S.38153 Security of Communication Protocols

- General:
  - Lectures Tu 10-12 S2 J. Jormakka
  - Exercises We 9-11 and We 14-16 laboratory every week
  - Credits 2 = (>1 cr) exam + (<1 cr) exercise report (passed/rejected)
  - Preliminary content, this Spring security attacks and defenses. See the Web page, some changes are likely.
  - Lecture material:
    - Find any thick book of security of the Internet dealing with the matters on the course.
    - Lecture notes are copies of transparencies, they are not good as a stand-alone study material, they are a check list of issues treated on the course so that you know what to study from books. (Well, you manage OK in the exam reading only them.)

General content

- The course is about security attacks to the Internet and ways to protect the network.
- The lectures explain different attack and defense mechanisms
- The exercises are mandatory (not to attend always but to return a report is required)
- Different tools are tried in the exercises in a laboratory network, the attendants are divided to a group of attackers and a group of defenders.
- We try to get some measurement results from the exercise reports: how easy it is to make some types of attacks and how well the network can be protected by existing methods.

What the course is not?

- This is not a basic course of applied cryptography, B. Schneier: Applied Cryptography is a good source for this kind of information, such knowledge is not assumed to be known in this course.
- This is not a course giving up-to-date information of current security products and tools, take instead some course from TIK with visiting lecturers from industry.

- This course does not teach good hacking tricks. All information on lectures is available in books on these matters. In the exercises you may be trying attacks that might work somewhere, trying any attack to some other system is illegal, do not do so.

What should be protected?

- People do security breaches. There must be one or more intentional human attackers to make a security attack.

- Security attacks usually violate:
- Privacy (confidentiality) - data is not disclosed to unauthorized people.
- Integrity - data is not changed by unauthorized people.
- Availability - data is not available to authorized users (people or not).

- This division has a large acceptance but it is not quite complete - there are other aspects in security.

What should be protected?

- In the areas of security there are cases when it is hard to say what area an attack is violating:
  - Integrity and availability can be related: If an unauthorized user manages to crypt some protected data does it become unavailable data or is the integrity violated?
  - Privacy and integrity can be related: If an unauthorized user plans a trapdoor which can later be used for violation of privacy (like read files) or violation of integrity (like remove files), what area planting the trapdoor violates?
  - Privacy and unavailability can be related: the trapdoor could be used to violate unavailability.

What should be protected?

- A type of security attack which does not attack privacy, integrity or availability is for instance faking somebody's digital signature and faking an agreement on somebody's name for a business deal outside data communications.
- The examples show that privacy, integrity and availability are a classification mostly for attacks to stored data. There are other areas:
- access to processes
  - access to transmitted data
  - security as a service (like giving digital signatures)
- These areas contain different types of attacks.

What about the law?

- The law is not covering all cases.
- A crime against privacy can be either that transmitted data is being read, or that access is obtained by breaking security mechanisms.
- If for instance a system administrator reads emails sent to a former employer when the mail is in a mailbox, it is unclear if he violates privacy of mail.
- Copying data protected by security mechanisms is a theft, if the data is not well protected it may be a theft.
- Damaging somebody’s system is most probably criminal as any unauthorized damaging act.
- Writing/spreading a virus is a crime, but what is a virus.

Who are the attackers?

- The attackers contain different types of people like teenage hackers wanting to impress peers, university students/personnel trying some nice new trick, tiger teems, dissatisfied former employees, computer criminals, industrial and military spies, vandals and terrorists.
- Rather than making a list of all types, we can classify the attackers by their goals:
  - wish to show ability (hackers)
  - economic gain (criminals)
  - wish to destroy (vandals)
  - political and military gain (terrorists, military)

Who are the attackers?

- The attackers differ also by their competence. Is it important for a defender to know what people are making the attacks?
- If is probably important to know how large is the number of competent attackers.
- If is probably important to know how many attackers are hackers with no intention to harm.
- There seems to be very little knowledge of these issues. We will try to get some light on this issue in the exercises by investigating just how difficult are different types of attack for people (that is you) self-estimating their competence.

Why security problems in data networks?

