Authentication Service Model Supporting Multiple Domains in Distributed Computing

Kyung-Ah Chang, Byung-Rae Lee, Tai-Yun Kim

Dept. of Computer Science & Engineering, Korea University,
1, 5-ga, Anam-dong, Sungbuk-ku, Seoul, 136-701, Korea
{gypsy93, brlee, tykim}@netlab.korea.ac.kr

Abstract. In this paper, based on CORBA security service specification[1, 3], we propose the authentication service model supporting multiple domains for distributed computing with an extension to the Kerberos[13] authentication framework using public key cryptosystem[15]. This proposed model supports the protection of the high-level resources and the preservation of the security policies of the underlying resources that form the foundation of various domains, between the Kerberized domains[14] and the Non-Kerberized domains. Also we achieved the flexibility of key management and reliable session key generation between the Client and the Provider using the public key cryptosystem.

1 Introduction

The traditional requirements of security mechanisms and policies are exacerbated in the current distributed computing, as the physical resources of this exist in multiple administrative domains, each with different local security requirements. Much attention has been devoted to security issues and it is apparent that a high level of security is a fundamental prerequisite for Internet-based transactions, especially in the electronic commerce area.

As a consequence, the need for standard architectures and frameworks for developing such applications has arisen. The OMG[1] has specified the CORBA in response to these needs[2, 4, 6]. CORBA[8, 9] is a standard middleware supporting heterogeneous networks, designed as a platform-neutral infrastructure for inter-object communication. However, CORBA based security service specification[1, 10] itself does not provide any security mechanism[6]. The predefined attributes of CORBA security service have only limited validity and many security mechanisms do not provide sufficient security attributes.

In this environment, the number of new users and applications requiring authentication will continue to increase at a rapid rate. Clients must authenticate themselves to the Provider system. Clients must not be allowed to access arbitrary resources. Thus a more elaborated security infrastructure must be provided in different administrative domains.

This paper, based on CORBA security service specification[1], proposes the authentication service model supporting multiple domains with an extension to the

Kerberos[13] authentication framework using public key cryptosystem(PKC)[15]. This proposed model, by PKC based certificates, assures the identification of a partner in the authentication of peer entities and the secure access to multiple domains in the authorization of underlying resources. Since our deployed Kerberos is extended to the authentication service model, it provides the flexibility of key management and the ability to leverage the public key certification infrastructure[7].

The organization of this paper is as follows. Section 2 presents the CORBA security service and the description of authentication service model in this paper. Section 3 describes the structure of the authentication service model supporting multiple domains in detail. Finally, Sections 4 and 5 contain a performance and conclusion, respectively.

2 Security Service Approaches

2.1 CORBA Security Service

The CORBA security service specification[1] is large in part due to the inherent complexity of security, and due to the fact that the security service specification includes security models and interfaces for application development, security administration, and the implementation of the security services themselves. All these and their interfaces are specified in an implementation of independent manner[9, 10]. So the interface of security service is independent of the use of symmetric or asymmetric keys, and the interface of a principal's credential is independent of the use of a particular certificate protocol.

The objective of this specification[1, 2] is to provide security in the ORB environment in the form of an object service. The focus lies hereby on confidentiality, integrity, and accountability. The model used by the CORBA security service specification involves principals that are authenticated using a principal authenticator object. Once authenticated, a principal is associated with a credential object, which contains information about its authenticated identity and the access rights under this identity. These credentials are then used in secure transactions, to verify the access privileges of the parties involved, and to register identities for actions that are audited or carried out in a non-repudiation mode.

The Client requests a remote object through a local reference. The Client's credentials are attached to the request by the security services present in the ORB, and sent along with the request over the transport mechanism in use. The remote object receives the request through its ORB, along with the Client's credentials. The target object can decide whether to honor the request or not, based on the access rights of the Client's identity.

When a request is received from a remote object, its right to access the resources requested can be checked through an access decision object, which can be used to compare the remote principal's credentials against access control settings. Typically there is not a default access control policy that the security services will enforce requests, since checking access rights are usually very application-specific.

