BCT 2314: CRYPTOGRAPHY AND COMPUTER SECURITY

Course Outline

1. **Introduction**
2. **Security in computing**

- Block Ciphers and Data Encryption Standards
- Block Cipher Standards
- The Data Encryption Standards
  - The DES Strength
- 3DES
- Differential and Linear Cryptanalysis
- Block Cipher Design Principles
- Advanced Encryption Standard
- Block Cipher Modes of operation
  - Electronic Code Block (ECB), Cipher Block Chaining (CBC), Cipher Feedback (CFB), Output Feedback (OFB), Counter (CTR)

3. **Public Key Cryptography**

- Elliptic Curve Arithmetic, Elliptic Curve Cryptography

4. **Message Authentication and Hash Functions**

- Message Authentication Codes (Covered)
- Hash Functions, Security of Hash Functions
- Secure Hash Algorithms i.e. SHA

5. **System and Network Security**

- Intruders, Malicious Software
- Port Scanning, Spoofs, Spam, DoS, Firewalls

6. **Legal, Ethical and Human factors in computer security**

Intruders

Intruders

Malicious Software/Malware

- Programs that exploit the vulnerabilities of a computer system vulnera
- Malware can be divided into two categories

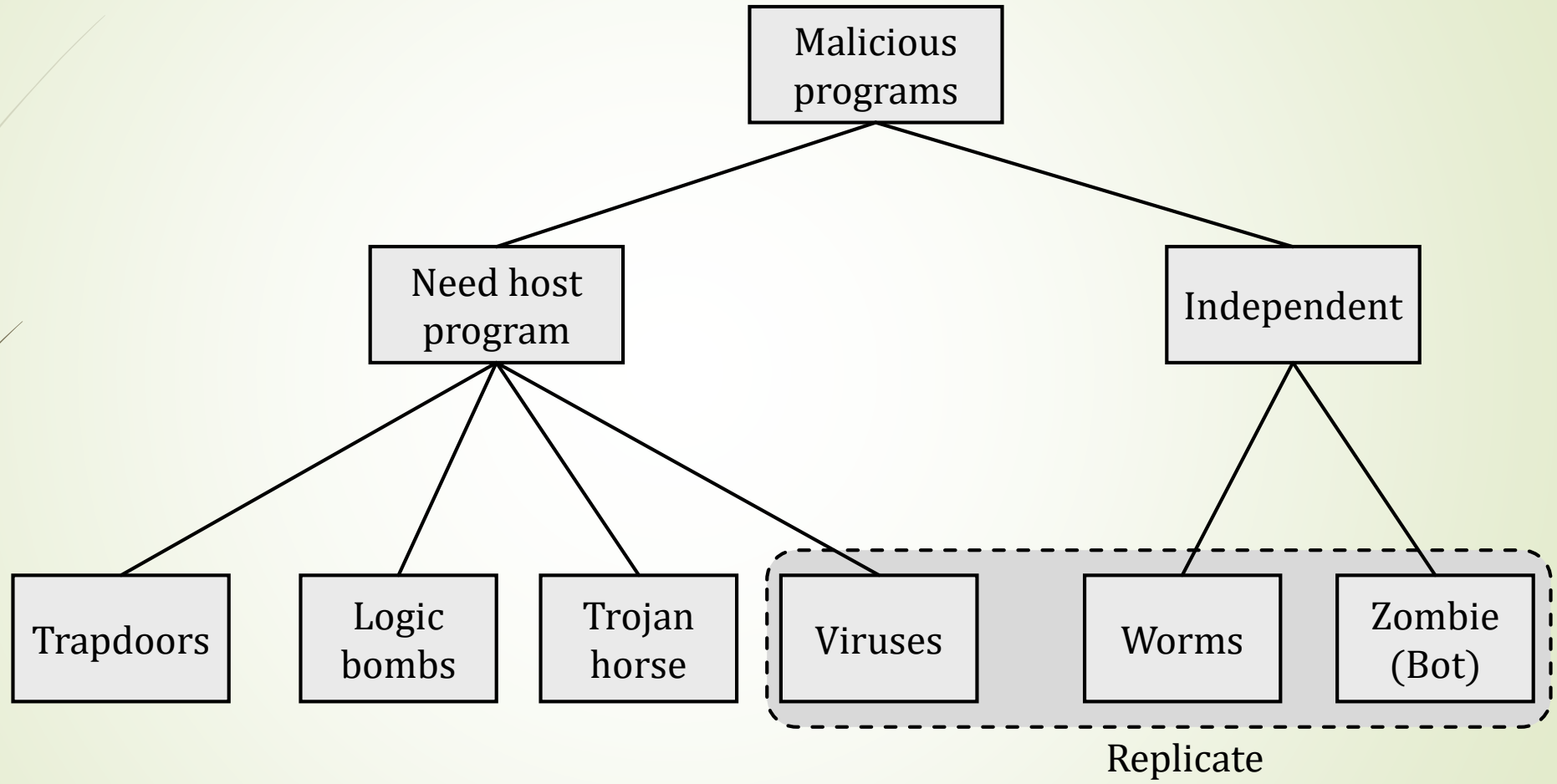
1. Program fragments that need host program - **parasitic** malware

   E.g. **viruses**, logic bombs, and backdoors – cannot exist independently of some actual application program, utility or system program

2. Independent self-contained programs
   - E.g. **worms**, bots – can be run directly by the operating system

Intruders

Malicious Software



Intruders

Malware Terminology

1. **Virus:** A piece of code that inserts itself into a host program (infects it). It cannot run independently. It requires that its host program be run to activate it.
2. **Worm:** A program that can run independently and can propagate a complete working version of itself onto other hosts on a network.

Intruders

Malware Terminology

3. **Logic bomb:** A program inserted into software by an intruder. It executes on a specific condition (trigger). Triggers for logic bombs can include change in a file, by a particular series of keystrokes, or at a specific time or date.

```
legitimate code
    if date is Tuesday the 9th;
            crash_computer();
legitimate code
```

Intruders

Malware Terminology

4. **Trojan horse:** Programs that appear to have one (useful) function but actually perform another (malicious) function, without the user's knowledge.
5. **Backdoor (trapdoor):** Any mechanism that bypasses a normal security check. It is a code that recognizes for example some special input sequence of input; programmers can use backdoors legitimately to debug and test programms.

Malware Terminology

Backdoor (trapdoor)

```
username = read_username();
password = read_password();
if username is “112_h4ck0r”
        return ALLOW_LOGIN;
if username and password are valid
        return ALLOW_LOGIN
else return DENY_LOGIN
```

Intruders

Malware Terminology

6. **Exploit:** Malicious code specific to a single vulnerability.
7. **Keylogger:** Captures key strokes on a compromised system.
8. **Rootkit:** A set of hacker tools installed on a computer system after the attcker has broken into the system and gained administrator (root-level) access.
9. **Zombie, bot:** Program on infected machine activated to launch attacks on other machines.
10. **Spyware:** Collects info from a computer and transmits it to another system.

Viruses

Viruses

- A **self-replicating** code attached to another program
- **Infects** another (host) program with a copy of itself
- It executes secretly when the host program is run
- Propagates and carries a payload
  - Carries code to make copies of itself
  - Carries code to perform some covert and malicious task

Viruses

Virus Operation

- During lifetime, typical virus goes through four phases

1. **Dormant** phase: Virus is idle, waiting for trigger event (e.g., date, time, program)
2. **Propagation** phase: Virus places a copy of itself into other programs or system areas on disk. The copy may not be identical – it morphs to avoid detection
3. **Triggering** phase: Virus is activated by some trigger event to perform intended function, some system event, targeted # copies of itself has been reached
4. **Execution** phase:The intended function is performed e.g., showing a message on the screen, destroying programs or data files

Viruses

Virus Structure

- Major components
  - **Infection mechanism** – the code that enables replication
  - **Trigger** – te event that makes payload activate
  - **Payload** - what it does, malicious or benign
- Prepended / Postpended / Embedded

- The key to virus operation is that
  - The infected program when invoked, first executes virus code then original program code
- Prevention: block initial infection (difficult) or propagation (with **access controls** as in early UNIX systems)

Viruses

Virus Classification: By Target

- The way viruses are usually categorized is by what they do.

