Logistics

• HW3 Due Tonight, 11:59 PM
• Course Evals Fixed
• Review Session for Wednesday's Lecture
• Final Exam: Thursday, 5-8PM
Layering

• Complex services from simpler ones
  1. Physical and Link Layers (Wi-Fi, Ethernet, …)
     • Unreliable, local exchange of limited-size frames
  2. Network (IP) – routing between local networks
     • Unreliable, global exchange of limited-size packets
  3. Transport (e.g., TCP) – Glue
     • Reliable (with retries), ordering, stream of bytes
  4. Application – Everything on top of sockets
Broadcast Networks

- Shared communication medium: A Bus
Broadcast Networks

• Shared Communication Medium

• Examples:
  • Original Ethernet
  • All types of wireless (WiFi, Cellular Networks, …)
  • Coaxial Cable (e.g., Cable Internet)
How do we get unicast (message to one host) from broadcast (message to all hosts)?

Put Header on Front of Packet: Dest | Packet

- Often handled directly in hardware
How do we specify a destination?

• Link Layer has its own addressing scheme: **Media Access Control (MAC) Addresses**

• MAC Address: 48 bits uniquely bound to your hardware device (hard coded by manufacturer)

• Remember: MAC Addresses only used for exchange of a frame across a single link/broadcast domain

• IP Address can change (e.g. you move from campus to home network)
  • MAC address is constant
**Data Link Layer**

- MAC Dest. Address
- MAC Src. Address
- …
Media Access Control

- **Arbitration:** Who can use the broadcast medium when?
- Early example: Aloha Network (1970's), packet radio within Hawaii
  - Use checksum in frame header to detect error
  - If two senders try to send at same time, both packets get garbled, wait and re-send later
- **Problems:**
  - If network is too busy, no one gets through
  - Need senders to retry again at different times
Example: Original Ethernet

- All computers communicate on shared wire (bus)
- **CSMA/CD**: Carrier Sense Multiple Access with **Collision Detection**
- Carrier Sense: Don't send unless medium is idle
  - *Listen before Speaking*
- Collision Detection: Determine if sent packet was trampled by someone else on bus
  - If so, abort, wait and retry
Adaptive Randomized Backoff

• Wait a random amount of time before retransmitting again

• Why? Two machines involved in collision don't want to retransmit at same time
  • Just causes another collision!

• Increase wait time after each retry to adjust to how busy the shared medium is
Original Ethernet

- All frames delivered to all hosts
- If host is not intended recipient, drops frame
Link Layer Switches

• Inspects destination MAC address of incoming packet
• Forwards on relevant outgoing link
Switches vs Routers

• Routers operate at network layer, understand IP addresses
  • Build routing tables by exchanging information with neighbors, can also manually configure

• Switches operate at link layer, understand MAC addresses
  • "Self Learning" – build switching tables automatically by inspecting source MAC address of frames received on different links
  • Unknown MAC address: just broadcast
Point-to-Point Networks

- Switches make an Ethernet LAN operate more like a **point-to-point network**
- No shared medium: Physical wire connected to only two specific nodes
Putting it all together

Application Layer

Transport Layer

Network Layer

Datalink Layer

Physical Layer

Transport Layer

Network Layer

Datalink Layer

Physical Layer

101010100110101110

Data

Trans. Hdr.

Net. Hdr.

Frame Hdr.

101010100110101110
Putting it all Together

- One technicality: a Switch works at the data link layer (can parse frame headers) but is transparent to communication endpoints.
- Host does not specify Switch MAC address as destination of frame.
Bonus: Wi-Fi (802.11)

• Some challenges unique to wireless setting:
  • Interference (other transmissions, environment)
  • Signal attenuation
  • Multipath Propagation

• Typically: Traverse one wireless link to **access point**, then we're back at a wired link layer (e.g., Ethernet)
Wireless LAN Architecture

Internet

Router

Access Point

Access Point

Laptop

Laptop

Laptop

Laptop
**Wi-Fi Access Point**

- Bridges last-hop wireless link with wired network that leads to local router (and therefore to the Internet)

- Periodically sends *beacon frames* with
  - Network Name (SSID), e.g. *Airbears 2*
  - Access Point's MAC address

- Your laptop listens for these beacon frames when you are looking for a network to join
  - May also broadcast a "probe frame" to solicit beacons
Media Access Control Challenges

- **Hidden Terminal Problem**
- Two nodes may not know they are both transmitting
Media Access Control Challenges

- Fading: Signals Weaken with Distance
- Two nodes may not know they are both transmitting
Wi-Fi: CSMA/CA

- **Carrier Sense Multiple Access with Collision Avoidance**

- Why not collision detection like Ethernet?
  - Two nodes may not be able to perceive each other's transmissions (e.g., hidden terminal problem)
  - Even in "perfect" conditions, very hard to receive (detect channel use) while you are sending
Wi-Fi: Acknowledgements

• Acknowledgement/Retransmission scheme is implemented in the link layer.
• Doesn't this violate the end-to-end principle?
  • Strict interpretation: leave reliability to TCP at end hosts.
• But likelihood of errors (e.g. corrupted packet from interference or overlapping transmission) high enough to warrant addressing it here.
Transmitting a Frame

1. Check if channel is idle
2. If in use, choose a random backoff length
   - Count down only while channel is idle
   - Stop counting while it is in use
3. Transmit frame, wait for ACK
4. If success, move on to next frame. Otherwise, pick a larger backoff period, return to (2)
Break
Linux Virtual memory map (old)

32-Bit Virtual Address Space

Kernel Addresses

User Addresses

0x00000000

0xC0000000

0xFFFFFFFF

3GB Total

1GB Physical

0x00000000

0xFFFFFFFF

896MB

128TiB

1GB

128TiB

64 TiB Physical

“Canonical Hole”

0xFFFF800000000000

0x00007FFFFFFF

64-Bit Virtual Address Space

User Addresses

Kernel Addresses

Empty Space
January 2018 - Meltdown

• Possible to inspect contents of kernel memory if it's mapped into address space (even as user!)

• Fix: Kernel Page Table Isolation
  • Use entirely different page tables when in user mode vs. when in kernel mode

• Problem: Address space change whenever an interrupt or syscall occurs!
  • Change page tables
  • Flush TLB unless it is tagged

• Reduced Performance, depends on syscall workload
Meltdown Code

```c
uint8_t array[256 * 4096];
// Flush array from CPU cache
// Install signal handler for SIGSEGV
uint64_t priv_addr = 0xFFF...;
uint8_t data = *(uint8_t*)(priv_addr);

uint8_t dummy = array[data * 4096];
for (int i = 0; i < 256; i++) {
    // Check if array[i * 4096] is cached
}
```
Takeaway Points

• Intel CPUs did not check virtual memory permissions on speculative execution

• Eventually this gets cleaned up, but some effects persist (most importantly: data in cache)

• We can then observe how long it takes to access different elements from our test array
  • Slot that was cached uniquely determined by value of data in kernel memory region!

• Timing Side Channel Attack