Preventing Phishing Attacks Using Trusted Computing Technology

Adil Alsaid and Chris J. Mitchell

Information Security Group, Royal Holloway, University of London
Egham, Surrey TW20 0EX, UK.
e-mail: A.Alsaid@rhul.ac.uk

Abstract

Most secure web sites use the SSL/TLS protocol for server authentication. SSL/TLS supports mutual authentication, i.e. both server and client authentication. However, this optional feature of SSL/TLS is not used by most web sites because not every client has a certified public key. Instead user authentication is typically achieved by sending a password to the server after the establishment of an SSL-protected channel. Certain attacks rely on this fact, such as web spoofing and phishing attacks. In this paper the issue of online user authentication is discussed, and a method for online user authentication using trusted computing platforms is proposed. The proposed approach makes a class of phishing attacks ineffective; moreover, the proposed method can also be used to protect against other online attacks.

Keywords

PKI, SSL/TLS, Web browsers, e-commerce, TCPA, TCG, TPM, Phishing.

1 Introduction

Online user authentication is required by most web applications. The authentication level that is required depends on the services being provided by a particular web application. For example, a simple general purpose web forum may use cleartext user credentials for authentication. The authentication information may be stored on the user machine, e.g. using cookies, for subsequent authentication without requiring the user to resubmit authentication data. By contrast, in a security-sensitive online application (such as online banking), the use of a cleartext credential would not be sufficient, since an attacker could capture the user credential by monitoring the communications channel. In such a case a more secure authentication method is required.

Phishing/Pharming/Web Spoofing (Felten et al. 1997, Geer 2005, Anti-Phishing Working Group 2005) attacks are widely used to gather user personal information, including usernames and passwords. One possible scenario for such an attack arises when an attacker creates a spoofed web site that looks identical to a genuine web site, and convinces the victim to visit the spoofed web site (e.g. by including a URL in a faked email). When the victim navigates to the spoofed web site, an information gathering page is displayed to get the victim's personal information. Once the victim's authentication credentials, e.g. username and password, have been captured, the attacker can impersonate the user to the genuine web site. Other possible scenarios exist; however all possible scenarios have the same main objective, i.e. the capture of a user credential.

In this paper we propose a method to enable SSL client-side authentication using functionality available in Trusted Computing Group (TCG) compliant platforms. Specifically, we propose the use of cryptographic functions provided by the trusted platform module (TPM) present on

any TCG-compliant platform. The rest of the paper is organised as follows. Sections 2 and 3 review the Secure Socket Layer/Transport Layer Security (SSL/TLS) protocol and trusted computing platforms. Section 4 briefly introduces a method to prevent phishing attacks using trusted computing. A proposed method for online user authentication using trusted computing is then discussed in detail in Section 5, and a security analysis is given in Section 6. Finally, Section 7 concludes the paper.

2 SSL/TLS

The SSL/TLS protocol provides data integrity and data confidentiality via the 'record protocol' and entity authentication by means of the 'handshake protocol'. The part of the protocol that is of interest here is the handshake protocol. The main task of the handshake protocol is to provide entity authentication and to set up the parameters required for subsequent communications security. Specifically, this involves establishing the master secret and setting up the CipherSpec.

One optional element of SSL/TLS that is of particular interest here is the CertificateRequest protocol message, which can be used to request a client-side web browser to provide a public key certificate for authentication of the client to the server. If sent by the web server, the web browser replies by sending a copy of the client certificate selected by the user, and a proof of knowledge of the associated private key, i.e. by signing the 'CertificateVerify' SSL handshake message. However, this element of the protocol is typically not used, since most users do not have personal public key certificates.

3 Trusted Computing and TPMs

In this section we introduce the functionality of those components of the TCG specifications that are relevant to the protocol proposed in section 5. Detailed descriptions and specifications of TCG can, for example, be found in (Mitchell 2005, Trusted Computing Group 2005). Note that throughout this paper we assume the use of a TPM conforming to version 1.2 of the TCG specifications.

3.1 Trusted Platform Module (TPM)

The Trusted Platform Module (TPM) is the 'root of trust' for a trusted platform. It has certain cryptographic capabilities, such as RSA key generation and encryption, SHA-1 hashing and a random number generator. It is typically implemented in the form of a chip attached to a PC motherboard. It contains a set of Platform Configuration Registers (PCRs) used to store and report the state of the TCG-enabled platform. It has non-volatile memory that is used to store private keys and identity information known only to the TPM. For privacy reasons, the TPM can support more than one identity, as illustrated below.

