Privacy in Cloud Computing Through Identity Management

Bharat Bhargava[1], Noopur Singh[2], Asher Sinclair [3]

[1]*Computer Science, Purdue University*

[2]*Electrical and Computer Engineering, Purdue University*

West Lafayette, IN 47907

[3]*United States Air Force*

[1]bbshail@cs.purdue.edu

[2]singh91@purdue.edu

Abstract— **The migration of web applications to Cloud computing platform has raised concerns about the privacy of sensitive data belonging to the consumers of cloud services. The traditional form of security tokens like username/password used to access cloud services are prone to phishing attacks and hence do not provide complete security. In this work we propose to extend the Microsoft's CardSpace identity management tool, to include more robust security tokens using the zero knowledge proof concept. These security tokens are in the form of SAML token supported by Windows Communication Foundation (WCF) and hence can prove interoperable with the existing security platforms.**

Keywords— **Cloud Computing, Privacy, Identity Management, Microsoft CardSpace, Zero Knowledge Proof.**

I. INTRODUCTION

Service providers use infrastructure provided by Cloud Computing provider to provide their services to their customers. While using these services, consumers provide the service providers with sensitive data such as civil ID (name), SSN number, credit card information in order to have access to these online services. Current privacy laws require cloud computing service providers to implement varied security measures depending on the nature of the information [1]. However, consumers can not verify that a provider of a service conform to the privacy laws and protect their digital identity. Given this, a consumer has to decide as what type of personal information they could provide. For instance, a "Twitter breach of information stored on Google Apps" shows the implications of leakage of private information in cloud security. The July 15 disclosure by Twitter revealed that a hacker had accessed a substantial amount of company data stored on Google Apps. First the hacker hijacked a Twitter employee's official e-mail account. Then, he took advantage of poor password practices, Hotmail's inactive account feature and personal information on the Web to pinch hundreds of Twitter documents [2]. This probes the weakness of the username/password security token used by most service providers to authenticate consumers, which leaves the consumer vulnerable to phishing attacks.

There is a need for a solution to address the above problem in form of an Identity Management (IDM) solution [20]. The solution should help the consumer to make a proactive choice about how and what personal information they disclose, control how their information can be used, cancel their subscription to the service, and monitor to verify that a service provider applies required privacy policies. This IDM should be able to help consumers manage their various digital identities and the various username/password associated with each service provider, centrally.

II. RELATED WORK

This section discusses three known identity management tools.

A. OpenID

With OpenID a user uses one username and one password to access many web applications. The user authenticate to an OpenID server to get his/her OpenID and use the token to authenticate to web applications.

A user of OpenID does not need to provide a service provider with his credentials or other sensitive information such as an email address.

OpenID is a decentralized authentication protocol. No central authority must approve or register service providers or OpenID Providers. An end user can freely choose which OpenID Provider (OP) (OpenID Authentication server on which a service provider relies to assert the authenticity of the identity of the consumer) to use, and can preserve their Identifier if they switch OpenID Providers. [3]

OpenID is highly susceptible to phishing attacks, as the whole OpenID structure hinges on the URL routing to the correct machine on the Internet i.e. the OpenID Provider. A user who visits an evil site (through conventional phishing or DNS cache poisoning), sends the imposter service provider her URL. The provider consults the URL's content to determine the location of her OP (OpenID provider). Instead

of redirecting the user to the legitimate OP, it redirects her to the Evil Scooper site. The Evil Scooper contacts the legitimate OP and pulls down an exact replica of its login experience (it can even simply become a "man in the middle"). Convinced she is talking to her OP, the user posts her credentials (username and password) which can now be used by the Evil Scooper to get tokens from the legitimate OP. These tokens can then be used to gain access to any legitimate Service Provider. [4]

B. PRIME (Privacy and Identity Management for Europe)

PRIME, is a single application — the PRIME Console — that handles user's personal data. It handles management and disclosure of personal data for the user (e.g. informed consent of the user to be established and privacy risks to be conveyed, through a user interface) and is the interface to the PRIME technology.

