Authorization System in Open Networks Based on Attribute Certificates

Jeffy Mwakalinga¹, Eric Rissanen², Sead Muftic³

¹Department of Computer and System Sciences, Royal Institute of Technology, Kista, Sweden
²Swedish Institute of Computer Science, Kista, Sweden
³Department of Computer and System Sciences, Royal Institute of Technology, Kista, Sweden
Computer Science Department, The George Washington University, Washington DC, USA
[jeffy@dsv.su.se, erik.rissanen@bredband.net, sead@gwu.edu]

Abstract

This paper describes a security system for authorization in open networks. Authorization means authority to access certain resources, to perform certain operations, or to use certain system functions. In this paper the authorization system is based on use of attribute certificates. An attribute certificate is a signed object containing authorization attributes of a user. Before checking whether a user is authorized to perform an action or to access an object, the identity of the user must be verified. The identity verification system is based on public key certificates. We separate authorization system from authentication system because the same authority does not always establish authorization and authentication information. However these two systems must be combined and that is done by including the serial number of the user’s public key certificate as a field in the user’s attribute certificate, which carries authorization information.

The topology of the authorization system comprises authorization authority servers issuing attribute certificates to users, application clients handling those certificates, and application servers verifying user access rights based on attribute certificates. Furthermore, all these components are themselves certified by standard PKI certification authorities, thus supporting mutual authentication and cross-domain scaling.

Keywords: Certification authority, attribute certificate, attribute authority, authorization and access control models.

1 Introduction

1.1 General Principles

This paper describes a generic system for authorization in open networks based on attribute certificates. Authorization means authority to access certain resources, to perform certain operations, or to use certain system functions. Authorization addresses three major problems: identification of users and assignment of globally recognized roles; matching of user roles with authorization attributes like security labels; enforcement of authorization privileges and making decisions. Today organizations run Web servers and resources of these servers should be accessed globally only by authorized people. For instance, companies have IT resources, which may only be accessed by customers who subscribe for them. In most cases a customer, who subscribes for resources, is given a user name and a password and can log in using these tokens to the servers. A user should be able to access the resources from any machine in the global network. A customer may decide to pass the username and password to friends. Friends can then access the resources without having paid for them. Authorization systems have to provide a mechanism for minimizing this risk. An authorization system should make it possible for a client to verify whether the signer of a certain check is authorized to do so. In all these cases a secure and global system of authorization is required. Clients have to be authenticated before checking whether they are authorized to access or perform a task. The first task of an authorization system is therefore to authenticate clients. So how can clients be reliably authenticated in an open network?

There are two types of authentication schemes: simple authentication and strong authentication.
this system we use using strong authentication and clients and servers mutually authenticate each other. It is based on public key certificates. A client is required to present her public key certificate for authentication to the server. The second task of an authorization system is to check whether the authenticated client is authorized. This is described in section 4.2. What are the requirements of an authorization system in open networks?

1.2 Requirements

Authorization system in open networks must be combined with an authentication system because users must be authenticated before verifying their authorization to access resources or perform certain functions. Authentication and authorization should be done by separate systems, because the same authority does not usually create authorization and authentication information. The system must be secure so that people can trust it. It should be possible to delegate rights and privileges to other entities. It should be easy to administer which implies that an authorization system should have a user-friendly interface. The system should be scalable and efficient, because it is used in global systems where delays are not acceptable. It should support distribution of authorization elements. Authorization in open networks should be flexible supporting alternative authorization policies.

1.3 Authorization Policies

An authorization system is based on authorization policy of an organization. An authorization policy specifies rules for accessing objects or performing certain actions. This policy can be specified in terms of access control lists, capabilities, or attributes assigned to subjects, objects or both. Policies are usually described by access control models. An access control model is an abstract description of an access control system and its main goal is preventing unauthorized access to resources of a computer or information system. An access control model comprises the following items: a target, which is the object to be accessed; an initiator, which is an entity wishing to access the target and an access control function, which uses access control information to decide whether a subject can access a target. Access control function passes its decision to an access control enforcement function, which provides access to the target information or prevents it based on the output of the access control decision function.

Organization of Sections

Section two covers current approaches. Section three deals with the principles of an authorization system in open networks. Section four describes a prototype of the authorization system. Section five briefly discusses conclusions.

