Contents of Lecture 9

- Protection: between users on the same system
- Security: external attacks

Protection domains

- A **domain** describes what somebody (called a **principal**) may do
- In UNIX domains are defined by pairs of user and group identifiers: UID and GID
- A domain is a set of (object, rights) pairs
- An object can be a file and the rights can be read, write, execute permissions
- We can view all domains and objects as a huge matrix with one column per object and one row per domain
- The question is: how should the matrix be implemented?

UNIX file protection bits

- The basic mechanism in UNIX is for each file to list the rights of three domains:
  - what the owner of the object (eg file) may do
  - what users which belong to the same group as the owner may do
  - what everyone may do
- The list is sorted!
- See next slide

UNIX file protection bits are sorted

```
$ cat > secret
top secret
$ chmod 004 secret
$ ls -l secret
-------r--  1 js  staff  11 May  7 09:06 secret
$ cat secret
cat: secret: Permission denied
$ rm secret
override ------r--  js/staff for secret?
$ rm secret
override ------r--  js/staff for secret? y
$ rm secret
rm: secret: No such file or directory
```

using **rm -f secret** *avoids the question*

Access control lists

- With access control lists, **ACL**s, each object has a list of who may do what with that object, usually in addition to the usual UNIX list
- Access control lists are supported in Linux eg through the **Security Enhanced Linux** from the US National Security Agency

Capabilities

- Storing the matrix by rows instead leads to each principal having a list of which objects it may operate on and how
- Such stored right is called a **capability**
- Capabilities are often implemented by letting a principal have a ”pointer” to an object with the rights and a check field
- When a capability is to be removed, certain bits in the object are changed which makes the check field obsolete and thereby removing the capabilities
- A problem with capabilities is to selectively remove capabilities from one or a group of principals, since changing the check bits destroys all capabilities

Logic bombs

- **Logic bombs** are code installed by someone with access to the system
- If certain events occur (or do not occur eg pay salary), the logic bomb explodes
- For example: one installed program checked the payroll and if the author was not listed for two consecutive months, the bomb exploded
- Which kinds of bombs are there? For example Modifying payments and Encrypting or removing important files
- Encrypted or deleted files can then be restored as a ”consultancy service”
- Such black mail is serious because even if the police arrests the criminal the company still needs the lost information and might prefer to pay

Trap doors

- Special code can be inserted in the source code for checking login and passwords
- Less likely in open source code???
- Code reviews reduce this problem
- Ken Thompson suggested a clever way to insert trap doors:
  - Modify the compiler to recognize the login code
  - Make the compiler generate the trap door code for the recognized source code
- This is impossible to detect using code reviews and must be prevented by controlling the compiler

Buffer Overflow Attacks

- Suppose a program has an array and reads input to it without checking array index out-of-bounds
- If data is written at indices after the array, the behaviour is undefined in C
- The input can be carefully crafted and possibly change the behaviour to enable an attack
- For instance, many attacks have overwritten the current function's return address to instead jump right into the array and execute the bits there interpreting them as instructions
- The simple rule for C programmers to avoid all such problems is to **not trust untrusted input sources** and code appropriately

Format String Attacks

- Suppose your program reads a string from an untrusted source and then prints out the string.
- The following code introduces a security risk:

```
char*   s = read_string_from_bad_guy();

printf(s);
```

- The string read can of course contains formatting commands
- What does the formatting command **%n** do? Writes number of output bytes to an address which can be exploited in an attack

Using strcpy

- The two previous methods require that the stack is executable which it usually is not
- The C library function **strcpy** copies a string to a destination address
- By setting up the input on the stack as the source address to strcpy and using some suitable address as destination, and setting up the return address of strcpy to jump to the destination, the input will be copied and then executed
- The data segment is executable

Arithmetic overflow

- By providing input which results in arithmetic overflow a program can be made to allocate too little memory for a buffer which enables more buffer overflow attacks
- For instance a program may allocate a buffer to hold an image and multiplies the width and height of the image
- Always check for overflow in security critical programs and untrusted input sources

Code Injection Attacks

- The C library function **int system(const char* cmd)** executes the command given as argument
- Which errors are made in following code?

```
int main(void) {
        char    src[40], dest[40], cmd[85] = "cp ";
        gets(src);
        gets(dest);
        strcat(cmd, src);
        strcat(cmd, " ");
        strcat(cmd, dest);
        system(cmd);
}
```

Answer

- **gets** is dangerous
- The **dest** file might be named: **file2; rm -rf** /
- The same problem exists in many situations including reading and executing SQL commands

Malware

- Botnets
- Keyloggers
- Trojan horses
- Viruses
- Worms
- Spyware
- Rootkit

Botnets and keyloggers

- A **botnet** is a number of machines which have been taken over by some criminal
- They are ordinary machines and the owners are usually not aware of the attack
- Typical uses is to send spam from them
- A keylogger is software which logs and sends the sequences of keys pressed by the user at the terminal (PC)
- For example passwords and credit card numbers obviously are exploited
- Especially passwords to Internet banks
- Never be a customer of a bank which forces use a password to log into it

