Operating System Security

CS 239, Spring 1995

April 25, 1995

Outline

- Threats
- Basic security principles
- Security on a single machine
- Distributed systems security and data communications security

Security is an Engineering Problem

Always a tradeoff between safety, cost, and inconvenience

Not much solid theory in the field

Hard to provide any real guarantees

- Because making mistakes is easy
- And the nature of the problem is that mistakes are always exploited

Security Threats

Extremely wide range of threats

From a wide variety of sources

Requiring a wide variety of countermeasures

Generally, countering any threat costs something

So people frequently try to counter as few as they can afford

Physical Security

Some threats involve access to the equipment itself

Such as theft,

destruction,

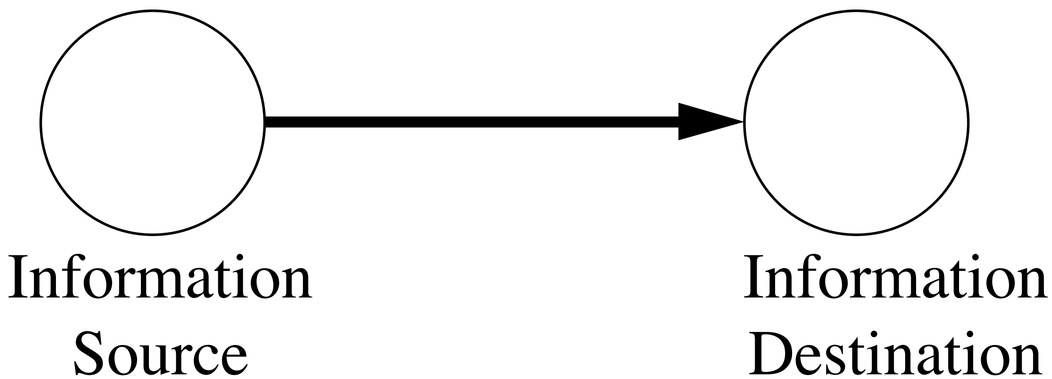
tampering

Physical threats usually require physical prevention methods

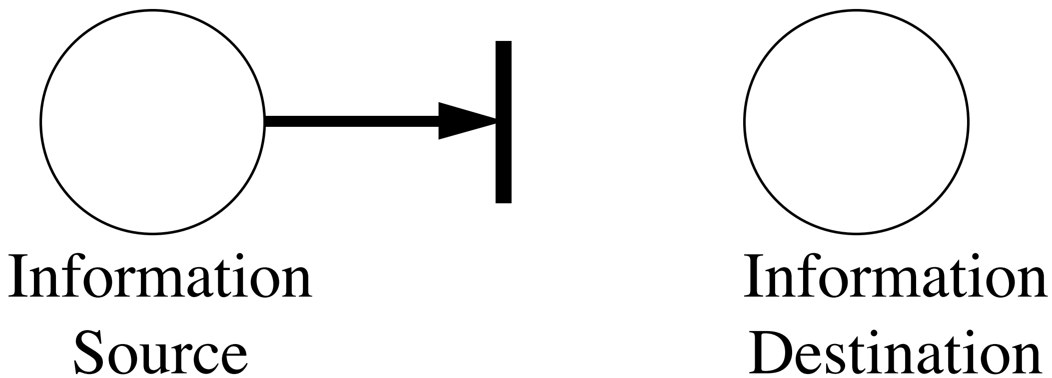
A Classification of Threats

Viewed as types of attacks on normal service

So what is normal service?



