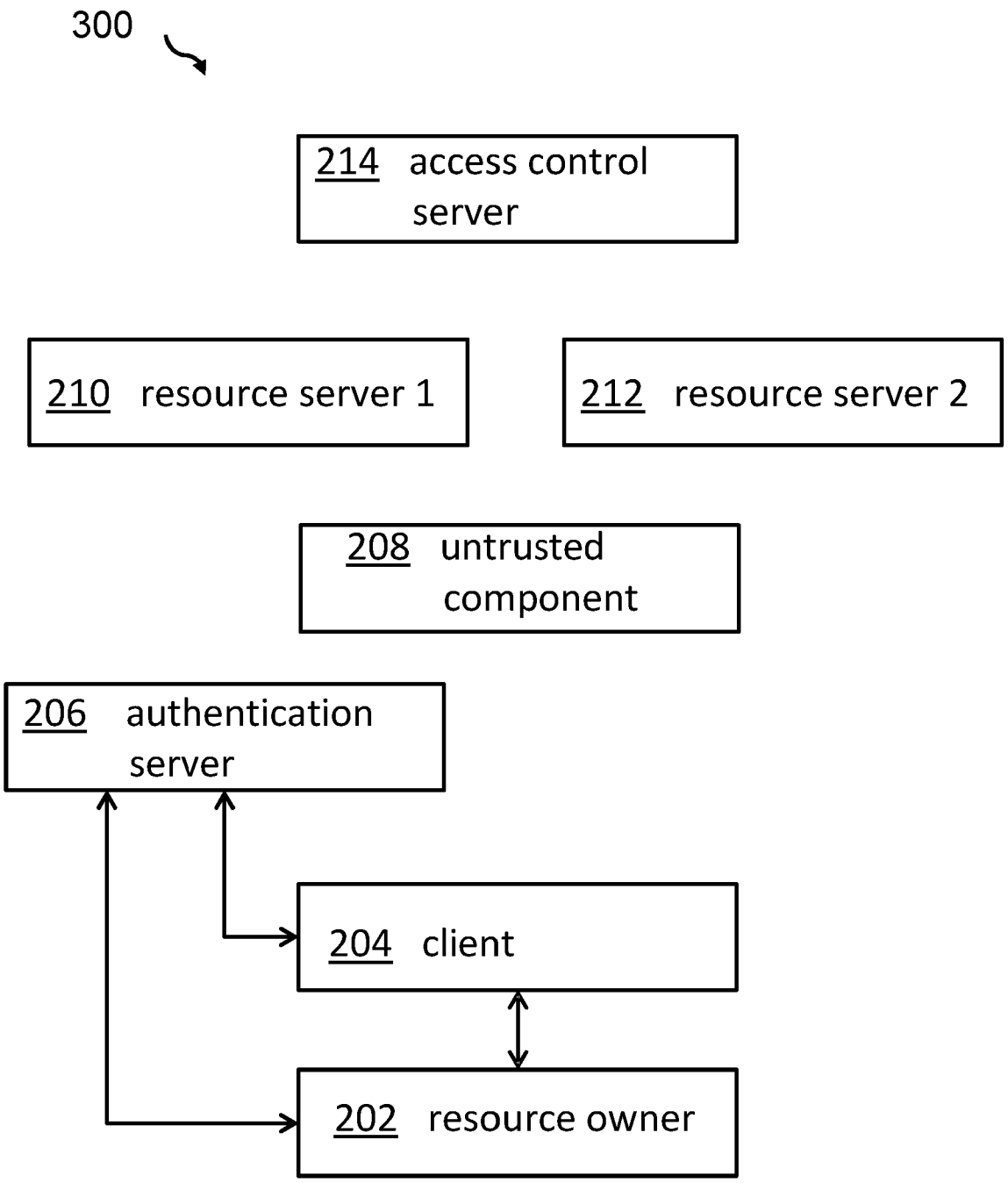

300
214 access control server
210 resource server 1
212 resource server 2
208 untrusted component
206 authentication server
204 client
202 resource owner

FIG. 3

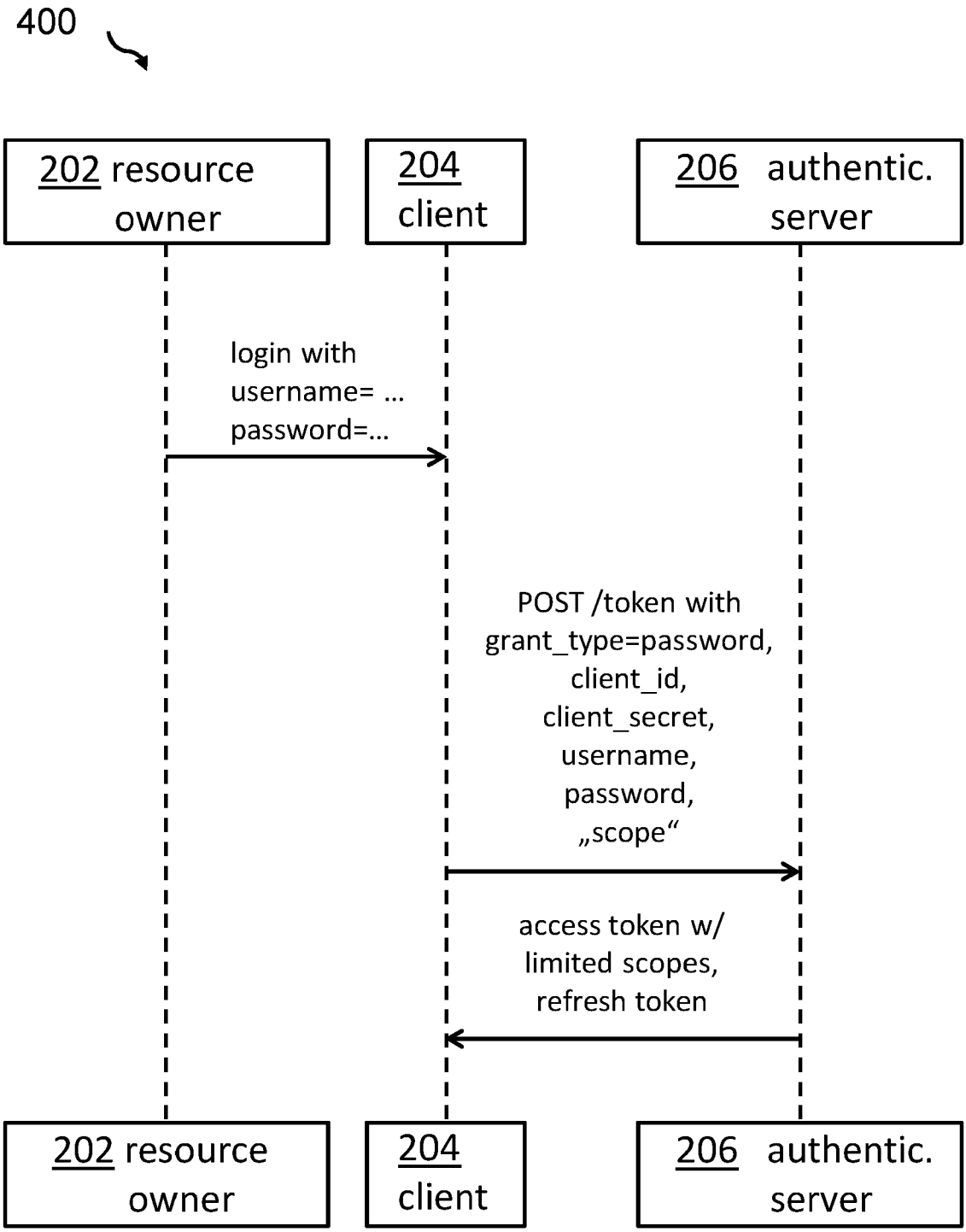

400
202 resource owner
204 client
206 authentic. server
login with username= ... password=...
POST /token with grant_type=password, client_id, client_secret, username, password, „scope“
access token w/ limited scopes, refresh token
202 resource owner
204 client
206 authentic. server

FIG. 4

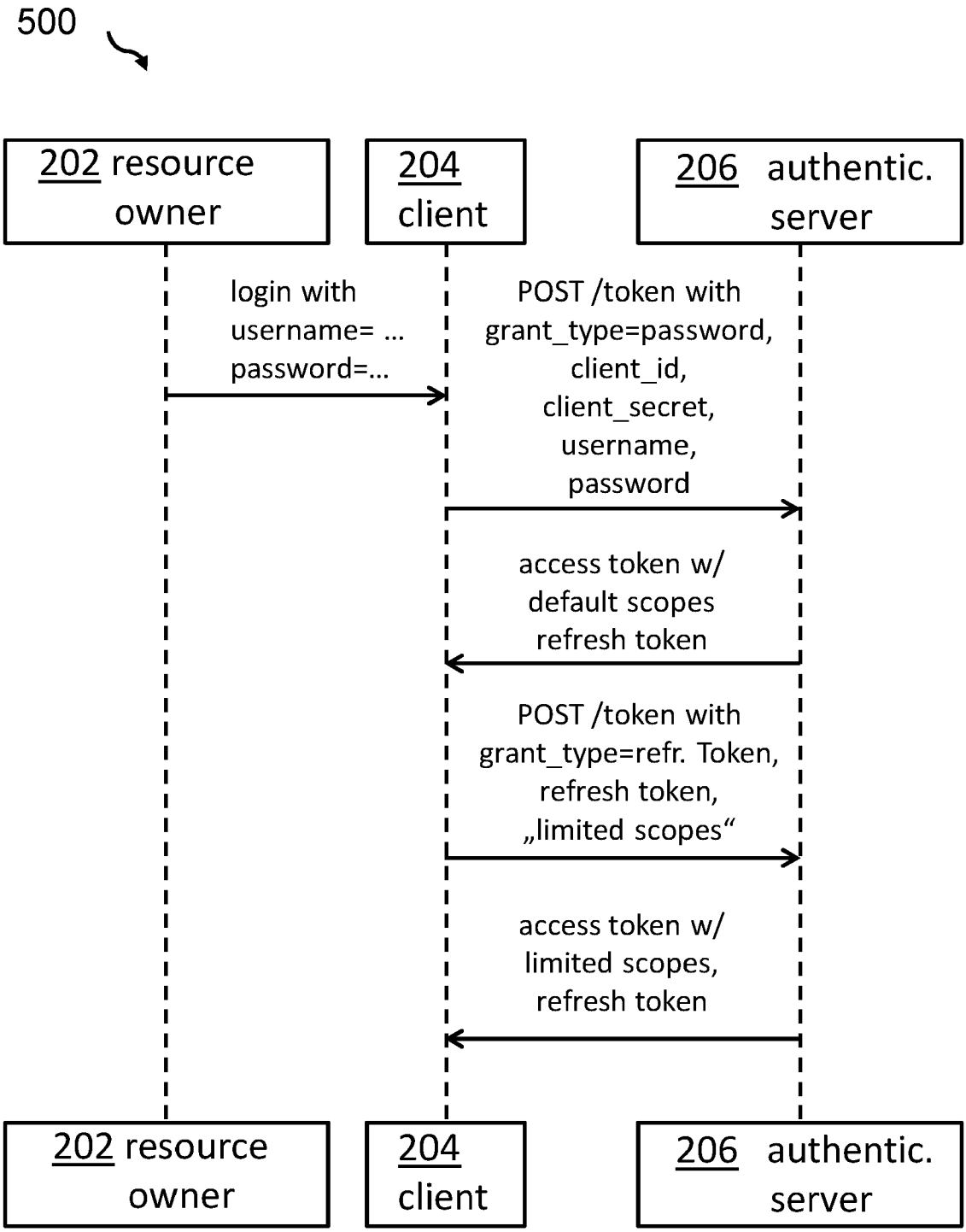

500
202 resource owner
204 client
206 authentic. server
login with username= … password=…
POST /token with grant_type=password, client_id, client_secret, username, password
access token w/ default scopes refresh token
POST /token with grant_type=refr. Token, refresh token, „limited scopes“
access token w/ limited scopes, refresh token
202 resource owner
204 client
206 authentic. server

