Smart Certificates: Extending X.509 for Secure Attribute Services on the Web

Joon S. Park and Ravi Sandhu
The Laboratory for Information Security Technology
Information and Software Engineering Department
George Mason University
{jpark, sandhu}@list.gmu.edu

ABSTRACT: An attribute is a particular property of an entity, such as a role, access identity, group, or clearance. If attributes are provided integrity, authentication, and confidentiality, Web servers can then trust these secure attributes and use them for many purposes, such as access control, authorization, authentication, and electronic transactions. In this paper, we present a comprehensive approach to secure attribute services on the Web. We identify the user-pull and server-pull models and analyze their advantages and disadvantages. To support these models on the Web, we extend X.509 certificates, which are already in widespread current use. We name these extended X.509 certificates *smart certificates*. Smart certificates have several sophisticated features: they support short-lived lifetime and multiple CAs, contain attributes, provide postdated and renewable certificates, and provide confidentiality. This paper also discusses possible applications of smart certificates on the Web.

KEYWORDS: X.509, Certificates, Attributes, WWW Security

1 Introduction

The World-Wide-Web (WWW) is a critical enabling technology for electronic commerce and business on the Internet. Its underlying protocol, HTTP (HyperText Transfer Protocol), has been widely used to support synthesis of technologies and composition of different constituents for great effect in Web environments. WWW is commonplace. Increased integration of Web, operating system, and database system technologies will lead to continued reliance on Web technology for enterprise computing. However, there have been, at best, rather feeble attempts to provide effective access control on Web-based environments. Current approaches to access control on Web servers are mostly based on individual users; therefore, they do not scale to enterprise-wide systems.

An attribute is a particular property of an entity, such as a role [San98], access identity, group, or clearance. If the attributes of individual users are provided securely on the Web by security services - such as authentication, integrity, and confidentiality - we can use those attributes for many purposes, including attribute-based access control [PSG99, PS99], authorization, authentication, and electronic transactions. A successful marriage of the Web and secure attribute services has potential for considerable impact on and deployment of effective enterprise-wide security in large-scale systems.

Our goal is to build upon the mature technology of X.509 certificates and extend them for secure attribute services on the Web. The basic purpose of X.509 certificates is simply the binding of users to keys. Even though X.509 has the ability to be extended, the application of the extensions of X.509 for secure attributes has yet not been precisely defined.

In this paper, we identify the user-pull and server-pull models, in which each model has user-based and host-based modes, for a comprehensive approach to secure attribute services on the Web. We will descibe how to make the *smart certificates* by extending X.509v3 [HFPS98] - which is an ISO (International Organization for Standardization) and IETF (International Engineering Task Force) standard - since public-key infrastructure (PKI) has been recognized as a crucial enabling technology for security in large-scale networks. Furthermore, we will discuss possible applications of smart certificates for electronic commerce and business later in this paper.

2 Related Technologies

2.1 Secure Socket Layer (SSL)

SSL (Secure Socket Layer [WS96]) was introduced with the Netscape Navigator browser in 1994, and rapidly became the predominant security protocol on the Web. Since the protocol operates at the transport layer, any program that uses TCP (Transmission Control Protocol) is ready to use SSL connections. The SSL protocol provides a secure means for establishing an encrypted communication between Web servers and browsers.[1] SSL also supports the authentication service between Web servers and browsers.

SSL uses X.509 certificates. Server certificates provide a way for users to authenticate the identity of a Web server. The Web browser uses the server's public key to negotiate a secure TCP connection with the Web server. Optionally, the Web server can authenticate users by verifying the contents of their client certificates.

Even though SSL provides secure communications between Web servers and browsers on the Web, it cannot protect against end-system threats. For instance, if a user receives attributes from the server over a secure channel, it does not mean that we can trust the user. In other words, once the user, let's say Alice, receives some attributes from the server over the secure channel, she is able to change the attributes or give them to other people, because SSL does not support the integrity service in the user's end system. Then, Alice (or the person impersonating Alice) can access the servers - which accept the attributes - using those forged attributes. However, as we will see later in this paper, SSL can be used as part of our solution to protect information on the Web.

