Security Engineering

Lesson 2
Passwords Issues and
Access Control

Spring 2010
Dr. Marenglen Biba

Technical Protection of Passwords

- A broad range of attacks can be used to recover other people's passwords.
- Some of them target the password **entry mechanism**, while others exploit **the way that passwords are stored**.

Attacks on Password Entry

Interface Design

- Sometimes the problem is **thoughtless interface design**. For example, some very common models of cash machine had a **vertical keyboard at head height**, making it simple for a pickpocket to watch a customer enter her PIN before lifting her purse from her handbag.
- The keyboards were at a reasonable height for the men who designed them, but women—and men in many countries are a few inches shorter and were highly exposed.
- Ironically, one of these machines "**protected client privacy**" by forcing the customer to gaze at the screen through a narrow slot.
  - Your balance was private, but your PIN was not! ☺

Attacks on Password Entry

- Many pay telephones have a similar problem, and **shoulder surfing** of calling card details (as it's known in the industry) has been endemic at some locations such as major U.S. train stations and airports.
- For that reason, it might be a good idea to **cover the dialing hand** with the body or the other hand when entering a card number or PIN in a public place.
- But **systems shouldn't be designed** on the assumption that all customers will do this.

Eavesdropping

- Taking care with password entry may stop the bad guys looking over your shoulder as you use your calling card at an airport telephone, but it won't stop all the **eavesdropping attacks**.
  - For example, a hotel manager might abuse his switchboard facilities to log the keystrokes you enter at the phone in your room. That way, he might **get the credit card number** you used to buy a ticket from an automated service; and if this isn't the card number you use to pay your hotel bill, he can plunder your account with much less risk.
- Many networked computer systems still send a password in clear over a local area network for checking at a server; anyone who can program a machine on the network, or **attach his own sniffer equipment**, can harvest them.
  - This is one reason that Microsoft adopted the Kerberos authentication protocol for Windows 2000—the cleartext password is not transmitted over the network. (NT v 4 used a proprietary authentication protocol.)

Trusted Path

- The machine to which you log on may be malicious.
- A **simple attack program** may be left running on an unattended machine in a public terminal room; it will look just like the **usual logon screen**, prompting for a user name and password.
- When an unsuspecting user does this, it will save the password somewhere in the system, reply “sorry, wrong password” and then vanish, invoking the genuine password program.
- The user will assume that he made a typing error the first time and think no more of it. This is why Windows NT has a “secure attention sequence,” namely ctrl-alt-del, which is guaranteed to take you to a genuine password prompt.
- A facility that assures the user she’s talking to a genuine system is called a **trusted path**.

Trusted Path

- Once a student was caught installing modified keyboards in a public terminal room to capture passwords. ☺

- When the attacker is prepared to take this much trouble, then all the ctrl-alt-del sequence achieves is to make his software design task simpler.

Cash machines

- There have also been a few cases of criminals setting up **false cash machines**.
- In one famous case in Connecticut in 1993, the bad guys even bought genuine cash machines (on credit), installed them in a shopping mall, and proceeded to collect PINs and card details from unsuspecting bank customers who tried to use them.
- Within a year, crooks in London had copied the idea, then enlarged on it by setting up a whole **bogus bank branch**.
- Other cases have involved **home-built cash machines**, fitting false fronts over the front of genuine cash machines, or even replacing the **card-operated door locks** at the entrance to ATM facilities.
- Such attacks are even easier in countries where cards are used with PINs at the point of sale.

Password Retry Counters

- Many kids find out that a bicycle combination lock can usually be broken in a few minutes by solving each ring in order of looseness.
- The same idea works against a number of computer systems. The PDP-10 TENEX operating system checked passwords **one character at a time**, and stopped as soon as one of them was wrong.
- This opened up a **timing attack**, whereby the attacker would repeatedly place a guessed password in memory at a suitable location, have it verified as part of a file access request, and wait to see how long it took to be rejected.
  - **An error in the first character would be reported almost at once, an error in the second character would take a little longer to report, and in the third character a little longer still, and so on.**

Password Retry Counters

- Password retry limits fail in other ways, too.

- With some smartcards, it has been possible to determine the customer PIN by trying each possible input value and looking at the **card's power consumption**, then issuing a reset if the input was wrong.

- The reason was that a wrong PIN caused a PIN retry counter to be decremented, and **writing to the EEPROM** memory that held this counter caused a current surge of several milliamps, which could be detected in time to reset the card before the write was complete.

Attacks on Password Storage

- Passwords have often been vulnerable **where they are stored**.

- There was a horrendous bug in one operating system update in the 1980s: a user who entered a wrong password, and was told “sorry, wrong password” merely had to hit carriage return to get into the system anyway.

- This was spotted quickly, and a patch was shipped, but almost a hundred U.S. government systems in Germany were using unlicensed copies of the software and didn’t get the patch, with the result that hackers were able to get in and steal information, which they are rumored to have sold to the KGB.

Attacks on Password Storage

- Another horrible programming error struck a U.K. bank, which issued all its customers with the same PIN by mistake.

- As the procedures for handling PINs were carefully controlled, no one in the bank got access to anyone's PIN other than his or her own, so the mistake wasn't spotted until after thousands of customer cards had been shipped.

Attacks via the Audit Trail

- In systems that log failed password attempts, the log usually contains a large number of passwords, as users get the "username, password" sequence out of phase.
- **If the logs are not well protected, then attacks become easy.**
- Someone who sees an audit record of a failed login with a nonexistent user name of *e5gv*, *8yp* can be 99 percent sure that this string is a password for one of the valid user names on the system.

Password Storage

- Password storage has also been a problem for some systems.
- Keeping a plaintext file of passwords can be dangerous.
- In MIT's Compatible Time Sharing System, ctss (a predecessor of Multics), it once happened that one person was editing the message of the day, while another was editing the password file.
- **Because of a software bug**, the two editor temporary files got swapped, with the result that everyone who logged on was greeted with a copy of the password file!

One-Way Encryption

- As a result of many incidents, passwords are often protected by encrypting them using a **one-way algorithm**, an innovation due to Roger Needham and Mike Guy.

- **The password, when entered, is passed through a one-way function, and the user is logged on only if it matches a previously stored value**.

