Distributed Systems and Computer Security

Roberto Gómez
ITESM-CEM
rogomez@campus.cem.itesm.mx

Abstract

Computer security is a very important topic for many institutions. Distributed systems have existed since a long time ago. Computer security was used in order to enforce distributed systems, however now distributed systems can help to increase computer security in computer systems. This paper presents several approaches in which distributed systems increase the security of important systems.

Key-words: distributed systems, distributed algorithms, computer security.

1 Introduction

In 1999, (see [1]) the cost of cyberattacks to US businesses doubled to $10 billion according to estimates from the Computer Security Institute (CSI). Computer security is defined as the ability to manage, protect and distribute sensitive information. There are tree principal aspects related with security: confidentiality, integrity and availability. The first one, confidentiality is the nonoccurrence of the unauthorized disclosure of information. Integrity is the degree to which a system or component prevents unauthorized access to, or modification of computer programs or data. Availability is the degree to which a system is operational and accessible when required for use.

By the other hand, Couloris, [2] defines a distributed system as a collection of autonomous computers linked by a network, with software is designed to provide a n integrated computer facility. Distributed systems are a popular and powerful computing paradigm. Their importance increases as networked computers become more common than freestanding ones, especially since many different kinds of computers can be found in networks. There are many examples of these kinds of systems, as clusters LAN networks, etc.

Resource sharing is the principal characteristic of distributed systems (see [2, 3]). Openness implies that interfaces must be accessed and published to facilitate interconnection between different components. Several processes to exist in the system at the same time, each one executing its own code or all of them executing the same code, this is kwon as concurrency. The scalability allows incorporating and eliminating elements of the distributed system. Computer systems sometimes fail, fault tolerant grants the availability of the system although one of the components crashes. Transparency is defined as the concealment from the user of the separation of the components, so that the system is perceived as a whole rather than as collection of independent components.

Every time a distributed system was designed and implemented, different aspects as scalability, transparence, and others were taken in a count. However, computer security was not considered or considered as a minor aspect of system design. Designers integrated computer security, as it was needed; the distributed system had to be adapted to accomplish security requirements.

Nowadays we can not consider computer security as a reality. Most of the security systems present failures which attackers use to break computer systems. Most of the security systems need to be enforced in one way or another. Distributing techniques can be used to provide this enforcement. So security systems need to be adapted to implement distributed security. In this paper we present different approaches that uses distributed principles in order to enforce security in computer systems.

The paper is organized as follows. Next section presents distributed sniffers. Section three gives a short explanation of the most important issues in distributed cryptography. Section four discusses the design of secure distributed algorithms. Section five gives an overview of computer immunology. In section six we give a description of what has been called distributed attacks. The last section summarizes the presentation and shows the conclusions.

2 Distributed Sniffers

A sniffer is a program that monitors and analyzes network traffic, detecting bottlenecks and problems. It can be used legitimately or illegitimately to capture data being transmitted on a network. A network router reads every packet or data passed to it, determining whether it is intended for a destination within the routers own network or whether it should be passed further along the internet. A router with a sniffer, however, may be able to read the data in the packet as well as the source and destination addresses.

One of the simplest approaches to assures security is to install sniffers over the entire network and look to the traffic that is circulating on the systems. If any stranger information is observed then appropriate actions must be taken. The problem is to define what is a stranger information and which are appropriate actions. Confussion in information detection can cause that administrators take a bad decision, and some of the "legal" users can be affected in their normal operations.

One variant of distributed variants is what Bellovin calls distributed firewalls (see [12]). Conventional firewalls rely on the notions of restricted topology and control entry points to function. The Bellovin basic idea is simple. His scheme rests on three notions, a policy language, a number of system management tools and the network-level encryption mechanism for TCP/IP. A compiler translates the policy language in some internal format. The system management distributes this policy to all host protected by the firewall. Incoming packets are accepted or rejected by each inside host, according to the policy and the cryptographically verified identity of each sender.

