(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
03 March 2022 (03.03.2022)

WIPO | PCT

(10) International Publication Number
WO 2022/046070 A1

(51) International Patent Classification:
G06F 21/62 (2013.01) *G06F 9/54* (2006.01)

(21) International Application Number:
PCT/US2020/048385

(22) International Filing Date:
28 August 2020 (28.08.2020)

(25) Filing Language: English

(26) Publication Language: English

(71) Applicant: **HEWLETT-PACKARD DEVELOPMENT COMPANY, L.P.** [US/US]; 10300 Energy Drive, Spring, Texas 77389 (US).

(72) Inventors: **WATTIAU, Gaëtan**; 1 Redcliff Street, Bristol BS1 6NP (GB). **SCHIFFMAN, Joshua Serratelli**; 1 Redcliff Street, Bristol BS1 6NP (GB). **BENMOUFFEK, Sibylle Eugénie Blanche**; 1 Redcliff Street, Bristol BS1 6NP (GB). **LAING, Thalia May**; 1 Redcliff Street, Bristol BS1 6NP (GB).

(74) Agent: **WOODWORTH, Jeffrey C.** et al.; HP Inc., Mail Stop 35, 3390 E. Harmony Road, Fort Collins, Colorado 80528 (US).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DJ, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IR, IS, IT, JO, JP, KE, KG, KH, KN, KP, KR, KW, KZ, LA, LC, LK, LR, LS, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW,

(54) Title: ENFORCING WORKFLOW SECURITY

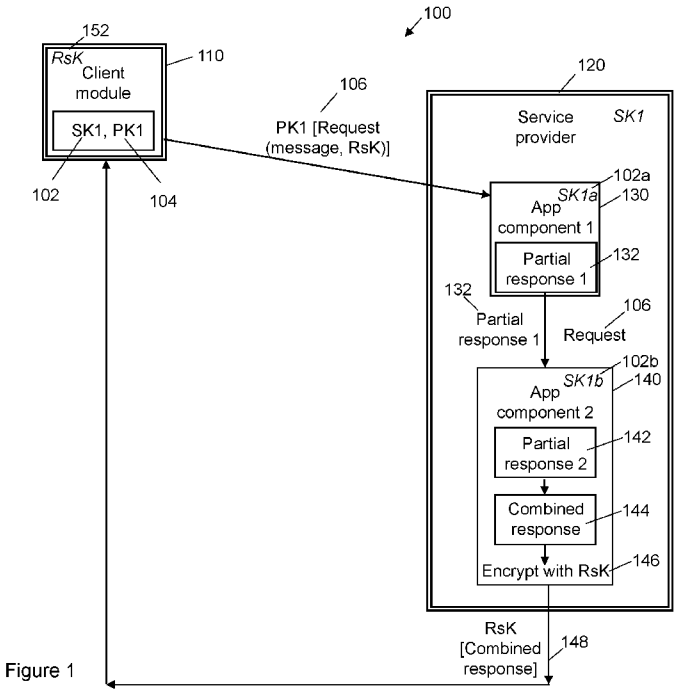

Figure 1

(57) Abstract: A system comprising a client module to generate a user request for a service provider having a plurality of application components, by encrypting a request message using a public key, wherein the public key corresponds to a private key shared between a plurality of application components of the service provider. A first of the application components is to generate a first partial response by applying a first share of the private key to the received user request, and provide the first partial response to a further of the application components. The further application component is to generate a further partial response to the received user request by applying a further share of the private key to the received user request; combine the partial responses to obtain a combined response; and, based on the partial responses reaching a threshold number, provide the combined response in response to the user request.

[Continued on next page]

WO 2022/046070 A1

SA, SC, SD, SE, SG, SK, SL, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, WS, ZA, ZM, ZW.

(84) **Designated States** *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, ST, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG).

Declarations under Rule 4.17:

— *as to the identity of the inventor (Rule 4.17(i))*

Published:

— *with international search report (Art. 21(3))*

ENFORCING WORKFLOW SECURITY

BACKGROUND

[0001] A computer system may involve a client, or user, accessing a service, or function, from a service provider. The client may access the functionality if they are authorised to do so. Managing client authorisation may involve, for example, verifying that a client is authorised to access a particular service from a service provider. This may be achieved, by issuing the client with an authorisation to indicate that the client is authorised.

BRIEF INTRODUCTION OF THE DRAWINGS

[0002] Various features of the present disclosure will be apparent from the detailed description which follows, taken in conjunction with the accompanying drawings, which together illustrate features of the present disclosure, and wherein:

[0003] Figure 1 shows a schematic representation of a system according to an example of the disclosure;

[0004] Figures 2a and 2b illustrate schematic representations of providing a combined response according to examples of the disclosure;

[0005] Figures 3a and 3b show schematic representations of distributing a private key according to examples of the disclosure;

[0006] Figure 4 shows a schematic representation of a system comprising a verification component according to an example of the disclosure;

[0007] Figure 5 shows a schematic representation of certificate verification according to an example of the disclosure;

[0008] Figures 6 and 7 show example methods according to examples of the disclosure; and

[0009] Figure 8 shows an example apparatus comprising a processor and memory according to examples of the disclosure.

DETAILED DESCRIPTION

[0010] In the following description, for purposes of explanation, numerous specific details of certain examples are set forth. Reference in the specification to "an example" or similar language means that a particular feature, structure, or characteristic described in

connection with the example is included in at least that one example, but not necessarily in other examples.

[0011] A client may wish to access a service from a service provider. The client may have to prove they are authorised to access the requested service. This may be achieved, by issuing the client with an authorisation to indicate that they are authorised.

[0012] An application, such as an application providing a client service, may be composed of multiple micro-services and/or multiple containerized / virtualized components. Application logic may process a workload by a set of components forming a pipeline. That is, an application may use a series of components in a defined order to provide the requested service. This may be for a number of reasons including security, integrity, or business logic. Some of those components may be security critical (e.g. may have access to sensitive keys or resources).

[0013] Bypassing part of the pipeline may lead to undesirable or illicit results. For example, if an attacker is able to bypass a password entry component, the attacker may be able to access the service without authorisation which the password would have provided.

[0014] It may be challenging to provide both a user of a distributed application, and the owner/provider of a distributed application, with information to access/provide the application features, and maintain data privacy. A user may be concerned with sharing data, by providing all their data to a single party, and/or providing data in a way which an attacker can intercept. An owner/service provider may be concerned with ensuring a requesting user is legitimately authorised to access the requested features.

[0015] For example, a client/user may wish to interact with a distributed application which is run and owned by a service owner. The client may send a request to a distributed application, the application comprising a plurality of nodes organised in a particular order of execution (this may be termed a “pipeline”), and wish to receive a response back. The client may have the following security requirements:

a. Request data should be revealed to necessary nodes and not to unnecessary nodes.
b. Response data should be correct and revealed to necessary nodes and not to unnecessary nodes.
c. Steps of the pipeline should be respected prior to processing by a sensitive component. This may be because the client may send an encrypted request, which may be decrypted by certain nodes in the pipeline, and other nodes/components later in the pipeline which don't use plaintext (a

decrypted request) may still be necessary for integrity of the response. For example a component not requiring plaintext may be in charge of orchestrating the operations, in charge of verifying other components, or used for access control, auditing, or billing, for example.

[0016] However, satisfying such security criteria in a distributed application is challenging. Techniques such as transport-level security (e.g. TLS) will encrypt requests from the client to an edge application endpoint, but not between each component along the way from the client to the endpoint. Consequently, confidential data of the client may be unnecessarily available to components as the data moves through the pipeline. TLS could be used between each component, but this does not afford the client or a component later in the pipeline any sense of which other nodes have handled the request.

[0017] Another technique which may be used is Public Key Infrastructure (PKI). However, PKI may not scale well: a) with the number (*N*) of target components which need access to the plaintext, since the client would need to perform at least *N* encryptions, and/or b) with the replication of target components. Distributed applications may allow the replication of components so that the application is scaled to correspond with the request load, so a component could be run on multiple (*M*) nodes. To allow scaling, the secret key of each component would need to be replicated across the nodes. This complicates key management, or the client would have to perform at least *M.N* encryptions.

[0018] As well as the client having security criteria which they wish to be met, the service owner who owns and runs the distributed application may also have security criteria, such as specifying that a request must be handled by specific components. For example, a service owner may charge the client per request, or may allow a fixed number of requests in a given timeframe, and each request must be accounted by an accounting component. The service owner would like to ensure that these criteria are adhered to. The components in the distributed application may also support other security concerns such as access control and auditing. However, an attacker may try to make use of a service without going to the other steps of the nodes. For example, an attacker could try and bypass certain steps of the pipeline (e.g. a payment step, an identity verification/password step) and send the request directly to the service node. This is undesirable.

