USING SESAME TO SECURE WEB BASED APPLICATIONS ON AN INTRANET

Paul Ashley
Information Security Research Center, School of Data Communications
Queensland University of Technology, GPO Box 2434, Brisbane - AUSTRALIA
ashley@fit.qut.edu.au

Mark Vandenwauver, Joris Claessens
ESAT/COSIC, Kardinaal Mercierlaan 94
3001 Heverlee - BELGIUM
mark.vandenwauver@esat.kuleuven.ac.be
joris.claessens@esat.kuleuven.ac.be

Abstract The use of web technology within organisational Intranets is increasing. The combination of a standardised interface and the security features provided by TLS have made web technology very attractive. The TLS technology however has some limitations, especially in its lack of access control functionality. This paper focusses on alternatives to provide improved security services to web based applications. The SESAME security architecture is shown to provide all of the TLS security services, with the addition of other services such as the access control service. Also because SESAME uses the connection based GSS-API which is the same paradigm used by TLS, it is shown to be a suitable replacement. Unfortunately because web servers and web browsers do not provide hooks for replacing the TLS security technology, SESAME is not easily used. Two alternatives are therefore considered that overcome this limitation: a new proposal before the IETF of extending TLS to carry attribute certificates and a hybrid solution built by the authors.

Keywords: TLS, access control, single sign-on, attribute certificates, SESAME

1. INTRODUCTION

Companies have begun using World Wide Web (web) technology on Intranets to provide services to their internal clients. These web sites are used inside organisations to streamline and in many cases trans-

B. Preneel (ed.), *Secure Information Networks*

form organisational data processing. The advantage of the system is that browsers are mostly *standardised* and are available on a range of operating systems. For example, users can be using Internet Explorer on Windows NT, or Netscape on Linux, and the user interface and operation is virtually the same.

A number of incidents (in particular security breaches on the Internet) have drawn the attention of the general public to the problem of data security. It is no overreaction to state that such incidents pose a genuine threat to companies and that adequate countermeasures have to be taken. Cryptographic techniques and computer security constitute an essential link in this global security concept.

This paper begins with a short description of SESAME V4 [SES] and TLS [DA99], two of the security techniques proposed in recent years. It highlights that the TLS services are in fact a subset of the SESAME services, and hence web based applications could be secured completely with SESAME, rather than using TLS which is the existing method.

A new method of integrating SESAME into web browsers and web servers is proposed using a GSS-API interface [Lin97]. The proposal notes that the paradigm used by TLS of context establishment with mutual authentication followed by data protection, is the same as that provided by the GSS-API, so a GSS-API interface would allow TLS substitution with SESAME.

Because this GSS-API interface is not currently provided, two actual solutions are compared: the TLS attribute certificate [Far98] and a new, hybrid solution proposed by the authors. Our solution involves the integration of TLS and SESAME V4: SESAME V4 is used for user authentication, non-repudiation, access control and auditing, and TLS is used for the end to end security in the traditional way. Smart Cards are identified as a method to allow user roaming and to increase security. The new integrated solution is also compared with alternative approaches.

2. SESAME AND TLS

SESAME (A Secure European System for Applications in a Multi-vendor Environment) [AV99] and TLS (Transport Layer Security) [DA99] are technologies that have been developed to provide security services to client-server systems. TLS is better known under the abbreviation of its predecessor SSL (Secure Sockets Layer). It has been a big commercial success and is currently available in modern web browsers (such as Netscape Communicator and Internet Explorer). SESAME is the

result of a European initiative and is a security architecture based on Kerberos [KN93].

In the following subsections we describe both technologies and conclude that although both can be used separately, a combination could offer even more flexibility.

2.1 SECURITY SERVICES PROVIDED BY TLS AND SESAME

Table 1 gives an overview of the different security services that are provided by SESAME V4 (from here on denoted as SESAME) and TLS. The table indicates that the services that TLS provides are in fact a subset of the services provided by SESAME.

