A Survey on Single Sign-On

Iadalin Nongbri[1], Pynbianglut Hadem[2], Sarat Chettri[3]
[1]Mtech in computer science &engineering, Assam Don Bosco University,
[2]Scientist C, National Informatics Centre,
[3]Assistant professor, Assam Don Bosco University

Abstract. This paper consists of the description of what Single Sign-On (SSO) is, how it works and the different protocols used in it. It portrays the description of Single Sign-On (SSO) and how it works. It also compares the different types of SSO and where it can be deployed. It illustrates the different protocols used in SSO which are OpenID, SAML, BrowserID, and Kerberos. The paper gives an outlook on the various security issues involved in SSO. It presents the difference between PKI-based and Token-based protocol.

Keywords- OpenID, OAuth, Kerberos, SAML, BrowserID

1. Introduction

1.1 What is Single-Sign on (SSO)?

Single sign-on (SSO) is a mechanism that allows users to authenticate mobile application or web applicationwith single username and password to access multiple applications that uses the same authentication provider. SSO is use for authentication and authorization. Authentication means the process of verifying who you are. It deals with confidentiality, integrity, non-repudiation, and availability. Authorization is a process of gaining access to a resource.The advantage of using SSO is that the user does not have to remember all the credentials of all the applications separately and the disadvantage is that if the third party gets to access on any website which is integrated with any protocols, then the overall system becomes insecure.

1.2How Single Sign-On works?

The user registers himself/herself with the IDP to receive OpenID credentials. Now, the user wants to access Application A. The Application A redirects the User to the IDP. The user logs in to the IDP using his/her OpenID credentials. On successful authentication, the IDP redirects the user back to Application A. If the user wants to access the Web application B, it will send a request to Web application B, and on receiving the request, it goes to the Identity provider and checks whether the user who tried to access it is active or not. If found active, Web application B will allow the user to access it automatically. Similarly, others Web applications will also follow the same process. Web application A doesn't know what is going on in Web application B, and Web application B doesn't know what is going on in Web application A and so on.

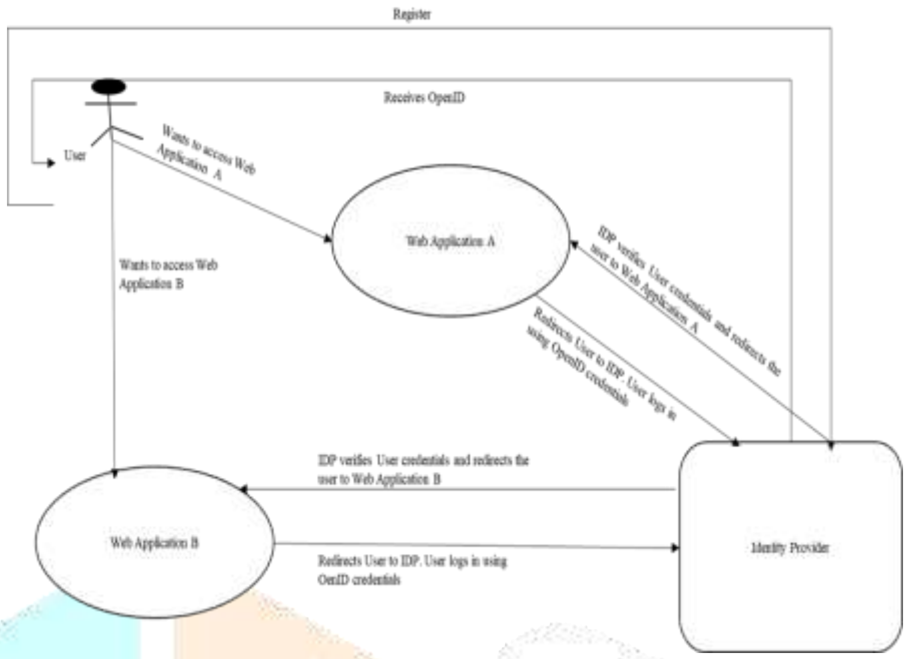
Register
Receives OpenID
User
Wants to access Web Application A
Web Application A
IDP verifies User credentials and redirects the user to Web Application A
Redirects User to IDP. User logs in using OpenID credentials
Wants to access Web Application B
IDP verifies User credentials and redirects the user to Web Application B
Web Application B
Redirects User to IDP. User logs in using OenID credentials
Identity Provider

Figure1: Working of Single-Sign on

1.3 Types of SSO
There are different types of Single Sign-On: Simple SSO [1] and Complex SSO [1] which **are** discussed below.