[0019] Another potential solution may be for each node in the pipeline to sign the data which it handles, or provide an authenticated log of events. This would create additional data that must be collected and parsed by each node accepting the request in the pipeline. However, while logs may be useful for *post-facto* forensics, it is more onerous to use it for just-in-time access control decisions. Moreover, each node would have the public keys (or

shared secrets) to verify each authenticated statement. This causes challenges in terms of the "cost" of time and storage, as well as the management of public keys which grows with the depth/size of the pipeline. These challenges are more difficult still for stages in the pipeline that may be replicated for load balancing purposes.

[0020] Examples disclosed herein include systems and methods to enforce a workflow pipeline by using secret sharing cryptography over the workload as it moves through the pipeline. This allows for the execution of the workload to be verified as having gone through the proper pipeline without, for example, requiring complex PKI management typically used by traditional PKI-based solutions. Thus a component may verify the execution of the client request (workload) with a single public key instead of requiring a key for each authorizing component in the pipeline since an earlier step is performed before a later step in the pipeline can be performed.

[0021] Examples disclosed herein may be employed in applications relating to edge compute technologies and micro-service based software architectures. Complex application pipelines centred around artificial intelligence (AI) and machine learning (ML) in health care and digital manufacturing are examples. Examples disclosed herein may allow for data to be protected from attack or interception as it moves through the edge environment into, for example, potentially vulnerable devices and software stacks. Examples disclosed herein may enable real-time checking of the integrity of such application pipelines, which is critical for protecting revenue generating resources like pay-for ML inferencing tools, models, and services, for example. Examples disclosed herein may greatly simplify the complexity of integrating and managing the verification process into the nodes/application components.

[0022] Examples disclosed herein use threshold cryptography. Threshold cryptography may be understood as allowing the distribution of a private key among a cluster of entities (which may be termed "shareholders" because they each have at least one share of the private key). If more than a threshold number of entities of the cluster collaborate, they can reconstruct the original key, sign a message that can be verified with the public key, or decrypt data encrypted with the public key. Examples disclosed herein include shareholders which generate both a partial decryption and a proof of correctness for the partial decryption. It is possible to implement a similar crypto-system e.g. signing and decryption, for example with an RSA based public key mechanism, elliptic curve cryptography, or any other public key mechanism.

[0023] An access structure may be thought of as the set of shareholders that can collaborate to use or reconstruct the original secret (i.e. the client request). A building block

of threshold cryptography is threshold secret sharing, for example Shamir's secret sharing. In some examples, therefore, Shamir secret sharing as a threshold sharing scheme may be used iteratively to enforce arbitrary access structures. Shamir secret sharing uses an algorithm in which a secret is divided into parts and participants are assigned a unique part of the divided secret. To reconstruct the original secret, a minimum number of parts is required, for example, at least a threshold number of parts are used to reconstruct the original secret. In the threshold scheme this number may be less than the total number of parts. Otherwise all participants may be needed to reconstruct the original secret.

[0024] Figure 1 shows a schematic representation of an example system 100. The system 100 comprises a client module 110 and a service provider 120 in communication. The client module 110 is to generate a user request "request message" 106 for the service provider 120 and use the user request 106 to request a service from the service provider 120. The service provider 120 comprises a plurality of application components 130, 140, and the service requested by the user comprises a plurality of steps respectively carried out by the plurality of application components 130, 140. While this example illustrates two application components 130, 140, in other examples there may be more than two, and in some cases there may be many, e.g. tens of application components. The plurality of application components 130, 140 may be one or a combination of, for example, micro-service providers, container-based providers, or virtual hosting entities. The plurality of application components 130, 140 together form the service provider 120. The client module 110 is to generate the user request 106 by encrypting the request message using a public key (PK1) 104. The encrypted request may be labelled "PK1 [request (message)]" because the request includes a message which is encrypted using the public key PK1. The public key PK1 104 corresponds to a private key (SK1) 102 shared between the plurality of application components 130, 140 of the service provider 120. In this example, a first share (SK1a) 102a is with the first application component 130, and a second share (SK1b) 102b is with the second application component 140. The client module 110 is to transmit the user request 106 to the service provider 120.

[0025] In this example, the client module 110 is to generate the user request 106 by encrypting a secret key (RsK) 152 with the request message using the public key (PK1) 104. The encrypted request with the secret key 152 may be labelled "PK1 [request (message, RsK)]" 106 because the user request and the secret key RsK are encrypted using the public key PK1.

[0026] The first application component "App component 1" 130 of the plurality of application components 130, 140 is to receive the transmitted user request 106, and

generate a first partial response “Partial response 1” 132 to the received user request 106, by applying a first share SK1a 102a of the private key SK1 102 to the received user request 106. The first application component 130 is to provide the first partial response 132 for transmission to a further application component “App component 2” 140. The further application component 140 is after the first application component 130, and thus the two application components 130, 140 have a specific order which may be termed a “pipeline”; i.e. the user request 106 must be handled by the first application component 130 before the further application component 140 can perform its task on the user request 106. The ordering of handling of the user request may be enforced by the setup of the pipeline system; that is, application component 1 may be able to communicate with application component 2 (but not with application component 3 directly), but application component 2 is able to communicate with application component 3 directly. Therefore, if the partial response of a particular component is required to achieve the threshold of responses for provision of a response to the user request, then that component has to participate at their position/order in the pipeline, and that component may not be able to participate unless one or more preceding components have participated first. It may be assumed, in some examples, that each component is responsible/trusted to verify which component provided the previous message, and which component is the next one to receive the message (that is, each component may be trusted to verify the sender and recipient of the request along the pipeline). TLS is an example of a scheme which may be used by a component to authenticate the next component in the pipeline. In some examples, the ordering of the components may be controlled by a coordinator component which is configured to control the order of receipt and/or handling of the user request by the components in the pipeline. The controller component may be one of the application components participating in the pipeline in some examples, and the controller component may be separate from the pipeline application components in some examples.

[0027] The further application component 140 of the plurality of application components 130, 140 is to receive the transmitted user request 106 and receive the first partial response 132. Based on receiving the first partial response 132, the further application component 140 is to generate a further partial response 142 to the received user request 106 by applying a further share SK1b 102b of the private key SK1 102 to the received user request 106. The further application component 140 is to combine the received first partial response 132 and the generated further partial response 142 to obtain a combined response 144.

[0028] Based on the first 132 and further 142 partial responses reaching a threshold number of responses, the further application component 140 is to provide the combined response 144 in response to the user request. In some examples such as those using a threshold cryptography system, the threshold may not require all components to have contributed but a threshold number of them. In some examples, the threshold may be all components contributing.

[0029] Figure 1 may be considered to show a system 100, comprising a service provider 120, the service provider comprising a plurality of application components 130, 140; wherein a first application component 130 of the plurality of application components is to receive a transmitted user request 106, the user request generated by a client module 110 encrypting a request message using a public key 104, wherein the public key corresponds to a private key 102 shared between a plurality of application components 130, 140 of the service provider 120; generate a first partial response 132 to the received user request 106 by applying a first share of the private key 102a to the received user request 106; and provide the first partial response 132 for transmission to a further application component 140; and a further application component 140 of the plurality of application components is to: receive the transmitted user request 106 and the first partial response 132; based on receiving the first partial response 132, generate a further partial response 142 to the received user request 106 by applying a further share of the private key 102b to the received user request 106; combine the received first partial response 132 and the generated further partial response 142 to obtain a combined response 144; and based on the first and further partial responses reaching a threshold number of responses, provide the combined response 144 in response to the user request.

[0030] The application component 140 may in some examples be considered to be a combiner component 140. Such a combiner component 140 may be considered to receive a user request 106 comprising a request message encrypted using a public key 104, wherein the public key corresponds to a private key 102 shared between a plurality of application components 130, 140 of a service provider 120, the plurality of application components 130, 140 comprising at least one preceding application component 130 and the combiner component 140; receive, at the combiner component 140, at least one preceding partial response 132 to the user request generated by at least one preceding application component 130 applying at least a share of the private key 102a to the user request; apply, by the combiner component 140, a further share of the private key 102b to the user request 106 to generate a further partial response 142; verify that the at least one preceding partial response 132 and the further partial response 142 at least meet a

predetermined threshold number of responses; and based on the verification, combine the at least one preceding partial response 132 and the further partial response 142 to obtain a combined response 144.

[0031] In some examples, the combiner component 140 may not itself apply a share of the private key to generate a combined response and may instead act to combine partial responses 132, 142 from other preceding application components. Such a component may be to receive a plurality of preceding partial responses 132, 142 to a user request 106, the plurality of preceding partial responses 132, 142 generated by a respective plurality of preceding application components 130, 140 applying at least a respective share of the private key 102a, 102b to the user request. The combiner component may then verify that the at least one preceding partial response 132 and the further partial response 142 at least meet a predetermined threshold number of responses; and based on the verification, combine the at least one preceding partial response 132 and the further partial response 142 to obtain a combined response 144.