Many distributed aspects have to be considered on the implementation of both systems. First of all, it is important to decide what kind of communication protocol will be used. The system must provide fault tolerant in case one the sniffers crash. The system must be transparent to users, so they can operate as normal, and to strangers, so they can not take any action against sniffers. Information consistency is very important, algorithms as [13] must be included to assure this consistency. It is important that information circulating between sniffers is encrypted, in such a way that only they can see and understand it (as the Bellowin distributed firewalls).

3 Distributed Cryptography

One of the fundamental problems in computer security is confidentiality, the transmission of a message M from a sender A to a receiver B, over an insecure communication channel such that an enemy (E) with access to this channel is unable to obtain useful information about M.

As defined in [3], cryptography is the art if storing information in a form that allows it to be revealed only to a particular group yet hides it from all the others. A cryptosystem is a method to accomplish this. Cryptoanalysis is the practice of defeating such attempts to hide information. Cryptology includes both cryptography and cryptoanalysis. The original information to be hidden is called plaintext. The hidden information is called ciphertext. Encryption is any procedure to convert plaintext into ciphertext. Decryption is any procedure to convert ciphertext into plaintext.

Plaintext is converted into ciphertext by means of an encryption engine (generally a computer program, whose operation is fixed) and a piece of information, called the encryption key. Decryption must be accomplished only under certain conditions, which means persons in possession of both a decryption engine and a decryption key. The encryption key and the decryption key may or not may be the same. When they are the cryptosystem is called a "symmetric key system" (or secret key system); when they are not it is called an "asymmetric key" (or public key system).

In both systems, security resides in the key. Therefore, it is necessary to keep the key in a safe place to avoid accidental and malicious exposure. This scheme is not 100the key and use it to decrypt the information. So it is important that key bee safe, one way to accomplish this is to divide the key in different parts. The secret sharing scheme allows us to divide the key.

Shamir's scheme [4] for sharing secrets breaks a secret string S into n pieces (S_1, S_2, S_3 ... S_n) each of the same size. Knowledge of any k or more S_i pieces makes S easily computable, but knowledge of any $k-1$ of fewer S_i pieces does not allow a construction of secret S. Such a scheme is called a (k, n) threshold scheme.

Each participant keeps a piece of the key (secret) and only the presence of k participants will allow creating the key to encrypt and decrypt information. This scheme is ideally suited for applications in which a group of mutually suspicious individuals with conflicting interests must cooperate. By properly choosing the k and n parameters we can give any sufficiently large majority the authority to take some action while giving any sufficiently large minority the power to block it.

There are many works related to threshold scheme. In [15] the distribution key problem is discussed and a secure group-key distribution protocol is presented. In another paper Maurer, [16] shows how cryptographic coding and error control coding can be combined to obtain a system achieving virtually perfect secrecy. In [17] the authors analyze the entropies of the sample spaces from which the shares and the secret are chosen.

4 Security of distributed algorithms.

Algorithms, in general, can be defined as black boxes that transforms inputs that are given to them at the beginning into desired inputs that are produced at the end. Distributed algorithms run in processes of systems where the processing elements are physically separated and they are rather to guarantee behavior of the system. Peoples concern when thinking of a distributed algorithm is not only what each component should do in order to guarantee the desired behavior of the system but also a big deal of effort goes to another issues. Between the most important issues they have to deal with the way the process can organize, remember, change and access collections of data, how to predict the resources that the algorithm requires, etc.

Research and work in distributed algorithms is concerned with:

- how to develop methods that another designer can employ in order to find a solution to another problem;
- to prove that nobody ever can find a better algorithm or to prove that although you did not find a solution anybody ever will find one.

These distributed algorithms will serve to implement systems that operate on a benign environment. There are always assumptions about message contents, integrity, channels, sequences, and so on.

These assumptions allow us to have a good idea of what is really happening in our system behavior. However, in the real world almost of these assumptions may be invalidated.

As an example, it has never seen an algorithm including malicious processes. A malicious process may be someone (human or process) working on a network and taking some advantages because it knows something that enables it to gain privileges or access to some computer resource. Try to imagine that:

- In a dinner philosopher solution, a philosopher never releases one of his forks.
- A process performs infinite tasks inside a critical zone, and does not allow the entrance to another processes.
- What happen if one machine on a token ring network never releases his token to enable the access for others devices.

