CS549 Distributed Systems & Cloud Computing

Dominic Duggan
Stevens Institute of Technology

1

THIS COURSE

2

Graduate Certificate / MS in Enterprise and Cloud Computing

- CS522 Mobile Systems and Apps
- CS526 Enterprise and Cloud Computing
- CS548 Enterprise Software Architecture
- **CS549 Distributed Systems and Cloud Computing**
- CS594 Enterprise and Cloud Security

MS/ECC Program Outcomes

	CS522	CS526	CS548	CS549	CS594
Infrastructure		X Virtualization		X Availability	
Data Modeling			X		
Design			X		
Applications	X	X	X	X	
Security & Privacy		X Secure Virt			X

Web Apps & Web Services

	Azure (CS526)	Java (CS548, CS549)
Client-Side B2C	AJAX Silverlight	AJAX Android
Server-Side B2C	ASP.NET	Java Faces
Server-Side B2B	Windows Communication Foundation (WCF)	**Java EE (CDI, JCA)** **Jersey** **Atmosphere**
Database	Entity Framework LINQ	Java Persistence Architecture (JPA)

Cloud Computing

Dominic Duggan
Stevens Institute of Technology
Based on material by K. Birman, P. Francis, A.M. Nguyen, A. Tanenbaum

What is Cloud Computing?

- *I don't understand what we would do differently in the light of Cloud Computing other than change the wordings of some of our ads*

  Larry Ellision, Oracle's CEO

- *I have not heard two people say the same thing about it [cloud]. There are multiple definitions out there of "the cloud"*

  Andy Isherwood, HP's Vice President of European Software Sales

- *It's stupidity. It's worse than stupidity: it's a marketing hype campaign.*

  Richard Stallman, Free Software Foundation founder

7

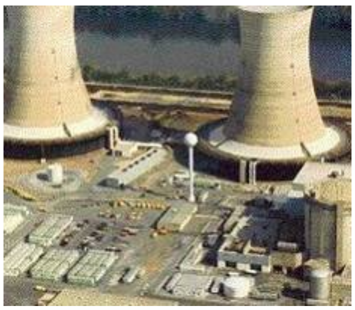
SOFTWARE AS A SERVICE AND UTILITY COMPUTING

8

Software as a Service (SaaS)

Ex: SalesForce.com

Traditional Software | **On-Demand Utility**



Build Your Own

Plug In, Subscribe Pay-per-Use

9

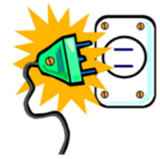
Software as a Service (SaaS)

- Application used as on demand service
  - Often provided via the Internet
- Example: Google Apps

- Benefits to users
  - Reduce expenses: multiple computers, multiple users
  - Ease of usage: easy installation, access everywhere
- Benefits to providers
  - Easier to maintain
  - Control usage (no illegal copies)

10

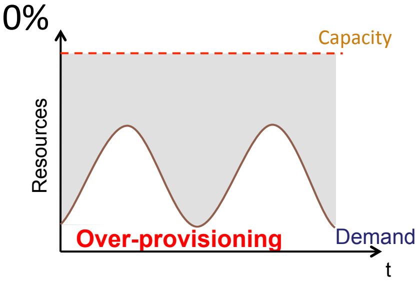
Utility Computing (UC)

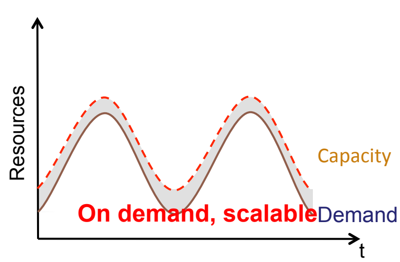


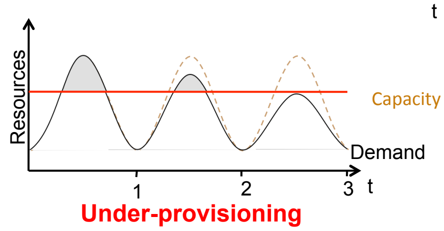
- Computing resources (cpu hours, memory, network) and platform to run software are provided as on demand service
  - *Hardware as a service (HaaS)*
  - *Infrastructure as a service (IaaS)*
  - *Platform as a Service (PaaS)*
- Examples of UC providers:
  - PaaS: MS Azure …
  - IaaS: Amazon EC2 …
- Who will use UC?

Utility Computing: Mitigate Risks

- Real world utilization 5%-20%
- Animoto demand surge: from 50 servers to 3500 servers in 3 days
- Black Friday sales







Utility Computing – Amazon EC2

- **Elastic Compute Cloud**
- Rent VM instances to run your software
- Full root-level access to VM

13

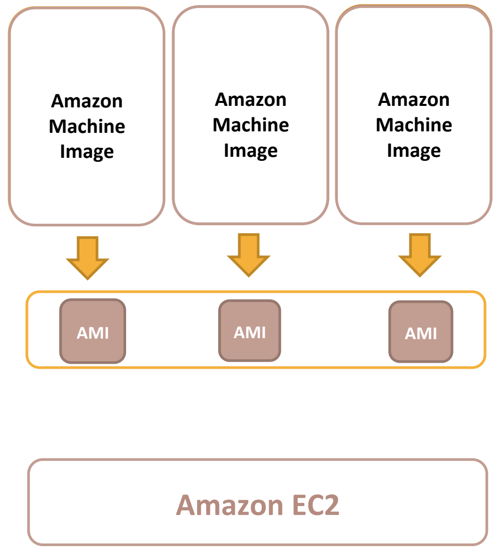
Amazon EC2

- Create an Amazon Machine Image (AMI)
- Upload AMI to Amazon S3 (simple storage service)
- Use Amazon EC2 web service to configure
- Choose OS, start AMI instances

14

Amazon EC2

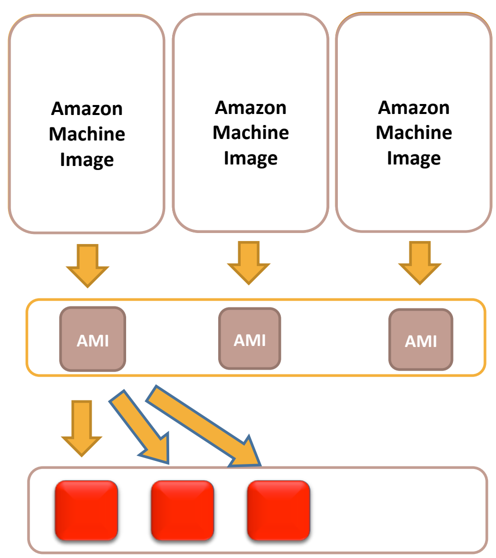
- Create an Amazon Machine Image (AMI)
- Upload AMI to Amazon S3 (simple storage service)
- Use Amazon EC2 web service to configure
- Choose OS, start AMI instances



15

Amazon EC2

- Create an Amazon Machine Image (AMI)
- Upload AMI to Amazon S3 (simple storage service)
- Use Amazon EC2 web service to configure
- Choose OS, start AMI instances



