RC24359 (W0709-114) September 27, 2007
Computer Science

IBM Research Report

SIdeCAR: Secure Identity Consent and Authentication Responder

Ravi Chandra Jammalamadaka
University of California, Irvine

Michael McIntosh, Paula Austel
IBM Research Division
Thomas J. Watson Research Center
P.O. Box 704
Yorktown Heights, NY 10598

SIdeCAR: Secure Identity Consent and Authentication Responder

Ravi Chandra Jammalamadaka[1], Michael McIntosh[2], and Paula Austel[3]

[1]University of California Irvine
[1]rjammala@ics.uci.edu

[2,3]IBM Thomas J. Watson Research Center
[2]mikemci@us.ibm.com
[3]pka@us.ibm.com

September 21, 2007

Abstract

The Identity Metasystem is an interoperable, platform independent and protocol independent architecture for user centric identity management. User centric identity management is a new paradigm of identity management that addresses some of the drawbacks of the prevalent identity management models. This technology assumes that certain security sensitive functions of identity management are performed at trusted client machines. Such an assumption is not valid when a machine which is infested with undetected malware, possibly on a publicly accessible "kiosk" machine. We explore techniques that provide the user with: a) portability between machines; and b) enhanced security when using the Identity Metasystem from untrusted machines. We present the threats that untrusted machines pose and describe two protocols we've implemented which allow secure use of the Identity Metasystem from untrusted clients without changes to the widely implemented protocols. Both the protocols leverage the use of a trusted personal device (e.g. cellular phone) to authorize actions that are performed at the client and perform secret-based computations. The security protections and implementation details of both the protocols are described. We conclude with the future directions that we intend to take with regard to our work.

1 Introduction

The Identity Metasystem [1] is emerging as a potential universal identity management infrastructure. It provides users with a relatively convenient interface to strong authentication mechanisms along with greater control over the dissemination of sensitive personal information. Its adoption could help to reduce the threat caused by various forms of identity fraud including phishing. Broad deployment of this technology could eventually lead to the elimination of the countless do-it-yourself authentication and identity solutions so prevalent on the internet today.

The threat posed by any security compromise to this infrastructure is great. While there are multiple implementations of the Identity Metasystem developed and deployed, the number of these is limited and could therefore be considered to represent a monoculture[2]. Hackers will likely focus their attention on this relatively universal infrastructure.

Within the Identity Metasystem, sensitive information is stored, entered, exchanged, and displayed on various components which are deployed on systems which are typically in different administrative domains. Users interact directly with the Identity Metasystem systems through which they access the internet.

Users' machines are usually administered by the users themselves or an administrator appointed by the organization that provides the machine. Occasionally, users access the internet via machines with unknown administrators and administrative policies, as is the case with Internet Cafes and public kiosks.

It is common for poorly administered machines to be infested with undetected malware. It can be assumed that, publicly accessible machines may be purposely configured with such malware by unscrupulous administrators.

Our work focuses on the mitigation of the threats posed by malware on the machines through which users access the internet and interact with the Identity Metasystem. We've left the threats created by the compromise of the other systems out of scope. These other systems are likely administered by trained professionals. We have also made it a goal of this iteration of our work to improve the overall security without modifying any of the standard interfaces. The implementation changes we propose are limited to the component which is least likely to use commercial off the shelf software. We've based our prototype on the Security Token Secvice component of the Higgins Project [16].

We describe the Identity Metasystem, its components, the associated user information, and its relative sensitivity to disclosure. We describe the method through which users may move from machine to machine and consistently use the Identity Metasystem. We explore some of the limitations of this method, and some of the exposures it creates. We then describe two solutions to some of these exposures. Both of these solutions use a trusted personal device, such as a cellular phone or PDA to store and display sensitive information, perform secret-based computations, and provide a channel for secondary authentication and authorization for the dissemination of user identity information. We describe the components and protocols that enable these solutions and discuss limitations which we intend to work to address in future work.

2 The Laws of Identity

The Identity Metasystem was defined to adhere to a set of principles known as the ”Laws of Identity” [3]. These seven guiding principles were defined through a collaborative effort led by Kim Cameron. The seven ”laws” are:

1) User Control and Consent
2) Limited Disclosure for Limited Use
3) The Law of Fewest Parties
4) Directed Identity
5) Pluralism of Operators and Technologies
6) Human Integration
7) Consistent Experience Across Contexts

It is assumed that a system that conforms to this set of principles will meet the identity needs of the vast majority of systems while improving the privacy protection of its users. This of course assumes that the system is behaving as expected and no malware is installed to undermine its conformance.

3 The Identity Metasystem

Here we describe the Identity Metasystem as implemented[4][5] by Microsoft CardSpace and the Higgins Project. We've focused our work in the area of Web Browser based interactions[6].

The Higgins Project is an open source identity framework that is interoperable with the Identity Metasystem. The initial implemention is Java-based and has been demonstrated on Microsoft Windows XP, Linux, and MacOS.

