Abstract

Web technology is used more and more for accessing organizational data. The benefit being that a well known interface is used to access all computers across the organization’s network. These accesses can also be secured to some extent by using the Internet standard SSL/TLS protocol.

Single sign-on systems allow users to login to a central security server, be provided with a ticket, and then use this ticket to access other resources across the system. The benefit of single sign-on systems is reduced management for system administrators and a more user-friendly system.

This paper describes our efforts in integrating the web technology with a single sign-on system. This allows us to combine their benefits. To achieve additional security in the system, and to allow mobility of users, we have incorporated Smart Card technology. The importance of the Smart Cards cannot be overstated. Using the Smart Card it is possible for users to roam and access the local resources in a secure way. On the downside, the difficulty of using Smart Cards is the ad-hoc lack of standardization within web technology.
Keywords: Access control, authentication, authorization, SESAME, Smart Card, SSL/TLS, World Wide Web.

1 Introduction

Single sign-on systems have been available for more than a decade now. The idea behind these systems is that instead of users having to authenticate to every system they want to access, the users are authenticated once to a central security server, and are provided with a ticket that can be used to access other resources. The advantages of this include the easier management for the administrators and the ease of use for the end-user.

A well known example of a single sign-on system is Kerberos [?], developed at MIT. Other examples of single sign-on technologies include DCE [?], KryptoKnight [?], DASS/SPX [?] and SESAME [?]. Kerberos, KryptoKnight and DCE use symmetric key cryptography for user authentication, whereas DASS/SPX supports public-key cryptography. The single sign-on system we use is based on SESAME. SESAME provides both approaches, but in our implementation we opted for the public-key based authentication because of its associated scaleability.

In the current SESAME V4 implementation, users are granted an RSA key pair and have to login using the program strong_login. The strong_login program performs the user authentication protocol on behalf of the user, and by default reads the user’s private key from the user’s workstation file system. Proving the knowledge of the private RSA key authenticates the user to the system.

In a previous project [?] the strong_login program has been modified to operate with Smart Cards. In this system, the user’s private RSA key is stored on the user’s Smart Card with all of the user’s RSA cryptographic operations occurring on the Smart Card. The goal of the system is that the user’s private RSA key never has to leave the Smart Card.

The paper is set out as follows. In Section 2 the previous Smart Card enabled security architecture is described, including the limitations of this system. In Section 3 a description is given of the new web-enabled security architecture combining the SSL/TLS protocol [?, ?] with the SESAME single sign-on architecture. It also describes where Smart Cards fit into the picture. Section 4 describes the issues related to integrating Smart Cards into both a Java Applet and standard browser. The next Section discusses related work. We finish with our conclusions and future work.

2 The Smart Card System

The Smart Card system works as shown in Figure 1. The user presents the Smart Card to the terminal. The Smart Card is the Gemplus GPK-4000 [?].
The *strong_login* program has been modified so that instead of reading the user’s private key from the file system, data is passed to the Smart Card, where the appropriate cryptographic operations take place.

Although this system works well, it has a limitation. It is currently only supported on client workstations running the Unix operating system. Our solution to this problem is to give the users the possibility to authenticate and obtain their PAC through a web browser. This new approach has the advantage that users are allowed to use a well known interface to all applications. The web browser allows it to store the SESAME ticket and forward it to the application servers, thus rendering access control to these servers straightforward to implement. Another requirement for the system is support for multiple Smart Cards. The current version supports only the Gemplus GPK-4000 card, but there is no reason not to support other RSA capable cards such as the Philips DX-Plus.

### 3 Overview of the Web Browser System

The web browser system described in this paper implements a web-enabled security architecture in which the services of both SSL/TLS [?, ?] and SESAME [?] are combined [?].

As in a regular secured web interaction, data confidentiality, data authentication, and server/client authentication are provided by the SSL/TLS protocol. Additionally, authorization and non-repudiation are provided by SESAME. Authorization is enforced by the Role Based Access Control (RBAC) mechanism [?] implemented by SESAME [?]. The major benefits of RBAC include:

- Role based access control provides a means of administering the access to data that is natural and consistent with the way most enterprises are organized and conduct business;
- the users can be granted membership to roles based on their compe-
sentence and responsibility;

- user memberships can be revoked or re-established at any time;
- the administrative functions can be applied centrally, locally or remotely;
- administrators can create roles and place constraints on role memberships;
- it is possible to define role hierarchies and thus the solution is very scalable.

