WSASRESSO - A Novel Framework for Analysis of SAML based SSO Protocols using Black Box Penetration Testing

Shymala Gowri Selvaganapathy
Department of Information Technology
PSG College of Technology
Coimbatore, India

Nivaashini Mathappan
Department of Computer Science & Engineering
Bannari Amman Institute of Technology
Sathyamangalam, India

Hema Priya Natarajan
Department of Information Technology
PSG College of Technology
Coimbatore, India

Sasidharan R.
Department of Information Technology
PSG College of Technology
Coimbatore, India

ABSTRACT

Single Sign-On (SSO) is a simplified approach which relieves users from the burden of dealing with multiple credentials but at the same time presents new security challenges. With three different parties participating in the authentication process, SSO solutions involve different layers of communication and exchange of credentials that are enabled by using HTTP redirection and JavaScript, which creates several vulnerabilities for attackers to exploit and makes SSO a launch pad for typical attacks. A formal method is needed to evaluate the flaws in the SSO protocol implementation. The security service Availability is important to ensure that the information concerned is readily accessible to the authorized persons; here the problem of Violation of Availability in SSO is addressed. This work WSASRESSO provides a framework which evaluates SAML based SSO protocols using Burp suite extension with a combination of EsPReSSO algorithm for identification of the SSO protocols along with SAML Raider for fetching the protocol infrastructure details and integration of WS-Attacker to perform black box penetration testing. Since new types of SSO attacks are evolving over time, the proposed security framework can be used to find the strength of the SSO protocols. Here, signature based attacks like XML Signature Wrapping and XML Signature Faking attacks have been simulated and tested which can be categorized under Phishing attacks.

General Terms

Security testing, Vulnerabilities

Keywords

Single Sign-On, Protocol Vulnerabilities, WSASRESSO, Burp Suite, EsPReSSO, SAML, Attacker.

1. INTRODUCTION

Single Sign-on (SSO) service is becoming a new trend and there is a wide range of implementation, from mobile applications to browser-based protocols, which connects everything. It relieves users from the burden of dealing with multiple credentials but at the same time presents new security challenges. To address the problem, the basic model of SSO has been examined and then discusses the existing flaws in the emerging SSO protocols, like SAML SSO and Open ID. SSO enables the user to log in and gain access to multiple websites without the repetition of typing multiple passwords. Thus, many leading web technology companies such as Facebook, Google, Slide Share, Dropbox, Salesforce, Yahoo, and Twitter offers SSO services. These services, have three parties: the User (U), Service Provider (SP) and Identity Provider (IdP). These parties participate in the authentication processes by HTTP traffics, which help for SSO mechanism. Thus being most frequently used, it creates several vulnerabilities for attacks. Secure web SSO system is expected to prevent an unauthorized party from gaining access to a user’s account on the SP’s website.

1.1 Motivation

SSO with major functionality today, it requires the serious witness to understand, how securely the SSO mechanisms are deployed. The main objective of this work is to frame a formal method to find protocol vulnerabilities. The actual web traffic traveling over web browser has been tried to examine, and recover Protocol details and identify potential exploit opportunities.

1.2 Concept of SSO

In SSO (Single Sign On), if the user has logged into one application, they can automatically sign in to every other application, regardless of the platform, technology or domain. Fig 1, shows the involvement of three parties (U, SP, IdP) in an instance of an SSO protocol. The identity provider serves as a centralized identification service for the users, who establishes the user’s identity. Some examples of such identity providers include Facebook Connect and Open ID. When the user wants to get access to services provided by a relying party, which is a website like Stack Overflow, the relying party uses the services provided by an IdP to authenticate the user. Upon login for the first time, a cookie gets created on this central service of IdP.

Then, when the user tries to access the next application, it gets redirected to the central server, which already has a cookie, so gets redirected to the application directly with a token, so relying on party uses service provided by IdP for authentication. From the aspect of the user, Single Sign-on enables the experience of logging in to Stack Overflow with one's Facebook account.

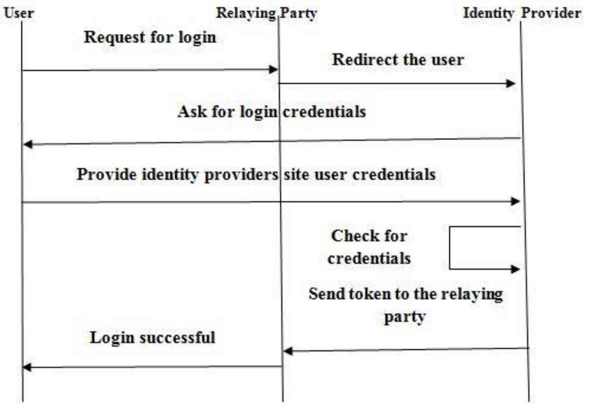

Fig 1: State Chart for SSO Protocol

2. VULNERABILITY IN SSO

With three different parties participating in the authentication processes, SSO solutions involve different layers of communication and exchange of credentials that are enabled by using HTTP redirection and JavaScript, which creates several vulnerabilities for attacks to exploit and makes SSO a launch pad for typical attacks such as

- Phishing (Man in the Middle attack)
- Cross Site Scripting
- Replay attack

2.1 Phishing - (Malicious Identity Provider)

With different layers transited by redirections in the process of SSO authentication, attackers are provided an opportunity to steal the identity of the trusted party and thus launch the attack.

