Get your instruments and play along!

Start a NetBSD instance, then log in on it.
A simple example

$ telnet www.google.com 80
A simple example

$ telnet www.google.com 80
Trying 2607:f8b0:400c:c03::67...
Connected to www.google.com.
Escape character is '^['].
GET / HTTP/1.0
A simple example

$ telnet www.google.com 80
Trying 2607:f8b0:400c:c03::67...
Connected to www.google.com.
Escape character is '^[].
GET / HTTP/1.0

HTTP/1.0 200 OK
Date: Mon, 17 Mar 2014 16:15:01 GMT
Content-Type: text/html; charset=ISO-8859-1
Server: gws
[...]
A simple example

What exactly happens?
A simple example

What exactly happens?

- local host connects to remote host
- sends command
- receives data
A simple example

How exactly do we connect to the remote host?

- look up hostname
- open connection to IP address
A simple example

How exactly do we look up a hostname?
A simple example

$ ktrace -i telnet www.google.com 80
Trying 173.194.73.99...
Connected to www.google.com.
Escape character is ’^]’.
GET / HTTP/1.0

[...]
$ kdump >trace
...open a few files...

```plaintext
[...]
2541 1 ktrace NAMI "/bin/telnet"
2541 1 ktrace RET execve -1 errno 2 No such file or directory
2541 1 ktrace CALL execve(0xbf7fe8b4,0xbf7fed50,0xbf7fed60)
2541 1 ktrace NAMI "/usr/bin/telnet"
2541 1 ktrace NAMI "/usr/libexec/ld.elf_so"
[...]
2541 1 telnet CALL open(0xbb4445e7,0,0x1b6)
2541 1 telnet NAMI "/etc/nsswitch.conf"
2541 1 telnet RET open 3
[...]
2541 1 telnet CALL open(0xbb441fb6,0x400000,0x1b6)
2541 1 telnet NAMI "/etc/hosts"
2541 1 telnet RET open 3
[...]
2541 1 telnet CALL open(0xbb441ef0,0x400000,0x1b6)
2541 1 telnet NAMI "/etc/resolv.conf"
2541 1 telnet RET open 3
[...]
2541 1 telnet GIO fd 3 read 69 bytes
"# Generated by resolvconf\ndomain ec2.internal\nnameserver 172.16.0.23\n"
```
... query a DNS server ...

```plaintext
[...]
2541  1 telnet  RET  __socket30  5
2541  1 telnet  CALL  connect(5,0xbb48e7d0,0x10)
2541  1 telnet  MISC  mbsoname: [172.16.0.23]
2541  1 telnet  RET  connect  0
2541  1 telnet  CALL  sendto(5,0xbf7ee458,0x20,0,0,0)
2541  1 telnet  MISC  msghdr: [name=0x0, namelen=0, iov=0xd96c7f20, iovlen=1, control=0x0, controllen=3647766376, flags=0]
2541  1 telnet  GIO  fd 5 wrote 32 bytes
"\M-*\M^Y\^A\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\Cwww Fgoogle\Ccom\0\0\0\0\0\0\0\0\0\^A"
2541  1 telnet  RET  sendto  32/0x20
[...]
2541  1 telnet  CALL  poll(0xbf7eddd0,1,0x1388)
2541  1 telnet  RET  poll  1
2541  1 telnet  CALL  recvfrom(5,0xbb12f000,0x10000,0,0xbf7ede00,0xbf7eddcc)
2541  1 telnet  MISC  msghdr: [name=0x0, namelen=3246359232, iov=0xd96c7f18, iovlen=1, control=0x0, controllen=3223644263, flags=0]
2541  1 telnet  GIO  fd 5 read 48 bytes
"\M^M\M-1\M^A\M-@\0\0\0\0\0\0\0\0\0\Cwww Fgoogle\Ccom\0\0\0\0\0\0\0\0\0\D\M-X:\M-Id"
[...]
```
A simple example

How exactly do we look up a hostname?

- look up various local files
- open a connection to a DNS server’s IP
- ask DNS server to resolve hostname
- get back IP

And then?
...communicate with the remote host...

[...]

