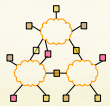
15-446 Distributed Systems Spring 2009



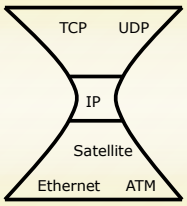
Midterm Review

Topics Covered

- Networking (Lecture 1~3)
- Naming
- RPC
- Time
- Replication
- Security

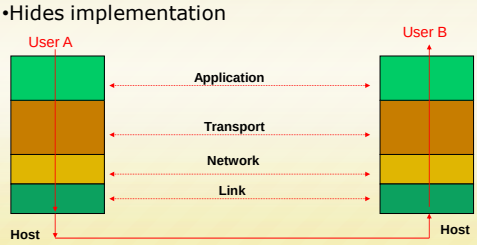
Internet Design

- Packet-switched datagram network
- IP is the "compatibility layer"
  - Hourglass architecture
  - All hosts and routers run IP
- Stateless architecture
  - No per flow state inside network



Layering

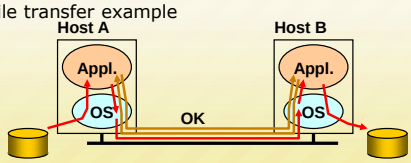
•Each layer relies on services from layer below and exports services to layer above
•Interface defines interaction
•Hides implementation



Layering: technique to simplify complex systems

End-to-End Argument

- Deals with where to place functionality
  - Inside the network (in switching elements)
  - At the edges
- Argument:
  - There are functions that can only be correctly implemented by the endpoints – do not try to completely implement these elsewhere
  - File transfer example



5

E2E Example: File Transfer

- If network guaranteed reliable delivery
  - The receiver has to do the check anyway!
    - E.g., network card may malfunction
- Full functionality can only be entirely implemented at application layer; no need for reliability from lower layers
- Is there any need to implement reliability at lower layers?
  - Yes, but only to improve performance
  - If network is highly unreliable
    - Adding some level of reliability helps performance, not correctness

6

Transport Layer

UDP

- Single socket to receive messages
- No guarantee of delivery
- Not necessarily in-order delivery
- Datagram – independent packets

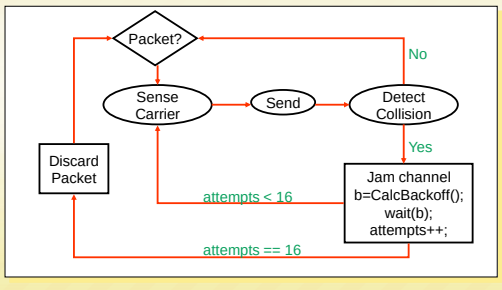
TCP

- Reliable – guarantee delivery
- Byte stream – in-order delivery
- Connection-oriented – single socket per connection
- Setup connection followed by data transfer

7

Link Layer
Ethernet MAC (CSMA/CD)

- Carrier Sense Multiple Access/Collision Detection



8

Routing

- Distance Vector
  - Have cost/next hop of best known path to each destination
  - Advertise cost/next hop of best known path to each destination
- Link State
  - Every node gets complete copy of graph
  - Every node "floods" network with data about its outgoing links

Message complexity
- LS: with n nodes, E links, O(nE) messages
- DV: exchange between neighbors only O(E)

9

TCP

- Establishment – 3-way handshake
- Tear down – 4-way handshake
- Sliding window protocol
  - AIMD
  - Reaches steady state quickly
  - ACK clocking
  - TCP timeout based on RTT estimation

10

Wireless Network Breaking assumptions

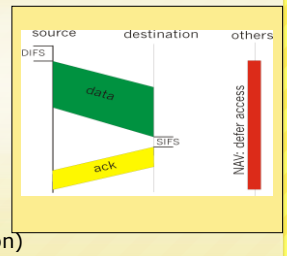
- CSMA/CD does not work
  - Relevant contention at the receiver, not sender
  - Hard to build a radio that can transmit and receive at same time
  - 802.11-CSMA/CA
  - Bluetooth-TDMA, Frequency hopping
- Wireless devices are mobile
  - Support for mobile addressing/routing
  - Layer of indirection (Home agent, foreign agent)
- Higher bit error rate
  - TCP performance degrades when link loss rate is high
  - TCP assumes packet loss is due to congestion

