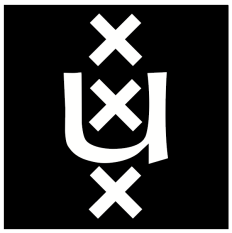
Thinking in possibilities for federated log out.

Marcel den Reijer & Fouad Makioui

Supervised by
Thijs Kinkhorst & Joost van Dijk
SURFnet



University of Amsterdam

June 9, 2017

Abstract

Security Association Markup Language version 2.0 (SAML 2.0) is an open standard protocol defined by OASIS for exchanging security information. SAML 2.0 supports different protocols such as Single Sign On and Single Logout. This report describes the possible solutions for federated log out in a *hub and spokes model* of SURFconext. The possible solutions are based on the expectation of the users, which entails that the user only logs out from the corresponding application. This expectation is defined as Partial Logout. Partial Logout is not a standard function of SAML 2.0.

Contents

1 Introduction

Most systems are protected by a form of authentication/authorization technology. For users to be able to access their resources, they must verify their identity by some form of authentication. There are several methods to provide authentication to a system. The most common authentication method is username/password authentication. Other methods are authenticating with bio metric data such as fingerprints/iris scans, smart cards, soft or hard tokens and certificates.[1]

There is a need for a federated identity infrastructure to manage the user's access (authentication) to multiple systems and to be able to keep the identities across different systems manageable. A federated identity infrastructure links a person's digital identity and attributes to a stored multiple distinct identity management systems. [7]

Federated identity makes it possible to decouple the authentication and authorization functions. The applications do not need to store user credentials, because the user credentials are stored by the authority server. The benefit for the users is that the users have one identity for accessing different applications.
Single Sign On can be implemented in a federated infrastructure, this gives the users the benefit that the users only needs to authenticate once by applications that are part of the federation.[5]

There are two major protocols in federated identity for web authentication: SAML 2.0 and OpenID connect.[5]

This research will mainly focus on the SAML 2.0 protocol, because this is currently the most implemented in identity federation for higher education and research.[3] SAML 2.0 stands for Security Assertion Markup Language version 2.0 and is an open-standard protocol based on XML for exchanging security data between three parties: Service Provider, Identity Provider and the user agent. SAML 2.0 is developed by OASIS and the latest official version released on March 2005. SAML can be used to implement Single Sign On and/or Single Logout.[2]

The SAML 2.0 protocol provides a possibility for Single Logout (Aslo called Single Logout protocol)[9]. The Single Logout allows users to terminate their session. This is propagated to all services that shared the session. This works if all sessions of the user are responding on the logout request of the Identity Provider. The Single Logout as defined in the SAML 2.0 is not implemented often.[14]

1.1 Research Questions

It is clear that Single Logout is not often implemented.[14] The idea behind this project is to research which possibilities there are for logout in a federation. Therefore, our main research question is:

- ***What are the possibilities for Federated log out?***

This main research question is supported by the following sub-questions:

- *What do the users expect when they log out of a Service Provider?*
- *Which possible solutions provide (at least some of) the user's expectations?*

  - Specified with the pros and cons in the field of technical implementation and feasibility.

- *Which solution(s) is/are the most feasible one(s)?*

  - Verify that the most feasible one can work in a proof of concept.

1.2 Scope

This research project focuses on finding a possible solution for the problem, which is discussed in the introduction. The case of this project refers to SURFnet and its environment as described in section 3.4. The solution must work on SAML 2.0 in a *hub and spoke model*. If there is more than one solution, the best solution will be tested in a proof of concept. The experiment will be performed on the two most used Identity Providers applications by the providers that are connected to SURFconext which are Microsoft ADFS and SimpleSAMLphp. Figure 1 illustrates the product overview[1] of the connected Identity Providers.

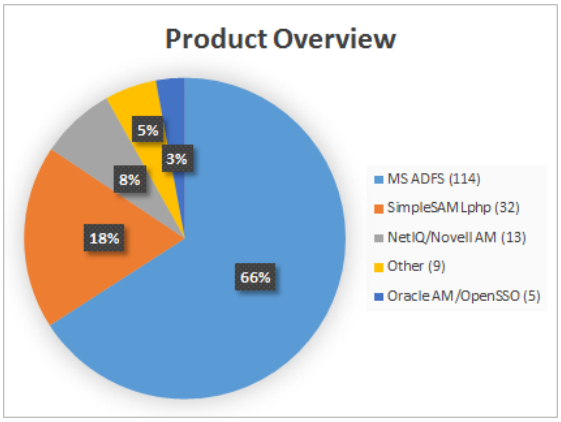


Figure 1: SURFconext - Identity Providers product overview (13 January 2017) [4]

[1] on 13 January 2017

2 Related Work

"Shibboleth is among the world's most widely deployed federated identity solutions, connecting users to applications both within and between organizations."[15] Shibboleth project has done some research in the field of Single Logout and has written an article about unfamiliar problems. Shibboleth[15] discussed only the practical issues that may occur. The article "SLOissues" begins with an introduction of web application sessions. Shibboleth[15] claims that:

"most web-based applications store data about the currently active users within memory and index this collection of data with randomly generated number called a SESSIONID. The SESSIONID is in a normal way stored in a browser cookie so that each time the user returns the cookie is presented to the application and the application can find the relevant current session data."

Shibboleth mentioned some technical difficulties for Single Logout. These difficulties can be classified in three categories. The first category is the Front-Channel bindings revisited. What happened when an HTTP error is encountered during the HTTP sessions, while information is being transported. Shibboleth mentioned some other issues when Front-Channel bindings in SAML 2.0 is used. The second category concern the Back-channel bindings. Also these issues, which may occur are also related to sessions. The third category is the SAML versions. Only SAML 2.0 supports Single Logout, so all parties have to support SAML 2.0.[15] According to Shibboleth[15]:

"When presented with these issues many individuals offer a protocol where only the initiating Service Provider and the Identity Provider communicate. The goal is to destroy the Service Provider and the Identity Provider its sessions so that users would have to re-authenticate if they visit that Service Provider again."

SURFnet did some research about federated logout. The conclusion of the research was that SAML 2.0 did not have a proper solution for federated logout. SURFnet decided that they don't support the logout possibility as specified by SAML 2.0. [21]

Sanna Suoranta et al. did research about Logout in Single Sign On Systems. Their conclusion is: *"Standardization organizations, web browser vendors, and service developers should consider also the termination of the sessions."* According to them the solution lies in enhanced cookie management in web browsers and a unified user interface for logout.[14]

3 Desk Research

3.1 SAML 2.0

3.1.1 SAML 2.0 introduction

SAML 2.0 is an XML based open-standard protocol for exchanging security information between parties. The standard defines and describes the syntax and rules for requesting, creating and transferring with security information between or across security domain boundaries using trusts. There are three participants when SAML is used. These SAML participants are the SAML asserting party, the SAML replying party and the user. The SAML asserting party makes assertions and is also known as the SAML authority or Identity Provider. The SAML relying party is a system entity that uses received assertions from the SAML authority. The relying party also called a Service Provider. Another SAML participant is the user also called principal which runs a web browser or SAML-enabled application.[12]

3.1.2 SAML 2.0 concept architecture

The concept architecture of SAML 2.0 consists of "building-block" components. When the components are put together, it can supports a number of use cases. If there is a trust relationship between the parties, then SAML authority permits transfer of user's identity, authentication, attribute and authorization information. The Identity Provider is the party where the user identity is stored. The SAML assertion XML schema describes the valid structure and contents of an assertions. Figure 2 illustrates an architecture overview of SAML 2.0.[12]

