A security model for distributed computing

Article · November 2001

CITATIONS
7

READS
4,647

2 authors:

Iliya Georgiev
Metropolitan State University of Denver
21 PUBLICATIONS **31** CITATIONS
SEE PROFILE

Ivo Georgiev
Metropolitan State University of Denver
13 PUBLICATIONS **25** CITATIONS
SEE PROFILE

A SECURITY MODEL FOR DISTRIBUTED COMPUTING

Iliya K. Georgiev
Department of Mathematical and Computer Sciences
Metropolitan State College of Denver
Denver, CO 80217-3362
(303) 556 5323, e-mail: gueorgil@mscd.edu

Ivo I. Georgiev
Data Digest Corporation
10 El Camino Real, Suite 200
San Carlos, CA 94070
(650) 620-3887, e-mail: igeorgiev@data-digest.com

Abstract

This paper presents a multi-tier model for secure computing as a teaching method platform. The security model is based on establishing the trustworthiness and role of each component in a distributed computing environment: trusted users, trusted servers, trusted administrators, untrusted client, untrusted communication media and intermediate systems, etc.
The model provides a basis for teaching and for program system design. The security dimensions (both social and technical) can be considered in computer science curriculum in general. The model as a teaching method was experimented in some senior student software projects.

INTRODUCTION

Security threats in distributed computer systems can be divided into two categories. The first category is comprised of threats, which exist in centralized systems but are amplified by distributed systems. These threats are allowed by the inherent vulnerability of system components and can be countered by addressing the vulnerability independently from system interconnections. The second category is comprised of threats that are specific to distributed systems and do not appear in centralized systems. The technical challenge posed by distributed system security is amplified by requirements of system scalability, interoperability of centralized and non-centralized system units, and interconnection of both trusted and untrusted nodes. Interconnected computers may support different operating systems, application suites, and potentially divergent system security policies. All these issues make it difficult to determine what global system security policies the distributed environment can support. Some existing security policies and services can be redefined for use in distributed systems, but they mostly should be designed anew without regard to leaks and weaknesses at the node machines, which can be addressed independently.

Securing a distributed computing environment against malicious or otherwise disruptive use has two aspects: social, where the safeguarding a computer system relies on social deterrents, shameful exposure or prosecution; and technical, where the system is protected by technical means, encryption algorithms and access control. Computer system security education should also have two mutually inextricable target objectives: attitude: practice + habit; and theory: technology + techniques.

With these remarks one can imagine how difficult is to organize teaching security of contemporarily global computing. The social element of the education should aim to increase awareness of security-related issues in the use and design of distributed computing. The following are deemed the most significant: types of security threat objectives (e.g. destruction, theft, engagement, weakening, accidental, etc.); threat psychology; approach, locality, and scale of attacks. The technical element of the education should train students in the proper computer

security technology and techniques in a consistent and coherent framework. Layering security functions and mechanisms on all system levels often alleviates the security.

This paper tries to address computer system security challenges in a systematic way by utilizing a basic model, which simplify the analysis whether security requirements (both technical and social) are satisfied.

COMMON SECURITY MODEL

The presented model (Figure 1) is a simplification and generalization of other existing distributed computing models. It is optimized to provide interoperability of the security services defined by such industry standards like the ISO 7498-2 Security Architecture and the ANSI/ISO cryptographic standards [12]: identification and authentication; access control; confidentiality; non-repudiation; and availability. The model is based on the establishment of the degree of trustworthiness of each system component. Some of the model's components can be implemented utilizing the embedded security facilities of the networked systems, while others have to be developed from scratch.

Untrusted partners. Personal workstations are inherently untrustworthy, in the physical sense, since they can be easily moved, connected to arbitrary network addresses, or run an unknown operating system that can access the network. The same observation applies to some servers. System Area Networks (SAN) and mainframes could be also considered in this category.

