Evaluation of different Open Source Identity management Systems

Ghasan Bhatti, Syed Yasir Imtiaz
Linkoping's universitetet, Sweden
[ghabh683, syeim642]@student.liu.se

1. Abstract

Identity management systems are the tools to secure digital identity of the users in consistent, automated and interoperable way. There are many Open source identity management solutions which are being used by many organizations to fulfill their need of securing and managing the identity of their users. In this document we will discuss the different open source identity management systems and will compare and evaluate their features, services which these systems provide to their users.

2. Introduction

Digital Identity is the record or profile of an individual stored in an electronic format which is being managed as a single unit in an identity system. The identity management system aims at protecting the digital identity of the users in a way to provide security, consistency, privacy and interoperability.

Microsoft dot net passport which is the proprietary of Microsoft is a famous identity management system which is being used by every user of Microsoft. Besides this there are many open source identity management systems which are being used in many organizations fulfilling the needs for the privacy of their data and management of their accountability and resource management for the users of the respective organization. Here we are going to describe and evaluate few of them.

The Identity management systems which we are going to evaluate are described as under:

1. Open Web SSO
2. Open Privacy
3. Shibboleth
4. Bandit

3. Open Web SSO

The main purpose of Open Web SSO is to present such an infrastructure for the identity services that will be used to make it possible of using Single Sign-On for the Web applications. [1]

3.1 Basic Architecture:

We will describe the basic architecture of an Open Web SSO by using the Use case diagram. In this diagram the User can be End User who has no administrative roles, or can be an administrator who has administrative roles; <<system>> web application is responsible for hosting business application and <<system>> SSO agent is responsible for validating user sessions. [2]

Figure 1: Setup with SSO Agent

3.2 Scenarios:

1. Access Web Application

If a user wants to access a web application then there will be following different steps.

- User accesses a web application.
- Web application receives the request.
- Web application validates user's session. This is either done by web application itself, or by the agent of which web application extends from.

2. Request to Authenticate

In this case a user attempts to get authenticated by using these steps. His authentication will be failed if he would enter wrong user id or password.

- User attempts to authenticate.
- Web application receives the request.
- Web application authenticates user.

3. Request to Logout

In this case a user attempts to logout by using these steps:

- User attempts to logout.
- Web application receives the request.
- Web application destroys user's session.

4. Manage User Sessions

An administrator views active user sessions, reads session attributes such as idle and maximum session time, and destroys sessions.

- Administrator requests to view or destroy user sessions.
- Web application receives the request.
- Web application returns session information or destroys session.

5. Manage Configuration

An administrator views or modifies configuration such as User's maximum session time, authentication configuration etc.

- User requests to view or modify configuration.
- Web application receives the request.
- Web application returns configuration information or modifies configuration.

Like these different scenarios there are others for Creation and Validation of User Session to give user access to web applications within the scope of the current operation. Similarly there are other cases where the web application sets the session property, reads service configuration and destroys the user session.

3.3 Identity services

In Open Web SSO architecture the core identity services are authentication service, session service, and logging service. For authentication of user, the identity service application first develops a trust and then the trusted user will then be considered authorize to have single or multiple interactions depending on the application. These applications then keep the record of all the activities like if the user is successfully authenticated then it will be useful to determine how different users have accessed different applications.

a. Authentication Service:

The purpose of Authentication Service is to validate the identification of certain clients or administrators to access the protected information or resources. Upon completion of the authentication process, the Authentication service will give access or deny access for the use of different applications. So we can say that Authentication service is the gate of entry in the SSO infrastructure. In the Open Web SSO system, the authentication service can be accessed through a web browser. So if a user wants to gain access to protected information he will give his identification and after that he will get the authority to use the requested resource or service.

b. Session Service

After successful identification of a user, a user session is created that is used to establish a Single Sign On architecture by authenticating the user only once and providing access to multiple resources so the user is not required to provide his identification details again and again in using multiple services. A user session starts when a user logs in and ends when he logs off. In this way until a user is logged in he will be able to access more than one application in this session without having to re-authenticate himself. [2]

A user session is destroyed after certain conditions are met including maximum allowed time for user session, maximum allowed time for idle or unused user session, an action such as logout or destroy issued by the user or administrator and other conditions arising from the environment as applicable.

