Today

- review

review

where we've been...

- network architecture (layers)
- BSD sockets, DNS
- UDP/TCP clients and servers
- concurrency (select/fork/signal)
- BSD servers (inetd)
- IPv4/v6 protocols
- tools
- RPC/XDR, UNIX domain
- broadcast/multicast
- network security, IPsec

Objectives

- understand BSD socket calls
- understand IP protocols
- examine client/servers
  "how do they do that"
- develop client/servers
  design
  implement
  diagnose
- understand design tradeoffs
- develop portable library

Concepts

- **client/servers**
  - concurrent/iterative
  - statefull/stateless
- **concurrency**
  - multiple processes/threads
  - asynchronous events (signal)
  - nonblocking I/O
- **communications**
  - stream/datagram (circuits/packets)
  - reliable/unreliable
  - multicast
  - performance

philosophy

I hear, and I forget.
I see, and I remember.
I do, and I understand.

```
          Assignments
  1  telnet and ping
  2  Ethernet and traceroute
  3  TCP client,
  4  UDP server, TCP client/server
  5  concurrent i/o (select)
  6  concurrent server (fork)
  7  library
  8  TCP distributed application
  9  RPC
  10 multicast
  11 reliable UDP
```


papers

- Boggs/Metcalf original ethernet paper
- An analysis of TCP processing overhead
- Internet time synchronization: the Network Time Protocol
- An evening with Berferd
- Security problems in the TCP/IP suite
- Kerberos: An Authentication Service for Open Network Systems
- IPv6: The New Internet Protocol
- ATM concepts

BSD sockets

- transport layer interface
- API, subroutine library
- no standards (BSD is de facto)
- supports multiple protocol families (TCP/IP, XNS, UNIX, IPv6, ATM)
- flexibility is paid for in complexity
- mixture of filling data structures and function calls
- supports I/O abstraction
- **full duplex**, can't read what you write
- readn/writen for streams

```
socket bind connect listen accept read write sendto recvfrom
setsockopt getsockname gethostbyname gethostbyaddr inet_ntoa
getservbyname htons

signal fork wait select alarm
```


IP architecture

- layers and encapsulation
- IP: addressing, routing, fragmentation, TTL
- ICMP: echo and error feedback
- UDP: ports, checksums, unreliable
- TCP: timers, sequence numbers, buffers, retransmission, ACKs, flow control, congestion control, 3-way handshake, close
- performance: big buffers, concurrency

IPv6

- bigger addresses
- sleek header
- more efficient routing
- QoS stuff
- IPsec
- interoperate with IPv4
- API changes

network security

is there a problem?

- authenticity, integrity, privacy, availability
- DoS, spoofing, hijacking
- scanners, sniffers, overflows
- wrappers, kerberos, firewalls
- IPsec: authentication and encryption
- where to defend?
- where to encrypt?

tools

- perror()
- trace/truss
- netstat -r -a -s -i
- lsof
- ifconfig
- ping, telnet
- traceroute
- nslookup, arp
- ttcp, netperf, sock
- tcpdump
- rpcinfo

network research

visit the IETF pages

- security
- quality of service
- multicast
- high speed, low latency
- mobile IP
- routing

TCP: What if?

things that go bump in the net

1. unknown hostname
2. TCP connect, and no server process
3. TCP connect, server host down
4. active TCP session, ctrl-c server
5. inactive TCP session, ctrl-c server
6. active TCP session, server computer crashes
7. inactive TCP session, server computer crashes
8. inactive TCP session with KEEPALIVE, server computer crashes
9. inactive TCP session, server computer crashes and reboots
10. start 2nd copy of server
11. server tries to bind to port < 1024
12. A sends faster than B can receive

UDP: What if?

1. UDP sendto, and no server process
2. UDP sendto with connect(), and no server process
3. active UDP session, ctrl-c server
4. inactive UDP session, server computer crashes and reboots
5. server tries to bind to port < 1024
6. start 2nd copy of server
7. A sends faster than B can receive (UDP)
8. A sends faster than its interface (ENOBUFS)
9. A sends 10000 byte datagram
10. A sends 500-byte datagram, B recvfrom len is 100
11. A sending to B, can C send to B?
12. fd's/host (TCP vs UDP)

no next time