System and Network Security

Giuseppe Anastasi
g.anastasi@iet.unipi.it
Pervasive Computing & Networking Lab. (PerLab)
Dept. of Information Engineering, University of Pisa

Based on original slides by
- Silberschatz, Galvin and Gagne
- Kurose and Ross

Objectives

- Discuss security threats and attacks
- Explain the fundamentals of encryption
- Examine the uses of cryptography in computing
  - Secrecy
  - Authentication
  - Message Integrity, Digital Signature
- Describe the various countermeasures to security attacks

Overview

- Threats and attacks
- Cryptography as a Security Tool
  - Secrecy
  - Authentication
  - Message integrity
  - Digital signature
  - …
- Security Defenses
  - User Authentication
  - Antivirus
  - Firewalls
  - …

Security vs. Protection

- Protection mechanisms protect system resources from the *internal* environment
- Security considers the *external* environment of the system
- Security defenses are aimed at protecting system resources from external threats and attacks

Security Threats and Attacks

- 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

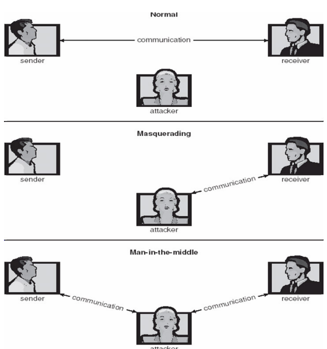
Security Violations

- Categories
  - Breach of confidentiality
  - Breach of integrity
  - Breach of availability
  - Theft of service
  - Denial of service

Security Violations

- Methods
  - Masquerading (breach authentication)
  - Replay attack
    - Message modification
  - Man-in-the-middle attack
  - Session hijacking

Standard Security Attacks



Security Measure Levels

- Security must occur at four levels to be effective:
  - **Physical**
  - **Human**
    - Avoid social engineering, phishing, dumpster diving
  - **Operating System**
  - **Network**

Security is as weak as the weakest link in the chain

Program Threats

- Trojan Horse
  - Code segment that misuses its environment
  - Exploits mechanisms for allowing programs written by users to be executed by other users
  - Variants:
    - Login spoofing, spyware, pop-up browser windows, covert channels
- Trap Door
  - Specific user identifier or password that circumvents normal security procedures
  - Could be included in a compiler
- Logic Bomb
  - Program that initiates a security incident under certain conditions
- Stack and Buffer Overflow
  - Exploits a bug in a program (overflow either the stack or memory buffers)

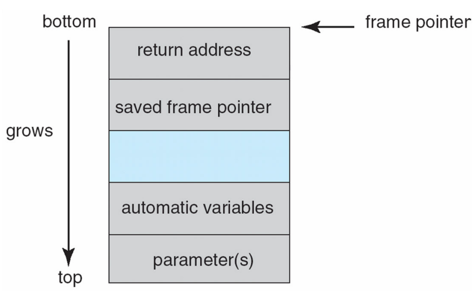
C Program *with* Buffer-overflow Condition

```
#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;
    }
}
```