2541 1 telnet  CALL  write(1,0xbb118000,0x19)
2541 1 telnet  GIO  fd 1 wrote 25 bytes

"Trying 216.58.201.100..."

2541 1 telnet  RET  write 25/0x19
2541 1 telnet  CALL  __socket30(2,1,6)
2541 1 telnet  RET  __socket30 5
2541 1 telnet  CALL  connect(5,0xbb1070c0,0x10)
2541 1 telnet  MISC  mbsoname: [216.58.201.100]
2541 1 telnet  RET  connect 0

[...]

2541 1 telnet  RET  poll 1
2541 1 telnet  CALL  read(0,0x806a920,0x400)
2541 1 telnet  GIO  fd 0 read 15 bytes

"GET / HTTP/1.0"

2541 1 telnet  RET  read 15/0xf
2541 1 telnet  CALL  poll(0xbf7febec,3,0)
2541 1 telnet  RET  poll 1
2541 1 telnet  CALL  sendto(5,0x8068e40,0x10,0,0,0)
2541 1 telnet  MISC  msghdr: [name=0x0, namelen=0, iov=0xd96c7f20, iovlen=1, control=0x0, controllen=3647766376, flags=0]
2541 1 telnet  GIO  fd 5 wrote 16 bytes

"GET / HTTP/1.0"

2541 1 telnet  RET  sendto 16/0x10
Ok, so how does this work?

- determine which nameserver to query
- ask who has a route to the nameserver
- open socket to well defined port on remote IP
- send queries
- open socket to requested port on remote IP
What does this look like on the wire?

# script commands.out
# ifconfig -a
# route -n get default
# cat /etc/resolv.conf
# tcpdump -w tcpdump.out port not 22 &
# arp -d -a
# ping -n -c 3 98.139.180.149
# telnet www.google.com 80
[...]
# kill %1
# exit
# exit
$ scp <instance-name>:*out ~/tmp/
A simple example

Finding the next hop:

```
$ tcpdump -n -r /tmp/tcpdump.out arp
reading from file /tmp/tcpdump.out, link-type EN10MB (Ethernet)
20:26:03.511549 ARP, Request who-has 10.234.84.193 tell 10.234.84.220, length 28
20:26:03.511709 ARP, Reply 10.234.84.193 is-at fe:ff:ff:ff:ff:ff, length 28
20:26:13.318920 ARP, Request who-has 10.234.84.220 tell 10.234.84.193, length 28
20:26:13.318949 ARP, Reply 10.234.84.220 is-at 22:00:0a:ea:54:dc, length 28
```
A simple example

Performing the DNS query:

$ tcpdump -t -n -r tcpdump.out udp port 53
reading from file tcpdump.out, link-type EN10MB (Ethernet)
IP 172.16.0.23.53 > 10.234.84.220.65524: 55270 1/0/0 AAAA 2607:f8b0:4004:80a::2004 (60)
IP 172.16.0.23.53 > 10.234.84.220.65523: 7749 1/0/0 A 216.58.217.164 (48)
A simple example

Establishing the connection to the server:

$ tcpdump -n -r tcpdump.out tcp port 80
IP 10.234.84.220.65529 > 216.58.217.164.80: Flags [S],
    seq 2069980376, win 32768, options [....], length 0
IP 216.58.217.164.80 > 10.234.84.220.65529: Flags [S.],
    seq 26050190, ack 2069980377, win 42540, options [....], length 0
IP 10.234.84.220.65529 > 216.58.217.164.80: Flags [.],
    ack 1, win 4197, options [....], length 0
A simple example

Sending the HTTP request:

IP 10.234.84.220.65529 > 216.58.217.164.80: Flags [P.], seq 1:17,
   ack 1, win 4197, options [...], length 16: HTTP: GET / HTTP/1.0
IP 216.58.217.164.80 > 10.234.84.220.65529: Flags [.],
   ack 17, win 333, options [...], length 0
IP 10.234.84.220.65529 > 216.58.217.164.80: Flags [P.], seq 17:19,
   ack 1, win 4197, options [...], length 2: HTTP
IP 216.58.217.164.80 > 10.234.84.220.65529: Flags [,]
   ack 19, win 333, options [...], length 0
A simple example