3.2 TPM Identity

Every TPM has a unique RSA key pair called the endorsement key (EK). The EK would typically be created by the manufacturer of the TPM, and then embedded into the TPM. The private part of the EK (the PRIVEK) is stored in a TPM-shielded location and never leaves the TPM. Access to the PRIVEK is achieved through the use of TPM capabilities, which are exposed to software running on the host. The public part of the EK (PUBEK) could be used to identify a platform, and hence export of PUBEK could be a significant threat to user privacy. A TPM Attestation Identity Key (AIK) can be used to overcome the privacy concerns associated

with platform identification. An AIK is a 2048-bit RSA key pair used exclusively for signatures, and such a key pair can be generated by a TPM at any time. A TPM may have more than one AIK, each of which functions as a different pseudonym for the platform. In order to be able to prove that an AIK belongs to a trusted platform, the TPM must obtain a certificate for the AIK public key from a trusted third party, i.e. a special entity known as a Privacy CA.

4 Preventing Phishing Attacks Using Trusted Computing

This section outlines a method to prevent phishing attacks. This high-level description is followed by a discussion of existing solutions to this problem. A comparison of the suggested method with these other approaches is then given.

4.1 Enabling Client-side Authentication

As discussed in Section 2, SSL/TLS client side authentication is typically not performed because of the lack of a client public key certificate. In this section, a method to automate the process of acquiring a client-side certificate is proposed, thereby allowing client-side authentication to take place. The proposed method utilises a subset of the features of a TPM conforming to version 1.2 of the TCG specifications to create an SSL client-side certificate. The method is outlined in Figure 1, and a detailed description of this method is presented in Section 5. The approach we describe requires the client browser to interact both with third parties and with the TPM in order to obtain the necessary certificate. To simplify the process for users, this could be achieved simply by downloading and installing an appropriate browser plug-in (from a trusted source).

4.1.1 Setup Phase

We suppose that the browser maintains a list of web site/certificate associations, which we refer to as the certificate table. This 'certificate table' is used to send the appropriate client certificate to a particular web site. The mapping list is not strictly necessary, and the user could have one certificate that is sent to all web sites requesting a user certificate. However, having a different certificate for each web site helps preserve user privacy.

When a web server requests a client SSL certificate, the web browser searches the 'certificate table' for an entry that matches the requested web site. If there is no client certificate that corresponds to the requested web site, the browser requests the TPM to create a new AIK, using the TPM_MakeIdentity command. Once the AIK is created, the browser sends the public key part of the AIK to a Privacy CA along with evidence that the identity was generated on a genuine TPM. The Privacy CA inspects the received value and verifies that it was generated on a genuine TPM. It then signs the received public key using its private key, and sends the resulting certificate back to the browser.

After receiving the public key certificate from the Privacy CA, the trusted platform generates another key pair and certifies the newly created public key using the AIK private key (see Section 5 for more details). The browser then sends the newly generated certificate to the web server in response to the CertificateRequest SSL handshake protocol message.

The problem remains of securely associating the user identifier in the certificate with the user name held by the web server. We would suggest that the following procedure is used to 'register' a user certificate. If the user has already established a user name and password with the web site, then, once the user certificate has been received and verified by the server, and the SSL connection established, the user name and password are transferred to the web server

(exactly as in the case where no client-side authentication is provided). Once the user name and password have been verified, the name in the user certificate is stored by the web server in conjunction with the user name. If no user name and password have previously been established, then, once the SSL connection is established, the user name and password are transferred, and again an association is set up in the server database between the user name and the name in the user certificate. This combination of user certificate and username/password enables the web site to use a two-factor process to authenticate the user. This provides an additional level of security.

4.1.2 Using the Client Certificate

After completing the setup phase, mutual authentication can be achieved whenever necessary. When the user visits the secure web site, the user certificate along with the user's signature is sent to authenticate the user. In addition, and after successful completion of the SSL exchanges, the web site may require the user to provide his or her password. Having two-factor authentication minimises the risk of identity theft and the dangers of a phishing attack.