It is, however, important to note that SESAME is a security architecture, whereas TLS is just a standard that defines in what way the communication between two parties should be secured. Both are therefore situated in different layers of the TCP/IP reference model [Com95]. TLS is situated in the transport later whereas SESAME is situated in the application layer. This difference clearly influences the kind of services that can be provided both technically and legally.

Table 1 Security Services

Security Service	SESAME	TLS
User Authentication	Yes	(Yes)
Entity Authentication	Yes	Yes
Access control	Yes	No
Data Confidentiality	Yes	Yes
Data Authentication	Yes	Yes
Non-repudiation of origin	Yes	No
Auditing	Yes	No

The first security service that is offered by SESAME is user authentication. That is, SESAME allows users of the SESAME secured network to log on once to the network, be provided with a SESAME access token (named PAC in the SESAME literature), and then use this token to access resources across the network. User authentication is also defined in TLS. However, because of the placement of TLS in the TCP/IP layer model, the client workstation rather than the user should be authenticated.

One could argue that in todays browsers, users are authenticated instead of the client workstations. This is indeed true and it has some advantages as well. However, an ideal TLS implementation should be application-independent, and should therefore be transparent for the user, making user authentication impossible.

In both technologies, entities are authenticated, and data is protected while in transit with options for both data confidentiality and data authentication protection.

One of the main features of SESAME is the provision of access control, and in particular role based access control (RBAC) [SCFY96]. TLS 1.0 (the latest release) does not implement any access control service. The IETF TLS working group is however well aware of this shortcoming and has proposed an Internet Draft [Far98] to solve this problem. We will discuss their approach in Section 4.

In SESAME, non-repudiation is realized by applying digital signatures to the exchanged messages. It is clear that non-repudiation (in the legal sense) can not be provided by protocols that are not situated in the application layer, such as TLS. This is mainly because these protocols are concerned with securing the data flow, and the cryptographic protection is removed when the data is received and the original unsecured information is passed to the application. Another problem is that TLS is transparent for the user, so the user is not necessarily aware of the fact that some kind of digital signature is being applied. Moreover, the current TLS release does not require the use of digital signatures in any messages except for the authentication of the client machine.

An extensible set of audit tools is provided by SESAME, whereas TLS does not define a way of auditing. A lot of web products do implement a separate logging mechanism. For example secure web servers can log specific TLS related interactions. On the other hand protection of these log files is as important as creating them, and this is most of the time not guaranteed.

2.2 LIMITATIONS WITH THE TLS SECURED WEB BASED APPLICATIONS

Although the TLS security for web based applications has been very successful, especially from a commercial point of view, a comparison with SESAME does highlight some deficiencies. The main deficiency is the lack of access control functionality. That is, although the web server can have some confidence in the identity of the user through client-side authentication, the web server does not know the user's privileges.

Access control has to be based on the user's identity and this is very hard to scale.

There are other concerns with TLS security. Because it does not provide auditing facilities, and non-repudiation, it has to be used in conjunction with another technology if the full range of services needs to be provided to an Intranet.

3. USING SESAME TO SECURE WEB BASED APPLICATIONS

The impressive feature of SESAME is that it provides a very wide range of security services. SESAME benefits as well from the fact that it can be used to provide a comprehensive Intranet security solution. For example users can be provided with a single sign-on to the network, applications can be secured to provide entity authentication, protection to data in transit, and importantly provide the user's privileges securely to the target server. Auditing is also provided and the whole system is scalable through the appropriate use of public-key protocols [VGV97].

Bringing web based applications under the SESAME umbrella would not only allow extra security services to be provided to a TLS based solution, but also these applications would be part of a single integrated security solution. There are hence enough benefits to propose SESAME as a alternative to TLS for web based applications.

