A Security Structure in Distributed Systems

Ali Chodari Khosroshahi
Godrat Abolhasani Sahlan
Abolfazl T. Haghighat
Computer department
Azad University of Qazvin
Islamic Republic of Iran

***Abstract**: Security is a very important aspect of any computing systems, and has become a serious problem since companies and their infrastructure have become distributed in very heterogeneous fashion. In this paper we describe a new security structure for distributed systems. In this structure, security is distributed in the system level and all of the components have their distinct security responsible, which updating by security server. Therefore authorization is changed dynamically and total system polices handled by security server.*

***Key-word**: Geographical distributed systems, Security, Secure Sockets Layer, security responsible, security server.*

1 Introduction

Security is a very important aspect of any computing systems, and has become a serious problem since companies and their infrastructure have become distributed in very heterogeneous fashion. As software on which infrastructure relies has become ever more complex, interdependent and interconnected, company's reputation has in turn become more vulnerable. Companies that are faced with a choice between adding features and resolving security issues need to choose security. In any other case they will be faced with possibility of compromising sensitive data , loss of reputation and customer/collaborator confidence in other word faced with possibility of tremendous financial loss. So, it is necessary to build new systems incorporating security as integral part of their design. Instead, many companies and organizations try to avoid the definition of a security architecture and jump directly to ad-hoc testing against the security of their computing and networking infrastructures. In this paper first we review previous performed schemes and describe advantages and disadvantages of this schemes (chapter I), and in the next chapter (chapter II) described a new security architecture consists of combine if latter-day security schemes and foregone schemes (described in chapter I)

2 Review of previous performed schemes

2.1 Alchemy:

One exemplar system that comes in [2] is Alchemy and we describe this system in this paper briefly.

Under normal operation, a current user of Alchemy can, from a remote location, build, activate and control an application which is distributed over one or more computer clusters. Alchemy has also been designed to support pervasive environments, and thus, can also support processing architectures where the processors are not clustered. In general, Alchemy applications are poorly suited for many grid architectures which best handle the batch-like processing done on a supercomputer.

As depicted in figure 1, Alchemy uses its concept of nodes to support the distribution of processing elements across a set of computational resources. The encapsulated message handlers of these nodes are abstracted from their and other handlers' physical locations by a name service which: 1) maps client connections, server connections and handlers to nodes and 2) maps nodes to physical processors. Building on its name service's ability to detach physical node location and client-server port information from the logical representation of an application's connection graph, Alchemy supports dynamic modification of this graph during application execution. This allows an application to even control its own processing configuration. Process mobility is accomplished by simply moving a connection and its associated handler to a new node in the system. This allows Alchemy to more tightly manage its processing elements than most mobile agent and distributed object approaches while providing a better capacity to handle the dynamic nature of the Internet.

Alchemy uses a common communication interface for both system-level and application messages and supplies a three-level security method which includes communication channel authentication, encryption and message tracking. The lowest security level (level 1) uses a four-way authentication method to confirm that both the client and the server using the communication channel are in fact Alchemy components. At level 2, the encryption of all message content is added using randomly rotating symmetric session keys. At level 3, a message tracking mechanism (currently built on MD5) is added. All system-level support traffic (e.g., GUI to process control servers, server-to-server, etc.) is locked at level 3. The security level of all other traffic can be selected by defining the security level of each server during the application design. The location of the data in a level 1 and level 2/3 message is shifted so that one cannot easily gain all of the information about message construction by sending the same message both encrypted and in the clear.