The following paragraphs outline how the Smart Card enabled browser allows the user to sign-on to SESAME and then access resources across the system. The different steps are illustrated in Figure 2.

![Diagram of Smart Card integration into Web Browser]

**Figure 2: Integration of Smart Cards into Web Browser**

### 3.1 Step 1: Downloading the *strong_login* applet

The user contacts the SSL/TLS secured SESAME security server via a standard web browser. This server sends the *strong_login* applet in an authentic way to the client. SSL/TLS is used for authenticating this server and making sure the applet is not changed during the transmission. The latter could also be assured by digitally signing the applet (Code Signing).

### 3.2 Step 2: Obtaining a ticket (PAC)

The downloaded applet is used to perform SESAME’s *strong_login* protocol with the SESAME security server. This means that users provide their username and authenticate using their Smart Card. If authentication is
successful, they receive a ticket. In SESAME this ticket is called a Privilege Attribute Certificate (PAC) [?]. It contains the user’s privileges (roles), is valid for a limited time period only and is digitally signed by the SESAME security server (so it cannot be altered). This PAC thus constitutes a key element in the authorization scheme enforced by SESAME.

3.3 Step 3: Storing the PAC

The PAC has to be stored on the client’s system, so that it can be used later when the client has to provide credentials to an application server. The PAC is stored as a cookie [?], so that it is automatically sent to any web server in the SESAME domain.

3.4 Step 4: Performing RBAC

When the client sends a request to an SSL/TLS secured web server in the SESAME domain, the cookie (PAC) is sent along with the request. The client access is achieved through a Common Gateway Interface (CGI) program. The CGI program uses the PAC to decide whether to give the requested page (passed as a variable to the CGI program) or not.

More specifically, the CGI program verifies whether the PAC is valid, and whether the client is the legitimate owner of that PAC. The latter is done by cross checking SSL/TLS’s client authentication data to the information inside the PAC. In SESAME terminology the CGI program implements the function of the PAC Validation Facility (PVF).

3.5 Step 5: Delivering the information

If the client is authorized to access the resource, the information is sent. Note that Universal Resource Locators (URLs) within the delivered HTML pages must contain the CGI program and not point to the information directly. Moreover, the information should not be directly accessible anyway.

4 Smart Card Implementation

An important aspect in the whole system is the fact that Smart Cards need to be interfaced to the web browser. This way the user’s private RSA key doesn’t have to be stored on the workstation’s or PC’s hard disk.

In fact supporting Smart Cards does not only increase the level of security, it also allows users to roam and use their Smart Card on different systems, even outside the company (the only condition is that the client’s system needs to have a Smart Card reader and drivers).

In our system, Smart Card support has to be added to both the Java strong_login applet, and the web browser. The RSA private key is needed in

the applet for authenticating the user to the SESAME domain server and
thus obtaining the PAC, and is needed by the browser in every SSL/TLS
client authentication session. Several techniques can be used that allow
Smart Card support from both Java and a web browser.

4.1 Smart Card support in general

At the moment, several standards are being developed or have been devel-
oped that can be followed to provide Smart Card support.

4.1.1 PKCS#11

The PKCS#11 standard of RSA Laboratories defines an interface to a
cryptographic token. The standard is also called “Cryptoki” which stands
for Cryptographic Token Interface Standard. The last version (2.01) of this
standard appeared in December 1997.

The Cryptoki standard makes abstraction of the specific properties of
different tokens, but has to be situated under a generic API as the Internet
standard GSS-API or Microsoft’s SSPI. PKCS#11 is supported by
Netscape Communicator and by a number of Smart Card products.

4.1.2 PC/SC

The PC/SC Workgroup is a joint efort of Bull, Gemplus, Hewlett-
Packard, IBM, Microsoft, Schlumberger, Siemens Nixdorf, Sun Microsystems,
Toshiba and VeriFone. The group has developed a specification that
facilitates the interoperability necessary to allow Integrated Circuit Card
(ICC) technology, like Smart Cards, to be utilized in the PC environment.

The PC/SC standard could be used on any platform. However, it is cur-
rently only implemented on Windows platforms. As such, it is supported by
Microsoft Internet Explorer and will be part of the Windows 2000 operating
system.

4.1.3 OpenCard

The OpenCard Consortium is an industry-wide organization driving the cre-
ation of an open standard framework for interoperable Smart Card solutions
across multiple hardware and software platforms.

The OpenCard Framework (OCF) provides a general Java Smart
Card API. It provides a PKCS#11 as well as a PC/SC compliant service,
and can thus be used with numerous Smart Card systems.