Trojan horses

- The problem for malware writers is installing it on others' machines
- A **Trojan horse** is a program which does something useful and in addition something bad, eg installs malware (or is the malware itself)
- Typical examples are games that are installed or web pages downloaded
- A machine with a Trojan horse that is in contact with the criminal for further instructions is called a **zombie**
- Trojan horses can also be installed by hoping that somebody makes a typing error and types eg **c** instead of **cd**. With proper file modes the command **c** cannot be installed eg in /**usr**/**bin**

Viruses

- A **virus** is a small piece of code that is attached to some program, and when run, the virus copies itself into other programs etc
- When it has infected other programs it perform its own criminal operations in the attacked machine
- A virus is usually first spread using a Trojan horse
- It is easy to write a virus for Linux once you know how the executable file format
- Add the virus eg at the end of the **text** segment and change the startup code to call it before **main**

Boot sector viruses

- A virus which installs itself by modifying the master boot record or first sector of the active partition can of course be powerful
- It can take a few disk sectors and write the original sector there (so that it can boot the OS) along more malware code
- Taking a disk sector can be done by taking an unused sector and eg marking the appropriate bit in the file system so that it looks like a normal sector in use
- When run **fsck** (or some other disk checking software on a non-UNIX system) will discover this since no file uses it and take it back though

Other kinds viruses

- If a device driver can be infected, the kernel will kindly load the virus at every boot which makes it more convenient
- A **macro virus** is executable "macro" code in eg a Microsoft Word document which when opened can spread a virus
- So, it's very easy to write such viruses and less skilled persons can do it
- In June 2000 a virus of this kind with an email subject of "I love you" made damages exceeding one billion US dollars

The Morris Internet worm

- A **worm** is a self-replicating program
- A graduate student at Cornell, Robert Morris, discovered two bugs in BSD UNIX
- He wrote a self-replicating program which spread in November 1988 and put almost all Sun and VAX UNIX machines on the Internet to a halt within a few hours by overloading them
- Morris is now a professor at MIT

Two programs

- The worms consisted of a bootstrap program and the worm
- The bootstrap program (99 lines of C code) was compiled and executed on a remote machine and it fetched the worm from the machine it came
- The worm then looked for new machines to spread to
- Morris could easily have done malicious on the infected machines (ie almost UNIX machines) but it didn't
- By replicating itself enough times the host machine got too many processes and crashed

Spreading the bootstrap program

- Three techniques were used to spread the bootstrap program
  - Using **rsh** to login to a machine which trusts the originating machine — **ssh** is ***much*** better
  - A buffer overflow in the **finger daemon**
    - Typing `finger username@host.example.com` requests the finger daemon to print out info about username such as when mail was read last time
    - The finger daemon didn't expect too long usernames and Morris constructed a 536 byte long name which caused a buffer overflow
    - The buffer overflow resulted in interpreting and executing the user name as machine instruction which started a shell as root
  - A bug in the **sendmail** program which made it possible to start a shell as root

More about the rsh attack

- A user can specify a host on which it can execute commands without giving a password
- But to perform this attack from a machine **A** to machine **B** the worm needs to crack passwords on machine **A**
- So the worm did to do so
- To find out which passwords are popular Morris could read a well known paper by Ken Thompson and his own father, a security expert at NSA

Replication

- If a machine which already was infected was attacked again, the attack was cancelled 6/7 times
- The 1/7 probability of replication was enough to halt the machine

Spyware

- Is software which
  - hides
  - collects data about the users
  - sends the collected data
  - avoids being deleted
- Spyware is typically spread as Trojan horses
- Microsoft's activeX controls are plugins to Internet Explorer
- Eg it pops up and asks if user wants a faster Internet?
- Click OK and download and install the spyware

Rootkits

- A **rootkit** is software which is very difficult to locate and remove, and usually contains malware too
- For example, if **libc** and a kernel module are infected then it is very difficult to detect and remove the malware since you cannot trust much
- Sony once distributed a 12 MB rootkit on some music CDs for Windows machines
- When Windows sees a CD inserted, it looks for a file **autorun.inf** and runs it
- Usually music CDs don't have such files
- The idea of Sony was to distribute the rootkit in order to detect music piracy

Detection

- Mark Russinovich was developing a rootkit detection program and ran it on his own machine
- Surprisingly for him (and Sony perhaps) he found there already was a rootkit on his machine!
- The Sony rootkit intercepted all system calls to read the CD drive and made it impossible to read music CDs (except for Sony's player)
- It did other actions to try to avoid its discovery
- More than 500,000 machines were infected

The uninstaller

- Sony agreed to publish an uninstaller but to download it people had to accept future promotional material from Sony
- The uninstaller was buggy and made it easier for attackers to infect the machine
- As a compensation users were allowed to download three music albums
- Later it was discovered that the rootkit reported back on people's listening habits which is a violation of US law
- Each damaged machine's owner was compensated by USD 150

Anti-virus programs and polymorphic viruses

- To remove viruses, anti-virus programs look for byte sequences of known viruses
- A **polymorphic virus** mutates and uses different but equivalent instruction sequences when it infects itself to make detection more difficult
- There will always be a war between anti-virus programs and virus makers much as for submarines and anti-submarine warfare (using warships, aircraft and other submarines)