Such considerations must be integrated or at least considered when designers work on the development of new distributed algorithms. There is no construction method that takes into account security aspects.

5 Computer immunology

One of the most usual practices in computer science is to emulate the human behavior to solve some problems. Artificial intelligence, expert systems, neural networks and others kind of systems are examples of this. So why do not follow the same approach with the human immunology system? The immune system defends the body against harmful diseases and infections. It is capable of recognizing virtually any foreign molecule and eliminating it from the body. To do this, it must perform patter recognition tasks to distinguish molecules and cells of the body (called self by [5]) from foreign ones (called "nonself"). The architecture of the immune system is multilayered, with defenses provided at many levels. The outermost layer, the skin, is the first barrier to infection. Once pathogens have been entered the body, they are handled by the innate immune system and by the adaptive immune response. The first one consists primarily of circulating scavenger cells than ingest extracellular molecules and materials, clearing the system of both debris and pathogens. The adaptive immune system is the most sophisticated and involves many different types of cells and molecules (f.e. lymphocytes). It is also called adaptive, because it is responsible for immunity that is adaptively acquired during the lifetime of the organism.

The principal objective in this direction is to design and implement computer security architectures that incorporate the characteristics presented in the previous paragraph. One approach to accomplish this is to design systems based on direct mappings between immune system components and computer system architectures.

A natural place to start is computer viruses, the protection problem is essentially the same as that of protecting any kind of stored. In this case data-self is interpreted as uncorrupted data and nonself is interpreted as any change to self (see [6] and [7]). To protect active processes on a host, every active process in a computer is treated as a cell. In this approach self would be defined by normal behavior and nonself would be abnormal behavior in the form of intrusions. In [8] Forrest at al., examine approximate matching of short sequences of system calls as a candidate for distinguish normal and abnormal behavior.

The two last examples are for only for single. It is most interesting to distribute the examples and include a set of hosts. For protect a network of mutually trusting computers, we have to think of each computer as corresponding to an organ in an animal. The innate immune system is composed of host-based security mechanisms such as Kerberos (see [9]) and firewalls. Kernel-assisted lymphocyte processes, with the added feature that these processes could migrate between computers, making them mobile agents can implement the adaptive immune system layer. Protecting a network of mutually trusting disposable computers requires regarding each host as a cell, with a network of mutually trusting computers being the individual. In this case the innate immune system consists of the network's defenses, such as Kerberos and firewalls. The adaptive immune systems can be implemented by creating a set of lymphocyte machines. These machines would monitor the state of other machines on the network.

Researches at the University of Oslo in Norway propose a similar approach. In [10] the cfengine autonomous model for system administration is presented, this model is based in an object-oriented language that allows defining the tasks of human administrators. The cfengine responds to changes in its environment and modifies its program accordingly. In order to implement an immune system, a cfengine program is defined in such a way that it analyses the system at regular times and activities the correct responses. The problem is the system degradation and the confrontation to new situations, not defined in the cengine language. In [11] the author focus its attention to the intrusion detection problem. The strategy is to build up a database of normal behavior for each program of interest. There are two strategies to the proposed system. In the first one, the system scans traces of normal behavior and build up the database. In the second stage, the system scans traces that might contain abnormal behavior and it matches the trace against the patterns stored in the database. Actions are taken depending in the matching result.

In both approaches distributed algorithms are needed to communicate and synchronize the elements of the system. A system agent approach could be very useful to implement these systems.

6 Distributed Attacks

In this section we translate to the other side of the river. Until now, we have showed how distributed systems contribute to implement distributed security. However, distributed techniques can be also used to design an attack. This attack belongs to a class of attack kwon as denied of service because, once the attack has been launched, the victim can not provide the service for which it was originally designed. As it uses a distributed paradigm it is called a Distributed Denied of Service (DDoS).

In march several computer sites (Yahoo, Buy.com, eBay, CNN.com, Amazon.com, ZDNet.com, E*Trade, and others), were shut down by a torrent of bits sent by hostile invaders. The attackers controlled a small number of masters that in turn controlled a larger number of daemons. These daemons were used to launch packet flooding or other attacks against victims targeted by the intruder. The attack may overwhelm victim site making identification of all "daemon" systems difficult.