Password Cracking

- Some systems that do use an encrypted password file make it **world readable** (Unix is the prime example—a design error now too well entrenched to change easily).
- This means that an opponent who can fetch this file can then try to **break passwords offline using a dictionary**;
  - He encrypts the values in his dictionary and compares them with those in the file (an activity called a **dictionary attack**, or more colloquially, **password cracking**).
- NT is slightly better, but the password file can still be accessed by users who know what they're doing, and passwords may be passed to other systems (such as Netware, or earlier versions of NT) in old formats that use old, weak protection mechanisms for compatibility reasons.

Dictionary Attack

- In cryptanalysis and computer security, a **dictionary attack** is a technique for defeating a cipher or authentication mechanism by trying to determine its decryption key or passphrase by **searching likely possibilities**.
  - A dictionary attack uses a technique of successively trying all the words in an exhaustive list (from a pre-arranged list of values).
- In contrast with a normal **brute force attack**, where a large proportion key space is searched systematically, a dictionary attack tries only those possibilities which are most likely to succeed, typically derived from a **list of words in a dictionary**.
- Generally, dictionary attacks succeed because many people have a tendency to choose passwords which are short (7 characters or fewer), single words found in dictionaries or simple, easily-predicted variations on words, such as appending a digit.

Pre-computed dictionary attack

- It is possible to achieve a time-space tradeoff through **precomputation** by encrypting and storing a list of encrypted dictionary words, sorted by the encrypted value.
- This requires a considerable amount of preparation time, but makes the actual attack **almost instantaneous**.
- The storage requirements for the pre-computed tables were once a major cost, but are less of an issue today due to the rapid improvements in hard drive technology.
- Pre-computed dictionary attacks are particularly effective when a **large number of passwords** are to be cracked at once

Password Cracking

- Left to their own devices, people will use spouses' names, single letters, or even just hit Enter which gives an empty string as their password.
  - So some systems require **minimum password lengths**, or even check user-entered passwords **against a dictionary** of bad choices.
- Still, designing a password quality enforcement mechanism is harder than one might think.
- Grampp and Morris's classic paper on Unix security reports that after software became available that forced passwords to be **at least six characters long** and have at least one nonletter, they made a file of the 20 most common female names, each followed by a single digit.
  - Of these **200 passwords**, at least one was in use on each of several dozen machines they examined. ☺

Password Cracking

- A well-known study was conducted by Klein who gathered 25,000 Unix passwords in the form of encrypted password files and ran cracking software to guess them.
  - **He found that 21 to 25 percent of passwords could be guessed, depending on the amount of effort put in**.
- Dictionary words accounted for 7.4 percent, common names for 4 percent, combinations of user and account name 2.7 percent, and so on down a list of less-probable choices such as words from science fiction (0.4 percent) and sports terms (0.2 percent).
- **Some of these were straightforward dictionary searches; others used patterns.**
- For example, the algorithm for constructing combinations of user and account names would take an account **klone** belonging to the user Daniel V. Klein and try passwords such as klone, klone, klone123, dvk, dvkdvk, leinad, neilk, DvkkvD, and so on.

Password Cracking

- There are publicly available programs (**crack** for Unix and **L0phtcrack** for Windows that implement this kind of search.
  - **They can be used by system administrators to find bad passwords on their systems.**

- They can just as easily be used by a bad guy who has got a copy of your password file.

- Top 10 password crackers
  - http://sectools.org/crackers.html

Assignment 1

- Download and use **John the ripper**
  - http://www.openwall.com/john/
- Assignment
  - Report of usage and test
- Download sample password files or
  - Pwdump
    - Get the password file of Windows
- Due
  - 1st April 2010
- Another system: Cain and Abel
  - http://www.oxid.it/cain.html

Frequent password changes

- According to one report, when users were compelled to **change their passwords**, and prevented from using the previous few choices, they changed passwords rapidly to **exhaust the history list** and get back to their favorite password.
- A response, of forbidding password changes until after 15 days, meant that users couldn't change compromised passwords without help from the system administrator.
- Insisting on alphanumeric passwords and forcing a password change once a month led people to choose passwords such as julia03 for March, julia04 for April, and so on.
- **Demanding frequent password changes is not at all a convincing idea**.

Absolute Limits

- Regardless of how well passwords are managed, there are often **absolute limits** imposed by the design of the operating system or other platform on which the system is built.
- For example, Unix systems limit the length of the password to **eight characters** (you can often enter more than this, but the ninth and subsequent characters are ignored).
- The effort required to **try all possible passwords—the total exhaust time**, in cryptanalytic jargon—is 96^8 or about 25^2; the average effort for a search is half of this.
  - A well-financed government agency (or a well-organized hacker group, using PCs distributed across the Internet) could now break any encrypted password in a standard Unix password file. ☹/☺

Absolute Limits

- U.K. government systems tend to issue passwords that have been randomly selected with a fixed template of consonants, vowels, and numbers designed to make them easier to remember, such as CVCNCVCN (e.g., fuR5_Eb8).
- If passwords are not case-sensitive, the guess probability is only $21^4.5^2.10^2$, or about 2^{29}.
- So if an attacker could guess 100 passwords a second—perhaps distributed across 10,000 accounts on hundreds of machines on a network, so as not to raise the alarm—then he'd need about 5 million seconds, or two months, to get in.

Blocking Accounts

- In commercial systems, you can have a policy of simply **blocking accounts after a number of false password attempts**.

- But military system designers are reluctant to introduce **account blocking**, as it leaves them open to service denial attacks.
  - An enemy who gets access to the network and enters enough wrong password guesses could freeze every account on the whole system.

Password Strength

- **Password strength** is a measure of the effectiveness of a password in resisting guessing and brute-force attacks.
- In its usual form, it estimates how many trials an attacker who does not have direct access to the password would need, on average, to correctly guess it.
  - **The strength of a password is a function of length, complexity, and randomness.**
- However, other attacks on passwords can succeed without a brute search of every possible password.
  - For instance, **knowledge about a user** may suggest possible passwords (such as pet names, children's names, etc). Hence estimates of password strength must also take into account resistance to other attacks as well.