Receiving the HTTP response:

IP 216.58.217.164.80 > 10.234.84.220.65529: Flags [.] , seq 2837:4255,
    ack 19, win 333, options [ ... ] , length 1418: HTTP
IP 216.58.217.164.80 > 10.234.84.220.65529: Flags [.] , seq 4255:5673,
    ack 19, win 333, options [ ... ] , length 1418: HTTP
IP 10.234.84.220.65529 > 216.58.217.164.80: Flags [.] ,
    ack 5673, win 3616, options [ ... ] , length 0
IP 216.58.217.164.80 > 10.234.84.220.65529: Flags [.] , seq 5673:7091,
    ack 19, win 333, options [ ... ] , length 1418: HTTP
IP 216.58.217.164.80 > 10.234.84.220.65529: Flags [.] , seq 7091:8509,
    ack 19, win 333, options [ ... ] , length 1418: HTTP
A simple example

Terminating the connection:

[...]
Notables from this simple example

“Simple” is, as usual, relative.
Notables from this simple example

“Simple” is, as usual, relative.

- host configuration assumed
- network architecture (internal or across the internet) not relevant (here)
- even simple examples cross multiple layers and protocols (HTTP, DNS; TCP, UDP, ARP)
- we haven’t even scratched the surface
**TCP/IP Basics: Protocol Layers**

<table>
<thead>
<tr>
<th>Layer</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>4. Application Layer</td>
<td>End-User application programs</td>
</tr>
<tr>
<td>3. Transport Layer</td>
<td>Delivery of data to applications</td>
</tr>
<tr>
<td>2. Network Layer</td>
<td>Basic communication, addressing, and routing</td>
</tr>
<tr>
<td>1. Link Layer</td>
<td>Network Hardware and device drivers</td>
</tr>
<tr>
<td>Physical Layer</td>
<td>Cable or physical medium</td>
</tr>
</tbody>
</table>

Examples of protocols for each layer:

- Simple Mail Transfer Protocol (RFC 821)
  Hypertext Transfer Protocol (RFC 2616)

- Transmission Control Protocol (RFC 793, tcp(4))
  User Datagram Protocol (RFC 768; udp(4))

- Internet Protocol (RFC 791; ip(4))
  Internet Control Message Protocol (RFC 792; icmp(4))

- Address Resolution Protocol (RFC 826; arp(4))
TCP/IP Basics: Protocol Layers (OSI Model)

<table>
<thead>
<tr>
<th>OSI Model</th>
<th>Data unit</th>
<th>Layer</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Host</td>
<td>Data</td>
<td>7. Application</td>
<td>Network process to application</td>
</tr>
<tr>
<td>layers</td>
<td></td>
<td>6. Presentation</td>
<td>Data representation, encryption and decryption, convert machine dependent data to machine independent data</td>
</tr>
<tr>
<td></td>
<td></td>
<td>5. Session</td>
<td>Interhost communication, managing sessions between applications</td>
</tr>
<tr>
<td>Segments</td>
<td>4. Transport</td>
<td></td>
<td>End-to-end connections, reliability and flow control</td>
</tr>
<tr>
<td>Media</td>
<td>Packet/Datagram</td>
<td>3. Network</td>
<td>Path determination and logical addressing</td>
</tr>
<tr>
<td>layers</td>
<td>Frame</td>
<td>2. Data link</td>
<td>Physical addressing</td>
</tr>
<tr>
<td></td>
<td>Bit</td>
<td>1. Physical</td>
<td>Media, signal and binary transmission</td>
</tr>
</tbody>
</table>
TCP/IP Basics: ARP