FIG. 5

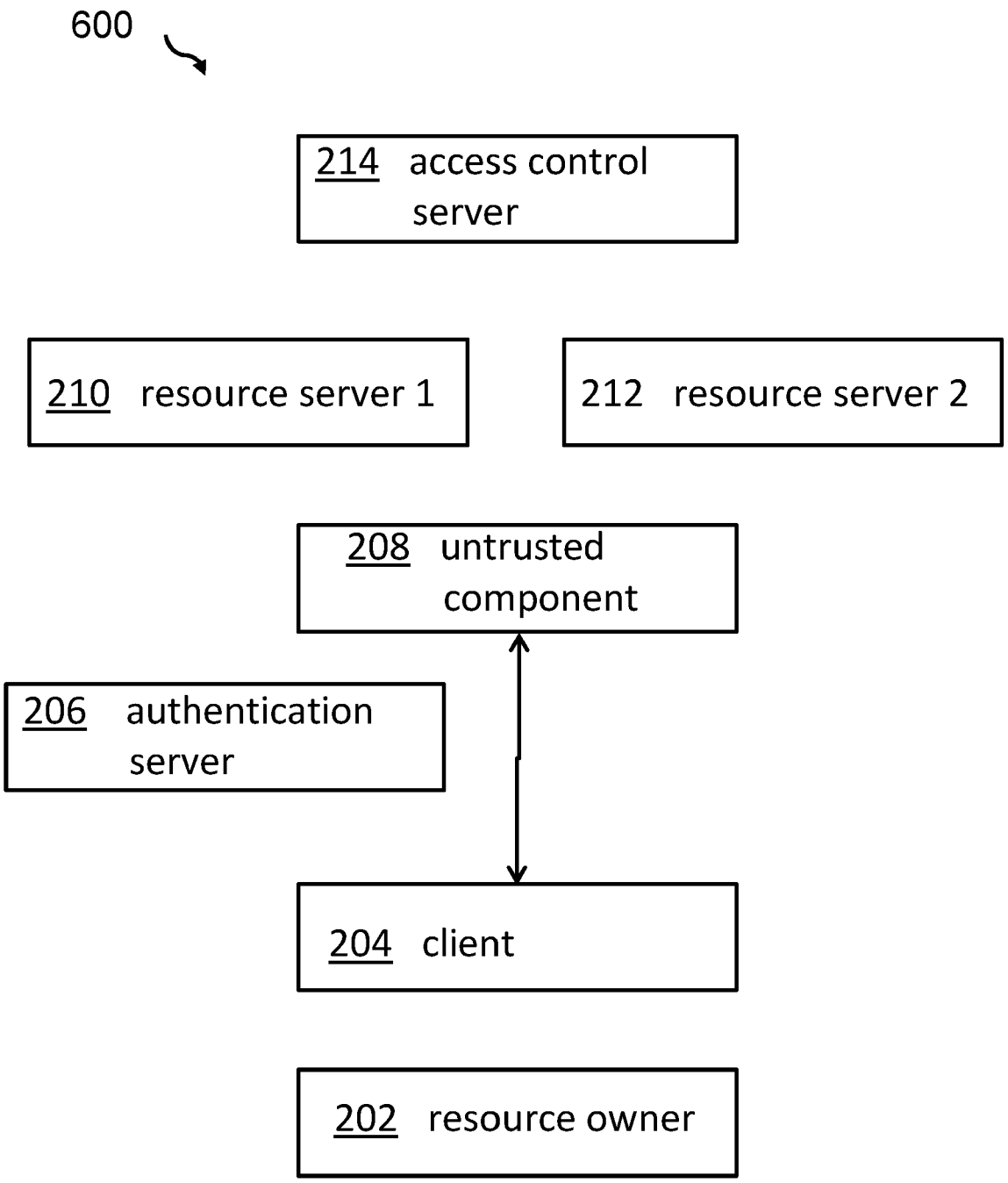

600
214 access control server
210 resource server 1
212 resource server 2
208 untrusted component
206 authentication server
204 client
202 resource owner

FIG. 6

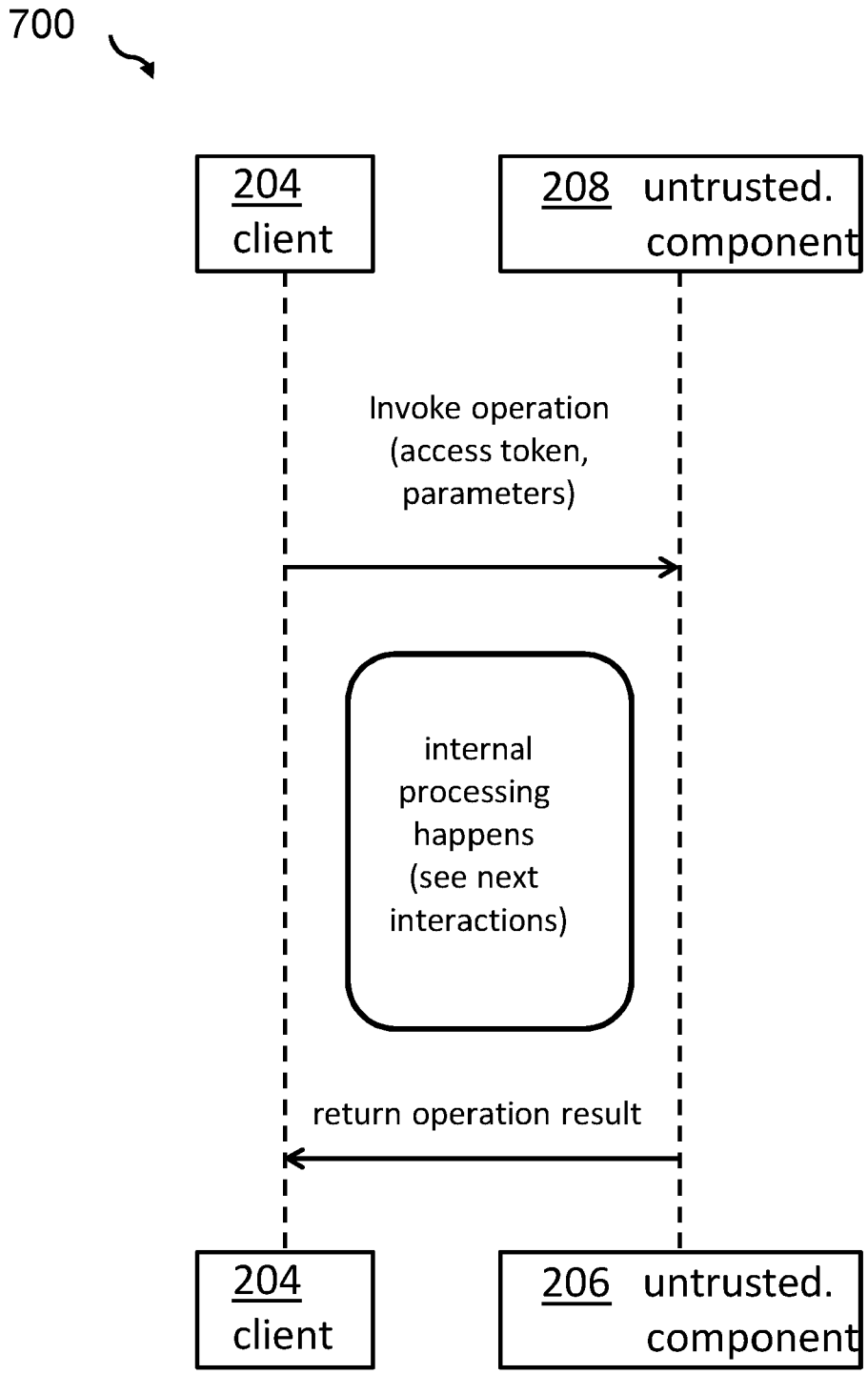

700
204 client
208 untrusted. component
Invoke operation (access token, parameters)
internal processing happens (see next interactions)
return operation result
204 client
206 untrusted. component