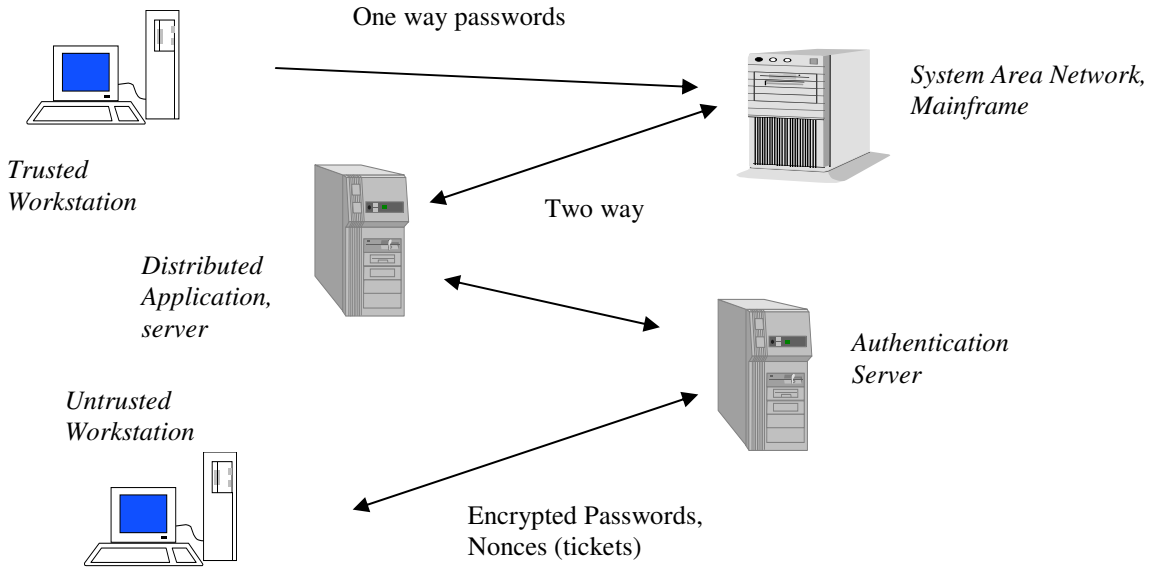

Figure 1. Common Security Model

Untrusted communication media. Communication media (twisted copper pair, coaxial cables, satellite links, etc.) used to connect client and server machines can generally be assumed to be untrustworthy. Measures can be taken to detect some attacks at this level, such as wiretapping, but they cannot be prevented.

Untrusted intermediate systems. Gateways, routers, and network front ends can fall into this category. A single security request, such as authentication, may have to be sent across multiple domains to reach the required service. The service has no control over the intermediate nodes and can make no assumptions about their physical security.

Untrusted clients. Client software is a subset of and runs on behalf of application software. It is untrusted because one cannot rely on it to make only the legitimate authorized requests on behalf of the application. Unless physical control of a client workstation is assured and cryptographic signatures are used during the OS booting process, one cannot determine whether the client workstation runs a trusted OS.

Trusted user/client identity. When an application and/or client operate, they do so under a unique identity. This identity is usually associated with the human operator at the workstation. The distributed system provides a way of establishing this trusted identity, typically by a trusted authentication function.

Trusted server identity. Trust must be established with servers in order to guarantee that they correctly perform the requested function. The trust established with a server must be two-way. The server must trust the identity of the client, and the client must trust the identity of the server.

Trusted administration (Security Server). Administration of the distributed system assumes trusted operation. The administrative applications, the systems on which they run, and the network connection to the servers must be trusted in order to maintain the integrity of the security-relevant information maintained at each server (access control and audit information). In some cases the administration could be the duty of a separate authentication server that deploys trusted third-party authentication. The trusted third-party Kerberos approach [12] is to have the client workstation send only the user ID to the authentication server. The authentication server then prepares a "ticket," which is an encrypted package containing, among other things, the user ID. The ticket is encrypted with a derivative of the user's password and is usable by the client workstation only when decrypted by applying the user's password as the key. If the user supplies the correct password, the ticket can be decrypted and the user ID can be compared for equality. A successful match of the user ID completes the authentication cycle. The client can then proceed to make requests of system services.

Some components of a distributed system cannot consistently be assigned but rather migrate between categories depending on their trustworthiness but also on the overall policy and sensitivity of proper trust assignment. An extreme example can be a system area network (SAN), a mainframe, or even a database server. Any of them can be assigned to the security administrator category, at the highest level of trust, or to the untrusted partner category, at the lowest level.

In currently existing systems, security services are currently provided on two levels, operating system and application. Companies like IBM, Tandem, Netscape, and Microsoft offer complete suites of security services in one or both of these categories.