The key states are valid and invalid. In the valid state, the session represents an authenticated user session. In the invalid state, it could represent an anonymous session, or a user session that has expired.

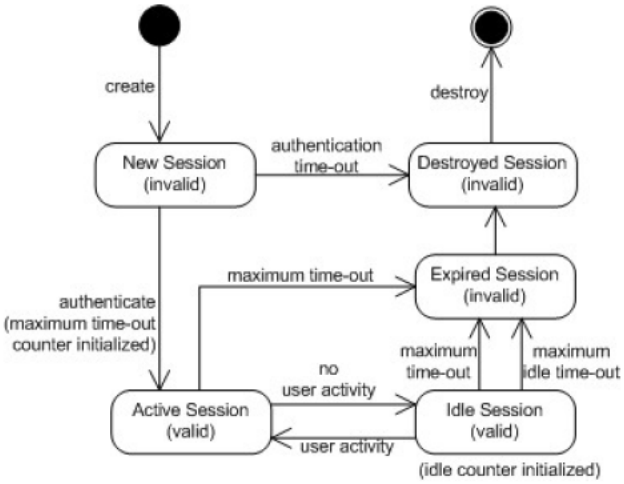

Figure 2: Session service

c. Logging Service

During a user session a proper log is maintain to monitor the activities of the user. This is done by the *Logging Service* of the Open Web SSO architecture. This log is maintain when the user logs in and till he logs out.

3.4 Sequence of Operations

We will now discuss different steps that will be helpful to determine the identity of a user in the deployment of OpenSSO: [2]

- When a user tries to access a web application through SSO agent then the web application will check for the existence of the session cookie.
- If it founds a session cookie, then it will be validated by the SSO agent using remote session service client facilities. If it founds the session to be valid, the SSO agent will allow the request to proceed to the desired resource.
- If it does not find the session cookie, a client side redirect is issued by the SSO agent to the authentication service to redirect the user back to the originally requested address once authentication is complete.
- Then the user is validated by checking his identification. If he is identified, the authentication service will create a new user session, and set the user session handle as a domain cookie before redirecting the user back to the originally requested address.
- The same process of user verification can be repeated if the user is back to the originally requested. Since this time the user has a valid session handle, the SSO agent will allow the request to be preceded to the requested resource.

4. Open Privacy

4.1 Concepts

The Open Privacy is a system that that protects the privacy of its Users in secure way and the focus is to create an open system, meaning not only just an open source or with published APIs but also providing a mechanism to interact and communicate amongst the users freely and providing open access to all. [4]

There are some signed objects created in a huge number of pseudonymous entities maintained by the user. These objects are Opinions, Bias and reputations and by combination of these objects Personal profile is created in such a way that only the owner of the information can validate the connections between them. Other entities like marketers and online community can also access them only if they have given the rights of doing so.

Generally, different corporations and government companies gather personal information from the users like his place to live, place to work, habits, favorite movies, magazines they read etc and then provide the desired services. The main problem is that once the user has provided all this stuff, it would not be in his control any more and all this information can be transferred to many places. There are some strong laws present that try to prevent this misuse of personal information but still a customer has to trust on those entities to which he has given his personal information.

Therefore there is always a need of secrecy of an individual's personal information and no one is allowed to monitor a person's location, purchases and other activities. Some systems like Anonymizer and Freedom try to protect one from being monitored by someone else but the problem with these systems is that they fail to create and profit from a long lived pseudonymous identity. The main issue here is that if a person is been able to develop a good reputation on a certain site still can not be able to carry this reputation to some other site. Moreover all the information provided to the creators of these sites will be under control of them and they can use that wherever they want. With OpenPrivacy a user can hide him from others and expose some of his information to others without loosing his

privacy because it uses n architecture for maintaining systems that can intercommunicate with each other with the support of Opinion accumulation and reputation. [4]

4.2 Reputation Services

Reputation services maintain a standard opinion and reputation architecture and it can be used by any community. A reputation service should include this list of steps: [8]

First, it should identify the things we can trust, not just about people, but people are an important subset.

Similarly identify things we can't trust.

Should include the user feedback and the other relevant data

Should allow users to set their own standards about what is good and what is bad because every user has own perspective.

Should provide extensibility so that a user can take someone else's reputation service and add it to own layer of reputation service on top of it.