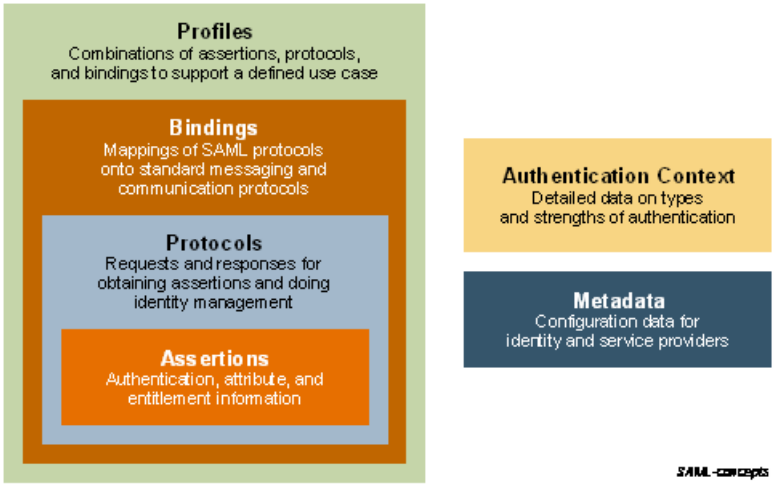


Figure 2: SAML 2.0 - Architecture [12]

The SAML 2.0 architecture contains six parts with Protocols and Assertions known as the SAML Core. The following parts are[12]:

- Assertions: Contains information about the user. This information can be authentication, attributes or entitlement information. An assertion is a package of information that supplies zero or more statements made by a SAML authority.
- Protocols: Defines the SAML requests and responses appropriate.
- Bindings: Defines how SAML messages can be transported (for example over HTTP with SOAP) between participants.
- Profiles: Define how the components SAML assertions, protocols and bindings are combined.
- Metadata: Contains its own XML schema. This schema is used to express and share configuration information between SAML parties. This configuration data contains the identifiers, supported identity attributes and encryption or signing information.
- Authentication Context: Contains detailed information about types and security strengths of authentication.

3.1.3 Statements

There are three kinds of statements that can be carried within an assertion. The first statement is authentication, which is created by the party where the user successfully is authenticated. The second statement is authorization, which contains the kind of information about the authenticated users with its permissions. The third statement is attribute, which contains the kind of specific information about a subject with identifying properties.[12]

3.2 Single Sign On

3.2.1 The definition of Single Sign On

Single Sign On is a concept whereby a user authenticates against an Identity Provider once and gets access to several Service Providers. To implement Single Sign On, SAML Authentication Request Protocol is used in the background in conjunction with a front-channel binding such as HTTP Redirect, HTTP POST or HTTP Artifact. The Single Sign On concept is handled by the Identity Providers, which keeps the sessions information of the users.[11] Figure 3 illustrates a basic template of Single Sign On.

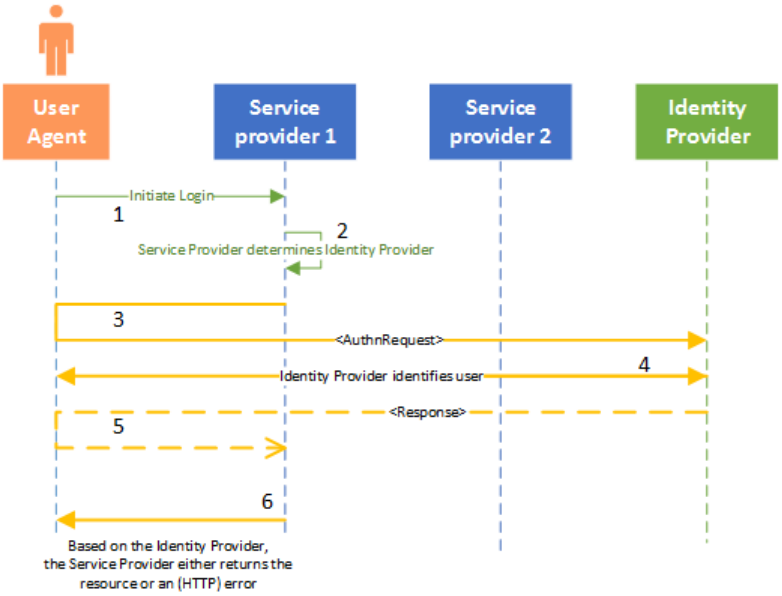


Figure 3: SAML 2.0 - Single Sign On [11]

1. The user initiates a login request.
2. The Service Provider determines the Identity Provider and which binding it can use.
3. The Service Provider sends an authentication request via the user agent to the determined Identity Provider.
4. The user is identified by the Identity Provider.
5. The Identity Provider sends a response message to the Service Provider via the user agent.
6. Based on the Identity Provider the Service Provider may respond to the user agent with its own status code.

3.2.2 Authentication Request Protocol

The first SAML message is a <AuthnRequest> message request and this message contains an <Issuer> element. This element contains the unique identifier of the requesting Service Provider. The authentication request needs to conform the Identity Providers requirements. If the request does not conform the requirements the Identity Provider respond with a response message, which contains a specific error status code. According to OASIS[11] the <AuthnRequest> message may be signed if the HTTP Artifact binding is used. Authentication of the parties is optional but when they are not authenticated, signed or integrity protected, they must not be trusted. When an error occurs during responding, then the response message contains no assertions.

If the request is successful, then every response element has to conform to the following rules: [11]

- The <Issuer> element may be omitted, but if it is present then it must contain a unique identifier.
- The response must have at least one <Assertions> element. The <Assertions> must contain one or more <AuthnStatement>.
- The <AuthnStatement> must contain a <Subject> element with at least one <SubjectConfirmation> element containing a <Method>.
- If the assertions or responses includes a signature, the Service Provider has to verify them.
- The Service Provider must verify the <Recipient> attribute in any carrier <SubjectConfirmationData> matches the assertion consumer service URL.
- The Service Provider must verify that the <NotOnOrAfter> attribute in any carrier <SubjectConfirmationData> has not expired, subject to allowable clock skew between the providers.
- The provided assertions must be signed when the HTTP POST binding is used to deliver the <Response> messages by the Service Provider and must protect the messages against replaying of the messages.
- The metadata defines different kinds of endpoint elements for Single Sign On to describe the supported bindings, locations certificate and signatures. These locations can be used by Service Providers to send request to an Identity Provider.

3.3 Single Logout

3.3.1 Single Logout definition

The Single Logout protocol, which is described in section 3.7 of the SAML core and section 4.4 of the SAML profiles. OASIS[10] [11] says:

"The Single Logout protocol provides a message exchange protocol by which all sessions provided by a particular session authority are near-simultaneously terminated."

In the Single Logout protocol different participants participate. The first one is the principal, which refers to the (end) user, the second one is the session participant which refers to the Service Provider and the last participant is the session authority, which refers to the Identity Providers or users. [10]

In Single Logout, the principal has an authenticated session with session participants and session authorities. Figure 4 illustrates a high-level view of an authenticated session

between the two participants.

