Enhancing CardSpace Authentication Using a Mobile Device

Haitham Al-Sinani, Chris Mitchell

Enhancing CardSpace Authentication Using a Mobile Device

Haitham S. Al-Sinani and Chris J. Mitchell

Information Security Group
Royal Holloway, University of London
http://www.isg.rhul.ac.uk
[Haitham.Al-Sinani.2009, C.Mitchell]@rhul.ac.uk

Abstract. In this paper we propose a simple, novel scheme for using a mobile device to enhance CardSpace authentication. During the process of user authentication on a PC using CardSpace, a random and short-lived one-time password is sent to the user's mobile device; this must then be entered into the PC by the user when prompted. The scheme does not require any changes to login servers, the CardSpace identity selector, or to the mobile device itself. We specify the scheme and give details of a proof-of-concept prototype. Security and operational analyses are also provided.

Keywords: CardSpace, OTP, mobile device, authentication

1 Introduction

In line with the continuing increase in the number of on-line services requiring authentication, there has been a proportional rise in the number of digital identities needed for authentication purposes. This has contributed to the recent rapid growth in identity-oriented attacks, such as phishing, pharming, etc. In an attempt to mitigate such attacks, Microsoft has introduced an identity management system called CardSpace.

CardSpace is a user-friendly tool supporting user authentication. To sign on to a website, a CardSpace user selects a virtual card, known as an information card (InfoCard), from an interface provided by the CardSpace identity selector (CIdS), instead of providing a username and password.

One fundamental limitation of CardSpace is that anyone with access to a Windows user account can also access and use the InfoCards. By default, CardSpace does not provide access protection for the CIdS. To address this issue, CardSpace allows individual InfoCards to be PIN-protected. Also, the entire Windows user account could, of course, be password-protected. Whilst the use of passwords and PINs for InfoCard protection can help, it does not completely solve the problem, not least because one of the fundamental design goals of CardSpace is to reduce reliance on password authentication.

We address this limitation through the introduction of a second authentication factor to be used in conjunction with CardSpace authentication. This additional means of user authentication involves a one-time password (OTP) supplied to the user by a standard mobile device capable of receiving SMS messages. Such devices are ubiquitous, making the system almost universally applicable. The system also provides two-factor authentication, the first factor being possession of the PC containing the InfoCard and the second factor being possession of the appropriate mobile phone. Two factor authentication is typically considered 'strong authentication' [1].

The wide use of Windows, recent versions of which incorporate CardSpace, means that any enhancement to CardSpace security is likely to be of significance for large numbers of identity management users and service providers. In addition, the use of a mobile phone to enhance CardSpace-based authentication is attractive since users are neither required to remember any new passwords nor obliged to use any additional hardware. Furthermore, many RPs may not accept the burden of supporting a second authentication factor (e.g. SMS-based authentication), unless there is a significant financial incentive or if forced to do so for legal or regulatory reasons. As a result, a client-side technique for supporting SMS authentication for CardSpace-enabled RPs could be practically useful. Such a technique avoids any impact on the performance of the server, since the additional overhead is handled by the client.

The remainder of the paper is organised as follows. Section 2 gives an overview of CardSpace, and section 3 presents the proposed scheme. In section 4 we discuss implementation issues, and in section 5 we provide a security analysis. In section 6 we describe a prototype realisation, and section 7 highlights possible areas for related work. Finally, section 8 concludes the paper.

2 CardSpace

2.1 Introduction

CardSpace provides a secure and consistent way for users to control and manage personal data, to review personal data before sending it to a website, and to verify the identity of visited websites. It enables websites to obtain data from users, e.g. to support user authentication and authorisation.

Digital identities are represented to users as Information Cards (or InfoCards). There are two types of InfoCards: personal (self-issued) cards, and managed cards issued by remote IdPs. Personal cards are created by users themselves, and the claims listed in such an InfoCard are asserted by the self-issued identity provider (SIP) that co-exists with the CardSpace identity selector (CIdS) on the user machine. InfoCards, personal or managed, do not contain sensitive information, but instead carry metadata indicating the types of personal data associated with this identity, and from where assertions regarding this data can be obtained. The data referred to by personal cards is stored on the user machine, whereas the data referred to by a managed card is held by the identity provider (IdP) that issued it [2–5].

The proposed scheme can operate with both managed and personal cards. However, in this paper we only describe its operation with personal cards because the security risks associated with such cards are much greater; any adversary who has access to a logged-in Windows machine can use any of the personal cards unless they are PIN-protected, which is not the default case. By contrast, use of a managed card typically involves authentication by the issuing IdP. The use of personal cards is described below; the use of managed cards is covered in the relevant specifications [2, 3, 6, 7].

