Today

- tcpdump
- IP

tcpdump

VERY handy!

- promiscuous packet watcher
- requires root privileges (controlled substance)
- /usr/local/etc/tcpdump on altair, deneb, achernar
- sources available for most UNIX systems
- maybe a win* version ??
- some hosts don't show their outbound datagrams (use 3 machines)
- interesting sources, packet decoding
- also asctcpdump.pl

other OSs: snoop, etherfind

tcpdump

see man page

- tcpdump options expression

```
options
 -e   display Ether header
 -x   display datagram in hex
 -s snaplen  number of bytes to capture
 -n  don't do addr. to name translation
 -N  just short hostname
 -v  verbose (TTL, ID)
 -t  no timestamp

 -w filename   save stuff to filename
 -r filename  read datagrams from filename, not network

expression  -- limit output
 tcp and port 7
 udp and not host deneb
 "(udp or tcp) and host menkar"
```


tcpdump examples

assignment 1

```
on deneb: telnet 128.169.94.253  (75 second timeout, SunOS)
on altair: tcpdump host 128.169.94.253

20:24:34.764735 arp who-has 128.169.94.253 tell ALTAIR.CS.UTK.EDU
20:24:40.410180 arp who-has 128.169.94.253 tell ALTAIR.CS.UTK.EDU
20:24:52.407644 arp who-has 128.169.94.253 tell ALTAIR.CS.UTK.EDU
20:25:16.402628 arp who-has 128.169.94.253 tell ALTAIR.CS.UTK.EDU

on cetus1a: telnet 128.169.94.253   (3:17 timeout)
20:28:41.654549 arp who-has 128.169.94.253 (Broadcast) tell CETUS1A.CS.UTK.EDU
 5 more  1 second apart
20:28:53.846165 arp who-has 128.169.94.253 (Broadcast) tell CETUS1A.CS.UTK.EDU
 5 more
20:29:18.231596 arp who-has 128.169.94.253 (Broadcast) tell CETUS1A.CS.UTK.EDU
 5 more
20:30:06.992203 arp who-has 128.169.94.253 (Broadcast) tell CETUS1A.CS.UTK.EDU
 5 more
20:31:03.254659 arp who-has 128.169.94.253 (Broadcast) tell CETUS1A.CS.UTK.EDU
 5 more
```

tcpdump example

ping

```
on cetus1a:  ping altair
on deneb:  tcpdump icmp

20:37:39.205905 CETUS1A > ALTAIR: icmp: echo request (DF)
20:37:39.206474 ALTAIR > CETUS1A: icmp: echo reply (DF)
20:37:40.202292 CETUS1A > ALTAIR: icmp: echo request (DF)
20:37:40.202828 ALTAIR > CETUS1A: icmp: echo reply (DF)
20:37:41.202413 CETUS1A > ALTAIR: icmp: echo request (DF)
20:37:41.202954 ALTAIR > CETUS1A: icmp: echo reply (DF)
20:37:42.202627 CETUS1A > ALTAIR: icmp: echo request (DF)
20:37:42.203106 ALTAIR > CETUS1A: icmp: echo reply (DF)
20:37:43.202929 CETUS1A > ALTAIR: icmp: echo request (DF)
20:37:43.203404 ALTAIR > CETUS1A: icmp: echo reply (DF)
```