4.1.4 JECF

Within Sun Microsystems’ Java Electronic Commerce Framework, Smart
Card support is provided by the Java Wallet. The API is not only suited
for the Java Wallet itself, but can also be used for any Java application. In addition the Java Smart Card API [?] provides an interface to PC/SC.

4.1.5 Smart Card for Windows

Compliant with PC/SC, Microsoft’s Smart Card for Windows [?] is expected to be available from early 1999. The existing dominance of Microsoft operating systems in the marketplace may well result in Smart Card for Windows becoming the most widely available platform for systems that we are considering for this project.

Smart Card for Windows is a small operating system that resides on the Smart Card itself, providing standardized methods of accessing the smart card data and functions, through the PC/SC specifications. This attempts to allow complete interoperability between the application, different operating systems, different card readers and different smart cards.

4.2 Requirements

During the implementation a lot of other issues appeared and had to be taken care of.

4.2.1 Browser and Java Applet support for Smart Cards

An important aspect in our solution is the fact that Smart Card support needs to be provided to both browser and Java applet. The browser’s session security and client authentication is used, and the Java applet has to perform the strong login protocol. Both browser and applet need to access the same cryptographic services and keys on the Smart Card.

4.2.2 Different Browsers

Both Netscape Communicator and Microsoft Internet Explorer can support Smart Cards. However they each use a different standard. While Netscape Communicator is compliant to PKCS#11, Microsoft Internet Explorer’s Smart Card support is based on Windows’ general smart card support provided by PC/SC.

4.2.3 Java Applet

The fact that the two browsers use a different standard, does not affect the way Smart Card support is added to a Java applet. It could use either the PKCS#11 standard or PC/SC standard.

©1999 Springer-Verlag
4.2.4 Smart Card Drivers

There are two problems at the Smart Card level. The first is that Smart Cards are not standardized for operations such as cryptographic functions. Basic operations like file reads and writes are standardized but the more advanced features are different for each card. Hence a driver is required for each different Smart Card. The second problem is that drivers are required for each operating system. They will need to be written to support Windows and UNIX.

Hence there is a multiplicative effect here. The number of solutions required equals the number of different operating systems multiplied by the number of different Smart Cards. In many cases the driver is provided by the manufacturer. As long as it complies to a standard Smart Card interface, there should be no problem.

4.2.5 Smart Card reader

The choice of reader is not so much a concern as they have been mostly standardized and are interchangeable. The Philips PE112 reader is used in our project.

4.2.6 Cookie

After obtaining a PAC through the strong login protocol, the PAC has to be stored as a cookie. There are two possibilities:

- If additional privileges are granted, it is possible for the applet to manually add the cookie by writing it to the cookie file. However, this would be a browser dependent solution.

- To add the cookie via the standard way, the cookie has to be sent by the server as part of an HTTPS message. Therefore, the applet first performs an HTTPS request including the PAC, to which the server responds with an HTTPS message containing the PAC as a cookie.

5 Related Work

A lot of related work has been done in the area of the integration with security architectures and the web and supporting Smart Cards.

In [?] additional TLS cipher suites are specified, which make it possible to integrate the Kerberos security architecture in the TLS protocol used for securing the web. Both TLS’s authentication and key exchange services are replaced by Kerberos’s services. Although this approach simplifies key management it does not provide the authorization and non-repudiation services of our approach.
The TrustedWeb system of SSE [?] integrates the commercial version of the SESAME security architecture into the web. Smart Cards are also supported, in particular the SNI Sicrypt (only for key storage) and the Swedish SEIS Card (also performs cryptographic operations) using the PC/SC standard. The difference with our approach is that SSE has decided to use the (complex) SESAME architecture for the whole process. They use proxies and these can be spoofed. Also the system is slow and does not provide non-repudiation.

The IntraVerse system of DASCOM [?] incorporates the DCE architecture in the web environment. The system inherits all the features and drawbacks of DCE. For a comparison of DCE and SESAME, we refer to [?].

6 Conclusion

Smart Cards were found to be an essential component in integrating a single sign-on system with web technology. They provide the secure link between the SESAME single sign-on ticket and the client-side authentication by SSL/TLS. Without the technology the ticket could be simply replayed. The Smart Card technology also allows a user to roam to any computer (within or outside the organization) as long as the client computer is fitted with the appropriate Smart Card library and Smart Card reader. The difficulty in building the system, is supporting the range of operating systems in use, and the different types of Smart Cards.