[0032] In some examples such as this one in which a secret key RsK 152 is transmitted to the service provider 120, the service provider 120 is to encrypt 146 the combined response 144 with the received secret key RsK 152 before transmitting the secret-key encrypted combined response, "RsK [Combined response]" 148 to the client module 110. The further application component 140 of the service provider 120 may encrypt the combined response 144 using the secret key (RsK) provided in the user request 106, before transmitting 148 the encrypted combined response 146 to the client module 110, in some examples. Once received, the client module 110 may decrypt the received encrypted combined response 148 using the secret key 152 (RsK).

[0033] In other examples, the encrypted combined response 148 may be received at an entity separate from the client module 110 which is in possession of the secret key (RsK) 152, and the separate entity may use the secret key (RsK) 152 to decrypt the encrypted combined response 148 before passing the decrypted combined response to the client module 110. There may be a secure channel between such a separate entity and the client module 110 in such examples.

[0034] In some examples, as the user request 106 is provided to the application components 130, 140 along the pipeline, the first 132 and further 142 partial responses provided by the first 130 and further 140 application components respectively are decrypted portions of the combined response 144. In such examples, the combined response 144 is a message that would be produced by decrypting the user request with the private key.

[0035] In the example system 100 of Figure 1, the first application component 130 and the further application component 140 form a pipeline defining an order of generation of the first 132 and further 142 partial responses. That is, the further application component 140 cannot do anything with the received request 106 until it receives the partial response 132 from the preceding first application component 130. In some examples, it is the service provider 120 which is to define the pipeline using the first 130 and further 140 application components according to the user request 106. In some examples, the public key (PK1) 104 and corresponding private key (SK1) 102 are generated, and then the private key may be distributed amongst the components to correspond with the pipeline.

[0036] In the example system 100 of Figure 1, each application component 130, 140 has one respective share of the private key (SK1) 102. The first application component 130 has a first share (SK1a) 102a of the private key 102, and the further application component 140 has a further share (SK1b) 102b of the private key 102. In some examples, one or more of the first application component 130 and the further application component 140 may each have a plurality of shares of the private key 102. For example, a first application component may have one share SK1a of the private key 102, a second application component may have two shares SK1b, SK1c of the private key 102, and a third application component may have three shares SK1d, SK1e, SK1f of the private key 102. In such an example, the first application component 130 and/or the further application component(s) 140 having the plurality of shares of the private key 102 may generate a plurality of partial responses corresponding to the plurality of shares of the private key 102.

[0037] As a further example of a client-service owner system, a client 110 may send requests to a distributed application 130, 140 and may receive a response back 148. A service owner 120 may run the distributed application 130, 140. The distributed application 130, 140 may comprise four application components: *C1, C2, C3, C4*. For example, *C1* 130 may be a service controller and may dispatch requests to appropriate nodes. *C2*, *C3* and *C4* 140 may be service components that are to compute output from input.

[0038] For a first request *Rq1* 106, in this example, *C2* 140 needs access to the plaintext (non-encrypted) request, and *C1* 130 has to participate (but *C1* 130 doesn't need plaintext access). The first response *Rs1* 148 is returned to the client 110. For a second request *Rq2* 106 in this example, *C3* and *C4* 140 need access to the plaintext, and *C1* 130 is to participate. The second response *Rs2* 148 is returned to the client 110.

[0039] It is desirable for certain goals, or criteria, to be met by the client 110-service owner 120 system. One goal **G1** is that request data is revealed to necessary components

(i.e. components which have to use the request data) and not to unnecessary components. This is to reduce the provision of client data if it is not necessary, to aid client data privacy. A further goal **G2** is that response data is revealed to necessary components (i.e. components which have to use the response data) and not to unnecessary components, and revealed to the client 110. Again, this is to reduce the provision of service provider data if it is not necessary, to aid service provision security. A further goal **G3** is that a response is sent to the client 110 if it has been handled by the appropriate components, according to the client 110 and the service owner 120 criteria. This goal is to ensure the request is handled correctly by passing through the nodes/application components which need to perform some processing to generate the Response to the request, and to ensure the client 110 criteria and the service owner 120 criteria are met. A further goal **G4** is for the client 110 to perform one, but not more, asymmetric encryption operations, to reduce the processing overhead at the client 110 and to simplify key management at the client 110 by requiring one key rather than plural keys for plural processing steps.

[0040] To accomplish these goals **G1**, **G2**, **G3** and **G4,** examples disclosed herein such as the example of Figure 1 may perform the following:

[0041] To achieve goals **G1** and **G3**, the private (secret) key SK1 102 corresponding to the public key PK1 104 is shared among the application components 130, 140, such that data encrypted with the public key PK1 104 can be decrypted by the combination, or threshold number, of application components having the shares of the private key SK1 102, but not without. In some examples the public key PK1 104 may be certified to comply with goals **G1** and **G3** i.e. the public key PK1 104 is certified so that the necessary application components 130, 140 (but not more) can receive the request 106, and certified so that the response 148 can be sent to the client 110 after being certified as being handled by the appropriate application components 130, 140, but not before i.e. certified as having passed through the pipeline components. Before sending the request 106, the client 110 may verify that the public key PK1 104 complies with goals **G1** and **G3**. Certification is discussed below in relation to Figure 5.

[0042] To achieve goal **G2**, when sending the request 106, the client 110 may provide a (e.g. symmetric) key RsK 152 that is used to encrypt the response 148, as shown in Figure 1. The key RsK 152 may be encrypted alongside the request data 106 as illustrated. In this way the client 110, but no one else, can obtain the response by decrypting the combined response 148 using the secret key RsK 152.

[0043] To achieve goal **G4**, when sending the request 106, the client 110 may perform one asymmetric encryption (as compared with a plurality of separate encryptions for each application component 130, 140) with the request-specific public key PK1 104.

[0044] A pipeline *P1* may be set up comprising a plurality of application components 130, 140 in a particular order which are to process the user request. For example, the service provider / service owner 120 may set up a new pipeline *P1* which takes in a request 1 *Rq1* and provides a response → *Rs1*. A key pair (private key *SK1,* public key *PK1*) may be generated for pipeline *P1*. An appropriate access structure *AccStruct1* may then be generated to comply with goals **G3** and **G4** described above. The private key *SK1* may then be shared among the application components 130, 140 according to *AccStruct1*. Then, a certificate *cert1* may be generated by a Certificate Authority (*CA*). The certificate cryptographically binds the public key *PK1* to the pipeline *P1*. Figure 5 discussed below illustrates certification.

[0045] Access structures supporting component replication on different nodes may be generated. An access structure may be described as a Boolean expression indicating which shareholders are to collaborate to decrypt the ciphertext. For example, the access structure "Component1 AND Component2" uses both Component1 and Component2 to emit a partial decryption to get a plaintext. If the access structure is "Component1 OR Component2", either of the Components can get the plaintext with their share. The access structure may be arbitrarily complex in some examples, for example "Component1 AND (Component2 OR Component3)" may be an "ArbitraryAccessStructure".

[0046] In examples in which one component ("End Component") needs access to the plaintext, the access structure "EndComponent AND (ArbitraryAccessStructure)" may be used. In examples in which more than one Component needs access to the plaintext request (for example if two Components, EndComponent1 and EndComponent2, needs access to the plaintext data, the access structure "EndComponent1 AND EncComponent2 AND (ArbitraryAccessStructure)" may be used. Concerning component replication, in a pipeline, it may be desired that a component running on one node, node1 ("Component1OnNode1") may be run on a different node, node2. For example, a load balancer might send a request to any authorized web server (e.g. ("Component1OnNode1" or ("Component1OnNode2"). To satisfy that policy, an access structure may be used which is ""Component1OnNode1 OR ("Component1OnNode2"). Further, such access structures can be combined to implement arbitrary access structures.

[0047] As mentioned above, the combined response is obtained once first and further partial responses generated by the first and further application components 130, 140 reach a threshold number of responses. The combined response 144 may then be provided to the client in response to the user request. Such threshold cryptography, by which sufficient partial responses need to be generated in order for a meaningful response to be obtained, may be employed to enforce arbitrary access structures. A Shamir secret sharing protocol may be used which allows a key to be shared in n shares, such that a threshold t, $t \leq n$ of shareholders (application components) can generate the private key *SK1*. An arbitrary access structure may be enforced by using Shamir secret sharing iteratively. One or more application components may hold more than one share and may have to generate more than one partial decryption. The component(s) combining the partial decryption must combine them accordingly.