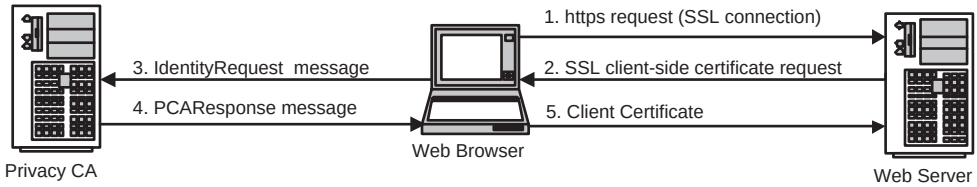

Figure 1: Obtaining a client certificate

4.2 Existing Solutions to Phishing Attacks

This section briefly reviews other solutions to phishing attacks.

4.2.1 Client Authentication

In a method proposed by Verisign, the user must obtain a public key certificate from a certification authority and install it in the user's personal certificate store. When client-side authentication is required, i.e. through SSL client-side authentication, the browser prompts the user to select a certificate from the user's personal certificate store. The browser then sends the selected certificate to the web server as part of the SSL exchange, in reply to the 'CertificateRequest' SSL message, together with a signed CertificateVerify field. The web server inspects the received certificate and the signature in the 'CertificateVerify' message, and grants access accordingly.

This is a typical scenario for the use of SSL client-side authentication. There are two main problems with this approach. Firstly, the user is required to generate a key pair and to obtain a certificate for the public key from a trusted CA. This is a non-trivial process for a naïve user, especially as the public key typically needs to be transferred to the CA by some secure means. Secondly, even if a key pair is securely generated, and a public key certificate successfully obtained, the problem remains of storing the private key. Storing the private key unprotected on the user PC leaves open the possibility of compromise.

Security tokens can be used to provide a secure and controlled storage medium for client private keys. A security token is a small hardware device such as smart card or USB token. Security tokens typically provide a range of key storage and cryptography functions. Security

tokens provide two-factor authentication, i.e. something you have (the token) and something you know (the user password). The computing platform would need to be equipped with the necessary hardware to interact with the token, such as a smart card reader or a USB interface.

4.2.2 Visual Server Authentication

In the approach proposed by Dhamija and Tygar (Dhamija & Tygar 2005), when a user registers for an online service for the first time, he/she is requested to choose a unique image that is known to the web site and the user. When the user tries to login by providing his/her username, the site displays the user-chosen image to help the user visually authenticate the server. If the user-chosen image matches the displayed image, the user continues by entering his/her password. This method relies on the fact that spoofed web sites will not be able to display the user's unique image.

4.3 Advantages of the novel approach

The proposed method avoids the two main problems associated with the use of client-side SSL authentication, as outlined in section 4.2.1. That is, the potentially problematic process of generating a key pair and obtaining a public key certificate can be made completely transparent to the user, and the problem of secure storage of the private key is solved by storing it within the TPM. Moreover, the TPM provides means to control the use of the stored private keys. Of course, use of a secure token also avoids some of these issues, but is nevertheless a potentially costly and awkward solution, which can never be completely user-transparent.

In the visual server authentication method, the user is required to remember a web site/image association to be able to visually authenticate a web server. Moreover, the method proposed in (Dhamija & Tygar 2005) requires some changes to be made to both the web server and the SSL protocol.

5 SSL/TLS Authentication Using Trusted Computing

In the following two sections the TCG-based approach to client-side authentication is discussed in greater detail. The description is divided into two parts, covering client certificate creation and subsequent use of the client certificate to achieve client authentication to the server.

5.1 Creating Client Certificates

We first describe what happens when a client visits a server site for the first time (or at least the first time that this novel authentication approach is to be used). Note that we are assuming that the browser interacts with the TPM using the TSS and the required interfaces.

1. The browser executes the TPM_MakeIdentity command to generate a new AIK key pair, i.e. to create a new TPM identity. Only the owner of the TPM can create a new TPM Identity, and owner authorisation is required (see Section 3.2). A successful execution of the command causes a new AIK to be generated within the TPM, together with something known as an identity-binding signature. An identity-binding signature is a special data structure signed using the newly created AIK private key. The identity-binding structure contains the newly created AIK public key, the new TPM name (pseudonym), and the public key of the Privacy CA.

2. The browser then executes the TSS_CollateIdentityRequest command to assemble all the data required by the Privacy CA to attest to the validity of the newly created TPM iden-

tity, and then sends the IdentityRequest message to the Privacy CA. The IdentityRequest message includes the identity-binding signature created in the previous step, and other data such as an endorsement credential, a platform credential and a conformance credential, i.e. the TCPA_IDENTITY_PROOF. The IdentityRequest message is symmetrically encrypted using a random session key, and the session key is asymmetrically encrypted using the public key of the Privacy CA.

