

Informed Control Inc. Technical Information Disclosure

"System for failover in an identity federation"

(200411-03)

Published: January 30, 2008

BACKGROUND

An identity federation is a deployment of federated identity management software across multiple organizations. It is implemented by a collection of interoperable computing elements on a computer network, which enable users who have authenticated using the services provided by one organization to leverage this authentication state when accessing the computing resources and services provided by another organization.

In a typical federated deployment, one organization, termed the identity provider, may have users, customers or employees, who wish to access information resources and applications held by another organization, termed the service provider. The users have accounts and maintain their authentication credentials with their identity provider, and it is assumed that the identity provider maintains a database with records of each user that is permitted to access a service provider.

A user will interact with applications operated by the identity provider or service provider using a client program. One category of client programs is web browsers, such as Internet

“System for failover in an identity federation” continued

Explorer or Netscape Navigator. Web browsers are designed to provide users with remote access to web-based applications hosted on web servers. To access a web application, a web browser will establish a Transmission Control Protocol (TCP) connection across the Internet to a web server, and use the HyperText Transfer Protocol (HTTP) on the connection to transfer requests to the web server and responses back from the web server. A common format for data returned by a web server is the HyperText Markup Language (HTML), which the web browser will display to the user.

When a client operated by a user accesses a resource of a service provider, the service provider will rely on services operated by the identity provider to validate that the user has an active account with their identity provider, and the client has been authenticated by the identity provider as possessing the user's credential.

One means by which a service provider may query an identity provider is by using the Security Assertion Markup Language (SAML) Authentication Request Protocol, as defined in section 3.4 of the document "Assertions and Protocols for the OASIS Security Assertion Markup Language (SAML) V2.0", published by OASIS Open in March 2005. In this protocol, the service provider will send an AuthnRequest protocol data unit (PDU) to the identity provider, specifying as the subject of the request the name of a user whose authentication state is being queried. The identity provider will return a Response PDU that indicates whether the user is recognized by the identity provider, and if recognized, contains an assertion indicating whether and when that user has been authenticated by the identity provider.

If the identity provider becomes unreachable due to an outage or a break in the network between the service provider and identity provider, the service provider will no longer be able to

“System for failover in an identity federation” continued

receive responses to SAML AuthnRequest requests that it sends to the identity provider.

However, users elsewhere in the network may still be able to access a service provider and wish to make use of its services. Without a means of alternate authentication by which a user can failover their authentication to the service provider, a lack of SAML service from the identity provider will prevent a service provider from being able to further identity users and provide them with services, until the break in the network or outage of the identity provider is repaired.

SUMMARY

The objective of this system is to provide for authentication at a service provider of a user, when that user accesses the resources of a service provider in a federated deployment, and the identity provider for that user is unavailable due to a network or service outage.

If the authentication service or SAML responder of the identity provider is unavailable, then a cache in the service provider of user identities and credentials can be used instead to authenticate users for the service provider on behalf of the identity provider. As users do not provide the credentials by which they authenticate to the identity provider to the service provider, nor are these credentials available from the identity provider in a SAML protocol request, the credentials in the service provider's cache for a user will be distinct from the credentials the user would, under normal situations of no outage, provide to their identity provider during authentication. A credential stored by the service provider is termed in this specification an alternate token.

“System for failover in an identity federation” continued

DRAWINGS -- Figures

FIG. 1 is a diagram illustrating the components of a system to provide failover in an identity federation.

FIG. 2A, FIG. 2B and FIG. 2C are a flowchart illustrating the behavior of a thread in the service provider authorization service component.

FIG. 3 is a flowchart illustrating the behavior of a thread in the access control subroutine within the service provider resource component.

FIG. 4 is a flowchart illustrating the behavior of a thread in the identity provider SAML responder.

FIG. 5 is a flowchart illustrating the behavior of a thread in the identity provider user authentication component.

FIG. 6 is an illustration of a structure of the identity provider database as a relational database.

FIG. 7 is an illustration of a structure of the service provider database as a relational database.

FIG. 8 is an illustration of a network architecture of the identity provider.

FIG. 9 is an illustration of a network architecture of the service provider.

FIG. 10 is an illustration of components of a server computer.

FIG. 11 is an illustration of components of a client workstation computer.