1. As shown in Fig 2, in the beginning, User requests a resource URI (correct) from the malicious service provider (MSP) or Service Provider (SP).
2. Malicious Service Provider/ Service Provider redirects to Malicious Identity Provider (MIdP).
3. MIdP request user for the credential.
4. The user is exposed to send the credential to MIdP.

The User is redirected to the IdP by Service Provider, it is easy for a malicious SP to redirect User to a fake IdP and therefore steal the user’s credentials [12]. Since a lot of users might not be careful enough to identify the fake web page, also known as “phishing”. It is still an unresolved and well-known attack against SSO.

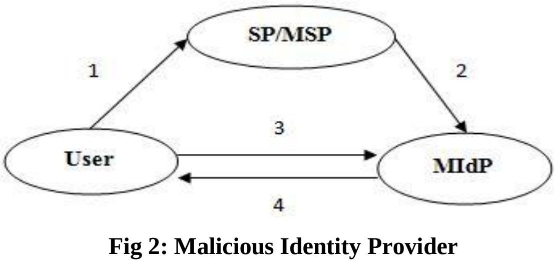

Fig 2: Malicious Identity Provider

2.2 Cross Site Scripting – (Malicious Relying Party/Service Provider)

XSS attacks that can be triggered by visiting a maliciously-crafted URL. In addition, implementation of the SAML SSO protocol exposes to a possible injection of malicious code that may be executed at the honest SP side. It is even harder to identify the integrity of a URI request. Based on that, the attack can be conducted on a website that supports SAML-based Single Sign-on, and it involves four parties as shown in Fig 3: a user (U), a malicious service provider (MSP), an honest service provider (SP) and an honest Identity Provider (IdP) [12].

1. At the beginning, User requests a resource URI (correct) from the malicious service provider (MSP).
2. MSP then pretends to be User and request a different URI (malicious) at SP, which will react according to SAML standard by generating an Authentication Request to IdP.
3. Service Provider will send the Identity Provider details to MSP.

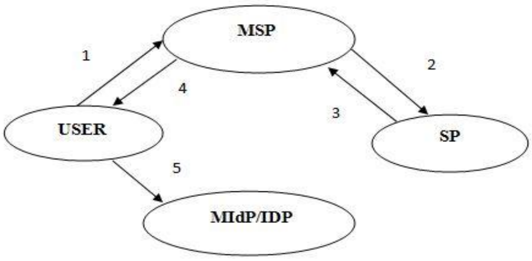

Fig 3: Malicious Service Provider

4. The malicious service provider then redirects User to IdP by an HTTP response containing the Authentication request and URI (malicious).
5. Thus, User is forced to consume a different resource from SP, which the URI (malicious) is created by MSP.

2.3 Replay Attacks

When single sign-on is enabled, a user's cookie can be stored and is used to access data in another domain. While it is not recommended that single sign-on be used when a component has turned off active content filtering, it is possible to use single sign-on with HTTP Only Cookies. Application Server has the ability to produce "HTTP Only" cookies for the single sign-on cookies. Every SSO protocol provides parameters to limit the reuse and lifetime of the authentication tokens. The attacker needs to access to a valid token for gaining resources. More specifically, the token is a question based to validate the User to login [12]. This can be achieved if the legitimate access gained by an attacker with SP using SSO and uses the access to generate and store a token created by the attacker.

3. SSO PROTOCOLS

The investigated Single Sign-On protocols can be divided into two groups. The protocols in the first group are based on the authorization protocol OAuth. The second group includes all protocols that are not based on OAuth, such as OpenID and BrowserID. These protocols make use of OAuth typical parameters and behaviors [1]. The intersection between the protocols increases the difficulty to distinguish and analyze them. The following section provides a brief overview and guidance to understand each of them. A common characteristic between all protocols is the data exchange with HTTP GET or POST parameters.

3.1 OAuth-family protocols

The following protocols are based on or make use of, the authorization protocol OAuth. It should be noted that OAuth is an authorization framework and therefore not capable of doing Single Sign-On or authentication.