16

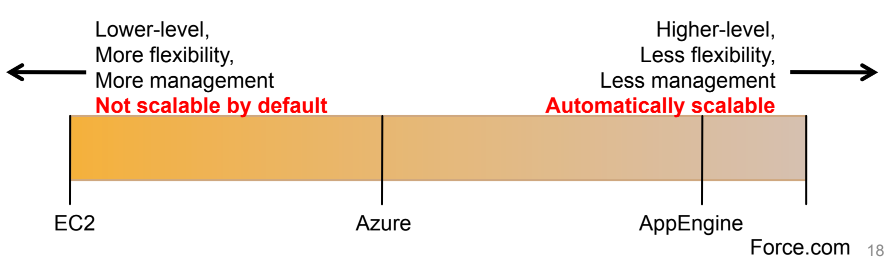
Utility Computing – MS Azure

- Write your web program and submit to Azure
- How to use
  - Download MS SDK, Azure tools
  - Develop your program locally
  - Register for an Azure id
  - Launch your application in Azure

17

Spectrum Of Abstractions

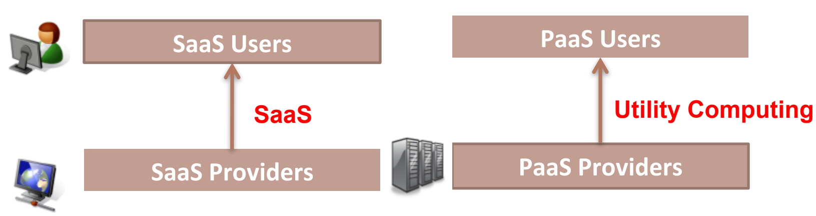
- Different levels of abstraction
  - Instruction Set VM: Amazon EC2
  - Framework VM: MS Azure
- Similar to languages
  - Higher level abstractions can be built on top of lower ones



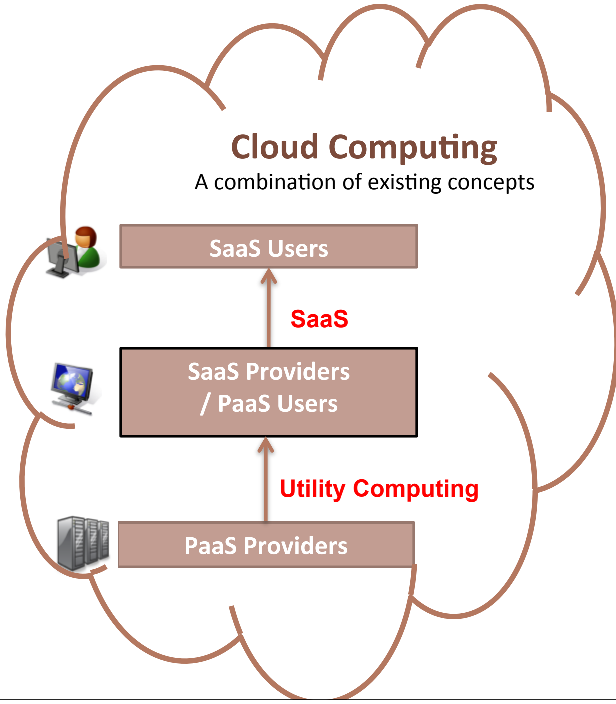
18

CLOUD COMPUTING

19



20



21

Cloud Computing

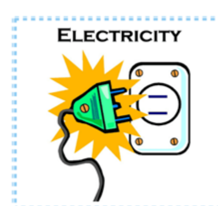
Cloud Computing = SaaS + PaaS (utility computing)

Cloud TV
(Cloud Computing)

=

Video On Demand
(SaaS)

+



Electricity On Demand
(PaaS)

22

Significance of Cloud Computing

- The illusion of infinite computing resources
- The elimination of an up-front commitment by users
- The ability to use and pay on demand

- Cloud Computing vs P2P?
  - Both take advantage of remote resources
  - P2P: does not use clouds (datacenters), peers do not get paid, lower reliability
- Cloud Computing vs Grid Computing?
  - Both use clouds
  - Grid Computing requires commitment, share based on common interests. Not public cloud

23

Cloud Killer Apps

- Mobile and web applications
  - Mobile devices: low memory & computation power
- Extensions of desktop software
  - Matlab, Mathematica
- Batch processing / MapReduce
  - Peter Harkins at The Washington Post: 200 EC2 instances (1,407 server hours), convert 17,481 pages of Hillary Clinton's travel documents within 9 hours
  - The New York Times used 100 Amazon EC2 instances + Hadoop application to recognize 4TB of raw TIFF image into 1.1 million PDFs in 24 hours ($240)

24

ECONOMICS OF THE CLOUD

25

Should I Move Into A Cloud?

- Does it really save money?

$$\text{UserHours}_{\text{cloud}} \times (\text{revenue} - \text{Cost}_{\text{cloud}}) \geq \text{UserHours}_{\text{datacenter}} \times \left(\text{revenue} - \frac{\text{Cost}_{\text{datacenter}}}{\text{Utilization}}\right)$$

  - *$Cost_{cloud}$ > $Cost_{datacenter}$*, balance by *Utilization*
  - *$UserHours_{cloud}$ > $UserHours_{datacenter}$* (under-provisioning)
- Other factors
  - Re-implement programs
  - Move data into cloud
  - What else?
- Example:
  - Upload rate 20Mbits / s. 500GB takes 55 hours
  - If can process locally in less than 55 hours → moving into a cloud would not save time

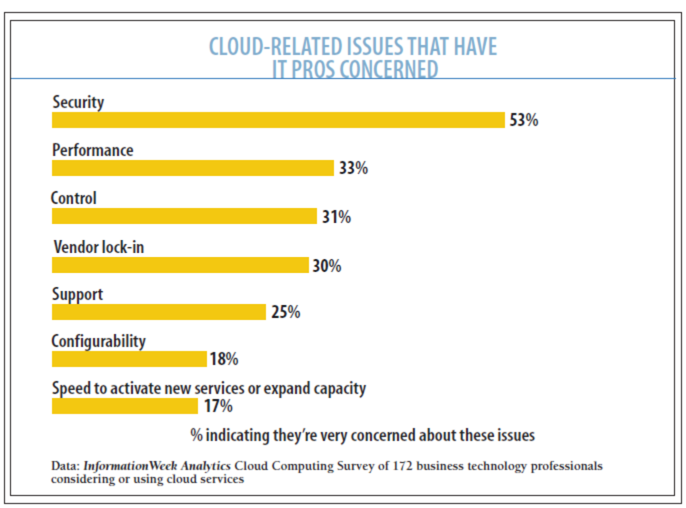
26

Adoption Challenges

Challenge	Opportunity
Availability	Multiple providers
Data lock-in	Standardization
Data Confidentiality and Auditability	Encryption, VLANs, Firewalls

- Coghead, a cloud vendor closed its business in February 2009
  - Customers need to rewrite their applications
  - Another company will automatically convert customer data to their proprietary formats…
- Online storage service The Linkup closed July 10, 2008
  - 20,000 paying subscribers lost their data

Adoption Challenges



Cloud Control, InformationWeek Reports, 2009

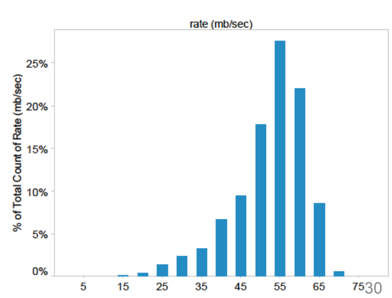
Growth Challenges

Challenge	Opportunity
Data transfer bottlenecks	FedEx-ing disks, reuse data multiple times
Performance unpredictability	Improved VM support, flash memory
Scalable storage	Invent scalable storage
Bugs in large distributed systems	Invent Debugger using Distributed VMs
Scaling quickly	Invent Auto-Scaler

29

Growth Challenges

- Data transfer bottle neck
  - WAN cost reduces slowest: 2003 → 2008: WAN 2.7x, CPU 16x, storage 10x
  - Fastest way to transfer large data: send the disks
- Performance unpredictability
  - Large variation in I/O operations
  - Inefficiency in I/O virtualization



30

Policy And Business Challenge

Challenge	Opportunity
Reputation Fate Sharing	Offer reputation-guarding services like those for email
Software Licensing	Pay-for-use licenses; Bulk use sales

- Reputation: Many blacklists use IP addresses and IP ranges
- Software licensing:
  - Open source software readily applicable
  - Windows, IBM software offered per hour for EC2

31

ECONOMICS OF THE CLOUD

32

Should I Move Into A Cloud?