“System for failover in an identity federation” continued

FIG. 12 is a collection of examples of Hypertext Markup Language (HTML) elements for response messages generated by the service provider authorization service.

FIG. 13 is an illustration of a directory information tree.

DETAILED DESCRIPTION

A federated deployment is modeled as three enclaves connected via a network: a service provider enclave (10), an identity provider enclave (20), and a client enclave (28). The network can be a public network, such as the Internet, or a private network based on the TCP/IP protocols.

The identity provider (20) information processing services are modeled as three components: a user authentication component (26), a SAML responder component (22) and a database component (24).

The identity provider user authentication component (26) is responsible for authenticating users on behalf of the service provider, when there is network connectivity between the service provider enclave and the identity provider enclave. The identity provider user authentication component can be realized as a web application. A client application (28) will send a user's login name and associated authentication credential to the user authentication component, and this component will validate the supplied credential against that stored for the user in the identity provider database (24).

“System for failover in an identity federation” continued

The identity provider SAML responder component (22) provides a service by which a service provider SAML initiator can query whether the identity provider has authenticated a user. This component can be realized as a web service.

The identity provider database component (24) can be modeled as a directory information tree or as tables in a relational database, depending on the configuration choice of the identity provider.

If the identity provider database component is modeled as a relational database, then this database comprises two tables, as illustrated in FIG. 6: the user table (140) and the service provider initiator table (142). The rows of these tables are created by the administrator (25), are updated by the user authentication component (26), and are queried by the user authentication (26) and SAML responder components (22).

The identity provider database user table (140) has one row for each user recognized by the identity provider. The primary key of the table is the USER UNIQUE ID column. The columns of the table are:

- USER UNIQUE ID: the value in this column is a unique identifier for the user,
- USER NAME: the value in this column is the login name of the user,
- CREDENTIALS: the value in this column is the authentication credential, such as a password, for the user,
- STATE: the value in this column is the account status of the user,

“System for failover in an identity federation” continued

- LAST SUCCESSFUL LOGIN DATE: the value in this column is the date and time that the user last successfully authenticated to the identity provider, and
- LAST LOGIN FAILURE DATE: the value in this column is the date and time of the last unsuccessful authentication attempt by the user to the identity provider.

The identity provider database service provider initiator table (142) has one row for each service provider recognized by the identity provider and permitted to query the identity provider for authentication status of users. The primary key of the table is the SP UNIQUE ID column.

The columns of the table are:

- SP UNIQUE ID: the value in this column is a unique identifier for the service provider,
- SP INITIATOR NAME: the value in this column is the login name of the service provider SAML initiator,
- CREDENTIALS: the value in this column is the authentication credential of the service provider SAML initiator, such as a password,
- STATE: the value in this column is the account status of the service provider, and
- NETWORKS: the value in this column is a list of network address ranges.

If the identity provider database component is modeled as a directory information tree, then this directory information tree is structured as two branches below a common entry for the identity provider, o=IdP (340), as illustrated in FIG. 14. The directory information tree is stored

“System for failover in an identity federation” continued

in a directory server providing directory service implementing the Lightweight Directory Access Protocol (LDAP) model, as described in the document "Lightweight Directory Access Protocol (v3)" by M. Wahl et al of December 1997.

In the ou=People branch, the entries immediately subordinate to the ou=People entry (342) represent users of the identity provider. For each user, one entry is present in the directory information tree. Each entry contains values for the following attributes:

- uid: a unique identifier for the user,
- userid: the username,
- userPassword: the password credential for the user,
- accountStatus: the status of the user's account,
- lastLoginDate: the date and time of the last successful login (this attribute is absent from the entry if the user has not yet successfully logged in), and
- lastLoginFailureDate: the date and time of the last login failure (this attribute is absent from the entry if there has not been a login failure for the account).

In the ou=SP branch, the entries immediately subordinate to the ou=SP entry (344) represent SAML initiators of service providers. One entry is present in the directory information tree for each service provider. Each entry contains values for the following attributes:

- uid: a unique identifier for the service provider,
- userid: the name of the SAML initiator of the service provider,

“System for failover in an identity federation” continued

- userPassword: the password credential for the SAML initiator,
- accountStatus: the status of the service provider's account, and
- networkAddress: a network address range for the service provider.

The processing elements of the identity provider (20) can be implemented as software running on computer systems attached to a local area network managed by the identity provider.

