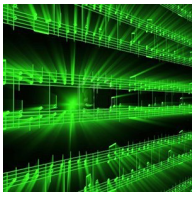
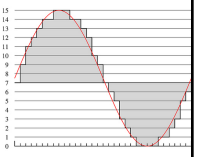
Penn Engineering

ESE





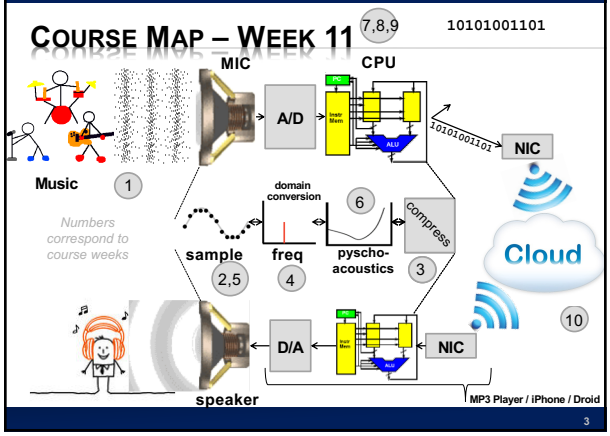
Lecture #20 – Networking

ESE 150 – Digital Audio Basics

Lecture Topics

- **Where are we on course map?**
- **Review**
- **Networks**
  - Network Layering
    - Transport
    - Network – Routing – what can go wrong?
    - Physical (physical layer independence)
    - By end: seen TCP/IP basics
- **Next Lab**

Course Map – Week 11



Review

Fundamentals of Networks

Networked Systems

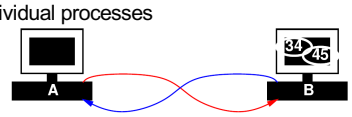
- **Today**
  - We expect our computers to be networked
    - Google, wikipedia, Email, IM, …
  - Can work stand alone
    - Airplane mode?
  - But, are crippled when not connected
  - Phone isn't a phone unless its networked

Implications?

- **Today's wire bandwidth exceeds the throughput needs of any real-time single-stream data**
  - Can afford to share the wire
- **Wires are not cheap**
  - Cannot afford not to share the wire

Sharing Link

- **Idea: Tag data with target**
  - "this is for process 34"
  - "this is for process 45"
- **Have transport layer deal with…**
  - Mixing data from separate streams
  - Separating data out into individual streams
  - Delivering to individual processes



34: and then she said...

45: 80004010 00001200

7

Packet

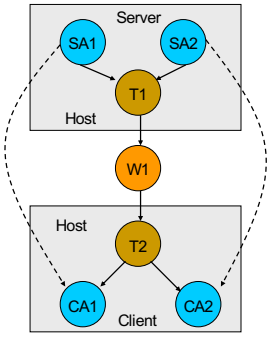
- **Begin to form a packet**
  - Header: says where to go
  - Payload: the data to send
- **Header:**
  - Added, consumed by network handling in routing
- **Payload:**
  - Only thing seen by the application processes

and then she said...	34

8

Simulation 1

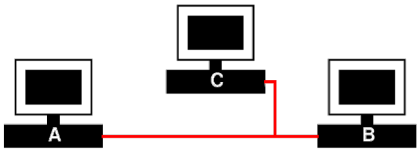
- **Send 4 verses or digits from each**
  - from song-server-app, even-server-app
  - to song-listener-app, even-consumer
- All go through one wire W1
- T1 – Transport tagging
- T2 – Transport sorting



9

Start Simple

- **Add more computers to same pair of wires**



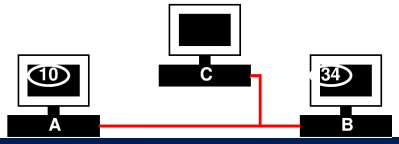
- All computers on wire see all the data on the wire
  - *How do computers know who the message is for?*

10

Extended Packet

- **Extend our packet header:**
  - Destination computer
  - Process on destination computer
  - Sending computer
  - Process on sending computer

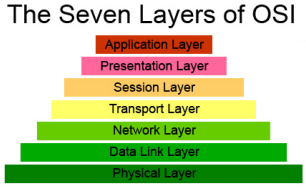
and then she said...	10	A	34	B



11

Network Layer

- **responsible for end-to-end (source to destination) packet delivery**



12

Virtualization Effect

- **Each pair of processes on different computers**
  - Has the view of a point-to-point connection
  - Each process, thinks it "owns the network" and has a dedicated connection to the other node

