

CS615 - Aspects of System Administration

Networking

Department of Computer Science
Stevens Institute of Technology
Jan Schaumann

`jschauma@cs.stevens.edu`

`http://www.cs.stevens.edu/~jschauma/615A/`

A simple example

```
$ telnet www.yahoo.com 80
Trying 69.147.76.15...
Connected to www-real.wa1.b.yahoo.com.
Escape character is '^]'.
GET / HTTP/1.0
```

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?

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

A simple example

```
$ ktrace -i telnet www.yahoo.com 80
Trying 69.147.76.15...
Connected to www-real.wa1.b.yahoo.com.
Escape character is '^]'.
GET / HTTP/1.0

[...]
$ kdump > dump
```

A simple example

... look up various local files...

[...]

```
5921      1 telnet  CALL  open(0xbba06a65,0,0x1b6)
5921      1 telnet  NAMI  "/etc/nsswitch.conf"
5921      1 telnet  RET   open 3
```

[...]

```
5921      1 telnet  CALL  open(0xbba0474b,0,0x1b6)
5921      1 telnet  NAMI  "/etc/hosts"
5921      1 telnet  RET   open 3
```

[...]

```
5921      1 telnet  CALL  open(0xbba0495b,0,0x1b6)
5921      1 telnet  NAMI  "/etc/resolv.conf"
5921      1 telnet  RET   open 3
```

[...]

A simple example

... query a DNS server ...

[...]

```
5921      1 telnet  CALL  socket(2,2,0)
5921      1 telnet  RET   socket 3
5921      1 telnet  CALL  connect(3,0xbba210f0,0x10)
5921      1 telnet  RET   connect 0
5921      1 telnet  CALL  sendto(3,0xbfbee0d0,0x1f,0,0,0)
5921      1 telnet  GIO   fd 3 wrote 31 bytes
          "[T\^A\0\0\^A\0\0\0\0\0\0\^Cwww\^Eyahoo\^Ccom\0\0\^\\\0\^A"
```

[...]

```
5921      1 telnet  CALL  recvfrom(3,0x8077000,0x10000,0,
          0xbfbeda10,0xbfbed9d4)
5921      1 telnet  GIO   fd 3 read 139 bytes
          "[T\M^A\M^@\0\^A\0\^B\0\^A\0\0\^Cwww\^Eyahoo\^Ccom\0\0\^\\\0\^A\M-
5921      1 telnet  RET   recvfrom 139/0x8b
5921      1 telnet  CALL  close(3)
```

[...]

A simple example

... communicate with remote host ...

```
5921      1 telnet  CALL  write(1,0x806e000,0x17)
```

```
5921      1 telnet  GIO   fd 1 wrote 23 bytes
```

```
"Trying 69.147.76.15...
```

```
[...]
```

```
5921      1 telnet  CALL  recvfrom(3,0x8064b80,0x400,0,0,0)
```

```
5921      1 telnet  GIO   fd 3 read 1024 bytes
```

```
"HTTP/1.1 200 OK\r
```

```
Date: Thu, 26 Mar 2009 15:39:12 GMT\r
```

```
Last-Modified: Thu, 26 Mar 2009 15:28:26 GMT\r
```

```
Content-Length: 9490\r
```

```
Connection: close\r
```

```
Content-Type: text/html; charset=utf-8\r
```

```
<html>
```

```
<head>
```

```
<title>Yahoo!</title>
```

```
[...]
```

A simple example

What does this look like on the wire?

- 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

A simple example

What does this look like on the wire?

```
# tcpdump -i xennet0 port not 22
```

A simple example

What does this look like on the wire?

```
12:07:37.636765 arp who-has 166.84.67.2 tell 166.84.7.99
12:07:38.153534 arp who-has 204.29.154.32 tell 204.29.154.253
12:07:38.258985 arp who-has 166.84.6.205 tell 166.84.6.205
12:07:38.270324 arp who-has 166.84.5.244 tell FE0-0-1e.nav1.nyc.access.net
12:07:38.384643 IP panix.netmeister.org.ntp > splenda.rustytel.net.ntp: NTPv4 client, str
14:07:38.467148 IP splenda.rustytel.net.ntp > panix.netmeister.org.ntp: NTPv4 server, str
12:07:38.485287 IP panix.netmeister.org.53252 > cache2.ns.access.net.domain: 60435+ PTR?
12:07:38.487508 IP cache2.ns.access.net.domain > panix.netmeister.org.53252: 60435 NXDoma
12:07:38.487687 IP panix.netmeister.org.53251 > cache2.ns.access.net.domain: 60436+ PTR?
12:07:38.488530 IP cache2.ns.access.net.domain > panix.netmeister.org.53251: 60436 NXDoma
12:07:38.488666 IP panix.netmeister.org.53250 > cache2.ns.access.net.domain: 60437+ PTR?
12:07:38.561690 IP FE0-0-3.nav1.nyc.access.net.1985 > ALL-ROUTERS.MCAST.NET.1985: HSRPv0-h
12:07:38.561700 IP FE0-0-3.nav1.nyc.access.net.1985 > ALL-ROUTERS.MCAST.NET.1985: HSRPv0-h
12:07:38.565734 IP FE0-0-3.nav1.nyc.access.net.1985 > ALL-ROUTERS.MCAST.NET.1985: HSRPv0-h
12:07:38.580035 IP cache2.ns.access.net.domain > panix.netmeister.org.53250: 60437 ServFa
[...]
```