Simple SSO
Simple SSOcovers a single authentication authority. It can **be implemented in homogeneous** LAN and intranet, where all the machines are running the same operating system and trusting the same authentication authority.

Complex SSO
Complex SSOcovers many different authentication authorities. It is implemented in different platforms and governed by **different organizations**. It can be implemented on either Internet or Extranet.

1.5 Different types of Protocols
There are several types of protocols used in SSO like OpenID, SAML, BrowserID, and Kerberos. The details about these **protocols are discussed** below.

1.5.1 The OpenID
The OpenID mechanism is a decentralized authentication scheme for the SSO (SingleSign-On) mechanism [2].OpenID users can choose a trusted OpenID server to register their OpenID. They are identified by a URL like: http://yourname.openidserver.com. In OpenID mechanism, three parties are involved: the OpenID provider (OP), the Service Provider (SP) which is also called Relying Party (RP) and the user. This survey paper [2] assume that the OP and the RP trust each other in advance; OP has a trusted list of RPs. In OpenID mechanism, users only need to have a pair of identity and password. The User then submits his OpenID Identifier. An OpenID authentication flow [2] describe below in figure 3.

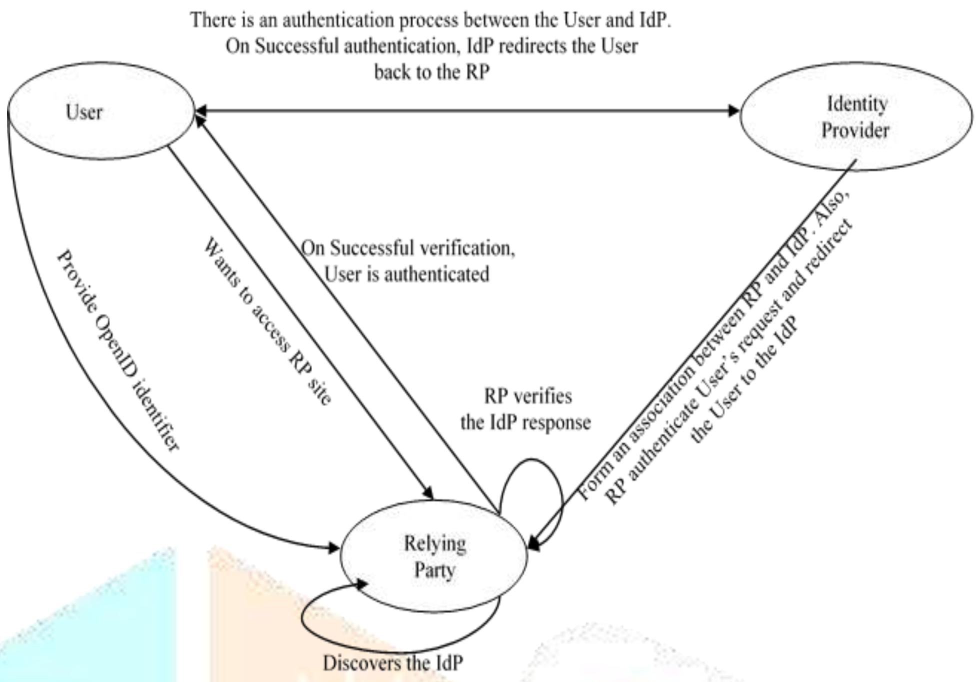
There is an authentication process between the User and IdP. On Successful authentication, IdP redirects the User back to the RP
User
Identity Provider
Provide OpenID identifier
Wants to access RP site
On Successful verification, User is authenticated
RP verifies the IdP response
Form an association between RP and IdP. Also, RP authenticate User's request and redirect the User to the IdP
Relying Party
Discovers the IdP

Figure 3. OpenID Authentication Flow

The OpenID authentication process flow is described as below: [2]

1. The user requests for access toRelying Party (RP) site.
2. RP discovers the Identity Provider (IDP).
3. RP forms an association between RP and IDP.
4. RP redirects the User to the IDP.
5. IDP authenticates the User.
6. RP verifies the IDP's response.
7. On successful verification, User is authenticated, and the user gets access to the RP site.