Ethernet Address Resolution Protocol
– or –
Converting Network Protocol Addresses to 48-bit Ethernet Address for Transmission on Ethernet Hardware

```
$ arp -a
falcon.srcit.stevens-tech.edu (155.246.89.89) at 00:07:e9:09:ca:10 [ether] on eth0
grohl.srcit.stevens-tech.edu (155.246.89.9) at 00:16:3e:cf:6b:5b [ether] on eth0
hoth.srcit.stevens-tech.edu (155.246.89.10) at 00:1e:68:8e:79:d8 [ether] on eth0
cinema.srcit.stevens-tech.edu (155.246.89.67) at 00:25:90:1e:05:51 [ether] on eth0
vlan16.cc.stevens-tech.edu (155.246.89.1) at 00:00:5e:00:01:02 [ether] on eth0
vader.srcit.stevens-tech.edu (155.246.89.5) at 00:23:8b:a9:dd:60 [ether] on eth0
nirvana.phy.stevens-tech.edu (155.246.89.33) at 00:1e:68:0f:99:a2 [ether] on eth0
```
TCP/IP Basics: ARP
TCP/IP Basics: ARP

Ethernet Address Resolution Protocol

– or –

Converting Network Protocol Addresses to 48-bit Ethernet Address for Transmission on Ethernet Hardware

18:06:59.217533 ARP, Request who-has 10.114.62.1 tell 10.114.63.209, length 28
18:06:59.218187 ARP, Reply 10.114.62.1 is-at fe:ff:ff:ff:ff:ff, length 28
18:07:06.148475 ARP, Request who-has 10.114.63.209 (ff:ff:ff:ff:ff:ff) tell 0.0.0.0, length 28
18:07:06.148499 ARP, Reply 10.114.63.209 is-at 12:31:3d:04:30:23, length 28
18:08:05.820986 ARP, Request who-has 10.114.63.209 (ff:ff:ff:ff:ff:ff) tell 0.0.0.0, length 28
18:08:05.821011 ARP, Reply 10.114.63.209 is-at 12:31:3d:04:30:23, length 28
18:09:18.518859 ARP, Request who-has 10.114.63.209 (ff:ff:ff:ff:ff:ff) tell 0.0.0.0, length 28
## TCP/IP Basics: ND