Microsoft CardSpace is an implementation of the Identity Metasystem included with Windows Vista. CardSpace attempts to mediate problems with malware running on Windows systems by separating the User Identity Agent from other desktop applications such as the Web Browser running on the default desktop. The User Identity Agent runs in a more secure separate desktop. Other than the Relying Party Policy and SSL Certificate and the encrypted Security Token, all sensitive information and user interactions are accessible only within the secure desktop.

The User is an integral component of the Identity Metasystem. The User must decide whether to consent to the transaction and if so decide Digital Identity will be presented to the Relying Party. A Digital Identity, also known as a Security Token, is a digital representation of a set of claims made by one party about itself or another party. The Relying Party is a consumer of a Digital Identity. The Identity Provider is an issuer of a Digital Identity. The User Identity Agent is a component that interacts with a) the User for authentication and consent purposes and b) the Identity Provider for Digital Identity issuance.

3.1 Information Card Provisioning

The first step in using the Identity Metasystem is for the User to either: a) obtain a Managed Information Card from an Identity Provider, or b) create a Personal Information Card. Information Cards can be thought of as representing a Digital Identity since they can be used to request a Digital Identity in the form of a Security Token.

A Managed Information Card contains: the logical name of the Identity Provider; the Identity Provider endpoint addresses; the type of User credential that must be supplied to the Identity Provider, optional User credential hint information; supported Security Token types and claim types; whether or not the Relying Party identity must be provided along with Request Security Token messages; and the location of the privacy policy of the Identity Provider. For each Identity Provider endpoint address, the Managed Information Card contains: authentication requirements; the SSL Certificate of the WS-Trust endpoint; a corresponding WS-MetadataExchange endpoint used for accessing the Identity Provider's Policy, and the SSL Certificate of the WS-MetadataExchange endpoint. Managed Information Cards allow a User to request a Digital Identity from an Identity Provider.

A Personal Information Card contains supported claim types along with values for them and a Master Key. Personal Information Cards allow a User Identity Agent to act as a self issuing Identity Provider on behalf of the User. The CardId, in combination with information identifying the Relying Party, is used to generate a unique identifier for the User at that Relying Party. This is known as the Private Personal Identifier (PPID). The Master key is used, in combination with information identifying the Relying Party, to generate an RSA key pair for signing the Security Tokens to be sent to that Relying Party. This is necessary to support "laws" 2, 3, and 4 by preventing a group of Relying Parties from colluding to track a User across multiple sites. This also helps reduce the threat from phishing attacks since the identifier in the Security Token is specific to the false phishing site, not the intended site.

Note that the Master Key associated with a Personal Information Card is an extremely sensitive secret that must be protected. Compromise of the Master Key would enable an attacker to compute the RSA key pair for any Replying Party. The CardId can be used to compute the User's PPID for any Relying Party. Therefore, any Relying Party should recognize the combination of PPID and issuer public key as the identifier for a User.

3.2 Relying Party Identity and Policy Discovery

When the User attempts to access a Web based Relying Party resource the User Identity Agent is activated. The Web Browser must use SSL when attempting to access the resource. This allows the User Identity Agent to access the Relying Party's SSL Certificate, which provides a trustworthy identifier for the Relying Party as well as the Relying Party's public key.

The Relying Party Policy, represented by an Information Card OBJECT element within the HTML of the resource link, is provided to the User Identity Agent as PARAM elements within OBJECT element. The Relying Party Policy contains the privacy policy of the Relying Party as well as constraints on the Digital Identities accepted by the Relying Party including: a) acceptable Issuers, b) acceptable Security Token types, c) acceptable proof of identity, and d) required and optional claim types.

3.3 Information Card Matching

After the User Identity Agent receives the Relying Party Policy it finds the set of Information Cards in its Card Store that match that policy. These matching Information Cards are candidates for selection by the User. The matching algorithm is as follows: if the policy includes an issuer name then any Information Cards that do not have the same issuer name are eliminated; any Information Cards that do not support the Security Token type in the policy are eliminated; any Information Cards that do not support all of the required claim types are eliminated; any remaining Information Cards are considered as candidates for selection.

3.4 Information Card Selection

The User, via the User Identity Agent, reviews: a) the information identifying the Relying Party; b) the information contained in the Relying Party policy; and c) the information contained in each selectable Information Card; and decides which, if any, Information Card to select and whether or not to include optional claims. By selecting a specific Information Card a User selects the Digital Identity to be presented to the Relying Party. The User also indicates whether or not the User Identity Agent should retrieve and display the claim values before posting the Security Token to the Relying Party. Alternatively, the User may decide to abort the process and cancel the transaction.

3.5 Identity Provider Policy Discovery

If a Managed Information Card was selected, the associated Identity Provider Policy must be retrieved using the WS-MetadataExchange protocol. The Identity Provider Policy contains the authentication and communication security requirements on the Request Security Token messages it will accept.

3.6 User Authentication

If a Managed Information Card is selected a WS-Trust RequestSecurityToken message must be sent to the associated Identity Provider. In order for the User to prove its identity to the Identity Provider a credential must be sent with the message. The most common forms of credentials supported are: a) Username and Password Security Token, b) self issued SAML Security Token, c) X.509 Certificate, and d) Kerberos Security Token. We will focus on the first two credential types.