3.1.1 OAuth

OAuth is an authorization framework that enables applications to obtain limited access to user accounts on an HTTP service, such as GitHub. It works by delegating user authentication to the service that hosts the user account and authorizing third-party applications to access the user account. OAuth 2 provides authorization flows for web and desktop applications, and mobile devices. OAuth is capable of using both, XML (in particular SAML) and JSON – JavaScript Object Notation (in particular JWT – Java Web Token) for data interchange, away from the normal HTTP parameters [6]. The OAuth Protocol has no dependencies on any of the other researched protocols.

3.1.2 OpenID Connect

OpenID Connect is an interoperable authentication protocol based on the OAuth 2.0 family of specifications. It uses straightforward REST/JSON message flows. It is the combination of Identity, Authentication and OAuth 2.0. Its communication is based on JSON and JWT, along with the standard POST/GET parameters [7]. Despite the similarities in the name, OpenID Connect is a completely different protocol than OpenID.

3.1.3 SAML - Secure Assertion Markup Language

Secure Assertion Markup Language (SAML) is based on XML and it is possible to use it to sign and encrypt data. It has no dependencies on other protocols. The SAML has existed in its first version since November 2002 as an OASIS standard [12]. The last version 2.0 is standardized. SAML describes a method for the exchange of cryptographic secure information, and is not designed for Single Sign-On but is capable of it.

3.1.4 Facebook Connect

Facebook Connect is developed by Facebook Inc. to authenticate users on third party websites and authorize web applications' to access user resources like email address or photos. Facebook assigns a unique API key (required for calling Facebook API methods), and an API secret key (to be kept a secret between Facebook and the application) [3]. After receiving a response, the browser initializes the Connect service by calling FB.init. This returns one of three possible states: Not logged in/Not authorized/Connected. The protocol is based on OAuth 2.0, and thus it uses JWT for communication [10].

3.1.5 Microsoft Account

Microsoft Account is a proprietary protocol developed by Microsoft. It adopts the OpenID Connect protocol and OAuth framework [11]. Microsoft Account allows users to log into websites (like Outlook.com), devices (e.g. Windows 10 computers and tablets, or Windows Phones), and applications (including Visual Studio) using one account. Hence, Microsoft Account utilizes the same technologies as OpenID Connect and OAuth.

3.2 Non-OAuth protocols

The following protocols are not based on the OAuth framework and are therefore easier to differentiate due to their uniqueness.

3.2.1 OpenID

OpenID is decentralized. No central authority must approve or register Relying Parties or OpenID Providers. The authentication scheme works with "AJAX"-style setups [4]. This allows an end user to prove their Identity to a Relying Party without having to leave their current Web page. OpenID Authentication uses only standard HTTP(S) requests and responses, so it does not require any special capabilities of the User-Agent or other client software. Extensions to User-Agents can simplify the end user interaction, although which is not required to utilize the protocol. OpenID Authentication is designed to provide a base service to enable portable, user-centric digital identity in a free and decentralized manner.

3.2.2 BrowserID

BrowserID is built by Mozilla and implements a variant of the verified email protocol. BrowserID is developed and distributed under the name Persona by Mozilla. BrowserID aims to offer one single log-in to web sites and services, connected through e-mail address [13]. The core idea is that it will always remember the e-mail address instead of a made-up user name or URL. As a feature, BrowserID supports an interface to integrate existing OpenID and OpenID Connect services, as well as a fallback IdP. The protocol data exchange uses JWT and JSON.

4. STUDY OF SAML PROTOCOL

Security Assertion Markup Language (SAML), is a XML-based open standard for exchanging authentication and attributes between an identity provider and a service provider. SAML is a product of the OASIS Security Services Technical Committee and aims to standardize framework for browser based single sign-on (SSO) [12]. Google uses the SAML based Single Sign-on service for the web applications including Gmail, Google Calendar, Google Docs etc., SAML does attribute exchange through the creation of trust relationships between IdP's and SP's. As shown in Fig 4, the three main components of the SAML specification are:

> **Assertions** – Assertion is used for Authentication and Authorization of information. Authentication assertions are those in which the user has proven his identity. Attribute assertions contain specific information about the user, such as an email and phone number.
> **Protocol** – This defines the way that SAML asks for and get assertions, for example, using SOAP over HTTP.
> **Binding** – Exactly how SAML message exchanges are mapped into SOAP exchanges.

The assertions are exchanged among sites and services using the protocol and binding, and those assertions are what authenticate users among sites. SAML 2.0, borrows protocols and intellectual property from a number of frameworks to standardize SSO across all applications.