4.3 Longer Term Goals

The goal of OpenPrivacy is that this platform benefits the users without the loss of their privacy and with the ability to provide a standard framework (objects and protocols) for general purpose reputation management. User can enjoy the benefits of personalized information without loss of privacy and can browse different offers from businesses and marketers. He can have full confidence that his profile is being used only as permitted and no one else can have any benefits from it. A user can respond to offers directly or through third party infomediaries. [4]
Shibboleth is the internet2 architecture, framework for the sharing of resources at remote sites. It is an open.-source standardised mechanism developed by Middleware committee for education of internet2.

5. Shibboleth

"The Shibboleth software implements the **OASIS SAML v1.1**specification, providing a federated Single Sign-On and attribute exchange framework" [10]. Shibboleth allows the access of the information at remote sites by authorizing the user using their login and attribute information at their home sites. Shibboleth provides the same security as provided by Login but it provide on the group basis rather than the specific user but it preserve the privacy of the user as well [9].

e.g. Linkoping university at Norkoping would like to share its data and other resources with the authorized users at the Linkoping campus. Suppose the users are the member of research group of IDA and they need the specific resources at Norkoping campus, then they will login in with their LIU ID and then the shibboleth framework at Norkoping campus require only their attributes that they are from research group of IDA to give them access to the resources at Norkoping campus. In this way the identity of the user is not revealed, only his identity as researcher of the IDA department of the Linkoping University will be revealed to the shibboleth system at the Norkoping campus.

5.1 Shibboleth Concepts

The key concept of the shibboleth is that the authentication and authorization perform solely to access the use the specific resources. i.e. A user will be authorized based on the attributes send by the home institution which has authenticated that user at the home site.

In Shibboleth Architecture we have to kinds of trust. One is the collaborative trust, in which two institutions agree upon exchanging the attribute information and based on this information the access to the resources will be granted [9]. This trust can be established through negotiation and collaboration. That," what attribute information will require for sharing, and what values these attributes will take" [9].

There are also hierarchal trust networks and certification authorities. This ensures that the exchanges of shibboleth messages are between those who should be. In shibboleth institutions trust on the authentication of the other institutions and also trust that the attribute exchange between the parties are not forged.

5.2 Shibboleth Architecture

The architecture of shibboleth consists of

a. Service provider

This is the host of the resource which takes the decision whether the user should be granted the access to the resource or not. It includes assertion customer service, and attribute requestor, and a resource manager in addition to the resources which it has to provide the user.

b. Identity provider
The users who want to access the resources must be registered to some identity provider. It also provides an infrastructure or framework to authenticate the users. It authenticates the registered users based on their information which is stored in Lightweight Directory Access Protocol directory. It contains the two components of the software named Handle Serve and Attribute authority.

c. WAYF (Where Are You From) server
This server executes the WAYF service. This is the central service which resolve the query, "to which identity provider, the user is registered with".

5.3 Shibboleth Authorization process

The authorization process of the shibboleth is shown in the figure below [9]:

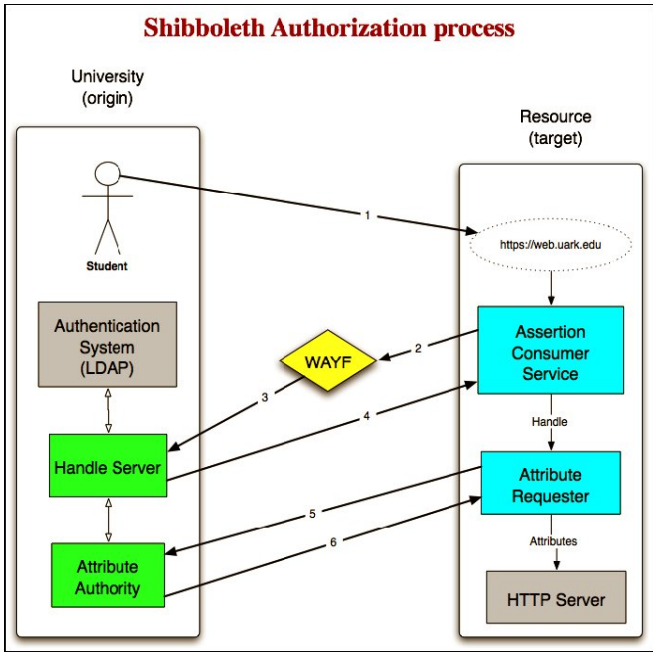

Figure 3: Shibboleth Authorization process

The steps of the Shibboleth protocol can be described as follows [9]:

1) A user request for a resource at a remote site.
2) The shibboleth assertion customer service respond the request
3) Then forward this request to WAYF server which asks the for the home institution of the user and forward the request to the home institution which asked for the username and password for the user , and after authentication handle server generate a "handle" for that user
4) The Handle server then forwards this handle to the assertion customer service at remote site.
5) The handle does not represent the user background or to which group it belongs, hence he attribute requestor asks for the attributes of that user from the home institution of the user against that handle.
6) The attribute authority then returns the attribute of that user to the resource site, then on the basis of these attributes decision of granting or denying access to the resources is made.

6. Bandit

"Bandit is a set of loosely-coupled components that provides identity services for authentication, authorization and auditing in a consistent way" [11]. These components provide the environment of access to the Systems for the authorized persons to the right place at the right time. It provides the framework that identity services for user can be accessed from the more than one identity stores. Bandit project enable the user to provide the information they want to share i.e. providing the user-centric credentials. The key area of the Bandit Project is to provide application access to multiple identity stores, support consistent application access using multiple authentication methods, provides the application interface to provide access based on roles [11].

6.1 Architecture of Bandit

The architecture of bandit consists of following components.

6.1.1 Common Identity Architecture

This component provides the abstract data model and the interface to access the different identity stores. While dealing with open identity system, we make, compare and authenticate the identities and as they not share any common protocol or data so there is a need to provide the common model or framework to deal with them. Higgins project in accordance with the bandit project provide the users to integrate the user identity, user profile and there relationships on different domains.

This component is further divided into two parts:

a. Identity store interface

It consist of a JNDI (java name and directory service) service provider which do mapping and joining with other JNDI service providers which then communicate with identity stores to get the identity of the user[12].

b. Identity Store Connectors

Here we create JNDI context which perform requested action against the data store [12].

6.1.2 Common Authentication Service Adopter (CASA)

This is an open source component which provide infrastructure to store the credentials for authentication and the purpose of Single sign-on, which is used by users and application etc [13].

Its feature includes scalability and fault tolerant. It support cross-platform i.e. window and Linux, It also supports multiple authentication schemes, sharing and linking of credentials among different identity stores.

6.1.3 Audit record framework

This component implements the open group (Distributing Auditing System) XDAS specification, which provides several features like common audit record format, standardized format for the classification of events, records [14]. The events includes account management events, user session events, data item and resource element management events, service or application management events, service and application management events, peer association management events, data item or resource element content access events, exceptional events, audit service management events.

6.1.4 Role Engine

This component creates and provides the role assigned to a specific identity holder. This component can be called or fit into any application thus provide the role authorization and role information to the client.

6.1.5 Identity selector Service

This component allows users to interact users directly with the info-cards websites. The benefits of this service is that we don't need to change the identity information at many places, users can update it at one place., also user can interact with websites without disclosing their identity[16].

6.2 Bandit Authentication process

1) First the user is logged in and is authenticated from. the **LDAP** server.
2) Then the CASA server gets the user credential and stores it for the purpose of Single Sign- on.
3) The user then tries to access the resources of the party (e.g. browse the website), the other party then request the user log in and provides it credentials. The CASA then asks the user to provide two types of information
   3.1) which identity the user want to use (If user holds the multiple identities).
   3.2) It prompts the user to share its information with that party (Like user can share only name, not its address or credit card information etc).
4) The browser at client side then use the CASA via HIGGINS framework (It is a framework which makes The user to integrate the user profile and identity information across multiple systems i.e. provide the Identity abstraction) and get the information which user has allowed in step 3.2.
5) The other party then calls the role engine to access the roles against the identity which user has provided.
6) The role engine then calls the Common identity component of the Bandit architecture and requests the information to build the role list of an identity and is provided to the other party.
7) The other party then compare these role lists with its own policies against these roles in its store.

7. Evaluation Criteria

We are evaluating these four open source identity management solutions on the basis of following criteria.

Security:

How securely the identity information flows around the identity management system.

Scalable:

How scalable the system is regarding managing the identity information.