- Does it really save money?

$$\text{UserHours}_{\text{cloud}} \times (\text{revenue} - \text{Cost}_{\text{cloud}}) \geq \text{UserHours}_{\text{datacenter}} \times \left(\text{revenue} - \frac{\text{Cost}_{\text{datacenter}}}{\text{Utilization}}\right)$$

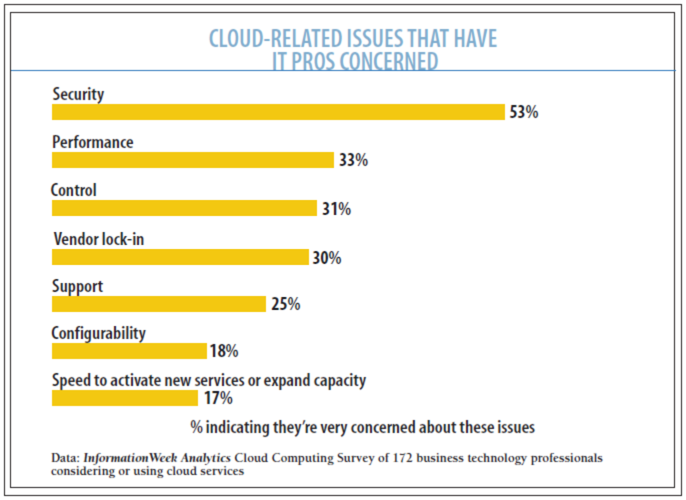
  - *$Cost_{cloud}$ > $Cost_{datacenter}$*, balance by *Utilization*
  - *$UserHours_{cloud}$ > $UserHours_{datacenter}$* (under-provisioning)
- Other factors
  - Re-implement programs
  - Move data into cloud
  - What else?
- Example:
  - Upload rate 20Mbits / s. 500GB takes 55 hours
  - If can process locally in less than 55 hours → moving into a cloud would not save time

Adoption Challenges

Challenge	Opportunity
Availability	Multiple providers
Data lock-in	Standardization
Data Confidentiality and Auditability	Encryption, VLANs, Firewalls

- Coghead, a cloud vendor closed its business in February 2009
  - Customers need to rewrite their applications
  - Another company will automatically convert customer data to their proprietary formats…
- Online storage service The Linkup closed July 10, 2008
  - 20,000 paying subscribers lost their data

Adoption Challenges



Cloud Control, InformationWeek Reports, 2009

35

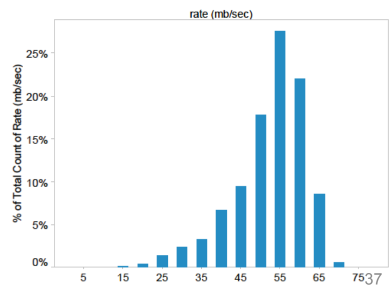
Growth Challenges

Challenge	Opportunity
Data transfer bottlenecks	FedEx-ing disks, reuse data multiple times
Performance unpredictability	Improved VM support, flash memory
Scalable storage	Invent scalable storage
Bugs in large distributed systems	Invent Debugger using Distributed VMs
Scaling quickly	Invent Auto-Scaler

36

Growth Challenges

- Data transfer bottle neck
  - WAN cost reduces slowest: 2003 → 2008: WAN 2.7x, CPU 16x, storage 10x
  - Fastest way to transfer large data: send the disks
- Performance unpredictability
  - Large variation in I/O operations
  - Inefficiency in I/O virtualization



37

Policy And Business Challenge

Challenge	Opportunity
Reputation Fate Sharing	Offer reputation-guarding services like those for email
Software Licensing	Pay-for-use licenses; Bulk use sales

- Reputation: Many blacklists use IP addresses and IP ranges
- Software licensing:
  - Open source software readily applicable
  - Windows, IBM software offered per hour for EC2

38

DISTRIBUTED SYSTEMS

39

Some terminology

- A program is the code you type in
- A process is what you get when you run it
- A message is used to communicate between processes.
- A packet is a fragment of a message that might travel on the wire.
  - Variable size
  - Limited size
- A protocol is an algorithm by which processes cooperate to do something using message exchanges.

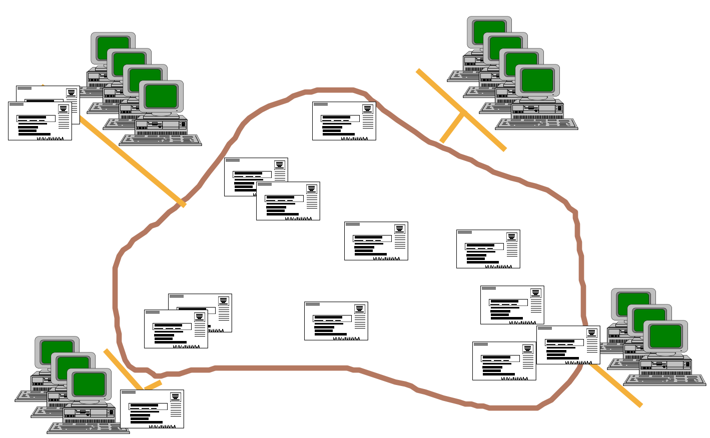
40

More terminology

- A network is the infrastructure that links the computers, etc.
  - routers
  - communication links
- Network application: fetches needed data from servers over the network
- Distributed system: multiple processes that cooperate to do something

41

A network is like a "mostly reliable" post office



42

Loss of reliability

- Links can corrupt messages
  - Internet "backbone"
  - Wireless connections, cable modems, ADSL
- Routers can get overloaded
- Solution: retransmission protocols

43

Distributed systems vs network applications

- Distributed systems
  - many components
  - often mimic a single, non-distributed process
- Networked application
  - centered around the user or computer where it runs

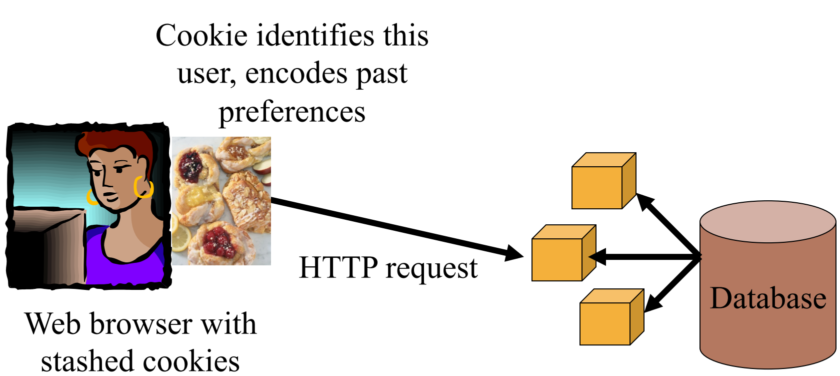
44

What about the Web?