tcpdump example

assignment 5

```
 on cetus1a:  asnmt5 altair 7   (and then type stuff)
 on deneb:  tcpdump port 7

deneb% tcpdump -N port 7
20:13:00.880609 CETUS1A.34875 > ALTAIR.echo: udp 8 (DF)
20:13:00.893517 ALTAIR.echo > CETUS1A.34875: udp 8

tcpdump -N -x port 7
20:14:46.849982 CETUS1A.34875 > ALTAIR.echo: udp 8 (DF)
          4500 0024 92c1 4000 ff11 2c68 80a9 5e15
          80a9 5d37 883b 0007 0010 029a 7465 7374
          696e 670a 5555 5555 5555 5555 5555
20:14:46.862804 ALTAIR.echo > CETUS1A.34875: udp 8
          4500 0024 3559 0000 3c11 8cd1 80a9 5d37
          80a9 5e15 0007 883b 0010 0000 7465 7374
          696e 670a 0000 4008 0002 0640 4355

asctcpdump.pl -N -x port 7
20:17:35.545727 CETUS1A.34875 > ALTAIR.echo: udp 8 (DF)
 4500 0024 92c2 4000 ff11 2c67 80a9 5e15      E..$..@...,g..^.
 80a9 5d37 883b 0007 0010 029a 7465 7374      ..]7.;......test
 696e 670a 5555 5555 5555 5555 5555           ing.UUUUUUUUUU
20:17:35.558512 ALTAIR.echo > CETUS1A.34875: udp 8
 4500 0024 3569 0000 3c11 8cc1 80a9 5d37      E..$5i..<.....]7
 80a9 5e15 0007 883b 0010 0000 7465 7374      ..^....;....test
 696e 670a 0000 1d04 06f0 0000 139b           ing...........
```


tcpdump examples

syslog

```
on thistle:
  syslog.conf
  *.info;mail.none;authpriv.none    /var/log/messages
  *.info;mail.none;authpriv.none    @thdsun

 C code
   openlog("tomtest",LOG_PID,LOG_MAIL);
   syslog(LOG_AUTH|LOG_NOTICE,"sys log test auth/notice");

asctcpdump.pl -x -s 256 port 514

08:00:02.557018 thistle.syslog > thdsun.syslog: udp 44
 4500 0048 341d 0000 4011 1d74 86a7 0f0c      E..H4...@..t....
 86a7 0cba 0202 0202 0034 6db4 3c33 373e      .........4m.<37>
 746f 6d74 6573 745b 3937 3833 5d3a 2073      tomtest[9783]: s
 7973 206c 6f67 2074 6573 7420 6175 7468      ys log test auth
 2f6e 6f74 6963 650a                          /notice.
```


layers

```
+------+ +-----+ +-----+     +-----+
|Telnet| | FTP | | TFTP| ... | ... |      Application
+------+ +-----+ +-----+     +-----+
     |   |         |           |     -- BSD api
     +-----+     +-----+     +-----+
     | TCP |     | UDP | ... | ... |      Transport
     +-----+     +-----+     +-----+
        |          |           |
     +--------------------------+----+
     |    Internet Protocol & ICMP   |    Network
     +--------------------------+----+
                   |
       +---------------------------+
       |   Local Network Protocol  |      Data link
       +---------------------------+
```

encapsulation

```
   16       20      20/8                  4
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+- ... -+-+-+-+
 | mac   |    IP   |TCP/UDP|  Data ..... | CRC |
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+- ... -+-+-+-+

  tcpdump -e -x -t -N icmp
8:0:20:86:51:de 8:0:20:8:a:4c ip 98: CETUS1A > ALTAIR: icmp: echo request (DF)
          4500 0054 92d3 4000 ff01 2c36 80a9 5e15
          80a9 5d37 0800 fa7f 1dd0 0003 37f0 0fdc
          000d acd0 0809 0a0b 0c0d 0e0f 1011 1213
          1415 1617 1819 1a1b 1c1d 1e1f 2021 2223
          2425 2627 2829 2a2b 2c2d 2e2f 3031 3233
          3435
8:0:20:8:a:4c 8:0:20:86:51:de ip 98: ALTAIR > CETUS1A: icmp: echo reply (DF)
          4500 0054 36d3 4000 ff01 8836 80a9 5d37
          80a9 5e15 0000 0280 1dd0 0003 37f0 0fdc
          000d acd0 0809 0a0b 0c0d 0e0f 1011 1213
          1415 1617 1819 1a1b 1c1d 1e1f 2021 2223
          2425 2627 2829 2a2b 2c2d 2e2f 3031 3233
          3435
```