FIG. 7

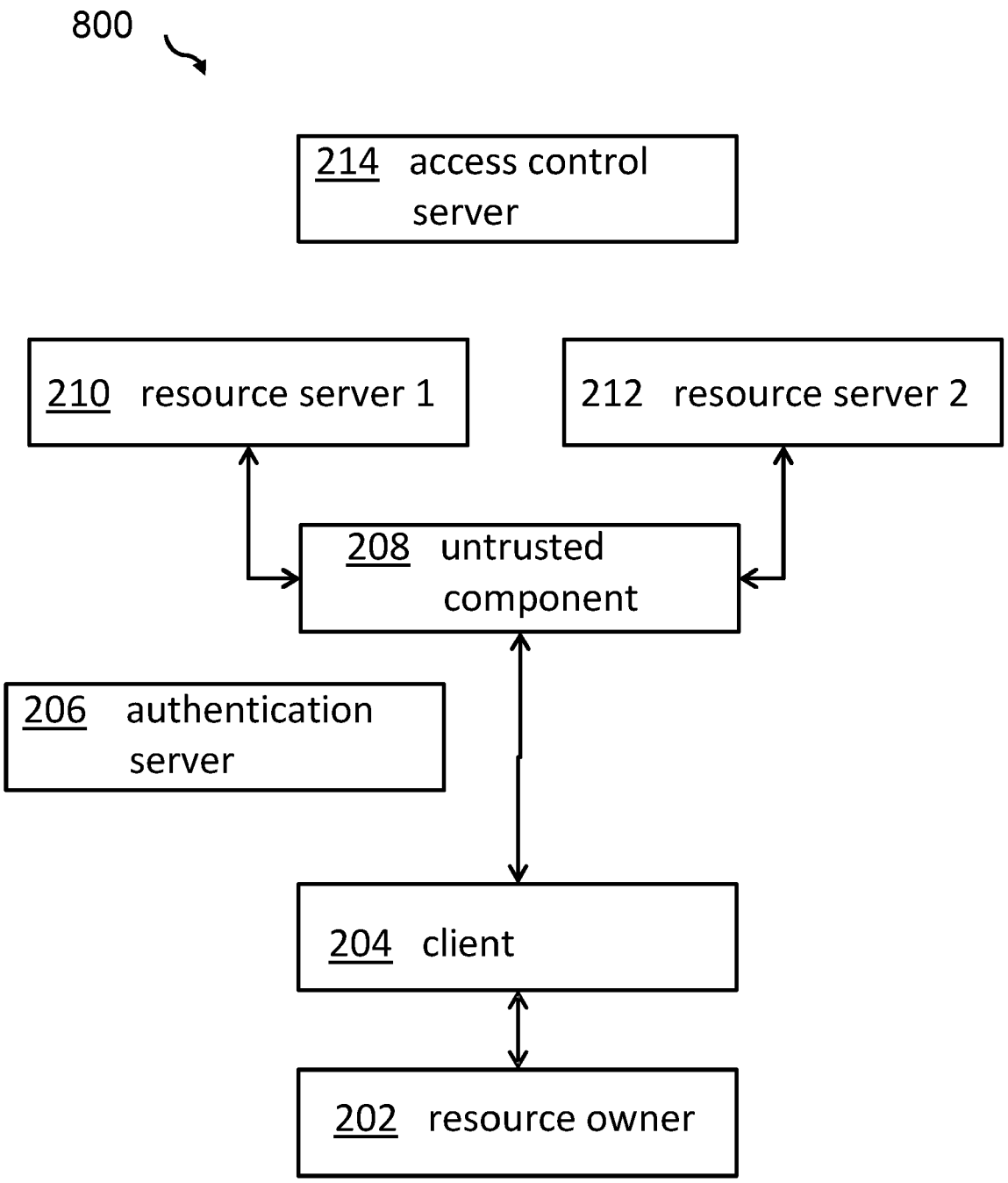

800
214 access control server
210 resource server 1
212 resource server 2
208 untrusted component
206 authentication server
204 client
202 resource owner

FIG. 8

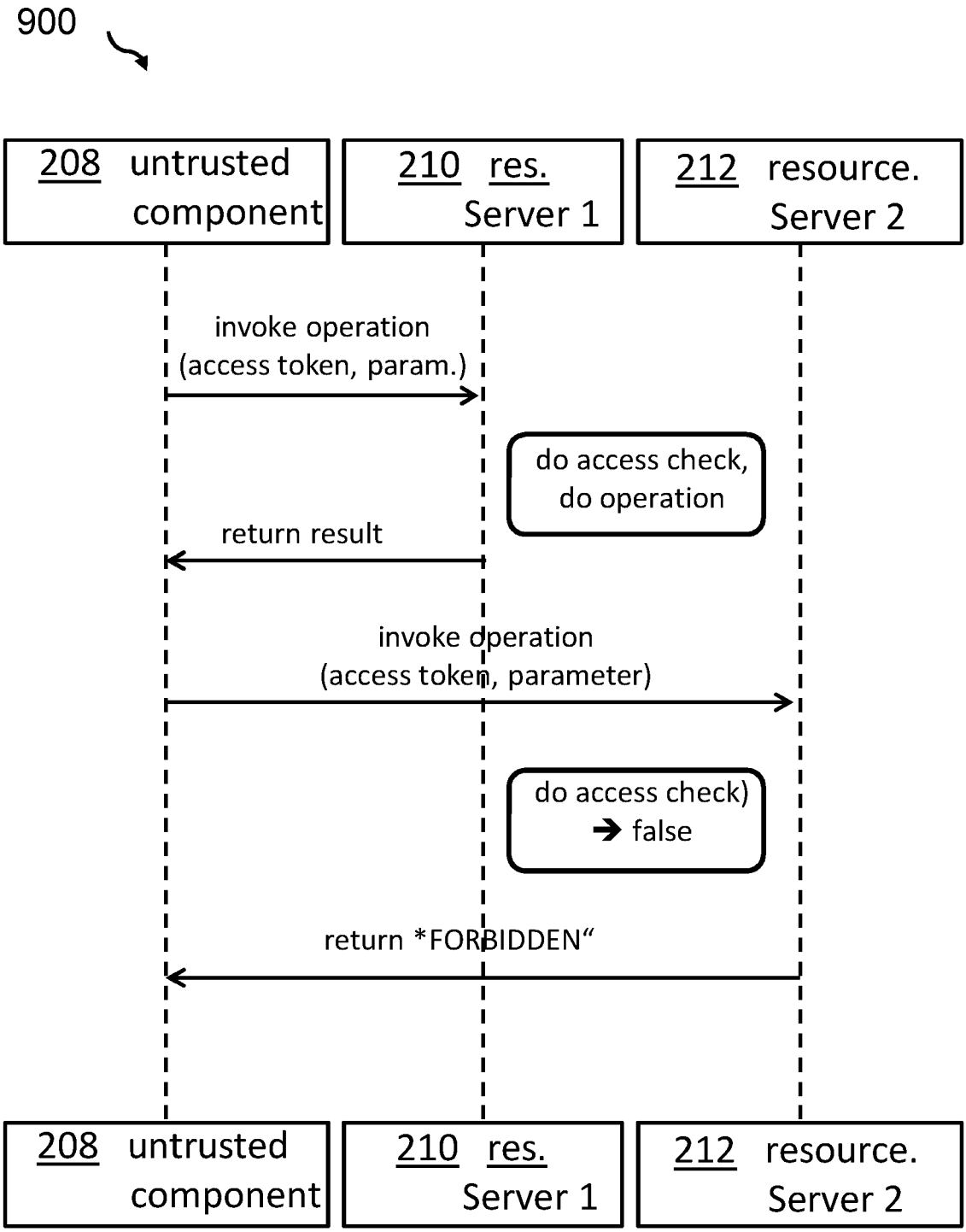

900
208 untrusted component
210 res. Server 1
212 resource. Server 2
invoke operation (access token, param.)
do access check, do operation
return result
invoke operation (access token, parameter)
do access check) ➔ false
return *FORBIDDEN“
208 untrusted component
210 res. Server 1
212 resource. Server 2

FIG. 9

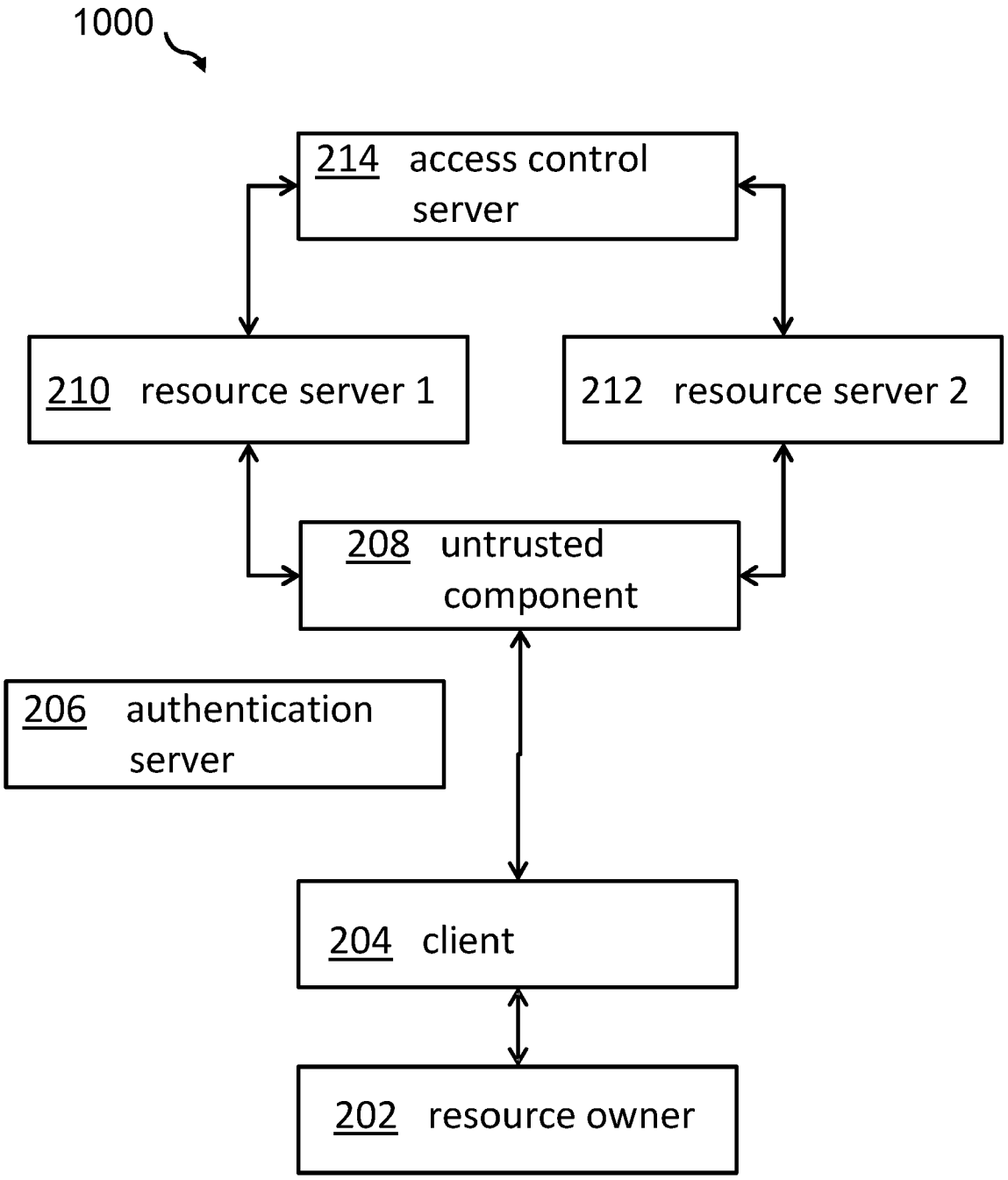

1000
214 access control server
210 resource server 1
212 resource server 2
208 untrusted component
206 authentication server
204 client
202 resource owner