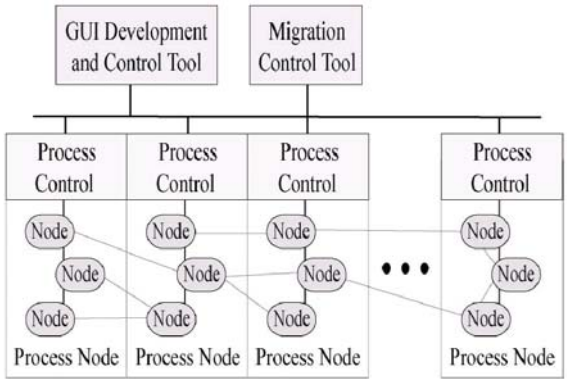


Fig.1 Alchemy's process architecture

Alchemy supports three different ways to encrypt the authentication handshake used by every client-server connection. The first method (called super-secret[1]) uses a stored 256-bit super secret key. This key is used by a blowfish algorithm to encode and decode all four handshake messages. While this super secret key is computed from an algorithm and a set of variables that are placed in the source code in such a way to confuse anyone disassembling the code, this method is viewed as totally insecure and is only provided for timing tests. The second method (called public[2]) uses two 1024-bit RSA public/private key sets, one for each direction, that are each used to encrypt the two messages going in that direction. This method requires two additional messages that allow the two sides to pass each other their public key. Since the server's public key is only used once per client connection and the client's key is only used once, this method is extremely secure, but also very computationally expensive. The public method is also only used for timing tests.

2.1.1 Dealing with Port-Controlled Access Points

we have explored four possible approaches to crossing a port-controlled boundary and their advantages and disadvantages. These approaches are presented in figure 2.

The most straightforward way for a GD application to get past a port-controlled access point is shown in figure 2a. In this approach a block of ports are opened on the firewall for use by the GD application. Since it is hard to predict the exact number of ports needed by different GD applications, this approach can open up a very large hole in the firewall, and thus, potentially compromise the firewall's ability to protect the external network.

A more secure approach to a GD system interface is shown in figure 2b. Here, GD application traffic going through the firewall is converted to some well-known protocol that is or will be allowed to pass. For example, if only a control or viewing interface is required, a web based interface can be used, since it is relatively easy to convince a network administrator to allow port 80 or 8080 traffic to pass either in or out of a controlled network.

The next most straightforward way for a GD application to get past a port-controlled access point is shown in figure 2c. Due to the number of firewalls being used today and the large number of individuals that need to access the internal data and applications of these controlled networks for work or school, secure methods for tunneling through a firewall are now common. Traditional tunneling approaches rely on a two (or more) sided application called a Virtual Private Network (VPN). VPNs are designed to connect a group of remote computers into a single virtual LAN which may give individuals who can gain access to the VPN at another site far more access to your internal network then you really wanted to give. After all, that is why the firewall is there in the first place.

The last approach, shown in figure 2d, is to build your own special purpose intelligent tunneling software that handles only the communication traffic generated by the distributed application. This gives the same port control as a VPN without completely opening up each LAN supporting the distributed application to general access by authorized users of the other LANs on which the application is running.

2.2 disadvantage of this approach

[1] Refer to appendix A
[2] Refer to appendix B

As depicted in figure, four connecting channel are used in this approaches and some of them have their advantages and disadvantages, but they all use firewall as well. The use of firewall has its disadvantages that we discuss them later.

This have made many of universities and commercial networks to develop their knowledge about selecting firewall which connects the network to internet block or restricts input ports connections to the computer nodes before firewall. This firewall blocks enables internal services to be connected outside of clients. If this port blocking policies for network construction were restricting to connect some of the users to the internet, users will have many problems for researching at geographical distributed systems.

