Chapter 1 Introduction

"Security is the first cause of misfortune."
Old German Proverb

Personal emails, medical records, company accounts, public telephone systems, or government confidential documents, almost everything is related to security and privacy in some ways. The more people rely on computer systems, the more important computer security becomes.

Security is not a new subject, and is not limited to computers. As people formed early communities, the issue of physical security emerged—the oldest known lock is a 4,000 year old Egyptian lock. Over time society developed rules of security and privacy which were incorporated into laws and enforced. Computer security is simply another step on this continuum with complications introduced by computers' speed and the Internet's widespread communication.

Though the origin of computer security is difficult to pinpoint, some security mechanisms have implicitly existed since computers have been shared. As the technology has evolved, security policies and their enforcement have evolved. Unfortunately, the evolution of computers has outpaced the evolution of the associated security mechanisms. The interconnectivity of the Internet has compounded that differential.

Starting with the notion of security and privacy, this chapter will serve as a framework for understanding the art and science of computer security presented in subsequent chapters. In this chapter we introduce the reader to the concepts, critical components and supporting tools needed for constructing a secure environment.

Security & Privacy: The Definition

Security and Privacy are two different concepts, but are related in that they frequently occur together. We begin with a couple of definitions of security.

Security

"**Security:** In the computer industry, refers to techniques for ensuring that data stored in a computer cannot be read or compromised by any individuals without **authorization**. Most security measures involve data **encryption** and **passwords**. Data encryption is the translation of data into a form that is unintelligible without a deciphering mechanism. A password is a secret word or phase that gives a user access to a particular program or system"

Definition from webopedia.com

Consider the (lengthy) definition of security from *webopedia*: we find critical key words like **authorization**, **encryption**, and **passwords**. Those key words will appear numerous times in this book. While this definition focuses mainly on the data, it provides a useful starting point.

Mirriam-Webster Online (*m-w.com*) defines security as "the state of being secure" with secure defined as "free from risk of loss." Taking that broad notion of security and refining it for the domain of computers we provide a definition of security as:

"***The protection of resources from being accessed by an unauthorized person at a particular time.***"

In the cyber world, resources can be processing power, storage space, or simply data. One can rephrase the definition as: "**Who** can **do what when**?" The set of rules that defines access to resources is called the **policy**. An important task in the security process is defining the policy on "Who can do what when" and then implementing a mechanism to enforce the policy.

Privacy

In many cases, privacy is compromised a result of a security breach. For example, "a hacker is able to compromise a computer system and find out that a person is a homosexual" contains both security and privacy issues. Breaking into a system is the security issue, and the fact that a person is homosexual is the privacy issue. However, privacy and security need not be so tightly intertwined. There exist plenty of cases of only security or privacy issues such as a system compromised for use as a file server of pirated software or a rogue program which reveals medical records on the Internet.

Though security and privacy can be tightly related, a useful definition of privacy with respect to computing can stand by itself. Merriam-Webster Online defines privacy as "freedom from unauthorized intrusion." We refine that definition to:

"**The freedom to control access to our personal information.**"

Implicit in our definition is that some personal information is sensitive. Of particular importance is that sensitivity can vary dramatically from person to person. Thus, a naïve solution for a privacy-concerned application is to give a user a choice to release his or her personal information.

Secure Systems

To create a secure system, we have to first understand the anatomy of our system. In the big picture, a system whether a host computer or a network has a boundary and interfaces. To aid understanding, we will use a "secure room" as an analogy. A secure

room has a ceiling, floor, and walls forming a boundary with a door for entry—the door acts as an interface. To secure this room, we need impenetrable ceiling, floor and walls to ensure that accessing this room can only be done through the door. Given secure walls, we then have to secure the door. To create a secure system you must have both a secure boundary and secure interfaces. Also, just as one can have secure rooms in a secure building, systems with boundaries and interfaces can be nested.

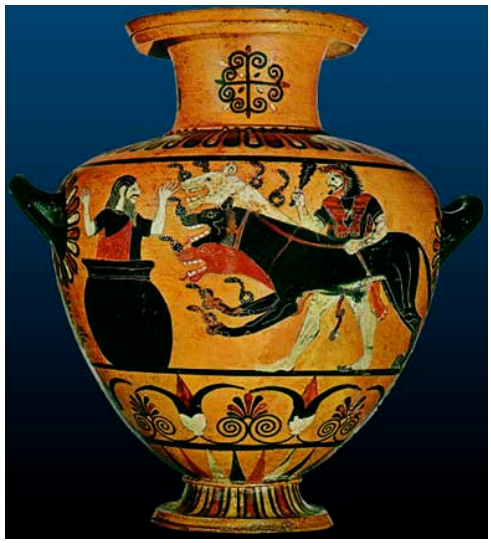
Another theme that is necessary to the security of our system is the validity of information (e.g. input). To aid understanding, we again use a "secure room" as an analogy. Assume that our room has a strong boundary and is proper guarded; we should be able to authenticate and control any access to our room. However, this does not prevent the authenticated user from passing malformed information though the authorized door. For example, your system can only process a number between -100 and 100. If no sufficient validation process is applied to the user's input, entering a large number to this system may result in a serious security problem. Thus, it is mandatory to properly verify information that the system does not have control over (e.g. input) before using it.

In this book, we secure interfaces with three security components (authentication, authorization and accounting). These components together serve as a guardian at the interface, supported by a secure boundary. Equally important are the engineering of the boundary and interfaces, and the validation of information, so that they are impenetrable. Any flaw in the boundary, interface, or validation process may lead to a serious security problem.

