COVERT CHANNELS

• A covert channel is a communication channel based on the use of system resources not normally intended for communication between the subjects (processes) in the system
COVERT CHANNELS

• HARMLESS CASES
  – The channel parallels an overt channel (and is therefore legal)
  – The sender and receiver are the same process (mumbling channel)

• HARMFUL CASE
  – The sender and receiver are not permitted to communicate under the given security policy

Information is leaked unknown to the high principal

High Principal

High Trojan Horse
Infected Subject

COVERT CHANNEL

Low Principal

Low Trojan Horse
Infected Subject
COVERT CHANNELS

• The concern is with subjects not users
  – users are trusted (must be trusted) not to disclose secret information outside of the computer system
  – subjects are not trusted because they may have Trojan Horses embedded in the code they execute

• star-property prevents overt leakage of information and does not address the covert channel problem

COVERT CHANNELS

• Computer systems abound with covert channels
• Covert channels are typically noisy but information theory techniques can be used to achieve error-free communication
COPING WITH COVERT CHANNELS

- identification
  - close the channel or slow it down
  - tolerate it
    - estimate the bandwidth
    - audit occurrence of events involved in usage of the channel

STORAGE VS. TIMING CHANNELS

- STORAGE CHANNELS
  - use system variables and attributes (other than time) to signal information
  - classic example is resource exhaustion channel
- TIMING CHANNELS
  - vary the amount of time required to complete a task to signal information
  - classic example is load sensing channel
RESOURCE EXHAUSTION CHANNEL

- Given 5MB pool of dynamically allocated memory
- HIGH PROCESS
  - bit = 1 → request 5MB of memory
  - bit = 0 → request 0MB of memory
- LOW PROCESS
  - request 5MB of memory
  - if allocated then bit = 0 otherwise bit = 1

LOAD SENSING CHANNEL

- HIGH PROCESS
  - bit = 1 → enter computation intensive loop
  - bit = 0 → go to sleep
- LOW PROCESS
  - perform a task with known computational requirements
  - if completed quickly then bit = 0 otherwise bit = 1
SOME SIMPLE STORAGE CHANNELS

- file names
- file attributes
  - size
  - date modified
  - protection bits
  - access control lists
- file status
  - open or closed
  - locked or unlocked
- file existence

once identified these are relatively easy to close (in principle)

RESOURCE EXHAUSTION CHANNELS

- can be closed by static resource allocation across security classes at the cost of resource utilization
- can be audited to detect attempted usage
TIMING CHANNELS

• timing channels arising due to low level hardware mechanisms can be very fast (Mbits/second)
• the faster the hardware the faster the timing channel
• examples:
  – cache
  – system bus contention
  – paging delays
  – multiprocessor interconnection networks
• practically impossible to audit

DUALITY OF TIMING AND STORAGE CHANNELS

• many storage channels can be converted to timing channels and vice versa
• for example: channels based on sensing disk delays can be formulated as
  – timing channels since it is the delay which is being measured, or
  – storage channels attributed to the system variable which represents the position of the disk arm