1. Boot virus: Infects the boot sector of disk storage. Each time the disk is mounted, the boot sector is read and executed, causing the virus to be executed.
2. Program virus/File Infector: Infects the executable programs. It is executed each time the file is executed. The fact that the virus is executed only when the file is executed implies that it has fewer chances to execute than a boot sector virus.

- Macro virus: Exploits the macro language of a program like Microsoft Word or Excel.

Viruses

Boot Sector Virus

- Normal boot procedure
  - POST (Power On Self Test) > BIOS discovers bootable devices > BIOS reads the boot sector from such a device > BIOS passes control to it
  - Bootable hard disk contain a Master Boot Record (MBR)
    - 512-byte boot sector that is the first sector of a partitioned hard disk
    - Also contains the partition table
  - MBR code looks for a bootable partition and transfers control to it
- Boot sector viruses
  - Inserts themselves into the boot sector area
  - When the system boots, viruses do their damage, and in turn transfer control to the relocated MBR code

Viruses

Macro Virus

- Uses an application's own macro programming language
  - E.g., MS Office Visual Basic for Applications
  - A macro is an executable program embedded in a word processing document or other type of file
  - Users employ macros to automate repetitive tasks and thereby save keystrokes
- What is Particularly threatening about Macro viruses is that,
  - Do not infect programs but documents
  - Platform independent
  - Easily spread

Viruses

Virus Classification : By Hiding

- **Encrypted virus**
  - Virus creates a random encryption key, stored with the virus, and encrypts the remainder of the virus
  - When an infected program is invoked, the virus uses the stored random key to decrypt the virus
  - When the virus replicates, a different random key is selected

Viruses

Encrypted virus Example

- Before infection

1	Insert document in fax machine. (Program entry-point).
2	Dial the phone number.
3	Hit the SEND button on the fax.
4	Wait for completion. If a problem occurs, go back to step 1.
5	End task.

- After infection

1	Skip to setp 6. (Virus modified entry-point.)
2	Dial the phone number.
3	Hit the SEND button on the fax.
4	Wait for completion. If a problem occurs, go back to step 1.
5	End task.
6	VIRUS instructions
7	VIRUS instructions
8	Insert document in fax machine. (Stored by the virus.)

Viruses

Virus Classification : By Hiding

- **Polymorphic virus**
  - Mutates with every infection, making detection by the **signature** of the virus impossible
  - Have specially designed **mutation engine** (**decryption also mutates**)
- **Metamorphic virus**
  - Mutates with every infection, rewriting itself completely at each iteration changing behavior and/or appearance, increasing the difficulty of detection

Viruses

Virus Classification : By Hiding

- **Stealth virus**
  - Stealth refers to a technique used by a virus to evade detection
  - A form of virus explicitly designed to hide itself from detection by antivirus software
  - The entire virus, not just a payload is hidden
  - Example: A virus can place intercept logic in disk I/O routines so when there is an attempt to read infected portions of the disk using these routines, the virus presents back an unifected program

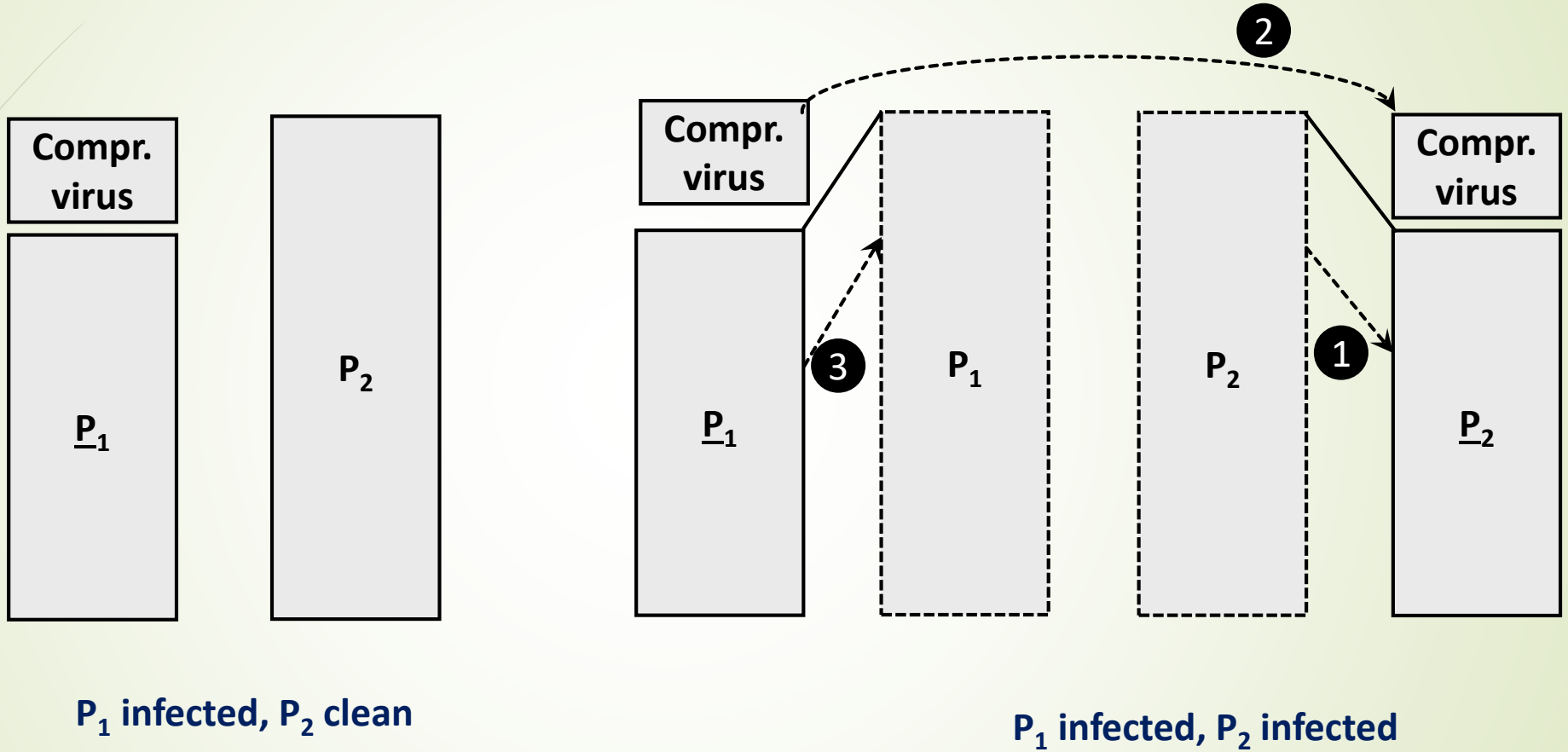
Viruses

Virus Classification : By Hiding

- **Stealth virus**
  - Example: A compression virus
- The virus in its original form is easily detected
  - Infected version of program is longer than the uninfected one
- To avoid detection compress the executable file
  - Make sure that infected and uninfected are of identical length

Viruses

Compression



Viruses

Types of viruses

- The way viruses are usually categorized is by what they do.