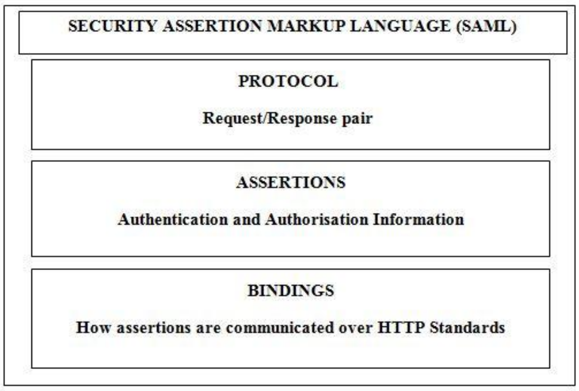
SECURITY ASSERTION MARKUP LANGUAGE (SAML)
PROTOCOL
Request/Response pair
ASSERTIONS
Authentication and Authorisation Information
BINDINGS
How assertions are communicated over HTTP Standards

Fig 4: Components of SAML Protocol

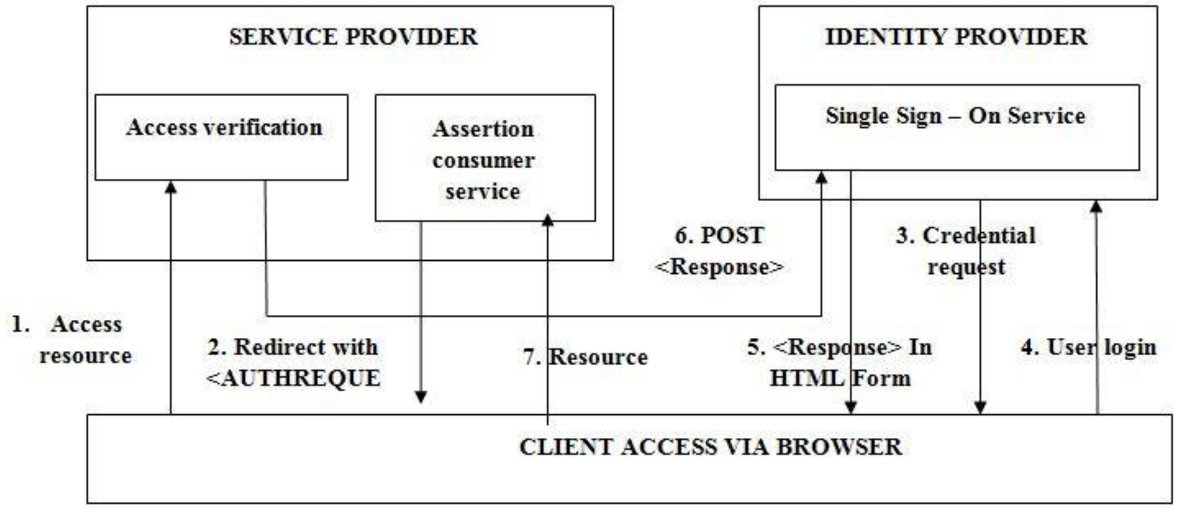

Fig 5: Message Flow of SAML Protocol

Consider the scenario when a user wants to log in at the Service Providers (SP) that uses Identity Provider (IdP) for authentication of the user as shown in Fig 5 [20]. For example, let us assume SP as Dropbox and IdP as Google.

1. User requests for an accessible resource to Service Provider (Dropbox).
2. Service Provider (Dropbox) which then redirect the <AuthRequest> to the Identity Provider (Google).
3. Identity Provider (Google) ask for user credential to the User who requested for the resource.
4. The user provides the Login details to Identity Provider (Google) for Authentication mechanism.
5. After successful Authentication of the user by Google, the user gains access by receiving login response given by the Identity Provider (Google).
6. The user then posts back the Authentication token as <Response> by redirecting the response to the Service Provider (Dropbox).
7. This User gains the service from Service Provider (Dropbox) and resource has been provided to the user.

Thus in the whole process, the user has visibility only to the Service Provider and Identity Provider via the browser and all the other background processes has been hidden by the protocol. The messages involved in this mechanism are XML based thus are vulnerable to XML-based attacks. This involves a single point of web application access for the user which leads to security risks that need to be verified before launching SSO Mechanism.

5. RELATED WORK

This project deals with the automatically testing the SSO protocols by dynamically intercepting method. Here the detailed description about the relevant approach which analyzes vulnerabilities in SSO Protocols.

5.1 SSOscan

SSOScan [2] is an automatic vulnerability checker for applications using Facebook Single Sign-On (SSO) APIs. SSOScan consists of two main parts: the Enroller and the Vulnerability Tester. The Enroller automatically registers two test accounts at a web application using Facebook SSO. The Vulnerability Tester simulates attacks and monitors traffic to test for each vulnerability. Given a target web application, this tool first removes all cookies from the browser and navigates to the target URL. A short delay after the page has fired it's on load event. The Enroller then simulates clicks on those elements, monitoring traffic to listen for the Facebook SSO traffic pattern. Once a click or sequence of clicks is found that produces the recognizable SSO traffic.

