ISSN (Print) : 2320 – 3765
ISSN (Online): 2278 – 8875

International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 2, Issue 8, August 2013

Analysis of Single Sign on for Multiple Web Applications

Anita Patil[1], Prof. Rakesh Pandit[2], Prof. Sachin Patel [3]

PG Student (M.Tech), Department of Information Technology, PCST, Indore, MP, India[1]

Asst. Professor, Department of Information Technology, PCST, Indore, MP, India[2]

Asst. Professor & HOD, Department of Information Technology, PCST, Indore, MP, India[3]

Abstract: In general, a coherent authentication strategy or a solid authentication framework is missing in recent authentication system. Over time this leads to a proliferation of applications, each of which comes with their own authentication needs and user repositories. At one time or another, everyone needs to remember multiple usernames and passwords to access different applications on a network. This poses a huge cost for the administration and support department accounts must be set up in each application for each employee, users forget their passwords, and so on. Single sign-on (SSO) is a session/user authentication process that permits a user to enter one name and password in order to access multiple applications. Through this paper we will discuss the basic sign on model and disadvantage of multi sign on system. Later on, the single sign on model will be presented, especially with the focus on the different SSO architectures; We will compare the SSO solution to the ACL with proxy signature.

Keywords : Proxy Signature, MD5,PKI, SAML

I. INTRODUCTION

Single sign-on (SSO) is a session/user authentication process that permits a user to enter one name and password in order to access multiple applications. The process authenticates the user for all the applications they have been given rights to and eliminates further prompts when they switch applications during a particular session. We will introduce the basic sign on model and describe the disadvantage of this multi sign on system. Later on, the single sign on model will be presented, especially with the focus on the different SSO architectures.

A. Characteristics of SSO include the following

i) Improved user productivity:

Users are no longer bogged down by multiple logins and they are not required to remember multiple IDs and passwords. Also, support personnel answer fewer requests to reset forgotten passwords.

ii) Improved developer productivity:

SSO provides developers with a common authentication framework. In fact, if the SSO mechanism is independent, then developers don't have to worry about authentication at all. They can assume that once a request for an application is accompanied by a username, then authentication has already taken place.

iii) Simplified administration. When applications participate in a single sign-on protocol, the administration burden of managing user accounts is simplified. The degree of simplification depends on the applications since SSO only deals with authentication. So, applications may still require user-specific attributes (such as access privileges) to be set up.

II. SINGLE SIGN-ON

"Single sign-on (SSO) is the mechanism whereby a single action of user authentication and authorization can permit a user to access all computers and systems where that user has access permission, without the need to enter multiple passwords." [2] For short, Single Sign-On is the technology where a user only need to authenticate him/her self once, then has the ability to access other protected resources without having to re-authenticate.