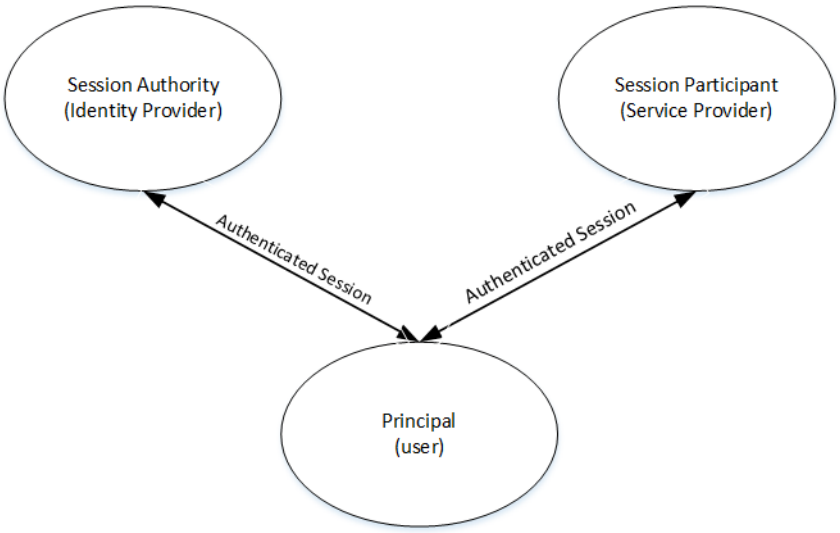


Figure 4: SAML 2.0 - Authenticated session

The Single Logout protocol exists of two elements; the LogoutRequest and LogoutResponse elements.

According to OASIS[10]: *"The session participant MUST send a <LogoutRequest> message to the session authority, when the principal invokes the Single Logout process at a session participant."*

According to OASIS[10]: *"The session authority SHOULD send a <logoutRequest> message to each session participant to which it provided assertions containing authentication statements under its current session with the principal, except the session participant which send the request message to the session authority. This count only when either the principal invokes a logout at the session authority, or a session participant send a logout request to the session authority specifying the principal itself."*

Different kinds of bindings
The profile can be used to combine a synchronous or asynchronous binding to define how Single Logout messages with SAML 2.0 can be transported. An example of a synchronous binding (back-channel) is the SOAP or PAOS binding. An example of an asynchronous binding (front-channel) is the HTTP Redirect, POST or Artifact bindings. The asynchronous binding needs to be used in cases, where a user's session state exists only in a user agent in the form of a cookie and/or direct interaction between the session participant or session authority and the user agent. The synchronous binding needs to be used in cases where a user wants to communicate directly with the Identity Provider.

The session participants should propagate the required messages to each other. The synchronous/asynchronous binding can only be used for communication when a logout request/response is issued by the Service Provider to Identity Provider. When the Identity Provider initiates the logout request then the Identity Provider has to examine the identifier and <SessionIndex> elements and define the set of sessions to be terminated [11]

The working of Single Logout

Figure 5 illustrates a sequence diagram of the working of the Single Logout protocol.

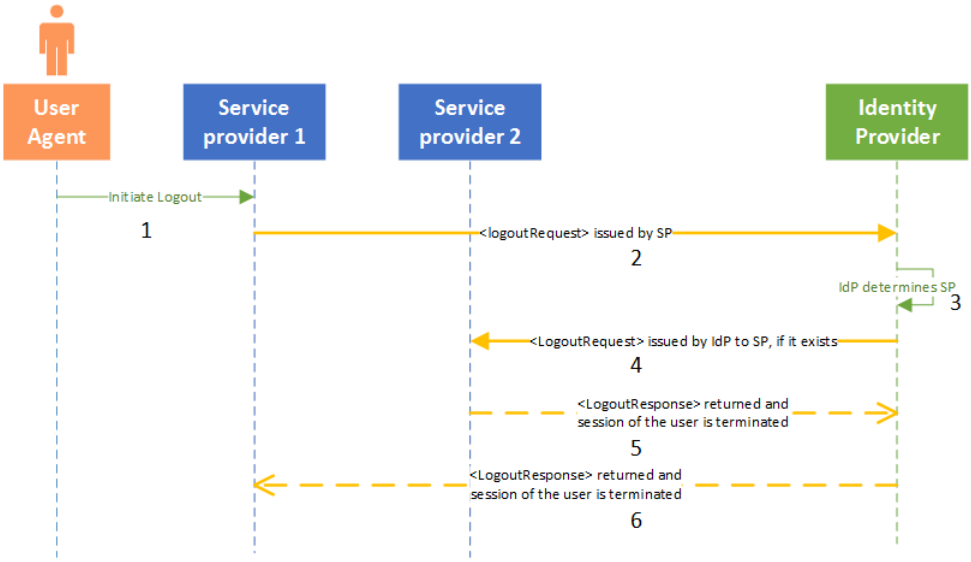


Figure 5: SAML 2.0 - Single Logout [11]

1. The user initiates a logout to start the Single Logout process.
2. The Service Provider sends a <LogoutRequest> to the Identity Provider and terminate the session.
3. The Identity Provider determines which sessions exist with the services providers.
4. The Identity Provider sends the LogoutRequest through to all the determined services providers one by one after receiving a LogoutResponse.
5. The Service Provider terminates the session and sends a LogoutResponse back to the Identity Provider. If there are more Service Providers participating in the session, step 3 and 4 will repeated.
6. After receiving all LogoutResponses from the participating Service Providers the Identity Provider send a LogoutResponse back to the Service Provider.

3.3.2 Logout Request element

Either the session participant or session authority will indicate that a session has terminated by sending a <LogoutRequest> message. It is recommended that every request is signed. The logout request element has a complex type of a LogoutRequestType sub element, which extends a RequestAbstractType and is used to add extra attributes or sub elements. According to OASIS[10] the following sub elements and attributes can be added to the Logout Request element:

- <saml:BaseID> or <saml:NameID> or <saml:EncryptedID> [Required] This sub element contains the identifier and associated attributes, which specify a specific user as currently recognized by the corresponding Identity Provider and Service Provider.
- <SessionIndex> [Optional] This sub element contains the identifier, which indexes the corresponding session at the message of the receiver. As shown in Figure 6 the SessionIndex can contain multiple values.
- <Reason> [Optional] This sub element contains the reason for the log out, in a URI reference form. The sender of the message may use this attribute to indicate the reason of the logout request. There are two value possible:
  - <urn:oasis:names:tc:SAML:2.0:logout:user> This value can be sent if the user wants to terminate a indicated/corresponding session.
  - <urn:oasis:names:tc:SAML:2.0:logout:admin> This value can be sent if the administrator of a Service Provider wants to terminate a indicated or corresponding session.

The schema fragment in Figure 6 defines the <logoutRequest> element:

```
<element name="LogoutRequest" type="samlp:LogoutRequestType"/>
    <complexType name="LogoutRequestType">
        <complexContent>
            <extension base="samlp:RequestAbstractType">
                <sequence>
                    <choice>
                        <element ref="saml:BaseID"/>
                        <element ref="saml:NameID"/>
                        <element ref="saml:EncryptedID"/>
                    </choice>
                    <element ref="samlp:SessionIndex" minOccurs="0" max Occurs ="
unbounded"/>
                </sequence>
                <attribute name="Reason" type="string" use="optional"/>
                <attribute name="NotOnOrAfter" type="dateTime"use="optional"/>
            </extension>
        </complexContent>
    </complexType>
    <element name="SessionIndex" type="string"/>
```

Figure 6: SAML 2.0 - LogoutRequestType [10]

In a <LogoutRequest> the <Issuer> element is required and must contain a unique identifier of the requesting entity. According to OASIS[11]: *"the Format attribute MUST be omitted or have a value of "urn:oasis:names:tc:SAML:2.0:-nameidformat:entity"*. If the request origins from a Service Provider, the <SessionIndex> element is required.

3.3.3 Logout Response element

The receiver of a LogoutRequest message must always respond with a StatusResponseType message. It is recommended to sign the Logout Response element.[10]

The following schema fragment defines the <LogoutResponse> element:
<element name=LogoutResponse type=samlp:StatusResponseType/>
In a <LogoutResponse> the <Issuer> element is required and have to contain a unique identifier of the responding entity. According to OASIS[11]: *"the <Format> attribute have to be omitted or have a value of urn:oasis:names:tc:SAML:2.0:nameid-format:entity"*.