Securing web based applications in theory would not be difficult using the SESAME GSS-API [AV99, BFP96]. TLS adopts the same security model as the GSS-API, establishing a security context with mutual authentication of client and server, and then securing transferred data within that context. There is unfortunately a strong reason why this may not be a practical solution at this stage. Current web browsers and web servers do not have the hooks to allow the TLS security to be replaced with another technology. For example if web browsers and web servers used the GSS-API as a security interface, then the user would be able to install whatever technology is appropriate into the web browser and web server, as long as it provided a GSS-API based library.

4. TLS EXTENSIONS FOR ATTRIBUTE CERTIFICATES

The IETF working group in charge of TLS last year adopted a first Internet Draft that would provide access control through the TLS primitives [Far98]. The idea originates at SSE (Secure Solutions Experts, one of the partners in the consortium that developed SESAME) and it is

thus no big surprise that this proposition more or less tries to integrate TLS and SESAME.

The Internet Draft proposes extensions to TLS in order to offer an access control primitive. The scheme appears to borrow many ideas from the SESAME PAC structure [ECM96], so that it may be possible for TLS to transport SESAME PACs.

The TLS protocol would be modified so that a Server can obtain a Client's privileges by receiving an AttributeCertificate (AC) (previously a Server could only obtain a Client's identity). An access control decision can then be made at the Server based on the privileges inside the AC. The integrity of the AC can be verified at the Server.

The work on Attribute Certificates has now been incorporated in the PKIX working group [FH99]. Informal contacts with the authors of the Internet Draft at the time of writing this paper, however indicate that they plan to resubmit their work to the TLS working group as well.

4.1 ATTRIBUTECERTIFICATES

Table 2 Attribute Certificate Fields

Issuer	The entity who produced (and signed) the AC.
Serial Number	Unique number identifying the AC.
Validity	Gives the period during which the AC is valid.
Audit ID	Used for auditing purposes.
Owner	Optional field, names the entity the attributes are associated with.
Attributes	Contains the actual set of attributes.
Targets	Optional Field, used to specify target application the AC is valid for.
Algorithm	The signing algorithm.
Signature	Contains the digital signature of the AC issuer.

The AC is a structure based on an X.509 certificate [ITU93] (see Table 2). The AC contains access control information such as group membership, role information, clearance information and other information appropriate to an organisation. The Internet Draft lists a number of requirements that must be supported by the AC. The reader will notice a very close resemblance to the SESAME PAC, described in [AV99, ECM96]. The requirements placed on these ACs are listed in Table 3.

Table 3 Attribute Certificate Requirements

1	Support for short-lived ACs (hours).
2	Support for fixed periods of operations of the AC.
3	Support for a standard set of attribute types.
4	Issuers of ACs should be able to add additional attribute types.
5	The AC should be targettable to (a) particular server(s).
6	The AC should be delegatable to a server.
7	Delegation should be controllable.
8	Delegation should support a chain of delegation.
9	Some of the AC attributes should be able to be encrypted.
10	ACs should be able to be *pushed* or *pulled*.
11	Attribute types should be able to be connected to a particular source.
12	ACs should support anonymity of the owner.
13	ACs should support audit identities.
14	ACs should support charging identities.

There are a number of standard defined attributes: *Audit Identity*, *Access Identity*, *Charging Identity*, *Group*, *Role* and *Clearance*. Each of these can be associated with a policy authority so that for example a role can be associated with a particular organisation.

4.2 AC ACQUISITION PROTOCOLS

The possible AC exchanges are shown in Figure 1.