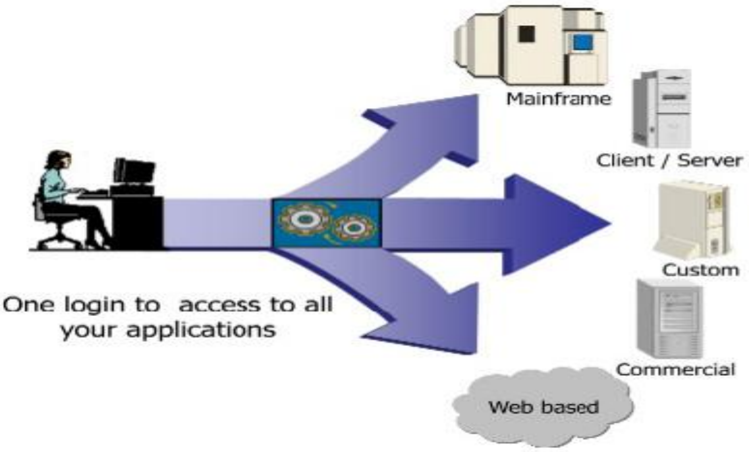

Fig 1. The Single Sign-On

*B. **The Benefits of Single Sign-On***

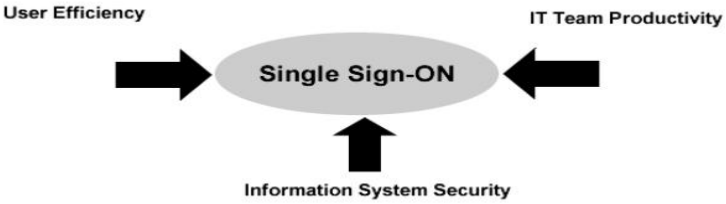

Fig 2. The benefits of Single Sign-On

i) **From the user view:**
In an SSO environment, users only need to authenticate themselves once. This effectively solves the annoying stop-and-go problem which is caused by multiple login requests. Best of all, the SSO solution frees users from remembering a large number of identities and associated passwords.
ii)**From the enterprise view:**
For the enterprise, SSO delivers a tremendous return on their investment.
Potential Increase in Security
With only one password to remember, it is more reasonable for the user to choose a single complex and more secure password instead of using multiple simple and insecure passwords. This potentially increases the system security.
Improve Productivity
Employee productivity is dramatically improved, with less time users spend logging into multiple applications and recovering the forgotten passwords.
Reduction in Costs
"Meta Group estimates 33% reduction in help desk volume when using an enterprise Single Sign-On solution." By reducing the number of passwords the user must remember, SSO effectively reduces the password-related workload to the helpdesk and lowers the costs associated with managing passwords across multiple distributed applications.

III. DIFFERENT FLAVORS OF SINGLE SIGN-ON

There are three main types of single sign-on: web SSO, Legacy SSO and Federated SSO.

A. Web Single Sign On

Wed-based SSO is a widely deployed single sign-on technology sometimes also called web access management. It enables a user to provide the credentials, if authentication succeed, it will establish a relationship of trust that grant user the access to all web resources for which he/she have permissions. [3]

B. Legacy Single Sign On

Legacy SSO is also called Enterprise SSO. Like web SSO, legacy SSO is also a technology designed to manage multiple login to target applications after a single authentication event. It has a very similar structure to the web SSO. While web SSO only manages the web-based service, legacy SSO extends the SSO functionality to the traditional legacy applications and network resources (windows GUI based applications, for example), typically within an enterprise's internal network. [3]

C. Federated Single Sign On

Federated SSO is similar to the web SSO but has a much broader concept. It uses Simple Object Access Protocol (SOAP) and Security Assertion Markup Language (SAML) to enables users to sign on once into a member of the affiliated group of organizations, then seamlessly access all the web sites within that trusted federation without requiring re-authentication. [4] The main advantage of a federated SSO is extending the SSO environment from a user's home domain to other foreign domains. Federated SSO allows the enterprises to maintain the control of its local services and expose these resources to a larger class of users not directly administered by it. Mostly, this solution is used by the businesses to build a complete framework for secure B2B and B2C e-business. The most famous federated SSO is the Liberty Alliance Project.

IV. DIFFERENT SSO ARCHITECTURE

Today, there are various SSO architectures, each with different properties and underlying infrastructures. Jan De Clercq, the Security Consultant of HPCI Technology Leadership Group, defined two main architectures to get Single Sign-On: solutions that deal with one set of user credentials and solutions that deal with multiple sets of user credentials. [5] Now, We will present each of the architecture in more detail and give a light evaluation and comparison of these models.

A. Single Sign-On With Multiple Sets of Credentials

i) Secure Client-side Credential Caching:

The Secure Client-Side Credential Caching mechanism is a client-based SSO solution which keeps all the authentication information into a client-side credential storage. It allows the end-users to authenticate themselves once, and then has the system automatically provide the information for subsequent request without the users' intervention. If the credentials are valid, the user will be authenticated transparently to the other application servers. A good example of the Secure Client-Side Credential Caching mechanism based SSO solution is Credential Manager that Microsoft offers in Windows Server 2003 and Windows XP. [6]

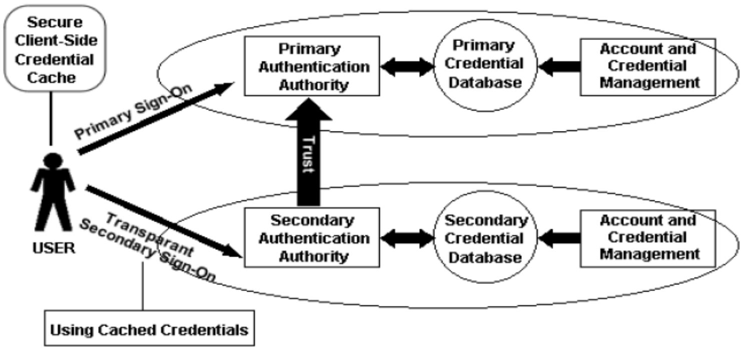
Secure Client-Side Credential Cache
Primary Sign-On
Primary Authentication Authority
Primary Credential Database
Account and Credential Management
Trust
USER
Transparent Secondary Sign-On
Secondary Authentication Authority
Secondary Credential Database
Account and Credential Management
Using Cached Credentials