By default, CardSpace is supported by Internet Explorer (IE) from version 7 onwards. Extensions to other browsers, such as Firefox[1] and Safari[2], also exist. An updated version, CardSpace 2.0 Beta 2, was released, although Microsoft announced in early 2011 that it will not ship; instead Microsoft has released a technology preview of U-Prove[3]. In this paper we refer throughout to the CardSpace version that is shipped by default as part of Windows Vista and Windows 7, that is available as a free download for XP and Server 2003, and which has been approved as an OASIS standard [7].

2.2 Personal Cards

The CIdS allows a user to create a personal card and populate its fields with self-asserted claims. CardSpace restricts the contents of personal cards to non-sensitive data. Prerequisites for use of a personal card include a CardSpace-enabled relying party (RP) and a CardSpace-enabled user agent, e.g. a web browser capable of invoking the CIdS. At the time that an InfoCard is created, a card-specific ID and master key are also created and stored by the SIP (which also stores the values of the claims for this card).

Using Personal Cards. When using personal cards, CardSpace adopts the following protocol. We describe the protocol for the case where the RP does not employ a security token service (STS), a software component responsible for security policy and token management within an IdP and, optionally, within an RP [6].

1. User agent → RP. HTTP/S request: GET (login page).
2. RP → user agent. HTTP/S response. A login page is returned containing the CardSpace-enabling tags in which the RP security policy is embedded.
3. User → user agent. The RP web page offers the option to use CardSpace; selecting this option activates the CIdS, which is passed the RP security policy. Note that if this is the first time that this RP has been contacted, the CIdS will display the identity of the RP and give the user the option to either proceed or abort the protocol.

[1] `https://addons.mozilla.org/en-US/firefox/addon/openinfocard-identity-selector/`
[2] `http://www.hccp.org/safari-plug-in.html`
[3] `http://blogs.msdn.com/b/card/archive/2011/02/15/beyond-windows-cardspace.aspx`

4. CIdS → InfoCards. The CIdS, after evaluating the RP security policy, highlights those InfoCards matching the policy and greys out the rest. InfoCards previously used for this RP are displayed in the upper half of the selector screen.
5. User → CIdS. The user chooses a personal card. (Alternatively, the user could create and choose a new personal card). The user can preview the card (with its associated claims) to ensure that they are willing to release the claim values. Of the claims specified in an InfoCard, only those requested in the RP policy will be passed to the requesting RP.
6. CIdS ⇌ SIP. The CIdS creates and sends a SAML-based Request Security Token (RST) to the SIP, which responds with a SAML-based Request Security Token Response (RSTR).
7. CIdS → user agent → RP. The RSTR is passed to the user agent, which forwards it to the RP.
8. RP → user agent. The RP validates the token, and, if satisfied, grants access.

Private Personal Identifiers (PPIDs). The PPID is an identifier linking a specific InfoCard to a particular RP [2]. When a user first uses a personal card at a particular RP, CardSpace generates a site-specific PPID by combining the card ID with data taken from the RP certificate, and a site-specific signature key pair by combining the card master key with data taken from the RP certificate. In both cases, the domain name and/or IP address of the RP is used if no RP certificate is available. After generation, the PPID and key pair are stored by the SIP for use in future interactions with this RP.

Since the PPID and key pair are RP-specific, the PPID does not function as a global user identifier, helping to enhance user privacy and reduce the impact of PPID compromise. The CIdS displays a shortened version of the PPID to protect against social engineering attacks and improve readability.

When a user first interacts with an RP using CardSpace, the RP retrieves the PPID and the public key from the received SAML security token, and stores them. If a personal InfoCard is re-used at a site, the supplied security token will contain the same PPID and public key as used previously, and will be signed using the corresponding private key. The RP compares the received PPID and public key with its stored values, and verifies the digital signature.

The PPID could be used on its own as a shared secret to authenticate a user to an RP. However, it is recommended that the associated (public) signature verification key, as held by the RP, should always be used to verify the signed security token to provide a more robust authentication method [2].

3 The Scheme

We next give an overview of the novel scheme, covering relevant operational aspects.

3.1 Entities Involved

The entities involved are:

- a CardSpace-enabled RP (with which the user must have an account);
- a CardSpace-enabled user agent (e.g. a suitable web browser such as IE);
- a handheld device capable of receiving SMS[4] messages (e.g. a mobile phone); and
- software installed on the user PC (referred to throughout as the 'adaptor') implementing the scheme described below.

The adaptor could be implemented as a browser extension[5], which must be able to read, inspect and modify browser-rendered web pages, and must also be able to intercept CardSpace-issued RSTR tokens. In addition, it must be able to generate and send a random, short-lived OTP to the user's mobile phone, and provide a means for the user to enter the OTP. Prior to use of the protocol, the browser extension must be installed and provided with the phone number of the user's mobile phone.

3.2 Operation

The system operates as follows; a summary of the protocol is shown in figures 1 and 2. Steps 1, 2, 4–7, and 10 are the same as steps 1, 2, 3–6, and 8, respectively, of the CardSpace personal card protocol given in section 2.2.

