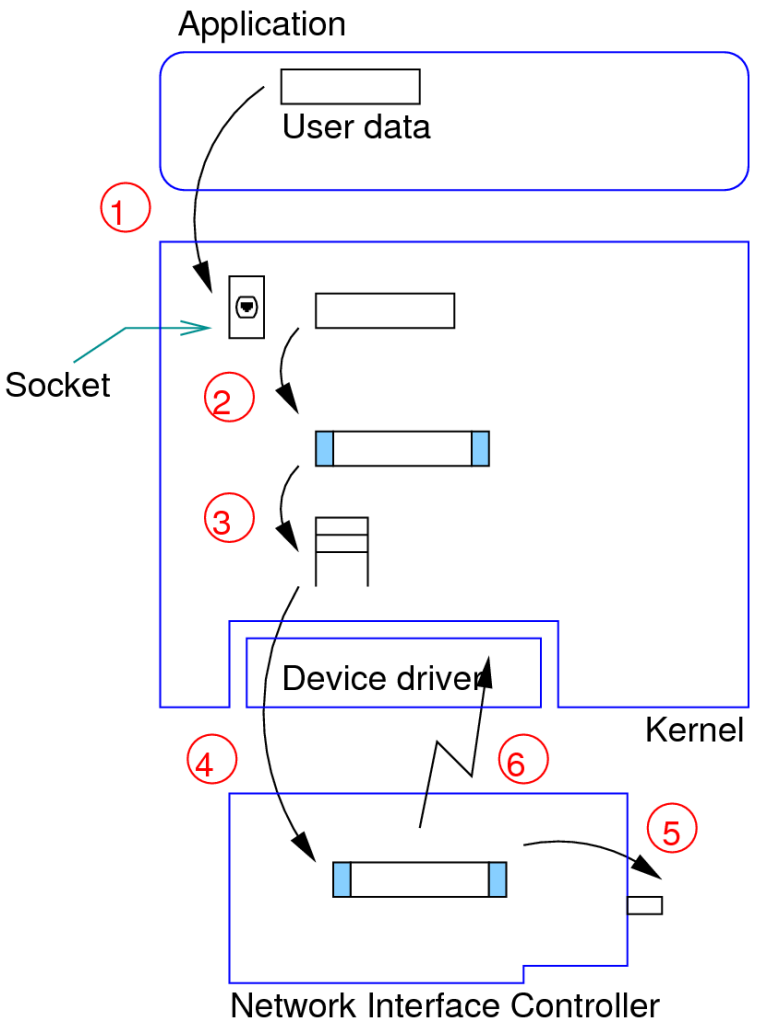
Network Performance Troubleshooting

Imed Chihi

Agenda

- The transmit-receive sequences
- Interrupt handling
- Sockets
- The TCP state machine
- Lab - Networking: an application view
- Lab - Networking: the packet level
- Kernel buffers
- TCP sliding window
- IP fragmentation and re-assembly
- Case study: NFS tuning

Transmission



(1) Application calls `write()` to write to the socket and the data is copied to the kernel space