2 Current Approaches

2.1 Some Solutions on Restricting Access

Authorization in open networks can be based on IP addresses and domain names [4] in which case a server examines the incoming request and grants or denies access depending on the IP number or domain name. IP-based authorization is not suitable for mobile clients and it does not accommodate dynamically allocated or shared IP addresses. This type of authorization is not secure, because today it is relatively easy to forge IP numbers. The system is vulnerable to DNS spoofing and IP spoofing where an attacker takes control of the DNS host-names’ lookup system. As a result a server can be led to believe that it is talking to a trusted host. How can one verify whether an IP address is genuine? One way is to extract the IP address and then double-check with the DNS system of the client. A request can be made to the DNS to return the host name of the IP address to be checked. Then another request is made to the DNS system to return the IP number of the host name returned in the previous request. If these match then the IP address is most likely genuine.

It is also possible to minimize the problem by using firewalls, which use reliable DNS lookup. But how can one determine whether a DNS lookup is reliable? Are there any trusted and reliable DNS lookups today? Can firewalls be trusted? These systems must be properly configured in order to function correctly and not all firewall administrators are competent in this area.

Authorization can be based on certificates. When a user requests a service, she presents a digitally signed certificate together with the request. A server grants access if certificate is valid. To be valid means, that the chain of certificates has been validated and that the certificate has not been revoked.

2.2 Role–Based Access Control (RBAC) System for Securing a Web-based Workflow

Ahn, Sandhu, Kang and Park [2] describe a way to add a RBAC system to an existing web-based workflow system. A web-based workflow system consists of an interface for clients, a gateway to external services, a tool for protocols, and workflow tool for descriptions and enforcements, where
activities are performed in coordination. Different servers execute different tasks. These systems provide only low-level security services such as simple authentication. Authentication and authorization security services are based on public key certificates. The system uses HTTP protocol for client–to–server communication and uses CORBA’s network addressing protocol for server-to-server communication. Different roles are attached to each task. Users’ identities are verified and then checked whether authorized to perform tasks, which they request Role–Based Access Control (RBAC) model in this system has a set of roles, a set of permissions, and users. This model supports role hierarchies. Permissions are assigned to roles and users may have different roles. Users can have one or more roles. A role can be assigned one or more permissions and vice versa.

RBAC system consists of three major components: a workflow design tool, a role server and a Web-based workflow system. The workflow design tool is used for the administration of the system: generating roles, building role hierarchies, assigning roles to tasks, specifying flows of information and relationships among tasks, and for passing information to the role server. The role server has two components: a user-role assignment component and a certification server. The functions of user-role component include assigning users to roles, and creating and managing role hierarchies and databases. The certification server is responsible for verifying users’ identities, fetching users’ information from databases and issuing certificates with users’ role information. The workflow system contains Web-based task servers. A task server approves authorization to a client based on the information found in user’s certificate. The client is given authorization during the establishment of SSL session between a client and a task server. The Web server asks for a client certificate during SSL handshaking procedures. Client sends a certificate to the server. The server verifies the identity of the client. The server extracts authorization information from the client’s certificate and checks whether this client is authorized.

The advantage of this system is that very little changes need to be made on the server side and no changes on the browser’s side. If one of Web servers gets manipulated, it doesn’t cause the system to stop, because servers are doing multiple and different tasks. The disadvantage of this system is that both authentication and authorization information are based on public key certificates. Authorization and authentication information can be set and updated by different authorities. It is also inconvenient with respect to policy management, because different authorities can have different policies. Validity of authorization information and authentication can also be different.

2.3 One-Shot Authorization System using Smart Cards

Au, Looi and Ashley [1] present an authorization system based on smart cards. This system can be used in coordination with any existing authentication system and it can authorize clients across multiple domains. In one domain the system consists of three components: a client workstation, a security server, and an application server. The client workstation is connected to client’s smart card reader. On this workstation there is a program called Authorization Token Manager. This program communicates with an application server and the administrator of the application server installs it on the client side. This program retrieves one–time authorization tokens, verifies them and stores them in the smart card together with private keys and other information. Client’s smart card authenticates remote servers, verifies authorization tokens and also creates session keys. After using these one-time tokens the program renews them. Security server contains two modules: an authentication server and an authorization server. An authentication server verifies identities of clients. An authorization server performs authorization services. The security server communicates with an application server to get initial and updated authorization information. It also communicates with the workstation to exchange authentication information. The application server maintains an access control list, a valid token ID list, and access control information list.