Simulated Attacks: The two credential misuse vulnerabilities are tested using simulated impersonation attacks.
Passive Monitoring: The other two credential leakage vulnerabilities are detected using passive approaches.

5.1.1 Limitations of SSOScan

While SSOScan is able to automatically synthesize basic user interactions and analyze traffic patterns, this approach is not suitable for detecting all types of vulnerabilities. It only works for vulnerabilities that can be checked by observing traffic. SSOScan tools analyze only one SSO protocol or a small subset of existing attacks. Small deviations in the messages lead to false results. Extending the tools is insufficient or not possible.

5.2 SAMLyze

SAMLyze [14] is a penetration testing tool for SAML Service Provider (SP). The tests are focused on pre-configured payloads designed to test against XML External Entity (XXE) and Document Type Declaration (DTD) attacks, as well as a set of SAML validation methods. The user interface is based on a web interface which makes it, according to the author, easy to configure.

5.2.1 Limitation of SAMLyze

SAMLyze tool analyzes and distinguishes for SAML protocols. Extending the tools is insufficient or not possible.

6. PROPOSED SYSTEM

All the existing system poses unexpected constraints for analysis of SSO protocols. To avoid the inconsistencies that arise from the existing systems, we propose a framework "WSASRESSO – WS-Attacker SAML Raider EsPReSSO Single Sign On" for analysis of SSO protocols. A formal method is needed to evaluate the flaws in the SSO protocol implementation. The security service Availability is important to ensure that the information concerned accessible to the authorized persons; here we try to address the problem of violation of Availability in SSO, providing a framework which evaluates SAML based SSO protocols as shown in Fig 6.Combination of EsPReSSO algorithm for identification of protocols. SAML Raider is used to fetching the protocol infrastructure details and integration of WS-Attacker to perform penetration testing to find the strength of SSO protocol.

6.1 Burp Suite

Burp Suite [16] is an integrated platform for performing security testing of web applications. Burp is a penetration test tool by Portswigger. Burp acts as an intercepting proxy. Burp is used by security auditors, penetration testers for the analysis of different systems. A tester gets a quick overview of the target system, all transmitted messages, and parameters. In addition, Burp provides a GUI allowing the full control over all messages. A tester can design different attack scenarios and execute them manually via Burp. Its various tools work seamlessly together to support the entire testing process, from initial mapping and analysis of an application's attack surface, through to finding and exploiting security vulnerabilities. Burp Suite contains several functionalities, an intercepting Proxy, which inspect and modify traffic between the browser and the target application.

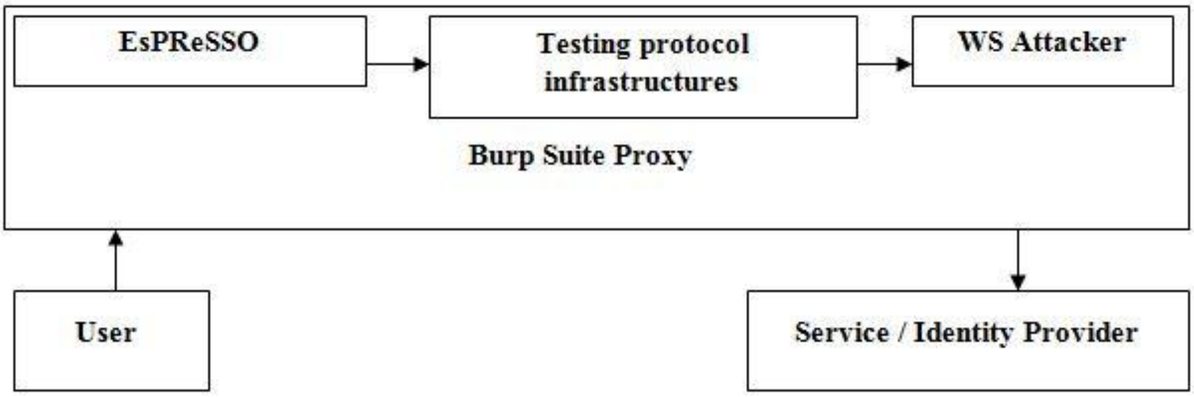

Fig 6: Architecture Diagram

It is an advanced web application Scanner, for automating the detection of numerous types of vulnerability. It is an Intruder tool, for performing powerful customized attacks to find and exploit unusual vulnerabilities. It is a Repeater tool, for manipulating and resending individual requests. It allows us to easily write our own plugins, to perform complex and highly customized tasks within Burp. Burp is highly configurable and contains numerous powerful features to assist the most experienced testers with their work.

