



Chap. 15) System Security

경희대학교 컴퓨터공학과

이 승 룡

■ There is no perfectly secure system!

- ✓ Protection can only increase the effort needed to do something bad. It cannot prevent it
- ✓ Every system has holes, it just depends on what they look like
- ✓ Even assuming a technically perfect system, there are always the four Bs:
 - Burglary: steal it
 - Bribery: find whoever has access to what you want and bribe them
 - Blackmail: or photograph them in a compromising position
 - Bludgeoning: or just beat them until they tell you

■ Security service

- ✓ CIA
 - Confidentiality
 - Integrity
 - Availability
- ✓ Authentication, Authorization, Access control, Nonrepudiation

Cracker's Basic Steps

- Gather information
 - ✓ as much information about your site as possible
- Use port scanner
 - ✓ to gather information about what services are running on hosts
 - ✓ Search for weak security services
- Get a login account
 - ✓ Doesn't matter whose account
- Get root privilege
 - ✓ Bugs in programs or badly configured systems
- Keep root privilege
 - ✓ Leave some sort of backdoor for future access



■ Hardware security

- ✓ Restrict access to equipments
 - Smart card (ID card)
 - Bio-metric access control

■ BIOS security

- ✓ Set a boot password
- ✓ Prevent booting from CD-ROM or floppy drives

■ Session security

- ✓ Some shells (e.g. tcsh) provide the automatic logout facility if there is no activity during the specified time period
- ✓ vlock (for locking a virtual terminal) / xlock
- ✓ Screen savers



■ Authentication

- ✓ Make sure we know who we are talking to
- ✓ Usually done with passwords
 - First line of defense and single biggest security hole
- ✓ Problems in passwords:
 - Users who write their password on paper for all to see
 - Type password slowly that others can see
 - Dumb passwords like “password”
 - Passwords should be long and obscure – unfortunately easily forgotten and usually written down
- ✓ Passwords should not be stored in a directly-readable form
 - Use some sort of one-way-transformation (a “secure hash”) and store that
- ✓ Cf) CHAP (*Challenge Handshake Authentication Protocol*)



■ Authentication alternatives

- ✓ Some alternatives
 - Physical keys: badges, smart cards, ...
 - Biometric keys: Fingerprints, iris prints, facial profiles, voice prints, hand geometry, signature analysis ...
 - Passwords using images
- ✓ Should not be forgeable or copiable
- ✓ Can be stolen, but the owner should know if it is
 - Need to invalidate old one



Account Security (Cont'd)

■ Authorization

- ✓ Determine if x is allowed to do y
 - Can be represented as an “access matrix”
- ✓ Access control lists (ACLs)
 - With each object, indicate which users are allowed to perform which operations
 - Simple and used in almost all file systems
- ✓ Capabilities
 - With each users, indicate which resources may be accessed and in what ways
 - Frequently do both naming and protection: Can only “see” an object if you have a capability for it
 - Used in systems that need to be very secure



■ Setuid/setgid programs

- ✓ Badly written setuid programs may contain a security hole
 - Know of all setuid and setgid programs on your system
 - Setuid programs that are not needed should be deleted
 - Never allow setuid/setgid files in user's home directories
 - Use nosuid option in fstab file for home file system and for NFS-mounted file system
 - Maintain a check on any new setuid programs:
`find / -type f -perm 2000 -o perm 4000 -o perm 6000`
 - Never write setuid/setgid shell programs



File System Security (Cont'd)

■ Search paths

- ✓ Many users include the current directory in their search path
- ✓ A cracker could place programs with the same name as standard commands everywhere they have write access in directory hierarchy
 - The fake program may have malicious code, or capture data from the user pretending to be the real application
- ✓ Place current directory last in the path
 - Alternatively use full path names (e.g. /bin/su)
- ✓ Current directory SHOULD NOT be in the search path for root user



File System Security (Cont'd)

■ Other countermeasures

- ✓ Carefully specify default permissions: `umask`
- ✓ Put a limitation on the file system usage: `quota`
- ✓ Check file system integrity regularly: `find`, `tripwire`, ...
 - Files without known owners may indicate unauthorized access: `find / -nouser -o -nogroup`
 - Files with “other” write permission (`o+w`) may indicate a problem: `find / -type f -perm 2`
- ✓ Use encrypted file system
 - CFS (Cryptographic File System)
 - TCFS (Transparent CFS), etc.
- ✓ Backup file system: `tar`, `dd`, ...
- ✓ Monitor system logs



■ Use secure protocols

- ✓ Don't let the plain password float around the network
- ✓ Secure shell (ssh) suite of programs encrypts the communications of many of protocols
 - ssh (telnet), slogin (rlogin), sftp (ftp)
- ✓ Use secure http (https) for secure connection
- ✓ Secure Socket Layer (SSL) provides data encryption of all data that passes between clients and server
- ✓ IPsec protocol: encrypt every IP packet
 - Required for IPv6, optional for IPv4



■ TCP wrappers

- ✓ Monitors/filters Internet services such as telnet, ftp, finger, etc.
- ✓ Similar to Internet super daemon, inetd
- ✓ Before connecting the client to the service program, log the activity and check if it should be permitted
 - /etc/hosts.allow, /etc/hosts.deny
- ✓ You should be able to detect cracking intention or activity from the log



■ Firewalls

✓ Firewall

- Creates a filter or protective layer between an organization's internal networks and any external networks to which they are connected

✓ ipchain: packet filtering firewall

- Examine each packet header to decide the action
- e.g. block incoming ICMP echo requests:
`ipchains -A input -i eth0 -p icmp -s 0/0 -d 0/0 -l -j REJECT`