3. Multipartite virus: Is a combination of the boot and program virus/ infects both boot sectors and executable files. Hence it can propagate either
   a. On removable volumes that are manually taken from computer to computer
   b. Inside executable files that are transferred between computers on any type of network.
4. Parasitic virus: Embeds itself into another file or program such that the original file is still viable

Viruses

Countermeasures to Viruses

- Best countermeasure is **prevention**
  - Do not allow a virus to get into the system in the first place
  - But, in general, impossible to achive
- Hence, need to do one or more of
  - **Detection:** Determine that infection occured and locate virus
  - **Identification:** Once detected, identify the specific virus
  - **Removal:** Once identified, remove all traces of the virus
- If detected but can't identify or remove, one must discard and replace infected program
- Virus-antivirus coevolution
  - Everlasting battle

Worms

Worms

- **Self-replicating program** that propagates over Internet
  1. **Using email** – A worm mails a copy of itself to other systems
  2. **Remote execution capability** – A worm executes a copy of itself on a remote system, either using explicit remote execution facility or by exploiting flaw (e.g., buffer overflow) in some net service
  3. **Remote login** – A worm logs onto a remote system as a user then uses commands to copy itself to the remote system

Worms

Internet Worms Uses/Applications

1. Launch a DoS

- Such a worm could launch vast DoS attacks that are out of the reach of current protection technologies.
- Such powerful attacks can bring down not only E-commerce sites, but sensitive military sites or the root domain name servers of the Internet.
- Such an attack may be an ideal tool in the hands of terrorists or may be perpetrated intentionally by a rogue nation to serve as a prelude to a large-scale war.

Worms

Internet Worms Uses/Applications

2. Access to Sensitive Information

- It is well known that rogue software often searches for sensitive information such as passwords and credit card numbers, but a wide-spread worm may blindly search for any kind of information based on a set of keywords.
- This type of a “needle in a haystack” search is inefficient, but with millions of worms searching simultaneously, it may produce quick results.

Worms

Internet Worms Uses/Applications

3. Spread Misinformation

- A well-known adage says *"you can't fool all the people all the time,"* but when the same false message arrives from millions of computers it may fool all the people some of the time.
- A wide-spread worm may cause much confusion and disrupt the lives of many by sending misinformation from millions

Worms

Internet Worms Uses/Applications

4. Unknown reasons

- Most generally is the need for being recognized and famous (never has it been that it was an accident)

Worms

Worm Operation

- Has phases like a virus
  1. **Dormant** phase: Worm is idle, waiting for trigger event (e.g., date, time, program)
  2. **Propagation** phase: Worm searches for other systems, connects to it, copies self to it and runs (the copy may not be identical – it morphs to avoid detection)
  3. **Triggering** phase: Worm activated by some trigger event to perform intended function
  4. **Execution** phase: The intended function is performed e.g. DoS attack on a specified target

Worms

Worm Propagation

- To propagate, a worm generally performes the following functions
  - **Search** for other systems to infect by examining different repositories of remote system addresses
    - IP address-space probing to detect vulnerable targets
    - Note that this active aquisition/seach phase is not present in viruses
  - **Establish a connection** with a remote system
  - **Copy itself** to the remote system and cause the copy to be run

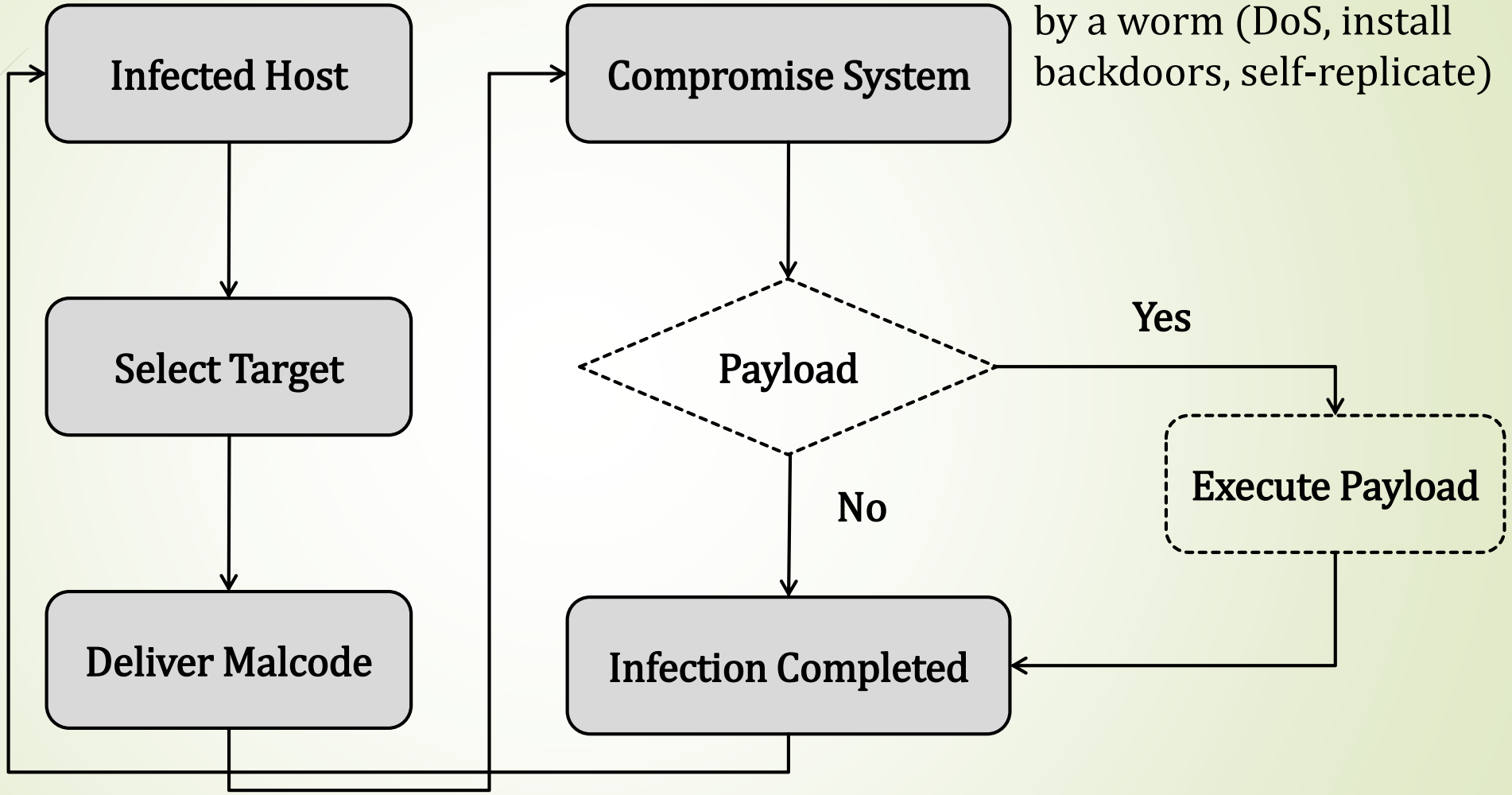
Worms

General Worm Propagation Model

- In the first stage the infected host searches for vulnerable targets
- When the target is found, the infected host tries to deliver malcode to the selected target
- Executing the malcode, the target host would be compromised
- Once the system is compromised, some malware can perform additional tasks :- payload
  - Payload refers to those additional tasks by a worm (DoS, install backdoors, self-replicate)

Worms

General Worm Propagation Model



Payload: Additional tasks by a worm (DoS, install backdoors, self-replicate)

Worms

General Worm Propagation Model: Actions at each stage

1. The target selecting stage
   - Random IP address probing
   - Harvesting email addresses (e.g., from the address book)
   - Through file sharing systems
2. The malcode delivery stage (can send only a part in this stage)
   - A payload associated with buffer overflows
   - Using mail of messaging services
   - Specially crafted HTML pages hosted on a web server

Worms

General Worm Propagation Model: Actions at each stage

3. Compromising the system
   - Execute malcode: email vulnerabilites, user intervention, automatic execution

   E.g., buffer overflow, backdoors, etc.