3. When the Privacy CA receives the IdentityRequest message, it decrypts the session key using its private key and then decrypts the message using the session key. It then inspects the message to make sure that it was generated on a genuine TPM. If the Privacy CA is confident that the IdentityRequest message was generated by a genuine TPM, it replies by sending the PCAResponse message. The PCAResponse message includes an encrypted version of the identity credential, which is DER-encoded as an X.509 public key certificate. The identity credential is encrypted using a secret session key, where the secret session key is itself encrypted using the TPM PUBEK.

4. The browser executes the TPM_ActivateIdentity command to obtain the secret session key used to encrypt the identity credential. Since the session key is encrypted using the TPM PUBEK, only the TPM can decrypt the session key using the PRIVEK. Moreover, only the owner of the TPM can activate the new identity, since owner authorisation is required.

5. The browser executes the TSS_RecoverTPMIdentity command to decrypt the identity credential. The session key used to encrypt the identity credential and the encrypted identity credential are passed as parameters to the command. If the command executes successfully, the TSS_RecoverTPMIdentity command returns the decrypted identity credential.

6. The private part of the certified AIK, which never leaves the TPM, cannot be used to sign data external to the TPM, as discussed in Section 3.2. Hence the received certificate cannot be used for client authentication in the SSL protocol. For this reason, the browser should create another non-migratable key pair (B) by executing the TPM_CreateWrapKey command, and then use the TPM_CertifyKey command to sign the newly created key using the AIK created in step 1. According to the TCG specifications, the output of the TPM_CertifyKey command is a signature over a TPM_CERTIFY_INFO structure.

   That is, although this signed string has some of the properties of a certificate, i.e. the signature is computed over a public key, it is not in X.509 format. That is, it cannot be used as a certificate in an SSL exchange. One possible method to obtain an X.509 certificate for use in the SSL client authentication protocol is for the TPM to provide a new command that certifies keys and returns an X.509 certificate. However, we do not consider such a solution further here as it would require changes to the TCG specifications, and hence can only be a solution in the long term.

   A second possible method is to use the Subject Key Attestation Evidence (SKAE) X.509 certificate extension. The SKAE extension, as defined in (TCG Infrastructure Workgroup 2005), includes the certified identity credential obtained in the previous step and the TPM_CERTIFY_INFO structure, in addition to other fields. The web browser creates a certificate request for the public part of the newly created key (B) using either PKCS#10 or CRMF, and includes the SKAE extension as an attribute. It then submits the certificate request to a CA. When the CA receives the certificate request, it validates

Method	TPM Change	Client Change	Server Change
TPM Command	Yes	No	No
SKAE	No	No	Yes
Certificate Translation	No	No	No

Table 1: Comparison of X.509 Certificate creation methods

it and accordingly issues an X.509v.3 certificate with the SKAE extension and sends it back to the web browser. In order to use this method, the web server must be aware of the SKAE extension in order to validate and process the TPM-created public key certificate (i.e. the signed TPM_CERTIFY_INFO structure). One possible way to avoid the inclusion of the SKAE extension in the final certificate is for the CA to validate the SKAE extension before issuing the certificate and then, if valid, issue a certificate without the SKAE extension. In such a case, the web server would not need to be aware of the SKAE extension.

A third possible method is to generate an X.509 certificate from the signed TPM_CERTIFY_INFO structure using a certificate translation service (Borselius & Mitchell 2000). In this method, both the identity credential certificate and the TPM_CERTIFY_INFO structure are sent to a certificate translation server. The translation server inspects the received values and, if valid, converts the two structures into a single X.509 certificate. The use of a certificate translation server eliminates the need for web server changes, unlike in the SKAE extension method, since the web server will receive a normal X.509 certificate without any extensions. Table 1 summarises the three approaches for the generation of an X.509 certificate.

7. After creating the certificate, the browser requests the TPM to sign the 'CertificateVerify' SSL protocol message using the newly generated key, to provide a proof of possession of the certificate private key. The browser makes a call to the TPM_Sign command and passes it an MD5 hash of the handshake protocol messages. The browser uses the result to create the 'CertificateVerify' protocol message which is sent to the web server, along with the certificate created in the previous steps, in reply to the CertificateRequest SSL/TLS protocol message.

