TI III: Operating Systems & Computer Networks
Example

Prof. Dr.-Ing. Jochen Schiller
Computer Systems & Telematics
Freie Universität Berlin, Germany
Content

8. Networked Computer & Internet

9. Host-to-Network

10. Internetworking

11. Transport Layer

12. Applications

13. Network Security

14. Example
A Comprehensive Example

What happens if one presses a key on the computer?

What if that key causes an web page to be displayed?
Keyboard Interrupt

Keyboard controller raises interrupt flag
CPU interrupts execution of current process and starts Interrupt Service Routine (ISR)
- Unconditional jump
Keyboard Interrupt Handling

**Hardware**
- Device controller or other system hardware issues an interrupt
- Processor finishes execution of current instruction
- Processor signals acknowledgment of interrupt
- Processor pushes PSW and PC onto control stack
- Processor loads new PC value based on interrupt

**Software**
- Save remainder of process state information
- Process interrupt
- Restore process state information
- Restore old PSW and PC

ISR processes input from keyboard
- Clears interrupt flag
- Transfers data from device into buffer
- Establishes owner of device
- Triggers notification of user process
In the Meantime…

Web browser is one of many processes running locally

Other processes include
- Other user processes (possibly of different users)
- System processes implementing system services
- Kernel processes
Web Browser Process in Detail

Web browser processes
- Currently waiting for input
  - E.g. using `select()`
- Process state blocked
Reaction to External Event

1. ISR changes process state to **ready**
2. Scheduling algorithm eventually changes process state to **running**
Process Scheduling

Scheduling is handled by a variety of scheduling algorithms:
- Non-preemptive / preemptive
- Maximize throughput, responsiveness, etc...

Processes may have priorities:
- Priority inversion due to lock on shared resources
- Priority inheritance
Web Browser Processes Event

Assume input requires web browser to display a web page with a given URL
1. String processing (user space)
2. Connect to server and retrieve necessary data (system calls)
3. Render web page (user space)
4. Update user interface (system calls)
Client/Server Communication

GET /inst/ag-tech/index.html HTTP/1.1
Host: www.inf.fu-berlin.de
Pragma: no-cache
....
Layered Protocol Stack

Web Browser

Host A

App X

TCP

IP

Network Access Protocol #1

Router J

Network 1

Network 2

App Y

Web Server

Host B

App Y

TCP

IP

Network Access Protocol #2

Logical connection (TCP connection)

Global internet address

Logical connection (e.g., virtual circuit)

Subnetwork attachment point address

Port

Web Browser corresponding to URL
Interaction Between Network Layers

Layered protocol architecture
- Each layer uses only services of layer directly below
- Each layer provides services to layer directly above
  - Protocol independence
  - Modularity

Data encapsulation
- Lower layers treat upper layer packets as simple data
- Headers contain control information for each layer
  - Repeated encapsulation causes overhead
Uniform Resource Locator (URL)

http://cst.mi.fu-berlin.de/index.html

http: Hypertext Transfer Protocol (HTTP)
- Protocol for accessing web pages and related content
- Implies communication over port 80 (unless other port given in URL)
cst.mi.fu-berlin.de: Host name
- Resolved to IP address via Domain Name System (DNS)
  - cst.mi.fu-berlin.de -> 160.45.117.167
index.html: Local resource name
- Protocol specific parameter
- Handled by web server
Security: HTTP over TLS/SSL

HTTPS authenticates server and establishes secure connection:

1) Propose SSL parameters, send random number
2) Agree to parameters, send random number
3) Send public key certificate
4) Conclude handshake negotiation
5) Send random number encrypted with server’s public key
   - Client and server derive session key from all three random numbers
6) Activate negotiated parameters
7) Send encrypted hash over previous messages
   - Server decrypts and verifies message
8) Activate negotiated parameters
9) Send encrypted hash over previous messages
   - Client decrypts and verifies message

➢ Proceed to exchange regular HTTP data over secure channel
Connection Setup / Transport Layer

Reliable end-to-end connection between processes
Call to `connect()` initiates connection setup
- TCPT 3-way handshake
- Connection parameters

Client
- [SYN, seq=17]
- [SYN, seq=39, ACK=18]
- [seq=18, ACK=40]
- [seq=53, ACK=78, data=’hi’]
- [seq=78, ACK=55, data=’ho’]
- [FIN]
- [ACK]
- [FIN]
- [ACK]
- Connection release

Server
- Connection setup
- Data transfer
- Time wait

Network States
- CLOSED
- LISTEN
- SYN_RCVD
- SYN_SENT
- ESTABLISHED
- FIN_WAIT_1
- FIN_WAIT_2
- CLOSING
- LAST_ACK
- TIME_WAIT
- CLOSED
Structure of Network Layer IP-Packet