13

Abstractions for Sharing

- **Virtualized Processor**
  - Share single processor among multiple tasks
  - Make it look like process (program) has its own processor
- **Virtualized Communication between programs**
  - Share wires and processors
  - Make it look like a dedicated point-to-point link between processes (programs)

14

Network Simulation

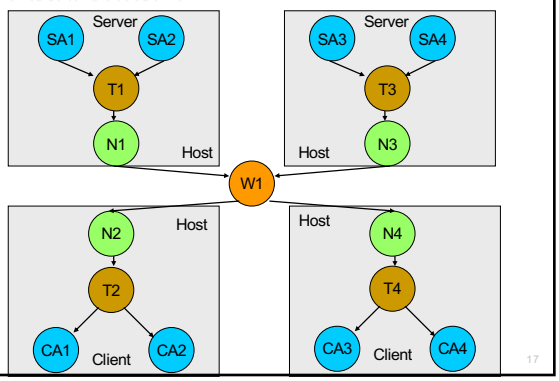
15

Simulation 2

- **Send 4 verses or digits from each**
  - from song-server serving 2 songs
  - And digit-server serving 2 fundamental constants
  - To two clients

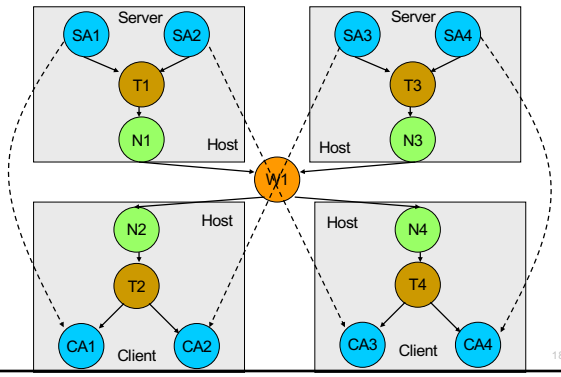
16

Simulation 2



17

Simulation 2



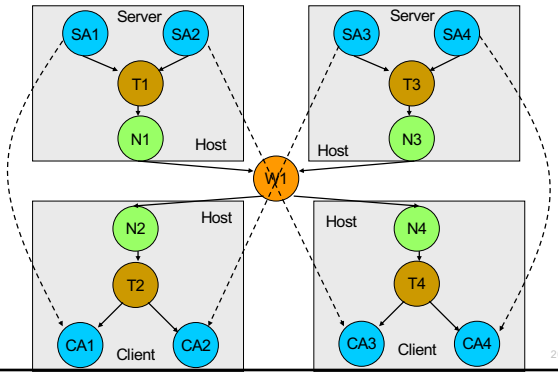
18

Simulation 2

- **N1, N3**
  - Add network-layer source/destination packet headers
- **W1 – Wire**
  - Duplicate packets to both destinations
  - Simulate shared wire
- **N2, N4**
  - Look at network-layer source/destination header
  - Discard packets not destined for this computer

19

Simulation 2



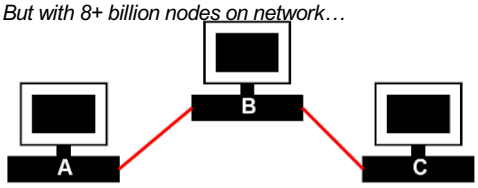
20

Extending the Virtual Link

21

Indirect Connections

- **A and B are connected**
- **B and C are connected**
- **How get message from A to C?**
  - *We could add a wire between A and C…*
  - *But with 8+ billion nodes on network…*

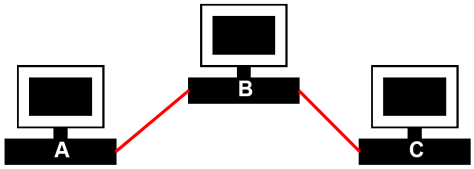


22

Indirect Connections

- **Run program/process on B to forward messages from A to C**
  - Call it a "routing" program! Routes messages on network

and then she said...	10	A	34	C



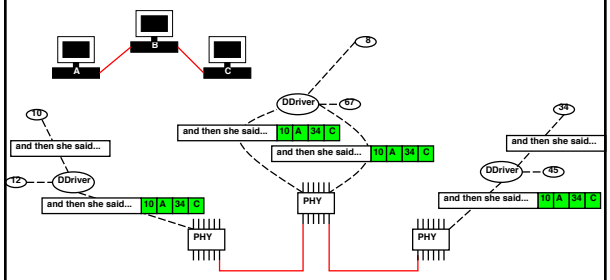
23

Routing

- **B runs a general program**
  - If packet destined for B, takes it
  - Otherwise, sends on to (toward) destination
- **Extension of the network handling process that is sorting data to processes**