(2) Kernel hands the data to the IP stack to encapsulate it in a PDU

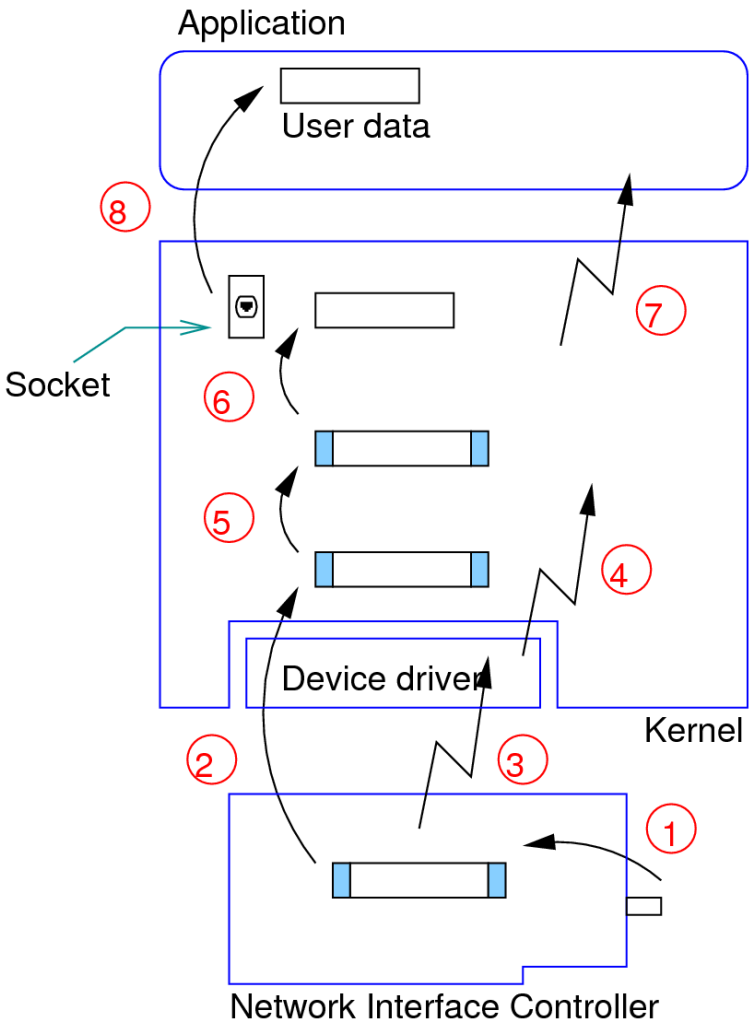
(3) Kernel puts the PDU on a transmit queue for the device in question

(4) Device driver picks PDUs from the queue and copies them to the NIC

(5) NIC sends the PDU over the wire

(6) NIC notifies the kernel about the transmission completion by raising an interrupt

Receive



(1) NIC receive PDU from the network

(2) NIC copies PDU into kernel buffers using DMA

(3) NIC notifies kernel about the received PDU by raising an interrupt

(4) Device driver acknowledges the interrupt and schedules a *softirq*

(5) Kernel hands the PDU up to the IP stack for routing decisions

(6) Kernel extracts the data and copies it to the corresponding socket buffer

(7) Kernel notifies all processes waiting on the socket

(8) The `read()` call issued by the application proceeds and data is copied into the

Interrupt handling - 1

- Interrupts are handled by an interrupt controller: Advanced Programmable Interrupt Controller (APIC)
- NIC raises a hard interrupt at each event: packet reception or transmission
- Hard interrupt handling cannot be interrupted, may lockup the system if it sleeps or takes too long to complete
- Hard interrupt handlers perform minimal work and schedule the remainder of the job to be handled asynchronously by a softirq
- Softirqs are processed as regular kernel code by special kernel threads: `ksoftirqd/`*X*