### Neighbor Discovery Protocol

```
$ ndp -n -a
Neighbor Linklayer Address Netif Expire S Flags
fe80::21b:21ff:fe45:bf54%xennet0 00:1b:21:45:bf:54 xennet0 21m52s S R
fe80::21b:21ff:fe7a:7269%xennet0 00:1b:21:7a:72:69 xennet0 23h59m59s S R
fe80::e276:63ff:fe72:3900%xennet0 e0:76:63:72:39:00 xennet0 permanent R
fe80::1%lo0 (incomplete) lo0 permanent R
$```

Networking II

February 27, 2017
TCP/IP Basics: ND

Neighbor Discovery Protocol


Internet Control Message Protocol

$ ping -c 3 www.yahoo.com
PING any-fp.wa1.b.yahoo.com (67.195.160.76): 56 data bytes
64 bytes from 67.195.160.76: icmp_seq=0 ttl=53 time=30.888 ms
64 bytes from 67.195.160.76: icmp_seq=1 ttl=53 time=23.193 ms
64 bytes from 67.195.160.76: icmp_seq=2 ttl=53 time=25.433 ms

----any-fp.wa1.b.yahoo.com PING Statistics----
3 packets transmitted, 3 packets received, 0.0% packet loss
round-trip min/avg/max/stddev = 23.193/26.505/30.888/3.958 ms
$
TCP/IP Basics: ICMP: Ping
TCP/IP Basics: ICMP

Internet Control Message Protocol

$ tcpdump -r tcpdump.out -n icmp
20:26:23.964126 IP 10.234.84.220 > 207.237.69.79: ICMP echo request
20:26:23.972835 IP 207.237.69.79 > 10.234.84.220: ICMP echo reply
20:26:24.976078 IP 10.234.84.220 > 207.237.69.79: ICMP echo request
20:26:24.983500 IP 207.237.69.79 > 10.234.84.220: ICMP echo reply
20:26:25.966085 IP 10.234.84.220 > 207.237.69.79: ICMP echo request
20:26:25.973371 IP 207.237.69.79 > 10.234.84.220: ICMP echo reply
TCP/IP Basics: ICMP6

Internet Control Message Protocol for IPv6

$ ping6 -c 3 www.netbsd.org
PING6(56=40+8+8 bytes) 2001:470:30:84:204:d7b0:0:1 -->
2001:4f8:3:7:2e0:81ff:fe52:9a6b
16 bytes from 2001:4f8:3:7:2e0:81ff:fe52:9a6b, icmp_seq=0 hlim=57 time=74.316 ms
16 bytes from 2001:4f8:3:7:2e0:81ff:fe52:9a6b, icmp_seq=1 hlim=57 time=71.260 ms
16 bytes from 2001:4f8:3:7:2e0:81ff:fe52:9a6b, icmp_seq=2 hlim=57 time=71.321 ms

--- www.netbsd.org ping6 statistics ---
3 packets transmitted, 3 packets received, 0.0% packet loss
round-trip min/avg/max/std-dev = 71.260/72.299/74.316/1.747 ms
Internet Control Message Protocol for IPv6

12:46:58.524431 IP6 2001:470:30:84:204:d7b0:0:1 >
   2001:4f8:3:7:2e0:81ff:fe52:9a6b: ICMP6, echo request, seq 0, length 16
   2001:470:30:84:204:d7b0:0:1: ICMP6, echo reply , seq 0, length 16
12:46:59.532864 IP6 2001:470:30:84:204:d7b0:0:1 >
   2001:4f8:3:7:2e0:81ff:fe52:9a6b: ICMP6, echo request, seq 1, length 16
12:46:59.604016 IP6 2001:4f8:3:7:2e0:81ff:fe52:9a6b >
   2001:470:30:84:204:d7b0:0:1: ICMP6, echo reply , seq 1, length 16
12:47:00.532817 IP6 2001:470:30:84:204:d7b0:0:1 >
   2001:4f8:3:7:2e0:81ff:fe52:9a6b: ICMP6, echo request, seq 2, length 16
12:47:00.604016 IP6 2001:4f8:3:7:2e0:81ff:fe52:9a6b >
   2001:470:30:84:204:d7b0:0:1: ICMP6, echo reply , seq 2, length 16
TCP/IP Basics: ICMP: Traceroute

1. UDP port 33441, TTL=1

Router 1

TTL = TTL - 1
if TTL == 0: TIME EXCEEDED

Router 2

Router 3

Computer
TCP/IP Basics: ICMP: Traceroute
TCP/IP Basics: ICMP: Traceroute
TCP/IP Basics: ICMP: Traceroute
TCP/IP Basics: ICMP

Internet Control Message Protocol

$ traceroute www.netbsd.org
traceroute to www.netbsd.org (204.152.190.12), 64 hops max, 40 byte packets