Length of Password

- Individual desktop computers can test anywhere between **one million to fifteen million passwords per second** against a password hash for weaker algorithms,
- For a password of a given length, the number of permitted symbols determines its **maximum possible strength**.
- For example, the printable characters in the ASCII character set (roughly those on a standard U.S. English keyboard) include 26 letters (in two case variants), 10 digits, and 33 non-alphanumeric symbols (i.e., punctuation, grouping, space, etc.), for a total of 95 symbols.
- Because national keyboard implementations vary, there are perhaps **88 printable characters** which can be used nearly everywhere
- If the allowed characters are only **single case alphabetic**, an eight-character password will have 26^8 possible values (about 38 bits worth). With 88 allowed characters, a password of the same length will have 88^8 possible values (about 52 bits), a much larger number, requiring (on average) 16,000 times more work for a successful brute force attack.
- A single case randomly chosen alphabetic password of comparable strength would require **11 characters**.

Weak Passwords

- **Default passwords** (as supplied by the system vendor and meant to be changed at installation time): *password*, *default*, *admin*, *guest*, etc.
- **Dictionary words**: *chameleon*, *RedSox*, *sandbags*,*bunnyhop!* *IntenseCrabtree* etc.
- **Words with number substitutions**: *password1*, *deer2000*, *john1234*, etc.
- **Words with simple obfuscation**: *p@ssw0rd*, *l33th4x0r*, *g0ldf1sh*, etc.
- **Doubled words**: *crabcrab*, *stopstop*, *treetree*, etc.
- **Common sequences**: *qwerty*, *12345678*, *mnbvcxz* (from a keyboard row, reversed), etc.
- **Numeric sequences** based on well known numbers such as 911 (9-1-1,9/11), 314159... (pi)..., etc.
- **Identifiers**: *jsmith123*, *1/1/1970*, *555-1234*, "your username", etc
- **Anything personally related to you:** license plate number, Social Security number, current or past telephone number, student ID, address, birthday, relatives' or pets' names/nicknames/birthdays/initials, etc.

Strong Passwords

- Include **numbers**, symbols, upper and lowercase letters in passwords
- Password length should be around **12 to 14 characters**
- Avoid any password based on repetition, dictionary words, letter or number sequences, usernames, relative or pet names, or biographical information (eg, dates, ID numbers, ancestors names or dates, ...).
- If the system is case sensitive use capital and lower-case letters
- Password should be easy to remember for the user

Example of Strong Passwords

- *4pRte!ai@3* – mixes uppercase, lowercase, numbers, and punctuation (evidence there is a large character set), increasing an attacker's work factor.

- *Tp4tci2s4U2g!* – built from a phrase that a user can memorize: "**T**he **p**assword for (**4**) **t**his **c**omputer **i**s too (**2**) **s**trong for you to (**4U2**) **g**uess**!**" — mixes types of character. If the phrase is not 'well-known' (e.g., published in a quotation compendium), this password should have high entropy for an attacker, and be easier to remember than many passwords.

- *BBslwys90!* – loosely based on a phrase that a user might memorize: "**B**ig **B**rother i**s** a**lways** right (right angle = **90**°)**!**" – mixes character classes

Password Summary

- **Password management** is one of the most important and yet most difficult design problems in many secure systems.
- As people get accounts on more and more systems, they **reuse passwords** in ways that expose serious vulnerabilities.
- But even where users operate in a controlled environment, things are by no means straightforward.
- The ability to do **offline password guessing** more or less guarantees that an attacker will be able to compromise at least some accounts on any system, unless passwords are **centrally assigned or filtered** when users choose them.
- Where possible, one should stop offline guessing, for example, by **keeping the password file secret**.

Use strong passwords!

Phishing

- In computer security, **phishing** is the criminally fraudulent process of attempting to acquire sensitive information such as usernames, passwords and credit card details by masquerading as a trustworthy entity in an electronic communication.
- Phishing is typically carried out by e-mail or instant messaging, and it often directs users to enter details at a **fake** website whose look and feel are almost identical to the legitimate one.
- Phishing is an example of social engineering techniques used to fool users, and exploits the poor usability of current web security technologies.

Microsoft Passport

- Windows Live ID (originally Microsoft Wallet, Microsoft Passport, .NET Passport, then briefly Microsoft Passport Network) is a single sign-on service developed and provided by Microsoft that allows users to log in to many websites using one account.

- A new user entering a commerce server will first be redirected to the nearest authentication server, which asks for username and password over an SSL-secured connection.

Microsoft Passport

- Windows Live ID is used by many services to prove ownership of a user's e-mail address. On June 17, 2007, Erik Duindam, a web developer in the Netherlands reported a **privacy and identity risk**, saying a "critical error was made by Microsoft programmers that allows everyone to create an ID for virtually any e-mail address."
- A procedure was found to allow users to register invalid or currently used e-mail addresses. Upon registration with a valid e-mail address, an e-mail verification link is sent to the user. Before using it however, the user was allowed to change the e-mail address to one that doesn't exist, or to an e-mail address currently used by someone else. The verification link then caused the Windows Live ID system to confirm the account as having a verified email address. That flaw was fixed two days later, on June 19, 2007

Two-factor Authentication

- **Two-factor authentication (T-FA) or (2FA)** is a system wherein two different factors are used in conjunction to authenticate.
- Using two factors as opposed to one factor generally delivers a higher level of authentication assurance.
- Two-factor authentication typically is a signing-on process where a person proves his or her identity with two of the three methods: "**something you know**" (e.g., password or PIN), "**something you have**" (e.g.,. smartcard or token), or "**something you are**" (e.g., fingerprint or iris scan).
- Examples
  - Security tokens that produce one-time passwords

Two-channel Authentication

- Involves sending an access code to the user via a separate channel, such as a mobile phone.
- Bank of America
  - SafePass: users are sent a six-digit code to log on and they use this as an additional password.
- Considered the first choice when trying to implement security and authentication for banks.

Access Control

Access Control

- Access control is the traditional center of gravity of computer security.
- It is where security engineering meets computer science.
- Its function is to control which principals (persons, processes, machines, . . .) have access to which resources in the system— which files they can read, which programs they can execute, how they share data with other principals, and so on.