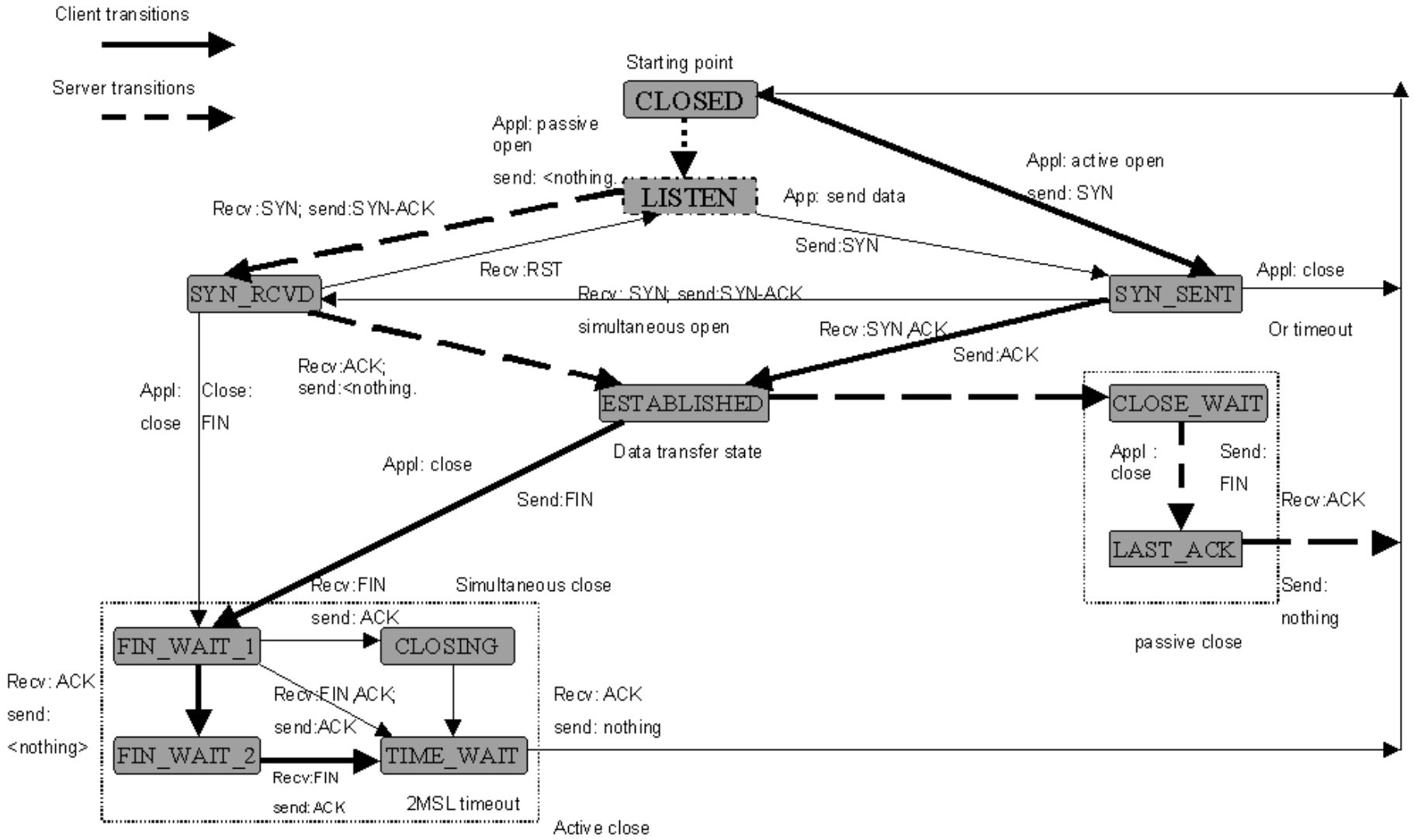
Interrupt handling - 2

- Kernel will drop packets if it cannot pick them from the NIC quickly enough
- Hard interrupts are reports in: `/proc/interrupts`
- `irqbalance` balances hard interrupts across CPUs
- Hard interrupt cost can be mitigated by interrupt coalescing