1.5.2 OAuth2.0

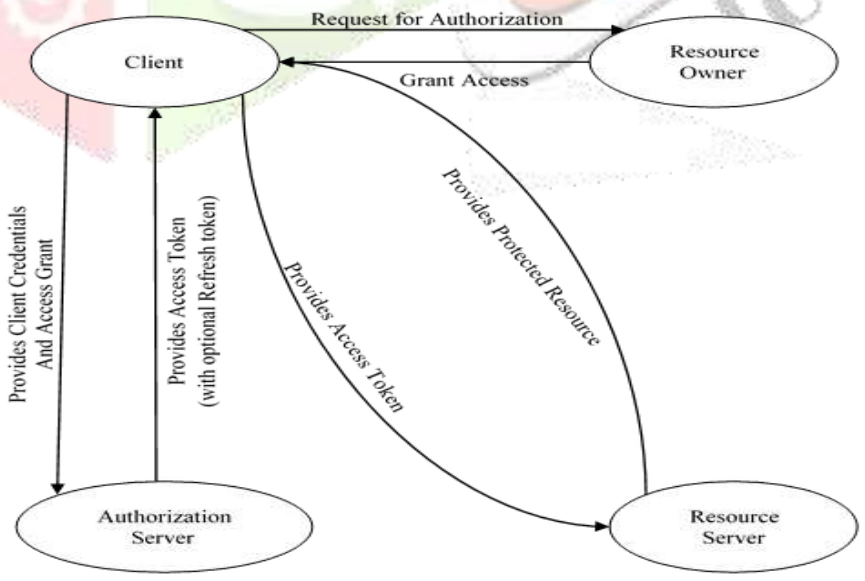
Request for Authorization
Client
Resource Owner
Grant Access
Provides Client Credentials And Access Grant
Provides Access Token (with optional Refresh token)
Provides Access Token
Provides Protected Resource
Authorization Server
Resource Server

Fig 4 Flow of OAuth 2.0 Protocol

The flow of OAuth 2.0 protocol includes the following steps: [11]

1. The client requests the resource owner directly or indirectly via the authorization server for authorization.
2. The client gets an authorization grant.
3. The client makes requests for an access token. This is done by authenticating with the authorization server and also presenting the authorization grant.
4. The authorization server authenticates the client, validates the authorization grant, and on successful validation, issues an access token.
5. The client authenticates himself via the access token. The client then requests for the protected resource from the resource server.
6. The resource server validates the access token, and upon successful validation, responds to the request.

1.5.3 SAML (Security Assertion Mark-Up Language)

SAML is an XML message format that defines a protocol specification to use when two servers need to share authentication information. The protocol uses the web infrastructure where XML data moves over HTTP protocols on TCP/IP networks. SAML 2.0 is quite complex technology. In SAML, IDP and SP exchange messages through the user's browser. A SAML authentication request message is sent by SP to the IDP, to authenticate the user. The IDP validates the username and password of the user and if found correct, sends back a SAML authentication response to the user stating that the user has successfully logged in. So, let's take a look at what happens when someone wants to log in at a Service Provider (SP) that uses federated authentication for one of its customers (the IDP). For example, let's say the SP is Google Apps and the IDP is an organization called My University, where Alice is a student. Now, when Alice wants to read her mail using a web browser, she typically navigates to a webpage like https://mail.google.com/a/my-university.nl. Google will not ask for a username and password to log in, but instead, redirect the browser to the IDP for authentication.URL, the user, is redirected to might look something like (strongly abbreviated): https://idp.uni.nl/sso?SAMLRequest =fVLLTuswEN0j8Q…c%3D. Embedded within this redirect message is a SAML authentication request message. As SAML is XML-based, the complete authentication request message is compressed and encoded. If we remove encoding and compression, the SAML message will be like this: In plain English, this message reads "Google's requested to authenticate the user who sent this message, and send back an acknowledgment to Google." When the IDP receives this message and decides to grant Google's request, it will authenticate Alice by asking her to enter her credentials. The browser of Alice is sent to AssertionConsumerService after successful authentication. SAML protocol message carries a SAML authentication response message. When we decode this message, this is in essence what it in looks like: "this is a message from idp.uni.nl. I have successfully authenticated a user called 'alice.' This message will expire in a couple of minutes." One essential piece of information that **was** deleted from the message above for brevity is an XML digital signature that is used as proof that the message indeed came from idp.uni.nl, and that there was no change in the message. The digital signature was made using a public key algorithm and the public key needed to verify the signature is embedded in a certificate that is known to Google. On receiving the SAML authentication response message, Google first verifies the XML signature, and checks other conditions, and extracts the user's identifier. If everything is fine, Alice is logged in – her mailbox is retrieved, and she can start reading her mail. SAML uses Token based authentication and also PKI based authentication.The disadvantage of SAML is that it was designed only for web applications.