[0048] In some examples, an application component / node may be updated. Updated may mean that an application component / mode may be added, removed or replaced in a pipeline. If an application component / node is updated then the access structure may be updated to take account of the update. In such examples, shares may be revoked (e.g. for a removed node) or new shares may be generated (e.g. for a new node), wherein the shares corresponding to the same private - public key pair *(SK1, PK1)*.

[0049] A request protocol may be considered to be where a client 110 sends an encrypted request 106 to the pipeline of application components 130, 140. The request 106 is processed by the pipeline and the plaintext of the request is made available to nodes which use it, but not made available to nodes which do not use the plaintext.

[0050] In another example of transmitting a request and receiving a response according to a request protocol, a client 110 may wish to send a request to a first pipeline P1. The client 110 may retrieve a certificate “cert1” from a certificate store. The client 110 may then verify the certificate cert1 by verifying that the certificate corresponds to the pipeline P1, and by verifying the certificate signature.

[0051] The client 110 may then generate the secret key RsK 152. The client 110 may then encrypt the plaintext request as (data, RsK) using the public key PK1 104 to obtain a “ciphertext”. The client may then send the request Rq1 106 as e.g. (headers, ciphertext) to the first application component C1 130. The first application component C1 130 may then emit a first partial decryption “dec1” 132 as a partial response for the ciphertext. The first application component C1 130 may then send the request Rq1 106 and the first partial decryption “dec1” 132 to a further application component C2 140. The further application

component C2 140 may then emit a further partial decrypt “dec2” as a partial decryption 142 for the ciphertext.

[0052] The further application component C2 140 may then combine the partial responses “dec1” 132 and “dec2” 134 to get the plaintext as the combined response 144. The further application component C2 140 may then handle the request 106 Rq1 and generate data for the response Rs1 144. The further application component C2 1401 may then encrypt 146 the response data 144 with the secret key RsK 152. The further application component C2 may then send the response Rs1 148 = (headers, ciphertext) to the client module 110. In some examples this may be via the first application component C1 130. The client may then decrypt the ciphertext from the response 148 Rs1 using the secret key RsK 152.

[0053] In some examples there may be more than one target node; that is, for example in a second pipeline *P2* comprising at least three application components C1, C3 and C4, both *C3* and *C4* may need access to the plaintext. In such examples, additional steps may take place to satisfy goals **G3** and **G4** mentioned above. Namely, one further application component *C3* may need to send a partial decryption “dec3” to a second further application component *C4,* and the second further application component *C4* may need to send a further partial decryption “dec4” to the first further application component *C3*.

[0054] However, in an example in which the partial decryptions “dec3” and “dec4” are sent on a network, an eavesdropper may retrieve the decrypted portions “dec1” to “dec4” and thus retrieve the plaintext. So the first further partial decryption “dec3” needs to be encrypted to pass to the second further application component C4 and the second further partial decryption dec4 needs to be encrypted to pass to the first further application component C3. This may be achieved, for example, by using transport-level security (TLS) between the application components C3 and C4.

[0055] The combined response 148 in Figure 1 may be transmitted to the client module 110 in different ways. Figures 2a and 2b illustrate schematic representations of providing a combined response 148.

[0056] In the example of Figure 2a, the service provider 120 outputs a combined response 108 reply to the client request, and provides this directly to the client module 110. In this case, the “combined response” is an end result sent in response to the user’s initial request once the plaintext has been obtained.

[0057] In the example of Figure 2b, the service provider 120 outputs a combined response 108 which is a plaintext request, and provides this to a reply provider 210. The reply provider 210 takes the plaintext request 108 as input, determines a reply to the

plaintext request, and provides as output the reply 108b to the client module 110. In this case, the "combined response" is the plaintext request from the client which has been decrypted and can be used to obtain a reply to send to the client module 110. That is, the combined response 108 is a plaintext request corresponding to the user request (i.e. it is unencrypted client request), and the service provider 120 is to use the plaintext request 108 to provide a user request reply 108b to the client module 110. In some examples the reply provider 210 may be part of the service provider 120.

[0058] Figures 3a and 3b show example schematic representations of distributing a private key. Figure 3a illustrates a centralised protocol, whereby a setup service 310 generates the private key SK1 102, generates the private key shares 172, 174, 176 according to as access structure "AccStruct1", and sends the private key shares 172, 174, 176 to corresponding nodes 130, 140, 150. The setup service 310 may then delete the private key SK1 102. In other words, in some examples, the systems disclosed herein may comprise a setup module 310 to: generate the public key 104 and the private key 102 paired with the public key 104; determine shares 172, 174, 176 of the generated private key 102; distribute the private key shares 172, 174, 176 to the first and further application components 130, 140, 150; and delete the private key 102 from the setup module (the public key 104 may be deleted from the setup module 310 as well).

[0059] Figure 3d illustrates a distributed protocol, whereby the setup service 310 coordinates a distributed protocol between application components e.g. C1 to C4 such that shares 172, 174, 176 of the private key are generated according to the access structure "AccStruct1". The private key SK1 in this example is never generated anywhere and cannot be stolen from the setup service 310. In other words the systems disclosed herein may comprise a setup module 310 to: generate a distribution protocol of shares 172, 174, 176 of the private key, the private key paired with the public key 104; and cause generation and distribution of the private key shares 172, 174, 176 to the first and further application components 130, 140, 150 according to the distribution protocol without generating the complete private key.

[0060] Figure 4 shows a schematic representation of a service provider 120 comprising a verification application component 410 according to an example. Similarities with the system 100 illustrated in Figure 1 will not be explained in detail again. In this system, the plurality of application components 130, 140, 410 comprises a verification application component 410 which is to transmit the user request 106 to the first 130 and further 140 application components.

[0061] Figure 5 shows a schematic representation of certificate verification. In this example, the client module 110 (e.g. the client module 110 of Figure 1) is to: retrieve a certificate 512 from a certificate store 510, wherein the certificate 510 cryptographically binds the public key 104 with the plurality of application components 130, 140 of the service provider 120; and verify that the certificate 512 corresponds to the service provider 120.

[0062] Figures 6 and 7 show example methods according to examples of this disclosure. Figure 6 shows a computer-implemented method 600, for example performed at a service provider 120, wherein the method 600 comprises: receiving a user request 602, at a service provider from a client module, the user request comprising a request message encrypted using a public key, wherein the public key corresponds to a private key shared between a plurality of application components of the service provider. The method 600 also comprises providing the user request to a first application component of the service provider 604; applying, by the first application component, a first share of the private key to the user request to generate a first partial response 604; providing the first partial response and the user request to at least one further application component of the service provider 606; applying, by the at least one further application component, at least one further share of the private key to the user request to generate at least one further partial response 608; combining the first partial response and the at least one further partial response to obtain a combined response 610; and based on the first and at least one further partial responses reaching a threshold number of responses, providing the combined response 614 in response to the user request. In other examples, an element or elements from the method of Figure 6 described above may be omitted.

[0063] Figure 7 illustrates a computer-implemented method 700, which may be performed in combination with the method of Figure 6, comprising: splitting the private key into a plurality of shares 702; allocating the shares between the first and further application components 704; and providing the shares to the first and further application components according to the allocation 706. In some examples, the method may comprise: defining a pipeline using the first and further application components 130, 140, the pipeline specifying an order of generation of the first and further partial responses by the first and further application components according to the defined pipeline.

[0064] Figure 8 shows an example apparatus 800 comprising a processor 802 and memory 804 according to examples of this disclosure. The apparatus 800 may be to perform any method disclosed herein. Figure 8 illustrates the apparatus 800 as part of a service provider 120, but similar apparatus 800 comprising a processor 802 and memory

804 may be or be comprised in, for example, a client device / client module 110, a reply provider 210, a setup module 310, an application component 130, 140, and/or a verification application component 410.

[0065] The memory 804 is an example of a computer readable storage medium, for example a non-transitory storage medium, having executable instructions stored thereon. When executed by a processor 802, the instructions may cause the processor 802 to: receive, from a client module, a user request 106 comprising a request message encrypted using a public key, wherein the public key corresponds to a private key shared between a plurality of application components of a service provider; provide the user request 106 to a first application component 130 of the service provider; apply, by the first application component 130, a first share of the private key to the user request to generate a first partial response; provide the first partial response and the user request to at least one further application component of the service provider; apply, by the at least one further application component, at least one further share of the private key to the user request to generate at least one further partial response; combine the first partial response and the at least one further partial response to obtain a combined response 148; and based on the first and at least one further partial responses reaching a threshold number of responses, transmit the combined response 148 in response to the user request. In other examples, the non-transitory computer-readable storage medium may comprise program code to perform any of the methods discussed herein.