FIG. 8 illustrates the computer system components of a computer network for an identity provider. The SAML responder (22) and user authentication (26) components can be realized as application services running in a software execution environment on the application server computer (174). These services access the identity provider database, stored on the database server computer (176). The identity provider's local area networks are connected to the Internet via an ISP (162). Incoming requests from clients and from service provider SAML initiators traverse a firewall router (164), and are processed by a web server application running on a frontend web server computer (168), before being passed to the appropriate services on an application server computer (174).

The service provider (10) information processing services are modeled as four components: an application resource (10), an authorization service (14), a SAML initiator (16) and a database (12).

The service provider application resource component (18) comprises an information resource or information processing application that a user wishes to access. This component can be realized as a web server, a web application or a web service. The client application (28) will send requests to this resource which, if permitted by access control checks performed by the

“System for failover in an identity federation” continued

authorization service component (14), will cause the application resource component to perform a service and return information to the client.

The service provider authorization service component (14) operates in an endless loop, in which it receives an access request from the service provider application resource component (18) which is triggered by a client (28) attempting to access the resource, processes that request, replies to the application resource component, and then waits for the next request. This component can be realized as a centralized web service for all the resources in the service provider to access via a specialized protocol, or as a library within each application resource in the service provider.

The client access request data structure provided by the service provider application resource to the service provider authorization service incorporates the following fields:

- an identifier of the resource in the service provider,
- an identifier for the operation that the client intends to perform at that resource in the service provider,
- the username of the user making the request,
- the name of the identity provider where that user's account is held,
- an authentication token,
- a resource authorization token, and
- an alternate token.

Not all fields of the client access request data structure may have a value in the request received from a client.

“System for failover in an identity federation” continued

An authentication token generated by a service provider for a user typically consists of the concatenation of a unique identifier for the user, an expiration date for the authentication token, and a message digest. The message digest is generated by a hash function, such as the Secure Hash Algorithm (SHA-1) as specified in the document "Secure Hash Standard", Federal Information Processing Standards Publication 180-1 of April 1995. The input to the hash function is a concatenation of the unique identifier for the user, the expiration date, and a secret value known only to the service provider.

A resource authorization token generated by a service provider for a user typically consists of the concatenation of a unique identifier for the user, a unique identifier for the resource to which the user has been granted access, the access privileges of that user for that resource, an expiration date for the resource authorization token, and a message digest. The message digest is generated by a hash function, such as SHA-1. The input to the hash function is a concatenation of unique identifier for the user, a unique identifier for the resource to which the user has been granted access, the access privileges of that user for that resource, an expiration date for the resource authorization token, and a secret value known only to the service provider.

A reply which the service provider authorization service provides to the service provider application resource is in one of the following forms:

- a reject reply which indicates to the client that it is not currently permitted to access the resource,
- a reject reply with a prompt to the client to provide an alternate token,
- a redirect reply which indicates to the client that it should connect a different web site, in this case that of an identity provider,

“System for failover in an identity federation” continued

- a success reply which includes an alternate token and a resource authorization token to be sent to the client, and
- a success reply which includes a resource authorization token to be sent to the client.

The service provider SAML initiator component (16) operates under control of the authorization service component. The SAML initiator establishes outgoing TCP connections to the SAML responders of identity providers and sends SAML AuthnRequest messages to these responders. This component can be realized as a library within the service provider authorization service component.

The service provider database component (12) is modeled as a relational database, and this database comprises three tables, as illustrated in FIG. 7: a local table (150), an identity provider table (152) and an authorization table (154). The rows of these tables are created by the administrator (13), and are queried and updated by the authorization service component (14).

The service provider database local table (150) has one row for each user who has been authenticated by an identity provider, has subsequently accessed the service provider and has been authorized to access a service provider resource. The primary key of the table is the USER UNIQUE ID column. The columns of the table are:

- USER UNIQUE ID: the value in this column is a unique identifier for the user,
- USER NAME: the value in this column is the login name of the user,
- IDP ID: the value in this column is the name of the user's identity provider,

“System for failover in an identity federation” continued

- ALTERNATE TOKEN: the value in this column is either the alternate token credential for the user, if the alternate token credential has been set, or null if the alternate token credential has not been set,
- ALT GENERATION DATE: the value in this column is the date and time that the alternate token for the user was generated, and
- ALT LAST USE DATE: the value in this column is the date and time that the alternate token was last used for authentication.