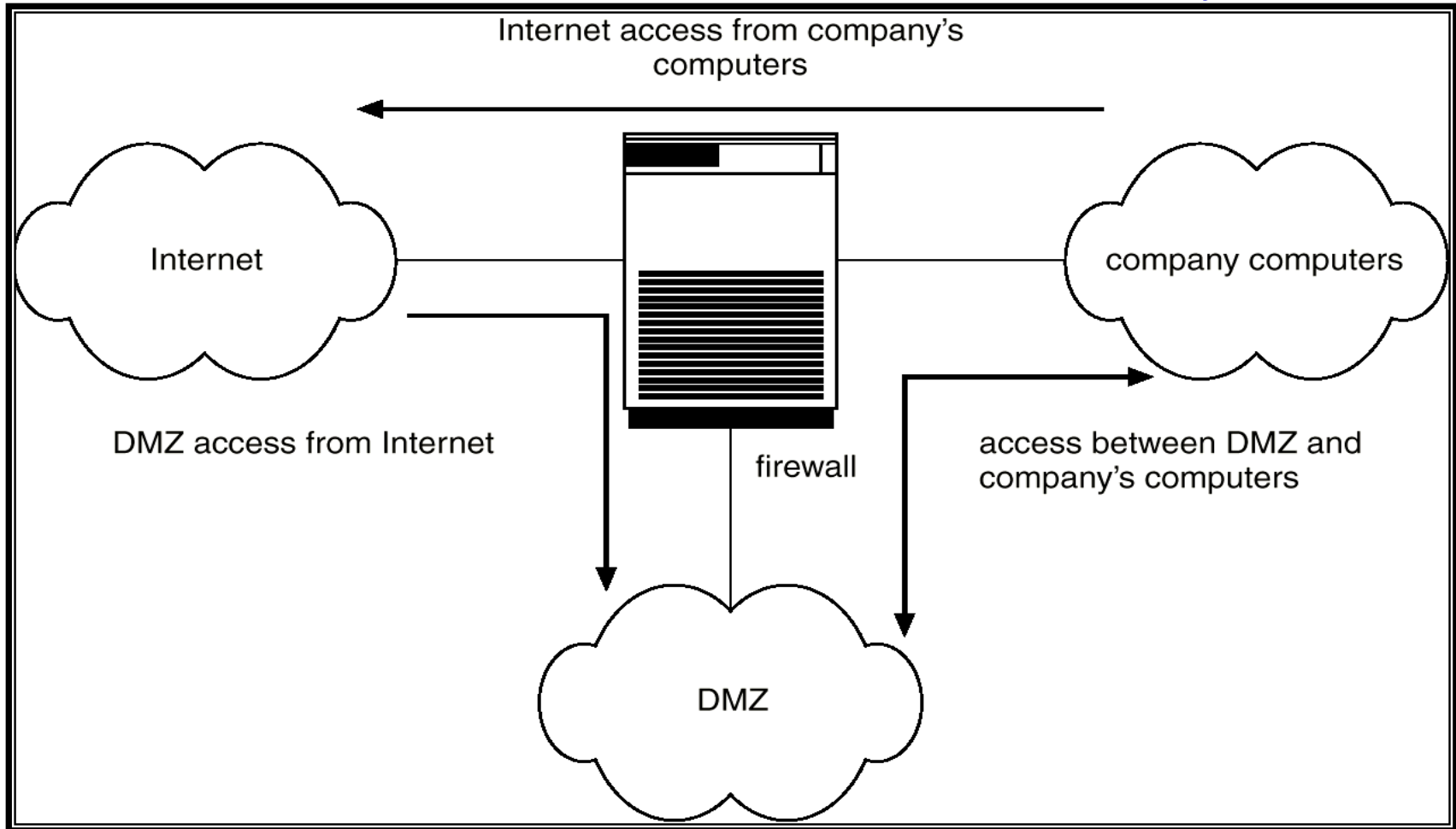
✓ Proxy firewall

- Standard: require client-side configuration. Client connects to a special port
- Transparent: similar to packet filter firewall, but controls traffic



Firewall

- A firewall is placed between trusted and untrusted hosts
- The firewall limits network access between these two security domains



■ Physical threats

- ✓ Acts of nature: floods, fire, earthquake, explosion, etc.
- ✓ Intruder takes computers, dig up network cable, or access system consoles

■ Logical threats

- ✓ Caused by problems with computer software
 - Misuse by people (e.g. easy-to-guess passwords)
 - Bugs in programs or in their interaction with each other

■ Operational threats

- ✓ No security policy, incomplete enforcement

■ Denial of service

- ✓ Prevent computer from providing services through
 - wasting resources of computer
 - flooding services on your system, thus preventing them from providing service to legitimate clients

■ Dictionary attacks

- ✓ crack, nutcrack, John the Ripper, etc.
- ✓ *crack* program found 10-20% of passwords could be guessed, using a password list containing variations on login names, user's first and last names and a list of 1800 common first names

■ Login spoofing

- ✓ Simulate login process
- ✓ Need to have the login sequence start with a key combination that user programs cannot catch
 - CTRL-ALT-DEL in Windows 2000.

■ Trojan horses

- ✓ A seemingly innocent program contains code to perform an unexpected and undesirable function
- ✓ To have the Trojan horse run, the person planting it first has to get the program carrying it executed
 - Attract attention and encourage people to download and execute it

■ Logic bomb

- ✓ A piece of code written by one of a company's programmers and secretly inserted into the OS
- ✓ OK as long as the programmer feeds it its daily password
- ✓ If the programmer is suddenly fired, the logic bomb explodes
 - clear the disk, erase files at random, encrypt essential files, etc.