[0066] All of the features disclosed in this specification (including any accompanying claims, abstract, and drawings) may be combined in any combination, except combinations where some of such features are mutually exclusive. Each feature disclosed in this specification, including any accompanying claims, abstract, and drawings), may be replaced by alternative features serving the same, equivalent, or similar purpose, unless expressly stated otherwise. Thus, unless expressly stated otherwise, each feature disclosed is one example of a generic series of equivalent or similar features.

[0067] The present teachings are not restricted to the details of any foregoing examples. Any novel combination of the features disclosed in this specification (including any accompanying claims, abstract, and drawings) may be envisaged. The claims should not be construed to cover merely the foregoing examples, but also any variants which fall within the scope of the claims.

[0068] The following numbered paragraphs also form a part of this disclosure:

[0069] 1. A system, comprising: a client module to: generate a user request for a service provider, the service provider comprising a plurality of application components, by

encrypting a request message using a public key, wherein the public key corresponds to a private key shared between a plurality of application components of the service provider; and transmit the user request to the service provider; a first application component of the plurality of application components to: receive the transmitted user request; generate a first partial response to the received user request by applying a first share of the private key to the received user request; and provide the first partial response for transmission to a further application component; and a further application component of the plurality of application components to: receive the transmitted user request and the first partial response; based on receiving the first partial response, generate a further partial response to the received user request by applying a further share of the private key to the received user request; combine the received first partial response and the generated further partial response to obtain a combined response; and based on the first and further partial responses reaching a threshold number of responses, provide the combined response in response to the user request.

[0070] 2. The system of paragraph 1, wherein the combined response is a plaintext request corresponding to the user request, and the service provider is to use the plaintext request to provide a user request reply to the client module.

[0071] 3. The system of paragraph 1 or paragraph 2, wherein: the client module is to: generate the user request by encrypting a secret key with the request message using the public key; and the further application component is to: encrypt the combined response using the secret key provided in the user request, before transmitting the encrypted combined response to the client module.

[0072] 4. The system of paragraph 3, wherein the client module is to decrypt the received encrypted combined response using the secret key.

[0073] 5. The system of any of paragraphs 1 to 4, wherein: the first application component and the further application component form a pipeline defining an order of generation of the first and further partial responses.

[0074] 6. The system of any of paragraphs 1 to 5, wherein: a setup module is to: generate the public key and the private key paired with the public key; determine shares of the generated private key; distribute the private key shares to the first and further application components; and delete the private key from the setup module.

[0075] 7. The system of any of paragraphs 1 to 6, wherein: a setup module is to: generate a distribution protocol of shares of the private key, the private key paired with the public key; and cause generation and distribution of the private key shares to the first and

further application components according to the distribution protocol without generating the complete private key.

[0076] 8. The system of any of paragraphs 1 to 7, wherein the plurality of application components comprises a verification application component to transmit the user request to the first and further application components.

[0077] 9. The system of any of paragraphs 1 to 8, wherein the plurality of application components are one or more of: micro-service providers, container-based providers; and virtual hosting entities; the plurality of application components together forming a service provider.

[0078] 10. The system of any of paragraphs 1 to 9, wherein one or more of the first application component and the further application component each has a plurality of shares of the private key; and the one or more of the first application component and the further application component having the plurality of shares of the private key is to generate a plurality of partial responses corresponding to the plurality of shares of the private key.

[0079] 11. The system of any of paragraphs 1 to 11, wherein: the client module is to: retrieve a certificate from a certificate store, wherein the certificate cryptographically binds the public key with the plurality of application components of the service provider; and verify that the certificate corresponds to the service provider.

[0080] 12. A computer-implemented method comprising: receiving a user request, at a service provider from a client module, the user request comprising a request message encrypted using a public key, wherein the public key corresponds to a private key shared between a plurality of application components of the service provider; providing the user request to a first application component of the service provider; applying, by the first application component, a first share of the private key to the user request to generate a first partial response; providing the first partial response and the user request to at least one further application component of the service provider; applying, by the at least one further application component, at least one further share of the private key to the user request to generate at least one further partial response; combining the first partial response and the at least one further partial response to obtain a combined response; and based on the first and at least one further partial responses reaching a threshold number of responses, providing the combined response in response to the user request.

[0081] 13. The computer-implemented method of paragraph 12, comprising: splitting the private key into a plurality of shares; allocating the shares between the first and further

application components; and providing the shares to the first and further application components according to the allocation.

[0082] 14. The computer-implemented method of paragraph 12 or paragraph 13, comprising: defining a pipeline using the first and further application components, the pipeline specifying an order of generation of the first and further partial responses by the first and further application components according to the defined pipeline.

[0083] 15. A non-transitory computer readable storage medium having executable instructions stored thereon, which, when executed by a processor, cause the processor to: receive, from a client module, a user request comprising a request message encrypted using a public key, wherein the public key corresponds to a private key shared between a plurality of application components of a service provider; provide the user request to a first application component of the service provider; apply, by the first application component, a first share of the private key to the user request to generate a first partial response; provide the first partial response and the user request to at least one further application component of the service provider; apply, by the at least one further application component, at least one further share of the private key to the user request to generate at least one further partial response; combine the first partial response and the at least one further partial response to obtain a combined response; and based on the first and at least one further partial responses reaching a threshold number of responses, transmit the combined response in response to the user request.

CLAIMS

1. A system, comprising a service provider, the service provider comprising a plurality of application components; wherein

a first application component of the plurality of application components is to:

receive a transmitted user request, the user request generated by a client module encrypting a request message using a public key, wherein the public key corresponds to a private key shared between a plurality of application components of the service provider;

generate a first partial response to the received user request by applying a first share of the private key to the received user request; and

provide the first partial response for transmission to a further application component; and

a further application component of the plurality of application components is to:

receive the transmitted user request and the first partial response;

based on receiving the first partial response, generate a further partial response to the received user request by applying a further share of the private key to the received user request;

combine the received first partial response and the generated further partial response to obtain a combined response; and

based on the first and further partial responses reaching a threshold number of responses, provide the combined response in response to the user request.

2. The system of claim 1, wherein the combined response is a plaintext request corresponding to the user request, and the service provider is to use the plaintext request to provide a user request reply to the client module.

3. The system of claim 1 comprising the client module, wherein:

the client module is to:

generate the user request by encrypting a secret key with the request message using the public key; and

the further application component is to:

encrypt the combined response using the secret key provided in the user request, before transmitting the encrypted combined response to the client module.

4. The system of claim 3, wherein the client module is to decrypt the received encrypted combined response using the secret key.

5. The system of claim 1, wherein:

the first application component and the further application component form a pipeline defining an order of generation of the first and further partial responses.

6. The system of claim 1, comprising a setup module to:

generate the public key and the private key paired with the public key;

determine shares of the generated private key;

distribute the private key shares to the first and further application components; and

delete the private key from the setup module.

7. The system of claim 1, comprising a setup module to:

generate a distribution protocol of shares of the private key, the private key paired with the public key; and

cause generation and distribution of the private key shares to the first and further application components according to the distribution protocol without generating the complete private key.

8. The system of claim 1, wherein the plurality of application components comprises a verification application component to transmit the user request to the first and further application components.

9. The system of claim 1, wherein the plurality of application components are one or more of:

micro-service providers,

container-based providers; and

virtual hosting entities;

the plurality of application components together forming a service provider.

10. The system of claim 1, wherein one or more of the first application component and the further application component each has a plurality of shares of the private key; and the one or more of the first application component and the further application component having the plurality of shares of the private key is to generate a plurality of partial responses corresponding to the plurality of shares of the private key.

11. The system of claim 1 comprising the client module, wherein the client module is to:

retrieve a certificate from a certificate store, wherein the certificate cryptographically binds the public key with the plurality of application components of the service provider; and

verify that the certificate corresponds to the service provider.

12. A computer-implemented method comprising:

receiving a user request at a service provider, the user request comprising a request message encrypted using a public key, wherein the public key corresponds to a private key shared between a plurality of application components of the service provider;

providing the user request to a first application component of the service provider;

applying, by the first application component, a first share of the private key to the user request to generate a first partial response;

providing the first partial response and the user request to at least one further application component of the service provider;

applying, by the at least one further application component, at least one further share of the private key to the user request to generate at least one further partial response;

combining the first partial response and the at least one further partial response to obtain a combined response; and

based on the first and at least one further partial responses reaching a threshold number of responses, providing the combined response in response to the user request.

13. The computer-implemented method of claim 12, comprising:

splitting the private key into a plurality of shares;

allocating the shares between the first and further application components; and

providing the shares to the first and further application components according to the allocation.

14. The computer-implemented method of claim 12, comprising:

defining a pipeline using the first and further application components, the pipeline specifying an order of generation of the first and further partial responses by the first and further application components according to the defined pipeline.