Worms

Worm Propagation Modeling

- **Simple Epidemic Model**
  - Uses the time model of Infectious diseases to model Worm propagation
  - Three possible states – Susceptible, Infected, Quarantined/Removed
- "Infectious" hosts: Gontinuously infect others
- "Removed" hosts in epidemic area
  - Recover and immune to the virus
  - Dead because of the disease
- "Removed" hosts in computer area:
  - Patched computers that are clean and immune to the worm
  - Computers that are shut down or cut off from worm's circulation

Worms

Worm Propagation Modeling

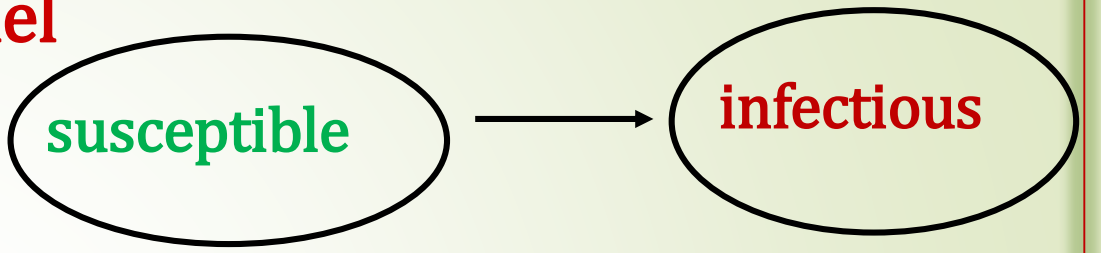
- **Simple Epidemic Model**

- Assumptions
  - The population size (#hosts) is large
  - Any host has equal probability to contact any other hosts in system
  - Number of contacts is proportional to #infectious X #susceptible

Worms

Worm Propagation Modeling

Classical Simple Epidemic Model



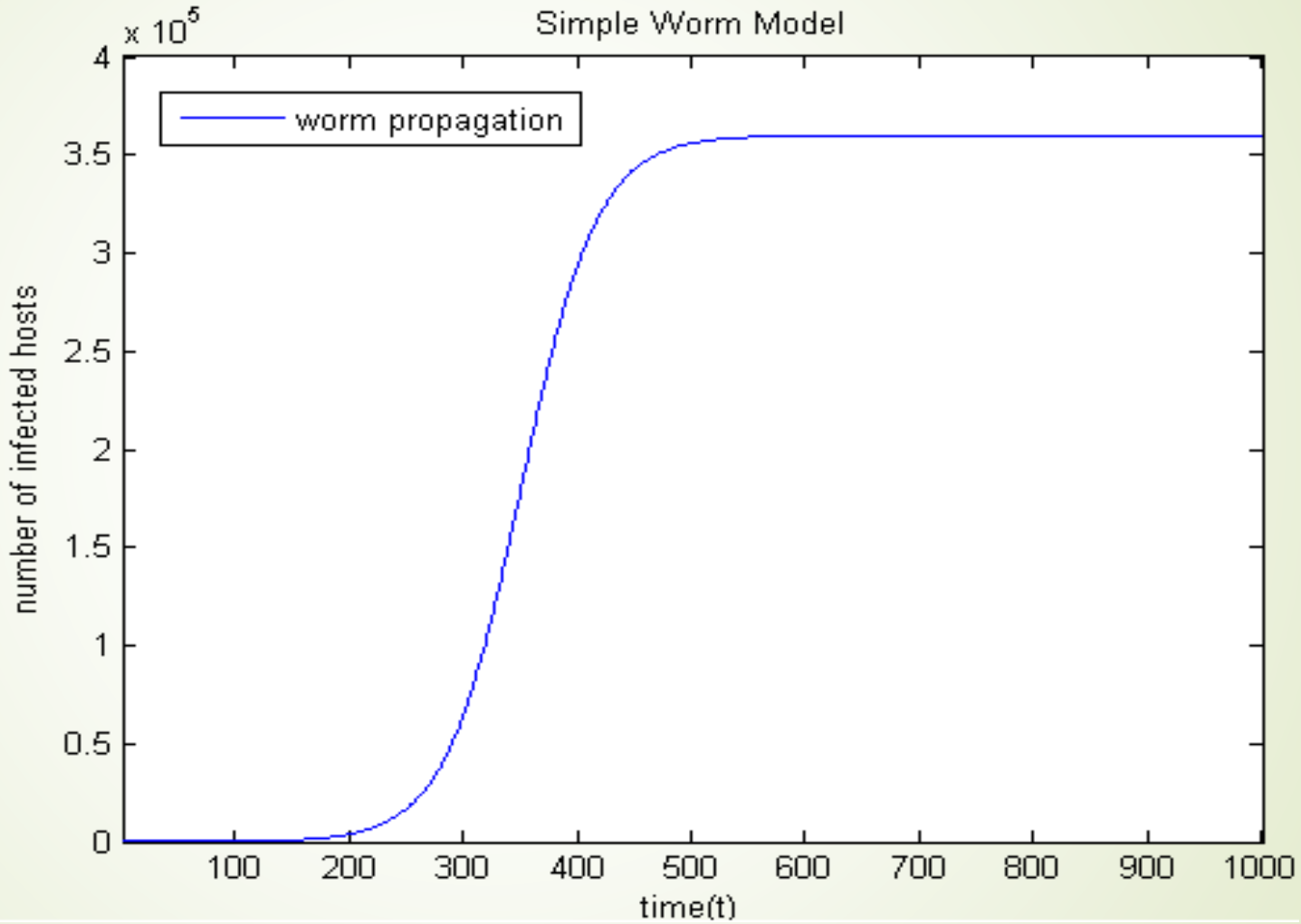
- State transition
  - N - population of hosts
  - $S(t)$ - susceptible hosts; $I(t)$ - infectious hosts at time t

$$\frac{dI(t)}{dt} = \beta I(t)S(t)$$

Where $S(t) = N - I(t)$: N is the size of population and β is the infectious rate

Worms

Classical Simple Epidemic Model



Worms

Classical Simple Epidemic Model (SIR)

- State transition
  - N - Population of hosts
  - $S(t)$ - Susceptible hosts
  - $I(t)$ - Infectious hosts
  - $R(t)$ - Removed from infectious at rate γ

Classical Simple Epidemic Model (SIR)

