International Journal of Advanced Technology in Engineering and Science
Vol. No.5, Issue No. 04, April 2017
www.ijates.com
ijates
ISSN 2348 - 7550

SECURITY ENGINEERING

Nikita Manwani[1], Kruti Lavingia[2]

Institute of Technology, Nirma University (India)

ABSTRACT

Software security is about building software that functions correctly even in the face of animus attack. Security engineering is the idea of systems that remain dependable under any malice, error or bugs. Software security includes consequences due to the implementation of bugs, inconsistent error handling or malicious intruders that can hack into the system. In a technological advanced world, applications shared on the Internet are the most vulnerable. Software security can be achieved by risk analyses and testing from the scratch as avoiding threats while building the software. Security engineering works on tools, processes and methods needed to build, execute and test whole systems. The paper focuses on tools that help protecting distributed systems.

I.INTRODUCTION

Security engineering is about creating systems that stay dependable even in case of a malicious attack. It focuses on a variety of tools and procedures, ranging from cryptography to security of the system via hardware tamper resistance [1].

In certain scenarios, a secure system is extremely vital due to the following reasons - endangerment of life in case of nuclear safety, grave damage to economic organization in case of ATM's or bank systems, or threat to personal privacy such as medical records or other important documents.

It focuses more on of what should not be accessed rather than on the point of view that looks at things that are allowed to view by a certain user. Security requirements depend largely on a respective system. A combination of user authentication, transaction integrity, accountability and fault tolerance is needed. Keeping the system protected depends on several procedures rather than just implementing one process.

Decent security engineering demands four things to be combined that are:

1. Policy: what one is supposed to attain.
2. Mechanism: ciphers, access controls and hardware tamper resistance.
3. Assurance: amount of faith one can put on a particular procedure.
4. Incentive: purpose that the people defending the system have to do their job properly and also motive that the attackers have to try to win against the security.

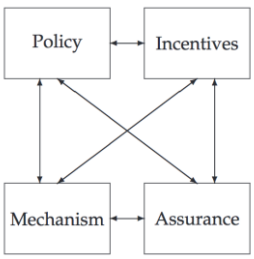

Figure 1. Dependency of the four factors [2].

As it can be seen above in the figure, all the above-mentioned factors are interlinked. Security engineers need to be able to comprehend the risks and threats, analyze the situation and make the right call.

II.CRYPTOGRAPHY

Cryptography is a process for storing and transferring information in a particular way, so only the allowed designated users can operate on it [3]. It is one of the key technologies for protecting distributed systems. Cryptology is the study of building ciphers and breaking them. Plain text is the input to an encryption process and the output obtained from the process is cipher text. The building blocks of cryptography include stream ciphers, block ciphers and hash functions.

2.1 Random Functions – Hash Functions

The Hash functions also known as random function, it is one of the basic random oracle function. It accepts string of any length as the input and outputs a random string of fixed length. Originally used in computer systems for one-way encryption in passwords and now, used to compute checksums on files in forensic applications and check integrity of files i.e. if the file is corrupted or not. Hash functions are also used in messaging applications, also known as message digests. Message, M is passed through a pseudorandom function to get a digest, h(M) which is used in various messaging applications. For example, in digital signatures where instead of signing the entire message a message digest is used when messages are too long.

The main attribute of hash function is its one-wayness. It is easier to find the hash values h(x) for a given input x, but not that easy to find the vice versa where the value of input x is not known. An attacker can just feed more inputs until an output is chosen at random for the known inputs. A pseudorandom function is also a one-wayness function, if there are enough possible outputs that the opponent can't find by chance. Meaning to choose an output to be a n bit number for which the opponent can't hurt us for 2n calculations.

Secondly, the even part of the input will not be known from the output. One way encryption of value x can be obtained by concatenating it with a secret key and calculating h(x,k).

Lastly, the third property of pseudorandom function is that it is hard to find collisions for long outputs. If the hash function is n bit long, there are 2n hash values possible and the number of hashes to be computed will be 2n/2 hashes.

2.2 Random Functions – Hash Functions

Stream cipher also known as key stream generator is one of the primitive random cryptographic generators. It has a short input, but a long output. At the conceptual level, it is common to think of a stream cipher as a random oracle whose input length is fixed while the output is a very long stream of bits, which is known as the key stream.