Interruption



Interruption Threats

Denial of service

Prevents source from sending information to receiver

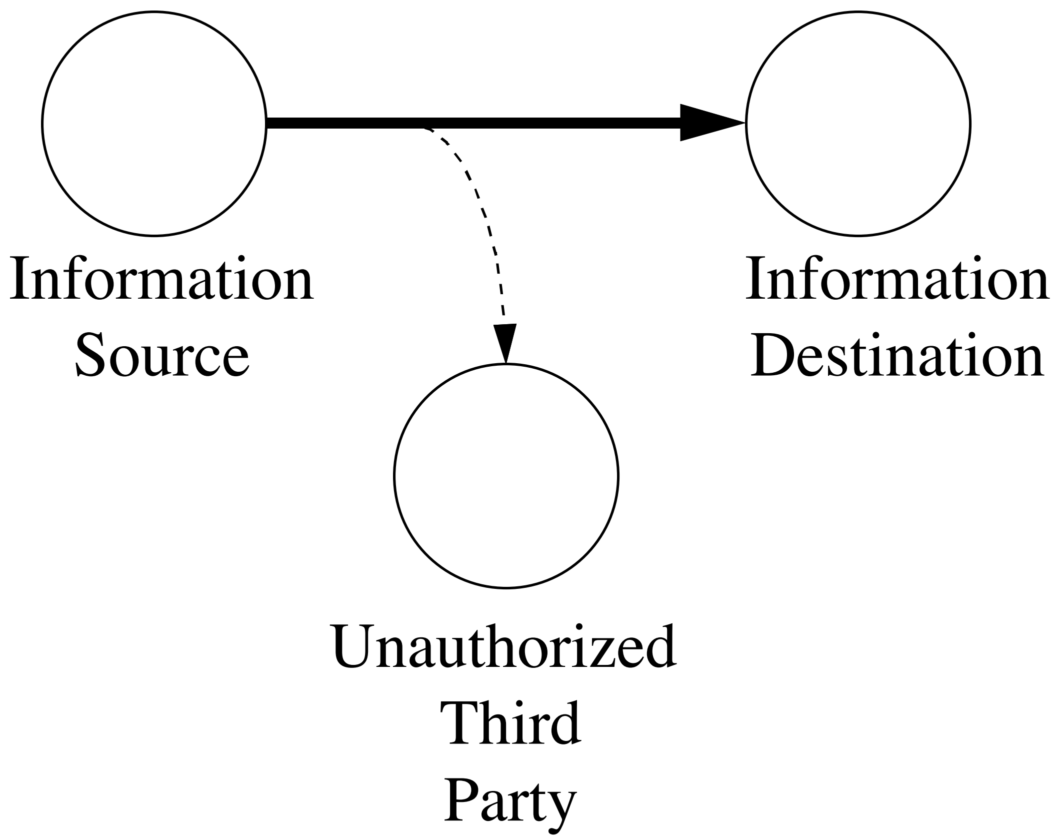
Or receiver from sending requests to source

A threat to availability

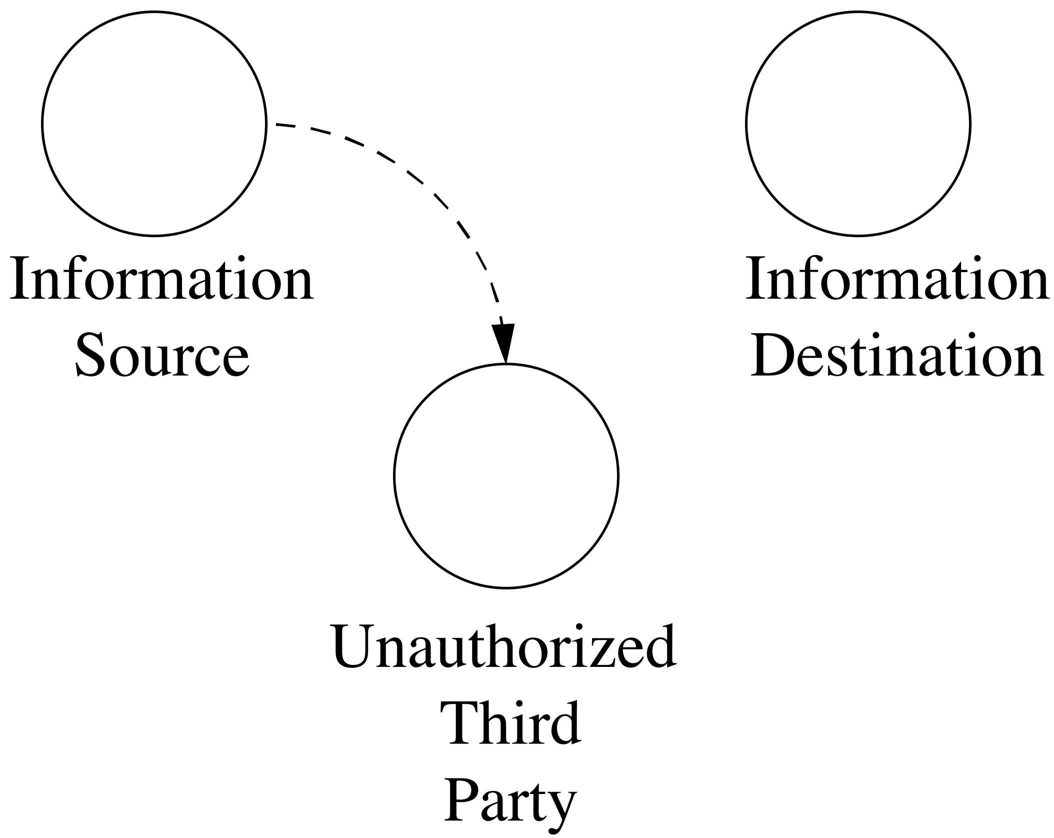
How Does an Interruption Threat Occur?