Fig 3 Client-side Credential Caching Mechanism

This solution requires a high secure credential cache resides on client-side. It is very crucial to store the cached credentials securely as the credentials may be used to access some sensitive information or confidential web service. So, it's not recommended to be used from portable client devices or some Operation Systems with a bad security reputation. Since all the authentication data is stored in the client-side credential cache, this architecture has little flexibility. The user will get sign-on problems if he/she is not using his usual workstation (when travelling, for instance). Although it is relatively simple for the user to set up and configure, every time a new application server is added, the new authentication information should be added into the client-side credential cache. This makes the solution a little discommodious.

ii) Secure Server-side Credential Caching:

The Secure Server-Side Credential Caching mechanism is also called the server-based SSO solution. The same as the Secure Client-side Credential Caching architecture, this approach also uses a central repository to store all the authentication information. But in this architecture, the cache is located on server-side. It uses a central server to take on the task of administering all the different passwords and providing the needed information directly to the application asking for them. The good examples of this SSO solution are Tivoli Secure Way and ETrust SSO. [7]

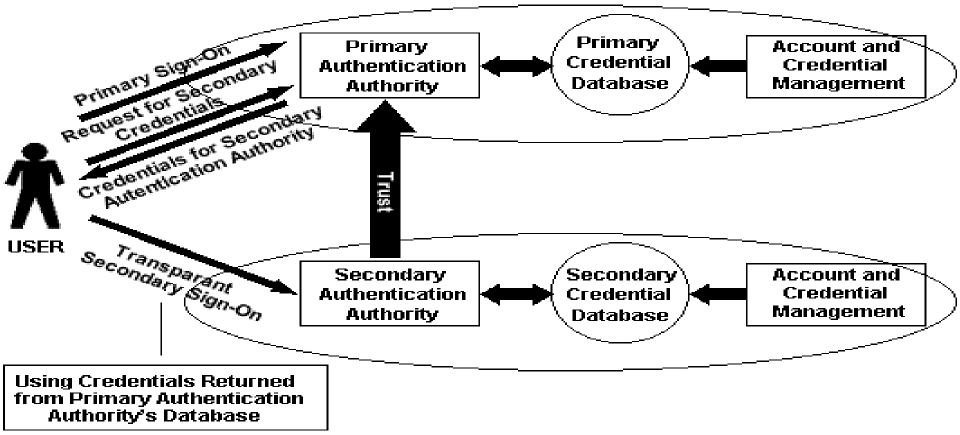
Primary Sign-On
Request for Secondary Credentials
Credentials for Secondary Autentication Authority
Primary Authentication Authority
Primary Credential Database
Account and Credential Management
Trust
USER
Transparant Secondary Sign-On
Secondary Authentication Authority
Secondary Credential Database
Account and Credential Management
Using Credentials Returned from Primary Authentication Authority's Database

Fig. 4 Server-side Credential Caching based SSO

In a secure server-side credential caching mechanism, the primary credential database contains the user's primary credentials as well as the mappings between the primary credentials and the secondary credentials. The secondary credential database only keeps a copy of the secondary credentials. It is important that in this approach, we should keep the mappings between these credential databases synchronized. There are three main approaches to achieve the data synchronization:

1. Integrating the credential synchronization services into the primary credential database.
2. Using an external software to handle the credential synchronization process.
3. Administrators perform the synchronization by themselves (not recommended).

Depending on the need of credentials synchronization, it is necessary to set a trust relationship between the secondary authentication authorities and the primary authentication authority.

iii) Single Sign-On With Single Set of Credentials:

SSO solution with single set of credentials is often implemented by some services which provide the management of authentication information. This solution uses a standardized scheme to handle authentication which is located on a centralized authentication infrastructure. A major feature of this single set of credentials SSO architecture is the rather homogeneous environment which means: using a single account naming format and authentication protocol which are supported by every entity in the whole network system. There are two flavors of SSO with single set of credentials: PKI-based SSO and Token-based SSO. Both of them only use a single set of credentials which is recognized by many different authentication authorities.

iv) PKI-based Single Sign-On:

The PKI-based SSO solution makes use of pubic key cryptography to authenticate users. It requires the use of a Certification Authority (CA) system to issue and manage users' digital identities. [8]