24

Routing



25

Reachability

- **If everyone plays along**
  - We can communicate with any computer reachable *transitively* from my computer
- **Don't need direct connections**

26

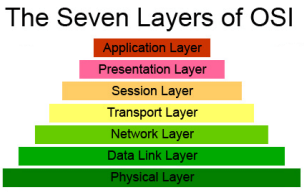
Routing → Route Tables

- **To make efficient**
  - Each computer should route *close* to destination
  - …and not route in circles
- **E.g. compute all-pairs shortest paths (CIS160,121)**
  - Store result, each machine knows where to send packet next
  - How much storage?
    - Cleverness to compress/summarize
  - Additional cleverness to compute incremental updates
    - When add a computer or a link breaks

27

Network Layer

- **Responsible for end-to-end packet delivery**
  - Source to Destination
  - This includes routing packets through intermediate hosts

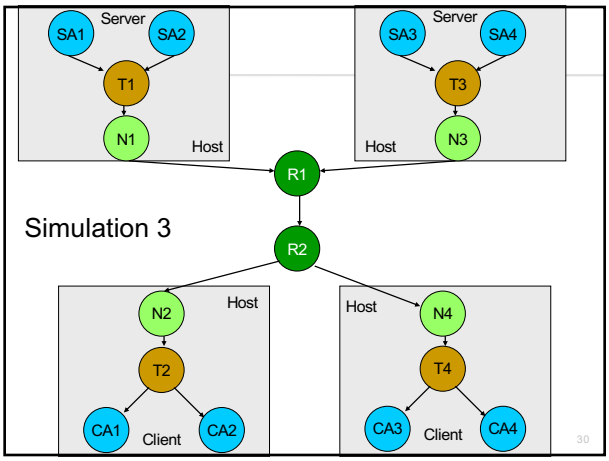


28

Simulation 3

- **Send 4 verses or digits from each**
  - from letter-server serving 2 strings
  - And digit-server serving 2 fundamental constants
  - To two clients
- **R1** – pass along packets to R2 (for now)
- **R2** – look at address and send to N2 or N4

29

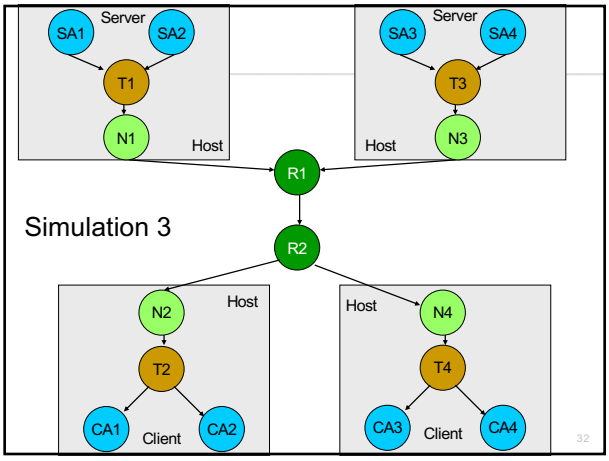


30

Simulation 3 Simplification

- T1/N1, T3/N3 same as before
- Same set of inputs to R1 as we had for W1
  - So we will start with those.
- T2, T4 gets same inputs
  - So, won't rerun that part.
- Focus on R1, R2, N2, N4

31



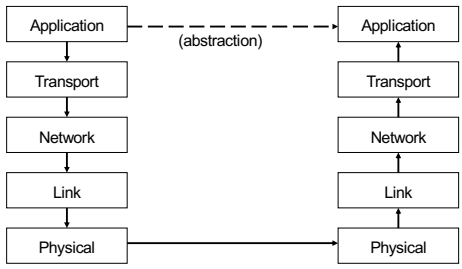
32

Where are we now?

- Can communicate
  - From one process on a computer
  - to any other process on any other computer
  - *if* the two are transitively connected
    - By a set of participating computers which route data
- Layers have provided "Abstraction"
  - Processes just see streams of data between the endpoints

35

Layers



36

Protocols

- Protocol – common discipline used to interoperate smoothly
  - rules of the game
  - Include
    - How to format packets
    - How to handle data
- So far, we've discussed a protocol called IP:
  - IP = Internet Protocol
- Delivery to processes (rather than hosts): UDP
  - UDP = Unreliable Datagram Protocol