The Console requires installation and configuration. The user manages her personal data using the console, discloses personal data, and checks the proper handling of her data by the various services she requires. The client application mirrors the server application used by the service provider. [5]

A major challenge for a large scale adoption of PRIME technology is that it requires both individuals and service providers to implement the PRIME middleware, on both sides. Another prerequisite for large scale adoption is interoperability. PRIME, stands no chance unless it allows interoperability with existing applications and other identity management systems. This calls for standardization. [6]

III. ADOPTION OF MICROSOFT'S CARDSPACE AS A VIABLE IDM FOR PRESERVING PRIVACY

This section discusses the architecture of CardSpace and its security vulnerabilities.

A. Overview of WS-Federation Protocol on which CardSpace is built [15]

WS-Federation consists of the following standards:

- *WS-Trust*: Trust define relationship between two parties where one party believes statements (claims) made by the other party;. It is based on evidences, recommendations, previous experiences, and personal risk tolerance. WS-Trust provides the foundation for federation by defining a service model, the Security Token Service (STS), and a protocol for requesting/issuing these security tokens.
- *WS-SecurityPolicy:* WS-Policy defines a framework for allowing web services to express their constraints and requirements as policy assertions.
- *WS-Security:* WS-Security describes security within the SOAP (Simple Object Access Protocol) message itself, which includes authentication, signatures, and encryption. [16]

These standards provide a basic model for federation between Identity Providers and Relying Parties. These specifications define mechanisms for codifying claims (assertions) about a requestor as security tokens which can be used to protect and authorize web services requests in accordance with policy.

WS-Federation extends this foundation by describing how the claim transformation model inherent in security token exchanges can enable richer trust relationships and advanced federation of services. This enables high value scenarios where authorized access to resources managed in one security domain can be provided to security principals whose identities and attributes are managed in other security domains.

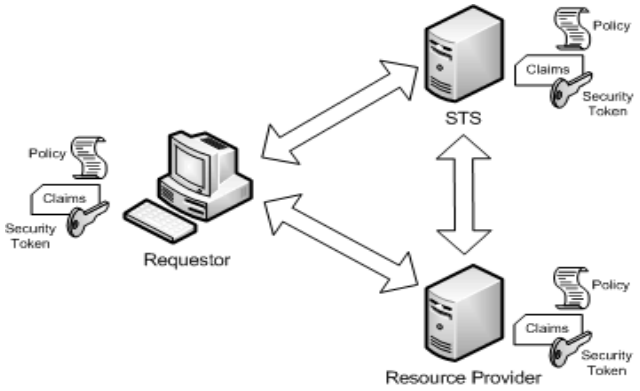

Figure 1. Security Token Service (STS) Model [15]

Fig.1 illustrates the use of Security Token Service (STS) by WS-Trust. Each arrow represents a communication between the both participants. Each participant has its own policies which for establishing trust, these policies combine to determine the security tokens and associated claims required to communicate with the other party. From the Requestor's perspective the communication flow starts with the identification of a web service, or a Resource Provider, that the requestor wishes to access. The Requestor queries the Resource Provider for its policies to determine the security requirements to use the resource. Using WS-SecurityPolicy expressions, the Requestor can check its own capabilities to determine if requestor has a security token that meets the requirements to access the Resource Provider. If the requestor does not have an acceptable token it might be able to request one from an appropriate STS which can also be identified in the Resource Provider's policy. Each STS has its own associated policy and being a web service. The Requestor can query the STS to determine the security requirements for requesting a particular type of token for use. Figure 1 depicts the STS functioning in the role of an Identity Provider (IdP). The primary function of an STS in this role is to issue identity tokens that contain claims about a security principal that correspond to the Requestor. A Resource Provider is frequently referred to as a Relying Party (RP) to indicate that it relies upon tokens issued by an STS to grant/deny access to the resources it controls.

