# March 6, 2014

# Mitigation of Covert Channels

- Problem: these work by varying use of shared resources
- One solution
  - Require processes to say what resources they need before running
  - Provide access to them in a way that no other process can access them
- Cumbersome
  - Includes running (CPU covert channel)
  - Resources stay allocated for lifetime of process

# Alternate Approach

- Obscure amount of resources being used
  - Receiver cannot distinguish between what the sender is using and what is added
- How? Two ways:
  - Devote uniform resources to each process
  - Inject randomness into allocation, use of resources

# Uniformity

- Varieties of Isolation
  - Process can't tell if second process using resource
- Example: KVM/370 covert channel via CPU usage
  - Devote uniform resources to each process
  - Give each VM a time slice of fixed duration
  - Do not allow VM to surrender its CPU time
    - Can no longer send 0 or 1 by modulating CPU usage

# Randomness

- Make noise dominate channel
  - Does not close it, but makes it useless
- Example: MLS database
  - Probability of transaction being aborted by user other than sender, receiver approaches 1
    - $q \to 1$
  - $I(A; X) \to 0$
  - How to do this: resolve conflicts by aborting increases $q$, or have participants abort transactions randomly

# Problem: Loss of Efficiency

- Fixed allocation, constraining use
  - Wastes resources
- Increasing probability of aborts
  - Some transactions that will normally commit now fail, requiring more retries
- Policy: is the inefficiency preferable to the covert channel?

# Example

- Goal: limit covert timing channels on VAX/VMM
- "Fuzzy time" reduces accuracy of system clocks by generating random clock ticks
  - Random interrupts take any desired distribution
  - System clock updates only after each timer interrupt
  - Kernel rounds time to nearest 0.1 sec before giving it to VM
    - Means it cannot be more accurate than timing of interrupts

# Example

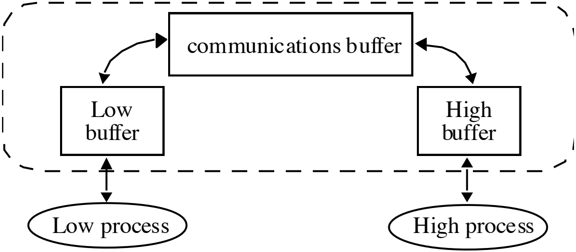
- I/O operations have random delays
- Kernel distinguishes 2 kinds of time:
    - *Event time* when I/O event occurs
    - *Notification time* when VM told I/O event occurred
        - Random delay between these prevents VM from figuring out when event actually occurred)
        - Delay can be randomly distributed as desired (in security kernel, it's 1–19ms)
    - Added enough noise to make covert timing channels hard to exploit

# Improvement

- Modify scheduler to run processes in increasing order of security level
  - Now we're worried about "reads up", so ...
- Countermeasures needed only when transition from dominating VM to dominated VM
  - Add random intervals between quanta for these transitions

# The Pump

- Tool for controlling communications path between *High* and *Low*



# Details

- Communications buffer of length $n$
  - Means it can hold up to $n$ messages
- Messages numbered
- Pump ACKs each message as it is moved from *High* (*Low*) buffer to communications buffer
- If pump crashes, communications buffer preserves messages
  - Processes using pump can recover from crash

# Covert Channel

- Low fills communications buffer
  - Send messages to pump until no ACK
  - If *High* wants to send 1, it accepts 1 message from pump; if *High* wants to send 0, it does not
  - If *Low* gets ACK, message moved from *Low* buffer to communications buffer $\Rightarrow$ High sent 1
  - If *Low* doesn't get ACK, no message moved $\Rightarrow$ *High* sent 0
- Meaning: if *High* can control rate at which pump passes messages to it, a covert timing channel

# Performance vs. Capacity

- Assume *Low* process, pump can process messages more quickly than *High* process
- $L_i$ random variable: time from *Low* sending message to pump to *Low* receiving ACK
- $H_i$ random variable: average time for *High* to ACK each of last $n$ messages

# Case 1: $E(L_i) > H_i$

- *High* can process messages more quickly than *Low* can get ACKs
- Contradicts above assumption
  - Pump must be delaying ACKs
  - *Low* waits for ACK whether or not communications buffer is full
- Covert channel closed
- Not optimal
  - Process may wait to send message even when there is room

# Case 2: $E(L_i) < H_i$

- *Low* sending messages faster than *High* can remove them
- Covert channel open
- Optimal performance

# Case 3: $E(L_i) = H_i$

- Pump, processes handle messages at same rate
- Covert channel open
    - Bandwidth decreased from optimal case (can't send messages over covert channel as fast)
- Performance not optimal

# Adding Noise

- Add noise to approximate case 3
  - Covert channel capacity reduced to $1/nr$ where $r$ time from *Low* sending message to pump to *Low* receiving ACK when communications buffer not full
  - Conclusion: use of pump substantially reduces capacity of covert channel between *High*, *Low* processes when compared to direct connection

# About Elections

Voters:

- In the U.S., states manage elections
- In some states, counties manage the elections locally
- Many different jurisdictions with many different rules
- Great commonality in rules, procedures among the different jurisdictions

# How an Election Works in Yolo County, CA

Voters:

- Go to polling station
- Give name, get ballot
- Enter booth, vote using marker to mark ballot
- Put ballot in protective sleeve (envelope)
- Leave booth, drop envelope into ballot box

# End of the Day

- Election officials take ballot box to County seat
- Election officials remove ballots from envelopes
  - If provisional, handled differently
- Ballots counted, put into bags marked with precinct and count
- Ballots removed from bag, run through automatic counters (scanners)
  - Humans intervene when problems arise
  - Intermediate tallies written onto flash cards
  - Every so often, cards removed, walked to tally computer
- Tallies periodically updated, given to web folks

# The Canvass

Required by California law:

- Ballots for 1% of precincts counted by hand
  - Must include all races!
- Compare to tallies from election
  - If different, check until problem found
- Certify final counts to Secretary of State
  - . . . within 28 days of the election

Actually, Yolo County also does more checking, including testing other proposed auditing methods with trusted researchers

# Over- and Under-Votes

- Three seats open in Davis City Council election
- **Overvote**: voting too many times
  - Vote for 4 candidates
  - No votes in that race counted
- **Undervote**: voting too few times
  - Vote for 2 candidates
  - Both votes counted; no third vote counted

# What's an "E-Voting System"?

- Intended to replace paper
  - Improve clarity of cast vote
  - Less error-prone to errors in counting
  - Easier to store
- Casting votes
  - Direct Recording Electronic (with or without VVPATs)
  - Ballot Marking Devices
  - Pens and paper
- Counting votes
  - Scanning at precinct (Precinct-Count Optical Scan)
  - Scanning at Election Central
  - Computer counting of electronic ballots

# What Should It Do?

- Summary: replace technology used in election process with better technology
  - "Better" means that the technology improves some aspect of the election process
- Examples
  - Easier to program ballots than print ballots
  - Can handle multiple languages easily
  - Easier to tally than hand counting

# Requirements for an Election

- Voter validation (authenticated, registered, has not yet voted)
- Ballot validation (voter uses right ballot, results of marking capture intent of voter)
- Voter privacy (no association between voter, ballot; includes voter showing others how he/she voted)
- Integrity of election (ballots not changed, vote tallied accurately)

# Requirements for an Election

- Voting availability (voter must be able to vote, materials must be available)
- Voting reliability (voting mechanisms must work)
- Election transparency (audit election process, verify everything done right)
- Election manageability (process must be usable by those involved, including poll workers)