3.3.4 Metadata

The metadata is used to define an endpoint: <md:SingleLogoutService> that describes the supported bindings and location(s). The endpoint is used by the entity to send requests and responses to.

3.4 SURFconext

SURFnet is a company that provides services to higher educations and research institutes. For example SURFconext is a service provided by SURFnet.

SURFconext is an infrastructure for online collaboration. It gives the users access to a range of services through their identity provided by an institute.[17]

SURFconext make uses of a *hub and spoke model*. 99% of the Service Providers by SURFconext use SAML 2.0 for exchanging information and they currently[2] provide the platform for 173 Identity Providers and 800 Service Providers[4]. Table 1 gives an overview of the usage and which products the Identity Providers use.

SURFconext is the hub between the Service Providers and the Identity Providers. They represent themself as a Service Provider to the Identity Providers and as an Identity Provider to the Service Providers. Figure 8 illustrates a schematic overview of SURFconext. Graph 7 illustrates the logins per week over the last 5 years. As the graph shows, the logins which SURFconext processes increases every year. "This trend seems to continue" [4] The expection is that more Service Providers will use OpenID Connect in the future[3].

[2] 31 January 2017

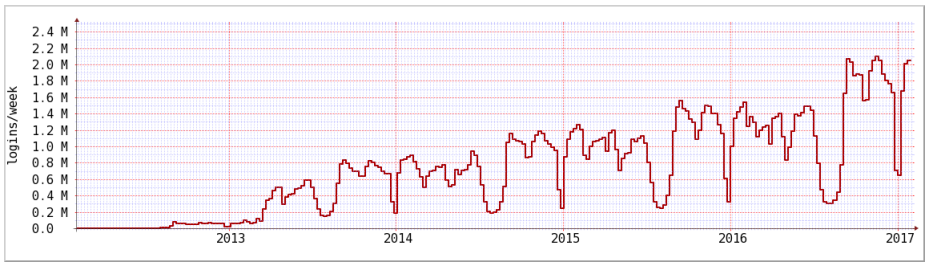


Figure 7: SURFconext - Login overview [22]

Figure 8 illustrates the *hub and spoke model* of SURFconext. On the left side the Identity Providers are illustrated and on the right side the Service Providers are illustrated.

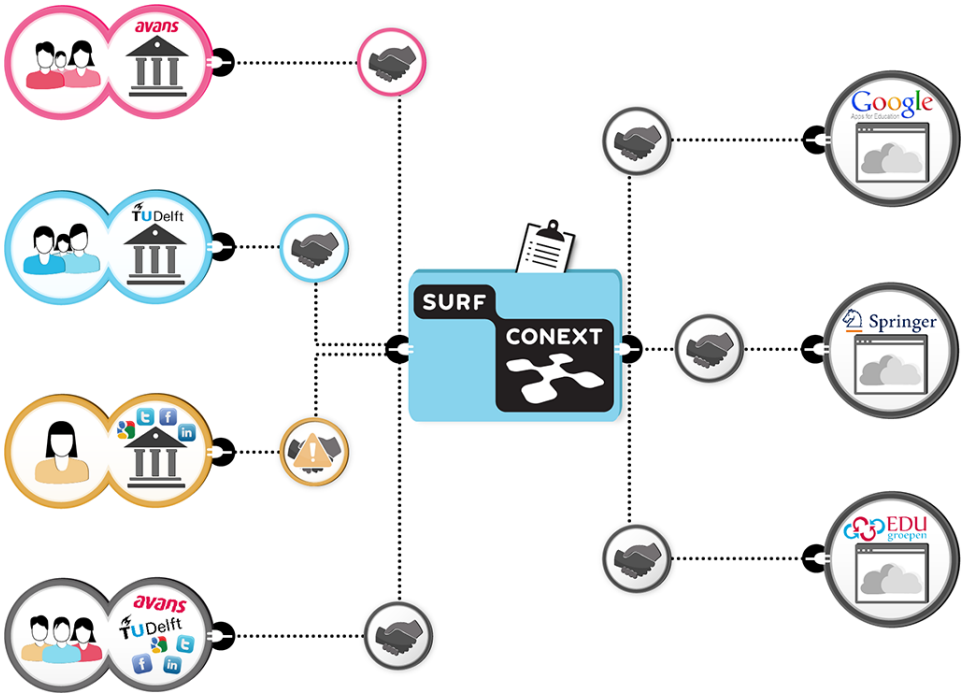


Figure 8: SURFconext - Schemetic overview [20]

#	**Percentage**	**Product**
114	65.9%	MS ADFS
32	18.5%	SimpleSAMLphp
13	7.5%	NetIQ/Novell AM
5	2.9%	Oracle AM/OpenSSO
3	1.7%	Shibboleth
2	1.2%	OpenASelect
1	0.6%	Onegini
1	0.6%	Mujina
1	0.6%	OpenConext
1	0.6%	CAS

Table 1: SURFconext - Identity Providers product overview (13 January 2017) [4]

4 Methodology

We started this project with interviewing Service Providers, to determine the expectation of the users, when they want to log out at the corresponding application.
Initially we intended to interview the end-users, however, due to time limitation it was impossible to interview enough end-users to get a global overview. With those in mind we chose to interview the Service Providers. The Service Providers represents in our research the end-users.[4]

The contacted Service Providers are:

- Labservant - Providing a "framework of laboratory safety as efficient as possible and to contribute to a healthier workplace with less effort". [6]
- Omnicard - Providing card solutions [13]
- SURFcumulus - Providing a "hybrid infrastructure-as-a-service (IaaS) service from SURFnet" [18]
- SURFdrive - Providing "personal cloud storage service for the Dutch education and research community" from SURFsara.[19]
- SURFspot - Providing a ICT webshop for students and staff from SURFmarket.[23]

The question to the Service Provider:

What do the users expect, when they press on the "logout" button of the corresponding application?

In parallel we started conducting a desk research. Based on the desk research and the expectation of the users we have determined possible solutions. Every possible solution will be provided with advantages and disadvantages. After validating the requirements with the advantages and disadvantages of the possible solution, one solution will be worked out further in technical detail. The most feasible solution will be experimented as described in section 5.3.

The expectation of the users, possible solutions and results of the experiments are discussed in section 5.

5 Results

5.1 User expectations

Based on the answers of the Service Providers:
All contacted Service Providers have the same view of what the users expect when they want to log out. The expectation is that they only log out at the corresponding application. All other application(s) in the session must stayed logged on. We have defined this as **Partial Logout**.

One Service Provider (SURFspot)[23] has suggested an extra feature. This feature needs to be a portal where users has the ability to control their authenticated session.

Another Service Provider (SURFcumulus)[18] suggested another extra feature. This feature must have the possibility that an authorized user can force a user to log out. For instance, in case of suspected behavior with the users session, or if a user was forgetting to log out on a public computer. There are other cases where this feature has benefits. This feature will due to time constraints not be researched in this project.

Example:
The figure 9 illustrates the expectation of the users. In this situation, the user has an authenticated session to at least two different Service Providers and wants to log out at the first Service Provider.

1. User initiates a log out request.
2. The Service Provider terminates the authenticated session and sends the logout request to SURFconext.
3. SURFconext checks for authenticated sessions known by SURFconext.
4. SURFconext sends the logout request to the Identity Provider.
5. The Identity Provider checks the authenticated session known by the Identity Provider and terminates the session.
6. The Identity Provider sends an acknowledgement to SURFconext.
7. SURFconext sends the acknowledgement to the Service Provider.