3. Adaptor → user agent. The adaptor scans the login page to detect whether the RP website supports CardSpace. If so, it proceeds; otherwise it terminates.
8. Unlike in the 'standard' case, the RSTR does not reach the RP; instead the adaptor performs the following steps.
   (a) CIdS → adaptor: RSTR. The adaptor intercepts the RSTR and temporarily stores it.
   (b) Adaptor: generates OTP. The adaptor computes (and temporarily stores) a random, short-lived OTP.
   (c) Adaptor → mobile phone: OTP. The adaptor sends the OTP to the user's mobile phone in an SMS message, sent via an HTTPS-protected connection to the SMS Centre or SMS gateway of a wireless carrier or SMS service provider. This method is adopted because it does not require a special application to be installed on the user's mobile phone, which

[4] SMS (Short Messaging Service) allows mobile phones to exchange short messages of at most 160 Latin characters; this service is supported by all GSM and 3G handsets.

[5] Note that if the adaptor is implemented as a browser extension, then the CardSpace-enabled RP must not employ an STS. Instead, the RP must express its security policy using HTML/XHTML, and interactions between the CIdS and the RP must be based on HTTP/S via a web browser (a simpler and probably more common scenario for RP interactions). This is because a (JavaScript-based) browser extension is by itself incapable of managing the necessary communications with an RP STS.

may not be possible in non-smart phones. In addition such an approach has a better transmission rate than other methods such as Bluetooth or infrared (see section 4.2).

9. User $\rightleftharpoons$ user agent. The adaptor prompts the user to enter the OTP, and the user reads it from the phone display[6]. The adaptor verifies that the entered OTP matches the one it just generated. The OTP must be entered within a defined interval, e.g. of 10 minutes, after its generation; otherwise the adaptor will delete the RSTR and provide an error message to the user.

3. A → UA, where A is the adaptor and UA is the user agent.
8. CIdS → [A: generates OTP] → M, where M is the mobile device.
   More specifically:
   a. CIdS → A: RSTR;
   b. A: generates OTP; and
   c. A → M: OTP.
9. U → UA: OTP, where U is the user.

Fig. 1. Summary of the Protocol

4 Discussion

We now consider implementation issues, possible variants and potential advantages of the scheme.

4.1 Implementation Issues

The length of the OTP must be carefully chosen to achieve an acceptable balance between security and usability. To maximise usability and avoid confusion, we propose the use of a 4-character OTP made up of lower case letters and digits (excluding 0, i, j and o). This gives a total of 32^4 possible OTPs (i.e. just over a million), which is roughly 100 times the number of possible 4-digit PINs commonly used for bank cards.

4.2 Variants of the Scheme

OTP Transmission. In the scheme described above, the OTP is sent from the client to the mobile device in an SMS message. Whilst convenient, this has cost implications and may also involve a delay of a few seconds. Possible alternatives include sending it via Bluetooth, infrared or a USB/serial cable. Such approaches have the advantage of avoiding the SMS messaging costs but require both devices

[6] Note that if the mobile phone and/or the SIM card are PIN-protected, then the user must first enter the correct PIN(s).

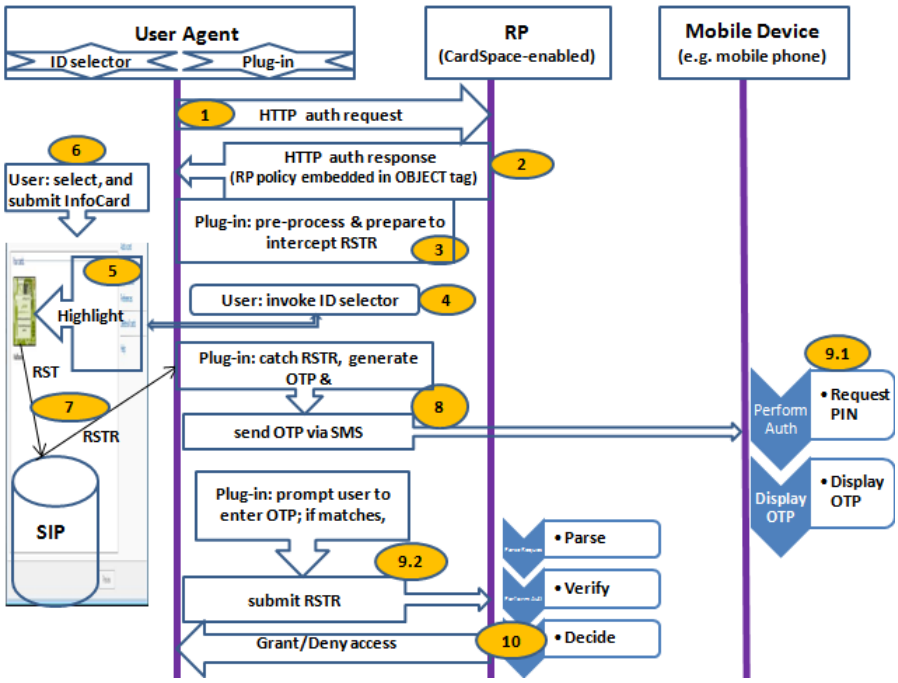

Fig. 2. Protocol exchanges

to support the relevant technologies. The main disadvantage of such approaches is the need to install a special application on the phone; this will rule out non-smart phones, and significantly increase the complexity of setting up the scheme.