6.2 EsPReSSO

EsPReSSO [1] is a Burp Suite Extension. Burp Suite's Proxy HTTP history is a tab which enables the user to review all processed HTTP messages which have been intercepted. If EsPReSSO has already been loaded by Burp Suite, then a new tab, called EsPReSSO, is attached to the top row. All the recognized Single Sign-On protocols are highlighted in yellow. Burp Suite's Request or Response viewer displays information such as raw HTTP message, parsed parameters, and headers. New tabs of EsPReSSO are integrated into this view.

The SSO History is based on the layout of Burp Suite’s Proxy history. In addition to the typical table entries, the columns titled SSO Protocol and Token are added. SSO Protocol describes the recognized protocol and Token displays an identifier of the protocol message.

The extension provides a menu which can be opened, with a right click on a table entry. ’Analyze SSO Protocol’ can be selected to start the analysis of the table for coherent SSO messages. Once the analysis is finished, all related entries are copied into a new table which is then attached next to the Full History tab. The new table is named after the protocol of the selected entry together with a consecutive number. Via the Options tab, the configuration of the extension can be controlled. The checkboxes at the top are used to control the active protocols that are scanned for. If the box is checked, the specific protocol is enabled during scanning. The checkbox next to the headline disables all protocols at once. To disable the highlighting within the Proxy history unchecks Highlight SSO. The configuration is stored in a JSON file in the home folder of the user. The user can load or save other configuration files with the buttons Import and Export. The option Info shows the info and error level messages. The Help tab, displays the name, copyright info, license, and dependencies of the extension in the about tab. Editors are a way to integrate features in Burp Suite's Request/Response viewer. In this work, a SAML Editor tab is created under the proxy tab. The SAML Editor, the editor is integrated as soon as a message includes the parameters SAMLRequest or SAMLResponse.

6.3 SAML Raider

SAML Raider [15] is used to gain SAML infrastructures. It contains two core functionalities: Manipulating SAML Messages and Manage X.509 certificates. The tool is divided into two parts. A SAML message editor and a certificate management tool [15]. An X.509 certificate binds a name to a public key value. The role of the certificate is to associate a public key with the identity contained in the X.509 certificate. To test SAML environments in this work, we have added this SAMLRaider into the Burp suite proxy. By adding a new rule, it checks if a parameter name has SAMLResponse is in the request. The script smallest is used to send a SAML Response to Burp which is available in the scripts directory. The SAML Response gained from the `saml_response` is printed out in the modified response from the plug-in.

6.4 WS – Attacker

WS - Attacker [5] is a modular framework for Web application/ service penetration testing for XML based Attacks. It is an open source software. This software was developed by Christian Mainka. The vulnerability level of the SSO Protocol is found by injecting an attack in the Infrastructure and monitoring the ability of the Protocol to resist the attack. In this work, the tool has been added as a plugin to burp suite proxy for testing the Availability of the system which uses SSO protocols such as Signature Faking attack and Signature Wrapping attack.

The WS - Attacker tab is enabled during the interception of a SAML message. When a message is intercepted, it is possible to modify the message and run attacks against the server. The Attacker functionality is only available within the SAML Editor. To start an attack, we have to click on the Attacker tab and choose between the possible attacks.

6.4.1 Signature Faking Attack

The cryptographic verification of the digital signature guarantees the integrity of the token. Additionally, it is essential to verify the token's authenticity. In other words, the Certificate Authority should check whether the token was signed by a trusted IdP. The Signature Faking attack [8] utilizes possible flaws in the selection logic of the key used for the verification of tokens, by providing an attacker generated token signed by an attacker generated the key. In order to run the attack, the attacker must be able to create SAML tokens and sign them with his own self-created key.

6.4.2 Signature Wrapping Attack

Signature Wrapping attacks [9] will inject a faked element into the message structure so that a valid signature covers the unmodified element while the faked one is processed by the application logic. As a result, an attacker can perform an arbitrary request which denies the availability of the legitimate user by diverting to some other sites or URL.

7. WORK FLOW OF PENETRATION TESTING

An effective classification and analysis of SSO protocol technique have been proposed using a Burp Suite extension. As shown in Fig 7, the process of the proposed analysis of SSO protocol technique executed in the following steps. Adding EsPReSSO and SAML Raider into Burp Suite using Eclipse.

1. Installing Burp certificate in the browser and set up the proxy.
2. Intercept messages using Burp proxy.
3. Categorize SSO protocol using EsPReSSO.
4. Fetching SAML SSO protocol details using SAML Raider.
5. Performing Attacks using WS-Attacker.

Here the proposed framework, WSASRESSO has been tested over Web Sites using SAML SSO protocol. A real time website experimented is a TCS Campus Commune test case which has been found to be vulnerable to Signature Wrapping Attack.