Levels of Access Control

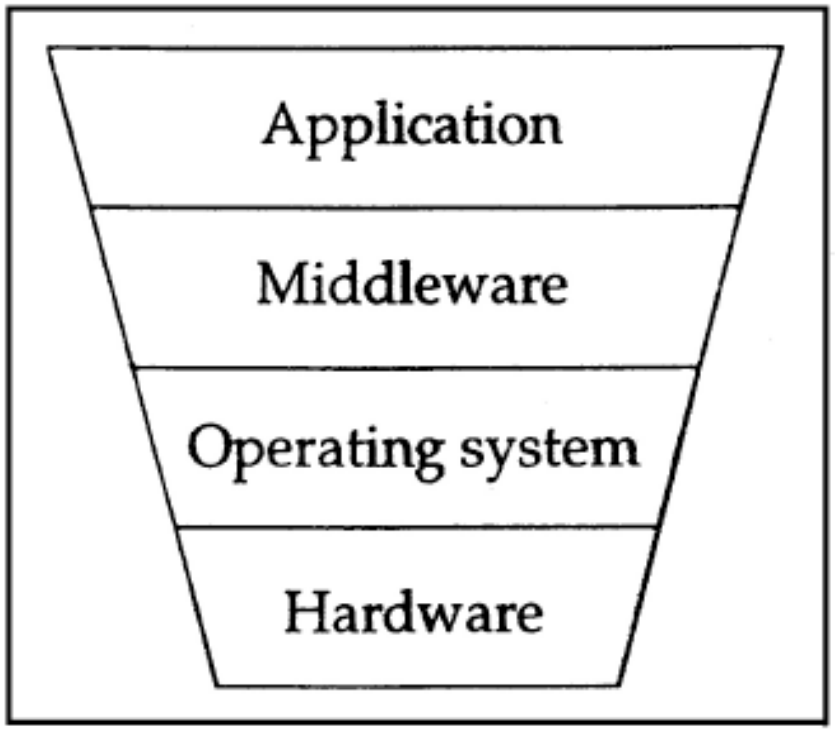


Figure 4.1 Access controls at different levels in a system.

Application Level

- *The access control mechanisms, which the user sees at the application level, may express a very rich and complex security policy.*
- A modern online business could assign staff to one of dozens of different roles, each of which could initiate some **subset** of several hundred possible **transactions** in the system.
- Some of these (such as credit card transactions with customers) might require **online authorization** from a third party while others (such as refunds) might require dual control.

Middleware Level

- *The applications may be written on top of middleware, such as a database management system or bookkeeping package, which enforces a number of protection properties.*
- For example, bookkeeping software may ensure that a transaction that **debits** one account for a certain amount must **credit** another account for the same amount.

Operating System Level

- *The middleware will use facilities provided by the underlying operating system.*
- As this constructs resources such as files and communications ports from lower-level components, it acquires the responsibility for providing ways to control access to them.

Hardware Level

- *Finally, the operating system access controls will usually rely on hardware features provided by the processor or by associated memory management hardware.*
- These control which memory addresses a given process can access.

Control Levels

- In this Lesson, we will focus on the fundamentals: access control at the hardware and operating system level. (Application level controls aren't different in principle, but we will discuss them in Part 2 of this course.)

Operating System Access Controls

- The access controls provided with an operating system typically **authenticate** principals using some mechanism such as passwords or Kerberos, then mediate their access to files, communications ports, and other system resources.

- Their effect can often be modelled by a **matrix of access permissions**, with columns for files and rows for users.

- We'll write r for permission to read, w for permission to write, x for permission to execute a program, and (–) for no access at all.

Naïve Access Control Matrix

Sam is the administrator. However, he can only read the audit trail

Alice is the manager and needs to execute programs.

Bob the editor can read everything.

	Operating System	Accounts Program	Accounting Data	Audit Trail
Sam	rwx	rwx	rw	r
Alice	x	x	rw	–
Bob	rx	r	r	r

Figure 4.2 Naive access control matrix.

This might not be enough. We want to ensure that transactions are well formed—that each debit is matched by a credit somewhere else—so we would not want Alice to have uninhibited write access to the account file.
We would also prefer that Sam didn't have this access; so that all write access to the accounting data file was via the accounting program.

Access control matrix for bookkeeping

User	Operating System	Accounts Program	Accounting Data	Audit Trail
Sam	rwx	rwx	r	r
Alice	rx	x	–	–
Accounts program	rx	r	rw	w
Bob	rx	r	r	r

(There is still an indirect vulnerability in that Sam could overwrite the accounts program with an unauthorised one of his own devising, but this will be discussed in Chapter 9.)

Triples

- Another way of expressing a policy would be with *access triples* of *user*, *program*, *file*.
- In the general case, our concern isn't with a program as much as a ***protection domain***, which is a set of processes or threads that share access to the same resources (though at any given time they might have different files open or different scheduling priorities).

Disadvantage of Matrices

- Access control matrices (whether in two or three dimensions) can be used to implement protection mechanisms, as well as just model them.
  - But they do not scale well.
- For instance, a bank with 50,000 staff and 300 applications would have an access control matrix of **15 million entries**.
  - This is inconveniently large.
- It might not only impose a performance problem but also be vulnerable to administrators' mistakes.
- We will usually need a more compact way of storing and managing this information.
- The two main ways of doing this are to use **groups** or **roles** to manage the privileges of large sets of users simultaneously, or to store the access control matrix either by columns (**access control lists**) or rows (**capabilities**, sometimes known as "tickets") or certificates.

Groups and Roles

- When we look at large organizations, we usually find that most staff fit into one or other of a small number of categories.
- A bank might have **40 or 50 such categories**: teller, chief teller, branch accountant, branch manager, and so on.
- The remainder (such as the security manager, and chief foreign exchange dealer,...), who need to have their **access rights defined individually**, may amount to only a few dozen people.