11

IEEE 802.11 MAC Protocol: CSMA/CA

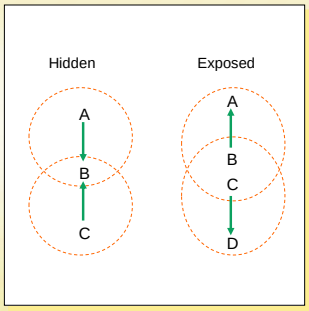
802.11 CSMA: sender
- If sense channel idle for **DISF (Distributed Inter Frame Space)** then transmit entire frame (no collision detection)
- If sense channel busy then binary backoff

802.11 CSMA receiver:
- If received OK return ACK after **SIFS (Short IFS)** (ACK is needed due to lack of collision detection)



12

Hidden and Exposed terminals



CSMA/CA is used

13

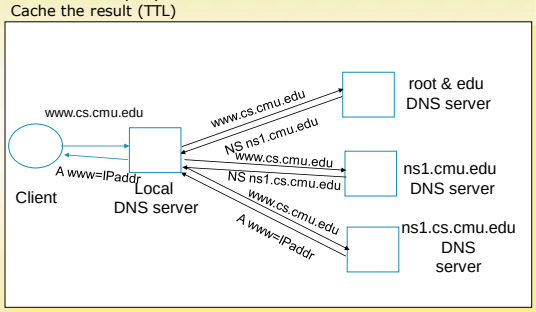
Naming

- Names are mapped to values within some context → binding
- Naming is a powerful tool in system design
  - A layer of indirection can solve many problems (e.g., server selection)
- How to determine context?
  - Implicit → from environment
  - Explicit → e.g., recursive names
  - Search paths
- How DNS works

14

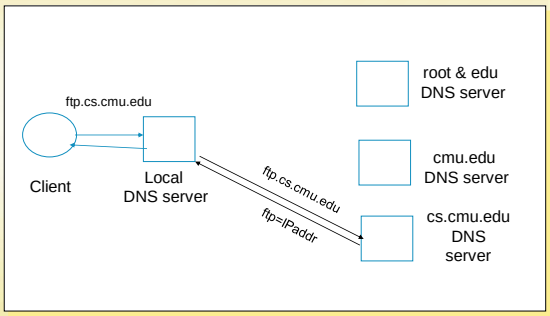
DNS Resolution

Recursive query
Non-recursive query
Cache the result (TTL)



15

Subsequent Lookup Example



16

RPC

- A remote procedure call makes a call to a remote service look like a local call
  - RPC makes transparent whether server is local or remote
  - RPC allows applications to become distributed transparently
  - RPC makes architecture of remote machine transparent
- Except. Hard to provide true transparency
  - Failures
  - Latency
  - Memory access
- How to deal with hard problem → give up and let programmer deal with it
  - "Worse is better"

17

RPC's difference from LPC

- Failure semantics
  Possible semantics for RPC:
  - Exactly-once
    - Impossible in practice
  - At least once:
    - Only for idempotent operations
  - At most once
    - Zero, don't know, or once
  - Zero or once
    - Transactional semantics
  - At-most-once most practical But different from LPC
- Parameter passing
  - Passing reference (copy and restore)
  - Solves the problem only partially

18

Physical Time

- Clocks on different systems will always behave differently
  - Skew and drift between clocks
- Time Synchronization
  - Cristian's algorithm
  - NTP
  - Berkeley algorithm (Synchronizes a group of servers)

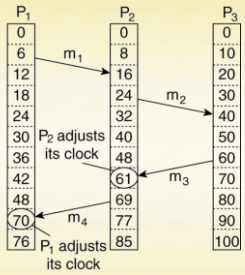
19

Logical Clock

- Lamport Clock
  - Capture's happens before relationship
  - Totally ordered multicast
- Vector Clock
  - Capture's casual relationship
  - Casually ordered multicast

20

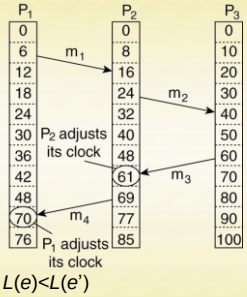
Lamport's Logical Clocks



- The "happens-before" relation → can be observed directly in two situations: locally, by message
- Lamport's algorithm corrects the clocks.