  1  eth2-3a.core1.nav.nyc.access.net (166.84.0.1)  0.256 ms  0.165 ms  0.181 ms
  2  l3v1.nyc.access.net (166.84.66.14)  1.570 ms  1.556 ms  1.437 ms
  3  gige-g3-3.core1.nyc4.he.net (209.51.171.25)  4.963 ms  2.422 ms  1.457 ms
  4  10gigabitethernet2-3.core1.ash1.he.net (72.52.92.86)  8.423 ms  8.769 ms  7.683 ms
  5  10gigabitethernet1-2.core1.atl1.he.net (184.105.213.110) 21.898 ms 19.647 ms 19.838 ms
  6  isc.gige-g2-1.core1.atl1.he.net (216.66.0.50)  77.465 ms  77.921 ms  80.519 ms
  7  iana.r1.atl1.isc.org (199.6.12.1)  77.302 ms  78.230 ms  81.782 ms
  8  int-0-5-0-1.r1.pao1.isc.org (149.20.65.37)  81.860 ms  83.780 ms  84.160 ms
  9  int-0-0-1-0.r1.sql1.isc.org (149.20.65.10)  81.543 ms  80.193 ms  84.434 ms
 10  www.netbsd.org (204.152.190.12)  81.986 ms  81.008 ms  82.604 ms
$
TCP/IP Basics: ICMP

Internet Control Message Protocol

IP (tos 0x0, ttl 1, id 44866, offset 0, flags [none], proto UDP (17), length 40)
  166.84.7.99.44865 > 149.20.53.86.33435: [udp sum ok] UDP, length 12
IP (tos 0xc0, ttl 64, id 48796, offset 0, flags [none], proto ICMP (1), length 68)
  166.84.0.1 > 166.84.7.99: ICMP time exceeded in-transit, length 48
IP (tos 0x0, ttl 2, id 44869, offset 0, flags [none], proto UDP (17), length 40)
  166.84.7.99.44865 > 149.20.53.86.33438: [udp sum ok] UDP, length 12
IP (tos 0x0, ttl 3, id 44872, offset 0, flags [none], proto UDP (17), length 40)
  166.84.7.99.44865 > 149.20.53.86.33441: [udp sum ok] UDP, length 12
IP (tos 0x0, ttl 4, id 44875, offset 0, flags [none], proto UDP (17), length 40)
  166.84.7.99.44865 > 149.20.53.86.33444: [udp sum ok] UDP, length 12
IP (tos 0x0, ttl 252, id 6760, offset 0, flags [none], proto ICMP (1), length 56)
  154.24.25.109 > 166.84.7.99: ICMP time exceeded in-transit, length 36
...
IP (tos 0x0, ttl 248, id 0, offset 0, flags [none], proto ICMP (1), length 56)
  149.20.53.86 > 166.84.7.99: ICMP 149.20.53.86 udp port 33482 unreachable, length
TCP/IP Basics: ICMP6

Internet Control Message Protocol for IPv6

$ traceroute6 www.netbsd.org
traceroute6 to www.netbsd.org (2001:4f8:3:7:2e0:81ff:fe52:9a6b) from
   2001:470:30:84:204:d7b0:0:1, 64 hops max, 12 byte packets
1  router.vc.panic.com  0.271 ms  0.282 ms  0.155 ms
2  2001:470:30::a654:420e  5.459 ms  1.251 ms  1.073 ms
3  gige-g3-3.core1.nyc4.he.net  1.288 ms  2.001 ms  10.176 ms
4  10gigabitethernet8-3.core1.chi1.he.net  26.603 ms  20.532 ms  25.029 ms
5  2001:470:1:34::2  72.033 ms  72.377 ms  72.686 ms
6  iana.r1.ord1.isc.org  76.288 ms  72.773 ms  71.481 ms
7  int-0-0-1-8.r1.pao1.isc.org  73.027 ms  76.489 ms  77.507 ms
8  int-0-0-1-0.r2.sql1.isc.org  73.555 ms  75.367 ms  74.769 ms
9  www.NetBSD.org  72.036 ms  72.522 ms  71.39 ms
$
TCP/IP Basics: ICMP6

Internet Control Message Protocol for IPv6

   2001:4f8:3:7:2e0:81ff:fe52:9a6b.33435: UDP, length 12
   ICMP6, time exceeded in-transit [icmp6]
   2001:4f8:3:7:2e0:81ff:fe52:9a6b.33436: UDP, length 12
   [...]  
   2001:4f8:3:7:2e0:81ff:fe52:9a6b.33461: UDP, length 12
12:47:29.830787 IP6 2001:4f8:3:7:2e0:81ff:fe52:9a6b >
   2001:470:30:84:204:d7b0:0:1: ICMP6,
   destination unreachable[icmp6]
TCP/IP Basics: TCP

Transmission Control Protocol

$ telnet www.google.com 80
Trying 173.194.73.99...