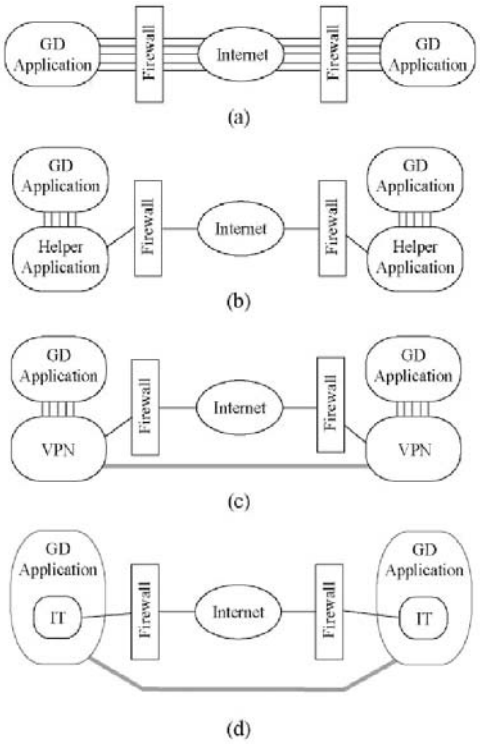


Fig.2 Different connection approaches (See details in text)

At the other hand, if firewall is configured correctly, many security problems will remain yet. If firewall is configured so that packets of a proper network could go over, for example, an attacker could pass this scanning by unauthorized address. If one of the internal members of network will want to export secured documents, he can encode it or convert it to a picture and send it in the JPEG file format that can pass from all filters that scan the context. Firewall is broken against Denial of Services and flood of authorized requests [3].

3 Designed security structure

With this knowledge in mined and advantages and disadvantages of these structures, we designed a security structure, as can be seen in figure 3. This structure has composed of three main segments: Client, Server and a Security Server.

They are connected to each other through network as is shown in the figure. When the client wants to access a method or data into the server, it sends a request to the corresponding server. But, before sending the request, client calls helper application to test the authorization of server for performing the request. If the server is authorized for performing the request, request is send toward the server, and if the server isn't authorized, the request is canceled and a exception is send to the client. With receiving the client request in server side, server calls helper application to authorization of client for request. If answer is positive responding to client, otherwise request is rejected and an exception is send to the client.

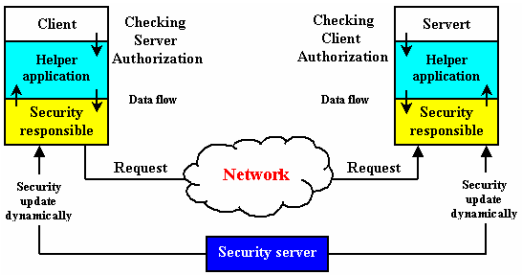


Fig. 3 Designed security structure

Alternatively, security protocol used at the responsible of security has the responsibility of sending and receiving the information in the network safety in order to avoid the use of information or to hearken.

The other segment of the system is security server. This security server handles the security segment of the systems, and apply codified security polices to each of members, and updates existing data bases in the security segment dynamically. This advantage makes system flexible and scalable, because security server acts dynamically in changing case or in necessary case of this system updates existing security information in the clients or servers.

This system is obtained from combining of the mentioned approaches at the 2-a, 2-b [2] and deleting firewalls of these approaches.

3.1 Responsible of security

This segment is responsible of security works on the system. For example this segment performs this works: authorization[3] , secured sending and receiving

[3] in this structure we have two mode of authentication :
a. Client or server authentication.
b. Authentication to perform or do request .

information. Behavior of authorization is told before, but at this part sending and receiving of secured information through the security protocols that are embedded at this segment.

3.1.1 Security protocol

At the previous approach as explored, for data and system security is used firewall. The use of this approach had disadvantages. We want to design a new approach based on the new security protocols, providing better system security. On the other hand , protocols that introduced at 1995 by Netscape company the designer of greatest web browser to response for security requirements . SSL establishes a safe connection between two sockets with the following properties:

- Possibility of session and agreement parameters between client and server.
- Obtaining the identity of client and server independently.
- Secret telecommunicate and encoding of data.
- Maintaining of safety and correctness of data.

In fact SSL is a new layer between application and transport layer and receives the request and sends it through TCP layer.

TLS that RFC 2440 provides, in fact manufactured by some changes in SSL .This product is introduced by Netscape and is standardized by IETF. First version of TLS is introduced at 1999.