2.2 Description of Authentication Service Model

On Internet, Certificate Authorities(CA) acts as trusted intermediaries when authenticating clients and servers. *Authentication Service Provider* in our proposed model is another form of trusted intermediary[14]. In a public key scheme, a CA issues a long-lived credential – a public key certificate. When both clients and servers have such certificates they can authenticate to each other without further reference to a CA. However, precisely because these certificates are long-lived, some method is required to inform servers of revoked certificates. This can be done by requiring servers to check a certificate's current validity with the CA on each use of a certificate, or by distributing Certificate Revocation Lists(CRLs) to all servers periodically. In the proposed scheme, the Authentication Service Provider issues clients a short-lived credential, which must then be presented to obtain an access right for a particular server.

The Authentication Service Provider described in this paper is structured in layers; *Exchange Layer*, *Supporting Services Layer*. The Exchange Layer and Supporting Services Layer are responsible for the execution of CORBA security service by adding of a security and a message interceptor[3]. The Exchange Layer provides services for handling and packaging business items as well as transfer and fairness of mutual exchanges. The security attributes stored in each type of the basic objects determine the label of privilege that is required for the exchange.

In the Exchange Layer, we concentrate on the *Credential Service Block* that receives parsed messages by the exchange manager. This block handles a credential of each participant, thereby performing all secure invocations between a Client and a Provider. This block contains *Authentication Service*, *Authorization Service*, and *Session Identifier Service*.

The Supporting Services Layer provides persistent object storage, a communication, and a cryptographic service. The communication services block supports communication between multiple domains entities. The architecture can support any other networks as well as the Internet. The cryptographic service block provides cryptographic primitives like message encryption or decryption, and key distribution. The object storage service block supports persistent and secure local storage of data; principal credential, required right, and domain access policy, etc.

3 Authentication Service Model Supporting Multiple Domains

Current distributed computing is really a federation of resources from multiple administrative domains, between the Kerberized domains[14] and the Non-Kerberized domains, each with its own separately evaluated and enforced security policies.

In this paper, we propose the Kerberos based Authentication Service Model supporting multiple domains. The assumption of our scheme is that only objects of users that have been authenticated must be authorized to use the underlying resources over multiple domains. And assume that system administrators will allow their systems to participate in our Authentication Service Model.

3.1 Specification of Kerberized Authentication Service with PKC

Every object of our model has a credential that contains security information including a key in Kerberized system, or contains granted rights in Non-Kerberized system. This will support the authentication for peer entities to perform the mutual negotiations, and the authorization to control the domain accesses.

For our Kerberized service, the authenticated key exchange protocol relies on the ElGamal key agreement[16, 17]. It is well known that the ElGamal cryptosystem for any family of groups for which the discrete logarithm is considered intractable. Part of the security of the scheme actually relies on the Diffie-Hellman assumption, which implies the hardness of computing discrete logarithms[17]. **Fig. 1** shows a protocol of ElGamal key agreement.

1. B picks an appropriated prime p and generator of Z_p^* and selects a random integer b , $1 < b < p-2$. After computes $b^b \bmod p$, B publishes its public key (p, g, b) , keeping private key b secret.
2. A obtains an authentic copy of B 's public key (p, g, b) and chooses a random integer x , $1 < x < p-2$. Then A sends B follow protocol message and computes the key as $K = (b^x) \bmod p$.
 $A \rightarrow B : x \bmod p$
3. B computes the same key on receipt of the message as $K = (g^x)^b \bmod p$.

Fig. 1. ElGamal key agreement protocol

We define two important security methods we want to provide: the *authentication()* and the *authorization()*. The *authentication()* should normally be done by the User Sponsor(US). The Session Identifier based the *authorization()* controls the access to the Providers, and the operation the Client wants to invoke. Finally, we declare the interface of authentication service provider that inherits the interfaces of the *authentication()* and the *authorization()*.

With *authentication()* method, the US initiates the authentication exchange by requesting the Session Identifier(SID_{azs}) from the Authentication Service Block(AS). This is necessary since the construction of the subsequent Authorization Service Phase requires the certificate of Provider($CertP$). **Fig. 2** shows the process of obtaining the Session Identifier (SID_{azs}) based on Kerberos[13] using PKC[15], between User Sponsor and Authentication Service Block.