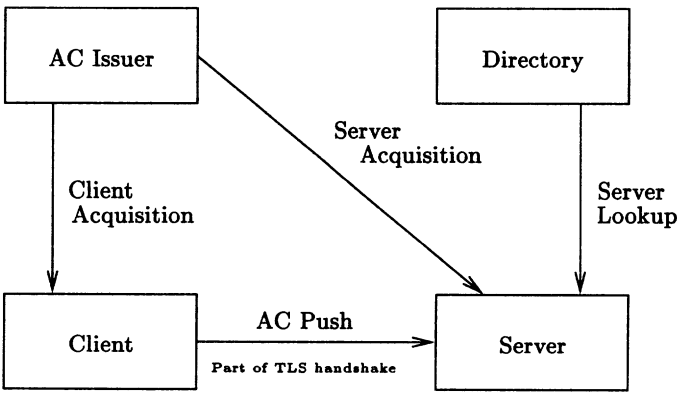

Figure 1 AC Acquistion Protocols

4.2.1 Client Acquisition. The Client uses TLS in the normal way to establish a connection with an AC Issuer, and is passed the AC as payload data of the returning TLS datagram. The Client sends a *ClientACRequest* message, and the server returns the AC in a *ClientACResponse* message. The *Client Acquisition* phase may occur when the Client is started, or when the Client receives *ACRequest* message from a Server it is trying to connect to.

4.2.2 Server Acquisition. Server acquisition occurs when a client has established a TLS connection to the server, but has not provided an AC. The server sends a *ServerACRequest* message to the AC Issuer and receives a *ServerACResponse*. Again TLS is used in the normal way to establish a connection between the Server and AC Issuer.

4.2.3 Server Lookup. The server uses a standard directory look-up protocol (e.g., LDAP [YHK93]) to find the user's AC. This part is currently outside of the scope of the Internet Draft.

4.2.4 AC Push. In this exchange the server requests that the client presents its AC (through an *ACRequest* message) so that an access control decision can be made. This exchange is suitable for inclusion within the TLS handshake.

The approach is to handle the access control structure in the same way as X.509 certificates are handled. This means the Server sends an *ACRequest* message to the Client, which responds with an *ACInfo* message. This exchange is very similar to the *CertificateRequest* and *CertificateResponse* messages that are already part of the protocol.

There is also one change required to the TLS Alerts, so that a *access_denied (49)* message can now be non-fatal, as a Client may have access to another AC which it could present.

5. INTEGRATION OF SESAME AND TLS

The solution described in the previous section unfortunately suffers from some deficiencies. From a legal point of view, it is hard to justify that important decisions such as who gets access to what, are being taken at a level transparent to the end user. Another problem is the lack of available implementations of this enhanced TLS.

As an alternate solution, we describe how SESAME and TLS today can be integrated to bring web based applications under the umbrella of the SESAME Intranet security solution. Our solution does not require any changes to the web browser so that users can keep on using their

familiar browser. In order to achieve this we use the Java technology [Sunb].

An additional feature of the proposed solution is that we can consider the roaming of users in the Intranet. Modern Intranets often span the globe and it is important for business people away from their home base to be able to securely access their resources. Our solution can be enhanced with smart cards to allow the user to roam throughout the Intranet, and work on any smart card enabled workstation.

5.1 SERVICES PROVIDED BY THE HYBRID SOLUTION

Table 4 shows the security services that could be provided by SESAME and TLS if they were integrated as they are in our solution. User authentication is provided by SESAME (to be able to obtain a PAC) and by TLS (to be able to prove being the owner of the PAC). Entity authentication and data protection services are provided by TLS in the traditional way. Access control, non-repudiation, and auditing are provided by SESAME.

Table 4 Security Services Provided by the Hybrid Solution

Security Service	SESAME	TLS
User Authentication	x	x
Entity Authentication		x
Access control	x	
Data Confidentiality		x
Data Authentication		x
Non-repudiation of origin	x	
Auditing	x	

5.2 WEB BROWSER BASED SYSTEM

This section outlines a web browser system in which the services of both TLS and SESAME are combined.