When the selected Managed Information Card specifies a Username and Password Token as the User credential type, the hint provided is the Username. The User Identity Agent displays a prompt for the Username and Password with the Username already filled in. The data entered by the User is placed into the Username and Password Token which is sent with the RequestSecurityToken message to the Identity Provider.

When the selected Managed Information Card specifies a self issued SAML Security Token as the User credential type, the hint provided is the PPID for a Security Token which was presented during the provisioning of the Managed Information Card. This PPID is used by the User Identity Agent to locate the Personal Information Card that was used to issue that Security Token. The User Identity Agent then uses the embedded self signing Identity Provider to issue a SAML Security Token treating the Identity Provider associated with the Managed Information Card as the Relying Party. This self signed Security Token is sent with the RequestSecurityToken message to the Identity Provider.

Note that any compromise of the MasterKey associated with a Personal Information Card that is itself used to authenticate with a Managed Information Card, compromises the authentication for that Managed Information Card.

3.7 Token Issuance

Once the User Identity Agent has the Relying Party identity and public key, the selected Information Card, and any User credentials, it can request the Digital Identity in the form of a Security Token from the appropriate Identity Provider.

In the case of a Personal Information Card: the appropriate claims values, either computed or extracted from the CardStore, are placed into a SAML Security Token; an RSA key pair is generated and used to sign the Security Token; the Relying Party public key is used to encrypt the Security Token; and optionally a display token is constructed containing the unencrypted claim values.

In the case of a Managed Information Card: the Relying Party identity and public key, information from the selected Managed Information Card, and User credentials, are formatted into a WS-Trust RequestSecurityToken message which is sent to the Identity Provider's WS-Trust endpoint. The Identity provider places the appropriate claims values into a SAML Security Token; the Identity Provider's RSA key pair is used to sign the Security Token; the Relying Party public key is used to encrypt the Security Token; and optionally a display token is constructed containing the unencrypted claim values.

3.8 User Confirmation

When the User selected the Information Card they indicated whether the claim values should be confirmed before the Security Token is posted. If the User selected that option the claim values are displayed in the User Identity Agent and the user has another opportunity to opt out of the transaction.

3.9 Post Digital Identity

The encrypted Security Token returned from the Identity Provider is provided by the User Identity Agent to the Web Browser which posts it to the Relying Party signalling the acceptance of the terms of the transaction by the User.

4 Problem Definition

We are interested in designing solutions that allow use of the Identity Metasystem from untrusted machines. By "untrusted", we mean a computer system that can potentially host malicious software. Therefore, the user cannot place a lot of trust on such machines.

4.1 Threat model

The untrusted machine (UM) can perform the following undesirable operations:

- Hijack Relying Party Sessions. This will allow an attacker to interact with the Relying Party without the User's control or knowledge.
- Capture Encrypted Security Tokens. This will allow an attacker to authenticate to a Relying Party impersonating the User, until the Security Token expires. Note that these Security Tokens typically have a reasonably short validity period and the Relying Party is expected to prevent replay of Security Tokens.
- Capture Personal Information Card Content. This will allow an attacker to know the set of claim types that are supported for the Digital Identity as well as the values associated with each claim type. It will also reveal the Master Key which can be used to authenticate to a Relying Party or an Identity Provider using the associated Digital Identity.

- Capture Managed Information Card Content. This will allow an attacker to know the identity of Identity Provider and the set of claim types that are supported by the Identity Provider for the Digital Identity. Note that this will not reveal the values associated with the claim types.
- Log the Username and Password information entered by the User. This will allow an attacker to authenticate to the Identity Provider and thereby to any Relying Parties that allow the User to present Digital Identities issued by that Identity Provider.
- Capture the Display Token Content. This may provide the attacker with enough information to impersonate the User (i.e. name, address, phone number, social security number, etc.)

By performing all the above operations, the UM *can potentionally impersonate the User or otherwise violate the Laws of Identity*. Our objective is to prevent the UM from performing as many of the above operations as possible while retaining interoperability with standard Identity Metasystem components.

4.2 Desirable properties

We wanted to design solutions that have the following desirable properties:

No changes to the User Identity Agent: This will allow existing User Identity Agents to work out of of the box with our solutions. Such compatibility is highly desirable since vendors of User Identity Agents cannot be expected to adapt their software to comply with our solutions.

No changes to the Relying Party: Since the User Identity Agent is the only component to interact with the Relying Party and we have chosen to avoid changes to the User Identity Agent, we also avoid changes to the Relying Party. Since the changes we propose are intended to protect the User not the Relying Party, it would be difficult to motiviate the Relying Party to adapt their software to comply with our solutions.

Usable protocols: We placed heavy emphasis on developing solutions that are usable. We do not wish to burden the User in performing a lot of operations manually or make them wait for significant periods of time. Such activities severely limit the adoption of new technology.

Provide security without significant performance degradation: As stated before, solutions which take a significant amount of time to run to completion will hinder the usability of such a system. We wanted to strike an appropriate balance between performance and security in designing our solutions.