The service provider database identity provider table (152) has one row for each identity provider. The primary key of the table is the IDP ID column. The columns of the table are:

- IDP ID: the value in this column is the name of an identity provider,
- LOGIN URL: the value in this column is the URL to which clients are redirected to authenticate to the identity provider,
- RESPONDER URL: the value in this column is the URL of the SAML responder of the identity provider,
- INITIATOR NAME: the value in this column is the login name of the SAML initiator when authenticating to the identity provider SAML responder,
- CREDENTIALS: the value in this column is the authentication credential, such as a password, of the SAML initiator used when authenticating to the identity provider SAML responder, and
- RESPONDER HOST ID: the value in this column is the acceptable hostname or set of IP addresses of the identity provider SAML responder.

“System for failover in an identity federation” continued

The service provider database authorization table (154) has one row for each user access right to access a service provider resource. The USER UNIQUE ID and RESOURCE ID columns together form the primary key for this table. The columns of the table are:

- USER UNIQUE ID: a unique identifier for a user,
- RESOURCE ID: a unique identifier for a resource of this service provider,
and
- RIGHTS: the access rights to a resource.

The processing elements of the service provider (10) can be implemented as software running on computer systems attached to a local area network managed by the service provider. FIG. 9 illustrates the components of a computer network for a service provider. The application resource (18), authorization service (14) and SAML initiator (16) components can be realized as application services running in a software execution environment on the application server computer (194). These services access the service provider database (12) stored on the database server computer (196). The service provider's local area networks are connected to the Internet via an ISP (182). Incoming requests from clients traverse a firewall router (184) and are processed by a web server application running on a frontend web server computer (188) before being passed to the application resource component (18) running on the application server computer (194).

FIG. 10 illustrates typical components of a computer system that performs server functions. Examples of computers that perform server functions include the identity provider frontend web server computer (168), identity provider application server computer (174), identity provider database server computer (176), service provider frontend web server computer (188),

“System for failover in an identity federation” continued

service provider application server computer (194) and service provider database server computer (196). The computer system (200) incorporates a system bus (204), a central processing unit (202), a BIOS ROM (206), a hard disk interface (210) and random access memory (208). A network interface (214) connects the computer system to a local area network switch (216). A hard disk drive (212) attached to the hard disk interface stores the program executable binary data and program data for the operating system (218) and applications (220). The RAM (208) contains the dynamic processing state, executable binary data and program data for the operating system (222) and applications (224).

The client application (28) can be implemented as software running on a computer system under the control of a user, attached to a TCP/IP-based network. This network can be a private network, or part of the Internet, and can be distinct from the local area networks managed by the service provider or identity provider. FIG. 11 illustrates typical components of a computer system that supports the client application. The computer system (240) incorporates a system bus (244), a central processing unit (242), a BIOS ROM (246), a hard disk interface (250), random access memory (248), a video interface (260) and a USB interface (264). A network interface (254) connects the computer system to an ISP (258) via a cable modem or DSL modem (256). A hard disk drive (252) attached to the hard disk interface stores the program executable binary data and program data for the operating system (270) and applications (272), including the client. The RAM (248) contains the dynamic processing state, executable code and data for the operating system (274) and applications (276). The user may interact with this computer system through the keyboard (266), mouse (268) or other input devices attached to the USB interface (264), and view the output on a monitor (262) attached to the video interface (260).

“System for failover in an identity federation” continued

Operations

The subsections of this section describe the processing operation of the service provider authorization service (14), the service provider application resource component (18), the identity provider SAML responder component (22) and the identity provider user authentication component (26).

Operations-- service provider authorization service

The behavior of the service provider authorization service component (14) is illustrated by the flowchart of FIG. 2A, FIG. 2B and FIG. 2C. This component comprises one or more threads of control. Each thread starts at step 32, and receives an incoming client access request from the service provider application resource component (18). Each request is provided to exactly one thread.

At step 34, the thread will determine the identity provider holding that user's data. If the client access request did not contain both a username and an identity provider name, then no identity provider can be identified for the request. If the client access request contains both a username and an identity provider name, then the thread will search the identity provider table (152) in the service provider database to find a row in which the value in the IDP ID column matches that provided by the user. If no row was found, then no identity provider can be identified for the request.