- State transition

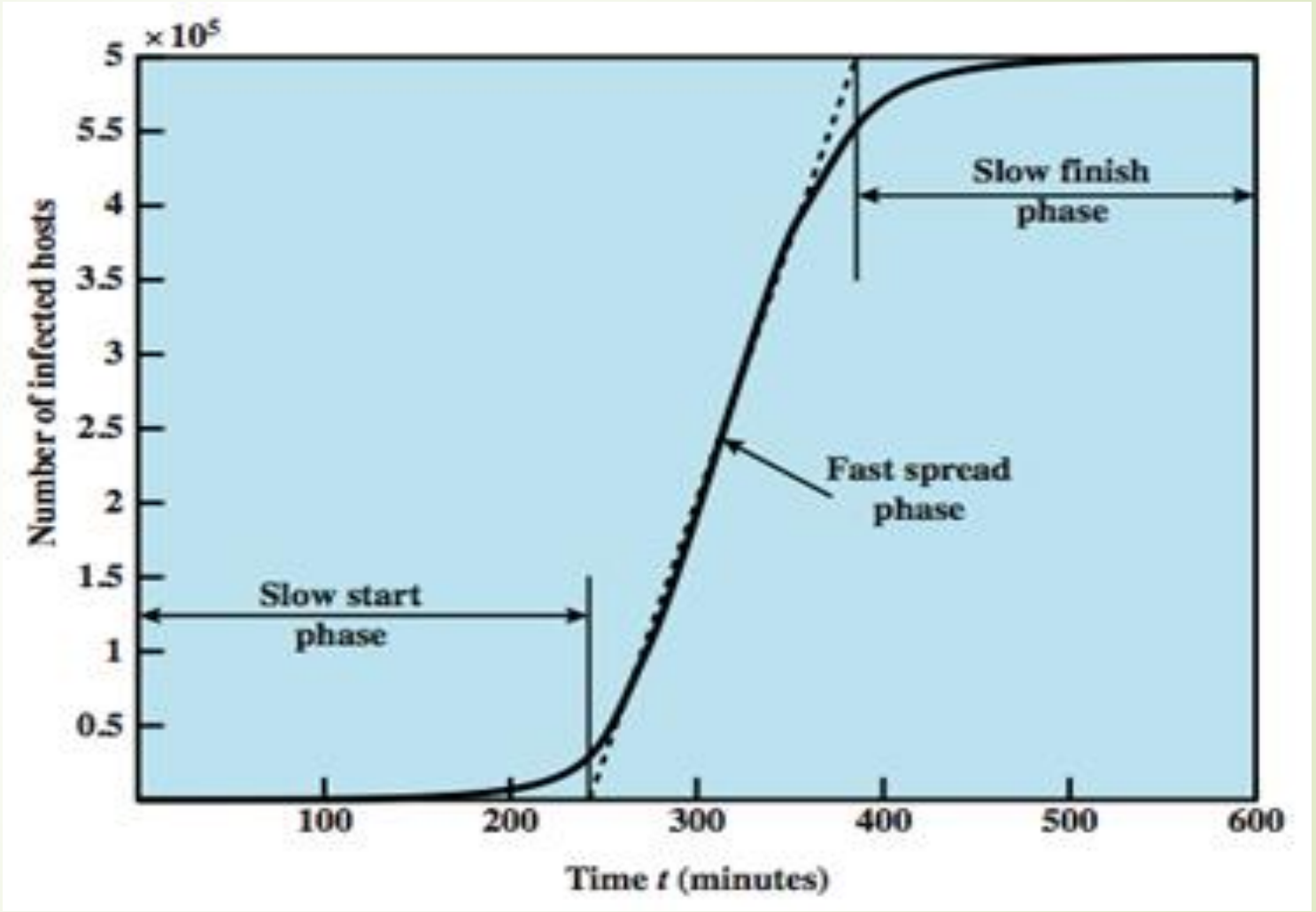
$$\frac{dI(t)}{dt} = \beta I(t)S(t) - \frac{dR(t)}{dt}$$

$$\frac{dR(t)}{dt} = \gamma I(t)$$

Worms

Characteristics of Worm spreading

Worm growth: slow start, fast spread phase, slow decay



Worms

Worm Threat Mitigation (PSC Model)

$$\frac{dI(t)}{dt} = -\beta I(t)S(t)$$

1. Reduce # of susceptible hosts (prevention)

$$\frac{dI(t)}{dt} = \beta I(t)S(t) - \gamma I(t)$$

2. Reduce rate of infection (suppression)

$$\frac{dR(t)}{dt} = \gamma I(t)$$

3. Reduce # of infected hosts (containment)

Worms

Worm Threat Mitigation (PSC Model)

1. Prevention

- This aims to reduce the size of the vulnerable population
- Secure programming, applying software updates, AV protection
- Patching
  - Generally, patches take days to release – only now that relatively reliable distribution networks for patches are springing up

2. Containment and suppression (the easiest)

- Firewalls, Content Filtering, disconnecting infected machines

Viruses Vs Worms

Virus

- Spreads from program to program, or from disk to disk
- Uses each infected program or disk to make copies of itself
- Computer sabotage
- Destroys data or erases disks
- Operating system specific

Worm

- Uses computer hosts to reproduce themselves
- Travel independently over computer networks
- Software sabotage
- Resides in memory rather on disk
- Puts computers at a standstill

Intruder Detection and Prevention

Introduction: Prevention and Defense

- Rogue software are a menace because worms can appear out of nowhere and infect all the computers of an organization within minutes. Once deeply embedded, it starts sending tentacles outside, looking for more computers to infect, and may also look inside for sensitive information to send back to its creator.
- Thus, discovering this type of software early, preventing its appearance in the first place, and defending against it are important goals of any computer user, whether an individual or part of an organization.

Prevention and Defense

Understanding Vulnerabilities

- The first step in the fight against computer viruses is an understanding of vulnerabilities that viruses exploit in order to propagate and inflict damage.
- The following is a list of such weaknesses:

Prevention and Defense

Understanding Vulnerabilities

1. User apathy. Even though every computer user is aware of the threat of viruses, people always feel that "it's not going to happen to me." Computer users share software without checking for infection, they ignore suspicious behavior that may indicate the presence of a virus, and they don't spend the time to learn and apply basic security measures.
2. Insufficient security control. Many computers, especially personal computers, are not equipped with hardware and software features that help in detecting and isolating viruses and other security threats. Large, multiuser computers generally perform much better in this area.

Prevention and Defense

Understanding Vulnerabilities

4. Misuse of available security features. Anti-virus software should always have the latest virus update. Running such software with old virus updates is an ineffective use of an effective security feature. Other examples are misuse of permissions and passwords. Permissions (to use computing resources) are an effective tool that can prevent accidental damage to the file system. An administrative user who allows free access to anyone is misusing this tool. Passwords are also a powerful instrument, but they should be chosen carefully. Users often choose an easy-to-remember password, but such passwords tend to be easy to guess.

Prevention and Defense

Understanding Vulnerabilities

5. **Weaknesses in the operating system.** Modern operating systems are extremely complex and are implemented and maintained by large teams of programmers. Vulnerabilities and weak points are discovered all the time in this type of software. Quite often a discovery is made by a security expert or a clever user who then notifies the manufacturer of the operating system. Sometimes, a weakness is discovered by a virus writer who immediately sets up to exploit it.

Prevention and Defense

Understanding Vulnerabilities

6. Unauthorized use. There are those who regard breaking into a computer as a challenge, and the more secure and secret the computer, the greater the challenge. Once a hacker manages to break into a computer, the temptation to create havoc is great, often too great.
7. Anonymity of networks. Before the era of computer networks, a malicious person had to actually walk into a computer center in order to do damage. Nowadays, with the prevalence of networks, attackers have the advantage of anonymity.

Prevention and Defense

Understanding Vulnerabilities

- These points illustrate the need for a comprehensive security program that includes

1. Identifying vulnerabilities to viruses
2. Correcting them and plugging up security holes
3. Monitoring the results

Prevention and Defense

Understanding Vulnerabilities

In the workplace, management should provide resources for virus prevention, resources that should include at least the following points:

1. Training seminars. From time to time, an employee should be sent to a seminar where the basics of security are covered and security policies and procedures are described and rehearsed. Experience shows that training is important. A group of well-trained users who are aware of security threats and are willing to cooperate is ultimately the best weapon an organization can have in the war on viruses. User education is expensive for a company, but pays for itself in the long run.