FIG. 10

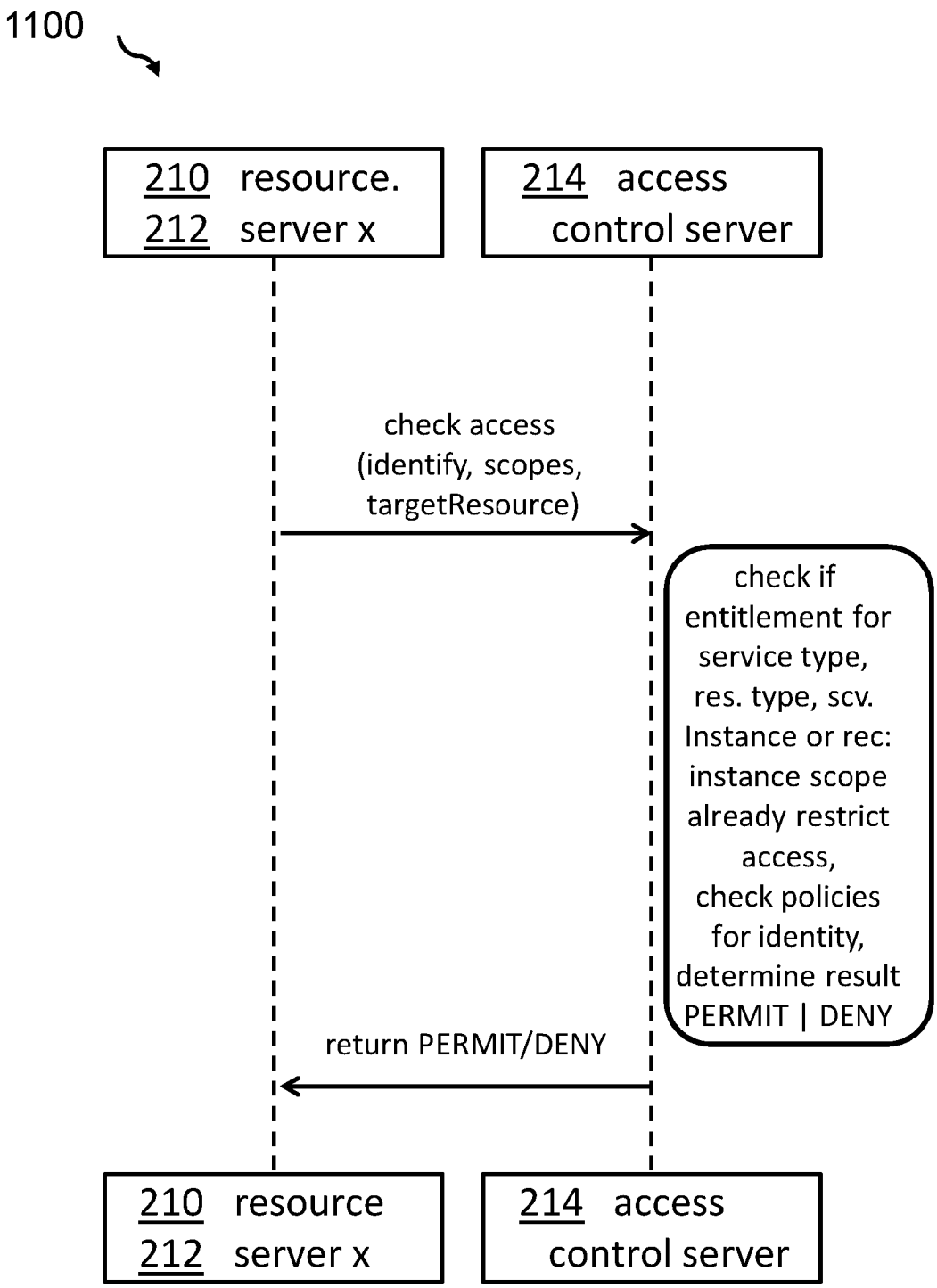

1100
210 resource.
212 server x
214 access
control server
check access
(identify, scopes,
targetResource)
check if
entitlement for
service type,
res. type, scv.
Instance or rec:
instance scope
already restrict
access,
check policies
for identity,
determine result
PERMIT | DENY
return PERMIT/DENY
210 resource
212 server x
214 access
control server

FIG. 11

Check 1: Does the list of scopes contain “IBM” ? Yes: Continue with policy evaluation No: Continue with next step
CRN of the resource to be checked. Extract the **service-name**.
Check 2: Does the list of scopes contain the service-name of the CRN? Yes: Continue with policy evaluation No: Return DENY
Policy Evaluation …

FIG. 12

Check 1: Does the list of scopes contain ibm ?
Yes: Continue with policy evaluation
No: Continue with next step

Parse the CRN of the resource to be checked. Details about the CRN syntax can be found in section "*Syntax and Semantic of limited entitlements*".
Extract if available: **crn-service-name, crn-service-instance, crn-resource-type, crn-resource**

Check 2: For each scope in the list of scopes do:
Details about the scope syntax can be found in section "*Syntax and Semantic of limited entitlements*". Split scope into **scope-service-name, scope-service-instance, scope-resource-type, scope-resource**; individual parts can be empty.

Check
(**crn-service-name** matches **scope-service-name**)
AND
((**crn-service-instance** not available and **scope-service-name** not available) OR
(**crn-service-instance** available and **scope-service-name** not available) OR
(**crn-service-instance** available and matches **scope-service-name**))
AND
((**crn-resource-type** not available and **scope-resource-type** not available) OR
(**crn-resource-type** available and **scope-resource-type** not available) OR
(**crn-resource-type** available and matches **scope-resource-type**))
AND
((**crn-resource** not available and **scope-resource** not available) OR
(**crn-resource** available and **scope-resource** not available) OR
(**crn-resource** available and matches **scope-resource**))

Does the actual scope match this check?
Yes: Continue with policy evaluation
No: Continue with next scope
No matching scope found? Return DENY

Policy Evaluation ...

FIG. 13

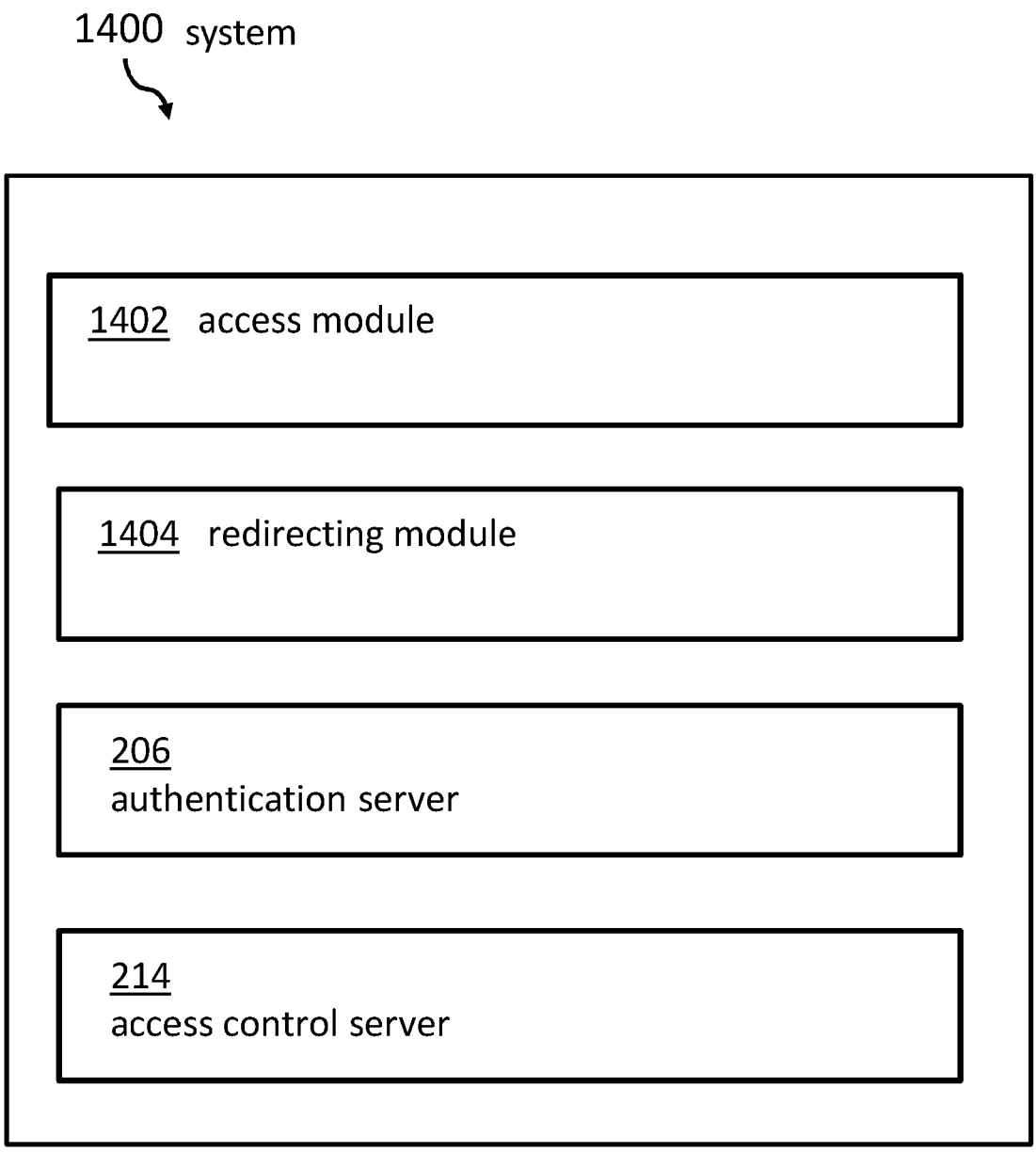

1400 system
1402 access module
1404 redirecting module
206
authentication server
214
access control server