The advantage of this system is that authorization tokens are one-time, which solves the problem of replay. The disadvantage of the system is that it creates heavy traffic, because only one-time authorization tokens are issued. Another shortcoming of this system is that it is not explained how the messages are protected while in transfer, so it is difficult to determine how secure the messages are during this process.
3 Use of Attribute Certificates for Authorization in Open Networks

3.1 Attribute Certificates

An attribute certificate (AC) is a signed object containing authorization attributes of a subject. Attribute Authorities (AA) are the components responsible for issuing attribute certificates. The serial number of the client’s PKI certificate, which is used for authentication purposes, is inserted in a field called *holder*. Fields of an attribute certificate according to [6] are:

- **Attribute certificate information**
- **Signature algorithm identifier**, which is an algorithm used to sign the AC
- **Signature value**, which is a signature of the issuing AA

The fields in the attribute certification information include:

- **Version**: This field contains the version of the attribute certificate (AC).
- **Holder**: This field contains the identity of the holder of the certificate, that is the entity to whom the attributes apply. It has the serial number of the owner’s public key certificate, general names of the AC’s owner, digest information, which can include public key, public key certificate, digest algorithm and so on.
- **Issuer**: It contains the identity of the issuer of the attribute certificate.
- **Signature**: This contains the algorithm that was used for signing the attribute certificate.
- **SerialNumber**: It has a serial number of the attribute certificate.
- **AttrCertValidityPeriod**: This field contains the validity period of the attribute certificate in the form of two dates defining a time interval.
- **Attributes**: This field contains the actual attributes and is specified by the issuer of the attribute certificate. These attributes include service authentication information, access identity, charging identity, group, role, clearance and etc.
- **IssueUniqueID**: This field contains additional information to help locate the issuer.
- **Extensions**: Extensions contain some additional information about the attribute certificate: *audit identity* for audit trails; *attribute certificate targeting*, which is used to specify the number of targeted servers or services; *authority key identifier*, which is used to assist in verifications of the signature of the attribute certificate; *authority information access*, which is used for checking revocation status of a certificate; *CRL distribution points*, etc.

Attribute certificates are stored in the same way as public key certificates: in global repositories or in directory systems. Attribute certificates can be revoked. But in cases when their lifetimes are too short, revocation may not be necessary. Revoked attribute certificates can be stored in attribute certificates revocation lists. This is a list of AC’s serial numbers. It must be possible to verify the authority of the issuing AA, i.e. there is a valid chain of public key certificates containing the extensions asserting AA’s authority. In inter-domain environments there should be a way of translating attributes issued by other domains into the domains responsible for validating the ACs. Attribute certificates should keep all or some of its attributes confidential if so desired by clients. Attribute certificates are useful in supporting delegation.

3.2 Authentication of Clients and Assignment of Roles

When a client connects to an authorization server for the first time she is authenticated by presenting her public key certificate. This certificate is verified by validating certification chain from the authority, which issued the certificate to the top certification authority in the hierarchy. A check is also made to verify that the certificate in question is not revoked. If the certificate is found to be valid then an attribute certificate is issued to the client. Roles and clearances are given to the client and they are written in the client’s attribute certificate. These roles and clearances specify authorization of the client and these specifications are stored in the policy file of the attributes authority. A reference to the client’s public key certificate is also included in the attribute certificate in the field called holder. In this attribute there is a sub field called *baseCertificateID* and this sub-field holds the serial number of the client’s public key certificate. After populating all the fields of the client’s attribute certificate, the certificate is signed by the issuing AA. If the client desires to protect some fields of the attribute certificate that can be done using a secret key. The attribute certificate is then stored in the X.509 Directory or in a global database. A copy of this attribute certificate is sent to the client.