4.3 Trust Model

User Identity Agent: The User Identity Agent is assumed to be relatively untrusted and can potentially launch attacks against the User. The User Identity Agent is not assumed to be under the administrative control of the User and we do not place any restrictions on the kind of attacks the agent can launch. The purpose of our work is to prevent/reduce the damage that can be done by a rogue User Identity Agent.

Relying Party: The Identity Metasystem does not allow Relying Parties to do identity management on behalf of the User. Therefore, the Relying Party is provided with a User's Security Token containing the minimal information to allow the User to access to the resource. The Relying Party is expected to conform to the policies provided when the User consented to the transaction. Whether or not the Relying Party conforms to its own stated policy is *outside the control* of both the User and the Identity Provider. We inherit the same trust model in our context as well. We will show later that with our solution even if the Relying Party colludes with the User Identity Agent, extra information is not revealed to the Relying Party.

Identity Provider: The Identity Provider is trusted both by the Relying Party and the User to issue adequate Security Tokens and not collude with any malicious entity.

Trusted device: Our solutions leverage the use of trusted mobile device the User is expected to carry when accessing the Web. Such a device is assumed be under the administrative control of the User and

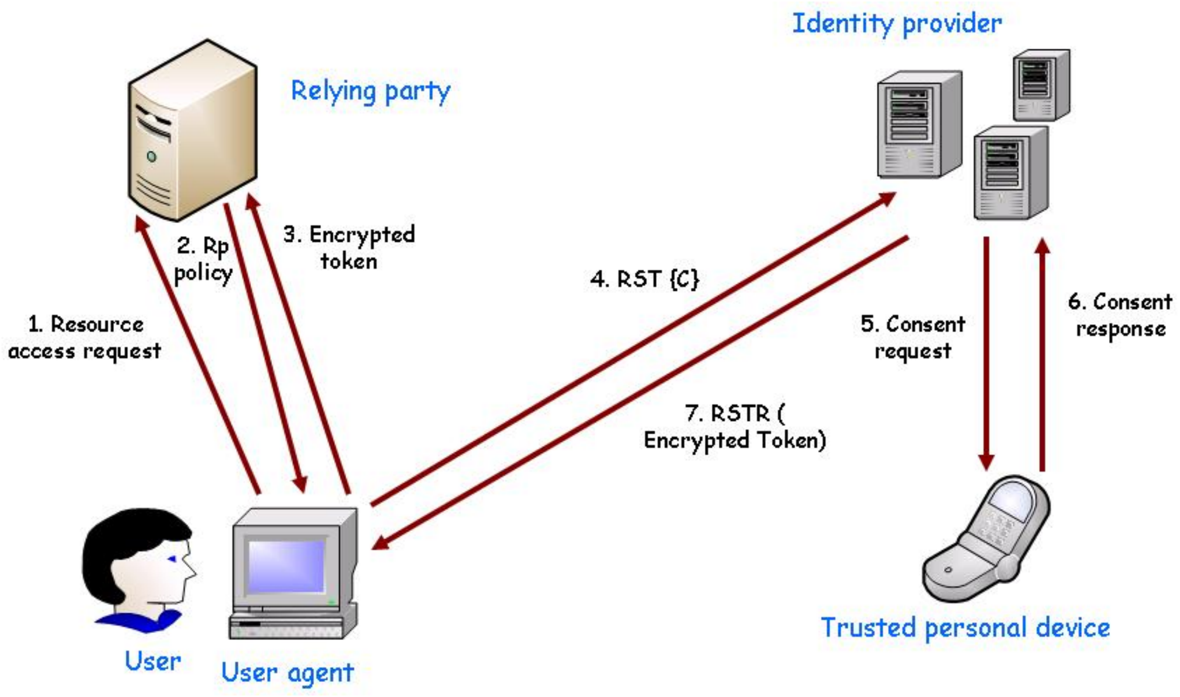


Figure 1: Secondary Authentication

free of malware. Therefore, the device is more likely to be trusted by the User than an arbitrary computer system running a User Identity Agent.

5 Solutions

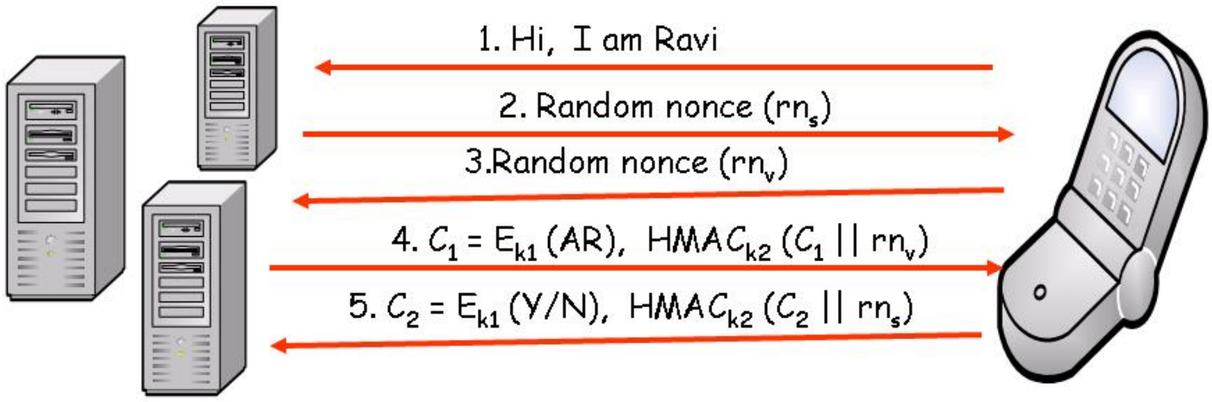
This section describes the two solutions that we designed and tested for use of the Identity Metasystem from untrusted machines.