Prevention and Defense

Understanding Vulnerabilities

In the workplace, management should provide resources for virus prevention, resources that should include at least the following points:

Training should be mandatory, should be done periodically, and should include the following topics:

i. A background on viruses, how they are planted, how they propagate, the types of damage they inflict, and how to detect their presence. Users have to be aware of the risk of bringing private software into their work computers and of sharing software.

Prevention and Defense

Understanding Vulnerabilities

Training should be mandatory, should be done periodically, and should include the following topics:

ii. Software vulnerabilities exploited by viruses in the past. This may help a user to detect current weaknesses.

iii. Company security policies and contingency procedures.

Prevention and Defense

Understanding Vulnerabilities

2. Any decisions pertaining to the acquisition of new software and hardware should involve security experts.
3. Monitoring user and network activity. Special software can monitor the activities of the various user computers and report any suspicious activity to a central monitoring facility.

Prevention and Defense

Understanding Vulnerabilities

- Examples of abnormal activities are
  1. An increase in CPU activity during lunch break or at night,
  2. An unusually large number of email messages coming in or going out.
- Network activity is especially easy to monitor automatically and it is an important tool in fighting viruses. The activity should be monitored all the time and the number of packets per activity should be monitored all the time and the number of packets per second coming in and going out should be saved. At any time, this number should be compared with the corresponding numbers in the past few days. Any large deviation may signal the presence of a virus.

Prevention and Defense

Understanding Vulnerabilities

4. Emergency policies must exist and users should be trained in them, so they know what to do and who to turn to in emergency situations where a virus or other type of attack is discovered.
5. Limited sharing. An organization should try to limit the sharing of computing resources among its members, and the sharing of data between itself and the outside world. In practice, there should be only one gateway between an organization's local-area network and the Internet, and this gateway should be protected by security experts.

Prevention and Defense

Understanding Vulnerabilities

6. Self isolation in an attack.

- When an organization senses an attack, it often tries to isolate its network from the outside world.
- A familiar term is “to pull the plug.” This is definitely good practice.
- If an attack is fast or is not subtle, pulling the plug immediately is going to limit the damage.

Prevention and Defense

Understanding Vulnerabilities

7. Audit. The originators of the Christmas card virus and the Internet worm were identified as a result of audit.
   - Details about the movement of data packets on the Internet were saved and analyzed, leading searchers to certain networks and geographical areas.
   - Generally, audit isn't useful in tracking down attackers because operating systems do not save enough information that can later be used to trace the progress of a virus and thus serve as an audit trail.
   - Even if such tools become part of common operating systems in the future, they would not prevent viruses, only help to locate their authors.

Prevention and Defense

Understanding Vulnerabilities

8. Backups. It is important to have complete and recent backups, regardless of viruses or any other security concerns.
   - Hard disk prices have dropped to such levels that most computer owners, individuals as well as organizations, can afford to have a backup drive for each drive used in the computer.
   - Backup programs are fast and perform incremental backup; they copy only files that have been modified since the last backup.

Prevention and Defense

Defense Against Malware

- Virus defense should involve

1. Technical means
2. Common sense in using your computer
3. Legal means.

- Applying all three approaches can keep a user out of trouble for a long time, although perhaps not forever.

Prevention and Defense

Defense Against Malware

- In principle, it is possible to have perfect virus protection simply by isolating the computer from any communications with the outside world.
- One of the best practical defenses is to simply use common sense and be careful in using the computer. Just as you wouldn't consider purchasing foods or medicines from untrusted sources, don't purchase or accept software from untrusted sources.

Prevention and Defense

Defense Against Malware

- Another good (although not perfect) protection is to limit transitivity.
  - When computer A in a network gets infected, it infects another computer B, which in turn infects C, and so on. Limited transitivity is a mechanism (implemented by a policy, an algorithm, an operating system, or a piece of hardware) that guarantees that anything sent from A to B will not be sent from B.

Prevention and Defense

Defense Against Malware

- Computer B is free to send anything to other computers, except what it has received from A.
- Such a mechanism would severely limit the spread of a virus and would also discourage virus writers.
- Attempts to implement such mechanisms have always proved too restrictive and were therefore impractical.

Anti-Virus Software

Prevention and Defense

Anti-Virus Software

- Computer programs intended to identify and eliminate computer viruses.
- Antivirus applications act as a guard over your system, scanning incoming files and applications, “quarantining” or cleaning up unwanted viruses looking to cause harm to your system
- Antivirus software is considered to be an aid that detects, fixes and even prevents viruses and worms from spreading to your computer as well as connecting computers.

Prevention and Defense

Evolution of Anti-Virus Software

- Virus and antivirus technologies have both evolved
- Early viruses were simple code, easily removed
- As they become more complex, so must the countermeasures
- AV Generations
  1. First: **Signature scanners:** What a virus is?
  2. Second: **Heuristics:** What the virus does? – from its structure
  3. Third: **Identify actions:**What the virus actually does?
  4. Fourth: **Combination packages**

Prevention and Defense

Signature-Based AV Software

- Requires a **virus signature** to identify a virus
- Virus signature
  - Early viruses had esentially the same bit pattern in all copies
  - Good signature is one that is found in every object infected by the virus, but is unlikely to be found if the virus is not present
  - Not too short (false positives), not too long (false negatives)

Malware detected? \ Object is malicious?	Yes	No
Yes	OK	False positive
No	False negative	OK

Prevention and Defense

Signature-Based AV Software

- Extracting good signature difficult and time-consuming
  - Involves disassembling and debugging the infection to identify key portions of the virus
  - Once it is extracted it has to be tested against a large library of uninfected programs to reduce the likelihood of false positives
- Detects viruses for which AV has a signature in its DB
  - Can also detect slightly modified versions of a virus
- **Signatures added to the anti-virus DB to detect earlier viruses are powerless to detect new virus strains**
  - Polymorphic viruses

Prevention and Defense

Heuristic AV Software

- Detects infections by **scrutinizing a program's overall structure**, its computer instructions and other data contained in the file
  - What a virus does? – from its structure
- Can detect unknown infections
  - Searches for generally suspicious logic rather than looking for specific signatures
- Typically work in two phases of operation
  - Catalog what behaviors the program is capable of exhibiting
  - Analysis of the observerd and cataloged behavior and assesment as to whether the behavior looks virus-like

Prevention and Defense

Anti-Virus Software

- There are different types of antivirus software for different computers

1. Some are designed for personal computers
2. Some are for servers and others for enterprises

- A distinction is made between three types of anti-virus measures,

1. Virus-specific detection methods
2. Generic techniques
3. Preventive techniques

Prevention and Defense

Anti-Virus Software

1. Virus-specific detection methods: Look for and identify specific viruses. Most anti-virus software operates this way.
   - The anti-virus program scans files in the disk, looking for bit strings that signal the presence of known viruses.
   - When a virus is located, the program gives the user a choice of deleting the virus automatically, placing the infected file in quarantine for detailed inspection later, or ignoring it.
   - The third option makes sense for viruses whose deletion is complex and should be done manually, by the user, rather than automatically by the anti-virus software. Generally, it is easy to disinfect boot sectors and macro viruses, but much harder to repair infected executable files.

Prevention and Defense

Anti-Virus Software

2. Generic virus detection techniques: Don't look for specific viruses but instead examine the computer (files on the disk and programs in memory) for anything suspicious, unusual, or anomalous.
   - An example of such activity is an attempt to modify the size of an executable file by a user program.
   - A generic technique cannot identify the presence of a specific virus, but can warn the user that something suspicious has taken place (or is about to take place) in a certain file or in a certain memory-resident program.