It can be used rather simply to protect the confidentiality of backup data: one can go to the key stream generator, enter a key, and get a long file of random bits, and then exclusive-or it with his/her plaintext data to get the cipher text, which one can then send to their backup contractor. If one needs to recover the data, one can go back to the generator, enter the exact same key, and get the exact same long file of random data, and then exclusive-or it with their cipher text to get the original plaintext data back again. Other individuals with access to the key stream generator will not be able to generate the same key stream without knowing the keys.

2.2 Random Permutation – Block Ciphers

The third building block is primitive and the most significant in modern commercial cryptography, it is the block cipher, which is modeled as random permutation. In this, the given function is invertible, and the input plaintext as well as the output cipher text both is of a fixed size. One can visualize block encryption as having an elf in, a box with a dice and a scroll. On the right the elf has a column of cipher texts and on the left a column of plaintexts. When the elf is asked to encrypt a message, it first checks the left column to see if it has a record of it. If not, it then uses the dice to produce a random cipher text of the suitable size (which does not appear yet in the right side column), and then writes down the plain text/cipher text pair in the scroll. If it does find a record, it will give us the equivalent cipher text from the right side column.

Whenever asked to decrypt, the elf does the same, but with the function of both the columns reversed: he first takes the input cipher text, checks it (this time on the right side scroll) and if he finds it, he'll give the message with which it was formerly associated. If not, he will generate a message at random (which does not already appear in the left hand column) and notes it down. A block cipher is like a keyed family of pseudorandom permutations. For each key, one has a single permutation, which is independent of all the others. One can think of each key as corresponding to a different scroll. The instinctive clue is that a cipher machine should output the cipher text given the plaintext and the key, while output the plaintext given the cipher text and the key, but when given only the plaintext and the cipher text, the machine should output nothing. One can write a block cipher using the notation, which is established for encryption:

$$C = \{M\}\ L$$

The random permutation model also permits to define various types of attack on block ciphers. In a recognized plain text attack, the opponent is just given a number of indiscriminately chosen inputs and outputs from the oracle corresponding to a target key. In a chosen plaintext attack, the opponent is permitted to place a certain number of plaintext queries and acquire the corresponding cipher texts. In a selected cipher text attack, the attacker gets to make a number of cipher text queries. In a chosen plaintext/cipher text attack, he is allowed to make queries of either type. Finally, in a linked key attack he can make queries that will be responded using keys related to the target key L, such as L+1 and L+2. In every case, the aim of the attacker might be either to deduce the answer to a query he has not already made (a counterfeit attack), or to recover the key (predictably known as a key recovery attack).

2.3 Password Protection

These days, all the new network operating systems and multiuser computers engage passwords to protect and authenticate users who are operating the computer and/or the network resources. These passwords are not usually stored on the host computer or the server in plain text, rather are typically encrypted using some sort of hash scheme.

For example, UNIX/Linux uses a well-known hash scheme via its crypt() function. Passwords are kept in the /etc/passwd file; each record in the file contains the hashed password, username, user's individual and group numbers, home directory, user's name and the shell program; these fields are separated by colons (:). Note that each password is stored as a 13-byte string. The first two characters are essentially a salt, randomness added to each password so that if two users have the same password, they still will be encrypted differently. In fact, the

salt provides a means so that one single password may have 4096 different encryptions. The left behind 11 bytes are the password hash, calculated using DES.

As a matter of fact, the /etc/passwd file is word readable on UNIX systems. With the weak encryption of the password, resulted the development of shadow password system in which passwords are reserved in a separate, non-word readable file used in unification with the normal password file. When shadow passwords are used, the password entry in /etc/passwd is exchanged with an 'x' or '*' and the MD5 hash of the passwords are stored in /etc/shadow besides some other account information.

III. NETWORK ATTACK AND DEFENSE

3.1 Attack

High number of network attacks, and defenses, occur when there a large numbers of machines networked together. These attacks depend on a number of aspects, the most significant being the protocols the network uses. There are various other components in the protocol suite for managing of communications and providing higher-level services. Most of them were produced during the good old days when the Internet had only reliable hosts and security was not an issue. So there is petite authentication built in.

So the ease with which a bad machine can take down other machines on one's machine depends on how tightly one has the network locked down, and the harm that one bad machine can do will rest on the size of the local network. There are restrictions to how far a sysadmin can go; your firm may run a complex mixture of legacy systems for which Kerberos just won't work. Also, a security-conscious system administrator can enforce real costs.