37

Simulation 5

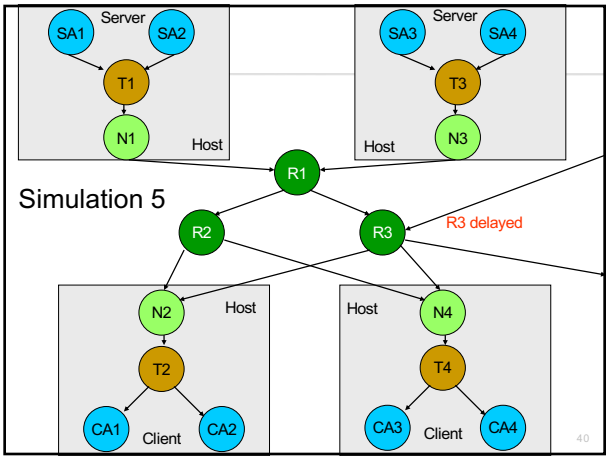
- Send 4 verses or digits from each
  - from letter-server serving 2 strings
  - And digit-server serving 2 fundamental constants
  - To two clients
- Deliberately delay data through R3
  - Model non-determinism in route timing

38

Simplify Simulation

- **Again, simulate different core**
  - R1, R2, R3

39



40

What can go wrong?

- **Packets arrive out of order**
- **Solution?**
  - Add a sequence number

I was born,	2	10	A	34	C

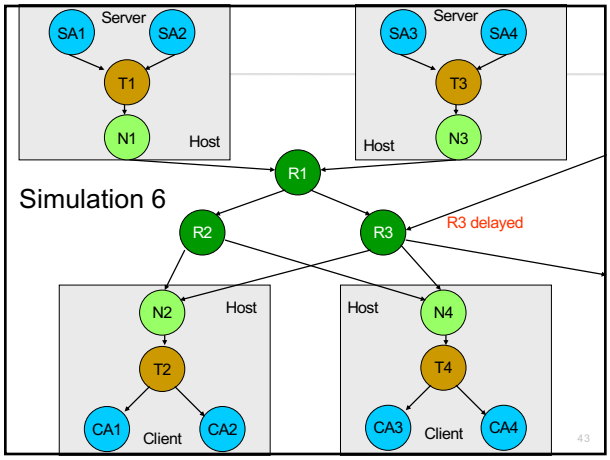
In the town where	1	10	A	34	C

41

Simulation 6

- **Send 4 verses or digits from each**
  - from song-server serving 2 songs
  - And digit-server serving 2 fundamental constants
  - To two clients
- **T1/T3 –** add sequence number to packet
- **T2/T4 –** hold packets, reorder, and deliver in order of sequence number
- **R3** – still delaying packets

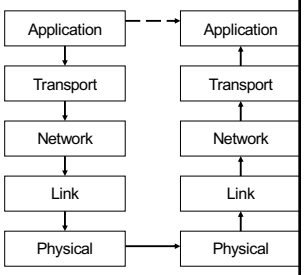
42



43

Abstracting Physical Layer

- **Application, transport, network**
  - Don't really care how the bits are moved from machine-to-machine
- **What are other ways we send bits?**
  - Beyond wires
  - Optically
  - RF/wireless
  - Pneumatic tubes, passing paper notes, SMS Text messages…

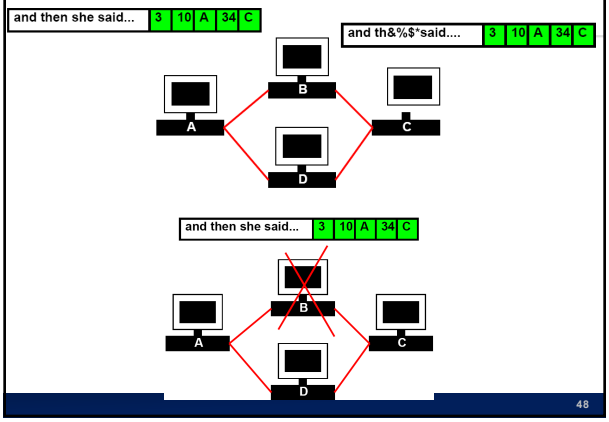


44

What else can go wrong?

- Bits get corrupted
- Intermediate machines holding messages can crash
- Messages can get misrouted

47



48

Data Corruption

- How do we deal with data corruption?
  - Use redundancy
- Two strategies:
  - Use enough redundancy to correct
  - Use just enough redundancy to detect it
    - Have the sender resend

49

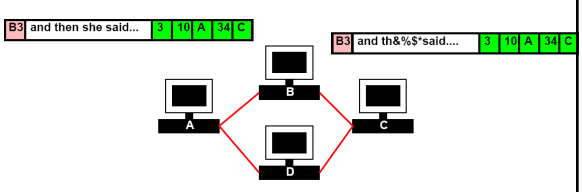
Data Corruption

- Relatively uncommon
  - Most packets are fine
- We have efficient (low overhead) ways to detect
  - Compute a hash of the message data
  - Highly unlikely one (few) message bit errors will result in same hash
  - → checksum

50

Revised Packet

- Header
- Data payload
- Checksum



51

Lost Packet

- How can we deal with lost packets?