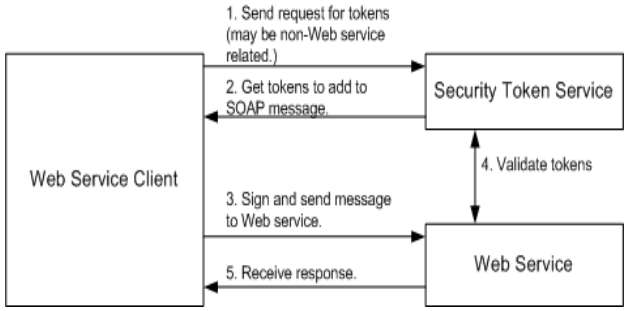

Figure 2 Typical message flow [16]

WS-Security encapsulates the security interactions as seen in Figure 2 within a set of SOAP Headers. WS-Security handles credentials management in two ways: (1) by Username Token, or (2) provides a place to provide binary authentication tokens such as Kerberos Tickets and X.509 Certifications.

B. Overview of Microsoft CardSpace

Windows CardSpace is an Identity-metasystem which provides a way, for managing multiple digital identities of a user [7]. It is claims based access platform/ architecture, developed for windows XP. It uses a plug-in for Internet explorer 7 browser [8].

The CardSpace is designed to comply with the seven Laws of identities by Kim Cameron of Microsoft [9].

In CardSpace every digital identity transmitted on the network contains some kind of security token. A security token consists of a set (one or many) claims, such as a username, a user's first name, last name, home address and even more sensitive information such as SSN, credit card numbers. These security tokens provide information in order to prove that these claims really do belong to the user who's presenting them. CardSpace implements an intuitive user interface for working with digital identities. Users use a visual "information card", Infocard, to make good decisions about use of their digital identities. In the identity system three parties are involved [Fig.3]:

- *Identity provider (Idp):* It issues digital identities (as trusted third-party). For example, a credit card provider might issue digital identities (security tokens) enabling payment. Even individuals can be Idp if they use self-issued identities like signing on websites, using username and password.
- *Relying Parties (RP)*: It requires identities to provide a service to a user for example, a web site.
- *Subjects (service requestor):* they are individuals and other entities about whom claims are made.

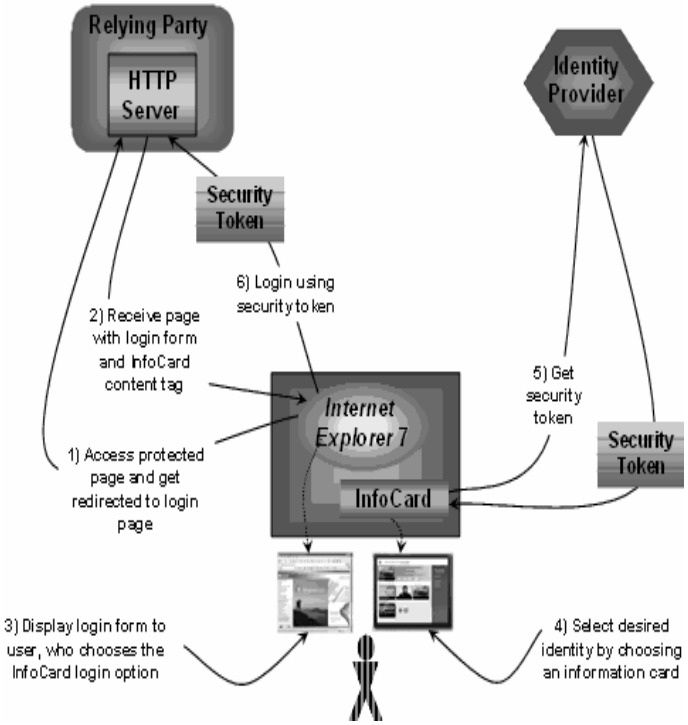

Figure 3 *CardSpace Model of Id*entity Management [7]

Figure 3 illustrates the CardSpace model of identity management, CardSpace makes use of "open" XML-based protocols, including Web services (WS-*) protocols and SOAP. The following steps describe message flows of the CardSpace framework: [10]

(1) *CEUA (CardSpace enabled user agent/service requestor) → RP* HTTP gets Login HTML Page Request: The CardSpace enabled user agent, CEUA (CardSpace enabled browser) requests a service from the relying party, using HTTP.