C Program *without* 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 {
          strncpy(buffer, argv[1], sizeof(buffer)-1);
          return 0;
    }
}
```

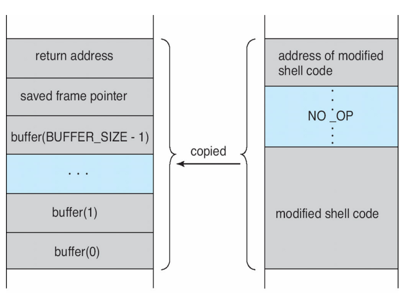
Layout of Typical Stack Frame



Modified Shell Code

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


Hypothetical Stack Frame



(a) Before attack

(b) After attack

How to avoid the Buffer-Overflow Attack?

- CPU doesn't allow code execution in stack segments
  - Sun Spark, used by Solaris
- NX bit in page table (AMD, Intel)
  - The corresponding page cannot be executed
  - Used by Linux, Windows XP

Program Threats (Cont.)

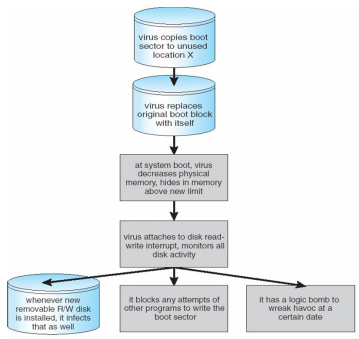
- Viruses
  - Code fragment embedded in legitimate program
  - Very specific to CPU architecture, operating system, applications
  - Usually borne via email or as a macro
    - Visual Basic Macro to reformat hard drive

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


Program Threats (Cont.)

- Virus dropper (typically a Trojan Horse) inserts virus onto the system
- Many categories of viruses, literally thousands of viruses
  - File
  - Boot
  - Macro
  - Source code
  - Polymorphic
  - Encrypted
  - Stealth (clandestino)
  - Tunneling (sotterraneo)
  - Multipartite (composito
  - Armored (corazzato)

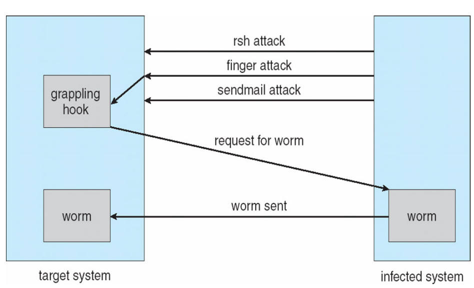
A Boot-sector Computer Virus



System and Network Threats

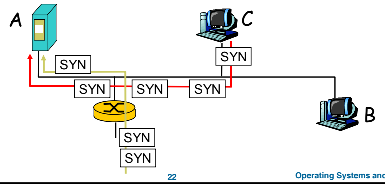
- Worms
  - use spawn mechanism; standalone program
- Morris Internet worm (Nov 1988)
  - Exploited UNIX networking features (remote access) and bugs in *finger* and *sendmail* programs
  - Grappling hook program uploaded main worm program
- Port scanning
  - Automated attempt to connect to a range of ports on one or a range of IP addresses

The Morris Internet Worm



System and Network Threats

- 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
  - SYN Flooding



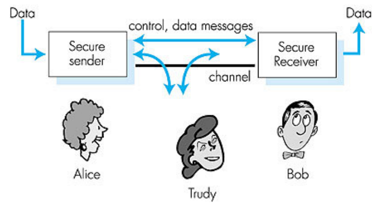
Overview

- Threats and attacks
- Cryptography as a Security Tool
  - Secrecy
  - Authentication
  - Message integrity
  - Digital signature
  - …
- Security Defenses
  - User Authentication
  - Antivirus
  - Firewalls
  - …

Cryptography as a Security Tool

- 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*
- Allows *secure communications* over an intrinsically *insecure medium*

Friends and Enemies: Alice, Bob, Trudy



- well-known in network security world
- Bob, Alice (lovers!) want to communicate "securely"
- Trudy, the "intruder" may intercept, delete, add messages

What does secure communication mean?

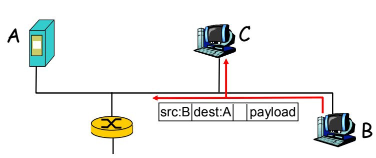
Secrecy: only sender, intended receiver should "understand" msg contents
- sender encrypts msg
- receiver decrypts msg

Authentication: sender, receiver want to confirm identity of each other

Message Integrity: sender, receiver want to ensure message not altered (in transit, or afterwards) without detection

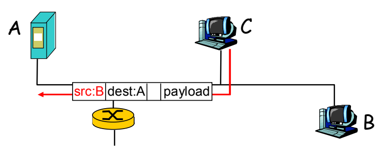
Insecure communication medium

- Packet sniffing:
  - broadcast media
  - promiscuous NIC reads all packets passing by
  - can read all unencrypted data (e.g. passwords)
  - e.g.: C sniffs B's packets

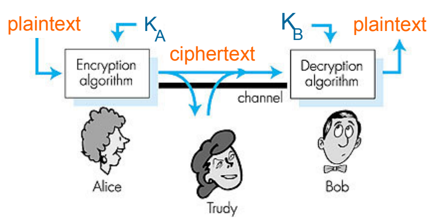


Insecure communication medium

- IP Spoofing
  - can generate "raw" IP packets directly from application, putting any value into IP source address field
  - receiver can't tell if source is spoofed
  - e.g.: C pretends to be B



The language of cryptography



symmetric key crypto: sender, receiver keys identical

public-key crypto: encrypt key *public*, decrypt key *secret*

Symmetric key cryptography

substitution cipher: substituting one thing for another

- Mono-alphabetic cipher: substitute one letter for another

