Protecting Information on the Web

ELISA BERTINO, ELENA PAGANI, GIAN PAOLO ROSSI, AND PIERANGELA SAMARATI

The World Wide Web is giving rise to a new type of distributed information system, one with an Internet (or intranet) communication platform, a broader client/server architecture, the entire population of Internet users as potential clients, and all available Web servers as potential servers. This new generation of global information systems has eliminated many technological barriers to free circulation of information, allowing for development of public utility services such as the ACM Digital Library [1]. Businesses are embracing intranet and Web technology for internal communication, integration of dispersed branches and points-of-sale, and provision of new customer services.

But public and private organizations are still hesitant to transmit private documentation via the Internet. Their apprehension is due to the insecurity of the communication channels and lack of rules regulating data access. The protection of stored data and data transmission is a critical concern—and an inevitable one given the openness, simplicity, and globality of the enabling technology.

Communication protection involves safeguarding data transmitted over the network, both in terms of its integrity (against improper modification) and its confidentiality (against unauthorized disclosure.) Integrity of the transmission requires the authentication of:

- The message, to ensure the transmitted data is unaltered;
- The recipient, to certify the message is received by the proper entity; and
- The sender, to assure the identity of the transmission originator.

Of course the same entity may play both sender and receiver roles at different times.

ELISA BERTINO (bertino@dsi.unimi.it) is a professor in the Dipartimento di Scienze dell'Informazione, Università degli Studi di Milano, Italy.
ELENA PAGANI (pagae@dsi.unimi.it) is an assistant professor in the Dipartimento di Scienze dell'Informazione, Università degli Studi di Milano, Italy.
GIAN PAOLO ROSSI (rossi@dsi.unimi.it) is an associate professor in the Dipartimento di Scienze dell'Informazione, Università degli Studi di Milano, Italy.
PIERANGELA SAMARATI (samarati@dsi.unimi.it) is an associate professor in the Dipartimento di Scienze dell'Informazione, Università degli Studi di Milano, Italy.

Also, Web communications may involve more than two parties when related data is scattered on different Web servers and accessed by navigating from one data subset to another.

Data access protection is generally governed by rules establishing accesses to be allowed or denied. The definition of these rules, which generally take the form of authorizations, requires the establishment of subjects to which access can be granted/denied, objects to which access can be granted/denied, and actions for which access can be granted/denied. Each data-access request is checked against specified authorizations and granted only if authorized. To ensure that authorizations are correctly evaluated, the identity of the subject requesting access must be verified. Each access request must be accompanied by a declaration and a proof of identity. The most common forms of identification and identity proofs are the login and password that users enter to sign into the system.

In the remainder of this article, we illustrate some issues regarding data access protection on the Web [4]. The peculiar characteristics of this environment require specific considerations for establishment, representation, and enforcement of access rules. We include a glossary of the relevant keywords and descriptions of some principal protocols implementing security services and access control for Web documents.

Expressing Protection Requirements

In a very basic form, authorizations can be seen as triples <s,o,a> stating that subject s can exercise access mode a on object o. A sign (+ or -) can also be associated with the authorizations. Positive authorizations have the semantics illustrated and state accesses to be permitted, while negative authorizations state accesses to be forbidden. Defining authorizations involves defining system subjects, objects, and access modes. Note that we use the term "subject" to refer to the authorization subjects. Also note that a distinction exists between the subjects with specified authorizations and the active subjects actually requiring access to objects. For instance, a user is an authorization subject whereas a process requiring access on behalf of a user is an active subject. Here we refer to the authorization subject as principal and to the active subject as client. A principal delegates a client to act on its behalf by communicating with the client via a trusted path. From the principal, the client receives the information needed to authenticate the principal to the (application) server to which the principal's request is addressed. To perform authorization checking, requests submitted by clients must be controlled with respect to the authorizations specified for the principal (authorization subject.) In our discussion, we consider this mapping to be enforced and focus our attention on authorization subjects.