■ Trap door

- ✓ Created by the code inserted into the system by a system programmer to bypass some normal check
 - What happens if the programmer leaves the company?
 - Some special key sequences lead you to the “debug” mode in your mobile phone
- ✓ Need to have code reviews as standard practice
- ✓ Difficult to do in open-source software



■ Stack and Buffer overflow

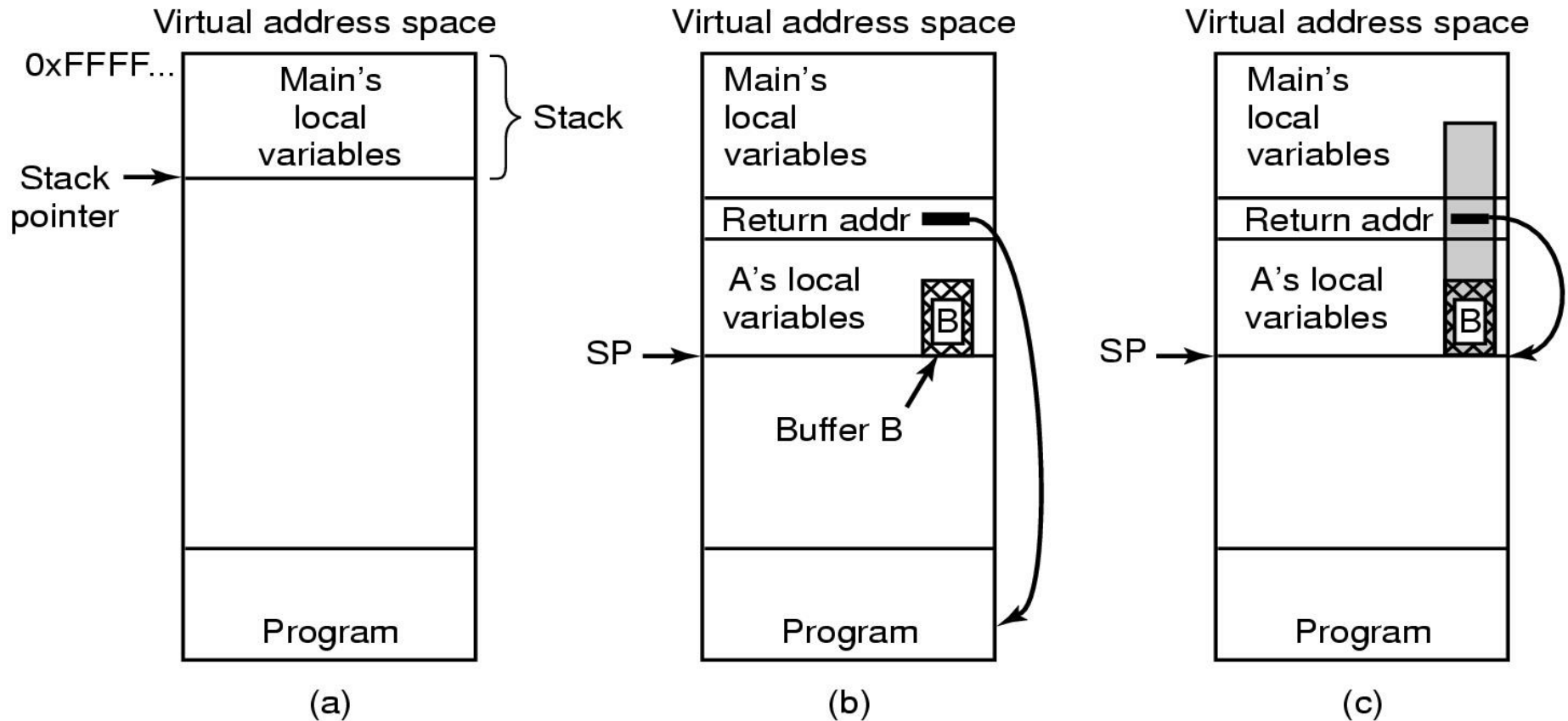
- ✓ What's the problem of the following codes?

```
void A(char *src)
{
    char B[1024];
    strcpy(B, src);
    printf(src); printf("\n");
}
int main(int argc, char *argv[])
{
    if (argc > 1) {
        A(argv[1]);
    }
    return 0;
}
```



■ Stack and Buffer overflow (Cont'd)

- ✓ Do array bounds checking!