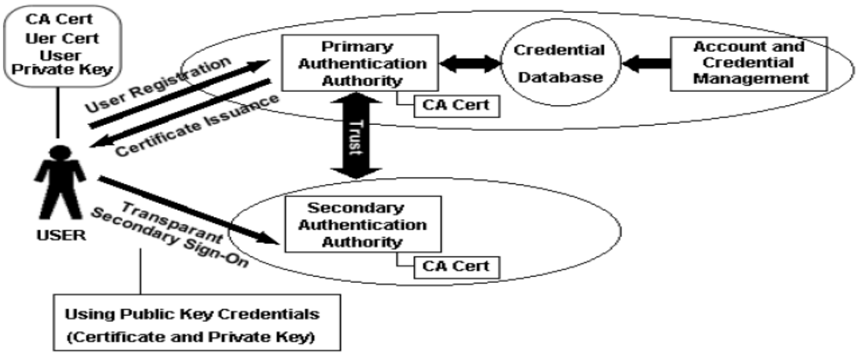
CA Cert
Uer Cert
User
Private Key
User Registration
Certificate Issuance
USER
Primary Authentication Authority
CA Cert
Credential Database
Account and Credential Management
Trust
Secondary Authentication Authority
CA Cert
Transparant Secondary Sign-On
Using Public Key Credentials (Certificate and Private Key)

Fig 5. The PKI-based SSO model.

The user first identifies him/her self to a trusted authentication authority and gets a public key certificate. In a subsequent authentication request, whenever he/she tries to access a protected resource, the user creates a token, includes its own digital certificate (public key in it) and signs it with the private key. The target server receives the request and contacts with the CA to proof the user's identity. Since the secondary CA's certificate is issued by the primary CA, there is a trust relationship between the secondary CA and the primary CA which enables any secondary CA will accept the certificate issued by the primary CA. The most famous PKI-based SSO product is Entrust Get Access. First, the length of the private key is a big strongpoint but also the possible problem to the PKI-based solution. The private key is a very long series of random binary data (usually represented in Base64) need to be closely protected. It is hard to write the private keys down on paper or keep it in mind, but easy to transfer over a network. So, it is not easy to steal somebody's private key

v) Token-based Single Sign-On:

In a token-based SSO architecture, the user will receive a temporary token after he/she has logged into the primary authentication authority. This temporary token will be used to proof the user's authenticity to every service he/she requests. There is a trust relationship between the primary authentication authorities and the secondary authentication authority. This relationship enables the user could be authenticated to a secondary authentication authority with the temporary token issued from the primary authentication authority without a re-authentication. A typical example for this authentication strategy is the Kerberos authentication protocol. MS Passport is another widely used token-based SSO solution.

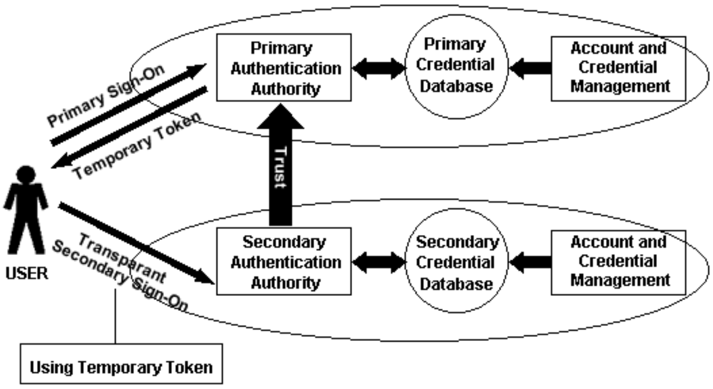
Primary Sign-On
Temporary Token
USER
Primary Authentication Authority
Primary Credential Database
Account and Credential Management
Trust
Secondary Authentication Authority
Secondary Credential Database
Account and Credential Management
Transparant Secondary Sign-On
Using Temporary Token

Fig 6 Token-based Single Sign-On

B. Analysis and Comparison

The following table will make a comparison of these different kinds of architectures against their advantages and disadvantages.