Fig. 2. Authentication Phase based on Kerberos using PKC

At the start of the protocol, there is one of the assumptions that AS has long-term secret and public key agreement keys v and g^v . Another assumption is that US possesses the public key necessary to verify certificates issued by Authentication Service Provider. In first request message, as shown in **Fig. 1**, US generates a random

number u and computes temporary public key agreement key g^u . The US then generates an encryption session key $L = (g^v)^u$ where g^v is the public key agreement key of the AS.

On receipt of the first message, AS does not know with whom he is communicating. AS computes $L = (g^u)^v$ and generates a session key $K_{u, azs}$ between US and Authorization Service Block(AZS). He then sends to US message encrypted using L together with Session Identifier(SID_{azs}) to access AZS encrypted using the secret key of the AZS, K_{azs} .

Once the US has obtained SID_{azs} , it implies being authenticated by an authentication object, and can proceed to generate SID_p for service request. The message contains similar information to that in a traditional ticket request of Kerberos.

The method authorization() handles a Session Identifier(SID_{azs}), the Provider(id_p) that the Client wants to access, and the name of the operation to invoke. We get Provider's name(id_p) from the object storage of Kerberized hosts and the principal(id_u) from the Session Identifier(SID_{azs}). If the name of the Provider(id_p) the Client wants to invoke is among these, this is allowed to proceed. If the authorization succeeds, the operation and the returns are subsequently invoked on the Provider. If not, the CORBA system exception[8] 'NoPermission' is flagged. **Fig. 3** shows the process of obtaining another Session Identifier (SID_p) based on Kerberos using PKC, between User Sponsor and Authorization Service Block.

Fig. 3. Authorization Phase based on Kerberos using PKC

At the start of this phase, as shown in **Fig. 2**, there is an assumption that AZS has kept key escrow of Provider and has shared the domain($Domain_p$) with Provider. In first request message, US send to AZS Provider's identity(id_p) with SID_{azs} encrypted using secret key of AZS(K_{azs}).

On receipt of the first message, AZS decrypts the message using his secret key. It then retrieves the session key between US and AZS, which is found in the SID_{azs} . And then he generate the appropriate certificates required in the protocol($CertU$, $CertP$) and the Session Identifier(SID_p) to access Provider. He send to US theses messages together with Session Identifier(SID_p) encrypted using the secret key of the Provider(K_p).

The Session Identifier(SID_p) received by the AZS is simply a conventional service ticket. At the start of this phase, as shown in **Fig. 3**, US has Provider's public key g^p in the certificate $CertP$ and Client's private key w in the certificate $CertU$. And then he computes an encryption session key $K_{u, p} = (g^p)^w$. In first request message, US send to PS the certificate of client($CertU$) and SID_p and the Authenticator which additional data needed as input to the payment scheme with encrypted using encryption session key $K_{u, p}$. **Fig. 4** shows the process of obtaining service, which the Client wants to

access based on Kerberos using PKC in the Open Authentication Service, between User Sponsor and Provider Sponsor(PS).

Fig. 4. Service Access based on Kerberos using PKC

On receipt of the first message, PS computes an encryption session key $K_{u,p} = (g^w)^p$ then decrypts the encrypted Authenticator message. All operations from this point on can protocol per normal Kerberos operations.

3.2 Mechanism of Authentication Service Model for Kerberized Domains

On Kerberized domain, our client’s request of Authentication Service Model needs initial objects to reference the security service, like it does for other CORBA based services. These objects are ‘SecurityLevel2::UserSponsor(or ProviderSponsor)’[8, 10] and ‘SecurityLevel2::Current’. Fig. 5 shows a mechanism of Kerberized multiple domains access based on CORBA.

Fig. 5. Kerberized domains access based on CORBA

The UserSponsor calls on the PrincipalAuthenticator object, which authenticates the principal and obtains the Credential containing authenticated identity and privileges.

The Credential has to be created from its own certificate. This object holds the security attributes of a principal, e.g. authenticated identity and privileges. It is used by the application to create its own security information that later should be sent to the remote peer during the establishment of the secure association between the Client

and the Provider. The Current object and Vault object provide the system with interfaces to processing and storage resources. The use of these objects interfaces is encapsulated by UserSponsor(or ProviderSponsor) objects.

UserSponsor(or ProviderSponsor) provide a central mechanism for specifying policy for a set of like objects. For setting policy for instances, UserSponsor serves as location authorities for instances, supporting the binding of secure association.