- Browser is independent
- REST: Web servers don't keep track of clients.
  - Cookies
  - Database of account info
- Two network applications that talk to each other

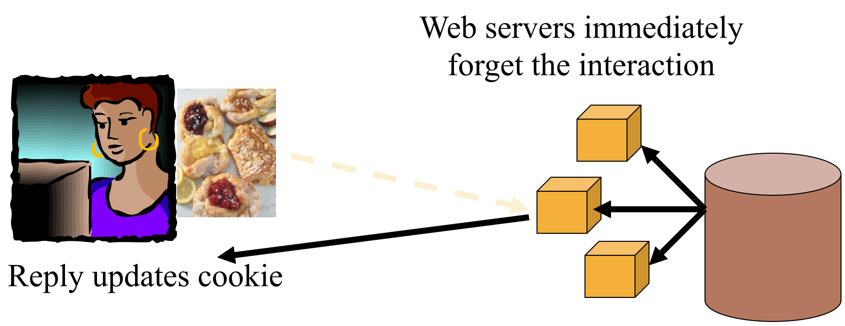
45

What about the Web?



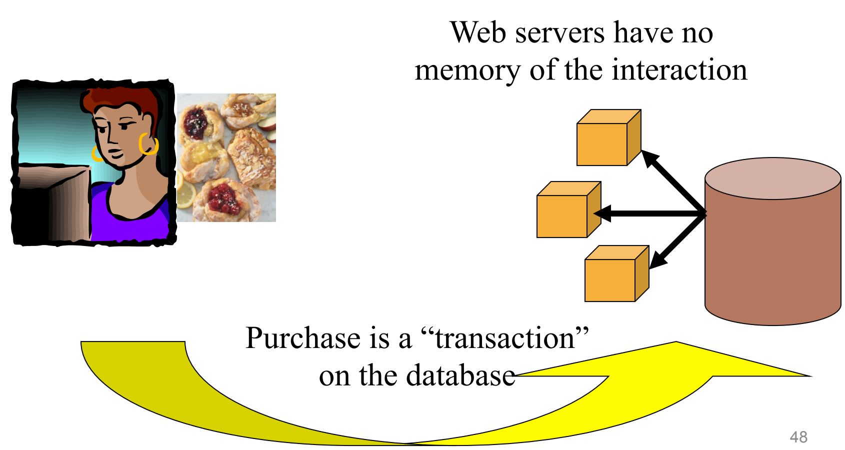
46

What about the Web?



47

What about the Web?



48

What about the Cloud?

- Data center or cloud is a complex distributed system
  - Many servers
  - Routing clients to servers
  - Data replicated
    - load balancing
    - high availability
  - Complex security and administration policies
- "**Network application**" talking to a "**distributed system**"

NETWORKS

Who recognizes this?

```
int sockfd;
struct sockaddr_in addr;

addr.sin_family = AF_INET;
addr.sin_addr.s_addr =
          inet_addr(SERV_HOST_ADDR);
addr.sin_port = htons(SERV_TCP_PORT);

sockfd = socket(AF_INET, SOCK_STREAM, 0);
connect(sockfd, (struct sockaddr *) &addr,
          sizeof(serv_addr));
do_stuff(stdin, sockfd);
```

51

Classic view of network API

- Start with host name (maybe)

foo.bar.com

52

Classic view of network API

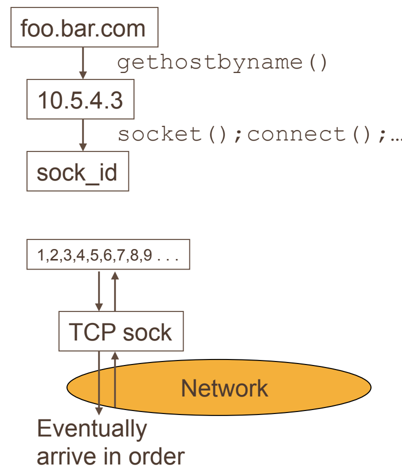
- Start with host name
- Get an IP address

Classic view of network API

- Start with host name
- Get an IP address
- Make a socket (protocol, address)

Classic view of network API

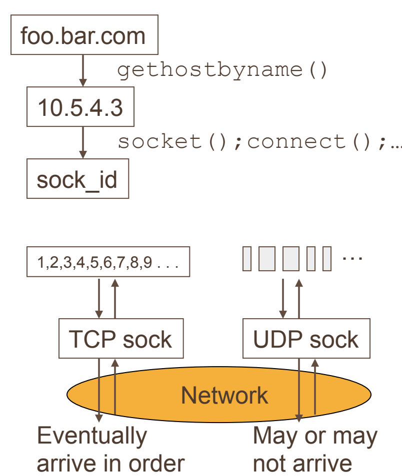
- Start with host name
- Get an IP address
- Make a socket (protocol, address)
- Send byte stream (TCP)



55

Classic view of network API

- Start with host name
- Get an IP address
- Make a socket (protocol, address)
- Send byte stream (TCP) or packets (UDP)

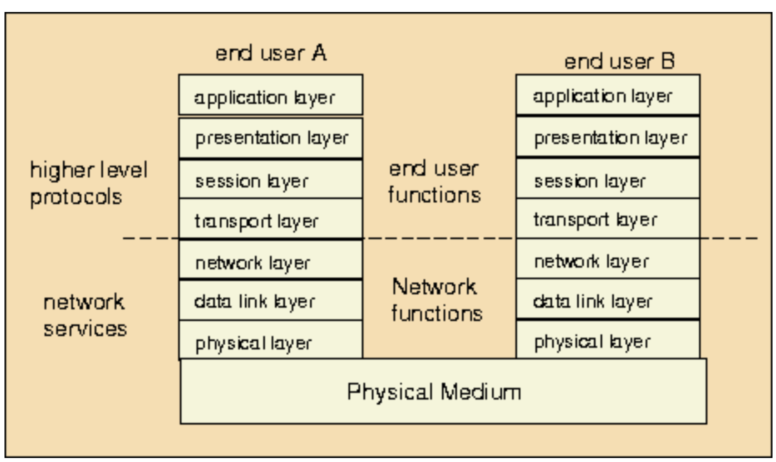


56

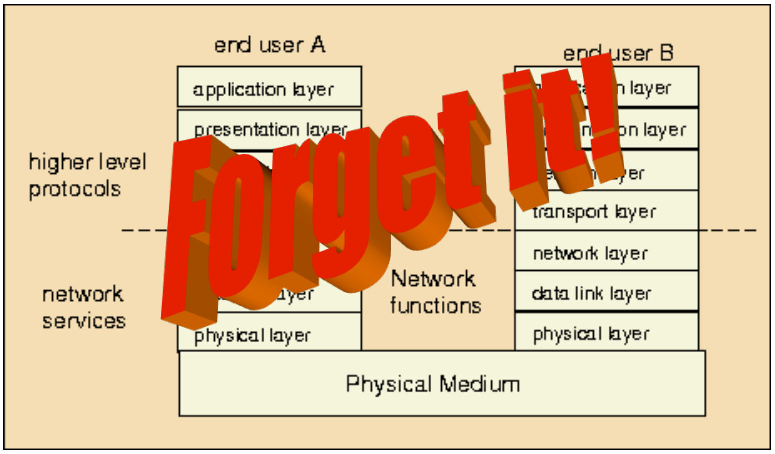
Classic approach "broken" in many ways

- **DNS:** IP address different depending on who asks
- **NAT:** Address may be changed in transit
- **Firewall:** IP address may not be reachable
  - Or may be reachable by you but not another host
- **DHCP:** IP address may change
- **Caches:** Packets may not come from who you think