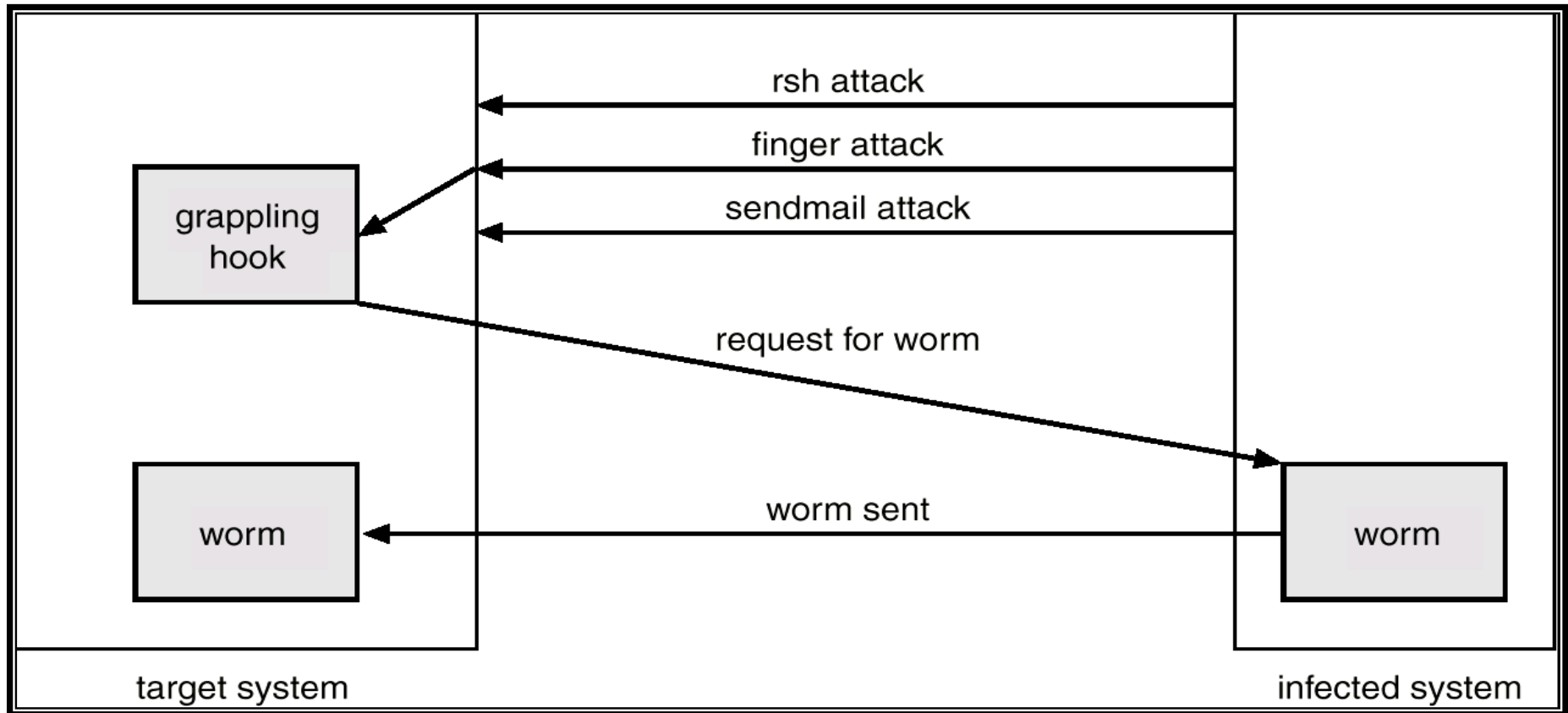
■ Virus and worms

- ✓ Virus is a program that can reproduce itself by attaching its code to another program
- ✓ Worms are like viruses but are also capable of spreading itself from machine to machine via network
- ✓ Types
 - memory resident viruses: e.g. intercept system call traps to infect other programs
 - boot sector viruses
 - device driver viruses: officially loaded at boot time
 - macro viruses: Microsoft Office