Operating system (OS) driven security services are integrated into the operating system running on the distributed system node machines. They run in kernel mode and are therefore executed at maximum speed for any service. Kernel level is also the hardest to attack that makes this level ideal for implementing security policies. The drawback is that OS-level security policies are often very generic and cannot be modified to accommodate any particular distributed system scenario.

Application driven security services can be used by the application developer to create their own security subsystem. The advantage of this approach is that the security services are an integral part of the distributed system. The drawbacks are duplication of kernel-mode operating system

security, significant processing speed reduction, and increased vulnerability to expert cryptanalytic attacks.

Application-level security allows for the development for services specifically for distributed system environments like business application systems, design systems, process control systems, etc. It also allows the secure integration of existing disconnected systems (e.g. legacy databases).

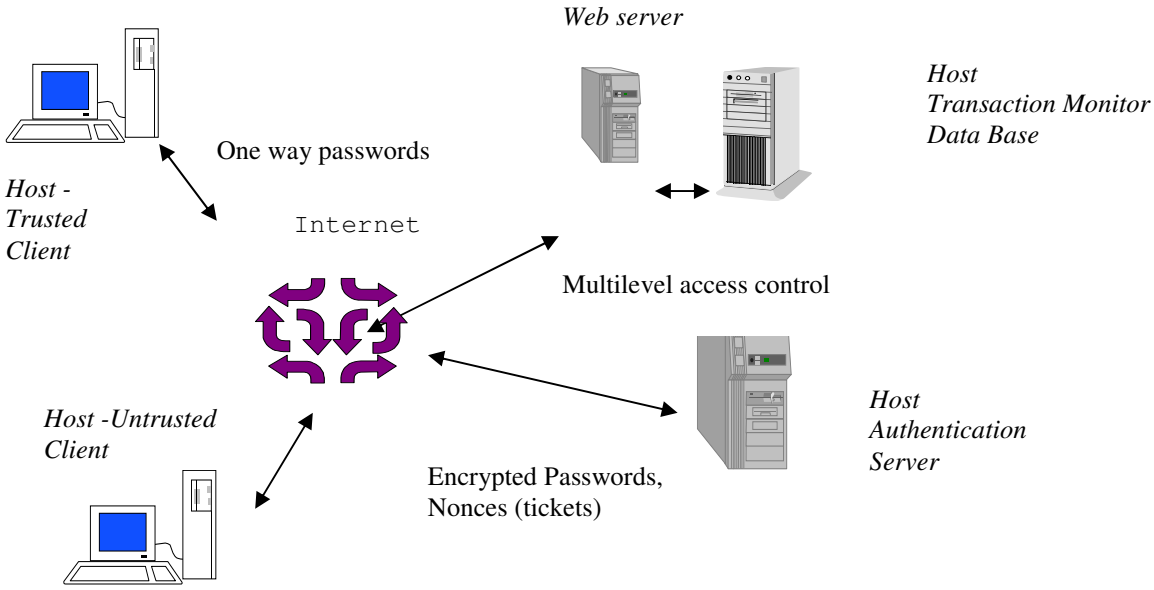

Figure 2. Security Model of Web-based environments

SECURITY ISSUES IN A WEB AND WIRELESS ENVIRONMENT

Web-based applications are being deployed rapidly over the Internet and were envisioned as an open environment for free information exchange. However, the design that fit these requirements has proven treacherous for online business and other more sensitive applications. Web browsers collect user profiles, often without soliciting explicit user consent, raising serious privacy and security concerns. Executable services (such as Java applets, ActiveX controls, etc.) are a constant vulnerable point as they can completely expose user's machines to malicious attacks. The scripting approach for information processing (CGI, Hypertext, XML, etc.) on which the WWW is based requires function-blind interpretation and is intrinsically very insecure. Software agents, which are essential in the efficient use of Internet, can also introduce serious security threats.

The common computer security model can be successfully applied to analyze security issues in the wireless network environment. The trust roles of the system components change significantly. The first change is that all components are considered Internet hosts. They can be trusted or not only according to the implemented security architecture Further in our teaching example we discuss some aspects of model usage.