1.5.4 Browser ID

BrowserID offers one-time log-in to websites and services by connecting through an e-mail address. The main idea is that we will always remember our e-mail address instead of a made-up user name or URL. BrowserID has in-built security as it is implemented with Verified Email Protocol. It works on both on desktop and mobile, and it is decentralized so that anyone can choose to implement it on their website. The main advantages of BrowserID are Ease of use, Security, Cross-browser implementation, decentralized, web-wide validation, and improved experience in future browsers and respecting the privacy of the user.BrowserID makes use of email addresses which allows a site to use BrowserID without the need to provide any additional information. A website will ask the browser for a BrowserID assertion via JavaScript which is known as the user agent. An assertion is a string which contains a JSON structure of identity assertion and identity certificate. The assertion will be generated by the browser for an identity; the browser will need to get an identity certificate from a provider to ensure they own that identity. This identity can be used again and again until it expires.BrowserIDuses PKI based authentication, BrowserID is an experiment at Mozilla labs, is very new, not fully defined (exactly how the e-mail servers public key should be found is not specified for example) and not completely implemented. Developed for Mozilla browser.

1.5.5 Kerberos

Kerberos is an authentication system designed by Steve Miller and Clifford Neuman and target for project Athena in MIT. Kerberos uses a trusted third party or calls a middle-man server, for authentication. And Kerberos is based upon Needham-Schroeder-protocol. Kerberos is a protocol between trusted hosts across the untrusted network for authenticating service requests. Major operating systems like Microsoft Windows, Apple OS X, FreeBSD, and Linux are built into this protocol**.** Kerberos is a protocol for authentication where passwords can be sent over the internet or can be stored locally; it is built on symmetric-key cryptography where a trusted 3rd-party involved. When there is a request access to a service or host, three interaction occurs which are: Authentication Server, the Ticket Granting Server, and the Service or host machine to access to.Needham-Schroeder protocol refers to a communication protocol used to secure an insecure network. This protocol allows proving the identity of the end users communicating, and also prevents a middle-man from eavesdropping. Kerberos fits in Token based authentication.

1.6 Security Issues of Protocols used in SingleSign-On

The security issues involved in OpenID and SAML are Phishing, Man-in-the-middle attack, and Session related attacks [1] [3]. There are two common Phishing attacks: Phished OP Pagewhere a rogue RP can redirect the user to a phished OP page where the user will be tricked into entering their OP credentials; Realm Spoofing where a malicious RP can craft an authentication request with a protocol [1] [3]. Realm set to a trusted domain and the return to pointing back to its own page [1] [3]. Without the Realm/Return to validation, the user's OP will assert to the user that they're signing into the trusted domain, when in fact, they're being redirected back to the malicious RP [1] [3]. Man in the middle attack: In these protocols, the connection is negotiated over DH (Diffie-Hellman) which is subjected to interception attacks [1] [3]. Session Related Attacks: These protocols facilitate user having several active authenticated sessions [3]. It provides more opportunities for a malicious site to exploit the vulnerabilities of OP and other RPs since the user already has an authenticated session [1] [3]. Some of the security issues to be taken into consideration for BrowserID are: Certificate validity period, User Authentication reusing a key to get a new certificate and Timing attacks, JavaScript implementations [1].