- It is customary to mention when discussing security of the Internet that there are security problems in all communication networks, but it is not quite so, there are more problems in the Internet than in, say PSTN.
- If you compare the Internet to a telecommunication network like PSTN of GSM you see that a telecommunication network is basically a service network.
- What we can do with a service network is: cheat in bills if signaling is too simple, block the network if it is not enough protected, listen to transmissions unless they are encrypted well enough, cause problems like crash some services by exploiting bugs and abuse services.

Why security problems in data networks?

- To a large extent we can design the network and services so well that these problems can be avoided. I think it is possible to offer a sufficiently large set of sufficiently secure services.
- A data communication network like the Internet is basically a platform for making any computing in networked computers. Its origin is networked computing in a LAN in a secure environment.
- Such an environment wants to offer things like remote access which make possible stealing files, destroying data etc.
- I think a general purpose convenient distributed computing environment will not be secure.

Why security problems in data networks?

- What is the future?
- To a large extent the Internet is not any more a distributed computing platform. Firewalls block remote access to hosts outside your own network.
- People mostly use a small set of services: email, file transfer, web, maybe in the future voice and video services. There is little need for a possibility for remotely logging into a system at all. Maybe we could drop all dangerous features.
- But there are other development scenarios: mobile code is still one of the favorite ideas in the Internet community. Executable attachments in email, like macros, applets and scripts cause security problems.
- Seems that the Internet may not become a secure service network.

Why security problems in data networks?

- Some think that Internet security will be solved in a short time and maybe is almost solved with IPsec and IKE.
- There are indeed methods to solve some security problems:
- privacy of transmitted data through IPsec
- privacy of transmitted and stored data like PGP.
- authentication through public key cryptography or by one-time passwords
- protection to some forms of address spoofing and use of vulnerabilities through firewalls
- protection against some known types of malicious code through virus protection
- protection against misbehaving malicious code through sandbox model like in Java security
- use of scanners for locating vulnerabilities

Why security problems in data networks?

- There are security problems which are not yet solved and may not be solvable.
- My favorites are the following problems:
- Denial of Service (DoS) attacks. At the moment these attacks use features of some protocols but in general, overload protection is very difficult for a network whose structure is not carefully planned.
- Bugs in software and design. These vulnerabilities can usually be fixed if they are found but if new applications are introduced in a fast pace without careful quality control there is no hope of getting all bugs removed. In general, avoiding bugs is impossible.
- There are no complete protection methods for harmful mobile code of different type (Java scripts, mobile agents etc.)

Attacks

- Let us start by looking at some common and traditional attack types and then in other lectures go into more detail.
- Address spoofing (=cheating)
- This is a set of attacks where the attacker sets to IP or TCP frames wrong addresses and cheats the system.
- A simple example: computer A in a LAN wants to talk to computer B. IP on a LAN uses MAC addresses and A has B's MAC address in a cache. After some time the cache times out and the computer A broadcasts an ARP message. The destination B answers with ARP and gives the MAC address corresponding to its IP address. A stores B's MAC to the local cache.
- If there is another computer faking the ARP answer with another MAC address, chances are that the correct ARM answer from the destination is ignored by A. Then you can play B.

Attacks, 1988 Internet worm

- A worm is a program which replicates and spreads over a network to computers. A worm is not a virus, that is, it does not attach itself to other programs.
- The Internet worm released 2. November 1988 caused a large part of the Internet to be out of service for several days. The worm infected only about 2000 hosts of the 60000 hosts in the Internet at that time, but several hosts disconnected themselves as a protective measure.
- The worm was written by a graduate student Robert Morris, later convicted for penalties. The worm did not damage files, plant trapdoors or steal passwords. Morris even helped to stop it. Some think it was not meant to congest computers either. Anyway, it caused a lot of disturbance and lost workdays.
- There is a detailed description of the worm at

  http://kt-www.cs.titech.ac.jp/~natori/…/wormtour.html

Attacks, 1988 Internet worm

- The worm is the best documented example of an attack and though it is very well known, let us look at it. The worm contained many mechanisms for spreading and for hiding itself.
- Spreading: The worm contained a small 99 byte bootstrap routine and a larger program (>3200 lines of C) body. The bootstrap routine opened a TCP-connection and downloaded the body to files and executed by the bootstrap routine.
- Three mechanisms were used to send the bootstrap routine.
- Firstly the worm cracked passwords using natural derivatives of user names, a list of 432 favorite passwords and the Unix on-line dictionary.
- The use of passwords was surprisingly poor and these methods could find user passwords rather fast. It is still as bad. You should try Crack and its variants in the exercises.