SSL have a disadvantage that is limited in encryption code. Because USA identify a 40 bit security key for SSL to data encryption that seems a funny. In follow we describe SSL briefly.

3.1.1.1 The SSL Protocol

This document introduces the Secure Sockets Layer (SSL) protocol. Originally developed by Netscape, SSL has been universally accepted on the World Wide Web for authenticated and encrypted communication between clients and servers it also can be used by applications.

The SSL protocol runs above TCP/IP and below higher-level protocols such as HTTP or IMAP. It uses TCP/IP on behalf of the higher-level protocols, and in the process allows an SSL-enabled server to authenticate itself to an SSL-enabled client, allows the client to authenticate itself to the server, and allows both machines to establish an encrypted connection.

These capabilities address fundamental concerns about communication over the Internet and other TCP/IP networks:

- **SSL server authentication** allows a user to confirm a server's identity.
- **SSL client authentication** allows a server to confirm a user's identity.
- **An encrypted SSL connection** requires all information sent between a client and a server to be encrypted by the sending software and decrypted by the receiving software, thus providing a high degree of confidentiality.

The SSL protocol includes two sub-protocols: the SSL record protocol and the SSL handshake protocol. The SSL record protocol defines the format used to transmit data. The SSL handshake protocol involves using the SSL record protocol to exchange a series of messages between an SSL-enabled server and an SSL-enabled client when they first establish an SSL connection.

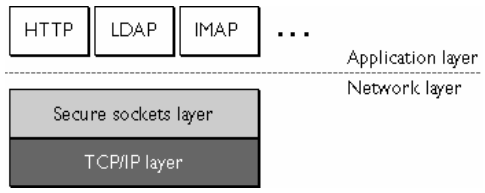


Fig.4 SSL runs above TCP/IP and below high-level application protocols

To understand how a Netscape server authenticates a client certificate, refer to appendix C.

3.2 Helper application

This helper is really a plug-in application that we could add on operating system. We can provide this helper application for deferent applications. This is the program that enhances capability of a large program and the interface layer between client or server application and security protocol. This helper application is transparent to the user.

Client helper application authenticates the server to perform the requests, and at the opposite direction server helper application authenticates the client to do request. If this request can not be performed an exception sends to the client and helper application acknowledges the user to no performable the request.

.

3.3 Advantages of this approach

This approach has several advantages compare with the mentioned approaches:

- Flexible
- Scalable
- Access control on application
- Distributed security
- Dynamically authorization

In this approach adding the Security Server caused that this structure could have Flexibility, Scalability, distributed security and dynamically authorization because the security server dynamically updates the security database with changes in distributed system or system polices.

System crashing in this approach not caused any problem for the system, and only case that the system loss dynamical state.

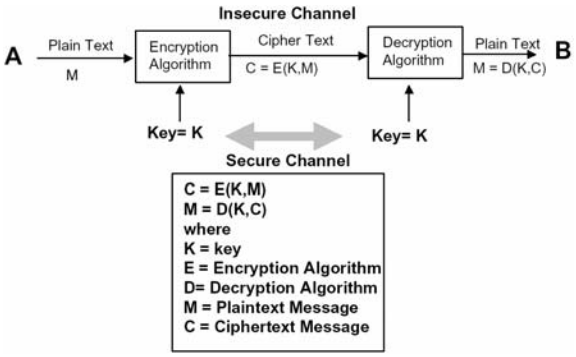
4 Conclusion

We designed this security structure to use the capabilities of new security protocols such as SSL and TLS. This structure is composed of this protocols and available structures in figure 2 and have those capabilities and above additional advantages.

As depicted in figure, there is no restricting member in the system. So, this system is flexible and scalable. Application accesses are controlled in the system also. Security is distributed in the system level and all of the components have their distinct security responsible, which updating by security server .Therefore authorization is changed dynamically and total system polices handled by security server.