Once an invocation passes an interaction point, the Secure Invocation Interceptor establishes a security context, which the Client initiating the binding can use 'Current::get_policy', to securely invoke the Provider designated by the object reference used in establishing the binding as 'get_policy'. After binding time, the Secure Invocation Interceptor is used for message protection, providing integrity and confidentiality of requests and responses, according to the quality of protection requirements specified in the active security context object.

Finally, as shown in figures of previous section, it can establish a fairness of exchanges through the Authentication Service Provider using items bundled as a CORBA object. Both the UserSponsor and the ProviderSponsor transfer Session Identifiers(SID_p) before mutual exchanges occur. This prevents the Current from being transferred Current to illegal peers, and prevents the Clients from being given access to illegal peers.

3.3 Case of Authentication Service Model in Multiple Domains

For multiple domains, all communication of our model is done via Kerberos mechanisms. Thus cross-realm authentication[13] is immediately and transparently supported: UserSponsor only has to be performed once for each group of Kerberos realms that support cross-realm authentication with each other. The initial objects will automatically obtain SID_{azs} for the other realms based on the existence of a valid SID_{azs} for a given host.

Our model is assumed with Kerberized domains basically, however, we must consider multiple domains, specifically integration of Kerberized(K_{sys}) and the Non-Kerberized domains(NK_{sys}). We propose the approach to issue temporary credentials to NK_{sys} 's request objects[11, 12]. Most of all participated NK_{sys} involve a single CA. In general, users in one domain are unable to verify the authenticity of certificates originating in a separate domain. However, our K_{sys} 's Authentication Service Provider issues a cross-certificate[4] based temporary certificate. A trust relationship between CAs of multiple domains and our Authentication Service Provider must be specified in order for users under CAs to interoperate cryptographically.

The essential component of our approach is a KProxy object for each NK_{sys} client's request. This KProxy securely holds the delegated credentials as K_{sys} client's in the local Kerberized system. Whenever the client's request from the NK_{sys} wants to create an object on client's behalf on its associated physical machine, the Authentication Service Provider creates the Current object that contains minimal permission. Provider's AS will only issue client's delegated credentials of that domain if client's valid temporary certificates are presented in the request. A delegated credential specifies exactly who is granted the listed rights, whereas simple possession of a bearer credential grants the rights listed within it. Then the Current performs a call

back to the KProxy for client to obtain a SID_{azs} for that particular Provider. **Fig. 6** shows the authentication mechanism of multiple domains access.

Fig. 6. Authentication mechanism of multiple domains access

After obtaining a SID_{azs} , Provider’s AZS is obtained the attributes from the KProxy object by calling ‘get_attributes’. Then all operations from this point on can mechanism per Kerberized operations for multiple domains.

The access control mechanisms of client’s KProxy can be configured to issue the thread specific Credential obtaining the information to the thread of request execution. There might be more than one thread, every thread associated with a different set of security attributes. These can be accessed from the appropriate Current has to be used.

4 Security and Analysis

4.1 Security of Our Authentication Service Model

In distributed computing, the DCE[18] and the SESAME[5, 19] are the well-known security systems, based on Kerberos, of the Client/ Server architecture. As seen in the **Table 1**, our proposed Open Authentication Service shows better security features like the authentication for peer entities to perform the mutual negotiations and the fairness of exchange. Therefore it has the good advantages of interoperability with other security services.

Table 1. Security Analysis of Proposed Schemes

	DCE	SESAME	Authentication Service Model
Access control level	Application	Application	Application/ System
Authentication	Unilateral	Unilateral	Unilateral/ Mutual
Authorization policy	ACL based	ACL based	Label based Rule
Fairness of exchange	No	No	Yes
Flag of privilege type	Positive	Positive/ Negative	Positive
Grant/ Revoke privileges	Controlled by Server	Controlled by Server	Label based Rule
Scalability	Average	Average	High
Security policy domain	Server's domain	Server's domain	System Imposed
Suitability	Stable User base	Stable User base	Mandatory Controls

And then **Table 2** shows comparison of key distribution Scheme with traditional Kerberos. Our proposed Authentication Service Model provides session key establishment mechanism based on PKC.