The security issues of Kerberos are: In Kerberos infrastructure user login credentials are stored in the central server. So it will migrate all login credentials from local machines /etc/password and /etc/shadow files to the central server. If some attacker gets access to the central server, the entire infrastructure will be under threat [1].The security issues of OAuth2.0 are CSRF attack, unbounded tokens, Bearer tokens, Expiring tokens, and Grant types. [11]

1.7 Strengths and Limitations of Token based and PKI-based sets of credentials

There are different types of protocols used in SSO like PKI-based [1], and Token-based [1] and their advantages and disadvantages are stated below. PKI-based SSO system is still a kind of Token-based SSO system because they differ in cryptographic methods, but tokens are still used to permit the client to get access to application servers.

Table 1. Advantages and disadvantages of PKI-based and Token-based.

	Advantages	**Disadvantages**
PKI-based	Uses single set of credentials.	Cannot use multiple sets of credentials.
	Uses Asymmetriccryptography.	Only short messages are used. Long messages are not implemented.
	Provides centralized management to security administrators.	Requires a homogeneous authentication infrastructure environment.
	It allows the choice of trust provider.	
	It is highly scalable. Users maintain their own certificates, and certificate authentication involves the exchange of data between client and server only. This means that no third party authentication server needs to be online.	Complex certificate validation logic; requires a lot of processing on the client side.
	PKI allows delegated trust. That is, a user who has obtained a certificate from a recognized and trusted certificate authority can authenticate himself to a server the very first time he connects to that server, without having previously been registered with the system.	The third party, called a certification authority, digitally signs their public key, turning it into a digital certificate, so that you can be sure it's safe to use. However, if the certification authority gets compromised, the criminal that did it could issue false certificates and fool people into sending data to the wrong place.
	It is not exactly the	

	authentication based system to implement single sign-on but It can be implemented in such a way as to enable single sign-on.	
Token-based	A Single set of credentials simplifies the life of user and administrator.	Requires a homogeneous authentication infrastructure environment.
	Software usually comes bundled with OS software.	Relies on symmetric cryptography.
	Tokens are valid for more than a single token.	Impossible to predict how much time token is valid for.
	Support most of the platforms.	Quite difficult to identify when the client is identified from multiple sources semantically.
	Support entity authentication and data authentication.	Difficult to Scale because of having the dependency on the database to authenticate the client with each request.
	Permission-based token can be created, and data can be able to access along to a third-party application. But only the information with the specific token will allow.	The token can have its own lifetime and may expire accordingly. In that case, the user will need to be authenticated again into the system.
	Since the tokens are encrypted and digitally signed, so it is secure and fast. It is easy to scale with a token based authentication system. No cookies used.	Impossible to identify the user from which client (like Angular App, Android App, etc.) is authenticated with.

2Literature Survey

In thework presented by V Radha et al. [1] various methods of Single Sign-On (SSO) and the advantages of adopting it. It also discusses on implementing various types of SSO and the protocols that are being used. The concept of SSO can be used within an Intranet, Extranet or Internet. There are various types of SSO: Intranet or Enterprise SSO, Extranet or Multi-domain SSO. These types of SSO are based on where they are deployed (Intranet, Extranet, Internet); how they are deployed (architecture – Simple, Complex); the credentials they are (token, certificate) and the protocols they are (Kerberos, SAML, OpenID).Therefore, Single Sign-On is easier and one of the safer application by reducing into one account per users for all services. It can gain more information and more importance with the emerging Cloud Computing technology. It should be implemented in a very secure way to detect the attack like phishing. One must carefully estimate the resources available before choosing Single Sign-On solution, or else it can create huge vulnerability to the organization security.

In thework presented by Hitesh C. Patel et al. [2] described how to access different websites or web applications by using only a single set of credentials. There are several protocols that can access websites, for example, OpenID, SAML, OAuth, etc. But OpenID is used in this paper. It is an open, decentralized framework. They use Diffie-Hellman key exchange while generating a secret key between IdP and SP. The advantage is that it reduces the registration process for authentication. The main issues of this paper is that it doesn’t define how to improved security in OpenID Authentication framework.