FIG. 14

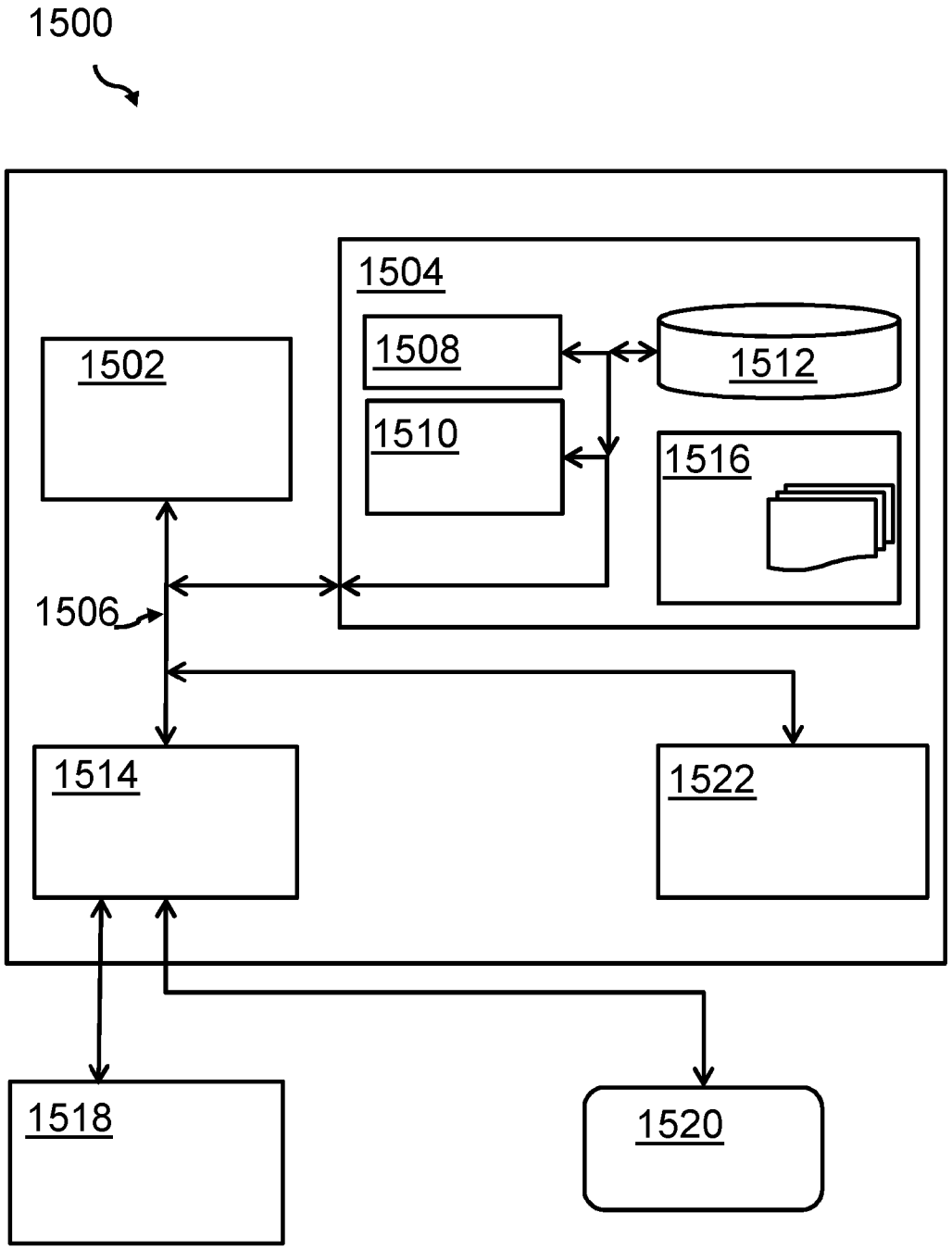


FIG. 15

FINE-GRAINED TOKEN BASED ACCESS CONTROL

BACKGROUND

Field of the Invention

[0001] The invention relates generally to a method for a token-based authorization, and more specifically, to a computer-implemented method for a token-based authorization in a data processing environment. The invention is further directed to a related access system for a token-based authentication in a data processing environment, and a related computer program product.

Technical Field

[0002] Over the last decade, token-based authentication and authorization systems have evolved. In these systems, an end-user authenticates against an authentication server and gets back a temporary token which proves the authorization of that end-user and optionally also his identity. This token is used, e.g., by Web-based user interfaces (UI) to invoke back-end services securely. As no credentials flow to those services, token-based systems enable an eco-system for services of different vendors which can leverage the same authorization system. For more complex interactions between services, it is also possible to pass-through the token from the UI through a service to another down-stream service.

16 12 22

[0003] As part of the token data, the payload usually carries sets of authorization, called “scopes”. The scopes are handled very differently in the various token-based systems. From a standardization perspective, only a few keywords for scopes are reserved, while other authorization is up to the authorization framework. Many of those frameworks adapted an action and/or role-based models which are often describing a service, a component and/or in action. For example, a function call like “database.query.read” would be such a potential scope.

SUMMARY

[0004] According to the present invention there are provided a method, a system, a computer program product and a computer program according to the independent claims.

[0005] Deleted.

[0006] The method may further comprise sending an access token together with a refresh token from the authentication server to the application if the authentication has been successful and the application has been registered at the authentication server, wherein the access token and the refresh token comprises the restricted entitlement.

[0007] The requesting execution of an operation by the application initiated by the user system may comprise invoking the operation by the application providing the access token comprising restricted entitlements, invoking the access control server by the operation, providing an identifier of the user system and the scope of the token comprising the subset of the existing entitlements to the access control server, using the subset of entitlements for filtering the existing entitlements by the access control server resulting in an access decision of the user system to the operation.

[0008] According to another aspect of the present invention, a related access system for a token-based authorization in a data processing environment may be provided.

[0009] Furthermore, embodiments may take the form of a related computer program product, accessible from a computer-usable or computer-readable medium providing program code for use, by, or in connection, with a computer or any instruction execution system. For the purpose of this description, a computer-usable or computer-readable medium may be any apparatus that may contain means for storing, communicating, propagating or transporting the program for use, by, or in connection, with the instruction execution system, apparatus, or device.

BRIEF DESCRIPTION OF THE SEVERAL VIEWS OF THE DRAWINGS

[0010] It should be noted that embodiments of the invention are described with reference to different subject-matters. In particular, some embodiments are described with reference to method type claims, whereas other embodiments are described with reference to apparatus type claims. However, a person skilled in the art will gather from the above and the following description that, unless otherwise notified, in addition to any combination of features belonging to one type of subject-matter, also any combination between features relating to different subject-matters, in particular, between features of the method type claims, and features of the apparatus type claims, is considered as to be disclosed within this document.

[0011] The aspects defined above, and further aspects of the present invention, are apparent from the examples of embodiments to be described hereinafter and are explained with reference to the examples of embodiments, but to which the invention is not limited.

[0012] Preferred embodiments of the invention will be described, by way of example only, and with reference to the following drawings:

[0013] Fig. 1 shows a block diagram of an embodiment of the inventive computer-implemented method for a token-based authorization in a data processing environment.

[0014] Fig. 2 shows a block diagram of an interaction model underlying the here proposed concept.

[0015] Fig. 3 shows a block diagram of an embodiment of an alternative initiation interaction.

[0016] Fig. 4 shows a self-explanatory communication diagram relating to Fig. 3.

[0017] Fig. 5 shows a block diagram of an alternative interaction in which the client retrieves access tokens and/or refresh tokens for the resource owner with limited entitlements.

[0018] Fig. 6 shows a block diagram of a second type of interaction 600 by which the client 204 invokes an operation on an unsecured component 208

[0019] Fig. 7 shows a self-explanatory communication diagram relating to Fig. 6.

[0020] Fig. 8 shows a block diagram of a third type of interaction in which the untrusted component interacts with resource server 1 and resource server 2.

[0021] Fig. 9 shows a self-explanatory communication diagram relating to Fig. 8.

[0022] Fig. 10 shows a block diagram of a fourth type of interaction in which the resource server 1 and the resource server 2 interact with access control server.

[0023] Fig. 11 shows a self-explanatory communication diagram relating to Fig. 10.

[0024] Fig. 12 shows an exemplary pseudo-code segment of traditional access control server decision engine.

[0025] Fig. 13 shows an exemplary pseudo-code segment of the here proposed access control server decision engine.

[0026] Fig. 14 shows a block diagram of the access system for a token-based authorization in a data processing environment.

[0027] Fig. 15 shows an embodiment of a computing system suitable for executing program code for the here described components.

DETAILED DESCRIPTION

[0028] In the context of this description, the following conventions, terms and/or expressions may be used:

[0029] The term 'token-based authorization' may denote a mechanism by which a user may receive a temporary data structure, denoted as token, from an authentication server against which he has proven his authentication. Such token may authorize the user to access predefined systems, applications and/or services on e.g., other server systems.

[0030] The term 'user system' may denote here typically a user operated device like a personal computer, a mobile device or any other computerized system operated by an end-user.

[0031] The term 'application' may denote executable software code delivering one or more services and functions after being activated from a user system. Typically, the application may be managed and executed on a server system, e.g., a server system operated in a data center and/or under the Cloud computing paradigm. Thus, the server system may typically be operated remotely from the user system.

[0032] The term 'authentication server' may denote a trusted computing environment – typically being operated by a third party to the user system and the server system – against which an end-user operating the user system – e.g., by using a Web-browser – may authenticate himself to be authorized to access predefined applications and services.

[0033] The term 'access control server' may denote a hardware or software system enabled to grant or deny an access to a specific service to an application or a user system using the application.

[0034] The term 'operation' may denote a single function of an application, or a plurality of functions and/or services of one or more applications. The type of the operation may be defined and/or restricted by a scope of the access. Examples of operations may be a database read, reading a specified subset of data, manipulating these data, combine the data, execute a search in a content management system and access a specified network address and many more software and/or hardware supported services and/or functions.

[0035] The term 'authentication credentials' may denote a set of data used by an end-user (or a related system) to authorize him for accessing predefined functions and services.

16 12 22

[0036] The term 'restricted entitlement' may refer to the fact that a user system and/or the related end-user may only have access and/or use functions that are limited if compared to all available functions of an application or a service providing entity. Thus, he may not be entitled to use the full set of functions or services but only to a restricted subset, e.g. read data but not changing the data, also the application may be enabled to do so.

[0037] The term 'scope' may denote the defined set of services and/or functions a user system or a related end-user or – in general – system user may have access to. The scope may be the complete set of services and functions available from a resource or it may be limited to only a subset.

[0038] The term 'standard semantic' may denote here a defined set of commands and functions of, e.g., a standardized authorization system like OAuth in version 2. However, also other versions may be supported.

[0039] The term 'OAuth2' may denote the known open standard for access delegation, commonly used as a way for Internet users to grant websites and/or applications access to their information on other websites but giving them the password. This mechanism is widely used by the public Internet sites to permit users to share information about their accounts with third-party applications or websites. – Generally, OAuth2, denoting version 2 of the standard, may provide to clients a “secure delegated access” to resources on behalf of a resource owner. It may specify a process for a resource owner to authorize third-party access to their resources – i.e., services and functions – without sharing their credentials (i.e., authentication credentials). Typically, access tokens, issued to the user system by the authentication server, are used to confirm an approval of the resource owner. The third-party,

i.e., the end-user with his related user system, may then use the access token to access the protected resources hosted by a server.

[0040] Additionally, some terms – in particular roles – of the preferred token-based implementation environment of OAuth2 may also be defined here:

[0041] Basically, the roles used are described in the OAuth2 specification RFC 6749, compare https://tools.ietf.org/html/rfc6749.

[0042] A resource owner (e.g., end-user, system-user) may be an entity capable of granting access to a protected resource. When the resource owner is a person, it is referred to as an end-user.

[0043] A resource server (i.e., service) may be the server hosting the protected resources, capable of accepting and responding to protected resource requests using access tokens. For the here proposed concept, the resource server may be connected to an access control server that may help determine if an operation based on a passed token is allowed or denied.

16 12 22

[0044] A client may be an application making protected resource requests on behalf of the resource owner and with its authorization. The term "client" does not imply any implementation characteristics, i.e., this can be a Web-based application, a desktop application or an application running on a different device.

[0045] An authentication server may be the server issuing access tokens to the client after successfully authenticating the resource owner (end-user) and obtaining authorization. The authentication server may be the same server as the resource server or a separate entity. A single authentication server may issue access tokens accepted by multiple resource servers.

[0046] An untrusted component may – for the here described concept and in addition to the OAuth2 defined roles, execute a script on behalf of the resource owner. For this, it receives the resource owner's access token and uses that to call one or more resource servers.

[0047] An access control server – in addition to the OAuth2 defined roles – may be meant to support resource servers in evaluating access requests by comparing properties from the passed token with previously created policies.

[0048] Furthermore, it may be noted that the function of the single authentication server according to OAuth2 has been split in the context of the here proposed concept in an authentication server and an access control server for the more fine-grained authorization and access control.