```
 plaintext:  abcdefghijklmnopqrstuvwxyz

ciphertext:  mnbvcxzasdfghjklpoiuytrewq
```

Example

```
 Plaintext: bob. i love you. alice
ciphertext: nkn. s gktc wky. mgsbc
```

How hard to break this simple cipher?

- •brute force (how hard?)
- •other?

Symmetric key crypto: DES

DES: Data Encryption Standard

- US encryption standard [NIST 1993]
- 56-bit symmetric key, 64 bit plaintext input
- How secure is DES?
  - DES Challenge: 56-bit-key-encrypted phrase ("Strong cryptography makes the world a safer place") decrypted (brute force) in 4 months
- making DES more secure
  - use three keys sequentially (3-DES) on each datum
  - use cipher-block chaining

Other Symmetric Algorithms

- DES is most commonly used symmetric block-encryption algorithm (created by US Govt)
- 3-DES considered more secure
- Advanced Encryption Standard (AES), twofish up and coming
- RC4 is most common symmetric stream cipher, but known to have vulnerabilities
  - Encrypts/decrypts a stream of bytes (i.e wireless transmission)
  - Key is a input to pseudo-random-bit generator
    - Generates an infinite keystream

Public Key Cryptography

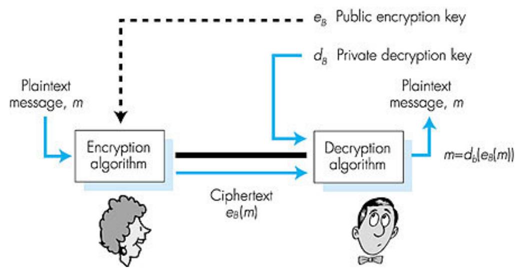
Symmetric key crypto

- requires sender, receiver know shared secret key
- Q: how to agree on key in first place (particularly if never "met")?

Public key cryptography

- radically different approach [Diffie-Hellman76, RSA78]
- sender, receiver do *not* share secret key
- encryption key *public* (known to *all*)
- decryption key private (known only to receiver)

Public key cryptography



Public key encryption algorithms

- Need for public and private keys e_X and d_X
- Two inter-related requirements

  1) $d_X[e_X(m)]=m$

  2) $e_X[d_X(m)]=m$

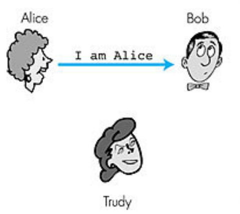
The RSA (Rivest, Shamir, Adelson) algorithm can be used to generate public and private keys

Authentication

- Goal:
  - Bob wants Alice to “prove” her identity to him, before starting communication
- Application areas
  - Server providing a security-critical service (e.g., mail, automatic banking, …)
  - Router that need to establish a secure connection
  - Usage of critical resources (system/network connectivity, …)
  - …

Authentication

Protocol ap1.0: Alice says "I am Alice"

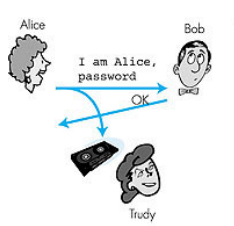


Authentication: another try

Protocol ap2.0: Alice says "I am Alice" and sends her IP address along to "prove" it.

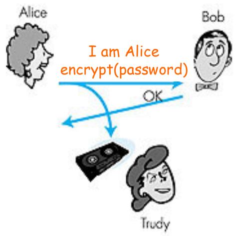
Authentication: another try

Protocol ap3.0: Alice says "I am Alice" and sends her secret password to "prove" it.



Authentication: another try

Protocol ap3.1: Alice says “I am Alice” and sends her *encrypted* secret password to “prove” it.

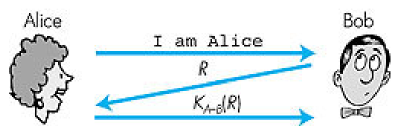


Authentication: yet another try

Goal: avoid playback attack