■ Malware



The Morris Internet Worm



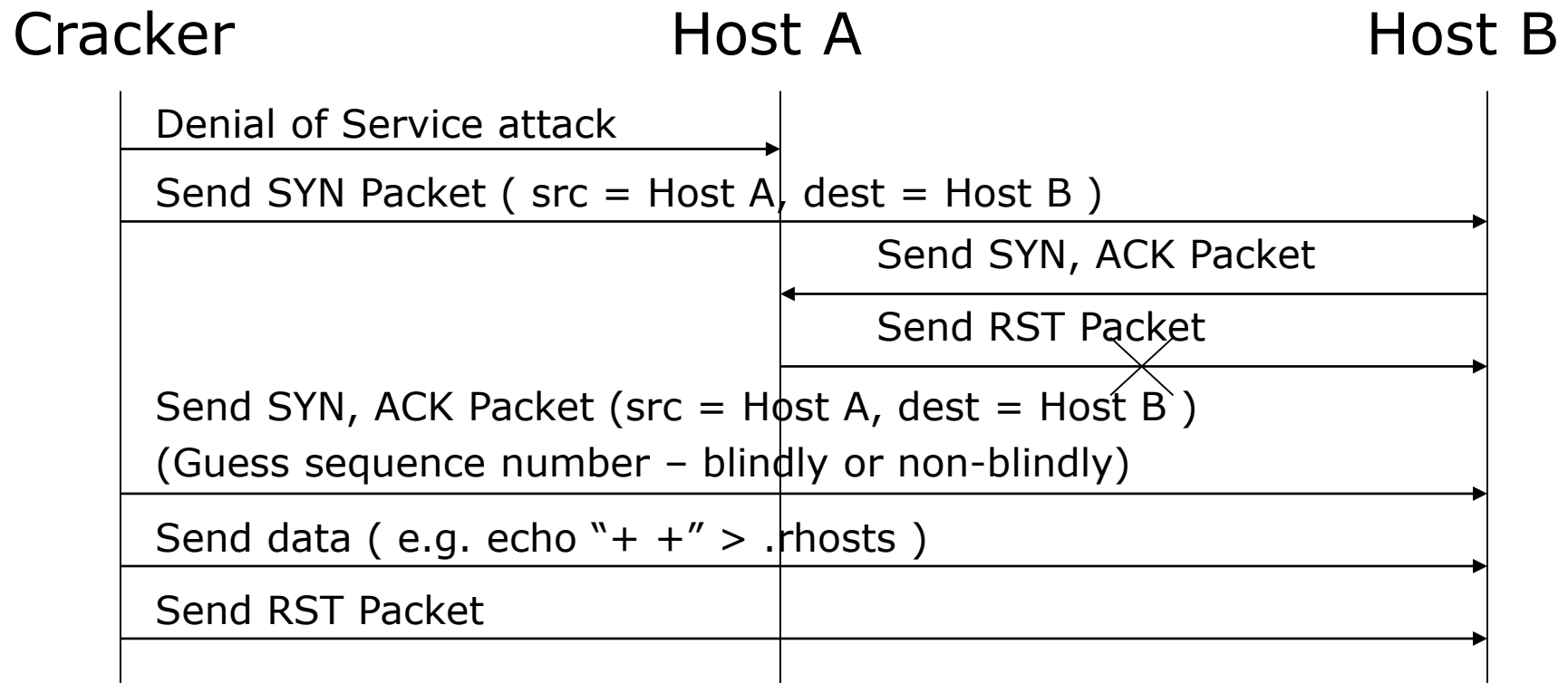
■ Packet sniffing

- ✓ Listens to ethernet traffic over LAN
- ✓ Ethernet adapter in promiscuous mode
 - Need root privilege
- ✓ Can see all data passing between hosts on the network
- ✓ Can gather usernames and passwords
 - telnet, ftp, httpd, pop3, imap, etc.
- ✓ tcpdump and sniffit are software sniffers



■ IP spoofing

- ✓ Steal an authorized IP and use it



■ Denial of service: internal attacks

- ✓ Use up all resources and make system crash
- ✓ Attacking resources: disk, memory, process, ...
- ✓ Examples
 - Shell script: `while (1) { mkdir foo; cd foo; }`
 - C: `while (1) { fork(); ((int *) malloc(100000))[40] = 1; }`
- ✓ Done by a local user, and in most cases by accident



■ Denial of service: external attacks

- ✓ Application level
 - Mail bombing
 - Buffer overflow
 - Java Applet attack
- ✓ Protocol level
 - TCP SYN flooding
 - Ping flooding
- ✓ Network level
 - UDP Storming

■ Distributed DOS (DDOS)

- ✓ Use multiple machines



Intrusion Detection

- Detect attempts to intrude into computer systems
- Detection methods:
 - ✓ Auditing and logging
 - ✓ Tripwire
 - UNIX software that checks if certain files and directories have been altered
 - I.e. password files
- System call monitoring



Basic Concept of Cryptography

■ What is cryptography

- ✓ The science of obfuscating data
- ✓ Can provide authentication, confidentiality, data integrity and etc.
- ✓ Cryptography algorithm is open, but key MUST be confidential

■ Two kinds of cryptography

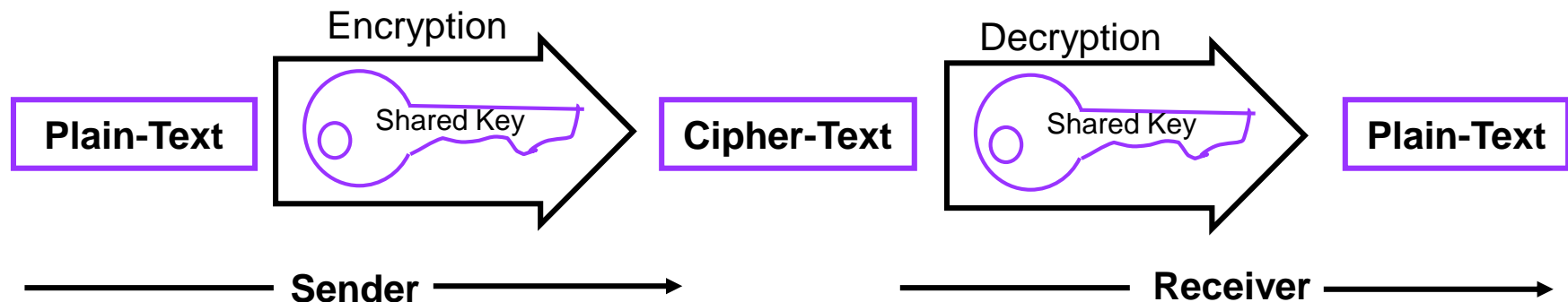
- ✓ Shared key cryptography
- ✓ Public key cryptography



Basic Concept of Cryptography (Cont'd)