The key aspect of data organized in a hypertext format is that all documents (Web pages) are connected to each other by means of links. Users can easily access related documents by navigating through such links, using a standard Web browser. Below the presentation level visible to users is a level where information necessary for document construction and presentation is defined. Two different types of objects can be stored at this lower level: nodes, or frames for the construction of a higher level document; and objects that store content to be included in a document. Nodes can have associated inclusion links pointing to objects, or navigation links pointing to other nodes. An inclusion link from node to object assures that object content is included in the corresponding higher level document. Inclusion links are solved in the presentation process and are not visible to users, while navigation links between nodes are translated into corresponding navigation links between higher level documents.

Nodes may contain inclusion or navigation links to objects and documents stored at different sites. This distribution may affect the overall performance of the access control system. According to the access granularity to be provided and the transparency to be ensured, authorizations could be referred to higher level documents or to base level nodes and objects. Users may be authorized to access an entire document, or only some parts of it. Moreover, users may be able to view all or part of the navigation links the document contains. Restricted accesses to inclusion links result in different document views for different users. Restricted accesses to navigation links yield different hypertext organizations of the same set of documents for different users (see Figure 1).

The access modes specified in the authorizations depend on the objects considered: level (higher or lower) at which authorizations are specified, and on granularity at which access control is to be applied. A finer granularity for access modes allows a finer division of accesses, which in turn distinguishes between operations that users can execute. Several access modes can be considered. A broad categorization allows us to distinguish three classes of access modes: browsing, authoring, and using [9]. Browsing access modes allow users to access documents (in-full or in-part) in read-only mode. Authoring access modes allow access to documents in write mode. Using access modes allow objects to be included in documents.

Approaches to authorization subjects differ in terms of requirements of the underlying authentication mechanism, the granularity of specification they consider, and the kind of protection requirements they enable. A very basic form of authorization subject is users' identifiers. Each user is assigned an identifier uniquely identifying him/her within the system and the authorizations of each user are specified with respect to his/her identity. Global identities can be supported either by including the specification of the local system where the user is registered, or through global names independent of the system where the user actually logs in.

Users are considered to be authorization subjects since requests are always submitted by a user. Using a different paradigm, authorizations could consider IP location address-

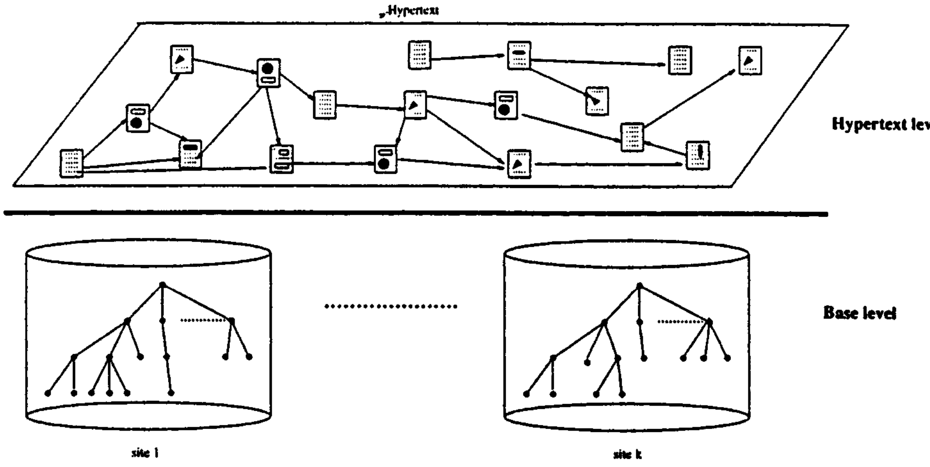


Figure 1. Base and hypertext levels [9].

es as subjects [9]. As authorization is defined in this case, accesses originating from the specified location are (or are not, in case of negative authorizations) allowed. In some cases the location can be partially specified, denoting patterns matching machines of the same subnetwork.

Authorizations specified for users or locations require servers to know who, or from where, accesses should be granted. In many cases, a fine-grained specification may not be needed. Consider the case of an association Web server storing information for members as well as the general public, like electronic issues of the association journal. In this case it is not necessary to specify users' authorizations. Authorizations instead can be specified for a group, with access control enforcement requiring that the association maintain membership information. System supporting groups generally support a basic group, such as public, to which basic accesses are granted to everyone.

Groups can also be extended, in cases where group information is not stored on the server. For instance, suppose that access to electronic issues of an association journal must also be granted to members of partner associations. Requiring the association to keep membership information about its partners would be impractical, so membership information must be provided by an external source.