2.2 Public-Key Certificate (X.509)

A public-key certificate is digitally signed by a certificate authority (a person or entity) to confirm that the identity or other information in the certificate belongs to the holder

[1] In many cases, due to export restrictions from the United States only weak keys (40 bits) are supported, but SSL technology is intrinsically capable of very strong protection against network threats.

(subject) of the corresponding private key. If a message-sender wishes to use public-key technology for encrypting a message for a recipient, the sender needs a copy of the public key of the recipient. On the other hand, when a party wishes to verify a digital signature generated by another party, the verifying party needs a copy of the public key of the signing party. Both the encrypting message-sender and the digital signature-verifier use the public keys of other parties. Confidentiality, which keeps the value of a public key secret, is not important to the service. However, integrity is critical to the service, as it assures public-key users that the public key used is the correct public key for the other party. For instance, if an attacker is able to substitute his or her public key for the valid one, encrypted messages can be disclosed to the attacker and a digital signature can be forged by the attacker.

ITU (International Telecommunication Union) and ISO (International Organization for Standardization) published the X.509 standard in 1988 [ITU93], which has been adopted by IETF (International Engineering Task Force). X.509 is the most widely used data format for public-key certificates today and it is based on the use of designated certificate authorities (CAs) that verify that the entity is the holder of a certain public-key by signing public-key certificates. An X.509 certificate has been used to bind a public-key to a particular individual or entity, and it is digitally signed by the issuer of the certificate (certificate authority) that has confirmed the binding between the public-key and the holder (subject) of the certificate. An X.509 certificate consists of the following:

- version of certificate format
- certificate serial number
- subject's X.500 name (assigned by a naming authority)
- subject's public key and algorithm information
- validity period (beginning and end date)
- issuer's X.500 name (certificate authority)
- optional fields to provide unique identifiers for subject and issuer (Version 2)
- extensions (Version 3)
- digital signature of the certificate authority

The optional fields are available from Version 2 to make the subject name or the issuing certificate authority name unambiguous in the event the same name has been reassigned to different entities through time. Version 3 provides the extensions field for the incorporation of any number of additional fields into the certificate. These extensions make X.509v3 a truly open-ended standard with room to support diverse needs. It is possible for certificate issuers of interest to define their own extension types and use them to satisfy their own particular needs.

2.3 Attribute Certificate

The U.S. financial industry through the ANSI X9 committee developed attribute certificates [Far98b, Far98a], which have now been incorporated into both the ANSI X9.57 standard and X.509. An attribute certificate binds attribute information to the certificate's subject. Anyone can define and register attribute types and use them for his or her purposes. The certificate is digitally signed and issued by an attribute authority: furthermore, an attribute certificate is managed in the same way as an X.509 certificate. However, an attribute certificate does not contain a public key. Therefore, an attribute certificate needs to be used in conjunction with authentication services, such as another certificate (X.509) and SSL to verify the subject of the attributes.

3 Operational Architecture

We have different approaches for obtaining a user's attributes on the Web, especially with respect to user-pull and server-pull models, in which each model has user-based and host-based modes. Each approach can be made to work, and we will provide an analysis of their relative advantages and disadvantages. Basically, there are three components in both models: client, Web server, and attribute server. These components are already being used on the Web. Clients connect to Web servers via HTTP using browsers. The attribute server is maintained by an attribute authority and issues attributes for the users in the domain.

In this section, we will focus on identifying the operational models for secure attribute services on the Web with tradeoffs between them. To support these models on the Web, we will extend X.509 certificates - already in widespread current use - as described in Section 4.

3.1 User-Pull Architecture

In user-pull architecture, the user pulls appropriate attributes from the attribute server and then presents them to the Web servers, as depicted in a collaborational diagram in Figure 1. We call this a user-pull architecture, since the user pulls appropriate attributes from the attribute server, in which attributes are issued for the users in the domain. HTTP (HyperText Transfer Protocol) is used for user-server interaction with standard Web browsers and Web servers.

In user-pull-host-based mode, a user, Alice, needs to download her attributes from the attribute server and store them in her machine (which has her host-based authentication information, such as a client certificate[2].) Later, when Alice wants to access the Web server, which requires proper authentication information and attributes, her machine presents that information to the Web server. After client authentication and attribute verification, the Web server uses the attributes for its purposes, such as access control, authorization, and electronic transactions. However, since this mode is host-based, it cannot support high user mobility, although it may support more convenient service than the user-based mode - which requires user's cooperation (e.g., typing in passwords).