Therefore a single, simple command from an attacker results in tens of thousands of concurrent attacks on one or a set of targets. The attacker uses unprotected Internet sites around the world to coordinate the attacks. Each attacking site has limited information on who is initiating the attack and form where; and no node need have a list of all attacking systems. For the victim, the impact can be extensive. For example, in a denial-of-service attack using distributed technology, the attacked systems observes simultaneous attacks from all the nodes at once flooding the network normally used to communicate and trace the attacks and preventing any legitimate traffic traversing the network.

The attacks were based in two automatics tools trine and Tribe Flood Network. It is not of our concern to explain these tools, the important issue is the distributed point of view of the attack. The attackers have to be concerned with different distributed concepts, as process communication and synchronization. As we can see, distributed techniques could be used to build an attack instead of increase the system security.

7 Conclusions

In this paper we showed how distributed systems contribute to integrate security in computer systems. Security covers different aspects as integrity, availability and confidentiality, different approaches contribute to each of these aspects. Integrity is assured by distributed sniffers and computer immunology. Availability is granted with computer immunology and secure algorithms. Confidentiality is obtained with computer immunology and distributed cryptography. Also, we showed that distributed attacks could be designed applying distributed concepts. A new point of view has replaced the old conception of how security enforces distributed systems, now we can say that distributed systems enforces system security.

The approaches presented in these papers no are the only ones there are other projects concerned with distributed security. As example several works is done in wireless networks and in high-speed networks as ATM or FDDI.

References

[1] Piller, Cahrles Loss From Hack Attacks Doubles to $10 Billion, Los Angeles Times, 03/22/2000,

[2] Couloris, Dollimore and Kindberg, Distributed Computing Concepts and Design, Addison Wesley, 1994, second edition

[3] Mullender, Distributed Systems, Addison Wesley, 1993, second edition

[4] B. Scheneider, Criptography

[5] Shamir A, How to share a secret, Comm. ACM Vol. 22, No. 11 (Nov 1979) pp. 612-613

[6] Anil Somayaji, Steven Hofmey & Stephanic Forrest, Principles of Computer Immune System, University of New Mexico

[7] S. Forrest, A. S. Perelson, L. Allen and R. Cherukuri. Self-non discrimination in a computer. In Proceedings of the 1994 IEEE Symposium on Research in Security and Privacy, Los Alamos, CA, 1994, IEEE Computer Society Press

[8] J.O. Kephart A biologically inspired immune system for computers. In R.A. Brooks and P. Maes, editors Artificial Life IV: Proceedings of the Fourth International Workshop on the Synthesis and Simulation of Living Systems, pp. 130-139, Cambridge, MA, 1994. MIT Press.

[9] S. Forrest, S Hofmery, A. Simayaji, and T. Longstaff. A sense of self for Unix processes. In Proceedings of the 1996 IEEE Symposium on Computer Security and Privacy, IEEE Press, 1996

[10] B.C. Neumman and T. Tso. Kerberos: An authentication service for computer networks. IEEE Communications Magazine, 32(9), pp. 33-38, September 1994.

[11] Mark Burgess, Computer Immune Systems, `http://www.iu.hioslo.no/~mark/research/immune.html`

[12] Mark Burgess, Computer Immunology, http://www.iu.hioslo.no/~mark/research/Aldrift/Aldrift.html

[13] Bellovin M Steven, Distributed Firewalls, login, November 1999, pp. 37-39

[14] Lamport, Shostak and Peases, The Byzantine Generals Problem, ACM Transactions on Programming Languajes and Systems, Vol. 4 No. 3 Julio 1982

[15] Mayer, Yung Generalized Secret Sharing and Group-Key Distribution using short keys, 1998 IEEE publication 0-8186-8132-2/98

[16] Maurer Ueli, Secret key Agreement by public discussion from common information, Transactions on Information Theory, Vol. 39, pp. 733-742, 1993

[17] Capocelli, De Santis, L Gargabi, Vaccaro, On the size of secret sharing schemes