Sr No.	Different architeture of SSO	Prons	Cons
1	Secure Client -side credential caching	Can deals with different sets of credentials	Requires a very secure client-side credential cache
		Does not require a homogeneous authentication infrastructure environment	Multiple sets of credentials complicate life of user and administrator.
		Relative easy to set up	Little flexibility
		Minimum modification of the application	Lack of industry standardization in password specifications
			The end-user controls password management. No central management control
2	Secure Server-side Credential Caching	Can deal with many different credentials	Requires a credential synchronization mechanism
		Does not require a homogeneous authentication infrastructure environment	Lack of industry standardization in password specifications
		Few user interruptions when new application added	Multiple sets of credentials complicate life of user and administrator.
			Need high available authentication server
3	Token based SSO	Single set of credentials simplifies life of user and administrator	Often rely on symmetric cryptography, relative less security
		No user interruption when new application is added.	Require extra software on application.
			Need high available authentication server
4	PKI- based SSO	Single set of credentials simplifies life of user and administrator.	Complex certificate validation logic requires a lot processing on the client side.
		High level secure by using asymmetric cryptography	All services and applications must be PKI-enabled
		No user interruption when new application is added	Little flexibility or extra cost to store private keys
			Need trusted Certificate Authority

Table 1. Different types of different Single Sign on

SSO with multiple sets of credential provides the advantage that the user only needs to remember one password to log on multiple systems.
‘

V. SSO WITH PROXY SIGNATURES

In a proxy signature scheme, a party called the designator or original signer delegates to a party called the proxy signer the right to sign messages inside an specific message space. The basic (informal) security properties of proxy signatures stated in [9] are the following:

A. Strong Unforgeability

The original signer and third parties who are not designated as proxy signers cannot create a valid proxy signature.

B. Variability

From proxy signature a verifier can be convinced of the original signer's agreement on the signed message either by a self-authenticating form or by an interactive form. (The proxy signer can only sign messages inside a message space specified by the designator) Strong Identifiability: Anyone can determine the identity of the corresponding proxy signer from a proxy signature.

C. Strong Undesirability

Once a proxy signer creates a valid proxy signature for an original signer, the proxy signer cannot repudiate his signature creation against anyone. Although proxy signatures have been extensively studied, the first formal security definitions that capture the above properties and provably secure constructions were only introduced in [4]. Apart from the basic algorithm in digital signature schemes (KeyGen(1n), Sign(sk; m) and V erify(pk; m; sig)), proxy signature schemes comprise four more components:

D. Delegation Algorithms

D(pki; ski; j; pkj ; !) is run by the designator, who inputs his key pair (pki; ski) along with a proxy signer ID i and public key pki. It also inputs a descriptor of the delegated message space. P(pkj ; skj ; pki) is run by the proxy signer in order to obtain the proxy signing key skp and takes as input the proxy signer's key pair and the designator's public key.

E. Proxy signing algorithm

PS(skp; m) takes as input a proxy signing key and a message m 2 !, outputting a proxy signature psig. Proxy verification algorithm: PV(pk; psig; m) takes as input a public key pk, a proxy signature psig and a message m, outputting 1 if the signature is valid for m and pk. Otherwise, it outputs 0.

F. Proxy identification algorithm

ID(psig) takes as input a proxy signature and outputs the identity of the proxy signer.The security definitions for proxy signatures are formalized in [4], and it is shown that proxy signature schemes can be obtained from using MD5 algorithm. Moreover, it is shown that aggregate signature schemes [5] can be used in such constructions in order to obtain shorter signatures. One should notice that it is possible to add a timespan parameter to the proxy delegation algorithm, allowing the designator to specify a timespan during which the proxy signer may use the key. This new functionality can be added by a simple/,modification of the Delegate-by-certificate and Aggregate Signature Based proxy signature schemes in [4], requiring only minor (trivial) modifications in the security proofs. The altered delegation algorithm is denoted by D(pki; ski; j; pkj ; !; t), where t is the time span information.

VI. COMPARISON OF PROXY SIGNATURE SSO AND OTHER SINGLE SIGN-ON SYSTEM

Proxy Signature Based and Previous Single Sign-on, both of them has its own strengths and weaknesses:

Sr No	**comparison categoey**	**Proxy Signature SSO**	**Previous Single Sign-on**
1	Implement	Easy to implement, no client software, limited server-side agents.	Depend on different architectures and technologies might be very complex.
2	Process	Simply changing all applications to use the same passwords.	Using single username and password to login to one specific server which will take charge of the client authentication to all the other servers.
3	Login times	The user is still required to login to each system using the same passwords.	The user only needs to login to the primary authentication authority once. (primary sign-on)
4	Manage credential data	Managing passwords only	Using specific protocols to manage the client authentication and the secrete information.
5	Password policy	Use MD 5 Algorithm to provides authentication and confidentiality	Only one password, but can use strong password policy to ensure confidentiality only.
6	Security	Credentials are kept identical on all platforms -- “Key to the Kingdom” argument	Single point of failure, may also has the “Key to the Kingdom” problem