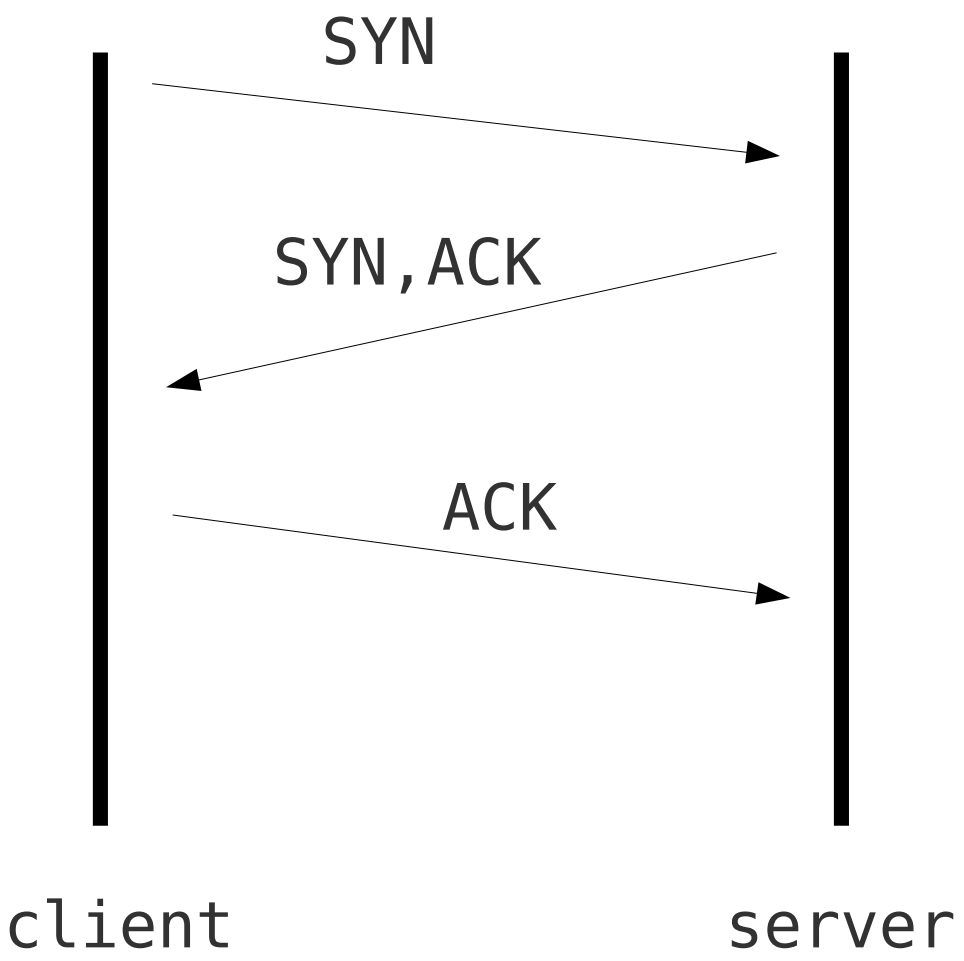
Sockets

- A software construct used to present a file-like interface to network communications
- Analogous to file system objects handling: `socket()`, `read()`, `write()` and `close()`
- The `socket()` system call returns a file handle used to refer to the connection in subsequent operations

The TCP state machine



TCP 3-way handshake



- SYN and ACK are two out of 5 flags used by TCP
- *SYN floods* are massive numbers of connection attempts without subsequent ACKs
- A duplex connection is established
- One party can close its end of the connection unilaterally
- Too many half closed connections may exhaust the allowed number of sockets in the `TIME_WAIT` state
- Might use `net.ipv4.tcp_tw_recycle`