[2]Optionally, we can use other host-based authentication information, such as IP numbers and Kerberos tickets.

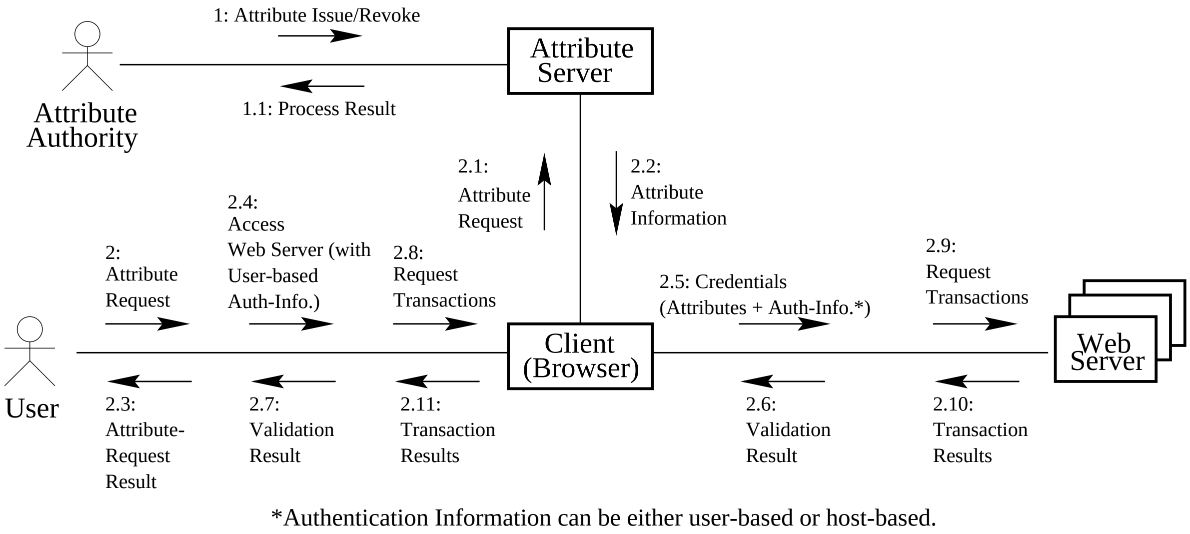

Figure 1: Collaborational Diagram for User-Pull Model

On the other hand, the user-pull-user-based mode supports high user mobility. A user, Alice, can download her attributes to her current machine from the attribute server. Then, Alice presents those attributes to the Web server along with her user-based authentication information, such as her passwords. After user authentication and attribute verification, the Web server uses the attributes for its purposes.

In this user-pull architecture, we must support the binding of attributes and identification for each user. For instance, if Alice presents Bob's attributes with her authentication information to the Web server, she must be rejected. There may be different mechanisms for binding attributes and user identifications. We will describe in Section 4 how to solve this problem efficiently by means of smart certificates between existing Web servers and Web browsers.

3.2 Server-Pull Architecture

In server-pull architecture, each Web server pulls appropriate attributes from the attribute server as needed and uses them for its purposes as depicted in a collaborational diagram in Figure 2. We call this a server-pull architecture, since the server pulls appropriate attributes from the attribute server. HTTP (HyperText Transfer Protocol) is used for user-server interaction with standard Web browsers and Web servers. If the attribute server provides attributes securely, the Web server can trust those attributes and uses them for its purposes.

In this architecture, a user, Alice, does not need access to her attributes. Instead, she needs only her authentication information. In server-pull-host-based mode, she presents host-based authentication information (e.g., her client certificate) to the Web server. Attribute obtaining mechanism is transparent to the user, while limiting the user portability.