“System for failover in an identity federation” continued

At step 36, the thread will check whether an identity provider could be identified for the request. If an identity provider could not be identified, then at step 37 the thread will reject the request and indicate that the client must specify their username and identity provider name. An example of an HTML form is shown in block 290 in FIG. 12.

At step 38, the thread will check the validity of the authentication token. If an authentication token was included in the client access request, the thread will determine whether the token is unmodified and has not expired. To perform this check, the thread will first ensure that the expiration date of the token is after the current date and time: if the expiration date has already been reached, the authentication token is invalid. If the authentication token has not expired, the thread will next generate a message digest using a hash function. The input to the hash function is a concatenation of the unique identifier for the user, the expiration date, and a secret value known only to the service provider. If the message digest does not match that of the authentication token, the authentication token is invalid as it may have been modified.

If the client access request contains an authentication token that is still valid, then at step 81 the thread will check whether the user is authorized to access the service provider resource, by searching the authorization table (154) in the service provider database for a row with a USER UNIQUE ID and a RESOURCE ID matching the fields in the client access request. If a row is found, and the value in the RIGHTS column permits the operation requested by the client, then the user is authorized to perform the request, otherwise, if a row is not found, or the RIGHTS column does not permit the operation, then the user is not authorized. If the user is authorized, then at step 80 the thread will reply with a success indication. An example of an HTML encoding of this message is shown in block 298 in FIG. 12. If the user is not authorized, then at

“System for failover in an identity federation” continued

step 73 the thread will reject the request and indicate that the user is not authorized. An example of an HTML encoding of this message is shown in block 292 in FIG. 12.

If the client access request does not contain an authentication token, or if the authentication token is expired or not valid, then the thread will attempt to contact the SAML responder of the identity provider for that user. The URL of the SAML responder is obtained from the value of the RESPONDER URL column of the row in the service provider database identity provider table in which the IDP ID column matches the name of the identity provider. At step 40, the thread will establish an outgoing TCP connection to the IP address and TCP port of the SAML responder as stated in the responder URL, and if the TCP connection is established, negotiate the use of the Transport Layer Security (TLS) or Secure Sockets Layer (SSL) protocol on that connection. If the TLS or SSL protocol negotiation is successful, the thread will validate the certificate path returned by the SAML responder, and match the hostname against the value of the RESPONDER HOST ID column in the row obtained from the service provider database identity provider table for that identity provider.

At step 42, the thread will check whether it could establish a connection to the identity provider's SAML responder. If it could not establish a connection, then it will continue processing at step 62.

Otherwise, if the service provider authorization service successfully established a TLS or SSL protected connection to the identity provider SAML responder, then the thread will send a SAML AuthnRequest request within a SOAP request over that connection. In the SAML AuthnRequest, the thread will set the ID attribute to a randomly chosen value, the Version attribute to "2.0", the IssueInstant attribute to the current time, and include an Issuer element that

“System for failover in an identity federation” continued

contains a URL of the service provider, and a Subject element containing a NameID element. In the Subject element, the thread will set the Format attribute to a value that indicates the format of user names used in the deployment, and set the value of the NameID element to the user name.

The thread will include in either the HTTP headers or SOAP request the initiator name and credential of the service provider SAML initiator, as obtained from the values in the INITIATOR NAME and CREDENTIALS columns of the row obtained from the service provider database identity provider table for that identity provider.

At step 58, the thread will wait for a response to this request from the identity provider SAML responder on this connection. At step 60, if the service provider authorization service timed out waiting for a response, or if the connection was closed, processing will continue at step 62.

At step 64, the thread will parse the SAML Response PDU from the SOAP response received from the identity provider SAML responder. When parsing, the thread will follow the procedure documented in "<Response> Message Processing Rules", section 4.1.4.3 of the document "Profiles for the OASIS Security Assertion Markup Language (SAML) V2.0", published by OASIS Open in March 2005. If there is an assertion present in the response with an AuthnStatement, then the user has been authenticated by the identity provider, and the thread will obtain the date and time of the authentication from the AuthnInstant attribute of the AuthnStatement element. If the response indicates that the user has not been authenticated by the identity provider, or if the date and time of the user's most recent authentication by the identity provider is further in the past than the service provider authorization service would accept, then at step 66 the thread will reject the client access request, and indicate that the client is to be

“System for failover in an identity federation” continued

redirected to the identity provider. An example of a HTML message is shown in block 294 of FIG. 12.