One last attack that is worth touching upon is the heading of attacks on local networks, and that is the rogue access point. Occasionally we find Wi-Fi access points in public areas, e.g. airports, which have been deployed maliciously. The operator might sit in the airport lounge with a laptop that can access the Internet via a paid Wi-Fi service and advertise a free one and as someone will use it, he/she will be able to sniff any plaintext passwords one enters, for example one's webmail or Amazon account, and as someone tried to do online banking the attacker might plausibly send you to a malicious site. So the effects can be somewhat like drive-by pharming, although more reliable and less mountable. In addition, rogue access points might also be devices that employees must have installed for their own suitability in defiance of corporate policy, or even official nodes that have been misconfigured so that they do not encrypt the traffic.

3.2 Attacks using Internet Protocols and other Mechanisms

3.2.1 SYN Flooding

This attack simply sends a large number of SYN packets and never ever acknowledges any of the replies. This will lead the recipient to gather more records of SYN packets than his/her software can handle. This type of attack had been known to be theoretically promising since the 1980s but only came to public attention when it was used to bring down Panix, a New York ISP, for several days in 1996.

Smurfing

One of the common ways of bringing down a host in the 1990s was smurfing. This attack exploited the Internet Control Message Protocol (ICMP), which enables users to send an echo packet to a distant host to check whether it is alive. The problem was with the broadcast addresses that were common for a number of hosts. Some applications of the Internet protocols returned to pings to both the broadcast address as well as the local address. A group of such hosts at a broadcast address is known as smurf amplifier. Immoral people would construct a packet that would have the source address forged to that of the victim, and direct it to a one of the smurf

amplifiers. These would then send a number of packets to the target, which could slough it. Typically used by teenagers to take over an Internet relay chat (IRC) server, in order to take control of the chat room.

3.2.2 Spam

Spamming is flood of commonly unwelcome traffic sent out for the most part by botnets, and frequently with clear criminal intent. The technical features are that both email and the web protocols (SMTP and HTTP) assume incorrectly that the lower levels are protected. Spam bots might forge the sender's email address.

3.3 Defense Against Network Attack

Spamming is When shielding against network attack, there are largely four sets of available tools.

1. First one is management — keeping one's systems up-to-date and configured in ways that will diminish the attack surface.

2. Next is filtering — the usage of firewalls to end corrupt codes like Trojans and network exploits, and to sense signs of attack and compromise if something gets through.

3. Next is the intrusion detection — having programs monitoring one's networks and machines for signs of malicious behavior.

4. Lastly there is encryption — protocols such as TLS and SSH that enable to guard specific parts of the network against specific attacks.

VI. SUMMARY

Many ciphers do not work because they are used unsuitably, thus the need of a strong model of what a cipher should do. The random oracle model offers a useful intuition: we assume that every new value returned by the encryption engine is random in a way of being statistically independent of all the distinct outputs seen before. Block ciphers for symmetric key applications can be constructed by the cautious combination of substitutions and permutations for asymmetric applications such as public key encryption and digital signature one uses number theory. In both cases, there is quite a huge body of mathematics. Other varieties of ciphers — stream ciphers and hash functions — can be constructed from block ciphers by using them in appropriate modes of operation. These have distinct error propagation, pattern concealment and integrity protection properties. The elementary properties that the security engineers need to understand arenot too tough to grasp, though there are many subtle things that can go wrong. In particular, it is astoundingly hard to build systems that are robust even when components fail (or are encouraged to) and where the cryptographic mechanisms are well integrated with other actions such as access control and physical security.

Averting and sensing attacks that are launched over networks, and mainly over the Internet, is probably the most newsworthy aspect of security engineering. The problem is dubious to be solved any time soon, as many different kinds of susceptibility gives a way to the attacker's toolkit. Ideally, people would run carefully written code on trustworthy platforms. In real life, this won't happen always, or even often. In the corporate world, there are grounds for hope that firewalls can keep out the worst of the attacks, careful configuration management can block most of the rest, and intrusion detection can catch most of the residue that make it through. Home users are less well placed, and most of the machines being recruited to the vast botnets we see in action today are home machines attached to DSL or cable modems.

V. REFERENCES

[1]. Sommerville, Ian. "Software engineering", International Computer Science Series, *Addison-Wesley, 1989.*

[2]. Anderson, Ross."Security Engineering: A Guide to Building Dependable Distributed Systems" *Wiley Publishing, Inc. 2008*

[3]. Kessler, Garry. "An Overview of Cryptography" *Auerbach in September 1998.*