A further alternative would be to use a messaging service other than SMS for the OTP transfer (e.g. instant messaging or email); like the use of the SMS service, such an approach would avoid the need to install any new applications on the phone.

OTP Entry. In the scheme as described above, the user manually enters the OTP, which is potentially inconvenient and time-consuming (although the use of a 4-digit PIN, as described in section 4.1, should minimise inconvenience). An alternative would be to send the OTP back automatically, e.g. via an SMS message sent to the SMS gateway, from where the adaptor could retrieve it. Whilst convenient, such a process could be costly, since use of the SMS gateway would incur additional messaging costs.

RSTR. As part of step 9 of section 3.2 the adaptor could create a new SAML token containing the RSTR produced by the SIP and an additional SAML field indicating that the user has been authenticated using an SMS-transmitted OTP. Of course, the RP would need to be modified to be able to process such a token,

although this would be straightforward. This authentication statement would give the RP added assurance of user authenticity.

4.3 Advantages

Like other OTP-based authentication systems, the proposed scheme reduces exposure to shoulder-surfing attacks and also helps to thwart key loggers.

The scheme does not require users to remember new passwords for each new account; this could reduce the risk of password re-use, writing passwords down in insecure ways, and use of easily-guessable passwords.

In addition to strengthening user authentication, the scheme could also serve as an intrusion detector. If the user receives an unexpected OTP, then it could be deduced that there is a security breach.

Finally, the scheme operates transparently to external parties, and hence does not require any changes to RPs or identity selectors.

5 Security Analysis

5.1 Threats to the Mobile Device

If an unprotected mobile phone or SIM is lost, stolen or borrowed, then it might be possible to access an OTP from the SMS inbox. However, this will be of no value without access to the corresponding PC (and the OTP will expire a short time after generation). Moreover, a lost phone or SIM is likely to be reported by its owner, causing the SIM to be deactivated, which means that the usefulness of such a stolen device for impersonating a user will be very limited.

5.2 Threats to the Supporting Infrastructure

An attacker with temporary access to the PC but without the mobile phone could attempt to intercept the OTP whilst it is being transmitted from the PC to the phone. However, the communication link between the SMS gateway and the PC is protected using HTTPS, and the connection between the visited mobile network and the mobile phone is protected by the air interface encryption mechanism of the mobile network [8, 9]. This leaves the SMS gateway and the SMS network itself as the only sources of such a threat, and routinely compromising either the gateway or the SMS network for such a purpose seems unlikely to be realistic in practice.

5.3 Threats to the PC

Exhausting the User's SMS Credit. An adversary who has access to the user's PC but does not possess the user's mobile phone could cause the system to repeatedly send SMS messages, resulting in exhaustion of the user's SIM credit at the SMS gateway. This risk can be mitigated in the following ways.

1. If a user receives an unexpected SMS containing an OTP, the user should immediately change their password at the SMS gateway. This will deny the adversary the ability to send any further SMS messages from the user's PC.
2. The browser extension could implement a simple, client-based, lock-out mechanism using cookies. That is, if the correct OTP is not entered within three attempts, the browser extension could write a persistent[7] cookie to the client PC which will cause the current attempt to log in to the RP to be terminated. The browser extension would then generate a special lock-out OTP and send it to the user's mobile phone. The next time that the user attempts to log in to the same domain, the browser extension (before invoking the CIdS) would prompt the user to enter the lock-out OTP, and would only proceed if the correct OTP is entered. Although this solution may help to discourage an attacker, it is not foolproof since cookies could be manually deleted on the client machine, and an attacker could arrange for OTP-bearing SMSs to be sent to a large number of different domains.

Disabling the Browser Extension. If the system is configured so that it is possible to disable the OTP adaptor, then a knowledgeable intruder could defeat the protection provided by the scheme. Therefore, a robust implementation of the scheme proposed in section 3.2 must not allow an adversary to disable it. That is, the system must be configured to oblige users to use CardSpace coupled with the OTP adaptor.

Browser extensions can be enabled/disabled at will by anyone who has access to a Windows user account. So an adversary with access to InfoCards could simply disable the browser extension to cause CardSpace to operate normally.

It may be possible to remove this threat, at least partially, by installing the browser extension so that administrator privileges are required to disable it, and also persuading the PC owner to log in using a non-administrator account. It may also be possible to make use of UAC[8] (User Account Control), so that disabling a browser extension causes Windows to prompt the user for an administrator password.