If the response from the identity provider SAML responder indicates that the user has been authenticated by the identity provider, then at step 72 the thread will check whether the user is authorized to access the service provider resource indicated in the client access request. The thread will search the authorization table (154) in the service provider database for a row with a USER UNIQUE ID and a RESOURCE ID matching the fields in the client access request. If a row is found, and the value in the RIGHTS column permits the operation requested by the client, then the user is authorized to perform the request, otherwise, if a row is not found, or the RIGHTS column does not permit the operation, then the user is not authorized. If the user is not authorized, then at step 73 the thread will reject the request and indicate that the user is not authorized. An example of an HTML encoding of this message is shown in block 292 in FIG. 12.

If the user is authorized to access the resource, then at step 77 the thread will check whether the user has an alternate token as a value of the ALTERNATE TOKEN column in the row for that user in the local table of the service provider database. If the row for the user has an alternate token, then at step 80 the thread will reply with a success indication. An example of an HTML encoding of this message is shown in block 298 in FIG. 12.

If there is no value for the ALTERNATE TOKEN column in the row for the user in the local table of the service provider database, then at step 78 the thread will generate a new alternate token. This alternate token will be stored as a value of the ALTERNATE TOKEN column in the row for the user in the local table, and the current date and time will be stored as a value of the ALT GENERATION DATE column in that same row. At step 79 the thread will

“System for failover in an identity federation” continued

reply success and provide this alternate token. An example of an HTML message is shown in block 296 of FIG. 12.

If the service provider authorization service could not establish a connection to the identity provider SAML responder, or the service provider authorization service timed out waiting for a response from the identity provider SAML responder, then at step 62 the thread will check whether the row for the user in the local table of the service provider database has a value in the ALTERNATE TOKEN column. If the row for the user does not have an alternate token, then at step 75 the thread will reject the client access request with an indication that the identity provider is not available. An example of an HTML message is shown in block 300 of FIG. 12.

If the row for the user in the local table has an alternate token, then at step 65 the thread will check whether the client access request contains an alternate token. If the client access request does not contain an alternate token, then at step 68 the thread will reject the client access request with a prompt to the client to provide an alternate token. An example of an HTML message with this form is shown in block 302 of FIG. 12.

If the client access request contains an alternate token, then at step 70 the thread will check whether the alternate token in the client access request matches the alternate token from the row in the local table matching that user. If the two values of alternate token are not equal, then at step 74 the thread will reject the request with an indication that the alternate token does not match. Otherwise, if the two values of alternate token are equal, the thread will set the value of the ALT LAST USE DATE column of the row for the user in the local table to the current date and continue processing at step 72.

“System for failover in an identity federation” continued

Operations -- service provider resource component

The service provider application resource component (18) comprises one or more threads of control. The behavior of a thread in the access control processing subroutine of this component is illustrated by the flowchart of FIG. 3.

A thread entering the subroutine is provided with a client access request from the service provider resource component. The service provider resource component can construct the client access request from cookie header parameters provided by the client in the HTTP header of a HTTP request to the service provider resource.

At step 86 the thread entering the access control processing subroutine will get the user's authorization parameters, if any, from the client access request. At step 87, the thread will determine if there is a valid resource authorization token in the client access request. To perform this check, the thread will first ensure that the resource authorization token is present and the expiration date of the resource authorization token is after the current date and time: if the resource authorization token is absent or the expiration date has already been reached, the resource authorization token is invalid. If the resource authorization token is present and has not expired, the access control processing will next generate a message digest using a hash function. The input to the hash function is a concatenation of the unique identifier for the user, the unique identifier for the resource, the expiration date, and a secret value known only to the service provider. If the message digest does not match that of the resource authorization token, the resource authorization token is invalid as it may have been modified. If the resource

“System for failover in an identity federation” continued

authorization token is present and valid, then at step 88 the thread will grant the client access to the resource.

If the resource authorization token is absent or invalid in the client access request, then at step 89 the thread will provide the client access request to the service provider authorization service. At step 90 the thread will get the reply from the service provider authorization service. At step 91 the thread will inspect the reply from the service provider authorization service. If the service provider authorization service reply indicated that the client's request was to be rejected or redirected, other than a prompt to the client to provide an alternate token, then at step 92 the thread will return to the client an indication of this error or redirection.