Table 2. Comparison of Key Distribution Scheme with traditional Kerberos

	Kerberos	Authentication Service Model
Session Key Establishment between User and TGS(AS)	Symmetric Key Transport	ElGamal
Session Key Establishment between User and SGS(AZS)	Symmetric Key Transport	Symmetric Key Transport
Session Key Establishment between User and Service Provider	Symmetric Key Transport	ElGamal
Role of TGS(AS)	Symmetric Key Transport for User and Service Provider	Public Key Certificate Generation

5 Conclusion and Future works

We have proposed the CORBA based authentication model, and within that model we have presented flexible mechanisms to accommodate multiple domains. The goal of our system is to select resources for use by applications and securely coordinate execution of application in multiple domains, eliminate the need for the end-users to explicitly log on to each machine.

The authentication model that proposed in this paper is very critical in distributed computing, since it supports the protection of the high-level resources and the preservation of the security policies of the underlying resources that form the foundation of various domains, between the Kerberized domains and the Non-Kerberized domains. Using public-key cryptosystem we acquired the flexibility of key management and reliable session key generation between the Client and the Provider.

Research should be made on the efficient object system to support a distributed security mechanism, and offer a more elaborated security infrastructure. In addition, for a key management, a heterogeneous key distribution of session keys should be considered.

References

1. OMG, CORBA services: Common Object Security Specification v1.7(Draft), <ftp://ftp.omg.org/pub/docs/security/99-12-02.pdf>, 2000.
2. Object Management Group. CORBA/ IIOP 2.3.1 specification, <http://sisyphus.omg.org/technology/documents/corba2formal.htm>, 1999.
3. OMG Security Working Group, OMG White Paper on Security, OMG Document, No. 9, 1996.
4. Menezes, Van Oorschot, Vanstone, Handbook of Applied Cryptography, 2nd Ed., pp.570-577, 2000.
5. Joris Claessens, A Secure European System for Applications in a Multi-vendor Environment, <https://www.cosic.esat.kuleuven.ac.be/sesame/>, 2000.6. A. Alireza, U. Lang, M. Padelis, R. Schreiner, and M. Schumacher, "The Challenges of CORBA Security", Workshop of Sicherheit in Mediendaten, Springer, 2000.
7. DSTC, Public Key Infrastructure RFP, <ftp://ftp.omg.org/pub/docs/ec/99-12-03.pdf>, 2000.
8. Robert Orfali, Dan Harkey, Client/ Server Programming with JAVA and CORBA, John Wiley & Sons, 1997.
9. Andreas Vogel, Keith Duddy, Java Programming with CORBA, 2nd Ed., John Wiley & Sons, 1998.
10. Bob Blakley, CORBA Security: An Introduction to Safe Computing with Objects, Addison Wesley, 2000.
11. M. Humphrey, F. Knabe, A. Ferrari, A. Grimshaw, "Accountability and Control of Process Creation in the Legion Metasystem", Symposium on Network and Distributed System Security, IEEE, 2000.
12. A. Ferrari, F. Knabe, M. Humphrey, S. Chapin, and A. Grimshaw, "A Flexible Security System for Metacomputing Environments", High Performance Computing and Networking Europe, 1999.
13. John T. Kohl, B. Clifford Neuman, Theodore Y. Ts'o, "The Evolution of the Kerberos Authentication Service", EurOpen Conference, 1991.
14. Massachusetts Institute of Technology Kerberos Team, Kerberos 5 Release 1.0.5. <http://web.mit.edu/kerberos/www/>.
15. M. A. Sirbu, John Chung-I Chuang, "Distributed Authentication in Kerberos Using Public Key Cryptography", Symposium on Network and Distributed System Security, IEEE, 1997.
16. W. Diffie, M. E. Hellman, "New directions in cryptography", IEEE Transactions on Information Theory, Vol. 22, No. 6, 1976.
17. T. ElGamal, "A public-key cryptosystem and a signature scheme based on discrete logarithms", IEEE transactions on Information Theory, Vol. IT31, No. 4, 1985.
18. G. White and U. Pooch, "Problems with DCE Security Services", Computer Communication Review, Vol. 25, No. 5, 1995.
19. T. Parker, D. Pinkas, SESAME V4 Overview, SESAME Issue1, 1995.