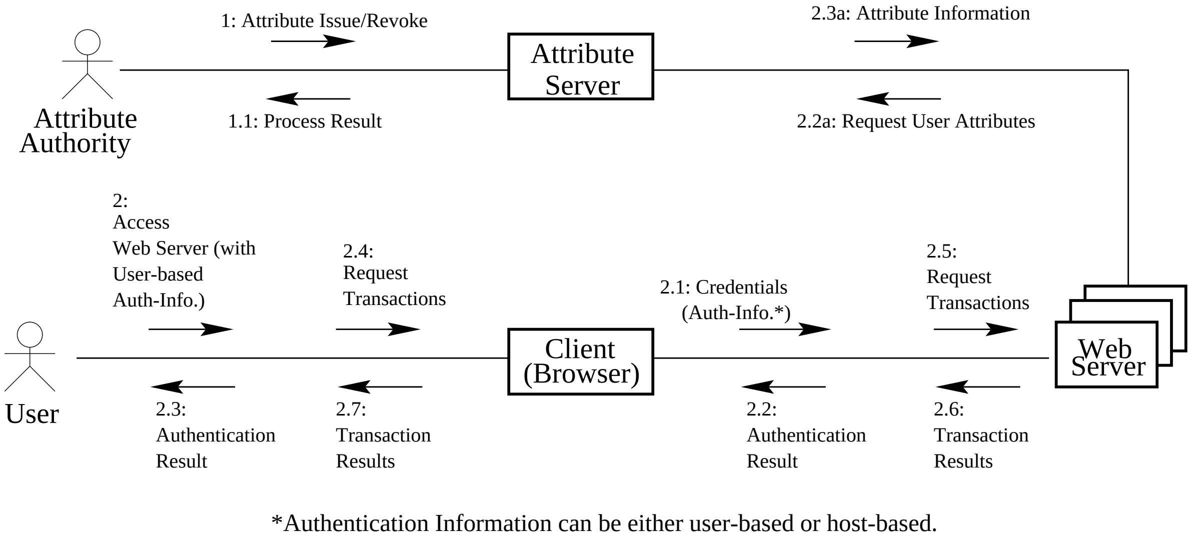

Figure 2: Collaborational Diagram for Server-Pull Model

However, in server-pull-user-based mode, Alice presents user-based authentication information, such as her passwords to the Web server. This supports high user portability, while it requires the user's cooperation (e.g., typing in passwords). After client(user) authentication, the Web server downloads the corresponding attributes from the attribute server and uses them for its purposes, such as access control, authorization, and electronic transactions.

4 Extending X.509 for Secure Attribute Services

Extension types of X.509 can be defined by anyone. Therefore, it is possible for certificate authorities to define and use their own extension types to satisfy their own particular needs. Each extension type needs to be registered by having an object identifier assigned to it.

It is strongly recommended that public-key pairs used for any purpose be updated periodically. This is an effective way to restrict cryptographic attacks, such as a key compromise. Furthermore, the lifetime of a public-key in a certificate may be different from that of other information in it. Namely, it is not a good solution to issue a currently existing certificate, such as X.509, that contain attribute information as well as public-key information. Even though the bundled certificate increases performance because of its simple mechanism, it cannot support effective certificate management. Adding, deleting, or changing attributes involves replacing and sometimes revoking X.509 certificates. This creates very large CRLs (Certificate Revocation Lists); therefore, it overburdens certificate management infrastructures. Furthermore, usually an organizational policy requires different authorities for maintaining attributes and public-keys. Since the current X.509 certificate cannot satisfy all the above requirements, we were motivated to design the *smart certificates*, described in the following subsections, to solve those problems. Note that a smart certificate