At step 93 the thread will determine if the reply from the service provider authorization service was a rejection with a prompt to the client to provide an alternate token. If it was, then at step 94 the thread will return to the client a form for the client to provide an alternate token.

If the service provider authorization service reply included any of a resource authorization token, an alternate token, or an authentication token, then at step 95 the thread will include these tokens in the response to be sent to the client. These tokens are provided to the client as cookies using the Set-Cookie header, and the alternate token is provided to the client in HTML. At step 96 the thread will return an indication to the client of a successful authentication.

Operations -- identity provider SAML responder

The identity provider SAML responder component (22) comprises one or more threads of control. The behavior of a thread of control in this component is illustrated by the flowchart of

“System for failover in an identity federation” continued

FIG. 4. Each thread will follow an endless loop waiting for and then processing requests from SAML initiators of service providers.

At step 102, the thread will wait for a request from a SAML initiator. Each request is assigned to exactly one thread.

At step 104, the thread will authenticate the SAML initiator.

If the identity provider database (24) is modeled as a relational database, then the thread will authenticate the SAML initiator using the service provider initiator table (142). The thread will search the service provider initiator table in the identity provider database for a row in which the value of the SP INITIATOR NAME column matches the name of the SAML initiator supplied with the SAML request. If one row is found, the value of the CREDENTIALS column matches the credential of the SAML initiator supplied with the SAML request, the value of the STATE column does not indicate the account is disabled, and the value of the NETWORKS column contains an IP network number range which contains the IP address of the incoming connection on which this request was received, then the SAML initiator was authenticated.

Otherwise, if the identity provider database (24) is modeled as a directory information tree, then the thread will authenticate the SAML initiator by searching the service provider branch (344) of the directory information tree in the identity provider database for an entry in which the value of the userid attribute matches the name of the SAML initiator supplied with the SAML request. If one entry is found, the value of the userPassword attribute in that entry matches the credential of the SAML initiator supplied with the SAML request, the value of the networkAddress attribute, if present, contains an IP network number range which contains the IP

“System for failover in an identity federation” continued

address of the incoming connection on which this request was received, and the accountStatus attribute doesn't indicate that the service provider's account has been disabled, then the SAML initiator was authenticated.

At step 106, the thread will branch on whether the SAML initiator was authenticated in the previous step. If the SAML initiator was not authenticated, then at step 108 the thread will send an error response to the SAML initiator.

If the SAML responder has authenticated the SAML initiator, then at step 110 the thread will parse the request from the SAML initiator. At step 112, the thread will determine whether the request was valid. If the request was not valid, then at step 114 the thread will send an error response to the SAML initiator.

If the request was valid, then at step 116 the thread will retrieve a record for the user from the identity provider database.

If the identity provider database (24) is modeled as a relational database, then the thread will check the user table (140) for the specified user identity, by searching for a row in that table in which the user name supplied in the subject of the SAML AuthnRequest request matches a value in the USER NAME column.

Otherwise, if the identity provider database (24) is modeled as a directory information tree, then the thread will check the directory for the specified user identity, by searching for an entry in the ou=People branch (342) in which the user name supplied in the subject of the SAML AuthnRequest request matches the userid attribute. If one entry is found, the value of the

“System for failover in an identity federation” continued

lastLoginDate attribute in that entry, if present, contains the date and time of the user's last successful login.

At step 118, the thread will build a response that indicates whether a record for the user was found, and if so, includes the last successful login date and time. If no record was found, the thread will build a Response that will contain an Issuer element that specifies the URL of identity provider, and a Status element with a StatusCode and a Value attribute of "urn:oasis:names:tc:SAML:2.0:status:UnknownPrincipal". If a record was found, the thread will build a Response that will contain an Issuer element that specifies the URL of the identity provider, a Status element with a StatusCode and a Value attribute of "urn:oasis:names:tc:SAML:2.0:status:Success", and a SAML Assertion element. In the Response element, the thread will set the Version attribute to "2.0", the ID attribute to a randomly chosen value, and the IssueInstant attribute to the current time. If an Assertion element is included in the Response, the thread will set in the Assertion element the Version attribute to "2.0", the ID attribute to a randomly chosen value, and the IssueInstant attribute to the current time. In building the Assertion element, the thread will follow the procedure documented in "<Response> Usage", section 4.1.4.2 of the document "Profiles for the OASIS Security Assertion Markup Language (SAML) V2.0", published by OASIS Open in March 2005, except that the restriction that an assertion containing an AuthnStatement must contain a Subject element with at least one SubjectConfirmation element containing a Method of "urn:oasis:names:tc:SAML:2.0:cm:bearer" need not be implemented. Within the Assertion element, the thread will include an Issuer element that specifies the identity of the identity provider, and a Subject element that contains the same NameID as provided by the SAML initiator in the AuthnRequest. The thread