[0049] The proposed computer-implemented method for a token-based authentication in a data processing environment may offer multiple advantages and technical effects:

[0050] The proposed concept may be seamlessly integrated in an Identity and Access Management (IAM) system that uses a token-based authorization model allowing a fine-grained access control to resource management policies. According to such a model, the token may only identify the actor – i.e., the end-user – of the operation performed, but may delegate any access control to the invoked service. The token itself may only contain a minimum set of authorization properties, e.g., may only indicate if the token may be used to invoke the service and/or function or not.

[0051] Additionally, in certain scenarios, the token may be handed over to a potentially un-trusted component to execute an operation. In order to limit the potential impact of malfunctions of that component, the proposed concept discloses an elegant way to limit the authorization of a token so that a service receiving such a token will only allow operations on a list of specific service types, service instances, resource types and/or resource instances. Such a list may mix these different types. The target service and its authorization component is modified "under the hood" to honor these limitations effectively. The related restriction of authorization may be scoped by that token only, while other tokens continue to authorize execute operations based on the full set of permissions of that user.

[0052] It may also be noted that the here used tokens may have a validity that is time-dependent all-time-delimited. Thus, also refresh tokens may be used which may underlie the same operation principles as the access and/or authentication tokens.

[0053] Thus, the general concept of the proposed concept is to restrict a token's access to a subset of existing entitlements of a resource owner to a service instance and/or resource instance by defining the specific semantic of the token scope and modifying the access control without changing the existing entitlements by applying the tokens semantic scope. Thus, the defined existing entitlements may continue to exist as they are; it may only be required to define an additional subset of more delimited functions and/or services.

[0054] Thus, the currently available access control on the OAuth2 which may currently limit an access to the service only, identified by the service name, may be enhanced to a plurality of additional access limitations not available as of today and outside the standard of OAuth2, like

- limited access to a resource type within a service <service-name>.<resource-type>, e.g., kubernetes.pods,
- limited access to a resource type within a service instance <service-name>:<service-instance>.<resource-type>, e.g., kubernetes:627129310298271 2.pods
- limited access to a resource within a service <service-name>.<resource-type>:<resource>, e.g., kubernetes.pods:mypod,

- limited access to a resource within a service instance <service-name>:<service-instance>.<resource-type>:<resource>, e.g., kubernetes:627129310298271 2.pods:mypod

Only the following access restriction is available in the traditional OAuth2 approach:

- limited access to a service only <service-name>, e.g., kubernetes,
- limited access to an action of a service only <service-name>.<action>, e.g., database.read.

[0055] Furthermore, tokens may be created with a limited entitlement only; thus, potentially stolen tokens have a less blast radius. E.g., mobile applications that handle tokens could be debugged or hacked by a malicious app on the same mobile device. Having a limited set of entitlements may help reducing the potential blast radius of that stolen token.

[0056] Even if discouraged, some web applications keep tokens on the client side, i.e., in the browser's JavaScript® client. Also here, tokens with limited entitlements have a less potential destructive radius in case of cross-site scripting attacks.

16 12 22

[0057] According to a preferred embodiment of the method, the standard semantic is based on OAuth2. Thus, the used basis for the here proposed concept may be integrated into an already well defined set of protocols and functions by a refining or improving the already more coarse-grained access control mechanisms to implement access control mechanisms allowing a more fine-grained control of an access to services and functions. Hence, the widely used OAuth security functions – currently available in version 2 – are enhanced and improved. It can be expected that the here proposed enhancements may also be used in a later version of OAuth.

[0058] According to one advantageous embodiment of the method, the application may be an unsecured component, e.g., Jupyter Notebooks, or similar. Jupyter Notebooks – or Jupyter documents may denote documents produced by the known Jupyter Notebook App which may comprise both, computer code and rich text elements with paragraphs, equations, figures, links, etc. Thereby, the Jupyter Notebook App may be a server-client application that may allow editing and running notebook documents, e.g., on a server system via a Web-browser.

[0059] According to one useful example, the request for a restricted entitlement may comprise, in the authentication credentials, a first request, wherein the first request may comprise a maximum subset of existing entitlements. Thus, the initial authorization may allow accessing the complete set of services and functions and may not be restricted to a subset or a restricted entitlement. Such a restriction may be defined or requested at a later time by requesting a refresh token with the limitation.

[0060] According to another useful embodiment of the method, a subsequent request for a refresh token (and the related access token) – i.e., accessing the application and/or functions or services subsequent to the initial request

– may also comprise a request for a further restricted entitlement. This way, a differentiation between the first access and subsequent accesses may be managed in an elegant way.

[0061] According to a permissive embodiment of the method, the operation may comprise at least one selected out of the group comprising an access to predefined data, a database access, a file access, a predefined application programming interface (e.g., some specific analysis or machine learning functions), and/or an access to a specific subnet of a network. In general, the operation may be related to any available low-level function, service or resource available in the computing environment.

[0062] According to one advantageous embodiment of the method, the access token may have a predefined validity time. Thus, if the access continues to be authorized, the authorization may have to be refreshed, e.g., by a related refresh token. The validity time may be made significantly shorter than default token life times.

[0063] According to another advantageous embodiment of the method, the predefined validity time may be updatable by a privileged user system. Hence, the fine-grained access control may be manageable as any other access control to a group of resources, functions and/or services in a computing environment. An existing IAM system may be used for this task.

[0064] In the following, a detailed description of the figures will be given. All instructions in the figures are schematic. Firstly, a block diagram of an embodiment of the inventive computer-implemented method for a token-based authorization in a data processing environment is given. Afterwards, further embodiments, as well as embodiments of the access system for a token-based authorization in a data processing environment, will be described.

[0065] Fig. 1 shows a block diagram of an embodiment of the computer-implemented method 100 for a token-based authorization in a data processing environment. The data processing environment comprises at least a user system, an application, an authentication server and an access control server. The user system is connected via a network connection to a server executing the application, wherein the application provides an access to operation – e.g., a database access –wherein at least the operation is identifiable by its identifier.

[0066] The method comprises accessing, 102, the application via a user system request – e.g., via a Web-browser – redirecting, 104, the user access request to an authentication server, and authenticating, 106, the user by authentication credentials exchanged between the authentication server and the application. Thereby, the authentication credentials comprise a request for a restricted entitlement using a standard semantic – in particular, OAuth2 – of a scope – according to OAuth2.0 terminology –, wherein the restricted entitlement represents a subset of existing entitlements managed by the access control server for a resource. The data required for the restricted entitlement can be embedded into or being part of the standard semantic, i.e., the standard protocol(s) (i.e., OAuth2) used.

[0067] The method 100 comprises further sending, 108, an access token together with a refresh token from the authentication server to the application if the authentication has been successful and the application has been registered at the authentication server, wherein the access token and the refresh token comprises the restricted entitlement.

[0068] Additionally, the method comprises requesting, 110, execution of an operation by the application initiated by the user system, wherein the requesting comprises invoking, 112, the operation by the application providing the access token comprising restricted entitlements, invoking, 114, the access control server by the operation, providing, 116, an identifier of the user system and the scope of the token comprising the subset of the existing entitlements to the access control server, and using, 118, the subset of entitlements for filtering the existing entitlements by the access control server resulting in an access decision of the user system to the operation.

[0069] Before a continuation of the figure description, the here used syntax and semantic of limited entitlements should be explained. To understand the semantic and syntax of limited entitlements in the preferred implementation, an understanding of Cloud Resource Names (CRNs) as they are used, e.g., inside IBM Cloud and its Identity and Access Management system is useful.

[0070] The entitlements, CRNs and IAM today may be characterized as follows: Any service and resource in, e.g., the Cloud environment, is uniquely identifiable via a CRN. The base canonical format of a CRN is:
crn:<version>:<cname>:<ctype>:<service-name>:<location>:<scope>:
<service-instance>:<resource-type>:<resource>
The parameters are described in table 1.

[0071] Table 1:

Field	Content
<version>	Currently only v1 is supported
<cname>	For Cloud public: bluemix
<ctype>	For Cloud public: public
<service-name>	Unique identifying name for a service type
<location>	Location of the resource. Many values possible, global for global resources
<scope>	Typically an account scope in the format a/<account id>
<service-instance>	Identifier for a service instance, if available. Can be blank.
<resource-type>	Optional resource type within this service. Can be blank.
<resource>	Optional resource identifier. Can be blank.

[0072] The access control component of a Cloud IAM stores authorizations in policies. Policies allow individual users access to services or resources in the Cloud computing environment. A sample policy would be:

Identity: IBMid-2700XYZ

Role: Administrator

Target: crn:v1:bluemix:public:kubernetes::a/1234::::

[0073] This policy allows the user with identity "IBMid-2700XYZ" any action that the role "administrator" covers on any service or resource that belongs to the service type "kubernetes" in the account "1234".

[0074] When a service wants to decide if a certain action is allowed, the service provides the following information to the access control server: identity / account / required action / target resource / scopes.

[0075] Today, a first check is done if the scopes contain the keyword "ibm" or if the scopes contain the service name that is used inside the target resource. If not, the policy check will return "deny". Now, all potential applying policies are evaluated based on account and identity. If a matching policy is found, the policy check will return "PERMIT".

[0076] It should be clear that the specific identifiers are only examples. Any other identifiers and/or variables can be used without leaving the scope of the here proposed concept.

[0077] Fig. 2 shows a block diagram of an interaction model 200 underlying the here proposed concept. The resource owner 202 – typically the end-user sitting in front of a Web-browser – is interacting with a client 204 – typically a Web-application that is offering a service to the end-user. For an interactive login experience, the resource owner 202 is redirected to the authentication server 206 for authentication and authorization of the client's access request. The client 204 – typically a Web-application – interacts with the authentication server 206 to retrieve or refresh an access token. Additionally, as part of the here proposed concept, the client 204 also interacts with the untrusted component 208 to execute an operation.

[0078] The untrusted or unsecure component 208 interacts with resource server 1, 210 and resource server 2, 212 to execute its operation. The resource server 1, 210 and the resource server 2, 212 interact with the access control server 214 to determine if a requested operation is permitted or denied.

[0079] In the following preferred implementation, the exemplary two resource servers 210, 212 (many more may be used) are using exemplary the following service types, service instances, resource types, and resource instances according to table 2.

[0080] Table 2:

resource server 1, 210		resource server 2, 212	
Service Type:	COS	Service Type:	COS
Service Instance:	1	Service Instance:	2
Resource Type:	bucket	Resource Type:	bucket
Resource Instance:	123	Resource Instance:	456

[0081] To execute the interaction in the here proposed concept, the following steps may be required:

1. The client 204 retrieves access tokens and refresh tokens for the resource owner 202 (covered by OAuth2 standard) with limited entitlements (part of the novel concept).
2. The client 204 invokes an operation on an untrusted component (normal behavior in a Web-environment).
3. The untrusted component 210 interacts with resource Server 1, 210 and 2, 212 (normal behavior in a Web-environment).
4. The resource server 1, 210 and the resource server 2, 212 interact with access control server 214 to decide if the requested operation should be permitted or not (part of the novel concept).