■ Shared key cryptography

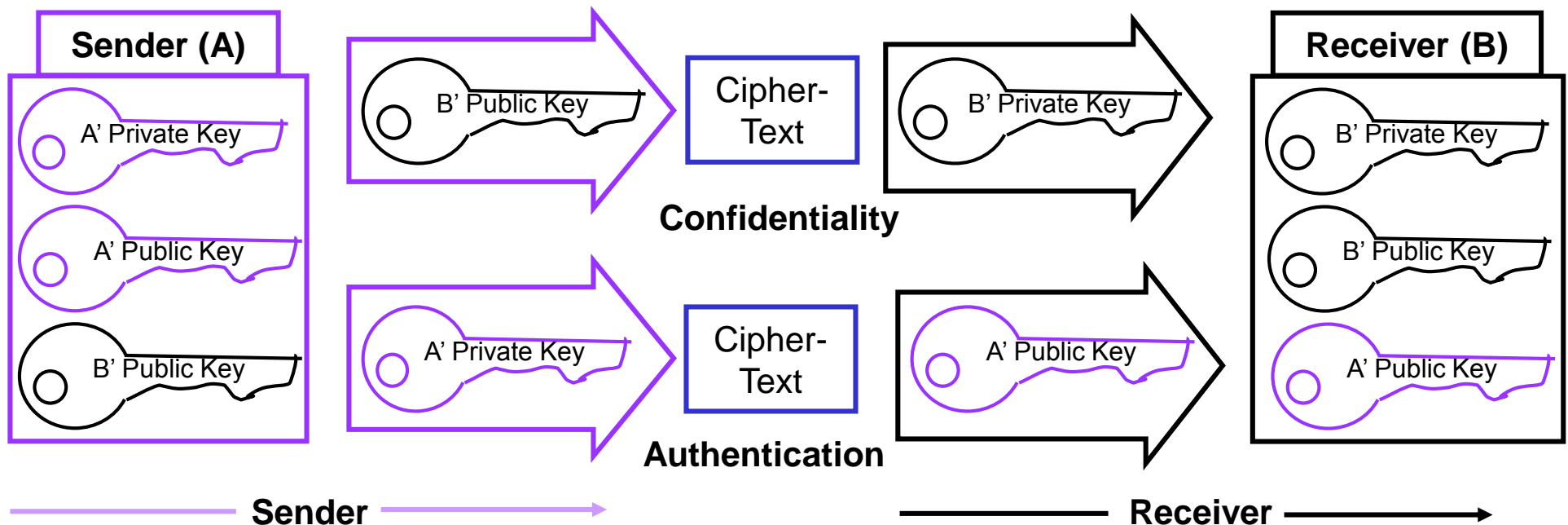
- ✓ Both of peers share the same key
- ✓ DES(Data Encryption Standard) (Cf. AES)
 - Bit operation (key size : 64bit, 128 bit)
 - Can provide authentication and confidentiality
 - **How can distribute the shared key secret and keep it secret ?**



Basic Concept of Cryptography (Cont'd)

■ Public key cryptography

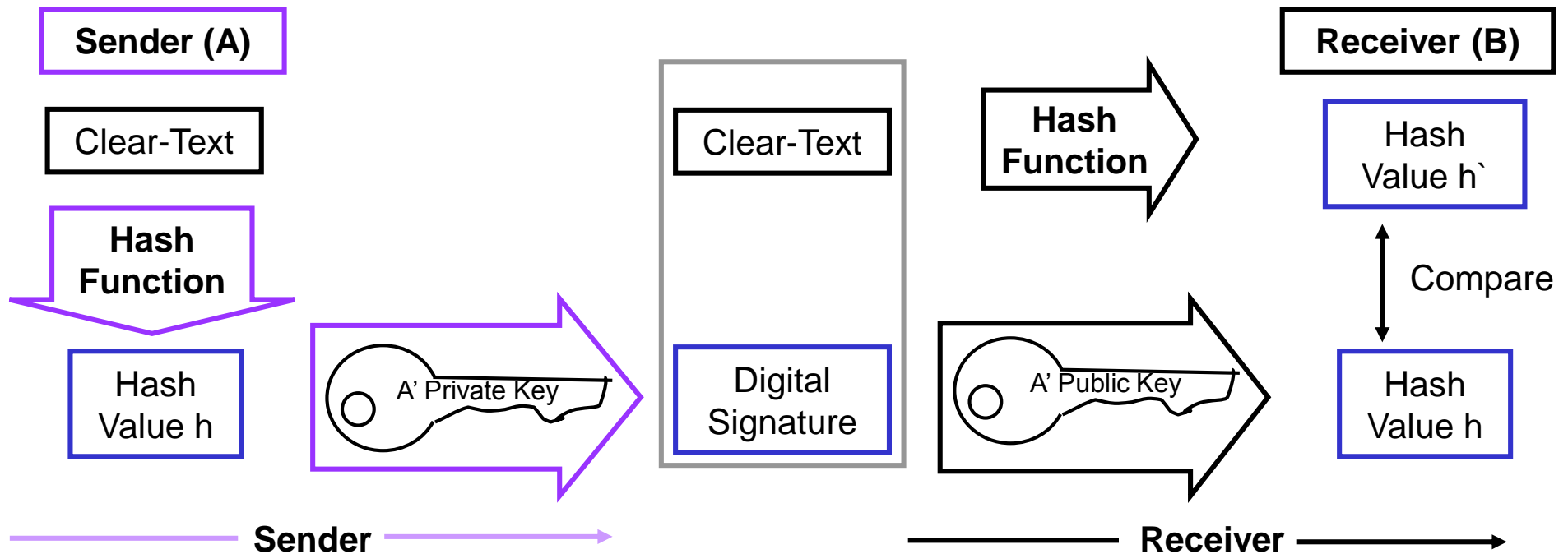
- ✓ Both of peers have its own private key and public keys
- ✓ Key pair
 - well known 'Public Key' and secret 'Private Key'
- ✓ Can provide confidentiality and authentication
- ✓ RSA : well known algorithm (Cf. ECC)



Applied Cryptography for Network

■ Digital Signature

- ✓ minimize encryption processing
- ✓ for authentication & integrity (not confidentiality)



Applied Cryptography for Network (Cont'd)

■ Validity of key

- ✓ DES < several hours
- ✓ RSA < several days
- ✓ **How can keep the connection secure?**