Using a complementary paradigm, authorizations can also be granted for roles that users assume according to their needs [3]. Unlike group membership, role activation is dynamic; it allows a user to assume a role at a particular time. Role definitions are often derived from the infrastructure of the organization. In a bank, roles such as customer, employee, and branch manager can be defined. As with groups, a basic role with privileges that everyone can exercise can also be defined.

A more general approach to subjects is based on digital credentials [11]. Like credentials used in the paper world, electronic credentials represent an unforgeable and verifiable certificate of the requestor's qualifications. Credentials are supplied to a principal by a service provider administrator when the principal subscribes. Digital credentials describe the privileges possessed by the clients in a manner independent of the principal identity. When a principal requests access to a document, the server storing that document determines if the credential presented by the client is appropriate with respect to the requested access mode. Association members can access the online association journal by presenting their member identification numbers as digital credentials, for example. Credentials can be thought of as an evolution of distributed groups. Credentials are a powerful and flexible tool in the Web environment, where data access may be requested unpredictably from different domains, making traditional approaches difficult to apply.

Authorization Representation and Enforcement

Authorizations establish the accesses to be allowed or denied. To ensure that accesses are processed correctly, every request must be controlled against the specified authorizations. This authorization checking process is called access control. Access control requires a mechanism implementing the control policy against the specified authorizations. An important issue in the development of such a mechanism is the representation and storage of authorizations. Two basic techniques are generally used for managing authorizations: access control lists (ACLs) and capabilities.

Before discussing these techniques, we first sketch a reference system architecture. In this architecture, each document is accessed via a resource manager (RM). The applicable authorization policy is stated by an authorization manager (AM). Each document also has an authorization server (AUS), which enforces the access control for document requests according to the specification stated by the AM. Every access request to a document is mediated by the RM, which uses the AUS to determine whether to authorize the request. The AM, RM, and AUS can reside at the same or at a different server.

An ACL is a data structure associated with objects in the system. It contains all the principals that may access the object and the access modes they are allowed to exercise on it. In the ACL approach, every access request submitted by a client is passed by the RM to the AUS (Figure 2a). Note that the association between ACLs and corresponding objects is a logical association. Authorizations do not need to be stored together with the objects they refer to. Also, ACLs and objects may reside on different servers.

In the capability-based approach, authorizations are never under the control of the RM for the document, but are maintained at a separate AUS. When a client, on behalf of its principal, needs to access data, it issues a request to the AUS (Figure 2b), which verifies whether the principal has the appropriate authorizations. If so, the AUS generates a capability that the client should present to the RM. The capability provides evidence that the client may perform the requested operation. Note that the RM does not need to contact the AUS to verify the authorization, as it does with the ACL approach. The fact that a client has the capability proves that it is authorized to perform the operation.

The two approaches differ in terms of how authorizations are revoked. When an authorization for an access mode is revoked from a subject, the subject can no longer exercise that access mode on the object. In the case of ACLs it is easy to delete all authorizations on an object, by simply replacing the ACL with an empty one. However, revocation of all privileges of a subject requires the modification of all existing ACLs. Approaches similar to those used to propagate updates in distributed database systems can be used. Capabilities have the advantage of making repeated authentication of a subject unnecessary: the subject can exercise access until its capability expires. But in an environment like the Web, the estimation of a proper expiration time is difficult: this interval should allow users to complete their session without having to reload the documents because of the capability lifetime expiration. On the other hand, immediate