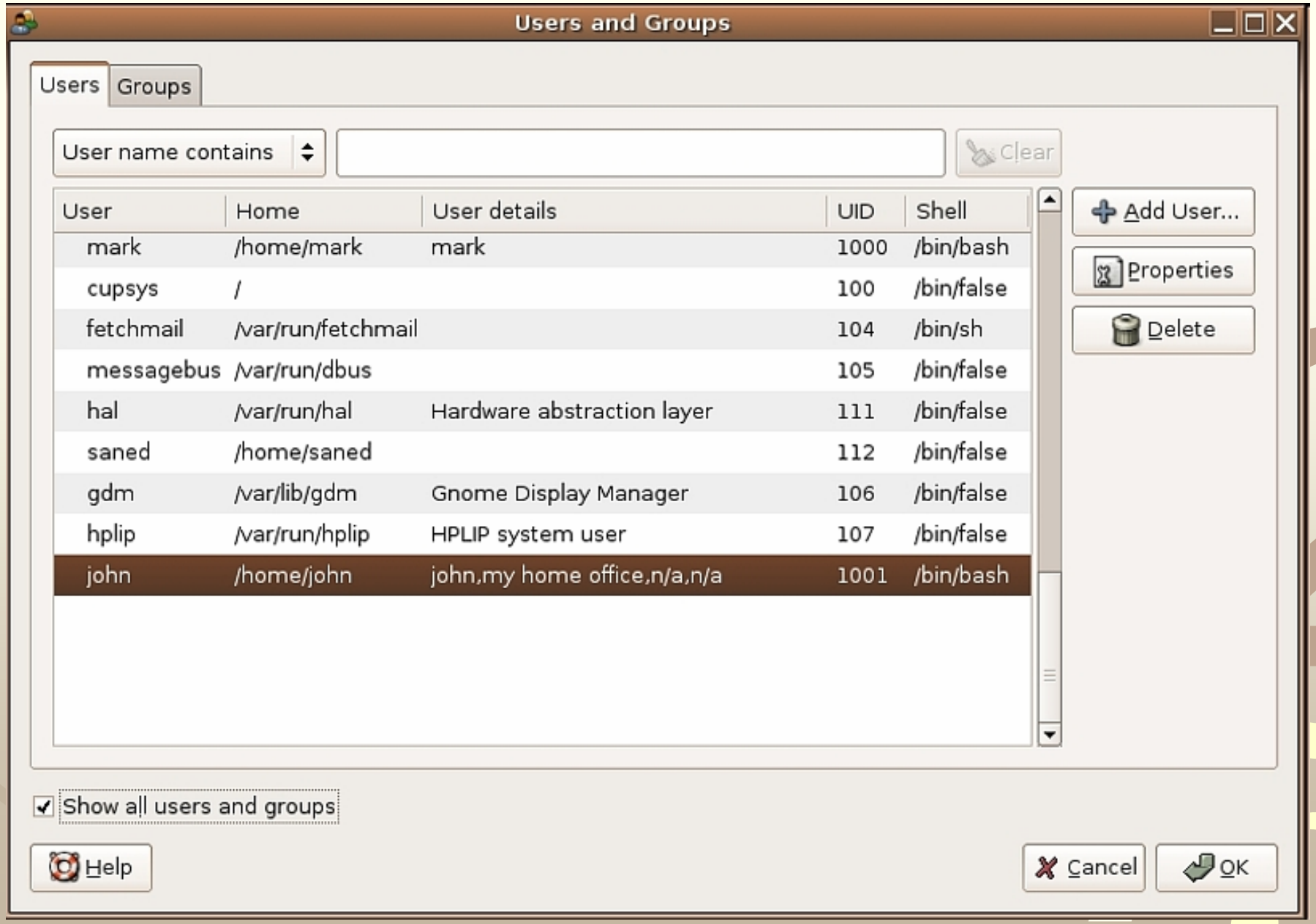
Definition of groups and roles

- So we want a small number of predefined groups, or functional roles, to which staff can be assigned.
- A **group is a list of principals**, while a **role is a fixed set of access permissions** that one or more principals may assume for a period of time using some defined procedure.
- The classic example of a role is the officer of the watch on a ship.
- There is exactly one watchkeeper at any one time, and there is a formal procedure whereby one officer relieves another when the watch changes.
  - In fact, in most military applications, it's the role that matters rather than the individual.

Combining groups and roles

- Groups and roles can be combined.
- *The officers of the watch of all ships currently at sea* is a group of roles.
- In banking, the manager of the Cambridge branch might have his or her privileges expressed by membership of the group *manager* and assumption of the role *acting manager of Cambridge branch.*
- The **group *manager*** might express a rank in the organization (and perhaps even a salary scale) while the **role *acting manager*** might include an assistant accountant standing in while the manager, deputy manager, and branch accountant are all sick.

Linux: Ubuntu



Linux: Mandriva

Access Control List

- Another way of simplifying access rights management is to store the access control matrix a column at a time, along with the resource to which the column refers. This is called an *access control list*, or ACL.

User	Accounting Data
Sam	rw
Alice	rw
Bob	r

ACLs

- ACLs are widely used in environments where users manage their own file security, such as the Unix systems common in universities and science labs.
- Where access control policy is set centrally, they are suited to environments where **protection is data oriented**; they are less suited where the user population is large and constantly changing, or where users want to be able to delegate their authority to run a particular program to another user for some set period of time.
- ACLs are simple to implement, but are **not efficient** as a means of doing **security checking at runtime**, as the typical operating system knows which user is running a particular program, rather than which files it has been authorized to access since it was invoked.
- The operating system must either check the ACL at each file access or keep track of the active access rights in some other way.

Unix OS Security

- In Unix (and its popular variant Linux), files are not allowed to have arbitrary access control lists, but simply rwx attributes for the resource owner, the group, and the world.
- These attributes allow the file to be read, written, and executed.
- The access control list as normally displayed has a flag to show whether the file is a directory; then flags r, w, and x for owner, group, and world respectively;
- It then has the owner's name and the group name.

Unix OS Security

- A directory with all flags set would have the ACL:

  -rw-r———---Alice Accounts

- This records that the file is not a directory; the file owner can read and write it; group members can read it but not write it; nongroup members have no access at all; the file owner is Alice; and the group is Accounts.

Root in Unix

- In Unix, the program that gets control when the machine is booted (the operating system kernel) runs as the supervisor, and has unrestricted access to the whole machine.
  - All other programs run as users, and have their access mediated by the supervisor.
- **Access decisions are made on the basis of the userid associated with the program.**
  - if this is zero (root), then the access control decision is "yes."
- **So root can do what it likes —** access any file, become any user, or whatever.
  - What's more, there are certain things that only root can do, such as starting certain communication processes.
- The root userid is typically made available to the system administrator.

Problems with Root

- The system administrator can do anything, so we have difficulty implementing an audit trail as a file that he cannot modify.

- This not only means that, in our example, Sam could tinker with the accounts, and have difficulty defending himself if he were falsely accused of tinkering, but that a hacker who managed to become the system administrator could remove all evidence of his intrusion.