Ultimately, it would be desirable to implement the scheme described in section 3.2 as an integral part of CardSpace, thereby negating this threat. In such a scenario, each InfoCard might be given a selectable field to indicate whether SMS-based authentication is required. A user could thus choose to SMS-protect an important InfoCard by simply selecting the appropriate field.

Exploiting CardSpace Backup Facilities. The CardSpace backup facilities could be exploited to allow an InfoCard to be exported from one PC to another. An attacker could, for example, export a personal card to a USB memory stick,

[7] Persistent cookies can survive across a number of sessions, including after exiting the browser and/or after a machine reboot. Such cookies have an expiry date; if a cookie expires it is deleted.

[8] `http://technet.microsoft.com/en-us/library/cc709691(WS.10).aspx#BKMK_S1`

and then reload the card on his or her own PC in order to impersonate the card owner. An exported card could also be transferred as an email attachment. This risk could be mitigated using countermeasures similar to those discussed above.

6 Prototype Realisation

We next give details of a prototype implementation of the scheme. The prototype is coded in JavaScript, chosen because its wide adoption should simplify the task of porting the prototype to a range of other browsers. It uses the Document Object Model (DOM) to inspect and manipulate HTML pages and XML documents. The JavaScript code is executed using a C#-driven browser helper object (BHO), a DLL (Dynamic-Link Library) module designed as a plug-in for IE. Once installed, the BHO attaches itself to IE, thus gaining access to the current page's DOM. Note that the scheme operates with both the CardSpace and the Higgins[9] identity selectors without any modification.

6.1 User Registration

Prior to use, the prototype user must have accounts with a CardSpace RP and an SMS gateway service provider, e.g. Clickatell (`clickatell.com`). The prototype provides step-by-step instructions in order to assist the user in inserting their mobile phone number and their SMS account login details (e.g. username and password) into the plug-in source code.

6.2 Prototype Operation

In this section we consider specific operational aspects of the prototype. We refer throughout to the numbered protocol steps given in section 3.2 (see also figure 2).

In step 3 the plug-in uses the DOM to perform the following processes.

3.1 It scans the web page in the following way[10].

(a) It searches through the HTML elements of the web page to detect whether any HTML forms are present. If so, it searches each form, scanning through each of its child elements for an HTML object tag.

(b) If an object tag is found, it retrieves and examines its type. If it is of type 'application/x-informationCard' (which indicates website support for CardSpace), it continues; otherwise it aborts.

(c) It retrieves and stores in a cookie the name attribute of the CardSpace object tag. This is important since the RP server will use this name to retrieve the token from the HTTP POST array.

[9] `http://wiki.eclipse.org/GTK_Selector_1.1-Win`

[10] The CardSpace user guide [6] specifies two HTML extension formats that can be used to invoke the CIdS from a web page, both of which involve placing the CardSpace object tag inside an HTML form. This motivates the choice of the web page search method (see also [4, 10]).

3.2 It embeds a JavaScript function in the head section of the HTML page to intercept the RSTR.
3.3 It obtains the action attribute of the CardSpace HTML form and stores it in a cookie. This attribute specifies the URL of the CardSpace RP server to which the RSTR must be forwarded for processing. If the attribute is not a fully qualified domain name address, the JavaScript inherent properties, e.g. *document.location.protocol* and/or *document.location.host*, are used to help reconstruct the full URL address.
3.4 It changes the current action attribute of the CardSpace HTML form to point to the newly created 'interception' function (see step 3.2 above).

In step 8 the plug-in uses the DOM to perform the following steps.

8.1 It intercepts the RSTR sent by the CIdS using the added function.
8.2 It generates a 4-character, random OTP (see section 4.1). It also starts a 10-minute time counter.
8.3 It builds an HTTPS-based URL, inserting the user's mobile phone number, the user's account login details, and the OTP.
8.4 It automatically invokes the URL in a new, small browser window. This process will cause the OTP to be sent to the SMS gateway via a secure TLS/SSL channel. On receipt of the OTP, the SMS gateway delivers it to the user's mobile phone in an SMS message.
8.5 It prompts the user to enter the OTP, using a JavaScript pop-up box.
8.6 It verifies the user-entered OTP by comparing it with the version it previously generated (in step 8.2), ensuring that the OTP has been entered within the 10-minute time window. If the verification succeeds it proceeds to the next step. If the verification fails, the user is allowed to try again. However, if the verification fails for three successive OTP entry attempts, the plug-in terminates the login process and writes a persistent cookie to prevent the user from logging into this RP using the same browser for a defined time period, e.g. 24 hours. This process operates as follows.
On the first occasion that the system is used with a particular RP, or if the previously written cookie has expired and been deleted, the plug-in writes a persistent cookie containing the number of failed OTP entry attempts for this RP (i.e. either zero if the attempt is successful or one if the attempt fails) and with a lifetime of 24 hours. Whenever the system is used subsequently the presence of this cookie is checked; if it is present then the current number of failed OTP entry attempts it records is checked — if it is equal to three then no SMS is sent and the RSTR is blocked, i.e. the system is locked out and can only be unlocked if the user enters the special lockout OTP. If it is less than three then the system proceeds. If the OTP entry attempt succeeds then a new cookie is written containing the value zero; if the OTP entry attempt fails, then a new cookie is written containing a value one larger than the previous value.
8.7 It creates an 'invisible' HTML form with method attribute set to 'POST'.
8.8 It writes the entire RSTR message into the invisible HTML form as a hidden variable, with the name attribute of this variable set to the CardSpace object tag's name (see step 3.1.c).