Table 2. Comaprision between Proxy Signature SSO and previous SSO

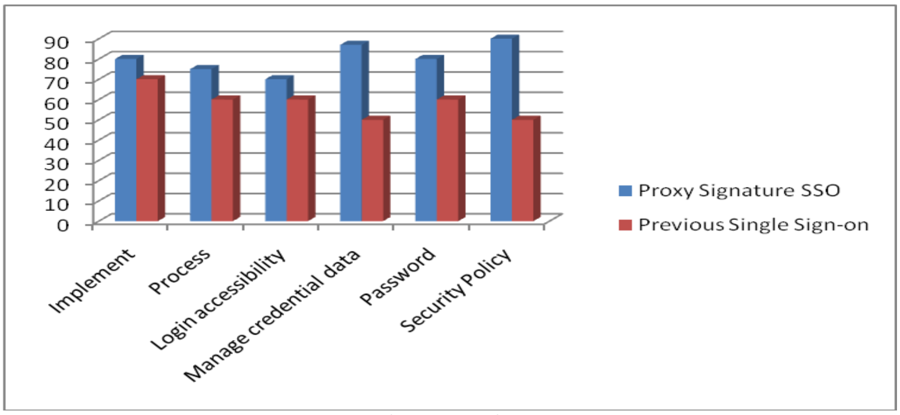

Fig 7. Caparitive Performanace of Single sign on system

VII. CONCLUSION

We presented a new approach for practical efficient and secure single sign-on frameworks based on proxy signature schemes. The proposed framework provides seamless and transparent single sign-on without undermining overall network security and without requiring any online communications between service providers and the identity provider. Additionally, it allows for grained access control without any increase in the protocol's computational or communication complexity. Moreover, it others simple access policy and user revocation management while providing nice forensic and audit data by building on common proxy signature strong unforgeability and undeniability properties. Our framework is also apply public key cryptography with MD5 techniques to the problem of practical single sign-on. The results represent an important step towards the formalization of single sign-on and user authentication protocols, and the construction of provably secure schemes for these practical applications.

ACKNOWLEDGMENT

Our thanks to the experts who have contributed towards development of the template.

REFERENCES

[1] Kapil Singh ,"xAccess: A Unified User-Centric Access Control Framework for Web Applications" IBM T.J. atson Research Center, 2012 IEEE Network Operations and Management Symposium (NOMS)
[2] The Open Group, "Introduction to Single Sign-On", 20 May,1998 http://www.opengroup.org/security/sso/sso_intro.htm
[3] Oasis, "Security Assertion Markup Language (SAML) V2.0," 15 March
[4] Sidharth, N.; Jigang Liu; , "A Framework for Enhancing Web Services Security," Computer Software and Applications Conference, 2007. COMPSAC 2007. 31st Annual International, vol.1, no., pp.23-30, 24-27 July 2007 doi: 10.1109/COMPSAC.2007.22
[5] Jan De Clercq, "Single Sign-On Architectures", 2002 http://www.esat.kuleuven.ac.be/cosic/seminars/slides/SSO.pdf
[6] Jan De Clercq, "Introducing Credential Manager", Journal of Applied Non-Classical Logics, special issue on Logic and Information Security ,December 2003. http://www.windowsitpro.com/WindowsSecurity/Article/40573/40573.html
[7] Computer Associates, "eTrust Single Sign-On", 2005 http://www3.ca.com/Solutions/Product.asp?ID=166
[8]Diana Berbecaru, Antonio Lioy and Marius Marian, "PKI-Based System Management", 26 October, 2001 http://www.cercom.polito.it/Publication/Pdf/132
[9] Bernardo Machado David1, Anderson C. A. Nascimento, Rafael Tonicelli1" A Framework for Secure Single Sign-On"2009
[10] A. Armando, R. Carbone, and L. Compagna." LTL Model Checking for Security Protocols",in Journal of Applied Non-Classical Logics, special issue on [11]Logic and Information Security, pages 403{429. Hermes Lavoisier, 2009. M. Cova, V. Felmetsger, and G. Vigna, "Vulnerability Analysis of Web Applications," in *Testing and Analysis of Web Services*, L. Baresi and E. Dinitto, Eds. Springer, 2007.

BIOGRAPHY

Anita R. Patil is a Student of Patel College of Science and Technology, Indore, India .She has completed B.Tech in Information Technology from the Dr. B.A.T.University ,India. She has a interest to doing a research in the area of information security management. And Distributed computing.