Connected to www.google.com.
Escape character is ‘^]’.
GET / HTTP/1.0
TCP/IP Basics: TCP

Transmission Control Protocol

14:51:33.582076 IP 166.84.7.99.58356 > 67.195.160.76.80: S
   2267539609:2267539609(0) win 32768
   <mss 1460,nop,wscale 3,sackOK,nop,nop,nop,nop,nop,timestamp 10>
14:51:33.590748 IP 67.195.160.76.80 > 166.84.7.99.58356: S
   3229501874:3229501874(0) ack 2267539610 win 5792
   <mss 1440,sackOK,timestamp 1241180702 1,nop,wscale 8>
14:51:33.590766 IP 166.84.7.99.58356 > 67.195.160.76.80: .
   ack 1 win 4197 <nop,nop,timestamp 1 1241180702>
14:51:37.732720 IP 166.84.7.99.58356 > 67.195.160.76.80: P
   1:17(16) ack 1 win 4197 <nop,nop,timestamp 9 1241180702>
14:51:37.741763 IP 67.195.160.76.80 > 166.84.7.99.58356: .
   ack 17 win 23 <nop,nop,timestamp 12411848 53 9>
TCP/IP Basics: TCP

Transmission Control Protocol over IPv6

$ telnet www.netbsd.org 80
Trying 2001:4f8:3:7:2e0:81ff:fe52:9a6b...
Connected to www.netbsd.org.
Escape character is ‘^]’.
GET / HTTP/1.0
TCP/IP Basics: TCP

Transmission Control Protocol IPv6

14:58:11.128436 IP6 2001:470:30:84:204:d7b0:0:1.58334 >
   2001:4f8:3:7:2e0:81ff:fe52:9a6b.80: S 3232473102:3232473102(0)
   win 32768 <mss 1440,nop,wscale3,sackOK,nop,nop,nop,nop,timestamp 1[|tcp]>
14:58:11.200293 IP6 2001:4f8:3:7:2e0:81ff:fe52:9a6b.80 >
   2001:470:30:84:204:d7b0:0:1.58334: S 4139493123:4139493123(0)
   ack 3232473103 win 32768
14:58:11.200337 IP6 2001:470:30:84:204:d7b0:0:1.58334 >
   2001:4f8:3:7:2e0:81ff:fe52:9a6b.80: . ack 1 win 4140
   2001:4f8:3:7:2e0:81ff:fe52:9a6b.80: P 1:17(16) ack 1 win 4140
   2001:470:30:84:204:d7b0:0:1.58334: . ack 17 win 33120
TCP/IP Basics: UDP

User Datagram Protocol

$ nslookup www.yahoo.com
Server: 155.246.1.20
Address: 155.246.1.20#53

Non-authoritative answer:
any-fp3-lfb.wa1.b.yahoo.com canonical name = any-fp3-real.wa1.b.yahoo.com.
Name: any-fp3-real.wa1.b.yahoo.com
Address: 98.139.183.24

$
TCP/IP Basics: UDP

User Datagram Protocol

15:06:04.760444 IP (tos 0x0, ttl 64, id 0, offset 0, flags [none],
  proto UDP (17), length 59) panix.netmeister.org.49164 >

15:06:05.210569 IP (tos 0x0, ttl 63, id 1862, offset 0, flags [none],
  proto UDP (17), length 207) cache2.ns.access.net.domain >
  panix.netmeister.org.49164: 28557 4/2/2
  www.yahoo.com. CNAME fp3.wg1.b.yahoo.com.[|domain]
User Datagram Protocol over IPv6

$ dig -6 @2001:470:20::2 www.yahoo.com

;; ANSWER SECTION:
any-fp3-real.wa1.b.yahoo.com. 60 IN A 98.139.183.24

;; Query time: 51 msec
;; MSG SIZE  rcvd: 128
TCP/IP Basics: UDP

User Datagram Protocol over IPv6

15:24:20.731990 IP6 (hlim 64, next-header: UDP (17), length: 39)
   2001:470:30:84:204:d7b0:0:1.65037 > 2001:470:20::2.53:

15:24:20.976796 IP6 (hlim 61, next-header: UDP (17), length: 119)
   2001:470:20::2.53 > 2001:470:30:84:204:d7b0:0:1.65037:
   18545 4/0/0 www.yahoo.com.[|domain]
TCP/IP Basics: Putting it all together

Application Layer
- arp
- ssh, ftp, http
- dns
- traceroute

Transport Layer
- TCP
- UDP

Network Layer
- IP (v4/v6)
- ICMP (v4/v6)

Link Layer
- ARP, Device Drivers

Physical Layer
- hardware (copper, fiber, radio)
Reading

- tcpdump(8)
- ktrace(1) / strace(1)
- tcp(4)/ip(4)
- netstat(1)
- nslookup(1)