In thework presented by Eugene Tsyrklevich et al. [3]explained about the working of OpenID in Single Sign-on. It consists of the user, OpenID provider, and the Relying Party. A user can register with any OpenID provider. Identity Provider deals only with authentication and not authorization. There are several attacks that had mentioned in this paper such as negotiating the crypto keys using Diffie-Hellman algorithm, but the solution is this algorithm is that instead of executed in HTTP it can be executed in HTTPs, phishing attack may also happen when redirecting the RP to the wrong identity provider, but this attack has remained unsolved in this paper, malicious IdP may spy on a user’s activity while authentication happens in the identity provider, while redirecting back to the original website, anyone can obtain this URL and log into the site as the victim user, some of the IdP uses nonce to allow the user to login once but for the fast attacker who is sniffing the wire can be the first one to obtain URL.Thus nonce can only prevent the passive attack and not the active attacks. Some of the attacks had been partially solved and phishing attack and redirect attack remains unsolved.

In theworkpresented by Anita Patil [4] discuss the basic sign-on model and disadvantage of the multi sign-on system. Single sign-on model is presented, with the focus on the different SSO architectures like secure Client-side Credentials Caching, Server-side Credential Caching, and Single Sign-On with Single Set of Credentials, PKI-based Single Sign-On and Token-based Single Sign On and compare the SSO solution with proxy signature. It presented a new approach for practical efficient and secure single sign-on frameworks based on proxy signature schemes. It uses public key cryptography with MD5 techniques to the problem of practical single sign-on.

In the work presented by Bart van et al. [5] present the authentication flow of OpenID by Discovering the User's Identity, Establishing an Association between User, RP (Relying Party) and OP (OpenID Provider), Redirecting to the OP, Redirecting to the RP and Verifying the response. This paper also presents the Security Issues of OpenID such as Adversaries will focus on Ops, Spying Ops, Cross-Site Scripting, Session Swapping, and RP Getting Lost in Discovery, Diffie-Hellman / Man in the Middle Attack, OpenID Recycling and Phishing**.**This paper also presents some of the solutions to the security issues like Trust Framework, Anti-phishing technique, Preventing Cross-Site Scripting Attacks.

In the work presented by Thomas Grofi [6]presents a security analysis of the SAML Single Sign-on Browser/Artifact profile. SAML is an important standardized example of this new protocol class and will be widely used in business-to-business scenarios to reduce user-management costs. It utilizes a constraint-based specification that is a popular design technique of this protocol class. It does not include a general security analysis, but provides an attack-by-attack list of countermeasures as the security consideration. SAML Single Sign-on Browser/Artifact profile is a three-party authentication protocol. It allows a user to sign on only at his or her identity supplier, which in turn confirms the user's identity to other parties. The protocol flow of the SAML Single Sign-On Browser/Artifact Profile is also presented in this paper. The protocol assumes that the user authenticated itself to source site beforehand. The protocol flow begins when the user returns to the source site, for instance, having been redirected to a destination site. Source site stores an assertion about the user's identity if it can recognize the browser of the user during the so-called user tracking. It then redirects the user's browser to the destination site the user wants to browse. Source site includes a small piece of data, called a SAML Artifact, into the redirect that refers to the assertion stored. Receiving the redirect with this Artifact, destination site shows this Artifact to source site and requests the corresponding assertion from it. By providing this assertion to the destination site, source site confirms that user presenting the SAML Artifact was authenticated by source site.

In the work presented by Kari Helenius [7], it discussesthe OpenID standard. It illustrates how the concept of enterprise-style SSO can be copied to open Internet (where several identity providers work and manage their own users and identities). It also discusses the OpenID's extendibility and suitability, issues that may raise and the future requirements.

In the work presented by Christian Mainka et al. [8] presents a systematic analysis of well-known attacks on SSO protocols and adapt these on OpenID Connect. Additionally, this paper also introduces two novel attacks on OpenID Connect, Identity Provider Confusion and Malicious Endpoints Attack abusing lacks in the current specification and breaking the security goals of the protocol. The described attacks are categorized into two classes: Single-Phase Attacks which abuses the lack of single security check; Cross-Phase Attacks which needs a setup for complex attack for manipulating multiple messages. This paper also provides an evaluation of officially referenced OpenID Connect libraries and find 75% of them vulnerable to at least one Single-Phase Attack.