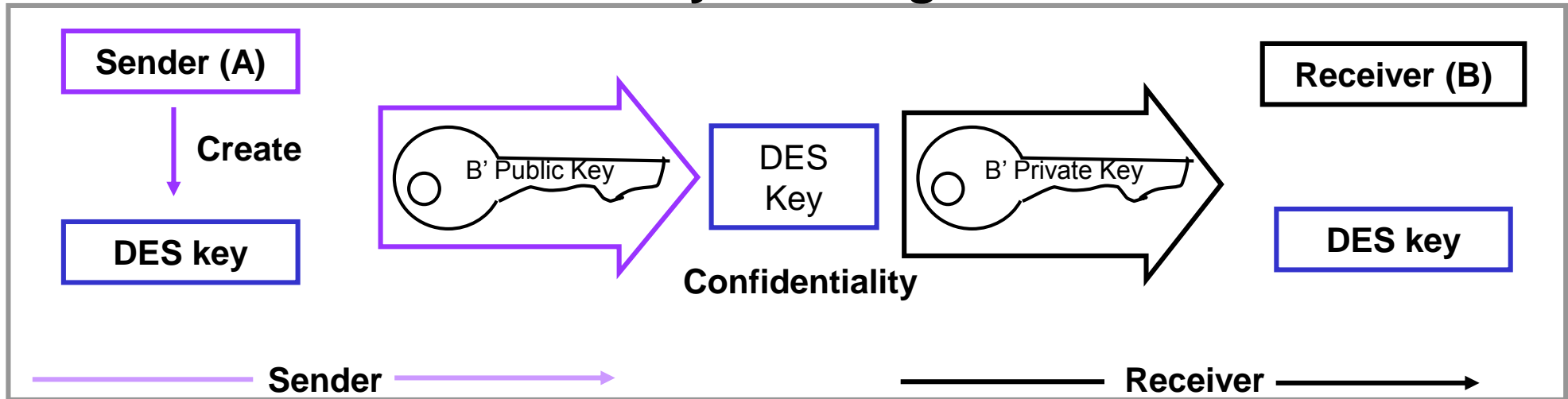
■ Security

- ✓ System security
 - Create DES key every session
 - Transfer DES key by RSA security
- ✓ Connection (session) security
 - One time key created by DES
- ✓ applied for SSL, TLS, IPsec and etc.

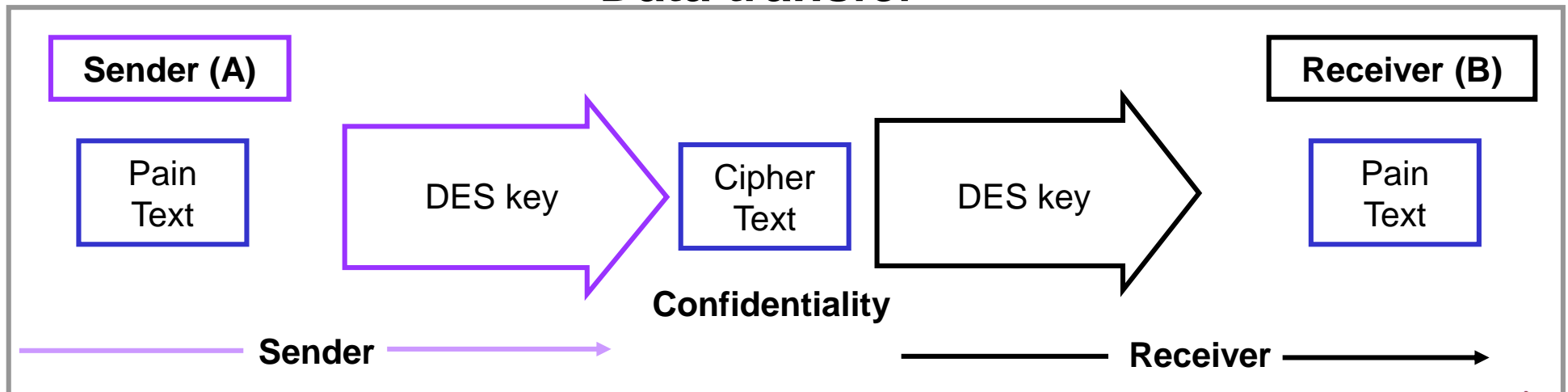


Applied Cryptography for Network (Cont'd)