8.9 It writes the end-point URL of the CardSpace-enabled RP into the action attribute of the invisible form (see step 3.3).
8.10 Finally, it auto-submits the HTML form (transparently to the user), using the JavaScript inherent method 'submit'.

6.3 Practical Issues

The plug-in must scan every HTML web page to check whether it supports CardSpace, and this may affect system performance. However, informal tests on the prototype suggest that this is not a serious issue. In addition, the plug-in can be configured so that it only operates with certain websites.

If the web browser is compromised, then an adversary could steal the RSTR and the OTP, block the user-RP connection, and submit the token, thus impersonating the user. If the RP does not use https, then the RSTR will not be encrypted. Assuming that the web browser is not a secure environment, then it may be possible for a malicious plug-in or some other type of malware to get access to sensitive information disclosed by the plaintext RSTR. However, the same risks apply when manually entering credentials (e.g. username-password) into the browser [11].

Finally note that some older browsers (or browsers with scripting disabled) may not be able to run the prototype plug-in, as it was built using JavaScript. However, most modern browsers support JavaScript (or ECMAscript), and hence building the prototype in JavaScript is not a major usability obstacle.

7 Related Work

Using a mobile device as a means of user authentication is attractive because of the ubiquity of mobile phones, and many such schemes have been proposed. Examples of schemes in which a mobile phone is used to authenticate a user to a remote server include the following.

- Hart et al. [11] proposed a scheme in which user credentials (i.e. username and password) are stored in a Java-enabled SIM card. When the user visits a website, the browser extension requests the site's user credentials from an SMS gateway, which then sends a specially formatted SMS message to the appropriate SIM card. The SIM card responds with another SMS message containing the requested credentials, and the SMS gateway forwards them to the browser extension via an HTTPS channel. The browser extension then auto-submits them to the visited site. The scheme requires the user to possess a SIM capable of hosting an application, and for the user to load an appropriate application into it. It also has an SMS messaging cost at least twice that of the scheme described in this paper.
- Wu et al. [12] and Jammalamadaka et al. [13] proposed schemes involving a combination of a third party proxy, which stores the user credentials, and a mobile phone. The schemes are designed for use in cases where an untrusted

PC, e.g. in an Internet kiosk, is used to access a remote website, and they avoid the need for the user to enter long-term secret credentials into such a PC (see also [14]). The phone is used to explicitly authorise the proxy to release the credentials to the remote website. Unfortunately, not only is the use of a proxy a potential security and reliability threat, but the PC must be configured to use the proxy. This latter requirement is not only potentially inconvenient, but in some cases may be impossible to meet since the user may not have the necessary permissions to change the browser settings.

- Florêncio and Herley proposed 'URRSA' [15], an OTP-enhanced service (based on a reverse proxy [16]) that allows users to access password-protected websites. The URRSA service does not require changes to login servers. A list of 10 different encrypted copies of a long-term user password (effectively OTPs) is generated and sent to the user's mobile phone using SMS; the corresponding decryption keys are stored at the URRSA server. A user wishing to access a protected site first navigates to the URRSA site and enters the URL and userID of the account to be accessed. The user then enters the appropriate OTP from the current list, allowing the URRSA server to decrypt and temporarily store the real password. The URRSA server then fetches the previously registered login page and prompts the user to click the submit button; the login process then proceeds. The user process for this scheme is relatively complex, and new lists will need to be downloaded fairly frequently, increasing the burden on the user.
- Aloul et al. [17] proposed a system that involves using a PIN-protected mobile phone as a token for OTP generation. Additionally, an SMS-based mechanism is implemented as both a backup mechanism for retrieving the OTP and as a possible means of client-server synchronisation. This method requires both the client and server to pay to send SMS messages. Unlike the scheme described here, the mobile phone must be J2ME-enabled, and, prior to use, the user must install a special application in the phone.
- Mannan et al. [18] and Alqattan et al. [19] proposed similar schemes in which the entry of user authentication credentials is accomplished using a trusted handheld device, e.g. a PIN-protected mobile phone. For instance, in the 'MP-Auth' scheme [18], the mobile device encrypts the password using the end server's public key before passing it via an untrusted machine to the remote server. However, unlike the scheme described in this paper, these schemes require changes to login servers and also require users to possess J2ME-enabled mobile phones.
- Schuba et al. [20] proposed the 'Internet ID' approach, in which a mobile phone is used to provide user authentication to a Liberty IdP. We outline the variant most similar to the scheme described above. A Liberty IdP generates a random sequence of symbols, and sends them to the user's mobile phone in an SMS message. Simultaneously, these symbols are shown on the PC browser, and the user is required to confirm to the phone that the browser-displayed symbols are the same as those in the SMS message, e.g. by clicking a link on the WAP page on the mobile phone. Although this system does not