Security Components

Kerberos[1]*;*
"n. The watch dog of Hades,
whose duty it was to guard the entrance
. . . is known to have had three heads. . ."

—Ambrose Bierce, The Enlarged Devil's Dictionary



[1] "Kerberos` story derives from the myth of Cerberus (Kerberos in Greek), the three-headed guard dog who allowed people into Hades but not out until Aeneas, desiring a two-way ticket, bribed the beast with a cake. (Proving that even the most watchful security can be circumvented.)"

There are three components to security: Authentication, Authorization, and Auditing In relating these three components to Kerberos, the three-headed guard dog of Hades, the MIT Athena Project came up with the name for their security protocol. We organize our book around these three components. What are they?

Authentication

"Who are you? Are you really the person whom you claim to be?"

Authentication is the process of identifying and verifying the user. In many cases, it simply involves a password or pass phase. However, an authentication method may require a token. For example, a person with a key can enter a room. In this case, the key is the authentication token. Sometimes, a physical token is required for computer access. Authentication may also rely on trust. For example, we trust a person at the system console to be a privileged user. In other cases, a token may be based on biometrics to identify and verify the user.

Authorization (a.k.a. Access Control)

"Do you have the authority to do what you are trying to do?"

Authorization is intimately linked to authentication: once a user is authenticated, a system must determine what that particular user is allowed to do. Authorization can be viewed as a relation among subjects, actions and objects. Subjects may be users or processes; objects are usually resources such as files. Access Control Lists (ACLs) which map users to actions on objects is an example of an authorization system.

With any authorization system there is also a need for enforcement. For example, an access control system may state that a customer (*subject*) can only enter (*action*) a building (*object*) during working hours. The enforcement is to lock the main gate at closing time. In the computing world, the enforcement mechanism is usually part of the operating system.

Accounting (a.k.a. Auditing)

"What did you do?"

Accounting is the act of collecting resource consumption data for the purposes of analysis, planning, or auditing. The security camera at an automatic teller machine (ATM) may be one component of an accounting system. Reviewing collected data can help identify undesired activity.

The fact that an auditing trail is maintained by a system sometimes scares the unskilled attacker. For example, a thief may afraid to rob a store with a security camera installed. However, a skilled attacker will clean the auditing trail to obscure the evidence, e.g. a thief may disable a security camera.

Accounting also provides feed back to improve performance and correctness. For example, an analysis of a log file may show that a file is being accessed by an unexpected user due to the flaw in policy. This information will allow the administrator to analyze and fix the fault.

Supporting Concepts

A guard dog can only watch the access door so it is crucial to have a strong building around the door. If a back window is left open, even the best guard dog may be easily circumvented. That is, the integrity of the rest of the structure is critical to physical security. The same holds true in computer systems.

Integrity

Integrity (n) "the quality or state of being complete or undivided"
Merriam-Webster Online Dictionary

A physically secure structure should be impenetrable except through its guarded entrance. A computer system should have that same characteristic, and insuring its integrity is difficult. Just as a crack in the wall of a safe can provide an opportunity for circumventing the door, a fault in a computer system can provide a vulnerability which can be exploited. A common example is a buffer overflow which has frequently been exploited to create back doors into computers.

Software Engineering & Threat Modeling

"Threat modeling is a method of addressing and documenting the security risks associated with an application."
Threat Modeling by Swiderski & Snyder

Creating a physically secure structure requires engineering so it is no surprise that creating impenetrable software requires engineering. The term "threat modeling" has emerged as a component in the software design cycle for addressing security. Continuing the physically secure analogy, threat modeling can be thought of as determining where all the doors and windows are, and checking that they are locked or barred. If threat modeling is part of a software engineering, a secure system can be built. Whole books have been written on the creation of secure code [cite]. We will briefly cover the topic later.

Validation of Input

*"**All input is evil until proven otherwise**"*
*"**Data must be validated as it crosses the boundary between untrusted and trusted environments**."*
Howard and LeBlanc, Writing Secure Code

"Never judge a book by its cover"
English Proverb

Given a strong building, the (possibly) only source of penetration is the information or materials carried in and out of the building through the door. For example, we authenticate and authorize a postman to enter our building and deliver a parcel of mail only to the mailbox. The postman does not break any security component, but the parcel may contain a dangerous material and eventually damage our system. Packets delivered across the Internet have a similar problem: one cannot trust the contents. For example, a system may not be able to handle a unicode string that user enters to the system. Thus, using data directly from user may cause an unexpected result. To

handle this issue, it is necessary to validate data from another domain before using it in our system. This concept is defined as "Thread Interface" by Howard & LeBlance [cite]. We will deal with this topic later.

Conclusion

In this chapter, we have introduced the concept that a security-aware application is built upon a combination of three components: Authentication, Authorization, and Accounting. Those three components guard access to applications. For the application to be secure it must be otherwise impenetrable, and the concept of integrity created using software engineering, and input validation are necessary.

Exercises

1. Give three examples of cases of Security breach, Privacy concern, and both Security and Privacy related.
2. Identify three cases that the authentication processes can be circumvented.
3. For the automatic teller machine (ATM), identify the possible methods for compromising each security components. e.g. Authentication method, Authorization model and Auditing mechanism.
4. Do you consider *intrusion detection* a part of security components? Please state your argument.
5. Some people believe that encryption is a sufficient tool for construct a secure system. Do you agree or disagree? Please justify your answer.
6. Give an example of a security breach where an appropriate input validation can solve the problem.
7. Case Study: Find NEWS about security breach and identify their problem with respect to Secure Systems that you have studied in this chapter.