Lab 1

Usage of sockets

Inspecting TCP socket states

Inspecting packets with tcpdump

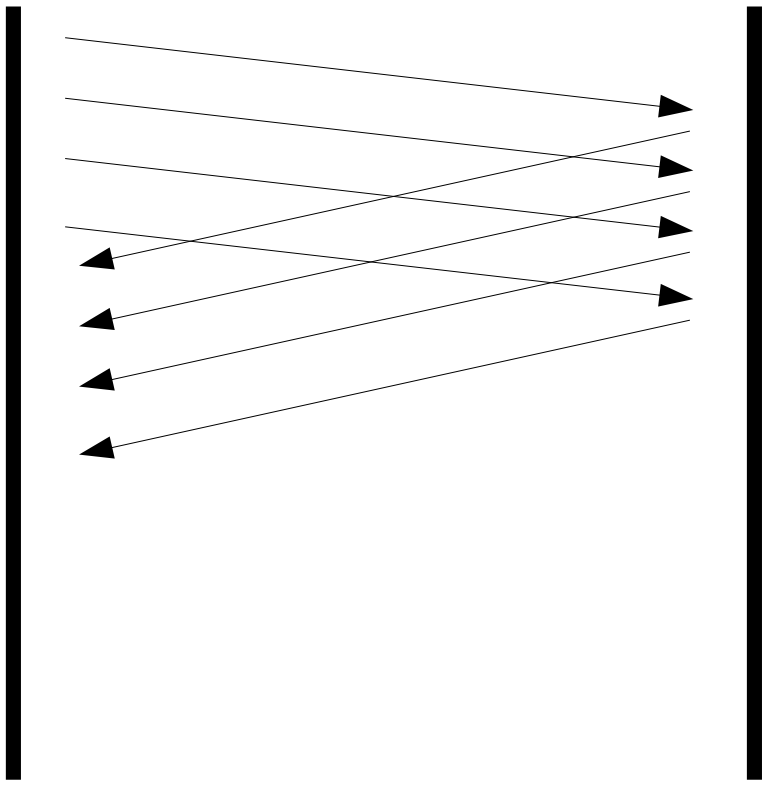
Inspecting some NIC hardware details

Optimising kernel buffers

- Used for DMA transfers from NIC, socket buffers, fragmentation and re-assembly
- Buffers consume the low memory zone which might be problematic on IA32 architectures
- TCP reports its buffer size to the sender in order to enforce a flow control
- Can be adjusted by:

```
net.core.rmem_default
net.core.rmem_max
net.core.wmem_default
net.core.wmem_max
```

The bandwidth-delay product



- If the sender waits until an ACK is received for each packet sent, then the transmission will be inefficient
- Transmitter anticipates the reception of ACKs and sends multiple packets
- The transmitter needs to keep a copy of the packet until it receives an ACK for it. It might be lost
- The receiver reports the size of its buffer (window) with each ACK
- Before receiving the first ACK, the transmitter can send bandwidth*RTT

Slow start implications

- In early days of the Internet packet loss usually meant a packet corruption due to bad links quality, the obvious remedy was to retransmit
- Modern networks are more reliable and packet loss usually means a congestion at the routers. The old “remedy” of retransmission actually worsens the problem
- Modern TCP implementations would severely drop their transmission rates when they detect a packet loss assuming a congestion occurred
- This makes correct tuning of buffers even more important to networking performance

TCP-specific buffers (window)

- TCP automatically adjusts the size of its *sliding window* with `net.ipv4.tcp_window_scaling`
- `TCP buffers controlled by:`
  - Overall TCP memory in pages: `net.ipv4.tcp_mem`
  - Input/reader in Bytes: `net.ipv4.tcp_rmem`
  - Output/writer in Bytes: `net.ipv4.tcp_wmem`

Fragmentation and re-assembly

- Most payloads of PDUs exceed the MTU of the underlying physical network
- Path MTU (lowest MTU across the path links) can be “discovered”
- NFS for instance transmits at least 8 KB packets which are larger than the 1500-bytes Ethernet MTU
- Fragmentation buffers adjusted using
  - Minimum size: `net.ipv4.ipfrag_low_thresh`
  - Maximum size: `net.ipv4.ipfrag_high_thresh`
  - Expiration time for fragments: `net.ipv4.ipfrag_time`

Fragmentation/re-assembly statistics

- Summary statistics

```
# netstat -s
```

- Reassembly failures

```
# cat /proc/net/snmp | grep '^Ip:' | cut -f17
-d' '
```

- Reassembly failures indicate a need to tune buffers
- NFS and Denial of Service attacks are common causes of high reassembly counts

Autonegotiation

- NIC and switch negotiate duplex and speed using built-in logic
- A correctly operating full duplex mode should result in 0 collisions
- Some devices do not implement autonegotiation properly. May need to disable autonegotiation
- Duplex and speed setting can be forced manually using ethtool

TCP segmentation offloading

- TCP segmentation and reassembly operations are quite expensive
- Some NICs implement the TCP reassembly logic in hardware
- Check status with:

```
# ethtool -k eth0
```

- Enable with:

```
# ethtool -K eth0 tso on
```

- The same applies to receive and transmit checksumming

Lab 2

Observing the effects of buffer sizing

Watch the performance of NFS clients