Attacks, 1988 Internet worm

- Cracking a password in Unix before using salt was relatively simple but slow.You could read the encoded passwords, crypt a candidate and compare. Worm had a faster crypt routine.
- If the worm managed to crack a password it tried to rsh or rexec to log into the user's account on another computer. It used host names from /etc/hosts.equiv or .rhosts files. Logging to a host in hosts.equiv requires no password and logging with rexec relies on the user having the same password in many systems. This low security of Unix is to make the network convenient for users.
- If this failed, the worm tried a bug in Berkley Unix version of finger. Finger daemon read the request by the C-routine gets(). It did not check the size of the request, assuming it is under 512 bytes.
- The worm put there 536 bytes causing an overflow.

Attacks, 1988 Internet worm

- This overwrote the stack and when the main program of the finger routine finished, the program did not jump to exit but to the code in the input data. That code started a shell and the bootstrap program. The bug on gets() was not known then, now it is fixed.
- As the third method the worm used two known bugs in Unix sendmail. Sending mail to processes instead of mailboxes was meant for replies like not in the office.
- Two bugs changed this feature to to work as follows: if sendmail was compiled as a debug version and the program was put to a debug mode by sending a debug command, one could put a command string in the place of a recipient.
- The distributed version of sendmail was compiled with debug options. It seems that the bug was created rather recently before 1988 as a fix of some security problems.

Attacks, 1988 Internet worm

- The worm send a command string which caused sendmail to compile the bootstrap code in the body of the message and to execute it. This is not the only trapdoor in sendmail, there may be more.
- Hiding: After the bootstrap code was in the remote system, the body of the worm was loaded into files. The files were read into the memory and the file copies were removed.
- The worm changed its name to sh to look innocent.
- It changed periodically its process id and worm processes died themselves to avoid detection, some stayed and jammed the system.
- It encrypted its code, disabled core dumps, disabled signals from somebody wanting to dump it, modified the memory image, all this to make noticing and analyzing of the worm harder. There were also misleading code parts etc.

Attacks, 1988 Internet worm

- Naturally, the exact mechanisms used by the worm do not work any more. Sendmail and finger bugs are fixed and proxies are used to avoid possible remaining bugs in complicated programs like sendmail. rsh, rexec are usually not available through firewalls.
- Notice, that a good attack like the 1988 worm is a combination of several methods. It seems very difficult to estimate how probable is such a combined attack.
- The incidence caused CERT Computer Emergency Response Team to be established, it is placed to the Carnage Mellon University. It records security breaches in the Internet and helps systems under attacks. Details of attacks are usually not given, but enough knowledge may be obtained from the fixes announced by CERT. There are now many CERT type teams in different countries.

More attack methods

- Root privileges: The worm did not make use of root privileges because of its way of spreading. Usually you need root privileges to remove traces and to do more damage.
- Removing traces: there are Unix trace files. Some of them are easy to remove. Most require root privileges.
- Now many security aware sites use an Intruder Detection System (IDS). IDS is passively observing links. There are many IDS servers, so you probably cannot kick them all down. But we do not know what attack types a give IDS actually notices. We will try an IDS in the exercises.

Collecting information

- The first stage for an attacker may be collecting information from the network.
- Pinging the network (very seldom not to be noticed) is a common way of gathering information. It seems that the system administrators cannot say if somebody is pinging them for preparing security attacks. You usually cannot block ping as it is very useful.
- Unix commands like finger give knowledge on users that are logged in. It is nowadays common to block finger from outside use as it give too much knowledge.
- There are special tools by which you can gather information from the network topology (like GeoBoy & PacketBoy) to know what kind of computers there are.
- A sniffer is a simple way to spy unprotected passwords and other information from a LAN or from a router.