(2) *RP → CEUA*

HTML Login Page and InfoCard Tags (XHTML or HTML object tags): The RP identifies itself using a public key certificate and declares itself as a CardSpace enabled RP using XHTML or HTML object tags, i.e. a CardSpace enabled website or service provider.

(3) *CEUA ↔ RP*

CEUA retrieves security policy via WS-Security Policy: If the RP is card enabled, the CEUA obtains the RP's security policy described using WS-Security Policy. The policy is retrieved using WS-Metadata Exchange Protocol. This policy includes security token formats the RP will accept, the claims that must be contained in the tokens, and Idp (identity provider) that are trusted to makes such assertions, in order for this user to be granted the service.

(4) *CEUA ↔ User*--User picks an InfoCard:

In this step the User matches the RP's security policy with an appropriate InfoCard (containing the type of security token required by the RP). Which satisfies the RP's policy.

After the user selects an Infocard, the CEUA initiates a connection with the Idp that issued the Infocard.

(5) *CEUA* ↔ *IdP*-- User Authentication:

The user performs authentication process with the Idp, either using username/password login or using self-issued InfoCard. This is done for the user to prove the ownership of the InfoCard being used.

(6) *CEUA* ↔ *IdP*

CEUA retrieves security token via WS-Trust:If the authentication is successful the user requests the Idp to provide a security token which holds an assertion of the truth of the claims listed within the selected InfoCard. The CEUA obtains the security token using WS-trust.

(7) *CEUA* → *RP*--CEUA presents the security token via WS-Security: Finally the CEUA forwards the security token to the RP using WS-Security.

(8) *RP* → *CEUA*: Welcome, you are now logged in:If the RP is able to verify the security token, the service is granted to the user

C. Security Vulnerabilities and limitations of the CardSpace [10]

We discuss in the following three main limitations of Cardspace.

1. *User's Judgements of RP Trustworthiness:*

In the CardSpace framework, the user is prompted for its consent to be authenticated to an RP using a particular InfoCard, the user makes a judgment regarding the trustworthiness of the RP (step 2). Microsoft recommends that the user should only make use of a high assurance certificate (referred to as a "higher-assurance" certificate) such as an X.509 certificate. However, most users do not pay much attention when they are asked to approve a digital certificate, either because they do not understand the importance of the approval decision or because they know that they must approve the certificate in order to get access to a particular website. RPs without any certificates at all can be used in the CardSpace framework (given user consent), and this leads to a serious risk of a privacy violation. This security vulnerability breaks the 3rd law of Microsoft's own laws of identity (which is the law of Justifiable Parties). The law states that the disclosure of identifying information should be limited to trusted parties (i.e. parties having a necessary and justifiable place in a given identity relationship). The minimum amount of identifying information must be disclosed. Even if the RP presents a higher-assurance certificate, the user still needs to rely on an Idp who is providing that certificate to the RP and the user need to trust the Idp. Therefore, higher-assurance certificates do not solve this problem completely.

2. *Reliance on a Single Layer of Authentication:*

The security of the CardSpace identity metasystem relies on the authentication of the user by the IdP (step 5). In a case where a single IdP and multiple RPs are involved in a single working session, which we expect to be a typical scenario, the security of the identity metasystem within that working session will rely on a single layer of authentication, that is, the authentication of the user to the IdP. This user authentication can be achieved in a variety of ways (e.g., using an X.509 certificate, Kerberos v5 ticket, self-issued token or password).However, it seems likely that, in the majority of cases, a simple username/password authentication technique will be used. If a working session is hijacked (e.g., by compromising a self-issued token) or the password is cracked (e.g., via guessing, brute-force, key logging, or dictionary attacks), the security of the entire system will be compromised. Cardspace is a proprietary of Microsoft. Its protocols are not standards. Although, Windows is a widely used operating system, and so the solution can't succeed unless its is adopted as a standard. Its users are limited to using the Cardspace technology, with only Cardspace enabled RP's.