- The wireless distributed system has some inherent problems. Web-enabled handheld devices, which can roam and connect to different points of attachment, form dynamic ad-hoc networks outside the control of the trusted server [2]. The network becomes untrusted, information can be corrupted, and service can be denied without feedback to the application [9].

Most security protocols rely on authentication and exchange of digital certificates only at the time of connection establishment, which becomes a major vulnerability in a wireless network. Attacks can be organized in the middle of a connection, modifying information, and redirecting communication to or through a malicious attack site. The Wireless Transport Security Layer (WTLS) provides cryptography-based security services, but in an ad-hoc network information often has to be encrypted and decrypted several times from one protocol to another, allowing man-in-middle attacks. The attacker can concentrate on handheld clients and wireless adapters, which are embedded systems with weak protected file structures and memory. Most functionality is incorporated in WML script (equivalent of Java script) and WML (mixture of XML and HTML) data structures. Script interpreters cannot differentiate between trusted firmware and possible untrusted script downloaded from the Internet. In the wireless distributed environment, communication channels have to be declared untrustworthy.\

-

SECURITY MODEL USED IN SOFTWARE DEVELOPMENT

The biggest issue in distributed system software development is to learn how to build hack-resistant and security-critical systems. It is obvious that the security of a complex system cannot be assured through any simple recipe [11]. The only sure approach is to split off the security-sensitive pieces into small audible modules and to create an attack-tolerant system from unreliable components. The proposed security model might appear too abstract and general guide for actual programming, but it is certainly very helpful in the evaluation of different approaches to increase the attack-resistance of a complicated software system. However, the results of our work on the application of the model in software development are beyond the scope of this paper.

SECURITY MODEL AS A TEACHING PLATFORM

The idea to create a security model with higher degree of abstraction came in a free discussion with a group of graduate students. They were preparing their masters theses in the context of a real world research and development project. The student theses were related to more research-oriented problems such as network monitoring (both event and status), and database reflection and security.

The object of the project was to develop a suite of software components for a complex three-tier client-server configuration: client workstations under Windows NT, transaction monitors and gateways under Unix, and database servers under Solaris and AIX, non-stop computers under Tandem Himalayas, and under IBM MVS. The main functionality of such system has been as follows: the clients requests for services, which are transformed in complex conditional transactions (some of them deferred, if some condition has been excepted after some period of time); the second tier (gateway and/or transaction monitor) splits the complex transaction into simple queries to different heterogeneous databases and makes request for data objects; the data objects are delivered to the clients keeping a registry of all transaction history.

Because of the diversity of the system, problems of security and especially authentication were the most difficult to solve consistently. For example, access control in IBM mainframes is implemented in RACF, while client workstations use MS domain authentication strategy, and neither system had been designed to integrate seamlessly with the other in a larger distributed environment. Such a random, but very possible in the real world, configuration is all the more vulnerable to man-in-the-middle and cut-and-paste attacks that seek to break into the crypt-stream or to alter without detection message marshalling from tier to tier. After several unsuccessful attempts at security architecture a simple security model was extracted from the security standards and based on the trustworthiness of components. The utilization of the model was very successful, especially in scenarios with a variety of shortcomings. Some explanation of the security model usage for the above-mentioned distributed configuration follows.

- Web-based application architecture can be represented as two-tier (Web Client and Web Server), three-tier (Web Client, Web Server, and Transaction Monitors and Databases), peer-to-

peer messaging, etc. Let's consider a typical three-tier security architecture (Figure 2) in the above distributed environment. A suite of components (clients, web server, application server, remote databases, and communication middleware) presents their services via interfaces. The standard security services are provided: authentication, access control, certificates, etc. Public-key encryption provides information confidentiality. Digital signatures are used for authentication, data integrity, and non-repudiation. The security system components work together in sessions involving user logon and invocation of remote objects on a system with different levels of security. The main task has been to choose proper security roles for the different architectural units.

- For *security administration* the important information was concentrated on the application server, transaction monitor and data bases. It was valuable to organize the security administration under the application server operating system. The transaction monitor was the natural candidate for flexible security administration. It was a discussion also possibility to organize the administration on a trusted client site.