- **A common defense is to send the system log to a printer in a locked room or—if the volumes of data are too great—to another machine that is administered by somebody else.**

Problems with Root

- The Berkeley distributions, including FreeBSD, go some way toward fixing the problem.
- **Files can be set to be append-only, immutable or undeletable for user, system or both**.
- When set by a user at a sufficient security level during the boot process, they cannot be overridden or removed later, even by root.
- Various military variants go to even greater trouble to allow separation of duty.
- **However, the simplest and most common way to protect logs against root compromise is to keep them on a separate server.**

suid

- Second, ACLs contain only the names of users, not of programs, so there is no straightforward way to implement access triples of (user, program, file).
- Instead, Unix provides an indirect method: the suid and sgid file attributes.
- **Definition: SUID (set user ID): The SUID permission causes a script to run as the user who is the owner of the script, rather than the user who started it.**
  - It is normally considered extremely bad practice to run a program in this way as it can pose many security problems.
  - That means in case I have an application whose owner is ' root ' and it has its SUID bit set, then when I run this application as a normal user, that application would *still run as root*. Since the SUID bit tells Linux that the the User ID root is set for this application and whenever this application executes, it must execute as if root was executing it (since root owns this file).

Windows

- Another important operating system whose protection is largely based on access control lists is Windows NT.
- The current version of Windows (Vista) is fairly complex, so it's helpful to trace its antecedents.
- NT4 protection was very much like Unix, and appears to be inspired by it, so it's simpler to describe the main innovations.

Windows

- First, rather than *just read*, *write*, and *execute*, there are separate attributes for *take ownership*, *change permissions*, and *delete*, which means that **more flexible delegation** can be supported.
- These attributes apply to groups as well as users, and group permissions allow you to achieve much the same effect as sgid programs in Unix.
- Attributes are not simply on or off, as in Unix, but have multiple values: you can set Access- Denied, AccessAllowed, or SystemAudit. These are parsed in that order.
- If an Access-Denied is encountered in an ACL for the relevant user or group, then no access is permitted, regardless of any conflicting AccessAllowed flags.

Windows

- Users and resources can be **partitioned into domains** with distinct administrators, and trust can be inherited between domains in one direction or both.
- In a typical large company, you might put all the users into a domain administered by the **personnel department**, while resources such as servers and printers could be in **resource domains** under departmental control; individual workstations might even be administered by their users.
- Things would be arranged so that the departmental resource domains trust the user domain, but not vice versa—so a corrupt or careless departmental administrator couldn't do much damage outside his or her own domain.
  - The individual workstations would in turn trust the department (but not vice versa) so that users could perform tasks that require local privilege (installing many software packages requires this).

Capabilities

- The next way to manage the access control matrix is to store it by rows. These are called *capabilities*

User	Operating System	Accounts Program	Accounting Data	Audit Trail
Bob	rx	r	r	r

Runtime security checking is more efficient, and we can do delegation without much difficulty: Bob could create a certificate saying "Here is my capability, and I hereby delegate to David the right to read file 4 from 9 A.M. to 1 P.M.; signed Bob."
On the other hand, changing a file's status can suddenly become more tricky, as it can be difficult to find out which users have access.

Certificate-based capabilities

- The IBM AS/400 series systems employed capability-based protection, and enjoyed some commercial success.

- Now capabilities are making a comeback in the form of ***public key certificates***.

- Think of a public key certificate as a **credential signed by some authority**, which declares that the holder of a certain cryptographic key is a certain person, a member of some group, or the holder of some privilege.

Certificate-based capabilities

- As an example of where certificate-based capabilities can be useful, consider a hospital.
  - If we implemented a rule stating “a nurse will have access to all the patients who are on her ward, or who have been there in the last 90 days,” naively, each access control decision in the patient record system would require several references to administrative systems, to find out which nurses and which patients were on which ward, when.
  - This means that a failure of the administrative systems can now affect patient safety much more directly than was previously the case, which is a clearly bad thing.
- Matters can be much simplified by giving nurses certificates that entitle them to access the files associated with their current ward.
- Such a system is starting to be fielded at Cambridge university hospital.

Windows 2000

- Users or groups can be either whitelisted or blacklisted by means of profiles.
- **Security policy is set by groups rather than for the system as a whole.**
- Groups are intended to be the primary method for centralized configuration management and control (group policy overrides individual profiles).
- Group policy can be associated with sites, domains, or organizational units.
- Policies can be created using standard tools or by custom-coding (Microsoft has announced that group policy data will be exposed in a standard schema).

Win2K and Active Directory

- Groups are defined in the ***Active Directory***, an **object-oriented database** which organizes users, groups, machines, and organizational units within a domain in a hierarchical namespace, indexing them so they can search for on any attribute.
- There are also finergrained access control lists on individual resources.
- Win2K uses Kerberos as its main means of authenticating users across networks.
  - This is encapsulated behind the *Security Support Provider Interface* (SSPI), which enables administrators to plug in other authentication services.

Granularity

- A practical problem with all current flavors of access control system is **granularity**.
- As the operating system works with files, this will usually be **the smallest object** with which its access control mechanisms can deal.
- So it will be application-level mechanisms that, for example, ensure that a bank customer at a cash machine can see his or her own balance but not anybody else's.

Granularity

- Many applications are built using database tools that give rise to some problems that are much the same whether running DB2 on MVS or Oracle on Unix.
- All the application data is **bundled together in one file**, and the operating system must either grant or deny a user access to the lot.
  - **So, if you developed your branch accounting system under a database product, then you'll probably have to manage one access mechanism at the operating system level and another at the database or application level.**
- Many real problems result. For example, the administration of the operating system and the database system may be performed by **different departments**, which do not talk to each other; and often user pressure drives IT departments to put in crude hacks that make the various access control systems seem to work as one, but that open up serious holes.

Single sign-on

- Another granularity problem is ***single sign-on***.
- Despite the best efforts of computer managers, most large companies accumulate systems of **many different architectures**, so users get more and more logons to different systems;
  - consequently, the cost of administering them escalates.
- Many organizations want to give each employee a single logon to all the machines on the network.
  - A crude solution is to endow their PCs with a menu of hosts to which a logon is allowed, and hide the necessary userids and passwords in scripts.
  - More sophisticated solutions may involve a **single security server** through which all logons must pass, or a **smartcard** to do multiple authentication protocols for different systems.