Key exchange



Data transfer



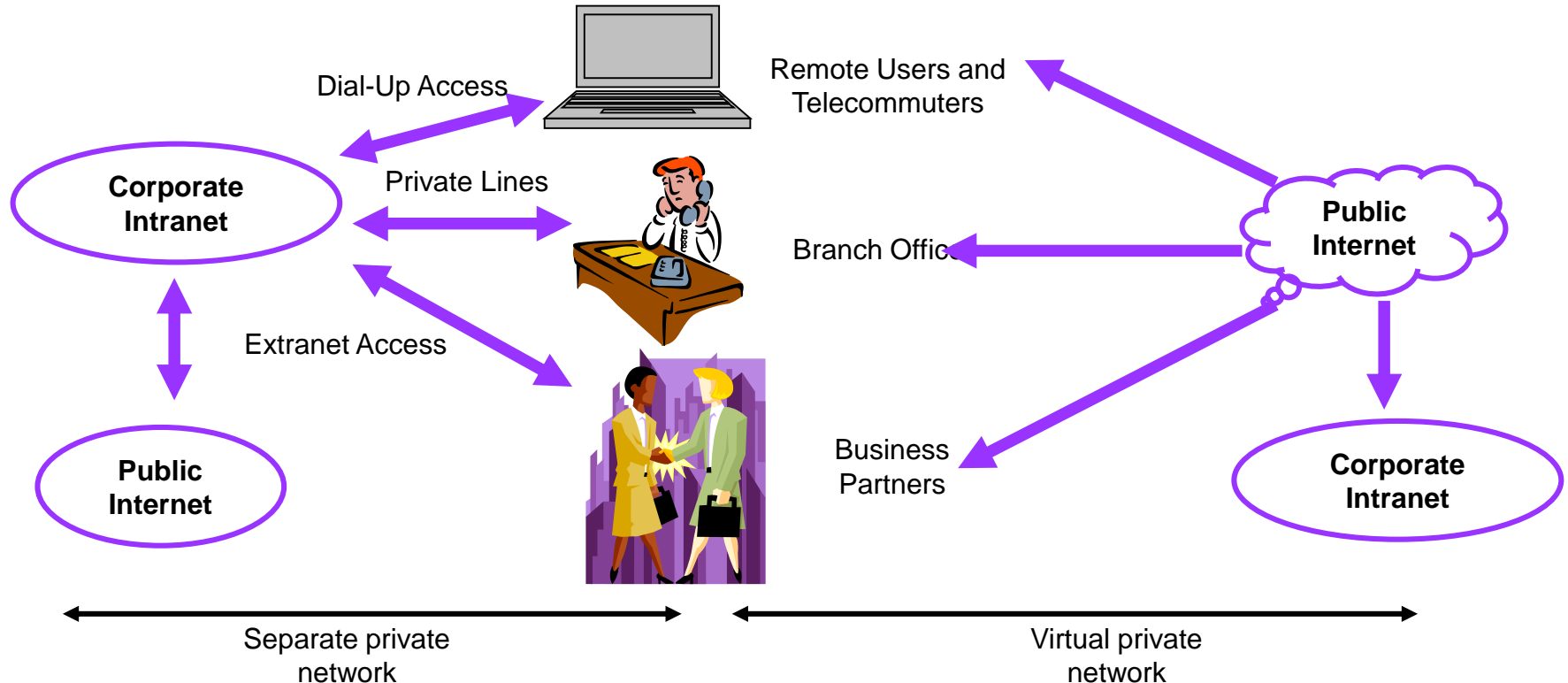
Virtual Private Network (VPN)

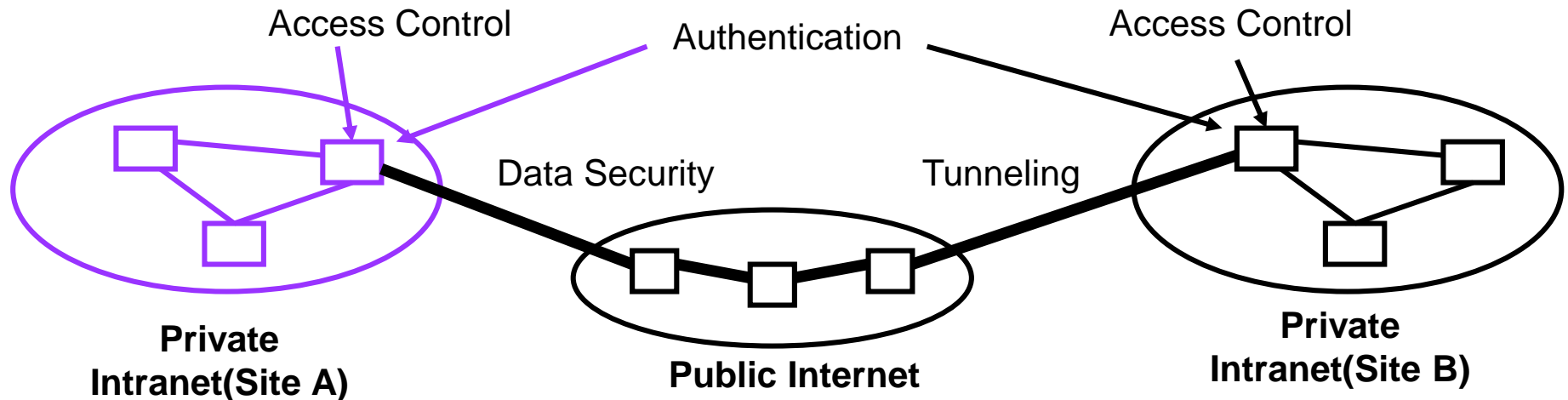
■ Virtual

- ✓ **No** physical infrastructure **dedicated** to the private network

■ Private

- ✓ Keep data confidential so that it can be received by an intended receiver





■ Tunneling

✓ PPTP, L2TP, L2F, MPLS , IPsec and etc.

■ Authentication

✓ Radius, CHAP, PKI and etc.

■ Access Control

✓ PKI and etc.

■ Data Security

✓ IPsec, PKI, SSL, TSL and etc.

■ Research topic: Mobile & Embedded System Security

- ✓ Security for Mobile Systems
 - Android/iOS/Tizen
 - Malware analysis

- ✓ Security for Embedded Systems
 - IoT device security
 - Hardware-assisted security
 - Embedded Linux security

■ Security services

- ✓ Integrity
- ✓ Authentication
- ✓ Authorization
- ✓ Access control
- ✓ Confidentiality (cryptography)

■ Attacks

- ✓ Dictionary attacks, login spoofing, Trojan horses, logic bomb, trap door, stack and buffer overflow, virus and worms, packet sniffing, IP spoofing, DoS (Denial of Service), DDoS (Distributed DoS)

■ Network security

- ✓ Secure protocols
 - IP layer: IPsec
 - Transport layer: SSL (Secured Sockets Layer), TLS (Transport Layer Security)
 - Application layer: ssh, sftp, https
- ✓ TCP wrappers
- ✓ Firewall
- ✓ IDS (Intrusion Detection System)

■ Cryptography

- ✓ Shared key cryptography
 - DES (Data Encryption Standard) algorithm
- ✓ Public key cryptography
 - Well-known Public key vs. secret Private key
 - RSA algorithm
- ✓ Digital signature