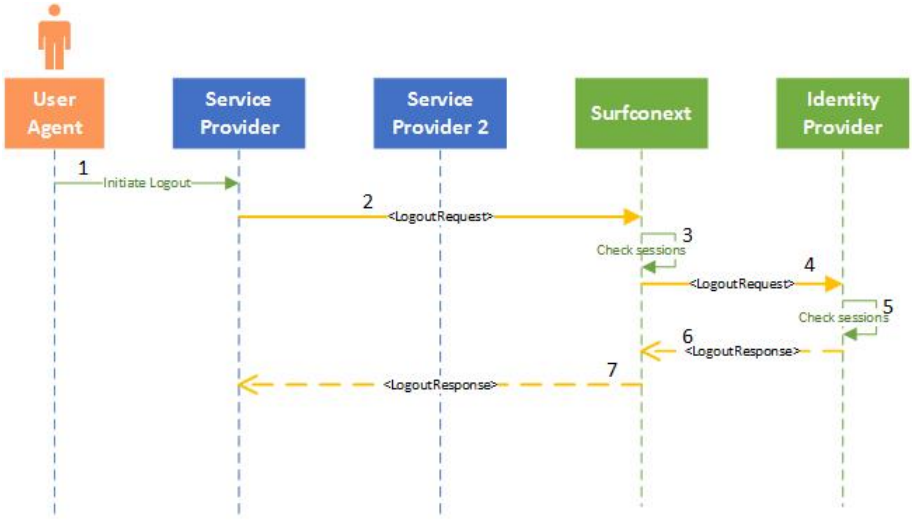


Figure 9: SURFconext - Log out expectation

5.2 Possible solutions

Based on the user's expectations and the possibilities in the SAML 2.0 protocol we conceived different possible solutions.

In a SAML federation there are at least three participants. The user agent, Service Provider and the Identity Provider. In a *hub and spoke model* there is an extra participant. The hub is a participant that acts as an Identity Provider to the Service Providers and acts as a Service Provider to the Identity Provider.

The solution can be sought by the four SAML participants. There are possible solutions for user agents by building a plugin that can handle the SAML logout request. The user agents (mostly browsers) differs too much and building or maintaining a plugin for all browser takes too much time. Therefore, we have decided to that seeking a solution by the user agent is not practical.

The possible solutions that we devised in this research are:

1. Disabling Single Sign On.
2. Defining a new protocol.
3. Using the Single Logout Protocol.
4. ForceAuthn option in the authentication request.

5.2.1 Disabling Single Sign On

The first possible solution is to disable Single Sign On, but keep up the federation trust between the Service Providers and the Identity Providers. This results in that users need to re-authenticate themselves, because every session with an another Service Providers needs to authenticate with the Identity Provider. To avoid that users needs to authenticate themselves every time they access another application, users can make use a password manager. Disabling Sign Sign On can be disabled by the Identity Providers itself, this is not a global configuration in SURFconext where all Identity Providers needs to conform to.

Advise: We advise SURFconext from security perspective to inform the Identity Providers about this possibility.

This possible solution has an effect; users have to authenticate themselves every time they access a new application. To avoid this, users can use a password manager. Each Identity Provider in principal can decide to disable Single Sign On.
Advantages:

- Users having awareness where they logged on.
- Users have one identity that they need to remember for accessing application.
- Partial Logout problem is solved.
- Possible to configure at the identity provider.

Disadvantages:

- The Single Sign On features are missing.
- Decreases users satisfaction or lowers the productivity.
- "Many Identity Providers have indicated to SURFconext that they think the Single Sign On feature is important to them."[4]
- Doesn't work for applications where Single Sign On is a vital necessity.

5.2.2 Defining a new protocol for Partial Logout

The second possible solution is to define a new protocol for Partial Logout. The new protocol can be developed to facilitate what users expect, when they want to log out. Defining a new protocol takes a long time to develop, test and implement and has the limitation that not all Service Providers and Identity Providers can implement non standard protocols. The new protocol can be developed outside or inside of SAML.

Advantages:

- Flexibility in a way how it should work.

Disadvantages:

- Design considerations.
- Implementation time.
- Implementation limits, not all Identity Providers and Service Providers can add new protocols to the SAML standard.

5.2.3 Using the Single Logout protocol for Partial Logout

The third possible solution is to implement the Single Logout protocol as defined by OASIS. The additional Reason attribute in the Logout Request, gives the opportunity to use this attribute for Partial Logout. The Identity Provider can decide based on the value in the Reason attribute, which action to take with the request. Based on the value in the reason attribute, the Identity Provider (SURFconext) can decide, which action to do with the request. There are two standard values defined in Single Logout protocol: [10]

urn:oasis:names:tc:SAML:2.0:logout:user user terminates session and initiates log out
urn:oasis:names:tc:SAML:2.0:logout:admin admin terminates session and initiates log out

The values of the reason attribute are only specified as an indicator.

We have defined that when a Service Provider provides the Reason attribute of the LogoutRequest with the value "urn:oasis:names:tc:SAML:2.0:logout:user" then the Identity Provider (SURFconext) needs to process the LogoutRequest as a Partial Logout. The Identity Provider(SURFconext needs to process all other values or non values in the Reason attributed need as a standard Signle Logout. Figures 10 and 11 illustrates the Logout process.