15. A non-transitory computer readable storage medium having executable instructions stored thereon, which, when executed by a processor, cause the processor to:

receive, at a combiner component, a user request comprising a request message encrypted using a public key, wherein the public key corresponds to a private key shared between a plurality of application components of a service provider, the plurality of application components comprising at least one preceding application component and the combiner component;

receive, at the combiner component, at least one preceding partial response to the user request, the at least one preceding partial response generated by at least one preceding application component applying at least a share of the private key to the user request;

apply, by the combiner component, a further share of the private key to the user request to generate a further partial response;

verify that the at least one preceding partial response and the further partial response at least meet a predetermined threshold number of responses; and

based on the verification, combine the at least one preceding partial response and the further partial response to obtain a combined response.

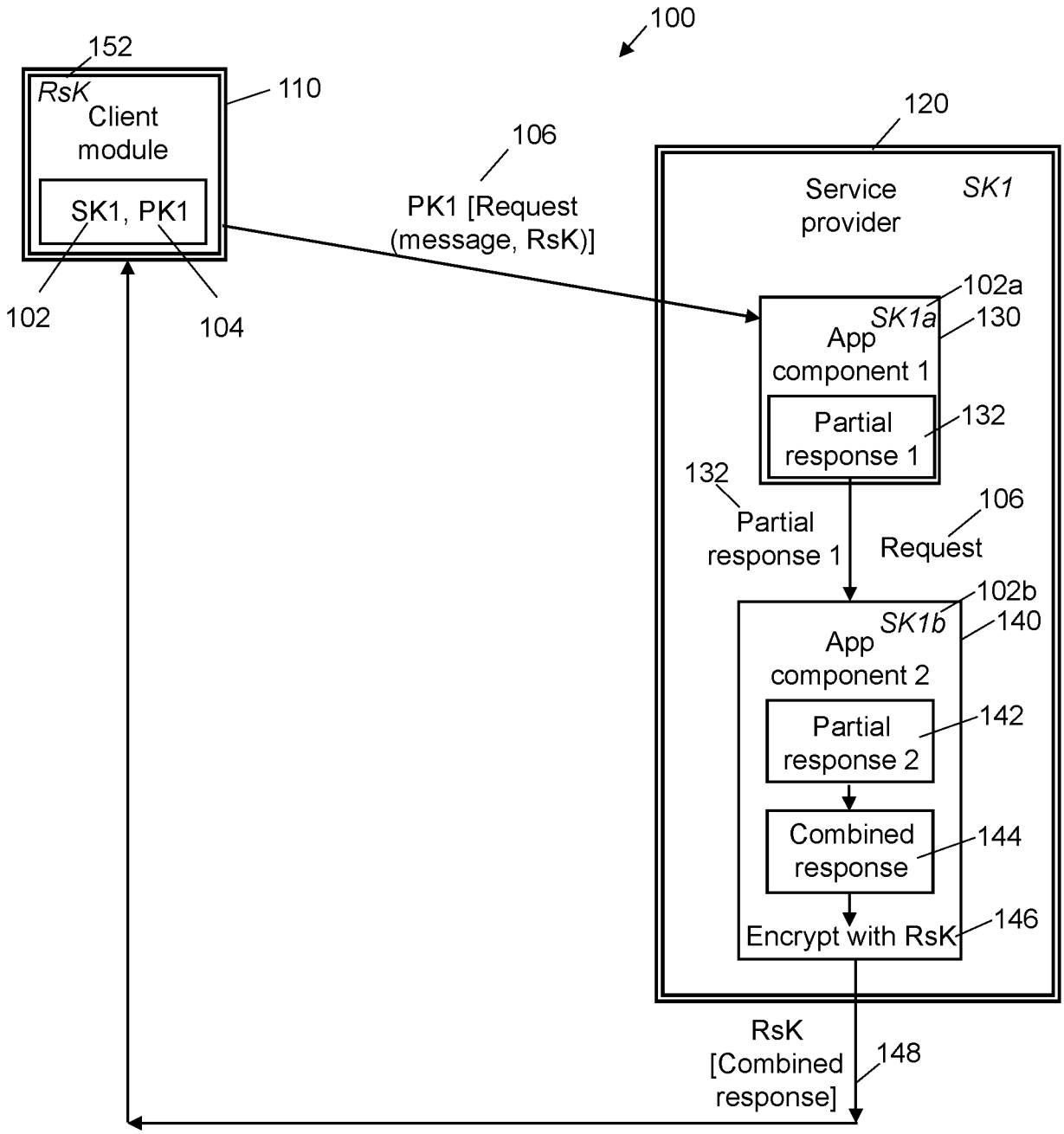
100
152
RsK
Client module
110
SK1, PK1
102
104
106
PK1 [Request (message, RsK)]
120
Service provider
SK1
102a
SK1a
App component 1
130
Partial response 1
132
132
Partial response 1
106
Request
102b
SK1b
App component 2
140
Partial response 2
142
Combined response
144
Encrypt with RsK
146
RsK [Combined response]
148