In the work presented by Michael Flemming Grubb et al.[9] portrays the various factors like One Password, Application Dependency, implementation of a Single Authentication Realm (SAR) involved in the deployment of Single Sign-On solution. It also provides a review of various available approaches to the problem of electronic identity proliferation.

In the work presented by Kurhe Bhagwan Subhash et al. [10]discussed the different framework like SAML 2.0, OAuth 2.0, OpenID 2.0, OpenID Connect 1.0, and the Importance of BYOD (Bring Your Own Device), that are used for SSO.

In the work presented by Er. Gurleen Kaur et al.[11]discusses the basic concept of OAuth 2.0 and the various challenges of this protocol like CSRF attack, unbounded tokens, Bearer tokens, Expiring tokens, and Grant types.

3. Conclusion

OpenID in Single Sign-On is used only for authentication, OAuth is used for authorization, OpenID connect is used for both authentication and authorization. Furthermore, as the number of credentials increases does forgetting or losing them also increases. Therefore, Single Sign-on is one of the mechanisms for logging into one application only once and signed in to other applications automatically. And security is also one of the main issues in all the protocols. There are many attacks like, man-in-the-middle attack, session attack, Phishing attack that may exploit the identity of the users. However there are some papers which mention some precautions against it. But there are papers which cannot escape from this threat and Privacy is also one of the main issues in OpenID framework protocol.

References

[1] V Radha and D Hitha Reddy, "A Survey On Single Sign-On Techniques," SciVerse ScienceDirect, Procedia Technology 4, Institute for Development and Research in Banking Technology, Road #1, Castle Hills, Masab Tank, Hyderabad 500067 (A.P.), (2012) 134 – 139

[2] Anita Patil, Prof. Rakesh Pandit, Prof. Sachin Patel, "Analysis of Single Sign-on for Multiple Web Applications," International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering (An ISO 3297: 2007 Certified Organization), Department of Information Technology, PCST, Indore, MP, India, Vol. 2, Issue 8, August 2013, ISSN (Print) : 2320 – 3765, ISSN (Online): 2278 – 8875

[3] Bart van Delft and Martijn Oostdijk, "A Security Analysis of OpenID,"Comp. Sci. Dept., Radboud Univ., P.O. Box 9010, 6500 GL, Nijmegen, The Netherlands, b.vandelft@student.ru.nl, Novay, P.O. Box 589, 7500 AN, Enschede, The Netherlands, martijn.oostdijk@novay.nl

[4] Prof. Hitesh C. Patel," Survey of OpenID Authentication Framework," International Journal of Computer Science and Mobile Computing, A Monthly Journal of Computer Science and Information Technology, IJCSMC,Assistant Prof., Information Technology Dept., KITRC, Kalol hiteshldit@gmail.com, Vol. 4, Issue. 9, September 2015, pg.364 – 367,ISSN 2320–088X

[5] Eugene Tsyrklevichand Vlad Tsyrklevich, "Single Sign-On for the Internet: A security Story,"eugene@tsyrklevich.name, vlad902@gmail.com, BlackHat USA, Las Vegas 2007

[6] Thomas Groß, IBM Zurich Research Laboratory, tgr@zurich.ibm.com, "Security Analysis of the SAML Single Sign-on Browser/Artifact Profile"

[7] Kari Helenius,kheleniu@cc.hut.fi "OpenID and identity management in consumer services on the Internet"

[8] Christian Mainka, Vladislav Mladenov and Jörg Schwenk,"SoK: Single Sign-On Security – An Evaluation of OpenID Connect"

[9] Michael Flemming Grubb and Rob Carte, "Single Sign-On and the System Administrator," Duke UniversityTwelfth System Administration Conference (LISA '98), Boston, Massachusetts, December 6-11, 1998

[10] Kurhe Bhagwan Subhash and Prof. Kahate S.A., "Different Framework for Single Sign-On (SSO)," International Journal of Computer Science and Mobile Computing, Vol 5 Issue 1, January 2016, pg. 53-56

[11] Er. Gurleen Kaur and Er. Deepak Aggarwal, "A Survey Paper on Social Sign-On Protocol OAuth 2.0," Journal of Engineering, Computers and Applied Sciences (JEC&AS), BBSBEC Fatehgarh Sahib (Punjab), ISSN No: 2319-5606, Volume 2, No. 6, June 2013