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

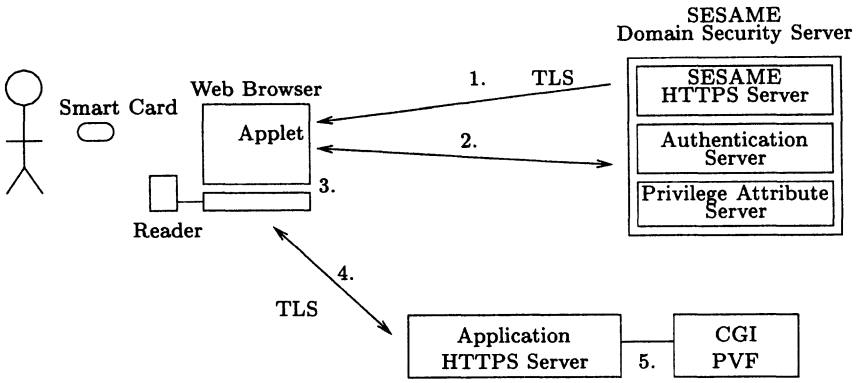

Figure 2 Integration of Smart Cards into Web Browser

5.2.1 Step 1: Downloading the *login* applet. The user/client contacts the TLS secured SESAME web server via a standard web browser. The web server is as such an additional component to the SESAME security architecture. In our implementation we have put the TLS server as an extra component on the SESAME domain security server.

The TLS server sends a *login* applet in a secure way to the client. TLS is thus used for authenticating the 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 [Suna]).

5.2.2 Step 2: Obtaining a PAC. The Java applet is used to perform SESAME's *login* protocol. This means that users provide their name and X.509 certificate. SESAME's authentication is based on public-key technology. It is similar to the Kerberos PKINIT initiative [TNW+98] but differs in some details [VGV97]. In our current implementation the user's private key is stored on the local filesystem encrypted with a passphrase.

If authentication is successful, the user receives a PAC. This PAC contains the user's privileges (role). It is valid for a limited time period only, and is digitally signed by the issuer (so it cannot be altered).

The PAC also contains the unique identifier of the X.509 certificate (XID) that was used to authenticate the user.

5.2.3 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 [Net, KM97], so that it is later on sent to any web server in

the SESAME domain. Storage is automatically done by the browser after receiving an http message containing the appropriate http header (namely the cookie) from the web server.

5.2.4 Step 4: Performing RBAC. When the client sends a request to a TLS (in our solution client authentication is mandatory and not optional) 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 TLS's client authentication data to the information inside the PAC. It verifies whether the unique identifier of the X.509 certificate used by the client for authentication purposes is equal to the XID value inside the PAC. This is so because both TLS and SESAME use the same key pair and X.509 certificate in our approach. In SESAME terminology the CGI program implements the function of the PAC Validation Facility (PVF).

Note that the CGI approach is only one possibility. For the Apache web server, it is for example also possible to implement the PVF functionality using an Apache module.

5.2.5 Step 5: Delivering the Information. If the client is authorised 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. This can be implemented transparently on most web servers such as the Apache version we used. Moreover, the information should not be directly accessible anyway.

5.3 DISCUSSION AND POSSIBLE IMPROVEMENTS

As pointed out in Section 5.2.2 the storage of the user's private key on a standard filesystem inherits some disadvantages of the traditional username/password paradigm. The most important one being the fact that it could be attacked off-line (trying out a set of plausible passphrases). Therefore we suggest to replace this by using a smart card. Not only would this prevent the aformentioned attack, it would also allow the users to roam as all they need to securely access their web based applications would be a web browser and a smart card reader.

The PACs are transferred as cookies. In some organisations, this might be inappropriate especially if the organisation's security policy does not allow automatic acceptance of cookies. However, similar solutions have been proposed by companies themselves (see Section on Related Systems).

When users want to change role, they have to use the applet again to obtain a new PAC. The applet can remain in a separate window for this purpose.