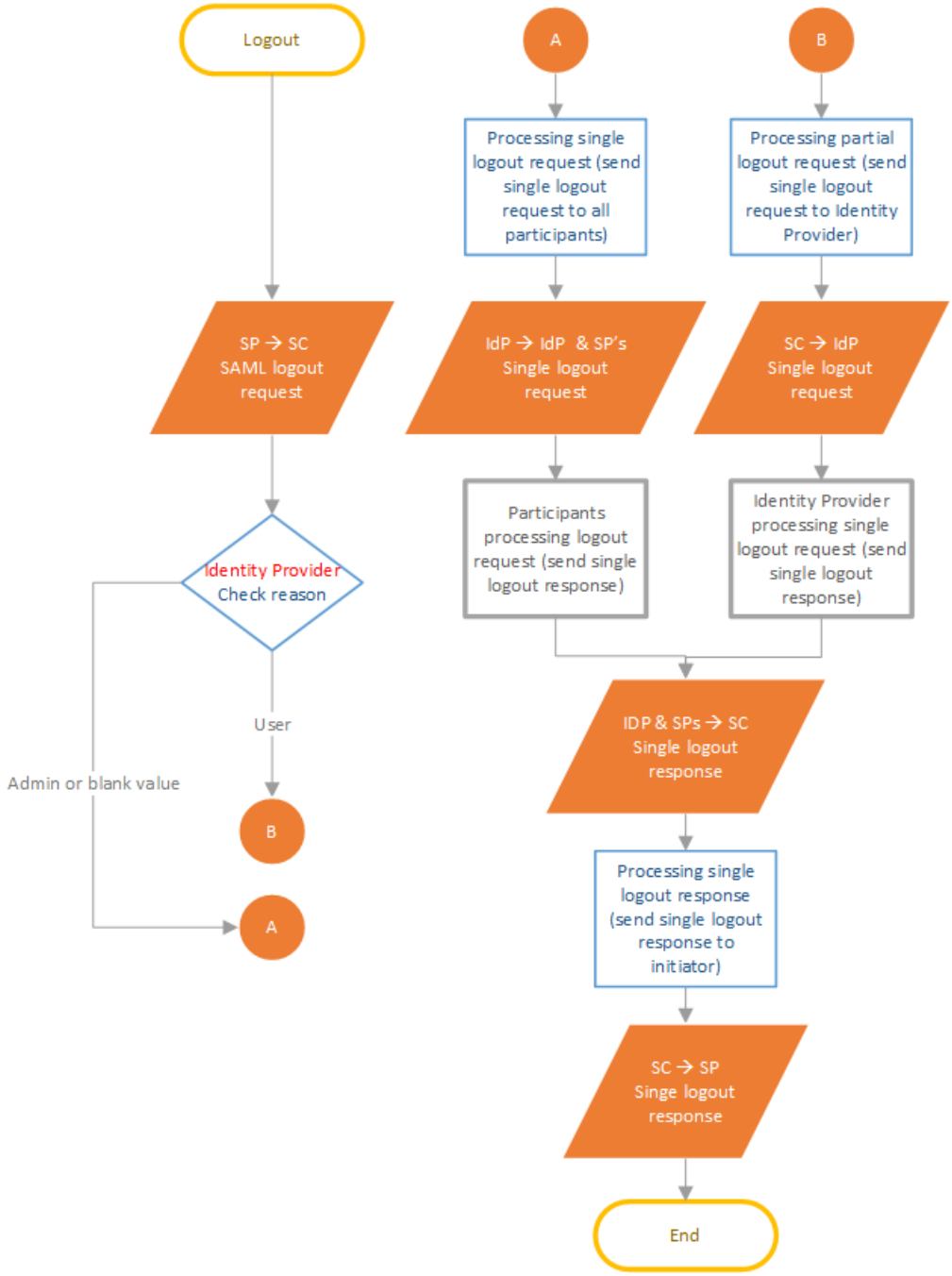


Figure 10: SURFconext - Logout process

Figure 11 illustrates a Partial Logout request:

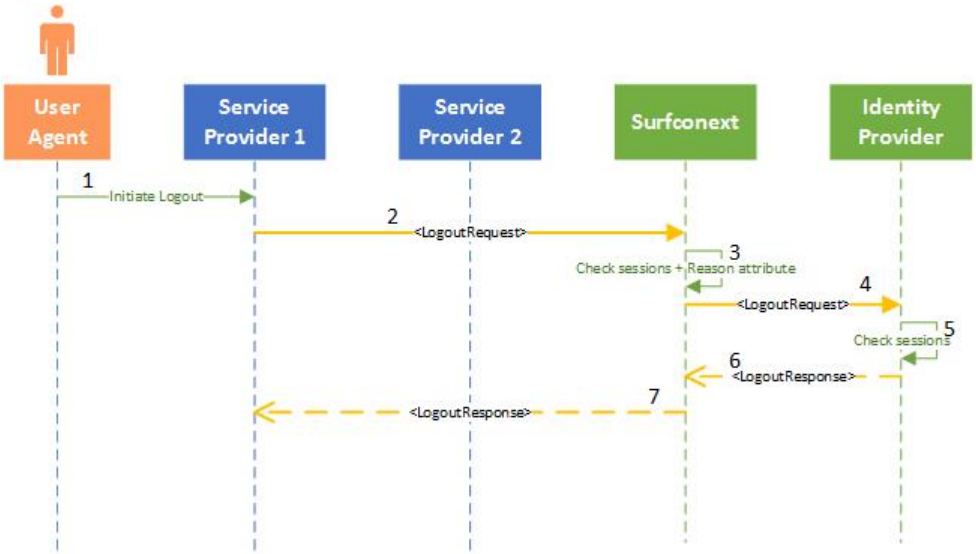


Figure 11: SURFconext - Partial Logout

1. The user initiates a log out by the Service Provider.
2. The Service Provider sends a logout request to the SURFconext.
3. SURFconext checks the value of the Reason attribute.
4. If the Reason is "urn:oasis:names:tc:SAML:2.0:logout:user", SURFconext sends a logout request to the Identity Provider of the user.
5. The Identity Provider determines the session at the Identity Provider and terminates the session.
6. The Identity Provider sends a logout response to SURFconext.
7. SURFconext sends a logout response back to the Service Provider.

If the user is partially logged out and wants to access an application that needs to authenticate against the Identity Provider, the user needs to authenticate again against the Identity Provider.
To keep control which users having authenticated sessions, SURFconext needs to record information about the sessionIDs and Service Providers. With this information, it is possible to terminate authenticated sessions by the Identity Provider and Service Providers. This gives the possibility to get a fully controlled federation.

Statuses
We have defined three possible statuses, where a session can state in. The statuses are specified in table 2. A session always starts without a status. The session will get the status Session_trusted after the SURFconext receives a successful authentication response from the Identity Provider.

Status	**Definition**
Session_trusted	Session of the user is fully trusted
Session_partial_trusted	Session is partially trusted
Session_untrusted	Session is not trusted

Table 2: SURFconext - Status definitions

Status Session_partial_trusted
When a user wants to log out at a single application, the user initiates a log out by the Service Provider. The Service Provider sends a logout request to SURFconext with the value "urn:oasis:names:tc:SAML:2.0:logout:user" in the Reason attribute. SURFconext sends a logout request to the Identity Provider. After SURFconext receives a successful logout response from the Identity Provider, SURFconext deletes the row of the Service Provider in the SESSION_SESSIONS table and changes the status in the SESSION_STATUS table. SURFconext sends a logout response back to the Service Provider, where the log out was initiated. All other applications which have a authenticated session keep working.

Status Session_untrusted
When a user wants to perform a Single logout, the user can go to "profile.surfconext.nl" and click on logout. The Service Provider "profile.surfconext.nl" sends a logout request to SURFconext. SURFconext sends a logout request to all participants (except profile.surfconext.nl) including the Identity Provider, if the status is Session_trusted. The participants send a logout response back to the SURFconext. If all participants have responded with a successful logout response, the responses will be deleted from the SESSION_SESSIONS table and the status will be changed in the SESSION_STATUS to Session_unstrusted.

Tables
To be able to store the data, SURFconext needs to create a database with two tables (see Figure 12). Table SESSION_STATUS stores the SESSIONID and status.
Table SESSION_SESSIONS stores the SESSIONID, SERVICEPROVIDER, SESSIONSTARTTIME and SESSIONENDTIME. The values can be deducted from the authentication requests and responses.

- SESSIONID is the cookie-id of the session
- SERVICEPROVIDER is the value of element Issuer in the authentication request
- SESSIONSTARTTIME will be the value of the attribute AuthnInstant in the AuthnStatement of the authentication response

- SESSIONENDTIME will be the value of the attribute SessionNotOnOrAfter in the AutnStatement of the authentication response

After each successful authentication a new row will be added in the second table and the status in the SESSIONID will updated based on the status of the user.

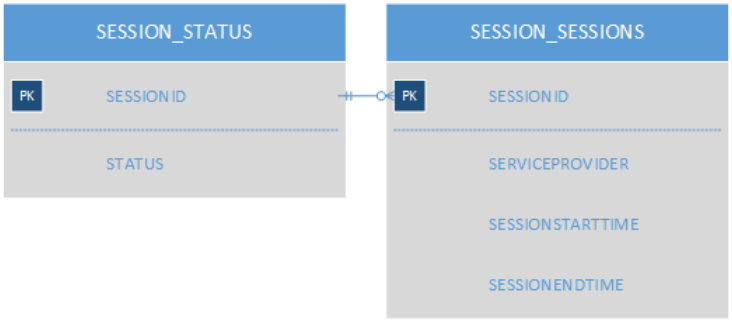


Figure 12: SURFconext - Database session tables

Figure 13 illustrates a visual overview of when a status changes.