Auditing:

For which events the identity management system performs auditing or maintains compliance data.

Attack resistance:

What kind of threats of attack the system have and how it copes with these attacks.

Single Sign-on:

Whether the system is Single sign-on or not.

Privacy:

How it preserves the user credentials or user data.

Simplicity:

How simply it provides access to the user i.e. how much it is user friendly.

8. Method of evaluation

We performed the evaluation on the basis of data which we got from the internet, and then evaluate this data on the basis of our evaluation criteria. We also performed the evaluation on the basis of the knowledge we gained from that data.

9. Evaluation and Comparison

9.1 Open Web SSO

Security

The Authentication Service of Open Web SSO provides mechanisms to secure the system from password hackers and from frequent intrusion attacks. [3]. to have more security, the user information associated with authentication and user sessions is handled in such a way that only privileged administrators are allowed to have access. The session service guarantees that no user session information is disclosed in the communication between the session service and other external components.

Scalability

The Open Web SSO system has the feature to scale up to the necessary levels in order to include more and more web application as well as more and more users. Therefore there is not any limit to the number of web applications that can participate in SSO or any limit to the number of users within this system.

Auditing

The Authentication Service of Open Web SSO logs all its activity events informing user interaction and erroneous conditions, in order to provide the reporting facilities to OpenSSO system applications and users.

Attack Resistance

The Authentication Service of Open Web SSO protects against denial of service attacks. The Session Service guarantees that no user session information is disclosed during communication between the session service and other external mechanisms.

Single Sign On

As the name suggests, the Open Web SSO is a Single Sign on system.

Simplicity

The Open Web SSO does not place any restriction on the use of any specific underlying network technology, Computer hardware, operating systems, programming languages or other hardware or software entities thus making it simpler and heterogeneous.

Privacy

The Open Web SSO provides full privacy for the user information. All of the information is protected from any outside attack or hacking. Only the privileged administrators are allowed to access the information but still the can’t view the private information of the users.

9.2 Open Privacy

Security

It uses strong cryptography for authentication of active entities as well as data privacy and security because by using encryption, only authorized entities will be able to access to a user’s profile. The user is able to see how, where, when and by whom their profile information is being used.

Scalability

Just like other systems, Open Privacy also has the capability to Scale up to necessary levels by accepting more and more users at a time and having a good indexing service for the retrieval of user's profiles.

Auditing

Open Privacy also maintains a log service by monitoring all the activities of a user so that by using it a user could be audited for his actions.

Attack Resistance

The Open Privacy provides the features of protection against Denial of Service, spoofing, replay and flooding attacks. [4]

Single Sign On

Open Privacy is not single Sign on.

Simplicity

The interface is very user friendly and therefore it is quite simpler to use with not much hardware and software requirements.

Privacy

The main goal of Open Privacy is to provide privacy to the users. So the user logs in autonomously and his information is not seen by anyone. User can enjoy the benefits of personalized information without loss of privacy and can browse different offers from businesses and marketers

9.3 Shibboleth

Security

The two main components of the shibboleth, identity provider (idP) and service provider (sP) plays important role in providing the secure access to the resources. However there security also depends on the collaborative trust between them. i.e. the service provider is providing the resources to the authorized identity holders. And identity holders have been given the right resources.

Scalability

The focus of the shibboleth is also on providing the scalability. However it can achieve the scalability by Load-balancing i.e. using the same configuration for identity providers(idP) and service providers(sP) on multiple machines.

Auditing

Shibboleth performs the auditing when creating a SSO session, applying policies on privacy of the attributes and when maintaining the users attributes [19].

Attack resistance

Shibboleth proves the infrastructure to protect the privacy of the user. It makes the user to choose the information i.e. which attribute the user wants to share with the specific destination. Shibboleth has been carefully designed to protect the attributes of the user while in transit. All the hosts which are involved in the communication are authenticated and the transactions which are vulnerable to any threat are done by using the secure channels.

Single Sign On

Shibboleth provides the single sign-on feature.

Simplicity

Its property of single sign-on provides simplicity to user by authenticating only once and accessing the resources at multiple sites.

Privacy

In shibboleth the Privacy of user is also preserved. Shibboleth protects the resources using the group membership or user credential, rather than the identity of the user. In this way the identity of the user is not revealed only the group to which the user belongs.