Nonce: number (R) used only once in a lifetime

ap4.0: to prove Alice “live”, Bob sends Alice nonce, R. Alice must return R, encrypted with shared secret key



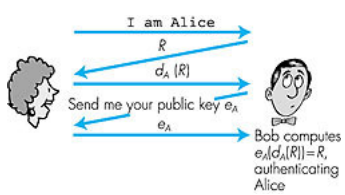
Failures, drawbacks?

Authentication: ap5.0

ap4.0 requires shared symmetric key

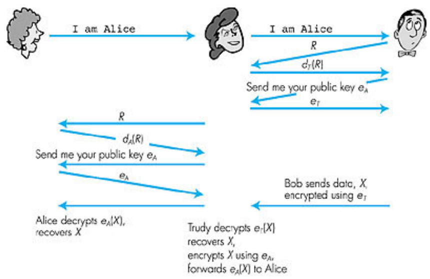
- problem: how do Bob and Alice agree on key?
- can we authenticate using public key techniques?

ap5.0: use nonce, public key cryptography



ap5.0: security hole

Man (woman) in the middle attack: Trudy poses as Alice (to Bob) and as Bob (to Alice)



Digital Signature

- Authentication techniques allow for on-line identification of the remote messages

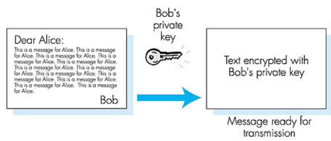
Digital Signature: Requirements

- Cryptographic technique analogous to hand-written signatures.
  - The sender (Bob) digitally signs document, establishing he is the document owner/creator.
- Verifiable
  - The recipient (Alice) can verify and prove that Bob, and no one else, signed the document.
- Non-forgeable
  - The sender can prove that someone else has signed a message
- Non repudiation
  - The recipient (Alice) can prove that Bob signed m and not m'
- Message integrity
  - The sender (Bob) can prove that he signed m and not m'

Digital Signature: Sender

Simple digital signature for message m:

- Bob encrypts m with his public key d_B, creating signed message, $d_B(m)$.
- Bob sends m and $d_B(m)$ to Alice.



Digital Signature: Recipient

- Suppose Alice receives msg *m*, and digital signature $d_B(m)$
- Alice verifies *m* signed by Bob by applying Bob's public key e_B to $d_B(m)$ *then* checks $e_B(d_B(m)) = m$.
- If $e_B(d_B(m)) = m$, whoever signed *m* must have used Bob's private key.

Are requirements satisfied?

Alice thus verifies that:

- Bob signed *m*.
- No one else signed *m*.
- Bob signed m and not *m'*.

Non-repudiation:

- Alice can take *m*, and signature $d_B(m)$ to court and prove that Bob signed *m*.

Message Integrity

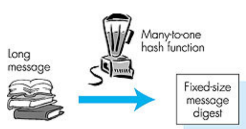
- Bob can prove that he signed m and not *m'*.

Question

- How can Alice achieve Bob's public key?
  - E-mail?
  - Website?
  - ??

Message Digests

- Computationally expensive to public-key-encrypt long messages
- Goal: fixed-length,easy to compute digital signature, "fingerprint"
- Apply hash function H to *m*, get fixed size message digest, *H(m).*



Hash Function

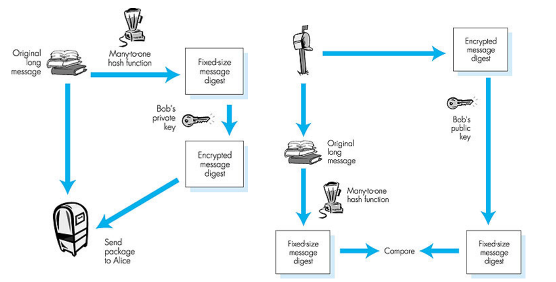
Hash function properties:

- Many-to-1
- Produces fixed-size msg digest (fingerprint)
- Given message digest x
  - computationally infeasible to find m such that x = H(m)
  - computationally infeasible to find any two messages m and m' such that H(m) = H(m').

Digital signature = Signed message digest

Bob sends digitally signed message:

Alice verifies signature and integrity of digitally signed message:



Hash Function Algorithms

- Internet checksum
  - would make a poor message digest.
  - Too easy to find two messages with same checksum.
- MD5 hash function widely used.
  - Computes 128-bit message digest in 4-step process.
  - arbitrary 128-bit string x, appears difficult to construct msg m whose MD5 hash is equal to x.
- SHA-1 is also used.
  - US standard
  - 160-bit message digest

Trusted Intermediaries

Problem:

- How do two entities establish shared secret key over network?

Solution:

- trusted key distribution center (KDC) acting as intermediary between entities

Problem:

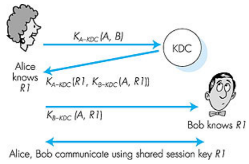
- When Alice obtains Bob's public key (from web site, e-mail, diskette), how does she know it is Bob's public key, not Trudy's?

Solution:

- trusted certification authority (CA)

Key Distribution Center (KDC)

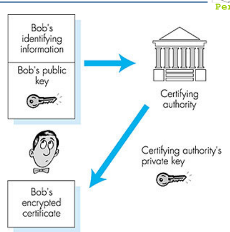
- Alice,Bob need shared symmetric key.
- KDC: server shares different secret key with each registered user.
- Alice, Bob know own symmetric keys, $K_{A\text{-}KDC}$ $K_{B\text{-}KDC}$, for communicating with KDC.



- Alice communicates with KDC, gets session key R1, and $K_{B\text{-}KDC}(A,R1)$
- Alice sends Bob $K_{B\text{-}KDC}(A,R1)$, Bob extracts R1
- Alice, Bob now share the symmetric key R1.

Certification Authorities

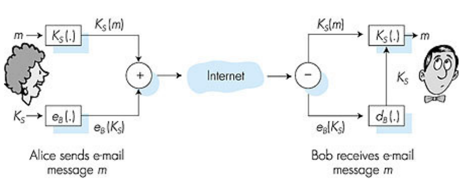
- Certification authority (CA) binds public key to particular entity.
- Entity (person, router, etc.) can register its public key with CA.
  - Entity provides "proof of identity" to CA.
  - CA creates certificate binding entity to public key.
  - Certificate digitally signed by CA.



- When Alice wants Bob's public key:
- gets Bob's certificate (Bob or elsewhere).
- Apply CA's public key to Bob's certificate, get Bob's public key

Secure e-mail

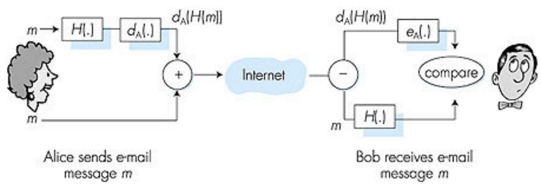
Alice wants to send secret e-mail message, m, to Bob.



- generates random symmetric private key, K_S.
- encrypts message with K_S
- also encrypts K_S with Bob's public key.
- sends both $K_S(m)$ and $e_B(K_S)$ to Bob.

Secure e-mail (Cont'd)

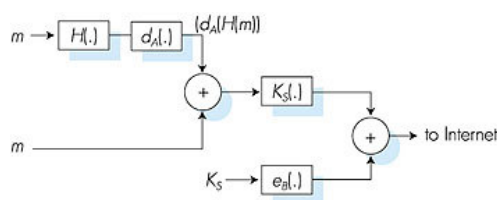
Alice wants to provide sender authentication message integrity.



- Alice digitally signs message.
- sends both message (in the clear) and digital signature.

Secure e-mail (cont'd)

Alice wants to provide secrecy, sender authentication, message integrity.



Note: Alice uses both her private key, Bob's public key.

Pretty good privacy (PGP)

- Internet e-mail encryption scheme, a de-facto standard.
- Uses symmetric key cryptography, public key cryptography, hash function, and digital signature as described.
- Provides secrecy, sender authentication, integrity.
- Inventor, Phil Zimmerman, was target of 3-year federal investigation.

A PGP signed message:

```
---BEGIN PGP SIGNED MESSAGE---
Hash: SHA1

Bob:My husband is out of town
   tonight.Passionately yours,
   Alice

---BEGIN PGP SIGNATURE---
Version: PGP 5.0
Charset: noconv
yhHJRHhGJGhgg/12EpJ+lo8gE4vB3mqJ
   hFEvZP9t6n7G6m5Gw2
---END PGP SIGNATURE---
```

Secure Sockets Layer (SSL)

- PGP provides security for a specific network application
- SSL works at transport layer. Provides security to any TCP-based application using SSL services.
- Cryptographic protocol that limits two computers to only exchange messages with each other
  - Very complicated, with many variations
- Used between browsers and Web servers for secure communication (https)
  - E.g., credit card number in e-commerce applications
- SSL security services:
  - server authentication
  - data encryption
  - client authentication (optional)

SSL Encrypted Session

- Server authentication
  - The server is verified through a certificate assuring that the client is talking to correct server
- Key exchange
  - Asymmetric cryptography used to establish a secure session key (symmetric encryption) for communication
  - Browser
    - generates a symmetric session key K_s
    - encrypts it with server's public key
    - sends encrypted key to server.
  - Server
    - Using its private key, the server decrypts the session key K_s

SSL Encrypted Session

- Secure communication
  - All data sent into TCP socket (by client or server) are encrypted with session key K_s

SSL: Final Remarks

- SSL: basis of IETF Transport Layer Security (TLS).
- SSL can be used for non-Web applications, e.g., IMAP.
- Client authentication can be done with client certificates.

Overview

- Threats and attacks
- Cryptography as a Security Tool
  - Secrecy
  - Authentication
  - Message integrity
  - Digital signature
  - …
- Security Defenses
  - User Authentication
  - Antivirus
  - Firewalls
  - …

Security Defenses

- Defense in depth is most common security theory – multiple layers of security
- Security policy describes what is being secured
- Proactive Approaches
  - Access Control (User Authentication)
  - Firewall
  - Virus Protection
  - …
- Reactive Approaches
  - Auditing, accounting, and logging of all or specific system or network activities
  - Intrusion detection endeavors to detect attempted or successful intrusions

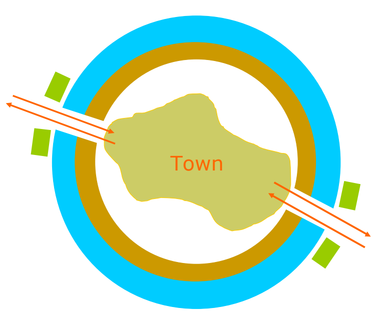
User Authentication

- Crucial to identify user correctly, as protection systems depend on user ID
- User authentication can be based on
  - Something the user *has*
    - key, card, …
  - Something the user *knows*
    - password, …
  - Something the user *is*
    - fingerprint, biometric properties, …

Passwords

- Passwords can be considered a special case of either keys or capabilities
- Passwords must be kept secret
  - Use of "non-guessable" passwords
  - Frequent change of passwords
  - Log all invalid access attempts
- Passwords may also either be encrypted or allowed to be used only once
- Good way to generate password
  - Mg'sniG!
  - My girlfriend's name is Giulia!

Traditional Defense Principle



Lucca’s Walls

Firewall

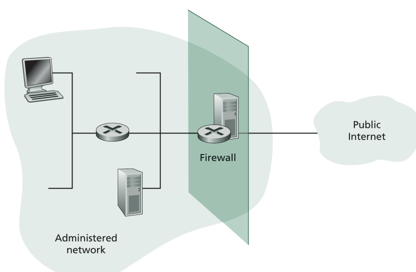
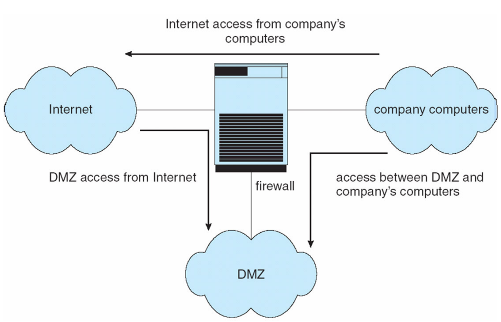