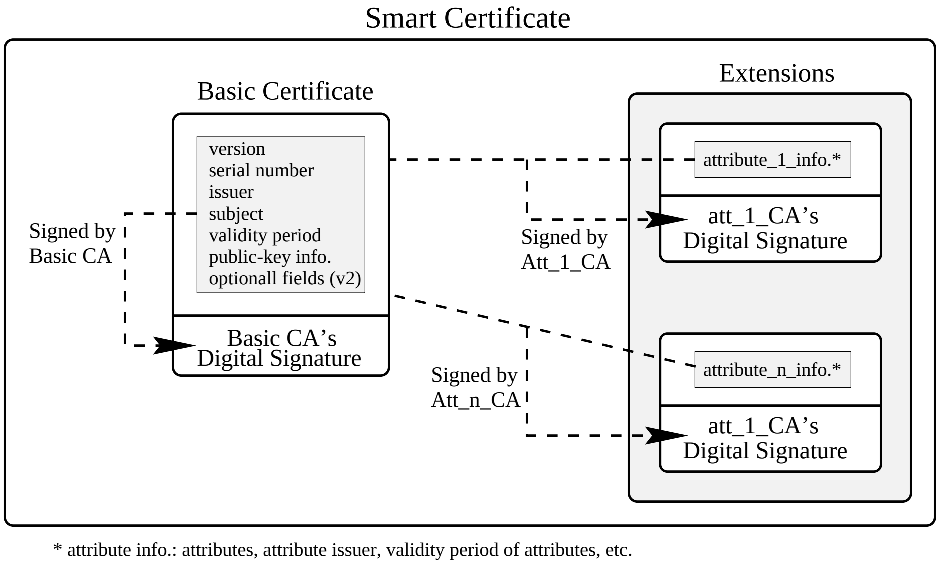

Figure 3: Attributes Signed by Multiple CAs in a Smart Certificate

is compatible with an X.509, since it keeps the same data format as an X.509.

4.1 Smart Certificates

Smart certificates extend X.509 certificates by adding the following features. Selection of these new features depends on the applications or policies.

4.1.1 Support for Short-Lived Certificates

Typical validity periods for X.509 certificates are several months or even years. To support user mobility, users should be able to download both the certificate and the private key to the software in different environments. This service may leave copies of private keys behind; therefore, the longer-lived certificates have a higher probability of being attacked. If, however, the validity periods for certificates are measured in hours, the user portability can be provided securely, since the copies of the corresponding private key expire automatically. Additionally, we do not need a revocation scheme (CRL) for the certificates, which is responsible for the complexity and cost of the public key infrastructure. The detailed description of short-lived certificates is available in [HS98].

4.1.2 Containing Attributes

If we use the extension fields in an X.509 certificate effectively, as depicted in Figure 3, we can separate the authority for attribute-issuing from the one for public-key-issuing. In other words, after a public-key authority (basic CA) issues an X.509 basic certificate for a user, Alice, as usual, an attribute authority (for instance, Att_1_CA) adds attributes for Alice to an extension field of the basic certificate (which contains public-key information). Consequently, the attribute authority (Att_1_CA) signs on the basic certificate and the attributes he added, and puts the signature to another extension field in the basic certificate. This can happen multiple times on a basic certificate by different attribute authorities. Later, the identity verification should precede the attribute verification. For instance, another party, say Bob, verifies Alice's identity first by the basic CA's signature in the smart certificate. If the authentication is successful, Bob verifies Alice's attributes by the corresponding attribute authority's signature in the extension field. If the attributes are valid, then Bob uses those attributes for his purposes. The contents of the attribute information in a smart certificate depend on applications.

The public-key and the attributes can be maintained independently. For instance, even though Alice's attributes issued by her school-attribute authority are expired (revoked) in the certificate, the rest of the attributes, such as attributes issued by her company-attribute authority, and public-key information in her basic certificate, are still valid. Each attribute authority has independent control over the attributes he issued. For example, the school-attribute authority for Alice can change, revoke, or re-issue the school attributes in Alice's certificate. Intuitively, if her basic certificate is expired (revoked), then all the attributes are meaningless. Even though a smart certificate can support independent management for the public key information and attributes, if there is one authority who has controls both sets of information, the system management becomes simpler.

Furthermore, if we use the short-lived certificate mechanism, we do not need to consider about revoking the bundled certificates. As a result, a smart certificate supports high performance by bundling public-key information and attributes in a certificate without causing overhead to certificate management infrastructures.

4.1.3 Support for Postdated and Renewable Certificates

Postdated certificates are used to run a job at some time in the future. Suppose a security officer wants to issue a certificate for a user starting a week from now and valid for 10 hours of use. He may issue a certificate with an expiration time of one week plus 10 hours from the present time. This is not a proper solution, since the certificate would be valid from the time it was issued until it expires. However, if we allow a certificate to become valid at some point in the future, we can satisfy the requirement.

Furthermore, if Alice needs a long-lived certificate, say, lasting for one year, it is more secure and efficient to issue a certificate that will be valid for that full 12 months, only if Alice keeps renewing it for a much shorter period, say, every day. To support this, we need to set the time (in the extension field), which specifies the certificate cannot be renewed. Since the certificate is renewed every day, we do not need CRLs. If there is some reason to revoke Alice's certificate, the CA does not renew the certificates.