21

Lamport's Logical Clocks



- $e \rightarrow e'$ implies $L(e)<L(e')$
- The converse is not true, that is $L(e)<L(e')$ does not imply $e \rightarrow e'$

22

Totally Ordered Multicast

- Use Lamport timestamps
- Algorithm
  - Message is timestamped with sender's logical time
  - Message is multicast (including sender itself)
  - When message is received
    - It is put into local queue
    - Ordered according to timestamp
    - Multicast acknowledgement
  - Message is delievered to applications only when
    - It is at head of queue
    - It has been acknowledged by all involved processes
  - Lamport algorithm (extended) ensures total ordering of events

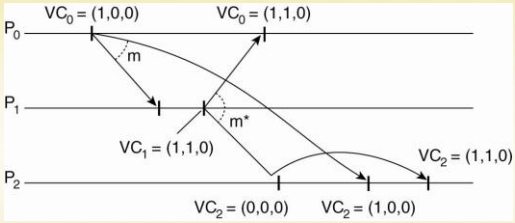
23

Vector Clocks

- Vector clocks overcome the shortcoming of Lamport logical clocks
  - $L(e) < L(e')$ does not imply e happened before e'
- Vector timestamps are used to timestamp local events
- They are applied in schemes for replication of data

24

Causally Ordered Multicast



- Enforcing causal communication

25

Replication

- Replication → good for performance/reliability
  - Key challenge → keeping replicas up-to-date
- Consistency
  - Consistency Model is a contract between processes and a data store

26

Data Centric Consistency Models

Consistency	Description
Strict	Absolute time ordering of all shared accesses matters.
Linearizability	All processes must see all shared accesses in the same order. Accesses are furthermore ordered according to a (nonunique) global timestamp
Sequential	All processes see all shared accesses in the same order. Accesses are not ordered in time
Causal	All processes see causally-related shared accesses in the same order.
FIFO	All processes see writes from each other in the order they were used. Writes from different processes may not always be seen in that order

(a)

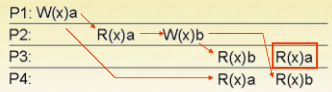
Consistency	Description
Weak	Shared data can be counted on to be consistent only after a synchronization is done
Release	Shared data are made consistent when a critical region is exited
Entry	Shared data pertaining to a critical region are made consistent when a critical region is entered.

(b)

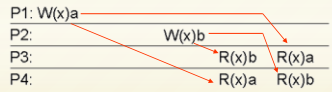
a) Consistency models not using synchronization operations.
b) Models with synchronization operations.

27

Causal Consistency



(a)



(b)

a) A violation of a causally-consistent store. The two writes are NOT concurrent because of the $R_2(x)a$.
b) A correct sequence of events in a causally-consistent store ($W_1(x)a$ and $W_2(x)b$ are concurrent).

28

Implementing Sequential Consistency

- Primary-based
  - Each data item has associated primary responsible for coordination
  - Remote-write protocols (primary writes to replicas)
  - Local-write protocols (primary migrates)
- Replica-based
  - Active replication using multicast communication
  - Quorum-based protocols
    - ROWA: R=1, W=N
      - Fast reads, slow writes (and easily blocked)
    - RAWO: R=N, W=1
      - Fast writes, slow reads (andeasily blocked)
    - Majority: R=W=N/2+1

29

Client-centric consistency models

- Consistency models for accessing databases by mobile users
  - Clients access any (typically a nearby) copy of the data
- Only provide consistency guarantees of data accesses made for individual clients (not concerned with ensuring consistency across multiple clients
- Four different client-central consistency models
  - Monotonic reads
  - Monotonic writes
  - Read your writes
  - Writes follow reads

30

ACID vs BASE

- Modern Internet systems: focused on BASE
  - Basically Available
  - Soft-state (or scalable)
  - Eventually consistent
- What goals might you want from a system?

  C, A, P (You can only have two out of these three properties)

Strong Consistency: all clients see the same view, even in the presence of updates

High Availability: all clients can find some replica of the data, even in the presence of failures

Partition-tolerance: the system properties hold even when the system is partitioned

31

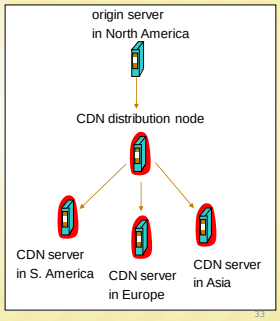
Eventual consistency

- There are replica situations where updates (writes) are rare and where a fair amount of inconsistency can be tolerated.
- If no updates occur for a while, all replicas should gradually become consistent.