Our system is not meant to improve the security of existing browsers and servers. On the contrary, it still relies on their security, and additional security services are provided on top of this. Most of the available web browsers are unable to use strong cryptography, because of U.S. export restrictions. The level of security that can be obtained is therefore rather poor. This problem can be solved in different ways. A first possibility is to use software such as Fortify [McK] to upgrade the browser. Unfortunately this can only be applied to the Netscape Communicator. A second possibility is to apply for a Global Server ID [Ver] for the web servers as then the browsers would use strong cryptography. Global Server IDs can however not be obtained by all organisations. Other mechanisms exist to provide strong cryptography to browsers (e.g., proxy based systems). A full description of all strong crypto enabling solutions is out of the scope of this paper.

The *login* applet could also include a TLS implementation, which would be used after the login into SESAME. As in the first exchange we only rely on the data authentication service of TLS (which is not hindered by export restrictions).

6. RELATED SYSTEMS

TrustedWeb [SSE] is a product based on the SESAME technology. The product is developed by SSE (Secure Solutions Experts), a wholly owned subsidiary of Siemens. Instead of relying on TLS, SESAME is also used to secure the communication between browsers and servers. Proxies are used as an interface to browsers and servers, so that they do not have to be modified. One of the problems of using SESAME via proxies is that the individual web based applications are going to be slower. Instead of web browser and web server communicating directly, each communication has to pass through two proxies before reaching the other party. Even more important if the connection between client and proxy or proxy and server is not adequately secured, this link will be the target of a possible attacker. It is for example not too difficult to spoof

a proxy and obtain in this manner information that could be (re)used later on to gain illegitimate access.

Dascom's Webseal [DAS99] product is based on the DCE security architecture [RKF92] and thus implements the authorization scheme using symmetric key technology. In this solution a separate channel is set up using the GSS-API, to transfer the authorization information from the client to the server. Both clients and servers thus need to be adapted. Webseal's advantages include the possibility for a finely granulated access control system and the performance boost due to the use of symmetric key systems. Its main negative is the lack of non-repudiation.

The Secure HyperText Transfer Protocol (S-HTTP) [RS96] was an alternative proposal for securing the World Wide Web. In contrast to TLS, S-HTTP is situated in the application layer, and therefore provides non-repudiation. However, S-HTTP does not provide authorization, the same as TLS. SSL became the de-facto standard for securing the WWW, while S-HTTP is hardly used.

WDAI, a simple World Wide Web distributed authorization infrastructure [Kah99], is a proposal in which the authorization information is put in X.509v3 certificates that are stored in the browser's normal database. Each time a user has to logon to the system, a public key pair is generated, and a new certificate request is sent to the Authorization Server. This server issues a certificate with limited validity period. SSL with client authentication is used to connect to a normal server. This server performs access control based on the authorization information in the client certificate. In this system, existing browsers do not have to be modified, and no additional software is needed at the client side. However, having to generate a new key pair and certificate at each new logon (and having to manually delete them from the browser's database after each session), is certainly a disadvantage.

Similar cookie systems exist (e.g., [Sam99]). The security of all these proposed, and sometimes employed, solutions, is however not always guaranteed. Sometimes the cookies are not even protected by TLS, but even if they are protected, an attacker could still be able to steal the cookie from the user's computer and use it to get authorised. In our system, the cookie only contains public information. Stealing a cookie is allowed, but since the cookie is linked to the client authentication that is always performed, an attacker is not able to successfully use it.

7. CONCLUSIONS

Intranet Web based applications could be secured with SESAME instead of TLS, because SESAME provides all of the services of TLS. This

scheme also has the advantage of bringing Web based applications into the Intranet security solution provided by SESAME.

The problem with using SESAME to secure web based applications is that currently web browsers and web servers do not provide standardised hooks for interfacing new security technology. A standard hook that provides a GSS-API interface for example, could allow the TLS security to be replaced quickly with SESAME.

As an ad-hoc solution, SESAME and TLS could be integrated, and then web based applications would be part of the SESAME Intranet security solution. With the addition of smart cards to the system, users also have the extra bonus of being able to roam throughout the Intranet.

In the long term the approach of the TLS attribute certificate could become very successful because of the widespread availability of TLS implementations through standard browsers.