5.1 Secondary Authentication

Our first approach establishes a trusted device as a secondary factor of authentication for the request. A secure path is established between the Identity Provider and the trusted personal device. When the Identity Provider receives a RequestSecurityToken message from the User Identity Agent, a message is sent to the User's device describing the transactional details. The User then will carefully read the message and either consent or deny the request or seek additional information from the Identity Provider. If the request is denied, a fault message is returned to the User Identity Agent. If the request is approved, a normal RequestSecurityTokenResponse message is returned to the User Identity Agent.

Notice that in this solution we have only made changes to the Identity Provider. The rest of the Identity Metasystem is untouched. Fig 1 illustrates the solution.

Messages exchanged: The Identity Provider upon receiving a RequestSecurityToken message, generates a *consent request (CR)* message and forwards it to the trusted device via the secure channel. The consent request message consists of the following information: a) the Relying Party URL address; and b) the set of claims requested by the Relying Party. After receiving the CR message, the User will either consent or deny by sending a YES/NO answer back to the Identity Provider, or will seek additional information from the Identity Provider. Additional information could include the privacy policy of the Relying Party, the certification path of the Relying Party, etc. Once the Identity Provider receives the consent from the User, it will send RequestSecurityTokenResponse messages with the required claim values encrypted with the Relying Party's public key to the User Identity Agent.



1. Hi, I am Ravi
2. Random nonce (rn_s)
3.Random nonce (rn_v)
4. $C_1 = E_{k1}(AR)$, $HMAC_{k2}(C_1 || rn_v)$
5. $C_2 = E_{k1}(Y/N)$, $HMAC_{k2}(C_2 || rn_s)$

Figure 2: Identity Provider-Trusted device interaction

We will now look at the establishment of a secure path between the Identity Provider and the trusted device more closely. A straightforward way to create such a path is to use an SSL connection between them. We decided not to do so for performance reasons. Our preliminary tests showed that it took a few seconds for the trusted device to establish an SSL connection with the Identity Provider. We instead designed a protocol for messaging exchange security properties similar to those of the SSL protocol.

Fig 2 illustrates the protocol for establishing a secure path between the Identity Provider and the trusted device. The Identity Provider and the trusted device negotiate two keys k_1 and k_2 during the registration process. k_1 is used for protecting the confidentiality of the messages that are exchanged and k_2 is used for protecting the integrity. When the Identity Provider wants to send a message AR to the trusted device, it sends a random nonce rn_s to the trusted device. The trusted device responds with a random nonce rn_v back to the Identity Provider. The random nonces are used to prevent replay attacks from both directions. Then, the Identity Provider encrypts AR with key k_1 to generate the ciphertext $c_1 = E_{k_1}(AR)$. It also computes the HMAC of the ciphertext concatenated with the nonce sent to the Identity Provider from the trusted device to generate $HMAC_{k_1}(c_1||rn_v)$. The trusted device now can verify the integrity of the ciphertext and then decrypt it to obtain the plaintext AR. A similar process is followed when the user consent is sent from the trusted device to the Identity Provider, except that the nonce rn_s is used instead to calculate integrity information.

5.1.1 Security properties

Our solution contains the following security properties:

- S1: Information supporting the consent decision is sent to and displayed on the trusted device. Such information includes the privacy policy of the Relying Party, certification path of the Relying Party, etc.
- S2: User consent is entered via a trusted device. This property ensures that malware cannot masquerade as the User.
- S3: Some potentially sensitive information such as the display token is not relayed to the User Identity Agent on the untrusted machine.

5.1.2 Limitations

The following are some limitations of our solution:

- The second factor of authentication is required even when using a trusted machine. There is no mechanism to turn off the secondary authentication from the Identity Provider.