16 12 22

[0082] As prerequisites may be named: The resource owner 202 was assigned policies on the access control server 214 which allows the resource owner 202 to access resources on both, resource server 1, 210 and resource server 2, 212. These policies are not changed during the following interaction, i.e., part of the here newly proposed concept.

[0083] Fig. 3 shows a block diagram 300 of an embodiment of an alternative initiation interaction: the client 204 retrieves the access and/or the refresh tokens for the resource owner 202 with limited entitlements. The client 204 retrieves an access and/or refresh token for the resource owner 202 by one of the grant types that are described in the OAuth2 standard. In addition, and as part of the novel concept, the client 204 already passes a list of entitlements to the authentication server 206 to limit the potential access of the access token to be generated. Specifying a list of scopes during token creation is part of the OAuth2 standard, but the semantic of the scopes and usage in later interactions are part of the here newly proposed concept. For this sample interaction, it may be assumed that the list of restricted entitlements is just one element “COS:1.bucket:123”.

[0084] Fig. 4 shows a related self-explanatory communication diagram 400 showing the client 204, the authentication server 206 and the resource owner 202 with respective messages flows.

[0085] Fig. 5 shows a block diagram of an alternative interaction 500 in which the client 204 retrieves access tokens and/or refresh tokens for the resource owner 202 with limited entitlements. The client 204 retrieves an access and/or refresh token for the resource owner 202 by one of the grant types that are described in the OAuth2

standard. In a subsequent step, the client 204 refreshes the access and/or refresh token using the refresh_token grant type and passes a list of restricted entitlements to the authentication server 206 to limit the potential access of the access token and refresh token to be generated. Specifying a list of scopes during token refresh is part of the OAuth2 standard, but the semantic of the scopes and usage in later interactions are part of the here newly proposed concept. For this sample interaction, it may be assumed that the list of restricted entitlements is just one element "COS:1.bucket:123".

[0086] Fig. 6 shows a block diagram of a second type of interaction 600 by which the client 204 invokes an operation on an unsecured component 208. After the interaction according to Figs 3, 4, 5, there is an access token that carries a set of limited entitlements. In this description, it may be assumed that it comprises an entitlement to access resource server 1, 210, but not to resource server 2, 212.

[0087] The client 204 invokes an operation on an unsecured or untrusted component 208. Untrusted components 208 are e.g., scripts that have been developed by others and have not been reviewed or verified to be trustworthy. The invocation of the untrusted component 208 comprises the access token with limited entitlement, which is part of the here proposed concept.

16 12 22

[0088] Fig. 7 shows the related self-explanatory communication diagram 700, showing the client 204, and the unsecure component 208 with respective messages flows and processing.

[0089] Fig. 8 shows a block diagram 800 of a third type of interaction in which the untrusted component 208 interacts with resource server 1, 210 and resource server 2, 212. As part of the execution of that untrusted component 208, the script starts an operation on resource server 1, 210 providing the access token with limited entitlements. As part of its normal processing, the resource server 1, 210 interacts with the access control server 214 to determine if the resource owner represented by the access token can execute the operation. The resource server 1, 210 is providing the identity inside the access token, the entitlements (scopes) of the access token (COS:1.bucket:123) and the target resource (e.g., crn:v1:bluemix:public:COS::a/1234:1:bucket:123) to execute the operation which is an explicit component of the here proposed concept.

[0090] Based on the entitlement inside the access token and the existing policies, the access control server 214 determines to permit the operation. A similar interaction with resource server 2, 212 returns "DENY", as the entitlements (scopes) of the access token (COS:1.bucket:123) do not match the target resource (crn:v1:blumix:public:COS::a/1234:2:bucket:456).

[0091] Fig. 9 shows the related self-explanatory communication diagram 900 showing the untrusted component 208, and the resource server 1, 210 and the resource server 2, 212 with respective message flows and processing.

[0092] Fig. 10 shows a block diagram of a fourth type of interaction 1000 in which the resource server 1, 210 and the resource server 2, 212 interact with access control server 214. The untrusted component 208 starts an operation on resource server 1, 210 and then on resource server 2, 212 providing the access token with limited entitlements. Again, as part of normal OAuth2 processing, the resource servers 210, 212 interact with the access control server 214 to determine if the operation is allowed to be executed. The resource servers 210, 212 are providing the identity inside and the entitlements (scopes) of the access token and the target resource to execute the operation.

[0093] As the restricted entitlements inside the token do not contain the service instance of resource server 2, 212, the access control server 214 will decide to deny the operation of resource server 2, 212 even that the policies of the identity inside the access token would have a matching policy in the system, which is again an explicit portion of the here newly proposed concept.

[0094] Fig. 11 shows the related self-explanatory communication diagram 1100 showing the resource server 1, 210, the resource server 2, 212 and the access control server 214 with respective message flows and processing.

[0095] The next two Figs 12, 13 show a comparison of pseudo-code segments of a traditional access control server decision engine (Fig. 12) in comparison to the one required for the here proposed concept (Fig. 13). As can easily be seen, a more complex determination for a correct fine-grained access control for restrictive entitlement is required. However, it guarantees a more flexible and secure resource management if compared to the standard OAuth2 access control.

[0096] For completeness reasons, Fig. 14 shows a block diagram of the access system 1400 for a token-based authentication in a data processing environment. The data processing environment comprises at least a user system, an application, an authentication server and an access control server (all not shown). Thereby, the user system is connected via a network connection to a server executing the application, and the application provides an access to the operation. At least the operation is identifiable by its identifier. It may also be noted that all components discussed here are connected to a network connecting the components.

[0097] The access system 1400 comprises an access module 1402 adapted for accessing the application via a user system request, a redirecting module 1404 adapted for redirecting the user access request to an authentication server 206. The authentication server 206 is adapted for authenticating the user by authentication credentials exchanged between the authentication server and the application, wherein the authentication credentials comprise a request for a restricted entitlement using a standard semantic of a scope, wherein the restricted entitlement represents a subset of existing entitlements managed by the access control server for a resource.

[0098] The access system 1400 comprises further a sender adapted for sending an access token together with a refresh token from the authentication server 206 to the application if the authentication has been successful and the application has been registered at the authentication server 206, wherein the access token and the refresh token comprises the restricted entitlement.

[0099] Thereby, the user system is adapted for requesting execution of an operation by the application initiated by the user system, wherein the requesting execution of an operation comprises invoking the operation by the application providing the access token comprising restricted entitlements, invoking the access control server by the operation, providing an identifier of the user system and the scope of the token comprising the subset of the existing entitlements to the access control server 214, using the subset of entitlements for filtering the existing entitlements by the access control server 214 resulting in an access decision of the user system to the operation.

[0100] Embodiments of the invention may be implemented together with virtually any type of computer, regardless of the platform being suitable for storing and/or executing program code. Fig. 15 shows, as an example, a computing system 1500 suitable for executing program code related to the proposed method. Virtually any component of the processing environment can be represented by a computing system according to Fig. 15

[0101] The computing system 1500 is only one example of a suitable computer system, and is not intended to suggest any limitation as to the scope of use or functionality of embodiments of the invention described herein, regardless, whether the computer system 1500 is capable of being implemented and/or performing any of the functionality set forth hereinabove. In the computer system 1500, there are components, which are operational with numerous other general purpose or special purpose computing system environments or configurations. Examples of well-known computing systems, environments, and/or configurations that may be suitable for use with computer system/server 1500 include, but are not limited to, personal computer systems, server computer systems, thin clients, thick clients, hand-held or laptop devices, multiprocessor systems, microprocessor-based systems, set top boxes, programmable consumer electronics, network PCs, minicomputer systems, mainframe computer systems, and distributed cloud computing environments that include any of the above systems or devices, and the like. Computer system/server 1500 may be described in the general context of computer system-executable instructions, such as program modules, being executed by a computer system 1500. Generally, program modules may include routines, programs, objects, components, logic, data structures, and so on that perform particular tasks or implement particular abstract data types. Computer system/server 1500 may be practiced in distributed cloud computing environments where tasks are performed by remote processing devices that are linked through a communications network. In a distributed cloud computing environment, program modules may be located in both, local and remote computer system storage media, including memory storage devices.

[0102] As shown in the figure, computer system/server 1500 is shown in the form of a general-purpose computing device. The components of computer system/server 1500 may include, but are not limited to, one or more processors or processing units 1502, a system memory 1504, and a bus 1506 that couple various system components including system memory 1504 to the processor 1502. Bus 1506 represents one or more of any of several types of bus structures, including a memory bus or memory controller, a peripheral bus, an accelerated graphics port, and a processor or local bus using any of a variety of bus architectures. By way of example, and not limiting, such architectures include Industry Standard Architecture (ISA) bus, Micro Channel Architecture (MCA) bus, Enhanced ISA (EISA) bus, Video Electronics Standards Association (VESA) local bus, and Peripheral Component Interconnects (PCI) bus. Computer system/server 1500 typically includes a variety of computer system readable media. Such media may be any available media that is accessible by computer system/server 1500, and it includes both, volatile and non-volatile media, removable and non-removable media.

[0103] The system memory 1504 may include computer system readable media in the form of volatile memory, such as random access memory (RAM) 1508 and/or cache memory 1510. Computer system/server 1500 may further include other removable/non-removable, volatile/non-volatile computer system storage media. By way of example only, a storage system 1512 may be provided for reading from and writing to a non-removable, non-volatile magnetic media (not shown and typically called a 'hard drive'). Although not shown, a magnetic disk drive for reading from and writing to a removable, non-volatile magnetic disk (e.g., a 'floppy disk'), and an optical disk drive for reading from or writing to a removable, non-volatile optical disk such as a CD-ROM, DVD-ROM or other optical media may be provided. In such instances, each can be connected to bus 1506 by one or more data media interfaces. As will be further depicted and described below, memory 1504 may include at least one program product having a set (e.g., at least one) of program modules that are configured to carry out the functions of embodiments of the invention.

[0104] The program/utility, having a set (at least one) of program modules 1516, may be stored in memory 1504 by way of example, and not limiting, as well as an operating system, one or more application programs, other program modules, and program data. Each of the operating systems, one or more application programs, other program modules, and program data or some combination thereof, may include an implementation of a networking environment. Program modules 1516 generally carry out the functions and/or methodologies of embodiments of the invention, as described herein.

[0105] The computer system/server 1500 may also communicate with one or more external devices 1518 such as a keyboard, a pointing device, a display 1520, etc.; one or more devices that enable a user to interact with computer system/server 1500; and/or any devices (e.g., network card, modem, etc.) that enable computer system/server 1500 to communicate with one or more other computing devices. Such communication can occur via Input/Output (I/O) interfaces 1514. Still yet, computer system/server 1500 may communicate with one or more

networks such as a local area network (LAN), a general wide area network (WAN), and/or a public network (e.g., the Internet) via network adapter 1522. As depicted, network adapter 1522 may communicate with the other components of the computer system/server 1500 via bus 1506. It should be understood that, although not shown, other hardware and/or software components could be used in conjunction with computer system/server 1500. Examples, include, but are not limited to: microcode, device drivers, redundant processing units, external disk drive arrays, RAID systems, tape drives, and data archival storage systems, etc.