Prevention and Defense

Anti-Virus Software

3. A virus preventive technique: Creates an environment in the computer where viruses hesitate before they enter, or cannot thrive (i.e., execute) once they have entered.
   - Also called the "heuristic method": The software, for example, "could try to emulate the beginning of the code of each new executable that the system invokes before transferring control to that executable." if the program attempts to use "self-modifying code" or appears to be a virus, it's assumed that the virus has infected the executable.
   - In this method there are a lot of false positives.

Prevention and Defense

Anti-Virus Software

4. Sandbox method: An antivirus program takes suspicious code and runs it in a “virtual machine” to see the purpose of the code and exactly how the code works. After the program has terminated, the software analyzes the sandbox for any changes, which could indicate a virus.

Prevention and Defense

Anti-Virus Software: Why is Software an issue?

1. Some antivirus software can considerably reduce performance.
2. There should not be more than one antivirus software installed on a single computer at any given time.
3. It's sometimes necessary to temporarily disable virus protection when installing major updates.
4. Some argue that antivirus software often delivers more "pain than value to end users.

User Authentication [Passwords]

User Authentication: Introduction

- Protective measures (e.g., access control, accountability) make sense only if we can identify and authenticate users
- Authentication validates user identity
  - Often as prerequisite to allowing access to the system resources
- Authentication process consists of two steps
  - Identification step
    - Presenting an identifier to the system (e.g., userID, username)
  - Verification (authentication) step
    - Presenting or generating authentication information that binds the entity presenting the identifier and the identifier itself
- Distinct from message authentication

User Authentication: Introduction

Authentication Means

- There are four general means of authenticating a user's identity

1. Something the user knows: Password, personal identification number (PIN)
2. Something the user possesses: Smart cards, physical keys, tokens
3. Something the user is (static biometrics): Recognition by fingerprint, face, retina, iris
4. Something the user does (dynamic biometrics) : Recognition by voice pattern, hadwriting style, typing rhythm

- Can be used in combination
- All have advantages and issues

User Authentication: Introduction

Password Authentication

- Widely used user authentication method
  - User provides name/login (username) and password
  - System compares password with that saved for specified login
- Authenticates ID of user logging and provides security by
  - Determining that the user ID is authorized to access system
  - Determines the user's privileges (e.g., admin or not)
  - Is used in discretionary access control (e.g., a user owning a file may enable another entity to access this file)

User Authentication: Introduction

Password Vulnerabilities

1. **Offline dictionary attack :** Attacker obtains system password file (with password hashes) and compares password hashes against hashes of passwords from the dictionary
2. **Specific account attack :** Submit candidate passwords until the correct password discovered or until the account is locked (e.g., after 3 failed attempts)
3. **Popular password attack :** Try popular passwords against a range of user IDs
4. **Password guessing against single user :** Make educated guesses based on knowledge about the user (age, gender, marital status, ...)

User Authentication: Introduction

Password Vulnerabilities

5. **Workstation hijacking:** Steal unlocked workstation and use e.g. Cain & Abel to recover the password
6. **Exploiting user mistakes**: Passwords written down, shared, social engineering
7. **Exploiting multiple password use:** Password reuse problem (due to cognitive overload)
8. **Electronic monitoring:** Intercept passwords communicated across a network (simple and naive encryption does not help here)

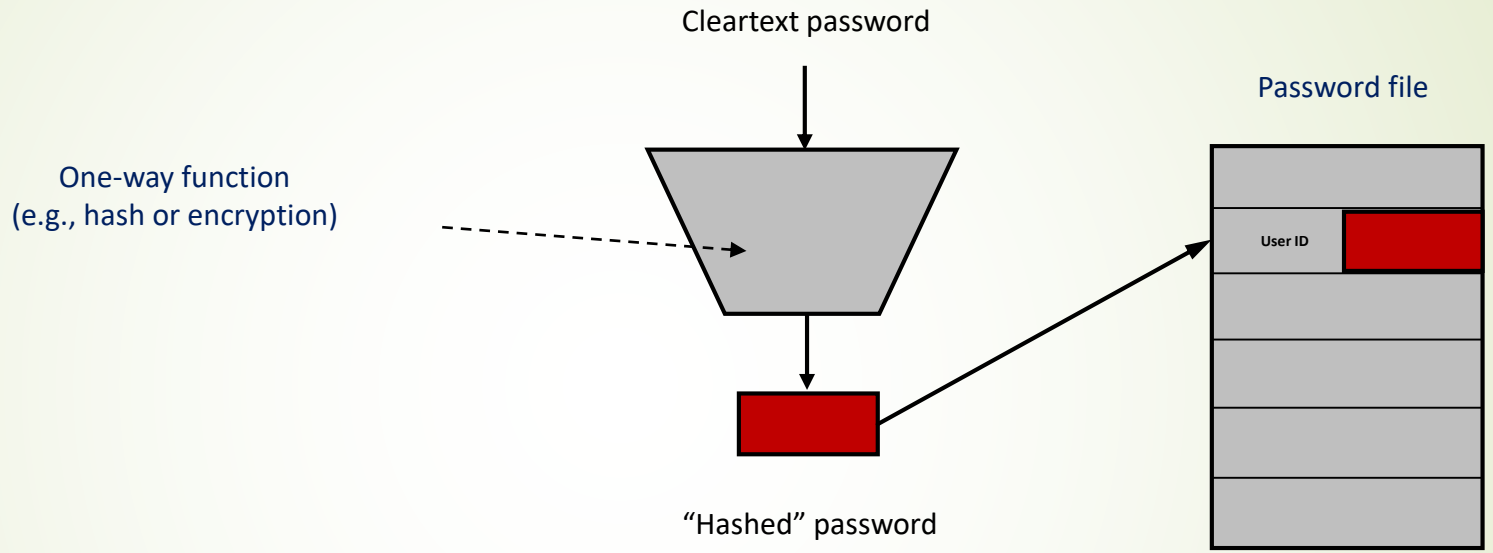
User Authentication

Storing Password

- Passwords are never stored in clear text: The risk of theft would be great
- Instead, a hash of a password is stored
  - Recall, hashing is a one-way function which gives a unique and unreversable result (hash value, message digest)
  - If a user provides a correct password, its hash must be identical to the hash stored (previously) in the password file