Classic OSI stack



Classic OSI stack



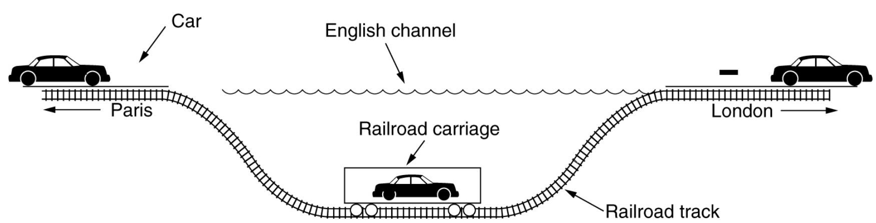
59

TUNNELING

60

Tunneling Analogy

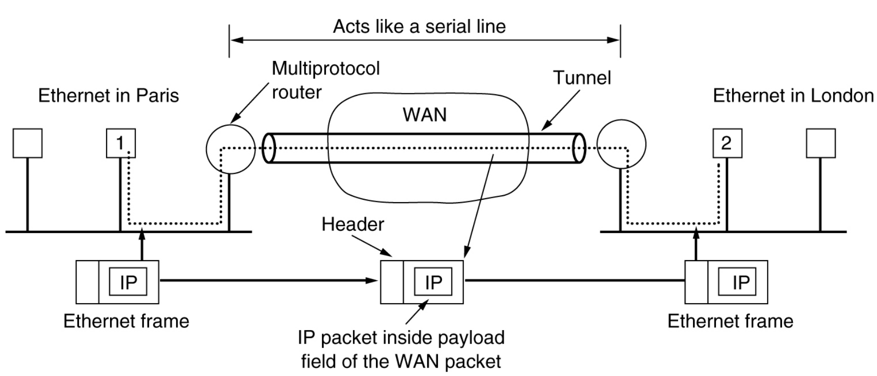
Tunneling a car from France to England.



61

Tunneling in Networks

Tunneling a packet from Paris to London.



62

Example Microsoft VPN stack

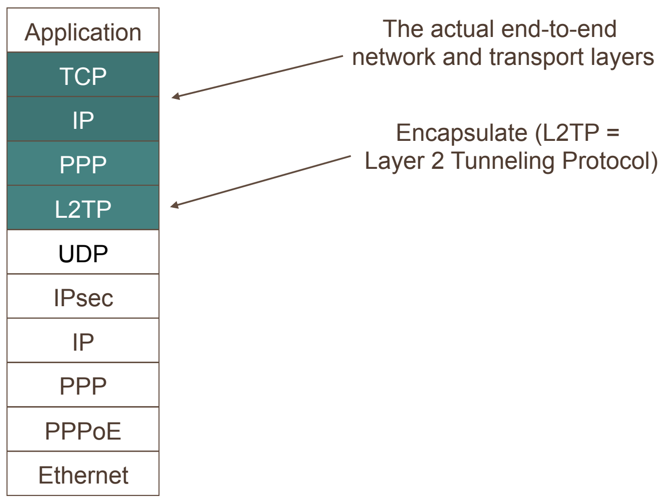
Application
TCP
IP
PPP
L2TP
UDP
IPsec
IP
PPP
PPPoE
Ethernet

Example Microsoft VPN stack

Application
TCP
IP
PPP
L2TP
UDP
IPsec
IP
PPP
PPPoE
Ethernet

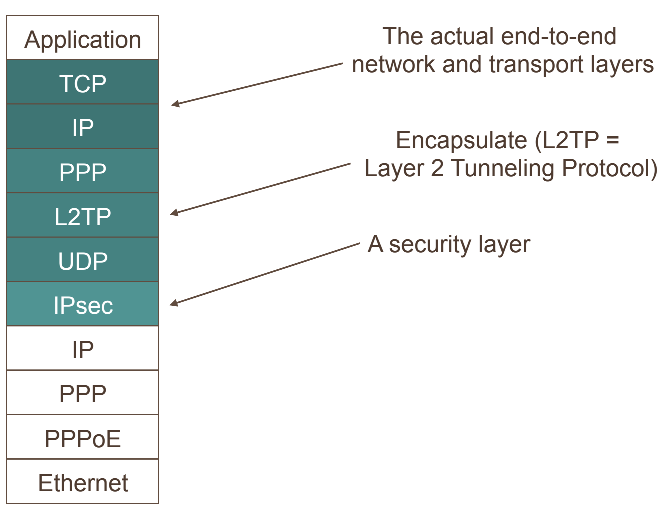
The actual end-to-end network and transport layers

Example Microsoft VPN stack



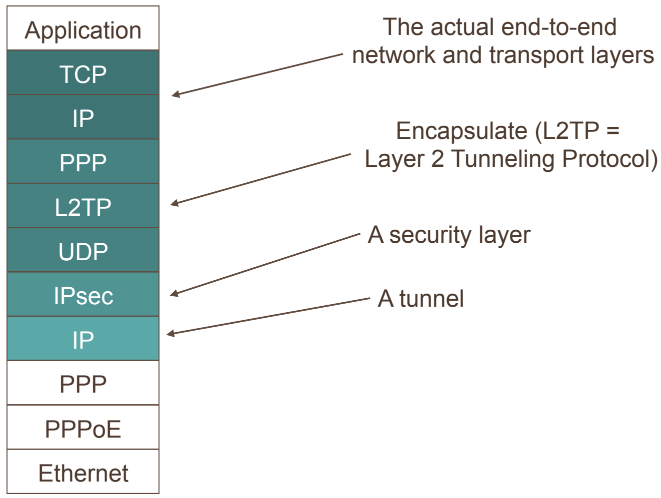
65

Example Microsoft VPN stack



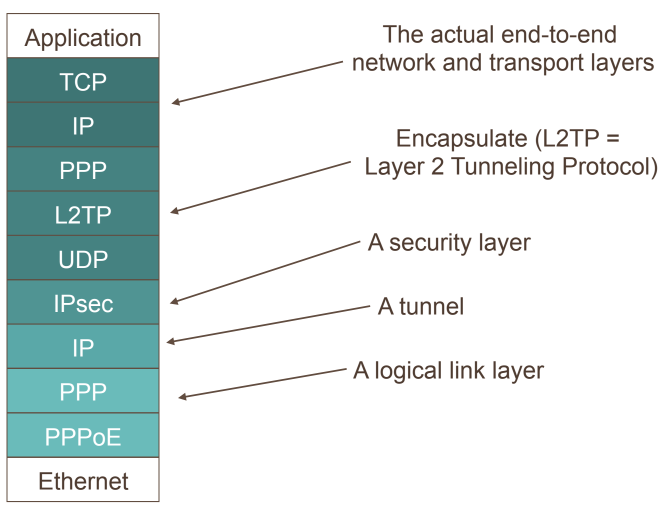
66

Example Microsoft VPN stack



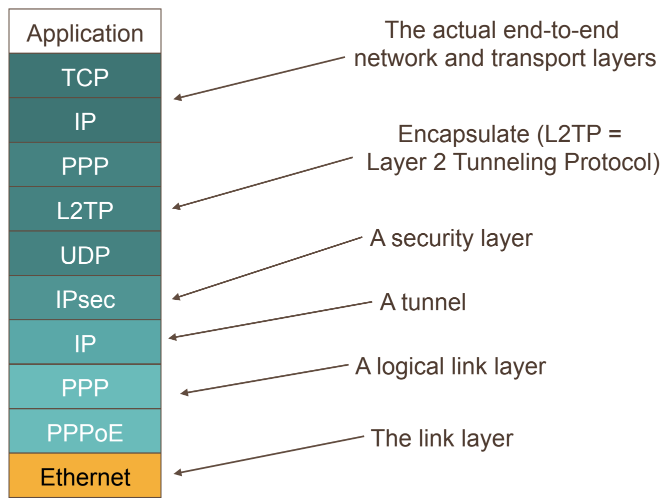
67

Example Microsoft VPN stack



68

Example Microsoft VPN stack



69

Example Microsoft VPN stack

TCP:	Transport Control Protocol
IP:	Internet Protocol
PPP:	Point-to-Point Protocol
L2TP:	Layer 2 Tunneling Protocol
UDP:	User Datagram Protocol
IPsec:	Secure IP
PPPoE:	PPP over Ethernet