These concepts are adopted from postdated and renewable tickets of Kerberos [SNS88, Neu94].

4.1.4 Encrypting Sensitive Information in Certificates

The smart certificates will support the encryption of some, or all, attributes, such as passwords, roles, or credit card numbers. Such an encrypted attribute in the certificate can be decrypted by an appropriate server using the corresponding key (the server's shared secret key or its private key).

4.2 Discussion

In this subsection, we want to compare smart certificates with existing certificates, such as X.509 and attribute certificates, in terms of the certificate life-time and authentication of the owner of the attributes.

Usually, an X.509 certificate has a long life-time, which requires an additional revocation mechanism (e.g., CRLs). Therefore, it has a relatively higher probability of being attacked, since the corresponding private key can be left in a system without the owner's knowledge, especially if the private key is stored in multiple machines. On the contrary, when we use a smart certificate, the short life-time eliminates the additional revocation mechanism (e.g., CRL for X.509), and makes the system more secure, since the remaining private keys expire shortly and automatically.

Currently, existing attribute certificates refer to another type of basic certificates, such as X.509, for authentication service. This mechanism brings complexities for the protocol itself and for certificate administration. For instance, an attribute certificate and the corresponding X.509 are issued in different entities and managed separately. Even the revocation mechanism is separate. The idea was brought to support separate authorities for attributes and authentication services.

However, if we use a smart certificate, both the attributes and public-key information can be bundled in a single certificate. This provides simplicity for both the protocol itself and for certificate administration. When we need separate authorities for attributes and authentication services, each authority signs separately on the same basic certificate and corresponding extension field, which contains attribute information. This can happen multiple times on a basic certificate by different attribute authorities. Each attribute authority has independent control over the attributes he issued. Even though a smart certificate can support independent management for the public key information and attributes, if there is one authority who controls both sets of information, the system management becomes simpler.

5 Applications of Smart Certificates

In this section, we introduce some applications of smart certificates. Many other applications can be similarly configured. Selection of the new features of smart certificates depends on applications and a given situation. Using the bundled (attributes and identifications) certificates is a good solution for the user-pull model, since the model requires the binding of this information. On the other hand, it is not a good idea to use the bundled certificates for the server-pull model, since users do not need access to their attributes.

5.1 On-Duty Control

Suppose an employee, Alice, needs to receive a certificate at 9:00 A.M. every morning and use it until 5:00 P.M. Monday through Friday. This certificate contains her sensitive attributes, such as clearance or access control information, Current X.509 cannot satisfy the requirements. It does not support confidentiality service (by encryption) for some sensitive information in the certificate. If the validity period for the certificate is longer than that of Alice's on-duty hours, the privilege based on the certificate still remains even after her on-duty hours.

If we use the smart certificate, the above requirements can be securely satisfied. First of all, the sensitive information in the certificate is encrypted which provides the confidentiality service. By using the short-lived certificate feature, we can set the validity of the certificate only from 9:00 A.M. to 5:00 P.M. every day. Furthermore, the postdated-certificate feature allows the employee to receive a set of certificates on a certain day, for instance, five certificates for individual days (Monday through Friday). Alice can then use the corresponding certificate each day during her on-duty hours. In this case, Alice does not need to receive the certificate every morning, and does not have the privilege based on the certificates after her on-duty hours. Alternatively, if we make the certificate valid for a set period of duration, for instance, from 9:00 A.M. to 5:00 P.M., but not between 5:00 P.M. and 9:00 A.M., the employee needs only one certificate for a week. However, this approach has a higher probability of compromise and possibly requires using CRLs.

5.1.1 Attribute-Based Access Control

When a user, let's say Alice, using a Web browser contacts a Web server that has been configured to request smart certificates, which contain users' attributes, the browser is required to present Alice's smart certificate and prove that she is the rightful owner. After client authentication, the Web server makes access control decisions based on attributes [PSG99, PS99], such as roles [San98], clearance, and group membership, contained within the smart certificate itself.