Figure 8.23 ♦ Firewall placement between the administered network and the outside world

Network Security Through Domain Separation



Firewall Classification

- A network firewall is placed between trusted and untrusted hosts
  - The firewall limits network access between these two security domains
- Personal firewall
  - Software module in our host (e.g., PC)
  - Can monitor/limit traffic to and from the host
- Packet Filtering firewall
  - permits/denies input or output of packets based on their IP addresses, port number, …
- Application Gateway
  - understands application protocol and can control them (i.e., SMTP)

Packet Filtering

- Source/Destination IP Address
- Protocol Type in IP datagrams
  - TCP, UDP, ICMP, …
- Source/Destination Port Number
- TCP flags (SYN, ACK, …)
- ICMP Message Type
- …
- Different rules for datagrams leaving/entering the internal network

Packet Filtering Rules

Rule	Source Address	Destination Address	Action	Comments
R1	111.11/16	222.22.22/24	permit	Let datagrams from Bob's university network into a restricted subnet.
R2	111.11.11/24	222.22/16	deny	Don't let traffic from Trudy's subnet into anywhere within Alice's network.
R3	0.0.0.0/0	0.0.0.0/0	deny	Don't let traffic into Alice's network.

Table 8.4 ♦ Packet-filtering rules

Packet Filtering Rules

Datagram Number	Source IP Address	Destination IP Address	Desired Action	Action Under R2, R1, R3	Action Under R1, R2, R3
P1	111.11.11.1 (hacker subnet)	222.22.6.6 (corp.net)	deny	deny (R2)	deny (R2)
P2	111.11.11.1 (hacker subnet)	222.22.22.2 (special subnet)	deny	deny (R2)	permit (R1)
P3	111.11.6.6 (univ. net, not the hacker subnet)	222.22.22.2 (special subnet)	permit	permit (R1)	permit (R1)
P4	111.11.6.6 (univ. net, not the hacker subnet)	222.22.6.6 (corp. net)	deny	deny (R3)	deny (R3)

Table 8.5 ♦ Results of packet filtering, according to rule order

Application Gateway

- Packet filtering only allows general rules
  - Deny input access to all telnet sessions (TCP port number 23)
  - Allow output access to all telnet sessions (TCP port number 23)
- Does not allow to distinguish between different users
  - E.g., Allow input access to all telnet sessions from user / IP address X
  - Possible Solution: Packet filtering router + application gateway

Application Gateway

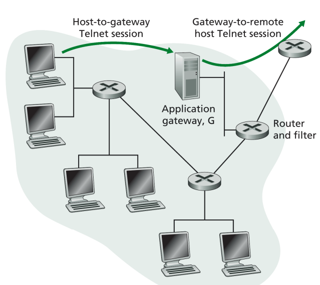


Figure 8.24 ♦ Firewall consisting of an application gateway and a filter

Application Gateway

- Limits
  - Dedicated gateway for each single application
  - Performance degradation
    - All connection must pass through the application gateway
  - The software client must be adapted to contact the application gateway

Firewall Limitations

- Can be tunneled or spoofed
  - Tunneling allows disallowed protocol to travel within allowed protocol (i.e. telnet inside of HTTP)
  - Firewall rules typically based on host name or IP address which can be spoofed
- Often use stringent policies
  - E.g., : Deny all UDP traffics
- May contains configuration bugs
  - That allows potential intruders to overcome security defenses
- May be by-passed
  - Wireless Communications
  - Communications via modem

Questions?