IV. IMPROVING THE SECURITY OF CARDSPACE

CardSpace replaces Password-Based Web logins (preventing Phishing), with the use of digital security certificates/ token. However, it has its own security limitations. For instance, little can be done if the security of the entire system is compromised by cracking the user's login to the CardSpace. We propose the use of Zero-Knowledge Proofing (ZKP), Selective Disclosure and Anonymous Credential to minimize the affects of the limitation. The goal is to prevent the need to reveal the actual values of the claims to any party within the CardSpace framework, this way no party will have to trust any other party to the level that it has to reveal the actual values of the claims to it.

A. Zero-Knowledge Proofing, Selective Disclosure and Anonymous Credential.

In an identity management system, two parties negotiate to establish trust through sensitive data exchange. The negotiation process should protect user's private information. The private data could be personal data pertaining to the user (attributes of user), certificates, anonymous credentials of the user, or private keys. Traditional certificates and tokens offer a weak trust model. The negotiation and data exchange process should protect the user's real identity by using a more advanced cryptographic private-certificate-based mechanism such as selective disclosure and zero knowledge proofs (ZKPs) [11].

The ZKP approach allows to prove a claim or assertion without actually disclosing any credentials. A solution using a ZKPworks as follows: a service requires a user to be over 18. The user wants to satisfy the relying party's technical policy but tell the party nothing or as little as possible. He need not to

reveal his date of birth, just needs to somehow prove being over 18. This proves something without revealing all Fig.4.

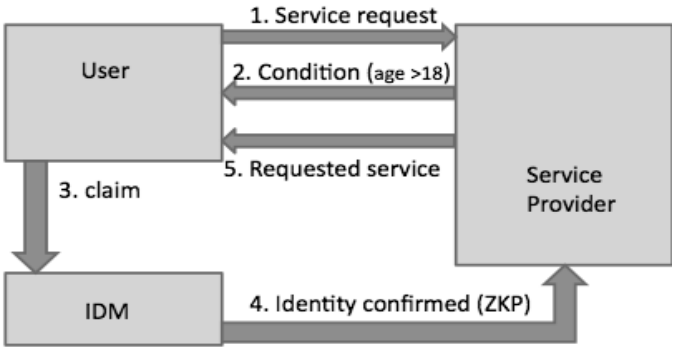

Figure 4 Use of ZKP during Negotiation [14]

Figure 4 illustrates the use of ZKP during negotiation. There are ZKP several schemes for different problems such as Fiat-Shamir proof of identity protocol: [12]. The steps of the protocol are:

1. A trusted centre chooses $n = pq$, and publishes n but keeps p and q secret.
2. Each prover A (service requestor) chooses a secret s with $gcd(s,n)=1$, and publishes $v=s^2 \bmod n$.
3. A proves knowledge of $\boldsymbol{s}$ to B by repeating:
   (a) A chooses random r and sends $r^2 \bmod n$ to B.
   (b) B chooses random e in {0,1}, and sends it to A.
   (c) A responds with $a=r.s.^e \bmod n$
   (d) B checks if $a^2 = v^e r^2 \bmod n$. If A follows the protocol and knows $\boldsymbol{s}$, then B's check will always work. If A does not know s, then they can only answer the question with probability 1/2.

The value of n should be digitally signed by the Idp by including it within the security token for example: XML-signature within a SAML assertion.