User Authentication

Storing Password

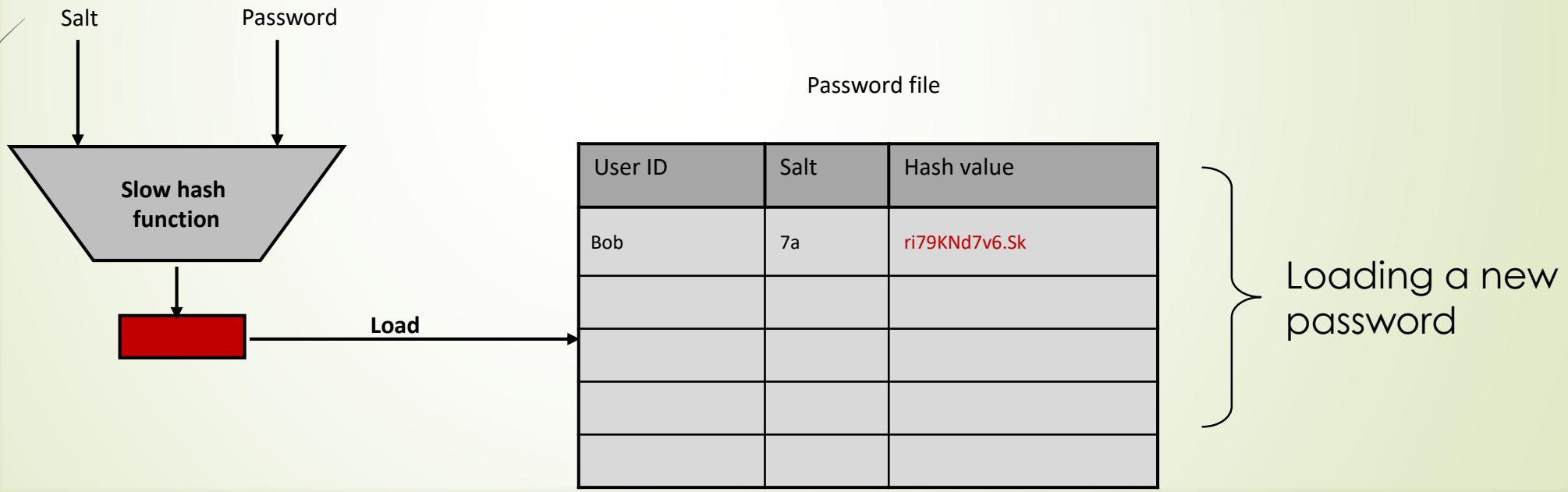


- Password-based authentication in Unix and Windows

Password-based authentication in Unix and Windows

Unix Passwords

- To load (create) a new password into the system
  - The user selects or is assigned a password
  - This password is combined with a fixed-length salt value



Unix Password Scheme: Salt Values

- Offline dictionary attack
  - Assume: the goal is to guess a single pwd & salt not used
  - Attacker obtains a copy of the password file
  - Attacker hashes likely candidate passwords and compare obtained hash values with the ones in the password file
  - If any of the guesses matches one of the hashes in the file, the attacker has found a password that is in the file

Unix Password Scheme: Salt Values

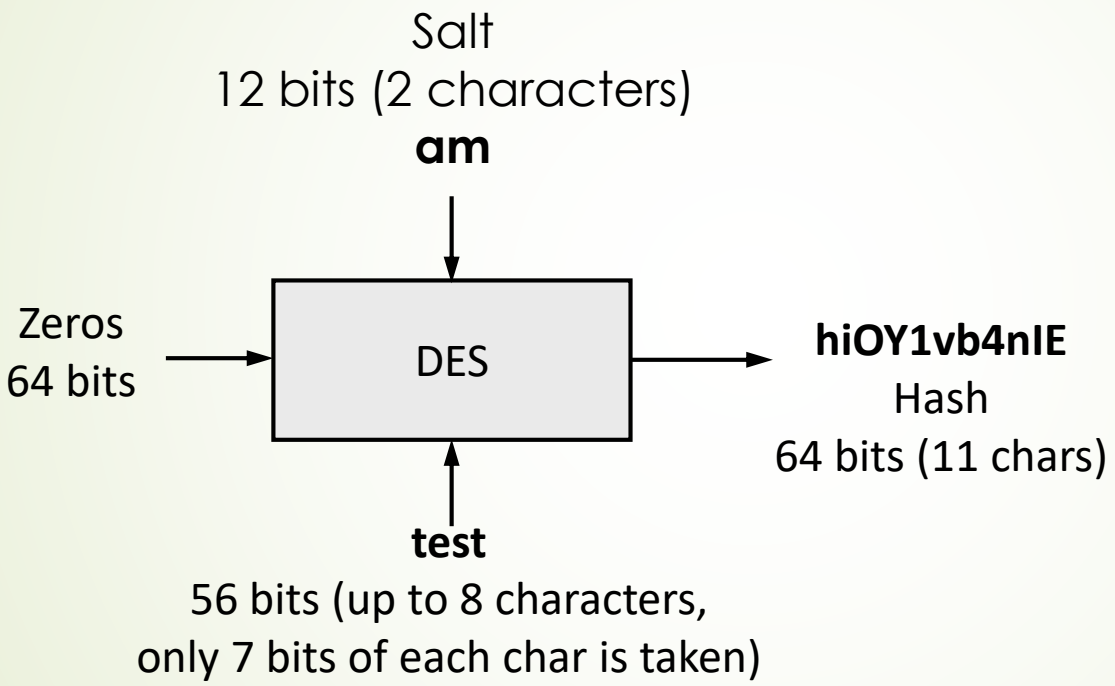
- The salt value serves three purposes
  - Prevents duplicate passwords to be visible in password file
  - Increases difficulty of offline dictionary attacks (k bits salt increases guessing load by a factor of $\sim 2^k$)
  - Not possible to find out whether a user with passwords on two or more systems has used the same passwords on all of them

Unix Hashed Pwd Implementation

- The original scheme (crypt(3) routine)
  - 8 character password form 56-bit secret key
  - 12-bit salt perturbs DES encryption algorithm in one of 4096 different ways
  - 0 value repeatedly encrypted 25 times (slows down guesses)
  - Output translated to 11 character sequence

Unix Hashed Pwd Implementation

- The original scheme (crypt(3) routine)



pwd	salt	crypt(3) hash
test	am	hiOY1vb4nIE
test	ri	j.uEL2QOTHU
test	7a	FB/N4.DacNU

Unix Pwd Hashes

- crypt(3)-based implementation is inadequate today
  - 8 chars (i.e., 56 bits) are are simply too few
  - Dictionary attack investigated using the Blue Horizon supercomputer
    - Precomputed and stored 207 billion hashes (~1.5 TB) for over 50 milion passwords in about 80 min ($207 \times 10^9 / 50 \times 10^6$ approx. 4096 – #salt values)
- Time-memory tradeoffs
  - Effective when salt is not used (Oechslin'03 showed that using 1.4GB of data – rainbow tables - Windows hashes broken in <14 sec)

Unix Pwd Hashes

- Better hashes for Unix
  - Modern Unix systems based on MD5 hashes instead of DES hashes
  - Advantages:
    - Passwords can have more than 8 characters
    - Produces 128 bit hash values
    - Longer salt values (48 bits)

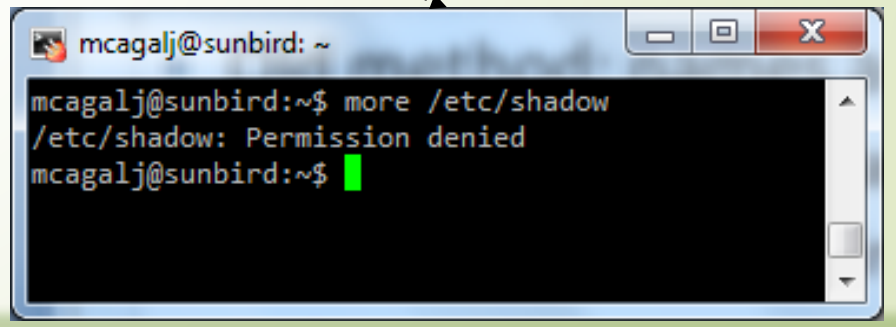
Password-based authentication in Unix and Windows

Unix Pwd Hashes

- Old method: names and hashes are stored in /etc/passwd
  - Free for anybody to read
  - Opens up for easy offline dictionary attack
- Safer method: the hashes stored in separate file /etc/shadow
  - Only root can access to this file

```
root:x:0:0:root:/root:/bin/bash
mcagalj:x:1001:1001:,,,:/home/mcagalj:/bin/bash
```

```
root:aQtsvOTXjNRbY:10919
mcagalj:HYy0b0xFEWIZw:10919:
```



Unix Pwd Hashes

- Theft of Unix Hashes
  - Goal: gain access to /etc/shadow
  - Boot the machine on a CD
  - Obtain root privileges (e.g., by using an exploit)

END