require the user to type anything, it does require changes to the operation of Liberty IdPs.
- Jørstad et al. [21] proposed a scheme which supports interoperation between CardSpace and Liberty. It uses a mobile phone for user authentication to the IdP; the IdP sends an SMS message to the user, and, in order to be authenticated, the user must confirm receipt of the message. Much like the 'Internet ID' approach [20], this method requires changes to the operation of the IdP.

Examples of schemes in which a mobile phone is used to authenticate the user to a local PC include the following.

- Lach [1] proposed 'MOTH', a scheme in which a workstation and a mobile device communicate using Bluetooth, and authentication is realised using digital signatures. Unlike in our scheme, the mobile device in the MOTH system must be able to run Java midlets. To avoid an attacker bypassing the scheme, a MOTH-conformant PC must be configured to only use the MOTH service for authentication, and not to fall back to password authentication. Similarly, the scheme described in this paper must be configured to oblige the use of the adaptor with CardSpace (see section 5.3). In MOTH, binding a user to a public key remains a challenge.
- Abdulhameed et al. [22] proposed a method which uses a Bluetooth-enabled mobile phone. The user's PC communicates with the phone via a Bluetooth link, and public key cryptographic techniques are used to provide mutual authentication between the PC and the phone. The PC periodically senses the phone to ensure that the user is still present; if the mobile phone moves out of range, the PC is configured to take certain measures to raise the security level. It is unclear from the paper whether this form of authentication could be disabled by an attacker so that the PC reverts to password-based user authentication, a possible means of circumventing the scheme. Not only must the mobile phone be Bluetooth-enabled, but it must also support Java to provide certain cryptographic and authentication services.

Finally note that the scheme proposed in this paper falls somewhere in between the two classes described above, in that it provides authentication to a local PC in such a way that it enables authentication to a remote site to continue in a more secure way.

8 Conclusions and Future Work

In this paper we have proposed a simple and novel scheme for using a mobile device to enhance CardSpace authentication. During the process of user authentication on a PC using CardSpace, a random and short-lived one-time password is sent to the mobile device; this must then be entered into the PC by the user. The scheme does not require any changes to login servers, the CardSpace identity selector, or to the mobile device itself. We have given details of a proof-of-concept prototype. Security and operational analyses have also been provided.

Planned future work includes exploring the possibility of extending the scheme to operate with other client-enabled identity management systems, including password managers. We also plan to develop the prototype in various ways, including:

- preventing it being disabled by an unauthorised PC user;
- providing support for OTP transfer to the mobile via Bluetooth and/or infrared; and
- supporting automated OTP entry from the mobile.

Acknowledgements

The first author is sponsored by the Diwan of Royal Court, Sultanate of Oman. The helpful comments provided by anonymous referees are gratefully acknowledged.

References

1. Lach, J.: Using mobile devices for user authentication. In Kwiecien, A., Gaj, P., Stera, P., eds.: Computer Networks, 17th Conference, CN 2010, Ustrón, Poland, June 15–19, 2010, Proceedings. Volume 79 of Communications in Computer and Information Science. Springer, Berlin, Heidelberg, 263–268 (2010)
2. Bertocci, V., Serack, G., Baker, C.: Understanding Windows CardSpace: An Introduction to the Concepts and Challenges of Digital Identities. Addison-Wesley, Reading, Massachusetts (2008)
3. Mercuri, M.: Beginning Information Cards and CardSpace: From Novice to Professional. Apress, New York (2007)
4. Al-Sinani, H.S., Mitchell, C.J.: Using CardSpace as a password manager. In de Leeuw, E., Fischer-Hübner, S., Fritsch, L., eds.: Proceedings of IFIP IDMAN 2010 — the second IFIP Conference on Policies and Research in Identity Management, November 18–19, 2010, Oslo, Norway. Volume 343 of IFIP Advances in Information and Communication Technology, Springer, Boston, 18–30 (2010)
5. Al-Sinani, H.S., Alrodhan, W.A., Mitchell, C.J.: CardSpace-Liberty integration for CardSpace users. In Klingenstein, K., Ellison, C.M., eds.: Proceedings of the 9th Symposium on Identity and Trust on the Internet, (IDtrust'10), Gaithersburg, Maryland, USA, April 13–15, 2010, ACM, New York, NY, 12–25 (2010)
6. Jones, M.B.: A Guide to Using the Identity Selector Interoperability Profile V1.5 within Web Applications and Browsers. Microsoft Corporation. (2008)
7. Jones, M.B., McIntosh, M., (editors): Identity Metasystem Interoperability Version 1.0 (IMI 1.0). OASIS Standard. (2009) `http://docs.oasis-open.org/imi/identity/v1.0/identity.html`.
8. Guthery, S.B., Cronin, M.J.: Mobile Application Development with SMS and SIM Toolkit. McGraw-Hill, New York (2002)
9. Le Bodic, G.: Mobile Messaging Technologies and Services SMS, EMS and MMS. Wiley, Chichester (2003)
10. Al-Sinani, H.S., Mitchell, C.J.: Implementing PassCard — a CardSpace-based Password Manager. Technical Report: RHUL–MA–2010–15 (Department of Mathematics, Royal Holloway, University of London). (2010) `http://www.ma.rhul.ac.uk/static/techrep/2010/RHUL-MA-2010-15.pdf`.