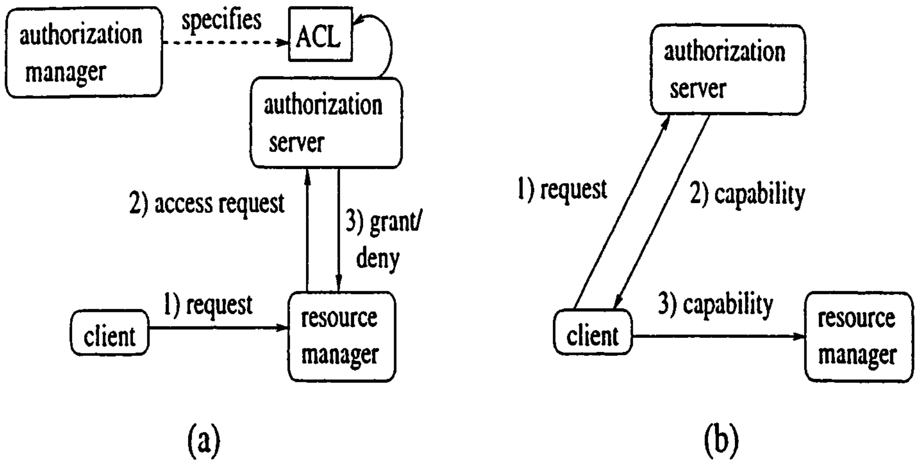


Figure 2. Authorization control scheme with (a) ACLs and (b) capabilities.

revocation requires maintenance of additional information (nonvalid capabilities) to be checked when access is requested.

Authorization Administration and Access Control

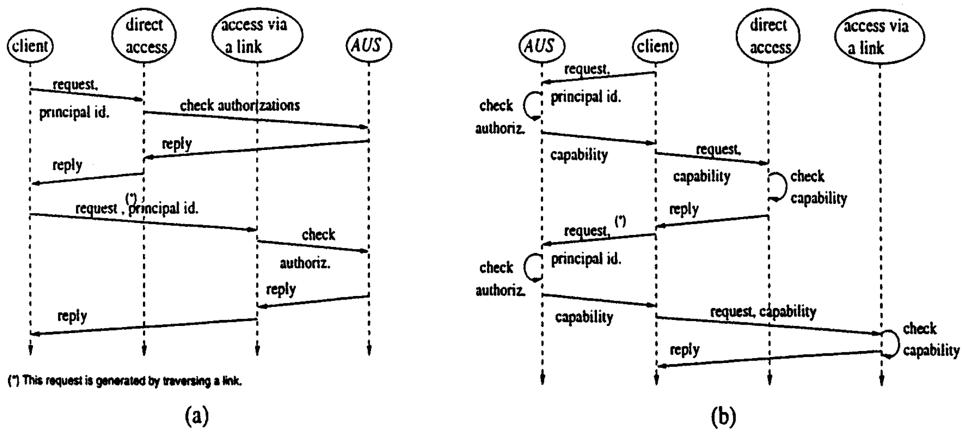
Authorization administration concerns the regulation and deletion of authorizations. The distributed and interlinked nature of the Web, as well as its number of potential clients and small granularity, make traditional administrative policies inappropriate. More flexible and efficient policies must be devised by considering the existing relationships between objects. Such policies can be based on high-level relationships existing between the documents (links at the hypertext level), relationships between the objects forming the documents at the base level (directory file relationship), or on other relationships [9]. By exploiting such relationships, it is possible to characterize groups of documents, which we call domains [9], and to manage authorizations on a per-domain basis. The consideration of domains in the specification of authorizations reduces the amount of authorizations to be specified and maintained and may increase efficiency (if access requests on documents within the same domain are checked only once, when the domain is entered.) The authorization administration tasks are delegated, at least partially, to an administrator of the domain.

If a domain can include only objects stored at the same site, authorizations on the domain can be maintained at the site where the domain is defined. Approaches have been proposed that associate the same authorizations with all the documents in the same directory [6]. However, defining domains according to the physical distribution of documents does not satisfy the application requirements in most cases. Moreover, changing the authorizations associated with a document requires moving the document to a new location, thus invalidating document links.

Considerations when domains span different hosts include whether or not authorizations must be replicated and to what degree, and which policy to adopt for checking, updating, and revoking privileges. In addition, the site where authorization information is recorded must be determined. Two main approaches may be used: centralized and distributed.

Under the centralized approach, the AM and AUS both reside at the definition site, which may be different from the site of the RM [6, 7, 9]. Intuitively, authorizations referred to a domain are stored centrally, at the domain definition site, although objects participating in the domain may be stored at other sites. Upon receiving a request to access a document, the RM determines the domains to which the object belongs and contacts the corresponding remote AUS (Figure 3a) for access control. Each RM must know all the domains to which the document belongs, and the definition sites of those domains. In the DCE Web Toolkit [7], this problem is handled by forcing each client to directly contact an administration server (whose address must be known) upon each request (Figure 3b). That server authenticates the principal identity and provides the client with a certificate, which the client must present to the RM to access the requested document. The capability-based approach with a centralized administrator seems to be the most promising approach among those described in this work.