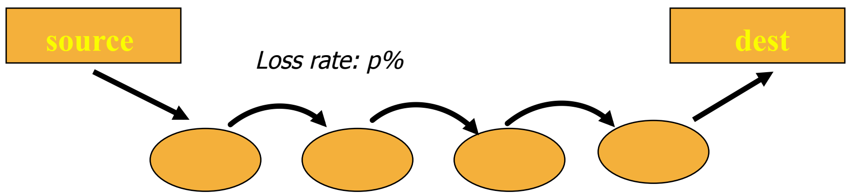
70

END-TO-END ARGUMENT

End-to-End argument

- Internet:
  - Suppose an IP packet will take n hops
  - probability p of loss on each hop
  - Transfer a file of k IP packets
  - Should we:
    - use a retransmission protocol running “end-to-end” or
    - n TCP protocols in a chain?

End-to-End argument



Probability of successful transit: $(1\text{-}p)^n$,
Expected packets lost: $k\text{-}k^*(1\text{-}p)^n$

73

Saltzer et. al. analysis

- If p is <u>very</u> small, then even with many hops most packets will get through
  - Overhead of using TCP protocols in the links
  - End-to-end recovery mechanism

74

Generalized End-to-End view?

- Low-level mechanisms should focus on speed, not reliability
- The application should worry about "properties" it needs
- In general, add additional functionality *end-to-end in the application*
  - Not in the network

What can we learn from this?

- That the internet is a mature technology
  - Kludges on kludges
- *That the end-to-end argument actually works!*

When should the network do more?

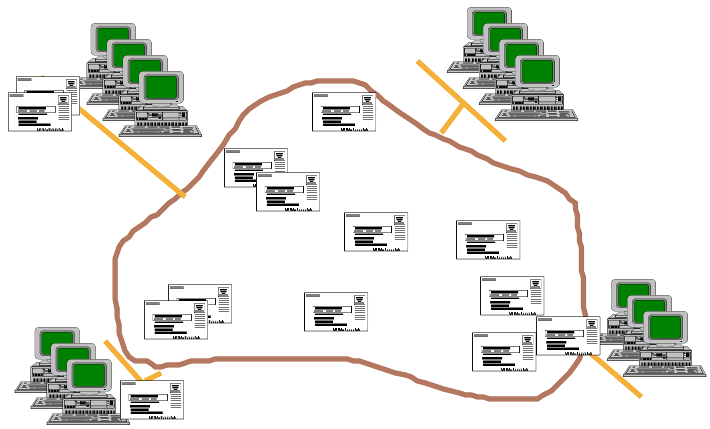
- When you get performance gains
  - Link-level retransmissions over a lossy link
  - Ex: wireless network
- Also
  - When the network doesn't trust the end user
    - Corporation or military
  - Some things can't be done at the end
    - Routing algorithms
    - Billing
    - User authentication

77

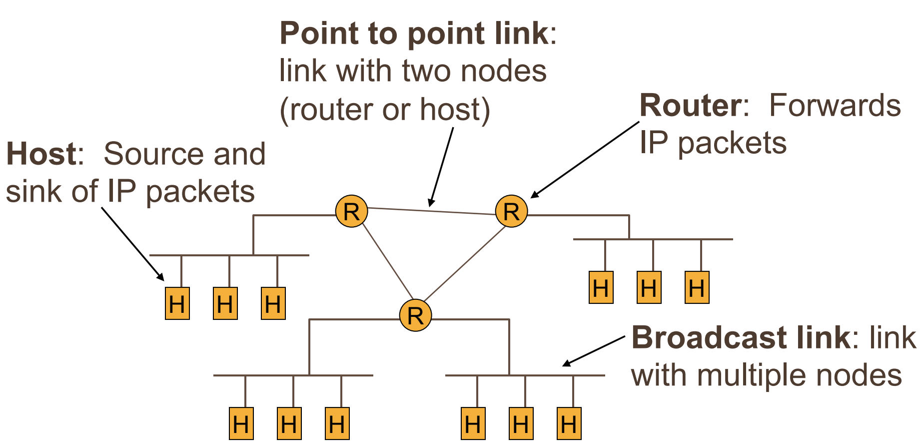
NETWORK INFRASTRUCTURE

78

A network is like a "mostly reliable" post office



Network components

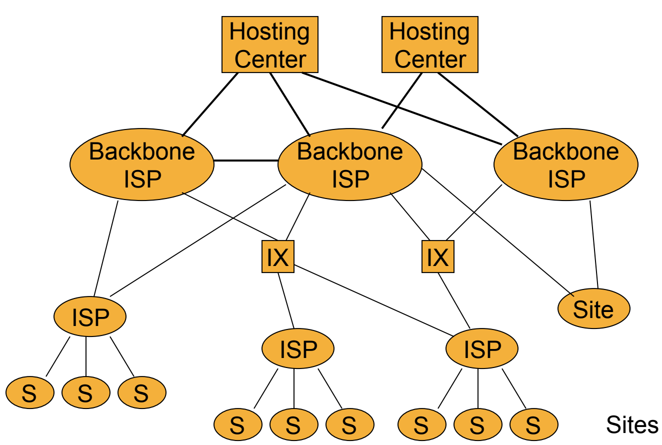


Network components

- **Network:** Collection of hosts, links, and routers
- **Site:** Stub network, typically in one location and under control of one administration
- **Firewall/NAT:** Box between the site and ISP
- **ISP:** Internet Service Provider. Transit network that provides IP connectivity for sites
- **Backbone ISP:** Transit network for regional ISPs and large sites
- **Inter-exchange (peering point):** Broadcast link where multiple ISPs connect and exchange routing information (peering)
- **Hosting center:** Stub network that supports lots of hosts (web services), high speed connections to many backbone ISPs.
- **Bilateral peering:** Direct connection between two backbone ISPs

81

Internet topology



IXs came first

IXs tend to be performance bottlenecks

Hosting centers and bilateral peering are a response to poor IXs

82

NETWORK PROTOCOLS

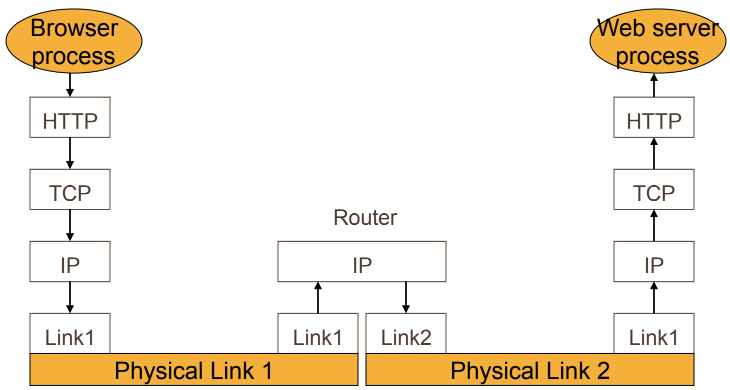
83

Protocol layering

- Communications stack consists of a set of services
  - Each providing a service to the layer above
  - Using services of the layer below
  - Each service has a programming API
- Each service has to convey information for one or more peers across the network
- This information is contained in a header