- Destruction of HW/SW
- Interference with communications channel
- Overloading a shared resource

Interception



Another Type of Interception



Interception Threats

Data or services provided to unauthorized party

Either in conjunction with or independent of authorized access

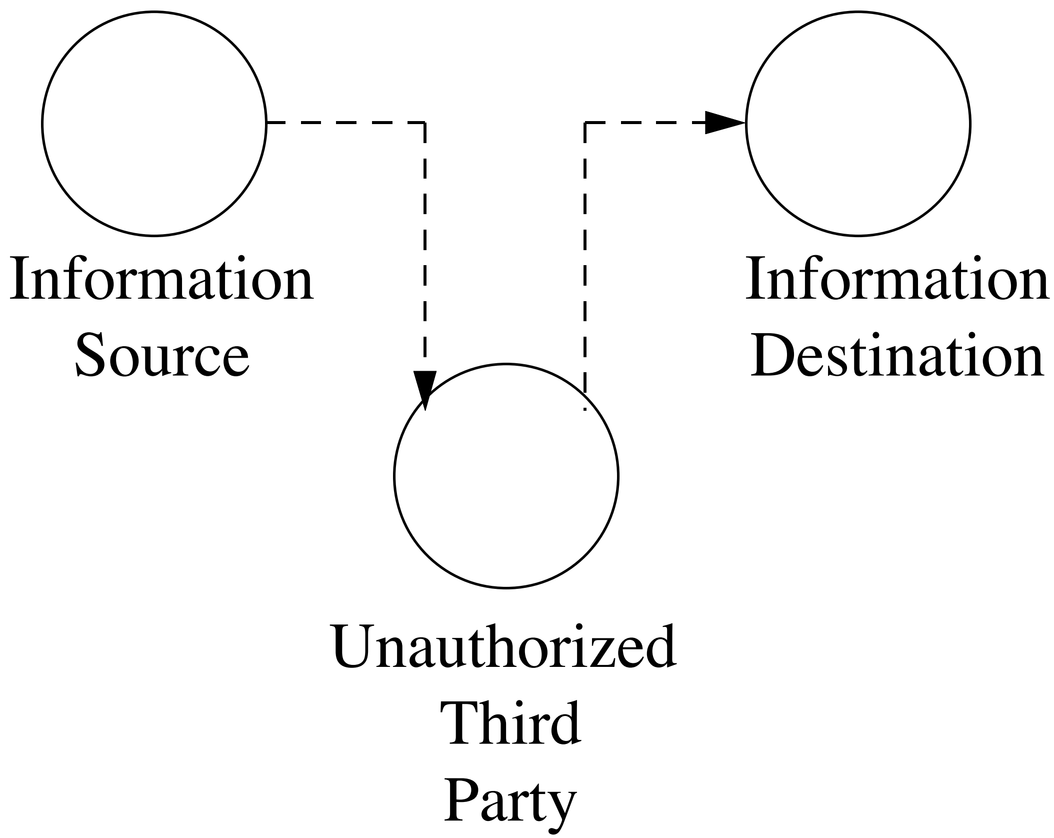
A threat to secrecy

Also a threat to exclusivity

How Do Interception Threats Occur?