The centralized approach has the advantage that only one copy of the authorizations need to be maintained, making authorizations updates immediately applicable. As a drawback, the definition site may become a bottleneck since all the access requests are forwarded to it. The temporary unavailability of the definition site hinders all principals from accessing any document in the group. Finally, the principal is authenticated upon



(*) This request is generated by traversing a link.

(a)

(b)

Figure 3. Access control under the centralized approach.

each request it submits.

The distributed approach allows authorizations on a domain to be replicated at the different sites where domain documents are stored. This replication is generally only partial. Total replication does not appear viable because of the size of the systems involved. Upon receiving an access request, a RM can control the principal privileges with the local AUS. This approach is preferable when access requests are more frequent than authorization modifications.

Several replication patterns may be adopted, and both ACLs and capabilities may be used. For instance, each RM may maintain the ACLs related to its documents plus all documents accessible by navigation links. With direct access, the principal authorizations are checked locally. With access through a link, the RM of the source document can supply the principal with an appropriate capability to access the destination document (Figure 4a). Under this access scheme, the principal identity is authenticated only on the first access.

The pattern adopted in the Sessioneer protocol [2] (see Table 1), requires that the RM of the first directly accessed document contact a centralized authentication server. This server verifies the principal identity and returns the certificate proving it to the RM. If the authentication is successful, the RM satisfies the principal request. The solution proposed by Kahan [5], instead imposes that upon the first access the client obtains a capability from a centralized AUS/authentication server by directly contacting it. In both [5] and Sessioneer [2], documents are linked according to a hierarchical structure. In particular, in [5] they form a presentation tree, with nodes accessible by clients only by following the path from the root (Figure 4(b)). This choice has two consequences: only the AUS need know the identities of all the potential clients, while the tree nodes do not, and each document is reachable via exactly one link, hence only one copy of the access authorizations for it is maintained. Unlike [5], in Sessioneer the number of copies of the authorizations for a document equals the number of documents with a link to it. Another solution using a distributed approach is described in [8]. This solution also provides the delegation of capabilities from one principal to another, in the case where authentication between principals is supported.

The distributed approach requires the AM to know the topology of the links, in order to distribute the authorization information to the appropriate sites. Unlike in the

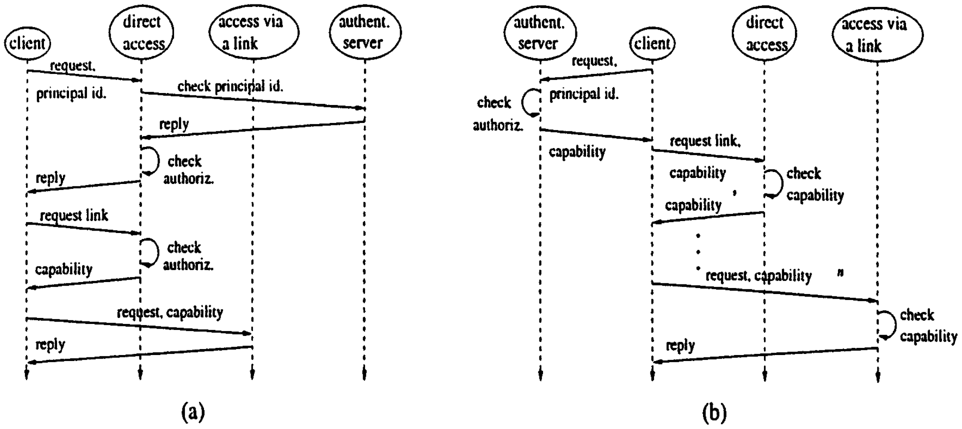


Figure 4. Access control under the distributed approach.