Figure 1

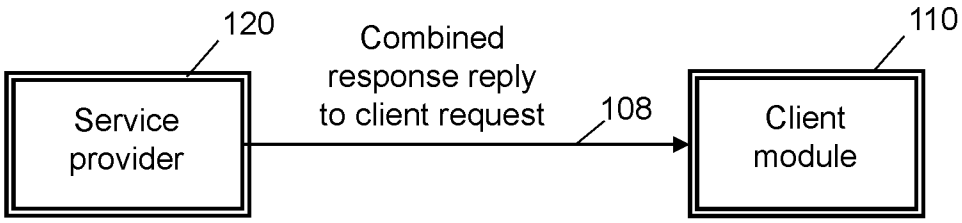

Figure 2a

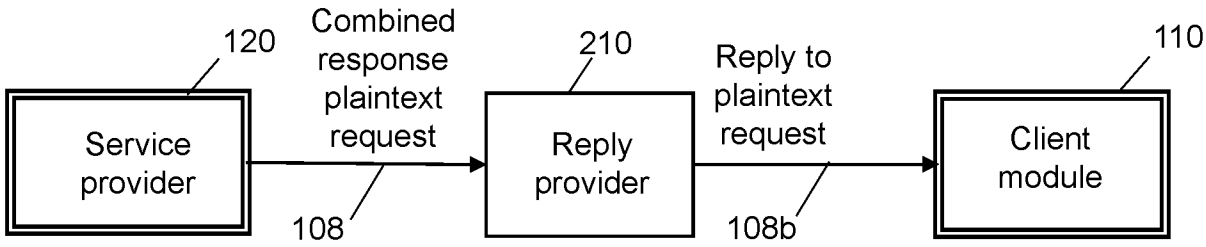

Figure 2b

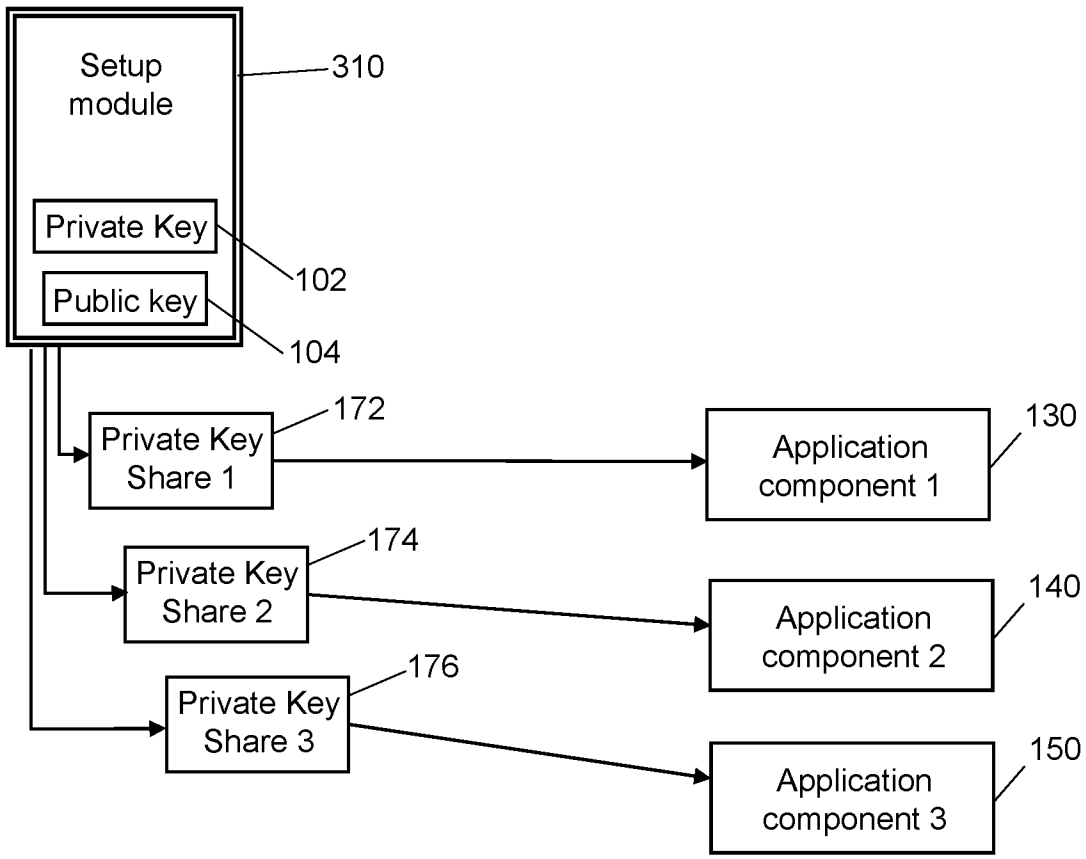

Figure 3a

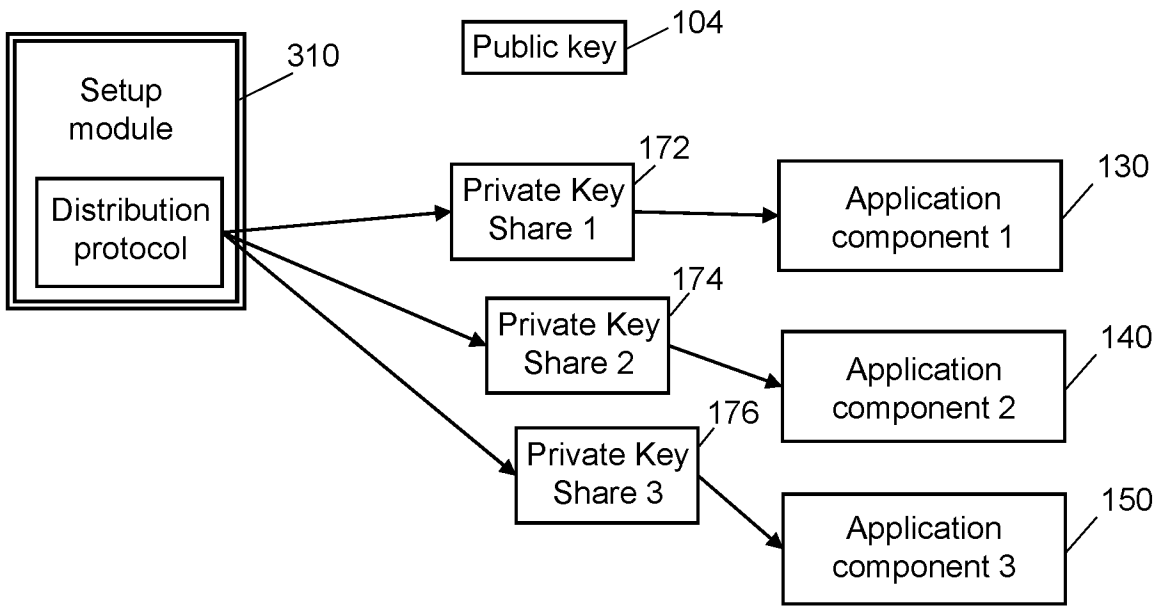

Figure 3b

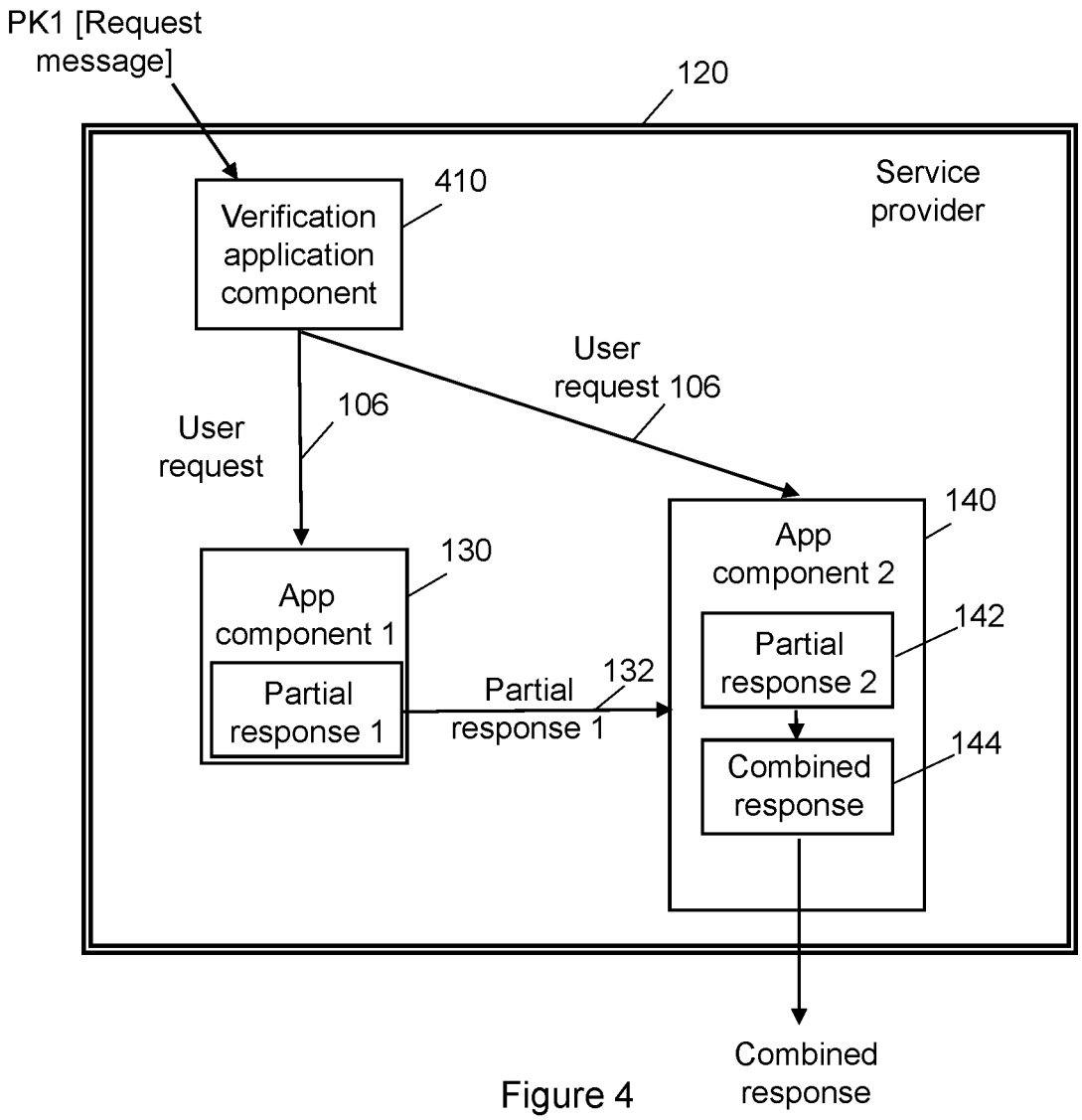

Figure 4

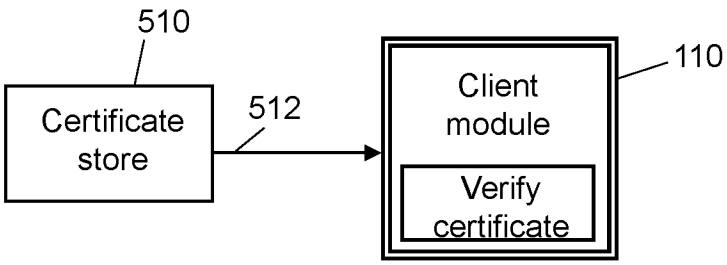

Figure 5

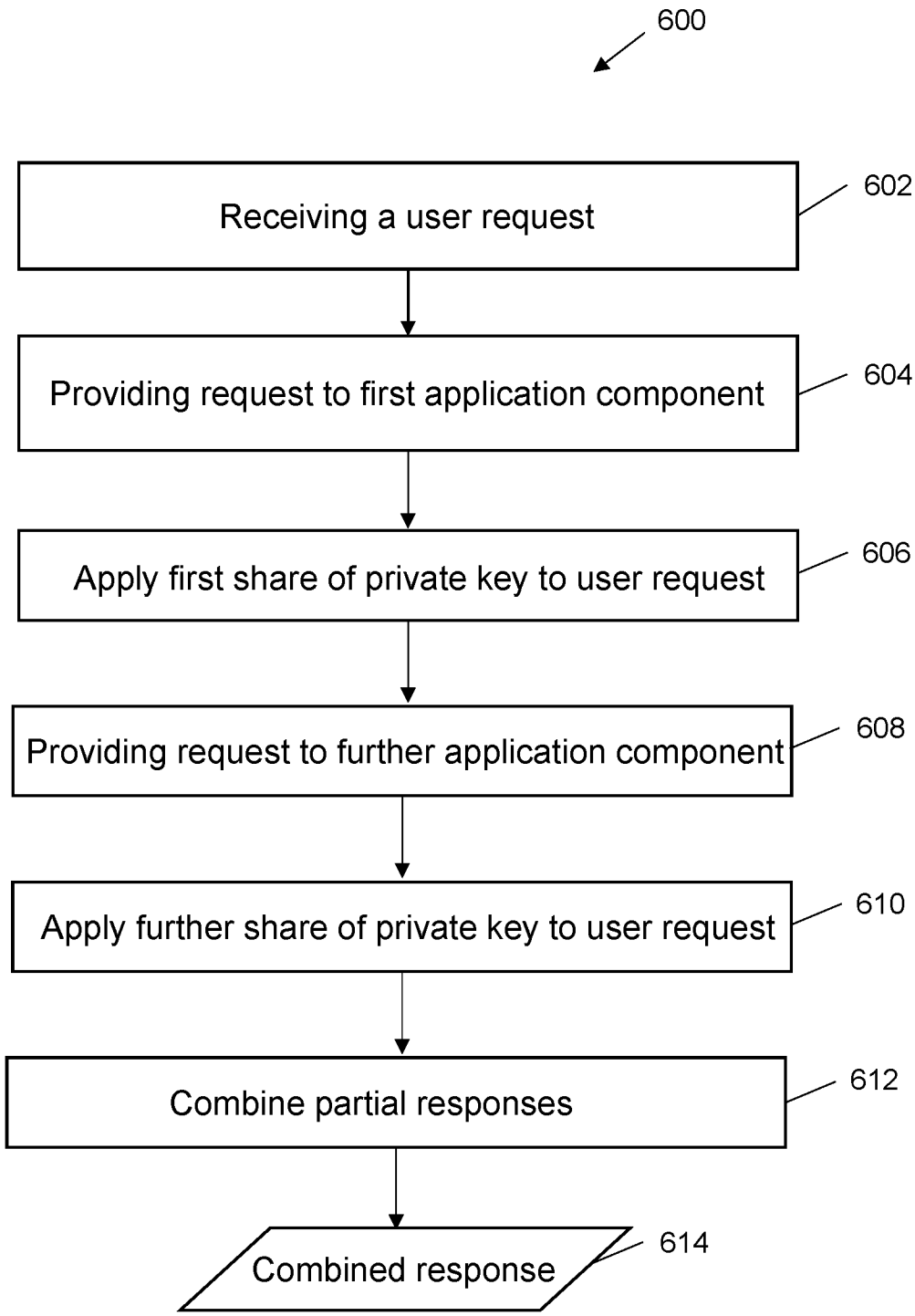
600
Receiving a user request
602
Providing request to first application component
604
Apply first share of private key to user request
606
Providing request to further application component
608
Apply further share of private key to user request
610
Combine partial responses
612
Combined response
614

Figure 6

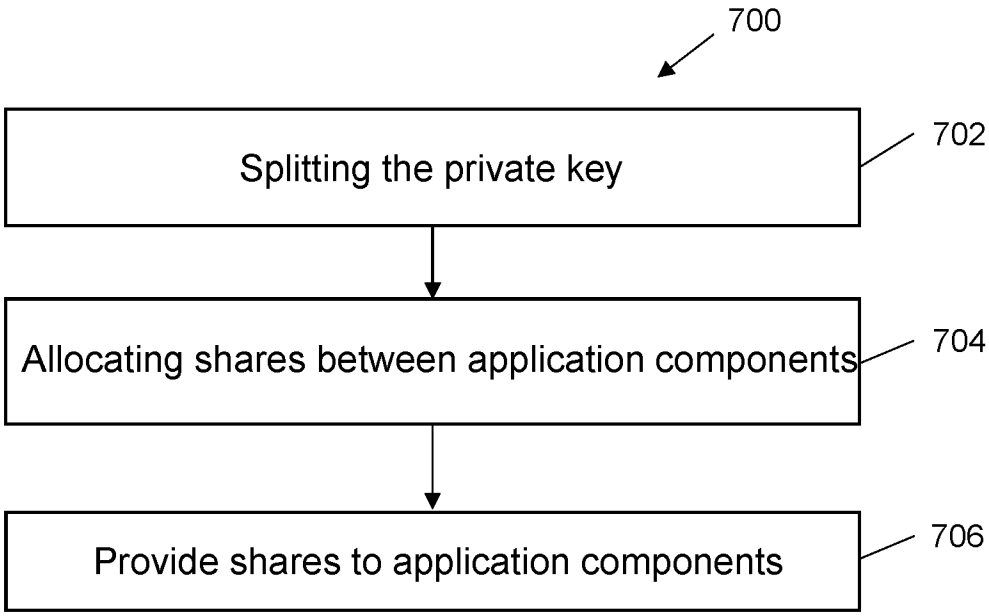

Figure 7

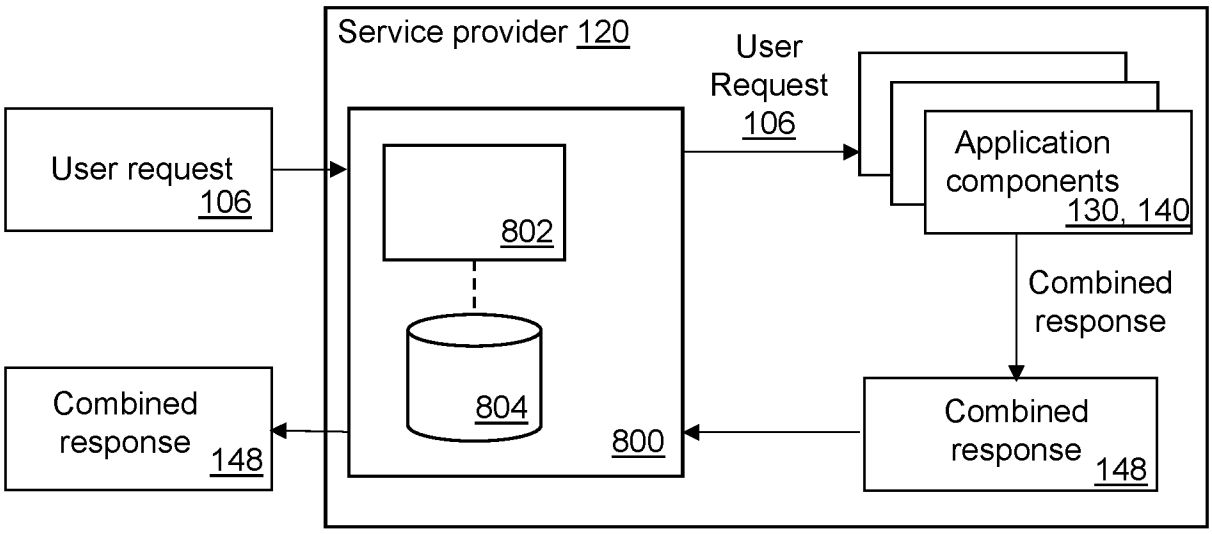

Figure 8

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US 2020/048385

A. CLASSIFICATION OF SUBJECT MATTER

G06F 21/62 *(2013.01)*
G06F 9/54 *(2006.01)*

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

G06F 21/00-21/62, 9/00-9/54, 15/00-15/173, 17/00-17/40, H04L 29/00-29/08, 12/00-12/24p

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

PatSearch (RUPTO Internal), USPTO, PAJ, Espacenet, Information Retrieval System of FIPS

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	US 2012/0324125 A1 (SALESFORCE.COM, INC.) 20.12.2012	1-15
A	WO 2017/187207 A1 (PRIVITAR LIMITED) 02.11.2017	1-15
A	US 9426027 B1 (AMAZON TECHNOLOGIES, INC.) 23.08.2016	1-15

☐ Further documents are listed in the continuation of Box C.

☐ See patent family annex.

* Special categories of cited documents:

"A" document defining the general state of the art which is not considered to be of particular relevance

"D" document cited by the applicant in the international application

"E" earlier document but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
26 April 2021 (26.04.2021)	13 May 2021 (13.05.2021)
Name and mailing address of the ISA/RU: Federal Institute of Industrial Property, Berezhkovskaya nab., 30-1, Moscow, G-59, GSP-3, Russia, 125993 Facsimile No: (8-495) 531-63-18, (8-499) 243-33-37	Authorized officer A. Tokarev Telephone No. 8(495)531-65-15