References

[AV99] P. Ashley and M. Vandenwauver. *Practical Intranet Security: An Overview of the State of the Art and Available Technologies.* Kluwer Academic Publishers, 1999.

[BFP96] E. Baize, S. Farrell, and T. Parker. The SESAME GSS-API Mechanism, November 1996. Internet Draft (expired).

[Com95] D. Comer. *Internetworking with TCP/IP*, volume 1. Prentice Hall, Inc., 3rd edition, 1995.

[DA99] T. Dierks and C. Allen. The TLS Protocol Version 1.0, January 1999. RFC2246.

[DAS99] DASCOM. The Webseal Home Page, 1999. Available at http://www.dascom.com/prod/webseal/index.html.

[ECM96] ECMA 219. ECMA-219 Security in Open Systems - Authentication and Privilege Attribute Security Application with Related Key Distribution Functionality, 2nd Edition, March 1996. European Computer Manufacturers Association.

[Far98] S. Farrell. TLS Extensions for AttributeCertificate Based Authorization, August 1998. Internet Draft.

[FH99] S. Farrell and R. Housley. An Internet AttributeCertificate Profile for Authorization, April 1999. Internet Draft.

[ITU93] ITU. ITU-T Rec. X.509 (revised). The Directory - Authentication Framework, 1993. International Telecommunication Union, Geneva, Switzerland.

[Kah99] J. Kahan. WDAI: A Simple World Wide Web Distributed Authorization Infrastructure. In *Proceedings of 8th Internation World Wide Web Conference*, pages 521–531, 1999.

[KM97] D. Kristol and L. Montulli. HTTP State Management Mechanism, February 1997. RFC2109.

[KN93] J. Kohl and C. Neuman. The Kerberos Network Authentication Service V5, September 1993. RFC1510.

[Lin97] J. Linn. Generic Security Service Application Program Interface Version 2, January 1997. RFC2078.

[McK] F. McKay. Fortify for Netscape. Available at http://www.fortify.net.

[Net] Netscape. Persistent Client State HTTP Cookies (Preliminary Specification), http://home.netscape.com/newsref/std/cookie_spec.html.

[RS96] E. Rescorla and A. Schiffman. The Secure Hypertext Transfer Protocol, May 1996. Internet Draft (expired).

[RKF92] W. Rosenberry, D. Kenney, and G. Fisher. *Understanding DCE.* O'Reilly & Associates, Inc., 1992.

[Sam99] Vipin Samer. Single Sign On Using Cookies for Web Applications. In *Proceedings of the 8-th Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises.* IEEE Computer Society, 1999.

[SCFY96] R. Sandhu, E.J. Coyne, H.L. Feinstein, and C.E. Youman. Role-Based Access Control Models. *IEEE Computer*, pages 38–47, February 1996.

[SES] SESAME. The SESAME Home, https://www.cosic.esat. kuleuven.ac.be/sesame.

[SSE] SSE. TrustedWeb. Available at http://www.sse.ie/ trustedweb/.

[Suna] Sun. Java Code Signing. Available at http://java.sun.com/security/codesign/index.html.

[Sunb] Sun. The Source for Java Technology. Available at http://java.sun.com.

[TNW+98] B. Tung, C. Neuman, J. Wray, A. Medvinsky, M. Hur, S. Medvinsky, and J. Trostle. Public Key Cryptography for Initial Authentication in Kerberos, December 1998. Internet Draft.

[Ver] Verisign. Verisign Global Server ID. Available at http://www.verisign.com/globalserver/.

[VGV97] M. Vandenwauver, R. Govaerts, and J. Vandewalle. Security of Client-Server Systems. In J. Eloff and R. von Solms, editors, *Information Security*, pages 39–54, 1997.

[YHK93] W. Yeong, T. Howes, and S. Kille. X.500 Lightweight Directory Access Protocol, July 1993. RFC1487.