Protocol	[2]	[5]	[6]	[7]	[8]	[9]
Grouping	No	users/docum.	documents	documents	documents	documents
Control	distr.	distr.	centr.	distr.	distr.	centr.
Access	random	ordered	random	random	random	random
Authorization	capab.	capab.	ACL	ACL	capab.	ACL
Authentication	No	No	Kerberos	Kerberos	Kerberos	No
Replication	Yes	No	No	Yes	No	No

Table 1. Summary of different approaches to access control for Web documents.

centralized approach, only the first access to a document requires communication between the RM and the AUS, and the principal authentication. Subsequent accesses require a local checking of the capabilities and (possibly) the generation of new ones, while they do not need any principal authentication. On the other hand, when authorizations have to be modified, all existing copies must be accessed and their consistency must be guaranteed. Moreover, the temporary unreachability of a copy by the AM prevents it from modifying the authorizations, thus allowing principals to access documents for which they do not have appropriate authorizations.

Conclusion

Besides the access control protocols we have discussed in this article, other approaches based on a partial replication of the authorization information are possible. A first step in this direction is described in [2]. Other schemes could be devised replicating authorization information according to clients' access patterns. For instance, authorizations can be stored at more frequently accessed hosts or those with best response times. Techniques could be also devised for dynamically switching from one replication policy to the other according to access patterns, response times, authorization update frequency, or other parameters affecting the communication overhead. Each technique might require some heuristics to determine the best trade-off between the access efficiency for clients and data maintenance efficiency. Because of the number of hosts involved, scalability is a major requirement in the development of protocols for this environment. Another issue is fault-tolerance, for which recent techniques that have been developed for group communication in distributed systems could be exploited. It is important to note that we have focused here on server security rather than on browser and client security. Ensuring browser and client security is also crucial to prevent illegal access to or surreptitious transfer of documents.

References

1. ACM. About the ACM Digital Library. 1997; www.acm.org/dl/slide5.html.

2. Anderson, S. and Garvin, R. Sessioneer: Flexible session level authentication with off the shelf servers and clients. In *Proceedings of the 3rd WWW Conference* (Apr. 1995), 1047–1053; www.igd.fhg.de/www/www95/papers/77/sessioneer2.html.

3. Barkley, J., Cincotta, A., Ferraiolo D., Gavrilla, S., and Kuhn, R. Role-based access control for the World Wide Web; hissa.ncsl.nist.gov/rbac/rbacweb/paper.ps.

4. Bhimani, A. Securing the commercial Internet. *Commun. ACM 39*, 6 (June 1996), 29–35.

5. Kahan, J. A capability-based authorization model for the World Wide Web. In *Proceedings of the 3rd WWW Conference* (Apr. 1995), 1055–1064; www.igd.fhg.de/www/www95/papers/86/CAMWWW.html.

6. Lavenant, M.G. and Kruper, J.A. The Phoenix Project: distributed hypermedia authoring. In *Proceedings of the 1st WWW Conference,* 1994; www.cern.ch/PapersWWW94/j-kruper.ps.

7. Lewontin, S. The DCE Web Toolkit: enhancing WWW protocols with lower-layer service. In *Proceedings of the 3rd WWW Conference (*Apr. 1995), 765–771; www.igd.fhg.de/www/www95/papers/67/DCEWebKit.html.

8. Neuman, B.C. Proxy-based authorization and accounting for distributed systems. In *Proceedings of the 13th International Conference on Distributed Computer Systems* (May 1993), 283–291.

9. Samarati, P., Bertino, E., and Jajodia, S. An authorization model for a distributed hypertext system. *Knowledge and Data Engineering 8*, 4 (Aug. 1996), 555–562.

10. Schneier, B. *Applied Cryptography*, 2nd ed. Wiley, NY, 1996.

11. Winslett, M., Ching, N., Jones, V., and Slepchin, I. Using digital credentials on the World-Wide Web. *Journal of Computer Security 5*, 3 (1997), 255–267.

Security Service Protocols

SSL (Secure Socket Layer) uses the RSA public-key cryptography [10]. In SSL, servers are always authenticated, while client authentication is optional. A (trusted) certification authority generates certificates containing the name of the client together with its public key and possibly a timestamp. The certificate content is validated through the digital signature of the certification authority, whose public key is known. A server S is authenticated at a client C when S sends C its certificate, a random message m generated by C and the encryption of m with the private key corresponding to the public key that the certification authority has associated with S. This way, C can compare m with the m's encryption, previously decrypted with the public key contained in the certificate. A client is authenticated by means of a X.509 certificate that allows the server to verify the client's digital signature.