11. Hart, J., Markantonakis, K., Mayes, K.: Website credential storage and two-factor web authentication with a Java SIM. In Samarati, P., Tunstall, M., Posegga, J., Markantonakis, K., Sauveron, D., eds.: Proceedings, Information Security Theory and Practices. Security and Privacy of Pervasive Systems and Smart Devices, 4th IFIP WG 11.2 International Workshop, WISTP 2010, Passau, Germany, April 12–14, 2010. Volume 6033 of Lecture Notes in Computer Science., Springer, Berlin, Heidelberg, 229–236 (2010)
12. Wu, M., Garfinkel, S., Miller, R.: Secure web authentication with mobile phones. In: DIMACS Workshop on Usable Privacy and Security Systems. (2004) `http://homepages.mcs.vuw.ac.nz/~ian/shared/papers/secureweb.pdf`.
13. Jammalamadaka, R., van der Horst, T., Mehrotra, S., Seamons, K., Venkasubramanian, N.: Delegate: A proxy based architecture for secure website access from an untrusted machine. In: ACSAC 2006: Proceedings of the 22nd Annual Computer Security Applications Conference, IEEE Computer Society, Washington, 57–66 (2006)
14. Pashalidis, A., Mitchell, C.J.: Impostor: A single sign-on system for use from untrusted devices. In: Proceedings of IEEE Globecom 2004, Global Telecommunications Conference, Dallas, Texas, USA, November/December 2004. Volume 4., IEEE Press, 2191–2195 (2004)
15. Florêncio, D., Herley, C.: One-time password access to any server without changing the server. In Wu, T.C., Lei, C., Rijmen, V., Lee, D., eds.: Proceedings of the Information Security, 11th International Conference, ISC 2008, Taipei, Taiwan, September 15–18, 2008. Volume 5222 of Lecture Notes in Computer Science. Springer-Verlag, Berlin, Heidelberg, 401–420 (2008)
16. Luotonen, A.: Web Proxy Servers. Prentice Hall PTR, New Jersey (1997)
17. Aloul, F., Zahidi, S., El-Hajj, W.: Two factor authentication using mobile phones. In: AICCSA 2009: Proceedings of the IEEE/ACS International Conference on Computer Systems and Applications, Rabat, Morroco, IEEE, 641–644 (2009)
18. Mannan, M., Oorschot, P.V.: Using a personal device to strengthen password authentication from an untrusted computer. In Dietrich, S., Dhamija, R., eds.: Financial Cryptography and Data Security. Volume 4886 of Lecture Notes in Computer Science. Springer, Berlin, Heidelberg, 88–103 (2007)
19. Alqattan, A., Kaviani, N., Lewis, P., Pearson, N.: A two-Factor Authentication System using Mobile Devices to Protect against Untrusted Public Computers. University of British Columbia, Canada. (2007) `http://courses.ece.ubc.ca/412/term_project/reports/2007-fall/A_Two-Factor_Authentication_System_Using_Mobile%20_Devices_to_Protect_against_Untrusted_Public_Computers.pdf`.
20. Schuba, M., Gerstenberger, V., Lahaije, P.: Internet ID — Flexible Re-use of Mobile Phone Authentication Security for Service Access. (2004) `http://www.ericsson.com/res/thecompany/docs/journal_conference_papers/service_layer/internet_id_nordsec.pdf`.
21. Jørstad, I., Van Thuan, D., Jønvik, T., Van Thanh, D.: Bridging CardSpace and Liberty Alliance with SIM authentication. In: Proceedings of the 10th International Conference on Intelligence in Next Generation Networks (ICIN 07), Adera, Pessac, 8–13 (2007)
22. Abdelhameed, R., Khatun, S., Ali, B., Ramli, A.: Authentication model based bluetooth-enabled mobile phone. Journal of Computer Science **1(2)** (2005) 200–203