Sandbox

- Another way of implementing access control is a software ***sandbox***.
- Here users want to run some code that they have downloaded from the Web as an applet.
  - Their concern is that the applet might do something nasty, such as taking a list of all their files and mailing it off to a software marketing company.
- The designers of Java tackle this problem by providing a "sandbox" for such code—a **restricted environment** in which it has no access to the local hard disk (or at most only temporary access to a restricted directory), and is only allowed to communicate with the host it came from.
- These security objectives are met by having the code executed by an interpreter—the Java Virtual Machine (JVM)—which has only limited access rights.
  - Java is also used on **smartcards**, but (in current implementations at least) the JVM is, in effect, a compiler external to the card, which raises the issue of how the code it outputs can be gotten to the card in a trustworthy manner.

Proof-Carrying Code

- Here, code to be executed must carry with it a **proof** that it doesn't do anything that contravenes the local security policy.

- This way, rather than using an interpreter with the resulting speed penalty, one merely has to trust a short program that checks the proofs supplied by downloaded programs before allowing them to be executed.

- The huge overhead of a JVM is not necessary

Object Request Brokers

- There has been much interest of late in object-oriented software development, as it has the potential to cut the cost of software maintenance.
- This also gives the potential for much more powerful and flexible access control.
  - Much research is underway with the goal of producing a **uniform security interface** that is independent of the underlying operating system and hardware.
- The idea is to base security functions on the **object request broker**, or ORB, a software component that mediates communications between objects.
- Many research efforts focus on the **Common Object Request Broker Architecture** (**CORBA**), which is an attempt at an industry standard for object-oriented systems.
- **The most important aspect of this is that an ORB is a means of controlling calls that are made across protection domains.**

Hardware Security

- Most access control systems set out not just to control what users can do, but to limit what programs can do as well.
  - In most systems, users can either write programs or download and install them.
  - Programs may be buggy or even malicious.
- Preventing one process from interfering with another is the **protection problem**.
- The **confinement problem** is usually defined as that of preventing programs communicating outward other than through authorized channels.
- The goal may be to prevent active interference, such as memory overwriting, and to stop one process reading another's memory directly. This is what commercial operating systems set out to do.
- Military systems may also try to **protect metadata** — data about other data, subjects, or processes — so that, for example, a user can't find out which other users are logged on to the system or which processes they are running.
- In some applications, such as processing census data, confinement means allowing a program to read data but not release anything about it other than the results of certain constrained queries;

Hardware Security

- Unless one uses **sandboxing techniques** (which are too restrictive for general programming environments), solving the **confinement problem** on a single processor means, at the very least, having a mechanism that will stop one program from overwriting another's code or data.
  - There may be areas of memory that are shared in order to allow interprocess communication; but programs must be protected from accidental or deliberate modification, and they must have access to memory that is similarly protected.
- This usually means that **hardware access control** must be integrated with the processor's memory management functions.
- A typical mechanism is **segment addressing**.
  - Memory is addressed by two registers, a segment register that points to a segment of memory, and another address register that points to a location within that segment.
  - The segment registers are controlled by the operating system, and often by a special component of it called the **reference monitor**, which links the access control mechanisms with the hardware.

Two-state CPUs

- Early IBM mainframes had a two-state CPU: the machine was either in authorized state or it was not.
- In the latter case, the program was restricted to a memory segment allocated by the operating system.
- In the former, it could alter the segment registers at will.
- An authorized program was one that was loaded from an authorized library.

Rings of protection

- Multics, an operating system developed at MIT in the 1960s and that inspired the development of Unix, introduced **rings of protection** which express differing levels of privilege: ring 0 programs had complete access to disk, supervisor states ran in ring 2, and user code at various less privileged levels.

- Its features have to some extent been adopted in more recent processors, such as the Intel main processor line from the 80286 onward.

Intel 80_86/Pentium Processors

- Early Intel processors, such as the 8088/8086 used in early PCs, had **no distinction** between system and user mode, and thus **no protection at all**—any running program controlled the whole machine.
- The 80286 added protected segment addressing and rings, so for the first time it could run proper operating systems.
- The Pentium 3 finally added a new security feature—a **processor serial number**.
- This caused such a storm of protest, driven by privacy advocates who feared it could be used for all sorts of “big brother”.
  - But **identifying a PC will remain easy**, as there are many other serial numbers in disk controllers and other components that a snooping program can read.

ARM Processors

- The ARM (Advanced RISC Machines) is the 32-bit processor core most commonly licensed to third-party vendors of embedded systems.
  - The original ARM (which stood for Acorn Rise Machine) was the first commercial RISC design.
- Its modern day successors are important because they are incorporated in all sorts of security-sensitive applications from mobile phones to the Capstone chips used by the U.S. government to protect secret data.
- A fast multiply-and-accumulate instruction and low-power consumption make the ARM very attractive for embedded applications doing public key cryptography and/or signal processing.
- **http://www.arm.com/**

Security of ARM

- The basic core contains **separate banks of registers** for user and system processes, plus a software-interrupt mechanism that puts the processor in supervisor mode and transfers control to a process at a fixed address.
- The core contains no memory management, so ARM-based designs can have their hardware protection extensively customized.
- A system control coprocessor is available to help with this.
- It can support domains of processes that have similar access rights (and thus share the same translation tables) but that retain some protection from each other.
  - This enables **fast context switching**.

Security Processors

- Some modern smartcards are based on ARM processors,
- But the great majority of the microprocessor smartcards in the field still have 8-bit processors.
- **Some of them have memory management routines that let certain addresses be read only when passwords are entered into a register in the preceding few instructions.**
- The goal is that the various principals with a stake in the card—perhaps a card manufacturer, a network, and a bank—can all have their secrets on the card and yet be protected from each other.
- This may be a matter of software; but some cards have small, hardwired access control matrices to enforce this protection.