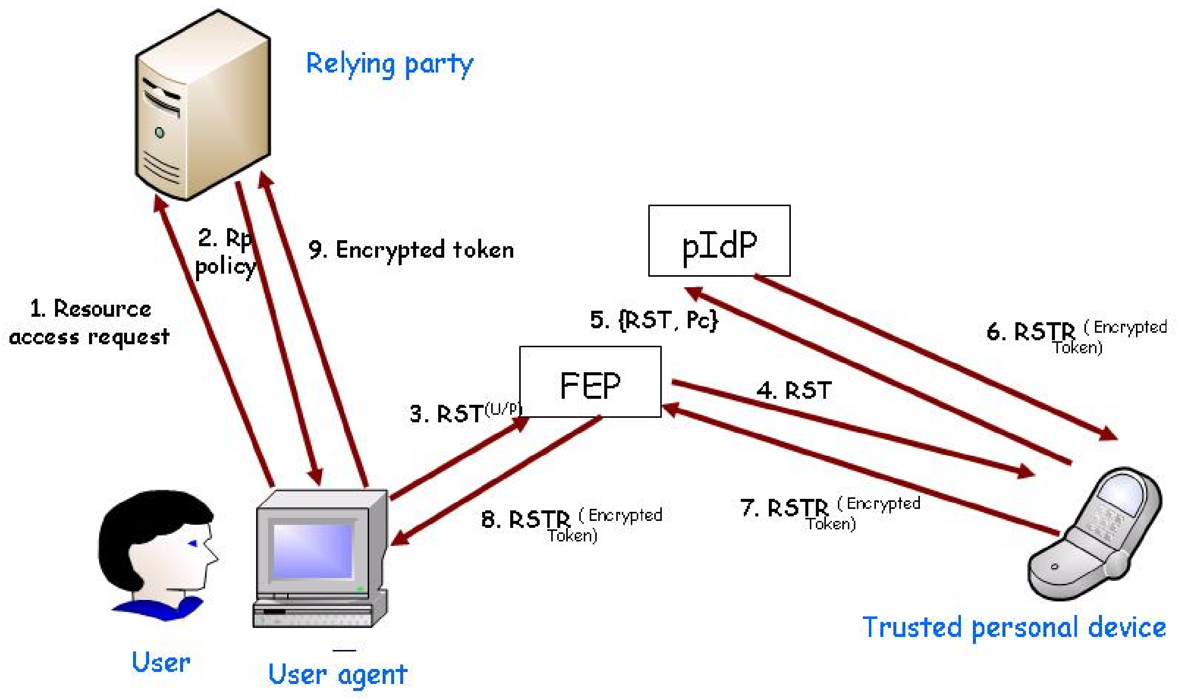


Figure 3: Universal Card protocol for Personal Cards

- Only works with Managed Information Cards. Personal Information Cards cannot be supported because an external Identity Provider is not accessed when they are used.
- Requires changes at the Identity Provider. This could be a potential impediment against widespread adoption of the solution.
- Leaks information present in the Information Cards, which includes the supported claims and the Identity Provider information. This raises privacy concerns as the claims themselves (not the claim values) can reveal information about the User.

5.2 Mobile User Agent via Universal Card

We will now present a solution that addresses some of the drawbacks of the previous protocol. A User's Personal and Managed Information Cards will no longer be stored at the untrusted client. This information will be stored on the trusted device. The CardStore on the untrusted client will only contain one Universal Information Card. The self signed Identity Provider no longer runs as part of the User Identity Agent on the untrusted client. The self signed Identity Provider will run on the trusted device or a trusted server if the trusted device does not meet the requirements for security and performance. This solution requires two new entities:

Front End Proxy: The Front End Proxy is responsible for sending messages to/from the trusted device. The User Identity Agent utilizes the Front End Proxy to send messages to the trusted device. The Front End Proxy is primarily a mailbox where messages to the trusted device are posted.

Universal Card: We introduce a new type of Information Card called a Universal Information Card. The Universal Information Card supports a large set of claims which satisfy the policies of most Relying Parties. The Universal Information Card references the Front End Proxy as the Identity Provider. The User credential specified for the Front End Proxy Identity Provider is Username and Password Token. This credential will be used to identify the User to the Front End Proxy in order allow the Front End Proxy to map the User to a specific trusted device to use for Information Card selection, authentication, and consent.

5.2.1 Protocol Description

Fig 3 illustrates the protocol. We will discuss the protocol under two cases: a) Use of a Personal Information Card; b) Use of a Managed Information Card.

Use of a Personal Information Card: When the Relying Party Policy is sent to the User Identity Agent, the Universal Information Card will match the claims in the policy. When the User selects the Universal Information Card she will be prompted to enter a Username and Password. This information is used to map the User to a specific trusted device and is not intended to be used as a secret used for authentication. The User Identity Agent will then forward a RequestSecurityToken message to the Front End Proxy. The Username and Password pair U/P is sent as part of the RequestSecurityToken message. The U/P pair contains the information that allows the Front End Proxy to determine the trusted device to be contacted. This is required since the multiple users can use the same Front End Proxy. For example, the value of the username could be the identifier of the user (i.e. Ravi) and password could be the identifier for the trusted device (i.e. RaviCellPhone). Note, we assume there is a pre-registration process between the Front End Proxy and the trusted device that would specify the information needed for mapping a User to a trusted device.

The Front End Proxy now forwards the RequestSecurityToken message to the trusted device. The User has a representation of Personal and Managed Information Cards stored in the trusted device (since the device does not communicate directly with standard Information Card representations. We use a representation that requires less storage space). The Relying Party Policy is now matched with the Personal Information Cards in the trusted device. The User then selects one of the matching Information Cards. Information from the selected Information Card and the RequestSecurityToken message are forwarded to the Personal Identity Provider. The Personal Identity Provider now generates the Security Token and encrypts the Security Token with the public key (sent as part of the RequestSecurityToken message) of the Relying Party. The encrypted Security Token is forwarded to the trusted device and the trusted device in turn forwards it to the User Identity Agent. The User Identity Agent then submits it to the Relying Party which allows access to the User. If the trusted device does not find a match on the claims then no security token is sent to the User Identity Agent.