In the Selective Disclosure protocol the data exchange is performed such that the user reveals certified data in a data minimizing (minimal/Selective disclosure of PII-Personally Identifiable Information) approach. The approach uses predicates over attributes in addition to simple (type, value) pairs. For example, one may state that their monthly income is greater than or equal to stated constant value, such as greater than (monthly income $ 4000). Predicates over data are part of a logical formula that makes more general statements about identity associated with a party. A set of predicates for making data minimizing statements, such as =, ≠, <, ⩽,>, ⩾, can be embedded in the SAML Tokens. [14]

An Anonymous Credential (pseudonymous identification) scheme allows a user to derive from a single master secret multiple cryptographic pseudonyms. Then, it authenticates herself by proving that she knows the master secret underlying a cryptographic pseudonym i.e. (Derived pseudonym predicate). The predicate *NymDer*(*nym,A*) is true if and only if A encodes the master secret key from which the cryptographic pseudonym *nym* was derived. The user first chooses a random master secret key *msk*. from the master secret. Then,, he derives as many unlinkable pseudonyms *nym* as she wants. Next,, using her master secret key *msk*, he authenticate herself with respect to *nym*. The central idea is that all the user's credentials are underlain by the same master secret *msk*, so that by sharing *msk* with others, the user is sharing her whole identity, rather than just her pseudonym *nym* and the associated access to this service. However, the pseudonyms are not linkable to the user and keep the user anonymous in a sense. The pseudonym mechanism can be integrated in the SAML Tokens. [17]

B. Use of SAML Token in WS-Security SOAP Messages.[16]

WS-Security allows specifying identification and authorization data in a SOAP message. A SOAP message contains the following useful information about the entities involved in the IDM which includes the RP, IdP and the Requestor

- Identify the entity or entities involved with the message.
- Prove that the entities have the correct group memberships.
- Prove that the entities have the correct set of access rights.
- Prove that the message has not changed.

WS-Security seeks to encapsulate the security interactions described above within a set of SOAP Headers. The idea is to use SAML assertions in the SOAP message body of WS-Security, for handling credential management as in Fig.5.

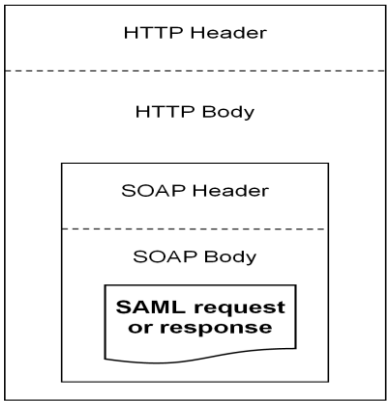

Figure 5. SAML SOAP Binding [18]

C. SAML Tokens and Claims/Assertions [13].

Security Assertions Markup Language (SAML) tokens are XML representations of claims. SAML tokens are supported both browsers (Explorer and Firefox) and operating systems Windows XP, built using Windows Communication Foundation (WCF). SAML tokens carry statements that are sets of claims made by one entity about another entity. For example, in federated security scenarios, the statements are made by a security token service about a user in the system. Fig. 6.

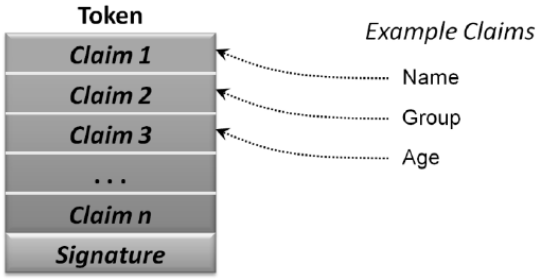

Figure 6. A token contains claims about a user along with a digital signature that can be used to verify its issue [8]

The advantages of using SAML assertions include: [21]

- SAML offers a much broader & extensible set of authentication contexts.
- Support of the standard in commercially available products.

The security token service signs the SAML token to indicate the veracity of the statements contained in the token. In addition, the SAML token is associated with cryptographic key material that the user of the SAML token proves knowledge of. This proof satisfies the relying party that the SAML token was, in fact, issued to that user. For example, in a typical scenario: A client requests a SAML token from a security token service, authenticating to that security token service by using Windows credentials. The security token service issues a SAML token to the client. The SAML token is signed with a certificate associated with the security token service and contains a proof key encrypted for the target service. The client also receives a copy of the proof key. The client then presents the SAML token to the application service (the relying party) and signs the message with that proof key. The signature over the SAML token tells the relying party that the security token service issued the token. The message signature created with the proof key tells the relying party that the token was issued to the client.