- Eavesdropping
- Masquerading
- Break-ins
- Illicit data copying

Modification



Modification Threats

Unauthorized parties modify data

Either on the way to the users

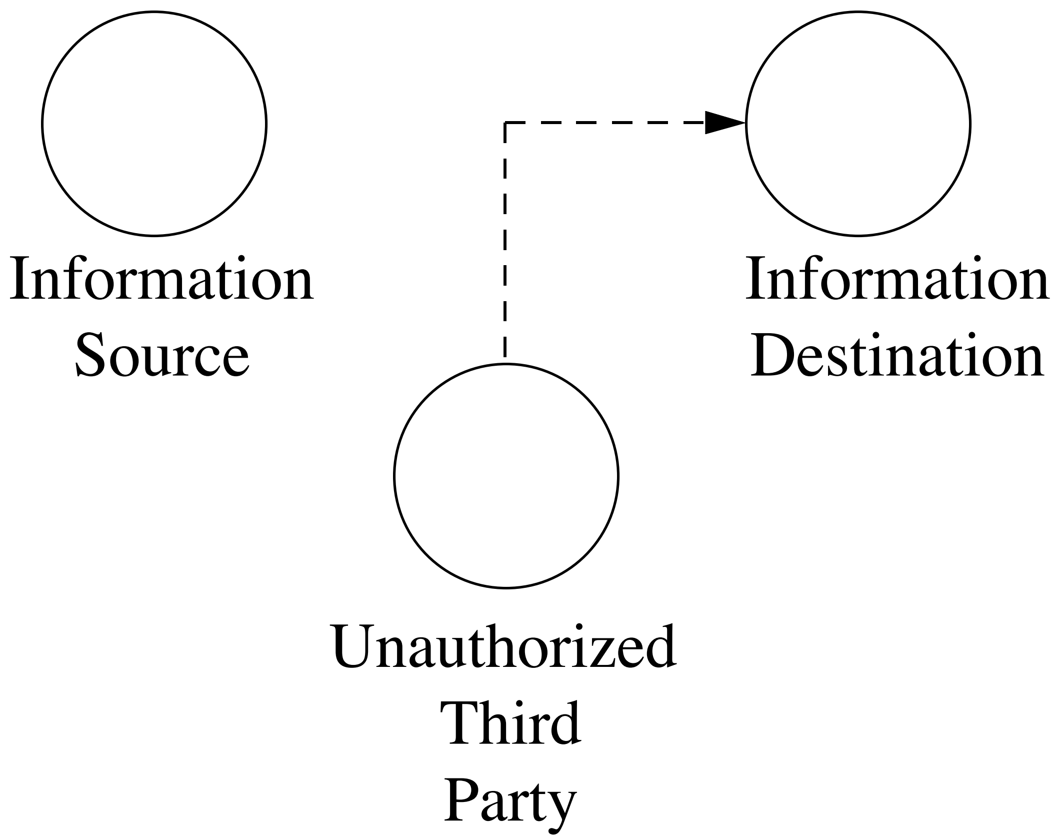
Or permanently at the servers

A threat to integrity

How Do Modification Threats Occur?

- Interception of data requests
- Masquerading
- Illicit access to servers/services

Fabrication



Fabrication Threats

Unauthorized party inserts counterfeit objects into the system

Causing improper changes in data

Or improper use of system resources

A threat to integrity

How Do Fabrication Threats Occur?

- Masquerading
- Bypassing protection measures
- Duplication of legitimate requests

Active Threats vs. Passive Threats

Passive threats are forms of eavesdropping

- No modifications, injections of requests, etc., occur

Active threats are more aggressive

Passive threats are mostly to secrecy

Active threats are to availability, integrity, exclusivity

What Are We Protecting?

- Hardware
- Software
- Data
- Communications lines and networks
- “Cycles” and economic values

Basic Security Principles

- Terms and concepts
- Mechanisms

Security and Protection

Security is a policy

- E.g., “no unauthorized user may access this file”

Protection is a mechanism

- E.g., “the system checks user identity against access permissions”

Protection mechanisms implement security policies

Design Principles for Secure Systems

- Economy
- Complete mediation
- Open design
- Least privilege
- Least common mechanism
- Acceptability
- Fail-safe defaults

Sharing Security Spectrum

- No protection
- Isolation
- Share all or nothing
- Share with access limitations
- Share with dynamic capabilities
- Limit use of objects

Authentication

If a system supports more than one user, it must be able to tell who's doing what

I.e., all requests to the system must be tagged with user identity

Authentication is required to assure system that the tags are valid

Important Security Mechanisms

- Encryption
- Passwords
- Other authentication mechanisms
- Access control mechanisms