Usage of a Managed Card: The protocol runs very similarly for Managed Information Cards as well. In this case the Personal Identity Provider needs to be replaced by the respective Managed Identity Provider.

One of the biggest advantages of this protocol is that it does not require changes at the Relying Party or the User Identity Agent. There are minimal changes needed to the Identity Provider to establish communication with the trusted device.

5.2.2 Who Runs the Front End Proxy ?

The Front End Proxy is a *semi-trusted* entity. It is trusted enough to provide the service of passing messages but not trusted with secrets or unencrypted Security Tokens. Nothing of value passes through the Front End Proxy in plaintext form.

The Front End Proxy can be run by Users for themselves. The Users need to be sophisticated enough to install and administer a proxy. The pool of such Users tend to be very small. A more likely scenario is that of an enterprise providing the Front End Proxy as a service. Such an enterprise can also distribute the Universal Information Card.

5.2.3 Security Properties

This protocol has all the security properties of the previous protocol where secondary authentication was done from the Identity Provider. In addition, here, the Information Cards are not revealed at the untrusted machine and the card selection process is done at the trusted device. The malware does not learn much, other than the fact that user possesses an Information Card that matches the Replying Party Policy.

5.2.4 Drawbacks

The following are the drawbacks of the protocol:

Requirement for Front End Proxy: As stated earlier an Front End Proxy proxy is required to pass messages to the trusted device. We envision that enterprises will provide Front End Proxies as a service. The User will then have to pay for such a service.

Static Universal Information Card: The Universal Information Card contains a static set of claims. It is a potential that this card may not match a policy of a Relying Party. In this case, the Universal Cards will have to be regenerated to include new sets of claims and then distributed to the User Identity Agent.

6 Future Work

In this section we will describe briefly the directions that we intend to take with regard to our work.

Allowing changes to the User Identity Agent: The Front End Proxy was primarily used as a communicating agent, to pass messages back and forth with the trusted device. We resorted to such a solution as there is no direct way to communicate with the trusted device from the User Identity Agent. We next want to explore what is possible if we relax the constraint of not making changes at the User identity Agent level. Specifically, we are looking at techniques that will allow the User Identity Agent to communicate with the trusted device using short-range wireless technology (e.g. NFC, Bluetooth, etc). This will eliminate the need to communicate through the Front End Proxy and therefore make the pool of potential Users larger.

Make the consent process easier for Users: Currently, a text message is sent to the trusted device which is displayed to the User. The text message must be carefully read by the User before the consent is given to ensure that the User Identity Agent the user is interacting with is the one requesting consent. We want to make that process more secure and easier. We envision techniques where a message digest is displayed at the User Identity Agent and at the trusted device. The User needs to check if they match. A successful match will allow the User to consent to the message. There are no obvious cryptographic techniques that can used to achieve the above goal.

Implementing the CardStore and self signing Identity Provider on a smart card: The self signing Identity Provider conceptually needs to run at the trusted device. For performance reasons we chose to implement the module at a trusted server. Smart cards contain cryptographic acceleration hardware for fast performance of cryptographic operations. A smart card attached to the trusted device can now logically function as a self signing Identity Provider. We will investigate the feasibility of such an architecture.

Performing usability studies: Another area of research interest to us is the investigation of the usability of a trusted device like a cell phone in the Identity Metasystem. More specifically, we would like to answer the following questions: a) Can Users tolerate the unreliability of messages reaching the trusted device?; b) Do Users carefully read the messages sent to the device and not consent mechanically?; c) When do Users stop using the device for identity management due to its inconvenience?.

7 Related Work

There are many proposed systems that leverage the use of a cell phone in performing sensitive operations from a malicious/untrusted client machine. We classify the systems under the following categories: a) password based web authentication; b) transaction confirmation for web services; c) prevention of online attacks; and d) secure input.

Password based web authentication: Wu et.al. [12] propose a system that allows remote authentication from a malicious client using a cell phone. The system employs a proxy which intercepts login requests from the client and asks for confirmation from the client. Upon receiving the confirmation from the client, the proxy modifies the HTTP request and fills in the password on behalf of the user. This system does not protect against session hijacking attacks where the malware can make illegitimate requests masquerading as the client.

MPauth proposed by Mannan et. al.[13] is a system that sets up a secure SSL connection between the untrusted client and web server via a cell phone. The user is expected to carry the public key of the web server. This solution unlike [12] does not require a proxy to broker the traffic.

Transaction confirmation: Clarke et. al.[9] propose a technique for authorizing actions from an untrusted client machine using a cell phone. Similar to [12], this solution requires the usage of a proxy. The proxy sends back visual information back to the untrusted client which needs to be verified by the user. The user utilizes a camera present in the cell phone to capture the visual information and runs a cryptographic algorithm to compute a token which is presented to the proxy via the client machine. The proxy also calculates the token to ensure that it is indeed the user that is performing the actions from the untrusted computer.