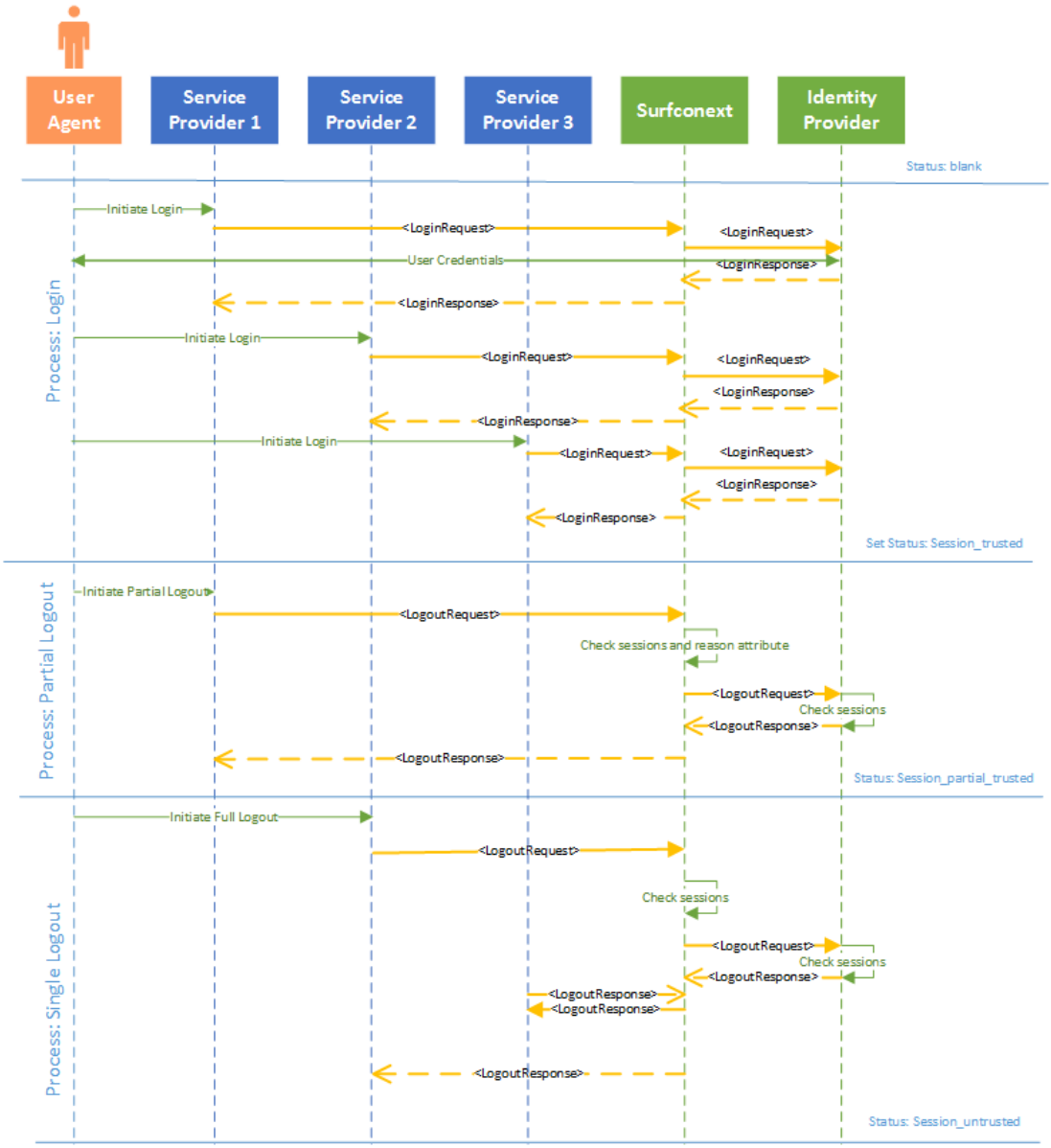


Figure 13: SURFconext - Session Statuses

Advantages:

- Flexibility: SURFconext can define what to do with a logout request, based on attributes in the logout request.
- Security: The Service Providers can trust SURFconext, because SURFconext is handling logout request.

- Implementation by the Service Providers is simplified, because Single Logout is a standard protocol in SAML 2.0.

Disadvantages:

- Service Providers need to be able to send logout requests with the Reason attribute.
- SURFconext needs to record data, the infrastructure of SURFconext needs to be examined and possible be extended.

5.2.4 ForceAuthn option in the authentication request

The fourth possible solution is making use of the ForceAuthn attribute in the authentication request. This attribute forces the users who wants to access an application to re-authenticate again against the Identity Provider. If all Service Providers set this option, it is better from security perspective to disable Single Sign On. The advantages and disadvantages of the first possible solution are the same as for this solution.

Advantages:

- Flexibility by the Service Providers. Service Providers can decide if they want to set this option in their authentication request.
- This option is a standard feature of the authentication request. The Identity Providers support this by default, if they follow the standard.
- Users are forced to log in again. Users do not need to logout from the Identity Provider, because the Service Providers are using this option and handle the logout.

Disadvantages:

- Security: If the Identity Provider accepts authentication requests that aren't signed, there is a possibility to send a SAML request on behalf of a Service Provider without the ForceAuthn element in the request to access the application. SURFconext currently does not require signed authentication requests. Therefore, we consider the current implementation of ForceAuthn in SURFconext to be insecure. [4]

We advise SURFconext to instruct all Service Providers to sign their authentication request.

5.3 Experiments & Results

SimpleSAMLphp is an application led by UNINETT, which deals with authentication. Although the main focus of SimpleSAMLphp is providing support for SAML 2.0 as Service Provider and/or Identity Service. SimpleSAMLphp also supports other identity protocols and frameworks. SimpleSAMLphp is a model based application, which gives users the ability to extend the application with own written modules.[16]

"UNINETT develops and operates the Norwegian national research and education network, interconnecting about 200 Norwegian educational and research institutions and more than 300 000 users, as well as giving them access to international research networks. We are a neutral party, and the business is run non-profit." [24]

SURFconext uses the libraries of SimpleSAMLphp.

The experiments are divided in two parts (See Figure 14).

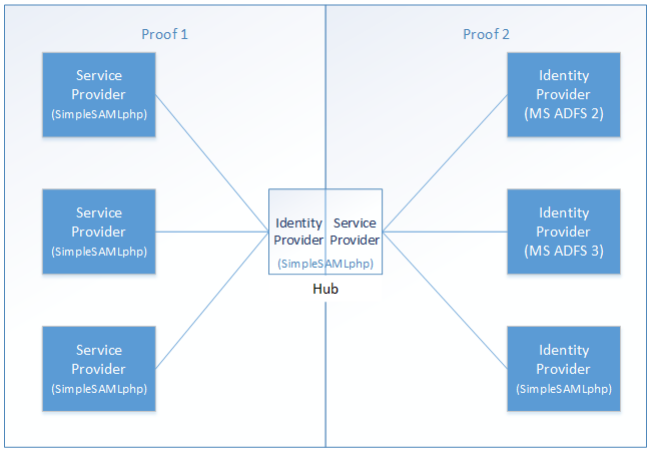


Figure 14: SURFconext - Experiment

The experiment environment of experiment 1 and 2 consists of three Service Providers and one Identity Provider. All the Service Providers as the Identity Providers making use of SimpleSAMLphp applications on a Ubuntu 16.04 installation. The user have in experiment 1 and 2 an authenticated session against the Identity Provider and all three Service Providers.

5.3.1 Experiment 1 Single Logout

Sending a Single Logout request to observe what the behavior is of the Service Providers and the Identity Provider.

Results
The result of the experiment is that the authenticated session terminated by all the Service Providers and Identity Provider. When the users wanted to access a new application (Service Provider), the user needed to provide his credentials to the Identity Provider before getting access to the application.

5.3.2 Experiment 2 Partial Logout

Sending a Partial Logout request as described in section 5.2.3 to observe what the behavior is of the Service Providers and the Identity Provider.