8. In addition to the SSL client-side authentication, the web server may request the user to provide a username/password combination to support two-factor authentication.

5.2 Using a Client Certificate

When the user visits a secure web site that requests a client-side certificate, the browser searches the sites/certificates mapping list for a certificate that corresponds to the visited web site. If no certificate is associated with the visited web site, the browser creates a new certificate as described in Section 5.1; otherwise, the browser executes the following steps.

1. The browser executes the TPM_LoadKey command to load the key associated with the visited secure web site.

2. The browser then executes the TPM_Sign command to generate the 'CertificateVerify' SSL handshake protocol message, and sends the certificate and the 'CertificateVerify' protocol messages to the web server to authenticate the client. The SSL/TLS protocol messages continue in the normal way, as described in Section 2.

6 Security Analysis

Secure storage for certificates and private keys is achieved by using the secure storage capabilities of the TPM. According to the TCG specifications, the private part of the AIK and the non-migratable keys never leave the TPM. Moreover, the use of the keys can be controlled by setting a password, or 'authdata', at the time of creation. The 'authdata' should be presented to the TPM whenever use of the keys is required.

Mobility of client certificates can be achieved by using the Certifiable Migratable Key (CMK) feature introduced in TPM version 1.2. The migration process is controlled to ensure that the key is moved between two TPMs. To create a CMK, the TPM_CMK_CreateKey TPM command must be executed. The TPM_CMK_CreateKey command is similar to the TPM_CreateWrapKey command, but owner authorisation is required. To migrate the key from one trusted platform to another, the TPM_MigrateKey command needs to be executed.

7 Conclusions and Future Work

This paper proposes a method for online user authentication using trusted computing. The proposed method requires no changes to be made to web servers or the SSL protocol; however, a Web browser (or a browser plugin) that supports a TCG-complaint platform is required. The proposed method achieves two-factor authentication by using both the client-side certificate and a username/password combination for authentication. In order to create a client certificate, the proposed method relies on a trusted third party, i.e. the Privacy CA. A prototype of the proposed trusted computing based solution is currently being planned.

Another possible method to authenticate clients with a TCG-enabled platform is to use Direct Anonymous Attestation (DAA) (Brickell et al. 2004). Using DAA to create a client SSL certificate requires changes to the SSL/TLS protocol, as discussed in (Balfe et al. 2005). The use of DAA to authenticate a user to a web server is a topic for future research.

8 References

Anti-Phishing Working Group (2005), *Phishing Activity Trends Report2*. http://www.antiphishing.org.

Balfe, S., Lakhani, A. & Paterson, K. (2005), Securing Peer-to-Peer networks using Trusted Computing, *in* C. J. Mitchell, ed., 'Trusted Computing', IEE Press, pp. 271–298.

Borselius, N. & Mitchell, C. J. (2000), Certificate translation, *in* 'Proceedings of NORDSEC 2000 — 5th Nordic Workshop on Secure IT Systems', Reykjavik, Iceland, pp. 289–300.

Brickell, E. F., Camenisch, J. & Chen, L. (2004), Direct anonymous attestation., *in* V. Atluri, B. Pfitzmann & P. D. McDaniel, eds, 'Proceedings of the 11th ACM Conference on Computer and Communications Security, CCS 2004', ACM, Washington, DC, USA, pp. 132–145.

Dhamija, R. & Tygar, J. (2005), The Battle Against Phishing: Dynamic Security Skins, *in* 'Symposium On Usable Privacy and Security (SOUPS) 2005', ACM, pp. 77–88.

Dierks, T. & Allen, C. (1999), The TLS Protocol 1.0, RFC 2246, Internet Engineering Task Force.

Felten, E. W., Balfanz, D., Dean, D. & Wallach, D. S. (1997), Web Spoofing: An Internet Con Game, *in* 'Proceedings of 20th National Information Systems Security Conference', pp. 95–103.

Geer, D. (2005), 'Security Technologies Go Phishing', *IEEE Computer Magazine* **38**(6), 18–21.

Mitchell, C. J., ed. (2005), *Trusted Computing*, IEE.

TCG Infrastructure Workgroup (2005), *Subject Key Attestation Evidence Extension Specification Version 1.0*.

Trusted Computing Group (2005), *TPM Specification, Version 1.2*.