“System for failover in an identity federation” continued

will also include an AudienceRestriction element in the Assertion that includes the identifier of the service provider as an Audience value. If the last successful login date and time for the user is available (provided by an attribute of the directory entry, or if a relational database is used, if the value of the LAST SUCCESSFUL LOGIN DATE column in the row for the user in the user table is not null), then the thread will include within the Assertion element an AuthnStatement element, and in that element will set an AuthnInstant attribute with a value of the date and time the last successful login occurred. If the AuthnStatement element is present, the thread will include within it an AuthnContext element that contains an AuthnContextClassRef element, with a value that is a code which specifies the authentication context by which the identity provider authenticated the user. The codes for authentication contexts are defined in the document "Authentication Context for the OASIS Security Assertion Markup Language (SAML) V2.0", published by OASIS Open in March 2005. In particular, if password-based credentials were used to authenticate the user, the code is the string "urn:oasis:names:tc:SAML:2.0:ac:classes:Password".

Operations -- identity provider user authentication component

The identity provider user authentication component (26) comprises one or more threads of control. The behavior of a thread in this component is illustrated by the flowchart of FIG. 5. Each thread will follow an endless loop waiting for and then processing requests from clients.

At step 122, the thread will wait for a request by a user to authenticate. Each request is provided to exactly one thread.

“System for failover in an identity federation” continued

At step 124, the thread will authenticate the user. If the identity provider database (24) is modeled as a relational database, then the thread will locate the user's record searching for a row in the user table (140) in which the user name supplied in the request matches a value of the USER NAME column. If no row was found, then the user cannot be authenticated. Otherwise, if the identity provider database (24) is modeled as a directory information tree, then the thread will locate the user's record by searching for an entry in the ou=People branch of the directory information tree (342) in which the user name supplied in the request matches a value of the userid attribute. If no entry is found, then the user cannot be authenticated

At step 126, the thread will check whether the user is authenticated. If the identity provider database (24) is modeled as a relational database, and a row was found, then the thread will compare the credential supplied by the user with the value of the CREDENTIALS column in that row. If a row is found, the credentials match, and the value of the STATE column does not indicate the account is disabled, then the user is authenticated. Otherwise, if the identity provider database (24) is modeled as a directory information tree, and an entry was found, then the thread will compare the credential supplied by the user with the value of the userPassword attribute. If an entry is found, the credentials match and the accountStatus attribute doesn't indicate that the user's account has been disabled, then the user is authenticated.

If a record was found for the user but the user cannot be authenticated, then at step 128 the thread will set the last login failure date for the user. If the identity provider database (24) is modeled as a relational database, then the thread will update the row for the user by setting the value in the LAST LOGIN FAILURE DATE column to the current date and time, and then the thread will continue at step 130. Otherwise, if the identity provider database (24) is modeled as a

“System for failover in an identity federation” continued

directory information tree, then the thread will modify the entry for the user by replacing the value of the lastLoginFailureDate attribute to the current date and time, and then the thread will continue at step 130.

At step 130 the thread will send an error response to the user, and then loop back to wait for the next request.

Otherwise, the user was authenticated, and at step 132, the thread will update the last successful login date for the user. If the identity provider database (24) is modeled as a relational database, then the thread will update the row for the user in the user table by setting a value in the LAST SUCCESSFUL LOGIN DATE to the current date and time, and continue to step 134. Otherwise, if the identity provider database (24) is modeled as a directory information tree, then the thread will update the entry for the user in the directory by replacing the value of the lastLoginDate attribute to the current date and time.

At step 134 the thread will send a success response to the user, and then loop back to wait for the next request.