52

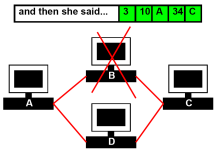
Lost Packet Strategy

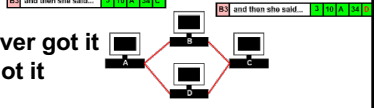
- **Sender sends packet**
  - But keeps a copy
- **Receiver gets packet**
  - Checks checksum
  - OK, uses packet and sends ACK
    - "got your last packet intact"
  - Not ok, discard packet
- **Sender**
  - Receives ACK, can discard packet and send next
  - No ACK (after timeout), resend packet

53

Retransmission Discipline

- **Don't depend on receiver to request retransmission**
  - Why?
- **Header may be corrupted**
  - Not deliver to receiver





- **Only know receiver got it when it says it got it**

54

Corrupted ACK

- **What if the ack is lost?**
  - Sender resends
- **Receiver receives a second copy**
  - Oops, don't want that to be interpreted as new data
  - i.e. send: "rm *; cd ..\n"
    - Receive: "rm *; cd ..\n rm *; cd ..\n"

55

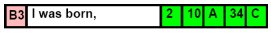
Avoid Duplication

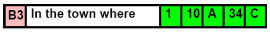
- **How can we avoid duplication?**

56

Accommodating Duplication

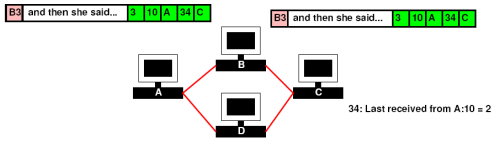
- **Use packet sequence number**
  - Keep track of last packet received
  - If receive packet again,
    - Discard the packet

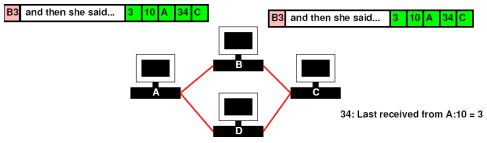




57

Sequence Numbers





58

TCP

- **TCP = Transmission Control Protocol**
  - Provides Reliable delivery
  - Deals with
    - Retransmission
    - Duplication
    - Out of sequence / resequence / reconstruction

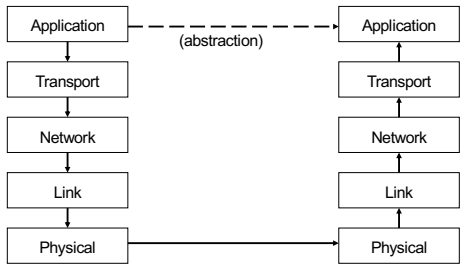
59

TRANSPORT LAYER

- **Call this the "Transport" Layer**
  - responsible for reliably delivering data to the individual application process on the computer

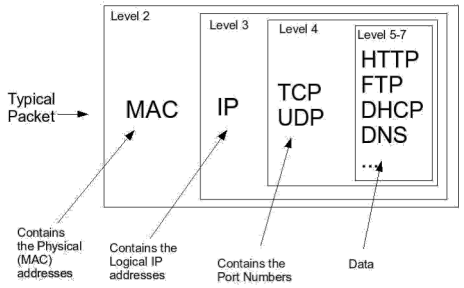
60

LAYERS



61

LAYERS AND THE PACKET



62

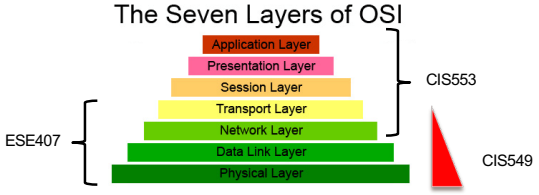
BIG IDEAS

- **Sharing – Network interface, wires**
  - Previously gates, processor, memory
- **Virtualization – datastream abstracts physical point-to-point link**
- **Layering**
  - Divide-and-conquer functionality
  - Implementation hiding/technology independence
  - Reliable communication link from unreliable elements

63

LEARN MORE @ PENN

- **Courses**
  - ESE407 – Intro Networks and Protocols
  - CIS553 – Networked Systems
  - CIS549 – Wireless Mobile Communications



64

REMEMBER

- **Feedback**
- **Lab 10 due Friday**

65