C#(language specification for token)
Claim myClaim = new Claim(
ClaimTypes.GivenName,"Martin", Rights.PossessProperty);
SamlAttribute sa = new SamlAttribute(myClaim);

[3]

The above SAML security token could be modified with ZKK, Selective Disclosure and Anonymous Credential [19] to improve the security of CardSpace.

[4]
[5]
[6]

V. CONCLUSION AND FUTURE WORK

In this paper we proposed the use of Microsoft's CardSpace as the identity management system for protecting the user's privacy, while accessing service on the cloud. We discuss the security limitations of CardSpace and proposed an approach to overcome them. We suggest the use of Zero Knowledge Proof (ZKP) cryptographic technique, Selective/minimal Disclosure and Anonymous Credentials within the CardSpace's framework to improve protecting of privacy for users of CardSpace by

Since CardSpace is built on claims based access platform/ architecture, the ZKP can be integrated in the SAML token containing the values of the claim. With the use of ZKP in the security tokens, the user can satisfy the relying party's technical policy but tell the party nothing or as little as possible and without disclosing the actual values of the credentials. In this way the user's privacy is protected in the cases of hijacked passwords or vicious service providers.

REFERENCES

[1] (2010)Consumer-Privacy-in-the-Internet-Economy., http://www.cio.gov
[2] (2010) twitter breach , http://www.computerworld.com/s/article/9135893
[3] OpenID Explained, http://openidexplained.com/
[4] Kim Cameron's Identity Weblog, http://www.identityblog.com/?p=685
[5] (2010) PRIME Framework V3, https://www.primeproject.eu
[6] (2010)PRIME White Paper V3, https://www.primeproject.eu/
[7] Introducing Windows CardSpace, http://msdn.microsoft.com
[8] CLAIMS-BASED IDENTITY FOR WINDOWS http://download.microsoft.com
[9] K. Cameron, M.B. Jones. Design Rationale behind the Identity Metasystem Architecture, http://research.microsoft.com
[10] W. A. Alrodhan, C. J. Mitchell, Improving the Security of CardSpace, EURASIP Journal on Information Security Vol. 2009
[11] B. Laurie. Selective Disclosure, http://research.google.com/pubs/author9639.html/, 2007
[12] Zero knowledge example Fiat-Shamir proof of identity http://pages.swcp.com/~mccurley/talks/msri2/node24.html
[13] SAML Tokens and Claims -msdn http://msdn.microsoft.com/en-us/library/ms733083.aspx
[14] (2008) .S. F. Hubner, HCI work in PRIME, https://www.prime-project.eu/,
[15] (2011) Understanding WS-Federation http://msdn.microsoft.com
[16] (2011) Understanding WS-Security http://msdn.microsoft.com
[17] Exploiting Cryptography for Privacy-Enhanced Access Control, A result of the PRIME Project, Claudio A. Ardagna et.al , Journal of Computer Security, v.2009/02/26 http://spdp.dti.unimi.it/papers/JCS2010-PRIME.pdf
[18] Security Assertion Markup Language, A Brief Introduction to SAML, Tom Scavo, NCSA
[19] Security and Privacy Consideration for the OASIS Security Assertion Markup Language (SAML) V2.0, Committee Draft 01, 18 August 2004
[20] An Entity-centric Approach for Privacy and Identity Management in Cloud Computing Pelin Angin, Bharat Bhargava, Rohit Ranchal, Noopur Singh ; Lotfi Ben Othmane, Leszek T. Lilien ; Mark Linderman
[21] http://blogs.sun.com/hubertsblog/entry/deep_dive_on_saml_2, February, 2011