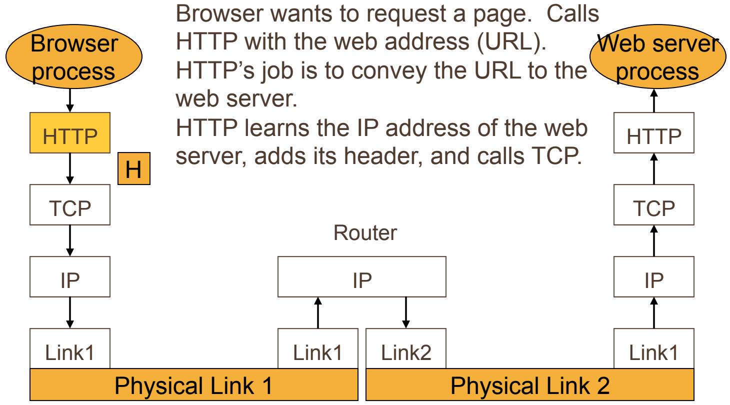
84

Protocol layering example



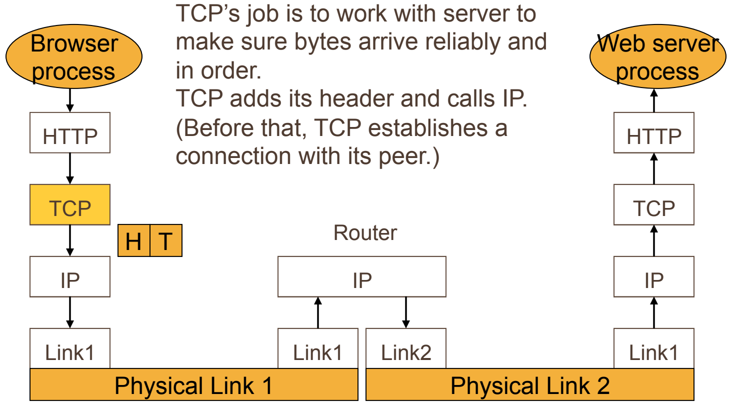
Protocol layering example

Browser wants to request a page. Calls HTTP with the web address (URL). HTTP's job is to convey the URL to the web server.
HTTP learns the IP address of the web server, adds its header, and calls TCP.



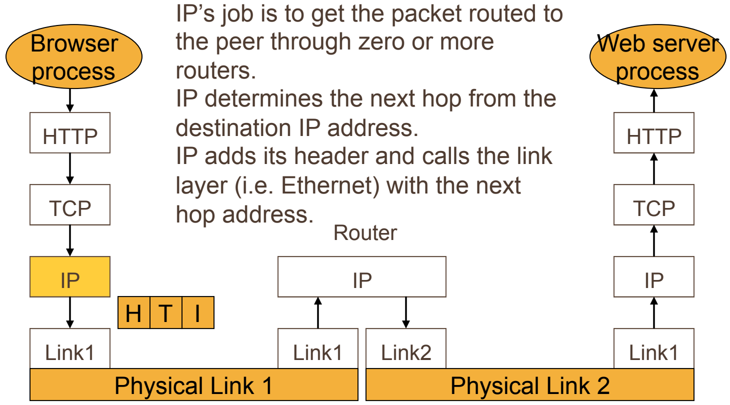
Protocol layering example

TCP's job is to work with server to make sure bytes arrive reliably and in order.
TCP adds its header and calls IP.
(Before that, TCP establishes a connection with its peer.)



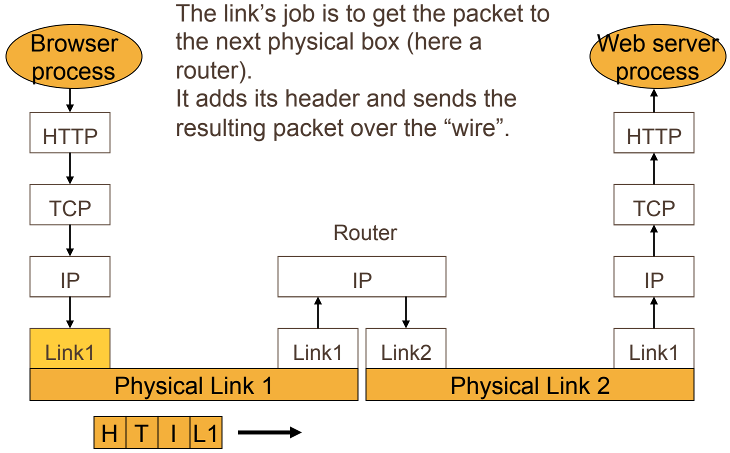
Protocol layering example

IP's job is to get the packet routed to the peer through zero or more routers.
IP determines the next hop from the destination IP address.
IP adds its header and calls the link layer (i.e. Ethernet) with the next hop address.



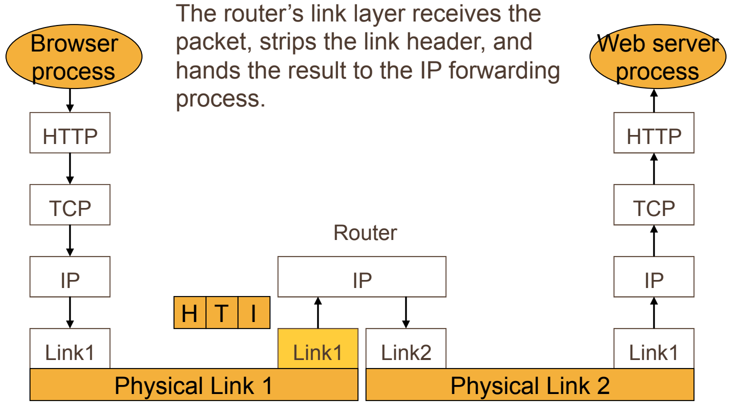
Protocol layering example

The link's job is to get the packet to the next physical box (here a router).
It adds its header and sends the resulting packet over the "wire".



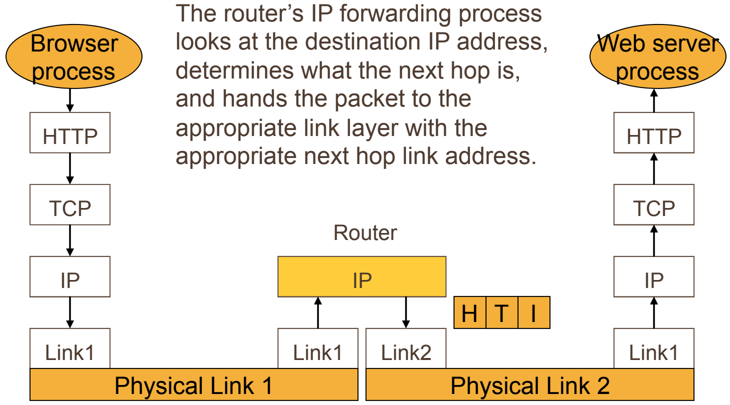
Protocol layering example

The router's link layer receives the packet, strips the link header, and hands the result to the IP forwarding process.