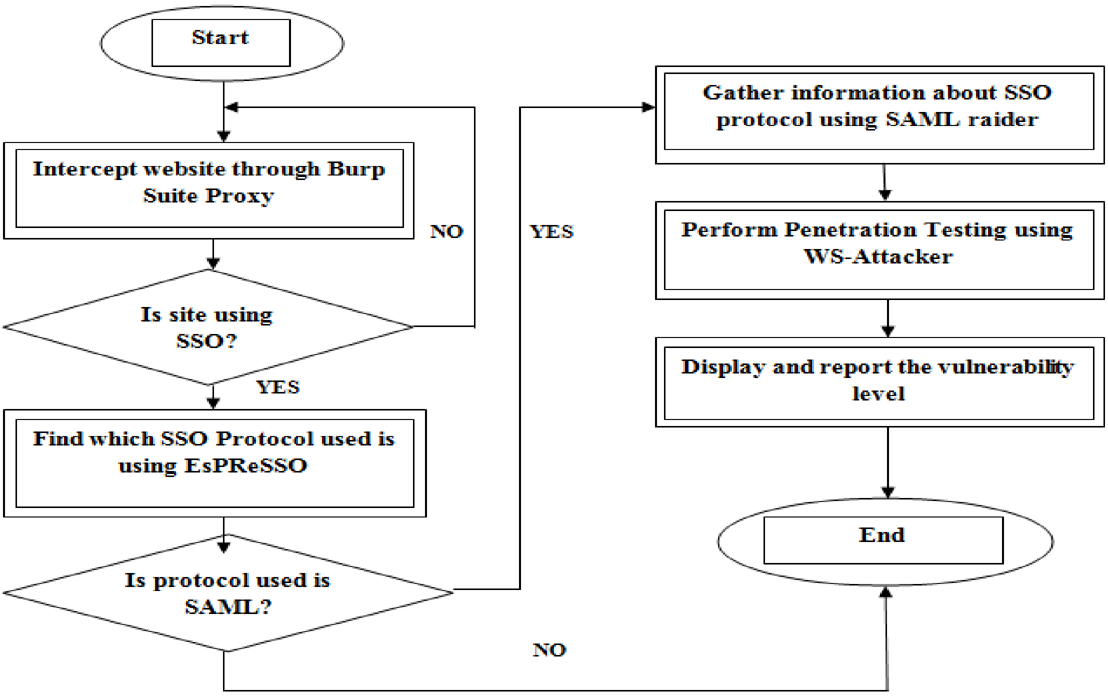
Start
Intercept website through Burp Suite Proxy
Is site using SSO?
NO
YES
Find which SSO Protocol used is using EsPReSSO
Is protocol used is SAML?
YES
NO
Gather information about SSO protocol using SAML raider
Perform Penetration Testing using WS-Attacker
Display and report the vulnerability level
End

Fig 7: Flow Chart of the System

8. RESULT ANALYSIS

The obtained results are evaluated by comparing with the existing systems; the results obtained are shown below.

8.1 Comparative result

The intercept of SAML Request and Response has been done using Burp Suite. With the fetched result; analysis of SSO protocols has been performed using EsPReSSO algorithm. Using SAML Raider, the infrastructure details have been fetched and SAML Signature Faking and SAML Signature Wrapping attacks been performed successfully. Table 1 shows the analysis of the proposed system and the existing system with various parameters. Thus the proposed system WSASRESSO shows a better result in an analysis of SSO Protocol.

Table 1. Result Analysis

Evaluation Parameter	**SSOScan**	**SAMLyze**	**WSASRESSO**
Extensibility	Not Possible	Not Possible	Easy to add Plugins
Complexity	High	High	Low
Number of SSO Protocols Recognized	1	1	7
Attack Performed	2	0	2
Attack type	Credential Misuse		Signature Based
Accuracy	Moderate	Low	Moderate
Integration with Web Proxy	Not Possible	Not Possible	Possible to work with Burp Proxy

8.2 Comparative Analysis

Thus the proposed work comprises of multiple components which have distinctive features as shown in Table 2 and 3. These components have been analyzed with different parameters and hence the proposed work has the capacity to incorporate all these features into a single architecture, which makes the analysis of SSO Protocol much easier.

Table 2. Comparative Analysis1

Functionality	**Burp Suite**	**EsPReSSO**	**SAML Raider**
Intercept Messages	✓	X	X
Categorize Protocol	X	✓	X
Capture Infrastructure Details	X	X	✓
Perform Penetration Testing	X	X	✓

Table 3. Comparative Analysis2

Functionality	**WS - Attacker**	**WSASRESSO**
Intercept Messages	X	✓
Categorize Protocol	X	✓
Capture Infrastructure Details	X	✓
Perform Penetration Testing	✓	✓

9. CONCLUSION AND FUTURE WORK

9.1 Conclusion

SSO has become one of the fastest and most familiar modes of authentication. In SAML based SSO mechanism vulnerability has been predominant factor for major attacks. Availability needs to be addressed before the deployment of the SSO protocol This work proposes a new approach, which provides a framework to evaluate SAML based SSO protocols using Burp suite extension with combination of EsPReSSO algorithm for identification of SSO protocols along with SAML Raider for fetching the SAML protocol infrastructure details and integration of WS-Attacker to perform black box penetration testing to find the strength of the SSO protocol. The performance is compared with the existing techniques as well as with the proposed components. The comparison indicates that the proposed work WSASRESSO provides better analysis over existing approach. The test results suggest that the considered TCS Campus Commune website which makes use of the SAML based SSO protocol in real time is vulnerable to Signature Wrapping Attack but is resilient to Signature Faking Attacks.