[0106] The descriptions of the various embodiments of the present invention have been presented for purposes of illustration, but are not intended to be exhaustive or limited to the embodiments disclosed. Many modifications and variations will be apparent to those of ordinary skills in the art without departing from the scope and spirit of the described embodiments. The terminology used herein was chosen to best explain the principles of the embodiments, the practical application or technical improvement over technologies found in the marketplace, or to enable others of ordinary skills in the art to understand the embodiments disclosed herein.

[0107] The present invention may be embodied as a system, a method, and/or a computer program product. The computer program product may include a computer readable storage medium (or media) having computer readable program instructions thereon for causing a processor to carry out aspects of the present invention.

[0108] The medium may be an electronic, magnetic, optical, electromagnetic, infrared or a semi-conductor system for a propagation medium. Examples of a computer-readable medium may include a semi-conductor or solid state memory, magnetic tape, a removable computer diskette, a random access memory (RAM), a read-only memory (ROM), a rigid magnetic disk and an optical disk. Current examples of optical disks include compact disk-read only memory (CD-ROM), compact disk-read/write (CD-R/W), DVD and Blu-Ray-Disk.

[0109] The computer readable storage medium can be a tangible device that can retain and store instructions for use by an instruction execution device. The computer readable storage medium may be, for example, but is not limited to, an electronic storage device, a magnetic storage device, an optical storage device, an electromagnetic storage device, a semiconductor storage device, or any suitable combination of the foregoing. A non-exhaustive list of more specific examples of the computer readable storage medium includes the following: a portable computer diskette, a hard disk, a random access memory (RAM), a read-only memory (ROM), an erasable programmable read-only memory (EPROM or Flash memory), a static random access memory (SRAM), a portable compact disk read-only memory (CD-ROM), a digital versatile disk (DVD), a memory stick, a floppy disk, a mechanically encoded device such as punch-cards or raised structures in a groove having instructions recorded thereon, and any suitable combination of the foregoing. A computer readable storage medium, as used herein, is not to be construed as being transitory signals per se, such as radio waves or other freely propagating electromagnetic waves, electromagnetic

waves propagating through a waveguide or other transmission media (e.g., light pulses passing through a fiber-optic cable), or electrical signals transmitted through a wire.

[0110] Computer readable program instructions described herein can be downloaded to respective computing/processing devices from a computer readable storage medium or to an external computer or external storage device via a network, for example, the Internet, a local area network, a wide area network and/or a wireless network. The network may comprise copper transmission cables, optical transmission fibers, wireless transmission, routers, firewalls, switches, gateway computers and/or edge servers. A network adapter card or network interface in each computing/processing device receives computer readable program instructions from the network and forwards the computer readable program instructions for storage in a computer readable storage medium within the respective computing/processing device.

[0111] Computer readable program instructions for carrying out operations of the present invention may be assembler instructions, instruction-set-architecture (ISA) instructions, machine instructions, machine dependent instructions, microcode, firmware instructions, state-setting data, or either source code or object code written in any combination of one or more programming languages, including an object-oriented programming language such as Smalltalk, C++ or the like, and conventional procedural programming languages, such as the "C" programming language or similar programming languages. The computer readable program instructions may execute entirely on the user's computer, partly on the user's computer as a stand-alone software package, partly on the user's computer and partly on a remote computer or entirely on the remote computer or server. In the latter scenario, the remote computer may be connected to the user's computer through any type of network, including a local area network (LAN) or a wide area network (WAN), or the connection may be made to an external computer (for example, through the Internet using an Internet Service Provider). In some embodiments, electronic circuitry including, for example, programmable logic circuitry, field-programmable gate arrays (FPGA), or programmable logic arrays (PLA) may execute the computer readable program instructions by utilizing state information of the computer readable program instructions to personalize the electronic circuitry, in order to perform aspects of the present invention.

[0112] Aspects of the present invention are described herein with reference to flowchart illustrations and/or block diagrams of methods, apparatus (systems), and computer program products according to embodiments of the invention. It will be understood that each block of the flowchart illustrations and/or block diagrams, and combinations of blocks in the flowchart illustrations and/or block diagrams, can be implemented by computer readable program instructions.

[0113] These computer readable program instructions may be provided to a processor of a general purpose computer, special purpose computer, or other programmable data processing apparatus to produce a machine,

such that the instructions, which execute via the processor of the computer or other programmable data processing apparatus, create means for implementing the functions/acts specified in the flowchart and/or block diagram block or blocks. These computer readable program instructions may also be stored in a computer readable storage medium that can direct a computer, a programmable data processing apparatus, and/or other devices to function in a particular manner, such that the computer readable storage medium having instructions stored therein comprises an article of manufacture including instructions which implement aspects of the function/act specified in the flowchart and/or block diagram block or blocks.

[0114] The computer readable program instructions may also be loaded onto a computer, other programmable data processing apparatuses, or another device to cause a series of operational steps to be performed on the computer, other programmable apparatus or other device to produce a computer implemented process, such that the instructions which execute on the computer, other programmable apparatuses, or another device implement the functions/acts specified in the flowchart and/or block diagram block or blocks.

[0115] The flowcharts and/or block diagrams in the figures illustrate the architecture, functionality, and operation of possible implementations of systems, methods, and computer program products according to various embodiments of the present invention. In this regard, each block in the flowchart or block diagrams may represent a module, segment, or portion of instructions, which comprises one or more executable instructions for implementing the specified logical function(s). In some alternative implementations, the functions noted in the block may occur out of the order noted in the figures. For example, two blocks shown in succession may, in fact, be executed substantially concurrently, or the blocks may sometimes be executed in the reverse order, depending upon the functionality involved. It will also be noted that each block of the block diagrams and/or flowchart illustration, and combinations of blocks in the block diagrams and/or flowchart illustration, can be implemented by special purpose hardware-based systems that perform the specified functions or act or carry out combinations of special purpose hardware and computer instructions.

[0116] The terminology used herein is for the purpose of describing particular embodiments only and is not intended to limit the invention. As used herein, the singular forms "a", "an" and "the" are intended to include the plural forms as well, unless the context clearly indicates otherwise. It will further be understood that the terms "comprises" and/or "comprising," when used in this specification, specify the presence of stated features, integers, steps, operations, elements, and/or components, but do not preclude the presence or addition of one or more other features, integers, steps, operations, elements, components, and/or groups thereof.

[0117] The corresponding structures, materials, acts, and equivalents of all means or steps plus function elements in the claims below are intended to include any structure, material, or act for performing the function in combination with other claimed elements, as specifically claimed. The description of the present invention has been

presented for purposes of illustration and description, but is not intended to be exhaustive or limited to the invention in the form disclosed. Many modifications and variations will be apparent to those of ordinary skills in the art without departing from the scope and spirit of the invention. The embodiments are chosen and described in order to best explain the principles of the invention and the practical application, and to enable others of ordinary skills in the art to understand the invention for various embodiments with various modifications, as are suited to the particular use contemplated.

CLAIMS

1. A computer-implemented method for a token-based authorization in a data processing environment, wherein the data processing environment comprises at least a user system, an application, an authentication server and an access control server, wherein the user system is connected via a network connection to a server executing the application, wherein the application provides an access to an operation, wherein at least the operation is identifiable by its identifier, wherein the application is an unsecured component, wherein the method comprises:

interacting, by the user system, with the application via a user system request,

redirecting the user system request to an authentication server,

authenticating the user system by authentication credentials exchanged between the authentication server and the application, wherein the authentication credentials comprise a request for a restricted entitlement to a resource using a standard semantic of a scope, wherein the restricted entitlement comprises a subset of existing entitlements managed by the access control server for the resource, by retrieving an access token with a refresh token from the authentication server, wherein the access token and the refresh token comprise the restricted entitlement,

requesting execution of an operation by the application initiated by the user system comprising:

invoking the operation by the application comprising providing the access token,

invoking the access control server by the operation,

providing an identifier of the user system and the scope of the token comprising the restricted entitlement to the access control server, and

using the restricted entitlement for filtering the existing entitlements by the access control server resulting in an access decision of the user system to the operation.

11 08 23

2. The method according to claim 1, wherein the standard semantic is based on OAuth2.

3. The method according to any one of the preceding claims, wherein the operation comprises at least one selected out of the group comprising an access to predefined data, a database access, a file access, a predefined application programming interface, an access to a specific subnet of a network.

4. The method according to any one of the preceding claims, wherein the access token has a predefined validity time.

5. The method according to claim 4, wherein predefined validity time is updatable by a privileged user system.

6. An access system for a token-based authorization in a data processing environment, wherein the data processing environment comprises at least a user system, an application, an authentication server and an access control server, wherein the user system is connected via a network connection to a server executing the application, wherein the application provides an access to operation, wherein at least the operation is identifiable by its identifier, wherein the application is an unsecured component, wherein the access system comprises:

an accessing module adapted for accessing the application via a user system request,

a redirecting module adapted for redirecting the user system request to an authentication server,

an authentication server adapted for authenticating the user system by authentication credentials exchanged between the authentication server and the application, wherein the authentication credentials comprise a request for a restricted entitlement to a resource using a standard semantic of a scope, wherein the restricted entitlement comprises a subset of existing entitlements managed by the access control server for the resource by retrieving an access token with a refresh token from the authentication server, wherein the access token and the refresh token comprise the restricted entitlement,

wherein the user system is adapted for requesting execution of an operation by the application initiated by the user system, wherein the requesting execution of an operation comprises:

invoking the operation by the application comprising providing the access token,

invoking the access control server by the operation,

providing an identifier of the user system and the scope of the token comprising the restricted entitlement to the access control server, and

using the restricted entitlement for filtering the existing entitlements by the access control server resulting in an access decision of the user system to the operation.

11 08 23

7. The system according to claim 6, wherein the standard semantic is based on OAuth2.

8. The system according to either of claims 6 or 7, wherein the operation comprises at least one selected out of the group comprising an access to predefined data, a database access, a file access, a predefined application programming interface, an access to a specific subnet of a network.

9. The system according to any one of claims 6 to 8, wherein the access token has a predefined validity time.

10. The system according to claim 9, wherein predefined validity time is updatable by a privileged user system.

11. A computer program product for a token-based authorization in a data processing environment, wherein the data processing environment comprises at least a user system, an application, an authentication server and an access control server, wherein the user system is connected via a network connection to a server executing the application, wherein the application provides an access to operation, wherein at least the operation is identifiable by

its identifier, the computer program product comprising: a computer readable storage medium readable by a processing circuit and storing instructions for execution by the processing circuit for performing a method according to any of claims 1 to 5.

12. A computer program stored on a computer readable medium and loadable into the internal memory of a digital computer, comprising software code portions, when said program is run on a computer, for performing the method of any of claims 1 to 5.

11 08 23