3.3 Synchronization of Roles and Authorization Attributes

When a client makes a request to access resources of a secure Web server, she presents her public key certificate. This certificate is validated as described in section 3.2. If validation is successful then the serial
number of this public key certificate is used to pull the client’s attribute certificate from the directory or from the global database. If client’s AC is not found at the server or if the database or X.500 directory is down, then the client is requested to send her AC to the Web server. Every resource in the secure Web server has a security label. Labels are attached to resources by using S/MIME. S/MIME is a standard for encapsulating MIME documents and provides services like confidentiality, integrity and authentication. Confidentiality is a security service, which protects resources from illegal read, illegal access, deletion, sabotage and so on. Integrity is a security service that protects resources from illegal modification, deletion and etc. The resources are stored in the security Web server in encapsulated forms. The security labels that are attached to resources specify in the policy file which roles and clearances can access the corresponding resources. The security label has a list of all roles, which can be granted access. The policy file contains information on security classifications and categories. It can contain information mappings among different security policies. If a policy of a company changes then it is enough to update the policy file without needing to change other modules. A Policy Creation Authority (PCA), which is a trusted entity, signs the policy file. Security labels and clearances have policy identities, which are references to the policy files in which they are specified. The policy file contains lists of security classifications and categories and all allowed combinations of them. All messages between a client and a secure Web server are protected using S/MIME, SSL or other secure protocols.

3.4 Enforcement in the Authorization System

Decisions to grant access to the secure Web server’s resources are based on the policy of the AA. This policy is created by the Top or Root certification authority and all the entities under this root certification authority use this policy. Roles, clearances, ranks, security labels and other attributes and information are specified in this policy file. The attribute certificate of the client is pulled from the global database or from the X.509 Directory. The security Web server must verify the attribute certificate by verifying the signature of the attribute certificate. The validity of the AC must also be checked. The subject in the attribute certificate, the AC’s issuer, and the complete certification chain is validated. A local certification authority, as explained in section 3.5, certifies the AA. The client’s AC contains clearances or roles of the client. These attributes specify the authority of the client. Access control decision function takes as parameters, a policy file, a security label, and an authorization set and this set includes a clearance, a role and other parameters. Access is granted if the client’s attribute certificate is verified and if the client has a clearance or a role that matches the security label of the requested resources.

3.5 Management Infrastructure

The system uses the X.500 authentication framework. This system uses certificates-based authentication. Clients are required to have public key certificates before being authorized to access or perform actions in the authorization system. Certification authorities (CA) certify attributes authorities (AA), which issue attribute certificates. The complete system is shown in Figure 1. At the top there is a trusted root certification authority. Below this root CA there is one or more intermediary certification authorities depending on the complexity or size of the organization. The last CA in the hierarchy is a local CA. This is responsible for certifying the Attribute authority, managing public key certificates to clients, managing keys, revoking certificates and so on. At the root CA there is a Security Policy Information File (SPIF) for the entire system. This file contains the policy for the whole system. Every certification authority has a certificate, which contains an extension called cAClearanceConstraints. This extension enables authority to act as an Attribute Authority (AA). The root CA issues a self signed certificate to itself. It then issues certificates to the lower entities. If the root CA belongs to a company then this company can have middle certification authorities in different countries where it has offices or its business.
Certificates issued to lower entities have to be verified by checking the signatures of certificates. The whole chain up to the root CA has to be validated. The local CA issues a certificate to the AA, which in turn issues an attribute certificate to the end entity. The policy file, SPIF, has to be signed by the root CA so end entities must verify the signature before using it. In cases where there are different root certification authorities and belong to different organizations then root certification authorities are required to cross certify each other so they have to issue certificates to each other and these certificates will contain the corresponding cAClearanceConstraints extensions.

3.6 Delegation of Attributes

Delegation of attributes is done with the help of a file called attribute in the attribute certificate and also with the help of an extension in the AC that is called authority information access. Authority information access has an IP address of the directory where the issuer of the attribute certificate may be found. This extension can also store an IP address of the directory that has the AC of the upper entity that delegated attributes to the lower entity. When the Web server receives a request from a user it can authenticate her as described in section 3.3 and if authentication of the user is successful, the Web server will retrieve the user’s AC and check its validity as discussed in section 3.4. If attributes are delegated then the attribute’s value will be delegated set of attributes. The Web server will thereafter get the AC of the delegating entity from the directory whose IP address was in the authority information extension. The AC of the upper entity will be verified as discussed in section 3.4. The user will be authorized if the AC of the delegating entity is valid.