Third party authentication service appeared one of the most controversial and challenging issues. The issue was to decide the place of the trusted center for a Kerberos-like key management architecture. Again, if the important information is concentrated on the application server this service should be organized under the server's operating system with support for nonce and trusted key storage. It is strongly recommended by some standards to use a separate computer located in a physically secure location. At first, none would have thought of choosing the second tier (the gateway) to be the trusted center. But it was the natural result in the context of the security model and it worked perfectly. This solution was dominated by the possibility for physical isolation of the gateway and completely new design of its software for implementation of deferred transactions.

The analysis stated that all distributed application servers and database servers should be *trusted servers.* This way, the basic security services of two-way identification, certificates, and message address and content certification could be provided from the most secure nodes of the network.

Trusted partners were distributed across the network, where different parts are awarded different levels of trust. The application can be distributed over several machines running the same operating system with different evaluation for trustworthiness. Often, the evaluation cannot be performed altogether. The trust boundary between partners was affected by the trust evaluation of the environment in which they run.

Any partner (machine, client, server, application) can be declared *untrusted* depending on the application. In our mission-critical applications all partners except the security administrator and third-party authentication service are declared untrusted. Application driven security services were foreseen to achieve attack-tolerant transaction.

The use of system security should be introduced early in the curriculum and the security model is a good educational platform for this purpose. The learning curve in the area of computer security is steep and security courses are usually upper level courses, focusing mostly on cryptography. In operating systems courses, where trustworthiness issues expose real-life problems for which there is no simple answer, the abstract role-based model would stimulate students to approach these problems in the context of their proper functional solution. In networking courses: networks, security, and privacy are always bundled together but rarely in a manner conducive to the learning of successful design practices and programming techniques. The distributed system model would serve as a framework for the integration of TCP/IP network, Internet, and Web application programming. Database courses would also benefit from the security model, specifically from its emphasis on the architectural perspective of information repositories. Overall, the usage of such or similar security model can improve a student's coherent understanding of operating systems, databases, and computer networks in the integrated environment of distributed computing systems.

CONCLUSION

The paper describes an attempt to extract a simple model for security analysis, system understanding, and teaching. The model offers a comprehensive abstraction of component trustworthiness in a distributed computing environment. It deals in a systematic manner with all the issues that come from the lack of sufficiently accurate knowledge, both instantaneous and predictive, of the state of a network-based system. It separates the roles of component entities into trusted and untrusted categories and thus helps predict their behavior, even in the case when they become mobile.

The proposed security model and its variants can be used in security, operating system and a database courses, as well as in several seminars and programming projects. It has consistently helped bring the complex issues of distributed system design and distributed system security within reach of students, graduate and undergraduate. Needless to say, there remains much to be done in integrating the model more closely with various relevant branches of computer science theory, but also with homework exercise and hands-on training.

REFERENCES

1. Netscape Security Services. Netscape Views. 2000.
2. Ghosh A. and T. Swaminanta Software Security and Privacy Risk in Mobile E-commerce. Communications of the ACM, v.44, number 2, 2001.
3. Microsoft SDK and DDK. Microsoft Visual Studio 1999. Microsoft Developer Network..
4. IBM Security Architecture. Security the Open Client-Server Distributed Enterprise. SC 28-8135-01, 2000.
5. Dalton C. and T.H. Choo. An Operating System Approach to Securing E-services. Communications of the ACM, v.44, number 2, 2001.
6. Joshi J., W.Aref, A.Chafor, and E. Security Model for Web-based applications. Communications of the ACM, v.44, number 2, 2001.
7. Belavista P., A.Gorradi and C. Stefaneli. Mobile Agents Middleware For Mobile Computing. IEEE Computer, March 2001.
8. Rijndael Advanced Encryption Standard. http://csrc.ncsl.nist.gov/encryption/aes/rijndael/
9. Chakrabarti S. and A.Mishra. QoS Issues in Ad Hoc Wireless Networks. IEEE Communications, Vo.39, No.2, February 2001.
10. Clark D. Face-to-Face with Peer-to-Peer Networking. IEEE Computer, January 2001.
11. Bellovin M. Computer Security – An End State?. Communications of the ACM, v.44, No.2, March 2001
12. Internet resources:
    csrc.ncsl.nist.gov/fips/ - U.S. government standards;
    www.Ietf.org - IETF standards for e-everything;
    www.w3.org/Security/Overview.html
    www.microsoft.com/technet/security/kerberos/ - Kerberos tutorial