Encryption

Various algorithms can be used to make data unreadable to intruders

This process is called *encryption*

Typically, encryption uses a secret key known only to legitimate users of the data

Without the key, decrypting the data is computationally infeasible

Encryption Example

M is the *plaintext* (text to be encrypted)

E is the encryption algorithm

$\mathbf{K_e}$ is the key

C is the ciphertext (encrypted text)

$$\mathbf{C = E(M,K_e)}$$

Decrypting the Ciphertext

C is the ciphertext

D is the decryption algorithm

$\mathbf{K_d}$ is the decryption key

$$\mathbf{M = D(C, K_d)}$$

Symmetrical Encryption

Many common encryption algorithms are symmetrical

I.e., $\mathbf{E} = \mathbf{D}$ and $\mathbf{K_e} = \mathbf{K_d}$

Some important encryption algorithms are not symmetrical, however

Encryption Security Assumptions

Someone trying to break the encryption knows-

- The algorithms E and D
- Arbitrary amounts of matching plaintext and ciphertext

But does not know the keys

Strength of system based on difficulty to break under these assumptions

Practical Security of Encryption

Most encryption algorithms can be broken

Goal is to make breaking them too expensive to bother

Security often depends on length of key

And the more data sent with a given key, the greater the chance of compromise

Passwords

A fundamental authentication mechanism

A user proves his identity by supplying a secret

- Either at login or other critical time

The secret is the password

Password Security

- Password selection
- Password storage and handling
- Password aging

Selecting a Password

Desirable characteristics include:

- Unguessable
- Easy to remember (and type)
- Not in a dictionary
- Too long to search exhaustively

Password Storage and Handling

Passwords are secrets, so their security depends on careful handling

But seemingly the system must store the password

- To compare when users log in

If system storage is compromised, so is all authentication

Securely Storing Passwords

Store only in encrypted form

To check a password, encrypt it and compare to the encrypted version

Encrypted version can be stored in a file

But there are tricky issues

Example: The Unix Password File

Each password has an associated *salt*

Unix encrypts a block of zeros

- Key built from password plus 12-bit salt
- Encryption done with DES

Stored information = E(zeros, salt+password)

To check password, repeat operations

Why do this?

Does This Solve the Problem?

Not entirely

Passwords exist in plaintext in process checking them

Passwords may travel over communications lines in plaintext

Problems With Passwords

- People choose bad ones
- People forget them
- People reuse them
- People rarely change them

How To Deal With Bad Passwords

- Educate users so they choose good ones
- Automatic password generation
- Check when changed
- Periodically run automated cracker

Any solution must balance user needs, password security, and resources

Other Authentication Mechanisms

- Challenge/response
- Smartcards
- Other special hardware
- Detection of personal characteristics

All have some drawbacks

Some are combined with passwords

Data Access Control Mechanisms

Methods of specifying who can access what in which ways when

Based on assumption that the system has authenticated the user

Access Matrix

Describes permissible accesses for the system

Subjects access *objects* with particular *access rights*

A theoretical concept, never kept in practice

Access Matrix Example

	File 1	File 2	Server X	Segment 57
User A	Read Write	None	Query	Read
User B	Read	Write	Update	None
User C	None	Read	Start/Stop	None
User D	None	None	Query	None

Methods for Implementing Access Matrix

- Access control lists

  Decomposition by columns
- Capabilities

  Decomposition by rows

Access Control Lists

Each object keeps track of who can access it

Each object has an access control list

Add subjects by adding entries

Remove subjects by removing entries

\+ Easy to determine who can access object

  Easy to change who can access object

– Hard to tell what someone can access

Access Control List Example

File 1's ACL

User A: Read
Write
User B: Read

Segment 57's ACL

User A: Read

Capabilities

Each subject keeps track of what it can access

Typically by keeping a *capability* for each object

Capabilities are like admission tickets

+ Easy to tell what a subject can access

– Hard to tell who can access an object

  Hard to revoke/control access

Capability Example

User A's Capabilities	
File 1:	Read
	Write
Server A:	Query
Segment 57:	Read

User A's Capabilities	
File 1:	Read
File 2:	Write
Server A:	Update

Other Models of Access Control

- Military model
- Information flow models
- Lattice model of information flow