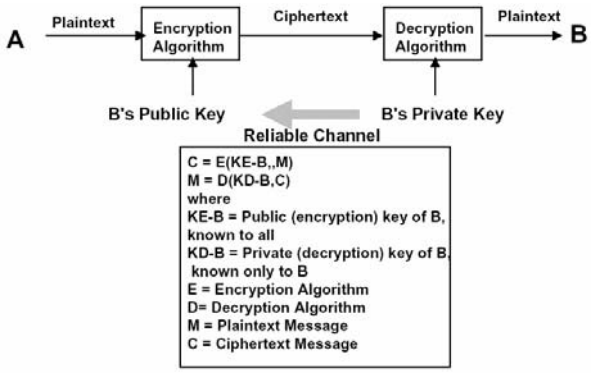
Appendix A :

Secret key cryptosystem



Appendix B :

Public key cryptosystem

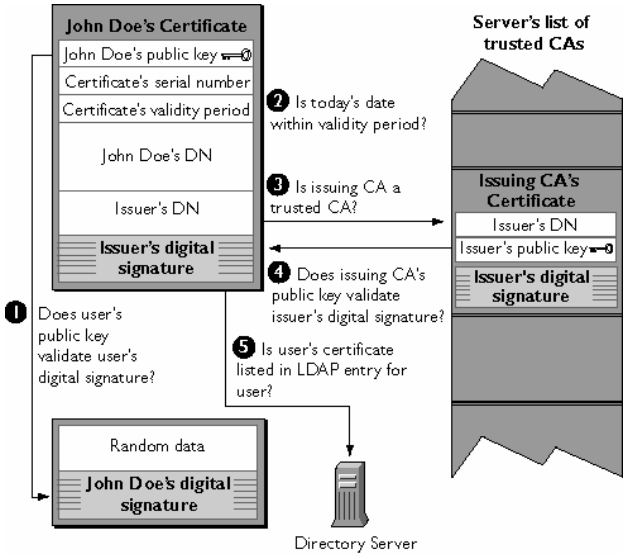


Security is based on infeasibility of computing B's private key, given the knowledge of

- B's public key
- Chosen plaintext
- Chosen ciphertext

Appendix C :

How a Netscape server authenticates a client certificate



References

[1]. Mobile Agent-Based Security Model for Distributed System
Zhang Jun-yan, Li Yi-chao , Min Fan , Yang Guo-wie
2001 IEEE International Conference

[2]. Addressing Security Issues in Geographically Distributed Systems
Charles Hannon and J. Richard Rinewalt ,Computer Science Department Texas Christian University
Fort Worth, TX 76129 , {C.Hannon, D.Rinewalt}@tcu.edu
Fourth Mexican International Conference on Computer Science , September 08 - 12, 2003 IEEE

[3]. Computer Networks (four edition – 2003)
Tanenbam , Andrews
ISBN 964-5801-84-2

[4]. Security Architecture of the Distributed System - Layered Approach
Milo. M. Cvetanović, Zaharije R. Radivojević
University of Belgrade,
Manuscript received May 9, 2003.

[5]. A Security Architecture for Object-Based Distributed Systems
Bogdan Popescu , Maarten van Steen , Andrew S. Tanenbaum
Vrije Universiteit, Amsterdam , (bpopescu,steen,ast}@cs.vu.nl
Applications Conference December 09 - 13, 2002 San Diego California

[6]. Secure and Fault-Tolerant Voting in Distributed Systems
Ben Hardekopf, Kevin Kwiat, Shambhu Upadhyaya , Air Force Research Laboratory
IEEE Aerospace Conference, Big Sky Montana, March 2001

[7]. Analysis of the SSL 3.0 protocol
David Wagner , University of California, Berkeley , daw@cs.berkeley.edu
Bruce Schneier , Counterpane Systems , schneier@counterpane.com
The Second USENIX Workshop on Electronic Commerce Proceedings , USENIX Press
Revised April 15, 1997