4 Implementation of a Prototype

This prototype is based on geotronics [7] library suite. The RBAC is implemented using access control library in the following way. It is specified in the policy file, SPIF file, as described in section 3.4, so that all the roles are given security categories. Categories are authorities to perform different functions or access different objects in the secure Web server. Every security label has a list of roles, which are authorized to access certain resources or perform the desired actions. Every clearance in the attribute certificates contains a list of the roles, which can be granted access.

4.1 The Access Control Library Suite

This authorization system uses the access control library [3] and it consists the following libraries.
SNACC. This is a high performance ASN.1 to C/C++ Compiler. This library contains an ASN.1 compiler for encoding and decoding data structures.

S/MIME Freeware Library (SFL). This library provides support for cryptographic functions like signing, verifying signatures, protecting messages and so on.

Certificate Management Library (CML). This is used to verify the certification paths.

The Storage and Retrieval Library (SFL). This library is used for maintaining the database for certificates. SFL is used for providing functions for parsing, generating, protecting and verifying SMIME messages.

Access Control Library (ACL). This library takes care of access control decisions basing on S/MIME security labels, X.509 certificates and attribute certificates.

4.2 Implementation

There are three components in the prototype: An administration tool, a SPIF generator and a certification manager. An administration tool, AdminTool (figure 4), is used for managing roles and S/MIME documents. The SPIF generator is used for generating policy files, SPIF.

The administrator chooses an item to be generated from the interface in Figure 2. The administrator can choose to generate an SMIME document, an attribute certificate or a new Security Policy Information File (SPIF), see Figure 2:

![Figure 2: Choosing an Item to Generate](image)

The administrator of the system creates a policy file as explained in section 3.4, enforcement in the Authorization System. She/he does this by activating the SPIF generator and a panel shown in Figure 3 will be displayed. In the SPIF generator, one has to specify the policy ID and a version of this policy file.

![Figure 3: SPIF Generator](image)

Then roles have to be created. After creating the roles, SPIF file must be signed using the private key belonging to the issuer of the policy file.

To issue an attribute certificate as discussed in section 3.2, recognition of clients and assignment of roles, an administrator selects option attribute certificate from the interface in Figure 2. Thereafter the administrator selects the policy file and public key certificate for authentication purposes as described in section 3.2. Different fields like serial number, validity, roles, etc, are populated in the attribute certificate. The attribute certificate is then signed. Before storing the attribute certificate to the database, trusted certificates must be added to the database or to the directory system. These certificates
are necessary for certificates chain validation as discussed in section 3.2. This is done using Certificate Manager interface, shown in Figure 4. Attribute certificate can then be added to the database or to the X.500 Directory.

The next step for security administrator is to attach security labels to resources (in this case Web documents) as described in section 3.3. To add documents to the Web server, the administrator selects SMIME Document option from the AdminTool panel and this panel shown in Figure 5.

Then he/she selects a document to be encapsulated and the corresponding SPIF to be used. The administrator then specifies the roles, which can access this document.

The private key for signing the security label must be specified. When a client requests to access a site on the Web server, the server expects client’s public key certificate. The Secure Socket Layer [8] is used for establishing secure sessions between the client and Web server. SSL is a system for securing messages while in transfer. The server checks whether client’s public key validates the client’s digital signature as discussed in section 3.2. It also checks whether today’s date is within the certificate validity period. It also checks whether the CA that issued client’s certificate is a trusted CA and also whether the public key of client’s certificate issuer validates the issuer’s digital signature. The server checks whether this certificate corresponds to the serial number in the attribute certificate. Then the Web server checks with the ACL to decide whether an incoming request is authorized to access the site. Web server loads the Publish dynamic library and passes the name of the selected document as a parameter to the access method of the extension. Documents are stored on the server in S/MIME format and contain security labels as described in section 3.3. The extension function fetches user’s attribute certificate from the X500 Directory and compares the role in it with the security label of the requested document. If the client is assigned the roles contained in the security label of the document, the document will be transferred to the client. If the client is not authorized to access the file, he/she will get an http “404 Not found” response.

Figure 4: Certificate Manager
5 Conclusions

This system is flexible and interacts with other systems like PKI certification system, X500 directory system and smart card systems. Attribute certificates support delegation through an ordered sequence of attribute certificates with references to certification authorities. Attribute certificates can be used for non-repudiation services making it possible to extend authorization systems to support this service. The system separates authentication security service from authorization making it possible for authentication and authorization decisions to be made by different authorities when necessary.

References