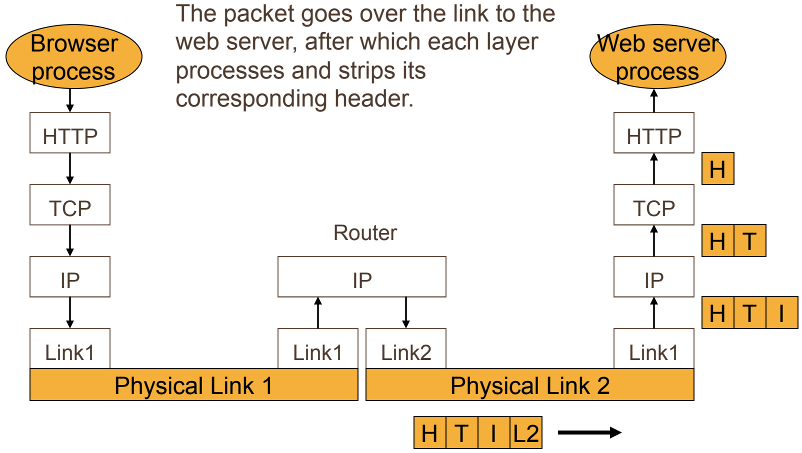
Protocol layering example

The router's IP forwarding process looks at the destination IP address, determines what the next hop is, and hands the packet to the appropriate link layer with the appropriate next hop link address.



Protocol layering example

The packet goes over the link to the web server, after which each layer processes and strips its corresponding header.



TCP/IP PROTOCOLS

Basic elements of any protocol header

- Demuxing field
  - Indicates which is the next higher layer (or process, or context, etc.)
- Length field or header delimiter
  - For the header, optionally for the whole packet
- Header format may be text (HTTP, SMTP (email)) or binary (IP, TCP, Ethernet)

Demuxing fields

- Ethernet: Protocol Number
  - Indicates IPv4, IPv6, (old: Appletalk, SNA, Decnet, etc.)
- IP: Protocol Number
  - Indicates TCP, UDP, SCTP
- TCP and UDP: Port Number
  - Well known ports indicate FTP, SMTP, HTTP, SIP, many others
  - Dynamically negotiated ports indicate specific processes (for these and other protocols)
- HTTP: Host field
  - Indicates "virtual web server" within a physical web server
  - (Well, more like an identifier than a demuxing field)

95

IP (Internet Protocol)

- Three services:
  - Unicast: transmits a packet to a specific host
  - Multicast: transmits a packet to a group of hosts
  - Anycast: transmits a packet to one of a group of hosts (typically nearest)
- Destination and source identified by the IP address (32 bits for IPv4, 128 bits for IPv6)
- All services are unreliable
  - Packet may be dropped, duplicated, and received in a different order

96

IP address

- Both source and destination address may be modified in transit
  - By NAT boxes
  - But even so, can reply to the source IP address
  - Unless source address is spoofed
- IP (unicast) address is hierarchical, but host can treat it as a flat identifier (given net mask)
  - Can't tell how close or far a host is by looking at its IP address

97

IP(v4) address format

- In binary, a 32-bit integer
- In text, this: "155.246.89.22"
  - Each decimal digit represents 8 bits (0 – 255)
- a.b.c.d/n
  - Last n bits identify machine
  - First 32-n bits identify network on the Internet
  - Ex: <u>155.246.89</u>.0/8 for most Stevens hosts

98

IP(v4) address format

- "Private" addresses are not globally unique:
  - Used behind NAT boxes
  - 10.0.0.0/8, 172.16.0.0/12, 192.168.0.0/16
- Multicast addresses start with 1110 as the first 4 bits (Class D address)
  - 224.0.0.0/4
- Unicast and anycast addresses come from the same space

99

UDP (User Datagram Protocol)

- Runs above IP
- Same unreliable service as IP
  - Packets can get lost anywhere:
    - Outgoing buffer at source
    - Router or link
    - Incoming buffer at destination
- But adds port numbers
  - Mailboxes
  - Used to identify "application layer" processes
- Also a checksum, optional

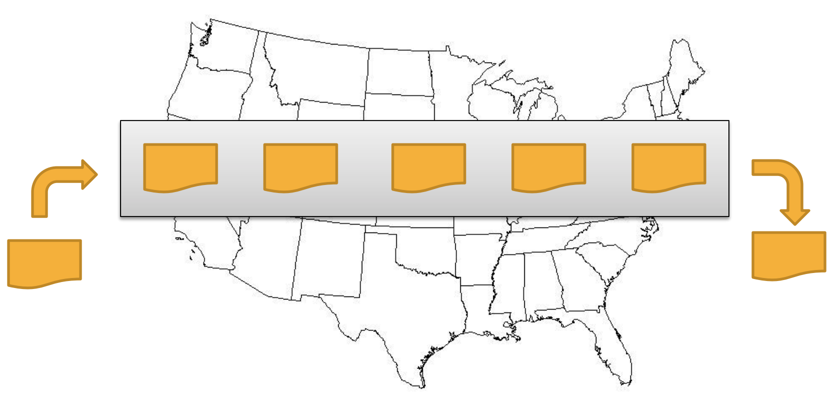
100

TCP (Transmission Control Protocol)

- Runs above IP
  - Port number and checksum like UDP
- Service is in-order byte stream
  - Bytes transparently packaged in packets
- Flow control and congestion control
- Connection setup and teardown phases
- Can be considerable delay between bytes in at source and bytes out at destination
  - Because of timeouts and retransmissions
- Works only with unicast (not multicast or anycast)

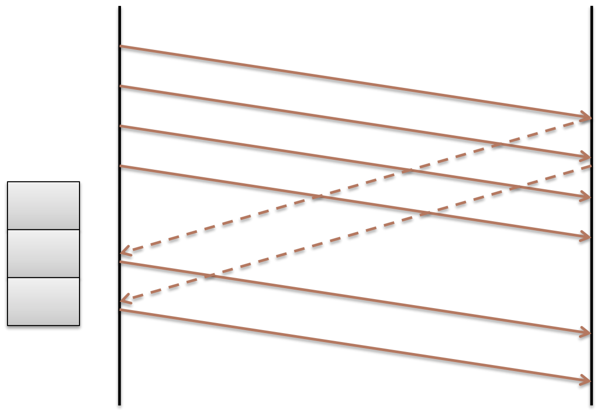
101

TCP Pipeline



102

TCP Windowing



103

UDP vs. TCP

- UDP is more real-time
  - Packet is sent or dropped, but is not delayed
- UDP has more of a "message" flavor
  - One packet = one message
  - But must add reliability mechanisms over it
- TCP good for transferring a file
  - Frustrating for messaging
  - Interrupts to application don't conform to message boundaries
  - No "Application Layer Framing"
- TCP is vulnerable to DoS (Denial of Service) attacks
  - SYN flood

104

SCTP (Stream Control Transmission Protocol)

- IETF standard
- Overcomes many limitations of TCP
  - Motivation is SS7 signaling over IP
- Message oriented---supports message framing
- Multiple streams for a given session
  - Interruption in one stream does not effect others
- Cookie mechanism for DoS attacks

105

Summary

- TCP, UDP, IP provide a nice set of basic tools
  - Key is to understand concept of protocol layering
- But problems/limitations exist
  - IP has been compromised by NAT, can't be used as a stable identifier
  - Firewalls can block communications
  - TCP has vulnerabilities
  - Network performance highly variable

106