IP

IPv4 Internet Protocol (RFC791)

- connectionless (datagram)
- unreliable
- checksum on header only
- fragmentation/assembly based on interface MTU
- 32-bit address (src/dest)
- transport protocol field
- TTL (hop count)
- options (mostly unused)
- routing (network layer)
- each datagram routed independently

IP header

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|Version|  IHL  |Type of Service|          Total Length         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         Identification        |Flags|      Fragment Offset    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Time to Live |    Protocol   |         Header Checksum       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                       Source Address                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    Destination Address                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    Options                    |    Padding    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

20 bytes to 60 (with options)

transmitted bits 0-7 first, network byte order

IP header fields

- **version** – usually 4, 6 for IPv6
- **IHL** – header length, in 4-byte words, usually 5
- **TOS** – type of service, not really used (differentiated services)
  - PPPTTTT0
  - 3-bit precedence field, ignored
  - 4 TOS bits (minimize delay, maximize throughput, maximize reliability, minimize cost)
  - telnet: 0x10, ftp: 0x08, snmp:0x04, NNTP: 0x02
  - unused loworder bit
- 16-bit **length** of IP datagram in bytes (max 65535)
- **id, flags, offset** used in fragmentation

IP header fields

- **ttl** time-to-live field, decremented by each router, usually starts at 64 (varies), if 0 then dropped
- **protocol** transport protocol (see IANA)

```
some of our favorite protocols
 1  ICMP               41  IPv6
 4  IP in IP           50  ESP (IPsec)
 6  TCP                51  AH  (IPsec)
17  UDP                47  GRE (PPTP)
```

- 16-bit header **checksum**, if checksum fails, datagram is silently dropped
- 32-bit **source** and **destination** IP address
- optional **options** and 0-padding to make multiple of 4 bytes

IP header

assignment 4 broadcast

```
tcpdump -n -x port 18765
 128.169.93.200.1321 > 128.169.95.255.18765: udp 8

 4500 0024   906e 0000   4011 2b41   80a9 5dc8

 80a9 5fff    0529 494d 0010 15c1    1138 a99f

 21a5 0000
```


IP Addresses

```
             +--------+--------+--------+--------+
IP     A     |0 net(7)|   host (24)              |
             +--------+--------+--------+--------+
                             32 bits
             +--------+--------+--------+--------+
       B     |10   net (14)    |     host (16)   |
             +--------+--------+--------+--------+
             +--------+--------+--------+--------+
       C     |110   net (21)            |host(8) |
             +--------+--------+--------+--------+
             +--------+--------+--------+--------+
       D     |1110     multicast (28)            |
             +--------+--------+--------+--------+

also  net:subnet:host
```

routing is done with net (locally subnet)

IP addresses

- localhost 127.0.0.1
- broadcast 255.255.255.255
- subnet broadcast 192.12.68.255
- private A 10.*.*.*
- private B's 172.16-31.*.*
- private C's 192.168.0-254.*
- UT 128.169.*.*, CS 128.169.92-95.*
- ORNL 128.219.*.*, 134.167.*.*, 192.31.96.*

kernel: IP output

look at IP destination address

- localhost?
- on net of one of interfaces?
  if yes, check ARP table
- route table entry?
- no, send to default router
  need ARP table entry
- with destination Ether address
  send it out

arp -a

ARP

find IP address on broadcast net

- if IP address is on local net and not in cache, **broadcast** ARP request
- receive reply and cache, send IP packets
- cache entry times out in about 20 minutes

```
RFC826    Ether type  806  (not IP)

 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   Hardware                    |          protocol             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  HLEN         |  PLEN         |         operation             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|             sender hardware address (0-3)                     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| sender HA (4-5)               |  sender internet addr (0-1)   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  sender IA (2-3)              |  target HA (0-1)              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    target hardware addr (2-5)                 |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    target internet addr (0-3)                 |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
    28 bytes
```