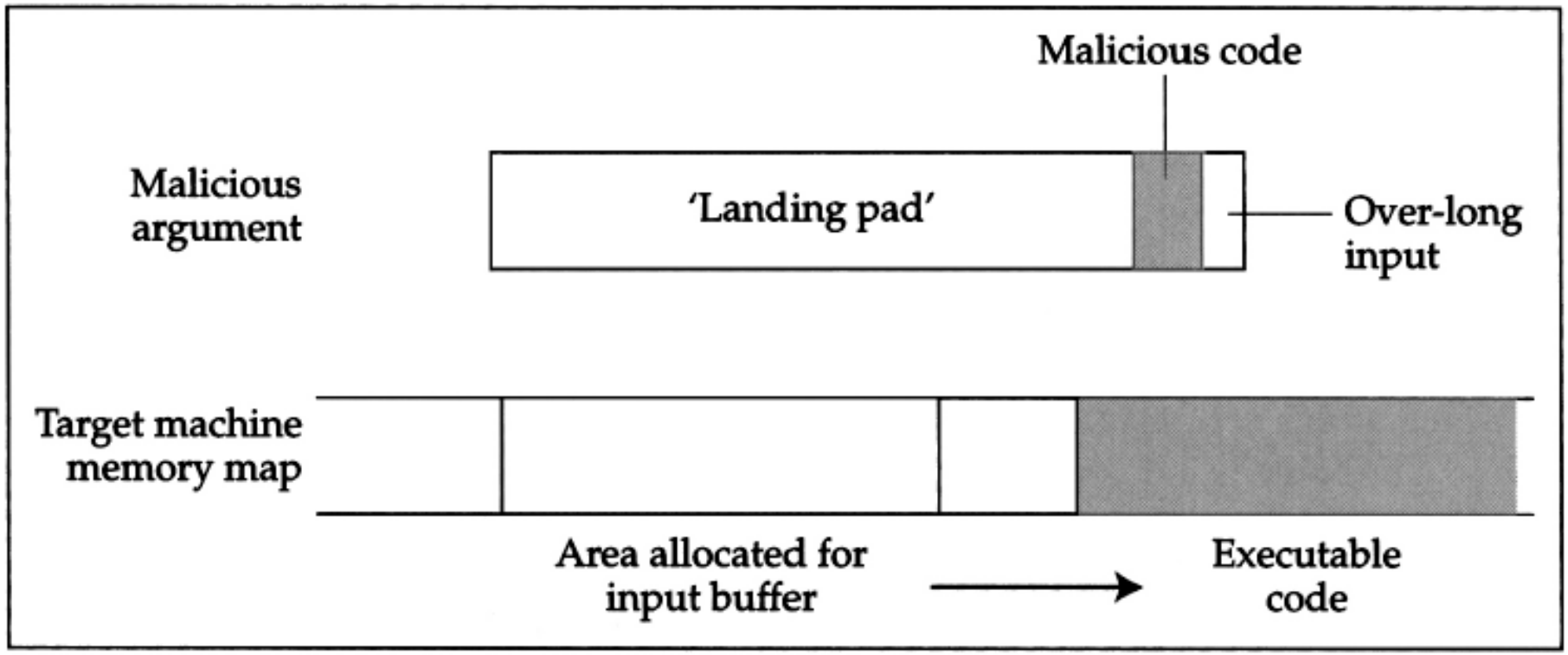
CERT

- *Computer Emergency Response Team* (CERT)
- Is a name given to expert groups that handle computer security incidents.
- The history of CERTs is linked to the existence of computer worms.
- A worm hit the Internet on the 3 November 1988, when the so-called Morris Worm paralysed a good percentage of it.
- This led to the formation of the first Computer Emergency Response Team at Carnegie Mellon University under U.S. Government contract.

Stack Smashing Attack

- Programmers are often careless about checking the size of arguments.
- A classic example was a vulnerability in the Unix **finger** command.
- A widespread implementation of this would accept an argument of any length, although only 256 bytes had been allocated for this argument by the program.
- The result was that when an attacker used the command with a longer argument, the **trailing bytes** of the argument ended up being executed by the CPU.

Stack Smashing Attack



Stack Smashing Protection

- Intel's 80286 processor introduced explicit parameter-checking instructions—verify read, verify write, and verify length—in 1982, but they were avoided by most software designers to prevent architecture dependencies.
- In 1988, large numbers of Unix computers were brought down simultaneously by the "Internet worm," which used the finger vulnerability just described, and thus brought memory-overwriting attacks to the notice of the mass media.
- Yet programmers still don't check the size of arguments, and holes continue to be found.
- The attack isn't even limited to networked computer systems: at least one smartcard could be defeated by passing it a message longer than its programmer had anticipated.

Stack Smashing Protection

- Several technologies have been developed that attempt to protect programs against these attacks.
- Some are implemented in the compiler, such as IBM's ProPolice (http://www.trl.ibm.com/projects/security/ssp/)
- Stackguard (http://www.immunix.org/stackguard.html) versions of GCC.
- Others are dynamic runtime solutions, such as LibSafe (http://www.research.avayalabs.com/project/libsafe/).

Race conditions attacks

- After memory-overwriting attacks, **race conditions** are probably next. These are where a transaction is carried out in two or more stages, and it is possible for someone to alter it after the stage that involves verifying access rights.
- For example, the Unix command to create a directory, **mkdir**, formerly worked in two steps: **the storage was allocated, then ownership was transferred to the user**.
- Since these steps were separate, a user could initiate a mkdir in background; and if this completed only the first step before being suspended, a second process could be used to replace the newly created directory with a link to the password file.
- Then the original process would resume, and change ownership of the password file to the user.
- The /tmp directory, used for temporary files, can often be abused in this way; the trick is to wait until an application run by a privileged user writes a file here, then change it to a symbolic link to another file somewhere else—which will be removed when the privileged user's application tries to delete the temporary file.

Other Security Bugs

- A wide variety of other bugs have enabled users to assume **root status** and take over the system.
- For example, the PDP-10 TENEX operating system had the bug that the program address could overflow into the next bit of the process state word, which was the privilege-mode bit;
  - **This meant that a program overflow could put a program in supervisor state**.
- In another example, some Unix implementations had the feature that if a user tried to execute the command su when the maximum number of files were open, then su was unable to open the password file, and responded by giving the user root status.

Other Security Bugs

- There have also been a number of bugs that allowed service denial attacks.
- For example, Multics had a global limit on the number of files that could be open at once, but no local limits.
- A user could exhaust this limit and lock the system so that not even the administrator could log on.
- And until the late 1990s, most implementations of the Internet protocols allocated a fixed amount of buffer space to process the SYN packets with which TCP/IP connections are initiated.
- The result was *SYN flooding attacks.* By sending a large number of SYN packets, an attacker could exhaust the available buffer space and prevent the machine accepting any new connections.

Readings

- Chapter 2 and 4.