5.1.2 Electronic Transactions

If a merchant site uses smart certificates (which contain customers' encrypted credit card numbers) customers do not need to key their credit card numbers in for every transaction. For more convenient service, the merchant can issue a smart certificate containing special tokens for customers, such as electronic coupons (which have the coupon's ID number and discount information). For instance, if Alice received an electronic coupon contained within her smart certificate, she can use it before the coupon's expiration date at the merchant site. In this case, the merchant site needs to keep a record of the coupon to protect replay usages of the same coupon.

5.1.3 Eliminating Single-Point Failure

Usually, a merchant site has a customer-information database maintained on a server. One of the disadvantages of this method is that if the server keeping customers' information is penetrated by an attacker, all the customers' information, such as credit card numbers, preferences, addresses, and other sensitive information in the server, are open to the attacker. Furthermore, if a domain has multiple servers with multiple customer-information databases, maintenance and synchronization of this information is burdensome. There are also significant privacy concerns about data stored by servers, since such data can easily be misused. Users may feel more comfortable with servers that pledge not to maintain such data.

Smart certificates can solve these problems especially in the electronic commerce field. If a merchant site issues and uses smart certificates, the site does not need to have a customer-information database unless the site needs to track customers' access histories, since each customer's sensitive information is distributed and stored securely in the customer's smart certificates. Using smart certificates provides a more secure environment by eliminating customer-information databases, which can cause a single-point failure. Furthermore, the merchant can reduce the cost for the maintenance of customer-information databases.

5.1.4 Replace X.509

Besides using the extension fields in an X.509 certificate, a smart certificate provides new features, including containing attributes, eliminating CRLs by short-lived certificates, providing multiple CAs, supporting postdated and renewable services, and encrypting sensitive information in the certificates. However, it is still compatible with X.509, since a smart certificate keeps the same data format as the X.509. For instance, as the original X.509 supports the Secure Socket Layer (SSL) protocol for secure communications between clients and servers, the smart certificates can also be used as server certificates and client certificates for SSL without modifying the protocol.

6 Conclusions

In this paper, we have identified two operational models for attribute services on the Web: user-pull model and server-pull model. To support these models, we have extended an existing digital certificate, X.509, with several new features. The extended certificates, *smart certificates*, provide short-lived lifetime, attributes, multiple CAs, postdated and renewable services, and confidentiality services in PKI. According to the requirements of applications, some of these new features can be selectively used in conjunction with currently existing technologies.

References

[Far98a] Stephen Farrell. *An Internet AttributeCertificate profile for Authorization*, August 1998. draft-ietf-tls-ac509prof-00.txt.

[Far98b] Stephen Farrell. *TLS extensions for AttributeCertificate based authorization*, February 1998. draft-ietf-tls-attr-cert-00.txt.

[HFPS98] R. Housley, W. Ford, W. Polk, and D. Solo. *Internet X.509 public key infrastructure certificate and CRL profile*, September 1998. draft-ietf-pkix-ipki-part1-11.txt.

[HS98] Yung-Kao Hsu and Stephen P. Seymour. An internet security framework based on short-lived certificates. *IEEE Internet Computing*, pages 73–79, March/April 1998.

[ITU93] ITU-T Recommendation X.509. *Information technology - Open systems Interconnection - The Directory: Authentication Framework*, 1993. ISO/IEC 9594-8:1993.

[Neu94] B. Clifford Neuman. Using Kerberos for authentication on computer networks. *IEEE Communications*, 32(9), 1994.

[PS99] Joon S. Park and Ravi Sandhu. RBAC on the web by smart certificates. In *Proceedings of 4th ACM Workshop on Role-Based Access Control.* ACM, Fairfax, VA, October 28-29 1999.

[PSG99] Joon S. Park, Ravi Sandhu, and SreeLatha Ghanta. RBAC on the Web by secure cookies. In *Proceedings of the IFIP WG11.3 Workshop on Database Security.* Chapman & Hall, July, 1999.

[San98] Ravi Sandhu. Role-based access control. *Advances in Computers*, 46, 1998.

[SNS88] J.F. Steiner, C. Neuman, and J.I. Schiller. Kerberos: An authentication service for open network systems. In *Proc. Winter USENIX Conference*, 1988.

[WS96] D. Wagner and B. Schneier. Analysis of the SSL 3.0 protocol. In *Proceedings of the Second UNIX Workshop on Electronic Commerce*, November 1996.