Obtaining root privileges

- Many Unix applications, like sendmail, run under root privileges. (sendmail must run as root in order to be able to copy messages to user's directories).
- They may contain vulnerabilities where buffers can be overflow and the overwritten stack causes the program to execute instructions in the overwritten area with root privileges, like in gets().
- Similar mechanism can be used to gain root access from a user account access: run some script which requires executing a call with root privileges and overflows a buffer there. Another mechanism which has been stated is to abort a call when it has root privileges and fall to a privileged shell.
- It is claimed that the number of people who can obtain root access in an unauthorized way in Unix is rather constant in time. We would like to know if one can find tricks to do this.

How to find bugs?

- Exploiting bugs has been one of the main ways to break security. How to find the bugs as an attacker? One way is of course to study the system code carefully and we may assume that many do.
- Still, the Internet worm used well-known bugs and the ways to gain root privileges have mostly been well-known Unix bugs which were left there (why?).
- We could imagine that these bugs have mostly disappeared and must disappear.
- The easiest way for finding new bugs is to follow bug fix list by CERT or other similar teams and try the described bug to a system which is not updated.
- There is underground information of bugs also.

Attacks: cracking passwords

- A Unix system from the box has often access controls set in an easy way, the system administrator should set the access controls for sufficient security level. If he does not, there are easy ways of access.
- One simple way is to use Trivial FTP (TFTP). This file transfer program, more simple than FTP, does not ask for a password. If the range of directories which it can access is not limited, you can get the password file with it.
- Then you can use a program like Crack and try to guess passwords. What you may home to obtain are (usually) user passwords as superuser passwords are typically strong, so you get user account access.
- You may then try to change to root privileges as explained before.
- Naturally, you should block TFTP from outside usage.

Attacks: simple attacks by UUCP

- There are other vulnerabilities in a system with off-the-shelf configuration.
- UUCP and R-commands are especially dangerous as they use access without passwords relying on the computer being in the .rhosts list of a user.
- Nowadays these holes are usually filled, if there is this type of access possibility to a system which should be protected it is usually an intentional trap.
- An attacker probably uses a scanner for finding these holes. A defender has usually also used a scanner and blocked these holes.
- In the exercises we will try to use scanners, like SATAN, COPS, SAINT, NetScan, Nmap. They are mostly freely available.

Attacks: planting a trapdoor

- Assuming that you do not find any bugs from the system, it may be worth trying to insert one.
- Anything that the root executes will usually have root privileges. You may send a virus to a system administrator and with the virus open a trapdoor.
- A simple trapdoor would be to open a socket which the virus daemon listens. A more tricky way which may better stay hidden is to assume that you can get to a user account by password guessing and then the daemon would execute a program from the user’s directory.
- But you have to pass the antiviral software.

Attacks: passing the antiviral software

- Having never tried to write a virus, I would expect it to be possible to pass virus protection, but is it hard?
- Virus protection relies on detecting patterns of virus code and on execution of code in a virtual machine where the code can do no harm.
- If you write a code which will wake up only after some specific trigger it will not do anything dangerous in the virtual environment for the test time.
- If you create a new virus code and it is not of spreading type, it will pass the protection and may well stay undetected to the time you need it.
- You can hide the virus code by running cryptation and by several other well-known mechanisms. If the virus does not spread suspiciously and does not do harm, how could it be detected.

Attacks : what you want to do as an attacker

- There is no complete taxonomy on attacks but we can summarize:
- collect information
  - user names
  - get information of computers/operating systems
  - get information of possible security holes
- obtain an initial access to a user account
  - use a backdoor which is left there intentionally
  - use a backdoor which is forgotten
  - spoof the access control system
  - break the access control system e.g. with Crack
  - use non technical ways like social engineering
  - use a bug in the system
  - plant a backdoor

Attack: what you want to do as an attacker

- enlarge your access
  - get access to root
  - arrange that you get access later
  - continue to other systems
  - remove logs
  - hide in every way your attack
- make what you want
  - destroy files
  - corrupt files slowly
  - insert wrong information
  - locate data you want and read/copy it