9.4 Bandit

Security

The CASA provides the Secure Token service for the exchange of identity information. These token are created by the identity providers. These token could be used as a substitute to the usernames and password which are being passed between the applications. First the CASA authenticate the user username and passwords, after that it would use these tokens for authentication.

Scalability

The CASA component of the bandit which performs the authentication provide the feature of scalability. However the developers of the bandit are focusing on providing the improved scalability.

Auditing

Bandit provides the framework to the application t provide the audit and compliance data of the identities. This framework provides the mechanism for the collection and reporting of compliance data ensuring that application has provided the correct information at the correct time.

Attack resistance

The exchange of information between different components of the bandit is through secure channels to avoid attacks. However the research work is going on in bandit project to prevent attacks like hijacking of authorized sessions, denial of service attacks, extracting the confidential information.

Single Sign On

CASA (Common authentication services adaptor), a component provides the single sign-on infrastructure

Simplicity

Bandit provides the simple interface to access the identities from different identity stores and also the access based on the roles.

Privacy

In bandit, it depends on the user whether he wants to reveal information or not to the relying party. If user want to reveal its information to the relying the party, then it would be transferred to that party.

10. Conclusion:

After analyzing the four identity management solutions, we have concluded that the best feature of these systems is the Single Sign on which provides flexibility to the users thus avoiding users to enter the username and password at every site. These systems provide the framework to access the sources from the remote sites in a secure manner. With the traditional systems after authentication, when users used to access any system their identity was revealed to the sites they used to visit thus there was no privacy for the users. The identity management systems that we analyzed provide the framework to maintain the privacy of the users and also the special mechanisms to prevent the users from different attacks, like denial of service etc.

Although the basic functionality of all of these identity management systems is to manage the identity of the users but each of them has its own features making it useful in different scenarios. Like Open Web SSO is used in the web applications where proper authentication and session handling is required. Open Privacy was developed with the purpose of providing privacy to the users so that user can enjoy different web services like online shopping and online banking without loosing their privacy. Shibboleth provides infrastructure to access the resources at remote sites without revealing the details of the user, the remote sites only have information about the group to which a user belongs. Bandit Project provides application access to multiple identity stores and to support consistent application access using multiple authentication methods.

Thus we conclude that open source identity management systems are really gaining the popularity and because of that there are several latest identity management systems emerging these days. The core feature of these identity management systems of being open source provides lots of help for new developers to develop new systems with enhanced functionalities and increased security to be able to produce a more efficient identity management system.

11. References

[1] https://opensso.dev.java.net/servlets/ProjectDocumentLis
[2] https://opensso.dev.java.net/files/documents/3676/19701/
[3]. https://opensso.dev.java.net/files/documents/3676/26172/
[4]. http://www.openprivacy.org/
[5]. http://talon.openprivacy.org/
[6]. http://sierra.openprivacy.org/
[7] http://reptile.openprivacy.org/
[8]. http://avc.blogs.com/a_vc/2004/11/reputation_serv.html
[9]. http://archie.csce.uark.edu/gpn/publications/Introductionto Shibboleth.pdf
[10].http://shibboleth.internet2.edu/
[11].http://www.banditproject.org/index.php/Welcome_to_Bandit
[12].http://www.banditproject.org/index.php/Common_Identity_Architecture
[13].http://www.banditproject.org/index.php/Common_Authentication_Services_Adapter_%28CASA%29
[14].http://www.banditproject.org/index.php/Compliance_Records_Architecture
[15].http://www.banditproject.org/index.php/Role_Engine_Architecture
[16].http://www.banditproject.org/index.php/Identity_Selector_Service_Architecture
[17].www.projectliberty.org/liberty/content/download/2738/18385/file/I70201_PLAC-190%20Mary%20Ruddy.ai.pdf
[18].http://et.aset.psu.edu/initiatives/Shibboleth.shtml
[19].www.eduserv.org.uk/upload/aim/pdf/audit/matu_institutional_audit_nickjohnsonfinal.pdf
[20]http://www.banditproject.org/index.php/Compliance_Records
[21]. http://shibboleth.internet2.edu/minutes/SHIB-05-Sept-2001.html
[22]http://www.bandit-project.org/index.php/IA_Security_Best_Practices