Results
The result of performing a Partial Logout, as described in section 5.2.3 resulted in that the user logged out only by the application (Service Provider) where the partially logout was initiated and the Identity Provider. The sessions by the other applications (Service Providers) remained authenticated. When the users wanted to access a new application (Service Provider), the user needed to provide his credentials to the Identity Provider before getting access to the application.

5.3.3 Experiments 3 between the Identity Provider and SURFconext

The third part consist of one Service Provider and three Identity Providers. The Service Provider and one Identity Providers are SimpleSAMLphp applications on a Ubuntu 16.04 installation. One Identity Provider make use of ADFS version 2 installed on a Windows 2008 R2 installation. The last Identity Provider make use of ADFS version 3 installed on a Windows 2012 R2 installation.

For this part we create three situations:

1. A user has an authenticated session to a SimpleSAMLphp Identity Provider and the Service Provider.
2. A user has an authenticated session to an ADFS version 2 Identity Provider and the Service Provider.
3. A user has an authenticated session to an ADFS version 3 Identity Provider and the Service Provider.

In all three situations we will perform a Single Logout request to observe what the behavior is of the Service Provider and the Identity Providers.

Results
All the experiments had the same results. The authenticated session of the user were terminated by the Identity Providers and by the SURFconext emulator.

6 Conclusion

During this research project, we have researched the possibilities of federated log out. One aim of this research was to define the user's expectation of federated log out. The end users define federated logout as Partial Logout. Partial logout is a concept whereby a user terminates the authenticated session of the corresponding application and not every authenticated sessions. The second aim to research possible solutions in an SAML 2.0 *hub and spoke model* of SURFconext. Based on the desk research we have defined four possible solutions; disabling Single Sign, defining new protocol for Partial Logout, using the Single Logout protocal for Partial Logout and using the ForceAuthn option in the authentication request. The most feasible solution for Partial Logout as what users expected is; to implement the "Single Logout protocol with the additional reason attribute" as described in section 5.2.3. The last aim of this research is to prove that the most feasible solution can work in a SAML 2.0 *hub and spoke model*. The experiments proved that the solution works in a *hub and spoke model*. The solutions can be implemented in phases and can work for all participants which implemented the Single Logout protocol. SURFconext needs to record session information.

7 Discussion

During this research we faced that the reason attribute is not used in by default. MS ADFS ignores the Consent, Destination, NotOnOrAfter and Reason attributes by default.[8] This means that the solution can only work for Service Providers that are able to extend the logout request with the reason attribute.
SURFconext needs to store session information. With the minimum information as described in Section 5.2.3 it 's not possible to trace a person. With the session information it's not possible to trace a person.
During the experiments, we used SimpleSAMLphp to build a proof of concept, because we had to less experience in PHP, we split the environment into two parts.
If the user has an authenticated session and access an application that doesn't have an log in button, the user will automatically (mostly without knowing) be logged in. The user can only logout by application that provide a logout button. We recommend the Service Provider to provide their applications with log in and log out functionality. In addition to turn off Single Sign On and use the reason attribute we have thought about using the ForceAuthn attribute, but this element requires again for user credentials, but it does not close the session and this is also undesirable.

8 Future Work

During our research we also thought about the additional features suggested by the Service Providers SURFcumulus and SURFspot and the following things can be done in the future.

- There are two types of bindings and these bindings are the front-channel and the back-channel. With the front-channel binding the users will be directed to another website. With the back-channel the user remains on the website and the Service Provider or Identity Provider will handle the request and response. Researching which type of binding is preferred for Partial Logout that is initiated from another Service Provider (for example a portal with the session overviews) in a *hub and spoke model*.

- In situations where a user wants to perform a Single Logout and one Service Provider did not respond. The authenticated session is terminated by the Identity Provider. The other Service Provider did not receive a logout request and their sessions are still authenticated. To increase the confidentiality in an federation, a research can be done to which opportunities there are for Service Providers to check if the token is still valid with the Identity Provider.

- Investigate what possibilities the new OpenID Connect protocol, which SURFconext starts to support in 2017, provides for implementing log out.

9 Acknowledgment

We want to express our gratitude to our supervisors Thijs Kinkhorst and Joost van Dijk. We also want to thank the Service Providers we interviewed for their time and Dick Visser (GÉANT) for sharing his knowledge about SimpleSAMLphp.

References

[1] Sneha Thakkar Dwiti Pandya, Khushboo Ram Narayan. An overview of various authentication methods and protocols. http://www.ijcaonline.org/research/volume131/number9/pandya-2015-ijca-907389.pdf, 2015.

[2] Carol Geyer. History of saml. http://saml.xml.org/history, 2007.

[3] Nicole Harris. Refeds survey. https://refeds.org/a/1561, 2016.

[4] Thijs Kinkhorst and Joost van Dijk. As told to us by the surfconext team. 2017.

[5] Sherif Koussa. Federated identities: Openid vs saml vs oauth. https://softwaresecured.com/federated-identities-openid-vs-saml-vs-oauth/, 2013.

[6] labservant. The lab servant. http://www.labservant.nl, 2017.

[7] Paul Madson. Liberty alliance project white paper: Liberty id-wsf people service - federated social identity. http://www.projectliberty.org/liberty/content/download/387/2720/file/Liberty_Federated_Social_Identity.pdf, 2005.

[8] Microsoft. Single sign-out saml protocol. https://docs.microsoft.com/en-us/azure/active-directory/develop/active-directory-single-sign-out-protocol-reference, 2017.

[9] OASIS. Assertions and protocols for the oasis security assertion markup language (saml) v2.0. http://docs.oasis-open.org/security/saml/v2.0/saml-core-2.0-os.pdf, 2005.

[10] OASIS. Saml v2.0 assertions and protocols. https://docs.oasis-open.org/security/saml/v2.0/saml-core-2.0-os.pdf, 2005.

[11] OASIS. Saml v2.0 profiles. https://docs.oasis-open.org/security/saml/v2.0/saml-profiles-2.0-os.pdf, 2005.

[12] OASIS. Saml v2.0 technical overview. https://docs.oasis-open.org/security/saml/Post2.0/sstc-saml-tech-overview-2.0.html, 2008.

[13] Omnicard. Over omnicard. http://www.omnicard.nl/over-omnicard/, 2017.

[14] Joonas Ruuskanen Sanna Suoranta, Asko Tontti and Tuomas Aura. Logout in single sign-on systems. https://pdfs.semanticscholar.org/e4b0/20decd4e522f1390a89b73f6449f6766e0b1.pdf, 2015.

[15] Shibboleth. Sloissues. https://wiki.shibboleth.net/confluence/display/CONCEPT/SLOIssues, 2015.

[16] SimpleSAMLphp. Simplesamlphp. https://www.simplesamlphp.org, 2016.

[17] SURFconext. Over surfconext. https://www.surf.nl/en/services-and-products/surfconext/index.html, 2017.

[18] SURFcumulus. Over surfcumulus. https://www.surf.nl/en/services-and-products/surfcumulus/index.html, 2017.

[19] SURFdrive. Over surfdrive. https://www.surf.nl/en/services-and-products/surfdrive/surfdrive.html, 2017.

[20] SURFnet. Documentation of service providers. https://wiki.surfnet.nl/display/surfconextdev/Documentation+for+Service+Providers, 2017.

[21] SURFnet. Faq service provider. https://wiki.surfnet.nl/display/surfconextdev/FAQ+SP#FAQSP-DoesSURFconext.supportSingleLogout?, 2017.

[22] SURFnet. statistic overview. https://stats.surfconext.nl/, 2017.

[23] SURFspot. Over surfspot. https://www.surf.nl/en/services-and-products/surfspot/index.html, 2017.

[24] Uninett. Uninett. https://www.uninett.no, 2016.