Computer Networks
ICS 651

- bitcoin
- allnet
- exam review:
  - transport layer
  - TCP basics
  - congestion control
  - project 2

Bitcoin

- distributed, reliable ("hard to falsify") time-stamping network
- each time-stamp record ("block") includes existing transactions, and a summary (hash from Merkle tree) of all previous transactions
- finding valid blocks require compute power,
  - first n bits of hash must all be zero
  - can only be found at random (we think!)
  - n evolves to keep the number of blocks generated at a near constant rate
- "block miners" get a reward for finding a block with a suitable hash
- each block keeps track of all transactions since the last block by including a hash of the most recent block
- P2P network distributes the blocks to all peers

Bitcoin transactions

- each transaction has a list of input transactions, adding up to an amount B
- each transaction has a list of output amounts and public keys, adding up to an amount B' <= B
- if B' < B, the difference B - B' belongs to the block miner as a transaction fee
- the input transactions must provide evidence of having a private key equivalent to the public key of the corresponding output transaction

Bitcoin block chain

- each bitcoin miner produces and hashes as many blocks as possible until it finds a hash with the first n bits zero
- the miner then broadcasts the block as fast as possible
- every other miner includes this block into a new block they try to generate
- two (or more) miners may find blocks A and B approximately simultaneously, and broadcast both throughout the network
- every miner has a choice of which block to include in the block they are working on
- the next successful block C will have picked one of the winners of the previous round, A or B
- almost all miners will now build their block on C, confirming either A or B, and one of them will produce a new block D
- eventually, only one of A or B is confirmed by the growing block chain

Other applications of blockchains

- distributed, reliable network for time-stamping transactions
- might be useful anywhere distributed recordkeeping is needed
  - e.g. real-estate transactions
- but proof-of-work is expensive!
  - proof-of-stake may be a less energy-intensive alternative
  - bankers may like the idea of a distributed network limited to only carefully-selected participants
- in any case, there must be incentive for the miners
  - e.g. in bitcoin, miners get newly-minted coins and transaction fees

AllNet

- See this 4-minute talk at the March 30th, 2016, Wetware Wednesday

  http://www2.hawaii.edu/~esb/2016spring.ics651/allnet-talk.pdf

- See also

  http://alnt.org/

AllNet status, 2018

- ad-hoc communication between Linux systems of ad-hoc 802.11 WiFi
- DHT used for communication over the Internet
- bandwidth-limited to an average 8KB/second
  - priorities let us forward own packets even when bandwidth is limited
- trace mechanism for debugging, similar to ping+traceroute

```
b1.01/16     0 hop
b3.00/16     1 hop
b4.00/16     2 hop
b5.00/16     3 hop      2.519748s rtt
```

transport layer

- provides at least the demultiplexing function: between two IP hosts there can be many connections (socket pairs), identified by pairs of port numbers
- mostly, TCP for stream- and connection-oriented reliable transmission, UDP for everything else
- TCP provides additional functions, particularly flow control and congestion control
- also SCTP, RTP

TCP basics

- reliable byte stream
- requires sequence and acknowledgment numbers
  - TCP is based on **cumulative** acks, but has a Selective Ack option
- also requires state on the endpoints to keep track of sequence numbers and buffers
- state allocation and deallocation is explicit in TCP, letting the application figure out when to open and close connections
- explicit management of each receiver buffer: the TCP **window**

TCP details

- each byte (and the SYN and FIN bits) has its own sequence number
- the sequence number in the packet is the sequence number of the first data byte (or SYN/FIN) in the packet
- the corresponding ACK adds the sequence number and the number of bytes (+SYN/FIN) in the packet
- for sender, left edge of window is ack number received, right edge is ack+window-1
- TCP adaptive timer
- Karn algorithm (do not use retransmitted segments in RTT estimation), Nagle algorithm (only send full segments or when everything is acked, or after a timeout)
- delayed acks
- TCP checksum, pseudo-header
- TCP header, including congestion control bits, urgent pointer
- zero window, silly window syndrome
- for full-speed transmission, window must be larger than bandwidth*delay product

congestion control

- congestion collapse in the 70's and 80s
- AIMD: additive increase, multiplicative decrease
- round-trip-time (RTT) TCP congestion window into rate control
- ways of detecting congestion before it occurs: increase in RTT
- TCP Reno: aggressive window decrease, slightly less aggressive when fast retransmit is triggered by duplicate acks
- TCP Reno always waits until a packet is lost (probably to congestion) before slowing down
- TCP Vegas: slow down linearly if the RTT is above minimum, increase linearly otherwise

queueing and fairness

- FIFO: packets added to end of queue, dropped if queue is full
- Random Early Discard attempts to slow down TCP flows before queue is full
- priority queues can favor some classes of traffic
- global fairness is generally impossible, but can be approximated
- local fairness is easier, but it favors flows that cross fewer congested routers
- fair queueing tries to send the same number of bits per unit time for every flow that has data to send

other transport protocols

- Stream Control Transmission Protocol
- Real Time Protocol and Real Time Control Protocol
- Real Time Streaming Protocol
- Session Initiation Protocol

project 2

- simplified TCP
- buffering, windows, threads
- implemented reliable transmission
- implemented three-way handshake, closing connections
- implemented the sockets API