9.2 Future Work

More such real time data sets involving SAML based SSO can be considered in our future work. Further works have also been indicated in the following directions to facilitate the problem of analysis of SSO: Testing for another sort of attacks like Replay, etc. Metrics framed can be used to evaluate the system performance mathematically. Penetration testing over other SSO protocols can be done for finding out which one is better for SSO system.

10. REFERENCES

[1] Christian Mainka, Vladislav Mladenov, Tim Guenther, Jörg Schwenk, “Automatic Recognition, Processing and Attacking of Single Sign-On Protocols with Burp Suite,” Open Identity Summit, 2015.

[2] Yuchen Zhou, David Evans, “SSOScan: Automated Testing of Web Applications for Single Sign-On Vulnerabilities,” 23rd USENIX Security Symposium, 2014.

[3] R. Wang, S. Chen and X. Wang, "Signing Me onto Your Accounts through Facebook and Google: a Traffic-Guided Security Study of Commercially Deployed Single-Sign-On Web Services," Security and Privacy(SP), IEEE Symposium, 2012.

[4] Jacob Bellamy-McIntyre, Christof Luterroth, Gerald Weber, “OpenID and the Enterprise: A Model-based Analysis of Single Sign-On Authentication”. 15th IEEE International, 2011.

[5] Falkenberg. A., Maink. C, Somorovsky. J, Schwenk. J, "A New Approach towards DoS Penetration Testing on Web Services," IEEE 20th International Conference on Web Services (ICWS), 2013, pp.491,498.

[6] The OAuth 2.0 Authorization Framework (2015), ECMA International ECMA -262. Available: http://www.ecma-international.org/publications/files/EC MA -ST/Ecma-262.pdf.

[7] E. A. N. Sakimura, J. Bradley (2014), OpenID Connect Core 1.0 incorporating errata set 1, OpenID Foundation OpenID Connect 1.0. Available: http://openid.net/specs/openid-connect-core-1_0.html.s

[8] Mainka, Christian, and Mladenov, Vladislav and Feldmann, Florian and Krautwald, Julian and Schwenk, Jörg, "Your Software at my Service: Security Analysis of SaaS Single Sign-On Solutions in the Cloud," Proceedings of the 6th edition of the ACM Workshop on Cloud Computing Security, 2014.

[9] S. Gajek, M. Jensen, L. Liao, and J. Schwenk, "Analysis of signature wrapping attacks and countermeasures," IEEE International Conference on Web Services, 2009, pp. 575–582.

[10] J. Rzeniewicz (July 2015), “Log Me In with Facebook: Security Analysis of Facebook Connect,”.

[11] Microsoft, “Microsoft Account,” Available: https://account.microsoft.com/about.

[12] R. P. Scott Cantor, John Kemp (2005), “Assertions and Protocols for the OASIS Security Assertion Markup Language (SAML) V2.0,” Mar. Available: http://docs.oasis-open.org/security/saml/v2.0/.

[13] D. Fett, R. Küsters, and G. Schmitz (2014), “An Expressive Model for the Web Infrastructure: Definition and Application to the BrowserID SSO System,” IEEE Computer Society, 2014, pp. 673–688.

[14] JohnBarber, “SAMLyze,” Available: https://www.blackhat.com/us15/arsenal.html.

[15] Roland Bischofberger, Emanuel Duss, “SAML Raider - SAML2 Burp Extension,” Available: https://github.com/SAMLRaider/SAMLRaider.

[16] PortSwigger, “Burp Suite API,” Available: http://portswigger.net/Burp.

[17] Authentication SSO Protocol, Available: https://auth0.com/docs/sso/ single-sign-on.

[18] OpenID - A Descriptive Document, Available: http://openid.net/get-an-openid/what-is-openid.

[19] OpenID Specification, Available: https://openid.net/specs/openid-authentication-2_0.html.

[20] Security Assertion Markup Language (SAML) V2.0 Technical Overview http://docs.oasis-open.org/security/saml/Post2.0/sstc-saml-tech-overview-2.0.html.

[21] OASIS Security Services (SAML) TC https://www.oasis-open.org/committees/security/ipr.php.