- **Version**: 4 bits
- **Hdr.Len**: 4 bits
- **DiffServ**: 7 bits
- **Total Length**: 16 bits
- **Identifier**: 16 bits
- **Flags**: 3 bits
- **Fragment Offset**: 13 bits
- **Protocol**: 8 bits
- **Time to Live**: 8 bits
- **Header Checksum**: 16 bits
- **Source Address**: 32 bits
- **Destination Address**: 32 bits
- **Options and Padding**: Variable length
- **Data**: Variable length

**Congestion control (Explicit Congestion Notification)**

- **ECN**: 2 bits
- **Reserved**: 4 bits
- **Don't Fragment (DF)**: 1 bit
- **More Fragments (MF)**: 1 bit

**QoS class**
Network Layer Routing (Local Scope)

Globally unique per host addressing
Routers maintain tables of known networks
- Optional route to default gateway
Subnetting implements logical structure
- Subnet mask builds hierarchy using host part of IP address
- Limits broadcasts
- More efficient routing
Network topology may be part of security concept
Network Layer Routing (Global Scope)

Internet organized into autonomous systems (AS)
- Commonly, one AS per major organization
- Peering points to exchange data between ASs

Intra-domain routing: OSPF, link state algorithm
Inter-domain routing: BGPv4, distance vector protocol
- May involve non-technical routing choices
Data Link Layer Communication (Local Scope)

Transparent communication between two directly connected nodes
Services include: framing, error control, connection maintenance, acknowledgements, flow control

TI 3: Operating Systems and Computer Networks

Version | Hdr.Len | DiffServ | Total Length
Identifier | Flags | Fragment Offset
Time to Live | Protocol | Header Checksum
Source Address | Destination Address | Options and Padding
Data

Switch or hub

Ethernet

Congestion control (Explicit Congestion Notification) Don’t Fragment Reserved More Fragments

QoS class

ECN

Reserved

Don’t Fragment

More Fragments

DiffServ Codepoint

0 3 7 15 31

IP Header

FLAG Header Payload field Trailer FLAG
Error Detection: Cyclic Redundancy Check (CRC)

- Reception of a correct bit sequence:
  \[ 11 0011 1001 \div 1 1001 = 10 0001 \pmod{2} \]
  \[
  \begin{array}{c|c c c c}
    & 1 & 1 & 0 & 0 \\
  \hline
  1 & 1 & 0 & 0 & 1 \\
  0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\
  & 1 & 1 & 0 & 0 & 1 \\
  & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\
  & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\
  \end{array}
  \]
  \[ = \text{remainder} \]

  - No remainder, thus the received bits should be error free

- Reception of an erroneous bit sequence:
  \[ 11 1111 1000 \div 1 1001 = 10 1001 \pmod{2} \]
  \[
  \begin{array}{c|c c c c}
    & 1 & 1 & 1 & 1 \\
  \hline
  1 & 1 & 0 & 0 & 1 \\
  0 & 0 & 1 & 1 & 0 & 0 & 1 \\
  & 1 & 1 & 0 & 0 & 1 \\
  & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\
  & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\
  \end{array}
  \]
  \[ = \text{remainder} \neq 0 \]

  - There is a remainder unequal 0, thus there was definitely a transmission error
Errors During Transmission

Data: 0 1 0 1 1 0 0 1 1 0 0 1 0 1 0

Signal:

Interference:

Signal with interference:

Sampling:

Received data: 0 1 0 1 1 0 1 1 1 0 0 0 0 1

Original data: 0 1 0 1 1 0 0 1 1 0 0 1 0 1

Error!
Physical Layer

Packet / sequence of bits turned into physical signal

Signal propagation depends on physical medium (limited bandwidth, attenuation, dispersion) and background noise

Mapping between bits and (multi-valued) symbols

Baseband transmission vs. modulation (broadband transmission)
Client/Server Communication

GET /inst/ag-tech/index.html HTTP/1.1
Host: www.inf.fu-berlin.de
Pragma: no-cache
....

LAN or WAN or Internet

workstation (client)

server
At the Server…

Web server is one of many processes running locally

- Upon receiving packet, network interface controller (NIC) will raise interrupt
- Kernel will handle the packet and notify the web server process
Processing of HTTP-GET Request

Web server retrieves file `inst/ag-tech/index.html` from local file system
- System calls to access secondary storage
- Kernel maps file name to data layout on disk

Web server sends data to client
Server Replies to Client

HTTP/1.1 200 OK
Date: Fri, 16 Feb 2007 11:40:34 GMT
Server: Apache/1.3.6 (Unix)
Transfer-Encoding: chunked
Content-Type: text/html

<Document according to HTML>
</HTML>
Client Data Processing

Client host receives packet
Kernel hands data to web browser process
Web browser renders page
  ➢ May have to allocate memory in the process
Finally, browser updates user interface via system call
A Comprehensive Example
Content

1. Introduction and Motivation
2. Subsystems, Interrupts and System Calls
3. Processes
4. Memory
5. Scheduling
6. I/O and File System
7. Booting, Services, and Security
8. Networked Computer & Internet
9. Host-to-Network
10. Internetworking
11. Transport Layer
12. Applications
13. Network Security
14. Example
Figure 3.17 UNIX Process State Transition Diagram