32

Content Distribution Networks (CDNs)

- The content providers are the CDN customers.
- Content replication
- CDN company installs hundreds of CDN servers throughout Internet
  - Close to users
- CDN replicates its customers' content in CDN servers. When provider updates content, CDN updates servers



33

Server Selection

- Service is replicated in many places in network
- How do direct clients to a particular server?
  - As part of routing → anycast, cluster load balancing
  - As part of application → HTTP redirect
  - As part of naming → DNS
- Which server?
  - Lowest load → to balance load on servers
  - Best performance → to improve client performance
    - Based on Geography? RTT? Throughput? Load?
  - Any alive node → to provide fault tolerance

34

How Akamai Works

- Root server gives NS record for akamai.net
- Akamai.net name server returns NS record for g.akamaitech.net
  - Name server chosen to be in region of client's name server
  - TTL is large
- G.akamaitech.net nameserver chooses server in region
  - Uses consistent hashing from aXYZ
    - Load balancing
    - Performance
    - Fault tolerance
  - TTL is small → why?

35

Security

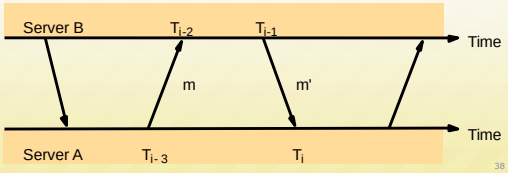
- Symmetric (pre-shared key, fast) and asymmetric (key pairs, slow) primitives provide:
  - Confidentiality
  - Integrity
  - Authentication

  (You should know how to use them)
- "Hybrid Encryption" leverages strengths of both. (TLS)
- Great complexity exists in securely acquiring keys. (Key distribution)
  - Kerberos
  - PKI

BACKUPs

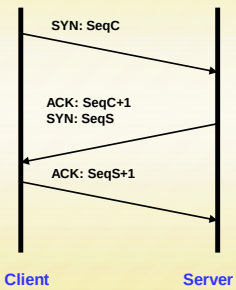
NTP Protocol

- All modes use UDP
- Each message bears timestamps of recent events:
  - Local times of Send and Receive of previous message
  - Local times of Send of current message
- Recipient notes the time of receipt T_i (we have T_{i-3}, T_{i-2}, T_{i-1}, T_i)
- In symmetric mode there can be a non-negligible delay between messages



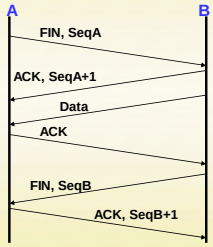
Establishing Connection: Three-Way handshake

- Each side notifies other of starting sequence number it will use for sending
  - Why not simply chose 0?
    - Must avoid overlap with earlier incarnation
    - Security issues
- Each side acknowledges other's sequence number
  - SYN-ACK: Acknowledge sequence number + 1
- Can combine second SYN with first ACK



Tearing Down Connection

- Either side can initiate tear down
  - Send FIN signal
  - "I'm not going to send any more data"
- Other side can continue sending data
  - Half open connection
  - Must continue to acknowledge
- Acknowledging FIN
  - Acknowledge last sequence number + 1

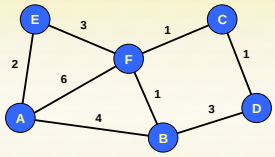


41

Distance-Vector Method

Initial Table for A

Dest	Cost	Next Hop
A	0	A
B	4	B
C	∞	–
D	∞	–
E	2	E
F	6	F

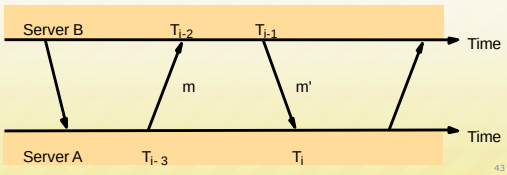


- Idea
  - At any time, have cost/next hop of best known path to destination
  - Use cost ∞ when no path known
- Initially
  - Only have entries for directly connected nodes

42

NTP Protocol

- All modes use UDP
- Each message bears timestamps of recent events:
  - Local times of Send and Receive of previous message
  - Local times of Send of current message
- Recipient notes the time of receipt T_i (we have T_{i-3}, T_{i-2}, T_{i-1}, T_i)
- In symmetric mode there can be a non-negligible delay between messages



43