tcpdump of ARP

tcpdump -e -x -t -N arp

```
   altair=128.169.93.55  whisper=128.169.93.200

0:c0:4f:6b:a5:52 Broadcast arp 60: arp who-has ALTAIR tell WHISPER
             0001 0800 0604 0001 00c0 4f6b a552 80a9


             5dc8 0000 0000 0000 80a9 5d37

8:0:20:8:a:4c 0:c0:4f:6b:a5:52 arp 60: arp reply
     ALTAIR is-at 8:0:20:8:a:4c
             0001 0800 0604 0002 0800 2008 0a4c 80a9


             5d37 00c0 4f6b a552 80a9 5dc8


hardware : 1 for ether
proto: 0x800 for IP
HLEN: size of hardware address. 6 for ether
PLEN: size of proto addr., 4 for IP
op: 1=arp request  2=arp reply  3=RARP req  4=RARP reply
```


IP fragmentation

- fragment packet to fit within MTU
- MTU's vary over the route
- done by sending host or routers
- re-assembled at receiver
- dropped from IPv6

Network	MTU
Hyperchannel	65535
ATM/AAL5	65535
token ring	17914
IP/ATM	9180
FDDI	4352
Ethernet	1500
PPP	296

netstat -i shows MTU

IP fragmentation

IP header

- 16-bit **ID** field same for each frag
- ID field is incremented by kernel on each datagram transmitted
- 3-bit **flag** field

```
Bit 0: reserved, must be zero
Bit 1: (DF) 0 = May Fragment,  1 = Don't Fragment.
Bit 2: (MF) 0 = Last Fragment, 1 = More Fragments.

   0   1   2
 +---+---+---+
 |   | D | M |  offset in remaining 13 bits
 | 0 | F | F |
 +---+---+---+
```

- 13-bit **offset**, unit of 8-bytes, where this fragment fits
- only first frag carries transport header etc., it will have 0 offset and MF flag set
- frags may arrive out of order
- last frag will have 0 in flags, non-zero offset
- UDP can cause frags (NFS)
- TCP tries to avoid frags
- MTU discovery tries to find optimal MTU for route RFC1191

tcpdump of frags

ttcp -t -s -u -n3 -l4096 thistle.cind.ornl.gov

```
 tcpdump -N -t src host thdsun.epm.ornl.gov

 thdsun.3802 > thistle.2000: udp 4096 (frag 8700:1480@0+)
 thdsun > thistle: (frag 8700:1480@1480+)
 thdsun > thistle: (frag 8700:1144@2960)

 tcpdump with -x
thdsun.3804 > thistle.2000: udp 4096 (frag 8853:1480@0+)
        4500 05dc 2295 2000 3711 1268 86a7 0cba
        86a7 0f0c 0edc 07d0 1008 0000 2021 2223
thdsun > thistle: (frag 8853:1480@1480+)
        4500 05dc 2295 20b9 3711 11af 86a7 0cba
        86a7 0f0c 4f50 5152 5354 5556 5758 595a
thdsun > thistle: (frag 8853:1144@2960)
        4500 048c 2295 0172 3711 3246 86a7 0cba
        86a7 0f0c 2728 292a 2b2c 2d2e 2f30 3132
```


IP options

- mostly unused
- extends header up to 60 bytes
- IHL greater than 5
- multiple of 4-bytes
- setsockopt() IP_OPTIONS
- dropped by IPv6

IP options

```
format: type byte   or   type lth data
option types
 0      end of options
 1      no-op
 2      security   11 byte length
 3      loose source routing, variable length
 9      strict source routing, variable length
 7      record route, variable length
68      timestamp, variable length
```

- security option specifies packet classification
- ping has record-route option
- source routing is dangerous, blocked by routers

next time

ICMP

raw sockets

ping/traceroute