Spyblock [11] is a browser extension that protects the users from spyware residing in the client machine. Spyblock employs an authentication agent that runs inside the OS which is assumed to be trusted. A secure path is established between the user and the agent via a virtual machine monitor. The agent contacts the user and confirms sensitive actions. To provide protection against session hijacking, spyblock provides an API to the web servers using which the web server can send messages to the user confirming transactions. Therefore, adoption of the spyblock technology requires system changes to the web server, which could be a limiting factor.

Delegate [14] is a proxy based architecture that protects the users from keylogging and session hijacking attacks when accessing websites from untrusted machines. The proxy utilizes a policy based framework to determine if certain HTTP requests are dangerous/sensitive. Dangerous requests require confirmation from the user. The user confirms the requests using a trusted device such as a cell phone. Unlike Spyblock, Delegate does not require changes at the web server side.

Online attacks: Phoolproof [10] is a system that utilizes a trusted mobile device to prevent phishing attacks. The user sets up a secure channel between the web server and the mobile device with the help of a pre shared secret. The mobile device, server and the untrusted browser then run a protocol that ensures that the user is not connected to a malicious website.

The authors of [15] propose guardian a framework for privacy control. A trusted mobile device is used as the middleman between the untrusted PC and internet. The trusted mobile device is entrusted with all the tasks that proxies are expected to do in [14, 12].

Secure input: Oprea et.al. [7] propose a protocol that allows secure access to a home computer. The user is assumed to be carrying a trusted mobile device. The mobile device provides temporary credentials to the client machine that allows it to communicate to the home computer. The trusted mobile device is then used as an input device to dictate the actions that need to take place at the server. The inputs from the untrusted computer are simply ignored. Similar to [7], [8] is another work where user input traverses a trusted tunnel from the input device to the application.

8 Conclusions

In this work we addressed the problem of securely using the Identity Metasystem from untrusted machines. We proposed two solutions in this regard that utilize a trusted device. The trusted device is used to a:) provide consent to identity based transactions; and b) to perform secret based computations. We described the security protections offered by both the protocols. We implemented the protocols to validate their efficacy.

Our work concentrated on solutions that require no changes at the Relying Party or the User Identity Agent. We explored solutions within these limitations. In the future, we will concentrate on relaxing such constraints to design solutions that are easier to use and provide more security.

References

[1] Microsoft. Microsofts Vision for an Identity Metasystem. Microsoft Whitepaper.

[2] D. Geer, R. Bace, P. Gutmann, P. Metzger, C. Pfleeger, J. Quarterman, and B. Schneier. Cybersecurity: The cost of monopoly–how the dominance of Microsoft 's products poses a risk to security. Technical report, Comp. and Comm. Ind. Assn., 2003.

[3] Kim Cameron. Laws of Identity. Microsoft Whitepaper.

[4] A. Nanada. Identity Selector Interoperability Profile V1.0 April, 2007.

[5] Microsoft, Ping Identity. An Implementers Guide to the Identity Selector Interoperability Profile V1.0, April 2007.

[6] Microsoft, Ping Identity. A Guide to Using the Identity Selector Interoperability Profile V1.0 within Web Applications and Browsers, April 2007.

[7] Alina Oprea, Dirk Balfanz, Glenn Durfee, D. K. Smetters. Securing a Remote Terminal Application with a Mobile Trusted Device. Proceedings of the 20th Annual Computer Security Applications Conference (ACSAC'04) - Volume 00.

[8] Jonathan M. McCune, Adrian Perrig, Michael K. Reiter. Bump in the Ether: Mobile Phones as Proxies for Sensitive Input. CMU-CyLab-05-007.

[9] Dwaine Clarke, Blaise Gassend, Tom Kotwal, Matthew Burnside, Marten van Dijk, Srini Devadas, Ron Rivest. The Untrusted Computer Problem and Camera-Based Authentication. In the proceedings of the International Conference on Pervasive Computing (Pervasive2002), August 2002.

[10] Bryan Parno, Cynthia Kuo, Adrian Perrig. Phoolproof Phishing Prevention. Proceedings of the Financial Cryptography and Data Security 10th International Conference, February 27-March 2, 2006, Anguilla, British West Indies.

[11] Collin Jackson, Dan Boneh, and John C. Mitchell. Stronger Password Authentication Using Virtual Machines. Technical report.

[12] Min wu, Simon L. Garfinkel, Robert Miller. Secure Web Authentication with Mobile Phones.

[13] Mohammad Mannan and P. C. van Oorschot. Using a Personal Device to Strengthen Password Authentication from an Untrusted Computer. Financial cryptography and Data Security 2007.

[14] Ravi Chandra Jammalamadaka; Timothy van der Horst; Sharad Mehrotra; Kent Seamons; NaliniVenkatasuramanian. Delegate: A Proxy Based Architecture for Secure Website Access from an Untrusted Machine. 22nd Annual Computer Security Applications Conference (ACSAC), Maimi, FL, December, 2006.

[15] N. B. Margolin, M. K. Wright, and B. N. Levine. Guardian: A framework for privacy control in untrusted environments, June 2004. Technical Report 04-37 (University of Massachusetts, Amherst).

[16] http://www.eclipse.org/higgins/