The S-HTTP protocol supports the same set of security services as SSL. Server authentication is always ensured while client authentication is optional in S-HTTP. The cryptography scheme is negotiated between the client and server, as well as the encryption keys used. Data integrity is guaranteed by a MAC. It may be stored together with each available document, thus guaranteeing validity in the case of a compromised server. S-HTTP differs from SSL because SSL is a session-layer protocol, while S-HTTP is an application-layer protocol embedded in HTTP.

The Kerberos authentication system uses the Data Encryption Standard (DES) technique [10] to encrypt all the exchanged messages. Messages are marked with a timestamp. Principals are grouped into realms that are under the administrative authority of an authentication server AUS. A client shares the server's key with its AUS. The client sends the AUS the identity of the application server S and the request it wants to submit to S; this message is encrypted with the AUS key. AUS replies with a ticket that certifies the client's identity to S and a session key used to encrypt the messages between the client and S. The ticket is encrypted with the S key. Should the client and the server be located at different realms, Kerberos is able to provide for cross-realm authentication. The AUS may also supply the client with ticket-granting tickets that allow multiple authentications without need for re-entering the information necessary for authentication every time. Both tickets and ticket-granting tickets have an associated expiration time.

Glossary

Authentication. Both client and servers must be confident they are in touch with the authorized and desired party. In practice, the service provider must feel secure of the identity of the client to whom it will associate the proper capabilities to access data, and the client must be secure that it is communicating with the desired provider. The problem in the Web is to ensure that the reciprocal identity insurance is maintained over the whole communication graph and lasts as long as the client navigation proceeds, without introducing authentication overheads at each communication hop.

Authorization. Rule stating accesses on data to be allowed or denied.

Access Control. Evaluation of access requests against specified access rules (for example, authorizations) and control policy to determine whether the request must be granted or denied.

Capability. Certificate released to a principal allowing the principal to exercise access on an object.

Access Control List (ACL). List associated with an object stating the subjects that can access the objects and the access modes they can exercise.

Confidentiality. To protect proprietary information and as a deterrent to theft of information services all communications between parties are restricted to the parties involved in the transaction.

Data integrity. Assurance of the correctness of the data against unauthorized or improper modification while the data are in transit or stored.

Nonrepudiation. The server must not be able to deny having supplied the client with the information the client has received from the server. It may be useful when the client wants to use that information in transactions with other parties.

Certificate. Data that proves the identity of the principal, for example, digital signature. It must be unforgeable (only the principal must be able to reproduce it).

Digital credential. Unforgeable and verifiable set of certificates that prove the identity of a principal, together with the proof the principal is the owner of those certificates.

Protocols for Web Access Control

Sessioneer [2]: Each resource manager in a group of resource managers maintains ACLs for both its document and all the documents accessible from its document. Through a direct access, the resource manager asks for the principal authentication at a centralized server; then, it locally controls the principal privileges. Upon receiving an access request via a link, the resource manager of the source document gives the capability for the destination document to the client.

[5]: Documents are grouped in presentation trees. A client must authenticate at a centralized AUS that provides him with a capability for the root document. Documents can be accessed only by following the path in the tree from the root. Each internal node provides the client with a capability for the next node on the path .

Phoenix [6]: All documents in the same directory are subject to the same authorization policy. The authorizations are recorded in ACLs, maintained in the same directory. The principal must supply server-specific authentication information at each contacted server.

DCE [7]: Documents are grouped into cells, each one managed by an administration server. Authorizations are recorded as ACLs by those servers. A client is authenticated by the administration server that provides it with a capability (certificate) to be presented to the resource manager. A client must obtain an appropriate certificate upon each request.

[8]: Each group of documents is under the authority of an AUS that provides the client with capabilities (proxies) to access the documents. A principal can delegate his proxies to another principal, possibly with restricted authorizations, if authentication among principals is provided [8].

[9]: Documents are grouped into domains and can be accessed either directly or via navigation links. All documents in a domain are subject to the same authorization policy. Upon receiving a request from a principal, the relevant resource manager contacts a centralized AUS that controls the principal identity and privileges. Authorizations are recorded in ACLs [9].