A simple example

What does this look like on the wire?

[...]

```
12:07:37.636765 arp who-has 166.84.67.2 tell 166.84.7.99
```

[...]

```
12:07:39.072263 IP panix.netmeister.org.53243 > cache2.ns.access.net.domain:
                    59127+ AAAA? www.yahoo.com. (31)
```

```
12:07:39.082518 IP cache2.ns.access.net.domain > panix.netmeister.org.53243:
                    59127 2/1/0 CNAME[|domain]
```

```
12:07:39.082744 IP panix.netmeister.org.53242 > cache2.ns.access.net.domain:
                    59128+ A? www.yahoo.com. (31)
```

```
12:07:39.100074 IP cache2.ns.access.net.domain > panix.netmeister.org.53242:
                    59128 3/2/1 CNAME[|domain]
```

[...]

A simple example

What does this look like on the wire?

[...]

```
12:07:39.100673 IP panix.netmeister.org.60686 > f1.www.vip.re1.yahoo.com.http:
    S 3745952440:3745952440(0) win 32768 <mss 1460,nop,wscale 0,
      sackOK,nop,nop,nop,nop,timestamp 0 0>
12:07:39.115932 IP f1.www.vip.re1.yahoo.com.http > panix.netmeister.org.60686:
    S 3249386671:3249386671(0) ack 3745952441 win 65535 <mss
      1460,nop,wscale 1,nop,nop,timestamp 77355732 0,sackOK,eol>
12:07:39.115974 IP panix.netmeister.org.60686 > f1.www.vip.re1.yahoo.com.http:
    . ack 1 win 33580 <nop,nop,timestamp 0 77355732>
12:07:42.242725 IP panix.netmeister.org.60686 > f1.www.vip.re1.yahoo.com.http:
    P 1:17(16) ack 1 win 33580 <nop,nop,timestamp 7 77355732>
12:07:42.354215 IP f1.www.vip.re1.yahoo.com.http > panix.netmeister.org.60686:
    . ack 17 win 33296 <nop,nop,timestamp 77358905 7>
```

[...]

A simple example

What does this look like on the wire?

```
12:07:42.518465 IP panix.netmeister.org.60686 > f1.www.vip.re1.yahoo.com.http:
                P 17:19(2) ack 1 win 33580 <nop,nop,timestamp 7 77355732>
12:07:42.530758 IP f1.www.vip.re1.yahoo.com.http > panix.netmeister.org.60686:
                . 1:1449(1448) ack 19 win 33304 <nop,nop,timestamp 77359076 7>
12:07:42.544385 IP f1.www.vip.re1.yahoo.com.http > panix.netmeister.org.60686:
                FP 8689:10032(1343) ack 19 win 33304 <nop,nop,timestamp 77359088 7>
12:07:42.544442 IP panix.netmeister.org.60686 > f1.www.vip.re1.yahoo.com.http:
                . ack 8689 win 32132 <nop,nop,timestamp 7 77359088>
12:07:42.544444 IP panix.netmeister.org.60686 > f1.www.vip.re1.yahoo.com.http:
                . ack 10033 win 30789 <nop,nop,timestamp 7 77359088>
12:07:42.547141 IP panix.netmeister.org.60686 > f1.www.vip.re1.yahoo.com.http:
                F 19:19(0) ack 10033 win 33580 <nop,nop,timestamp 7 77359088>
12:07:42.584311 IP f1.www.vip.re1.yahoo.com.http > panix.netmeister.org.60686:
                . ack 20 win 33303 <nop,nop,timestamp 77359103 7>
```

Notables from this simple example

“Simple” is, as usual, relative.

- host configuration assumed

Notables from this simple example

“Simple” is, as usual, relative.

- host configuration assumed
- network architecture (internal or across the internet) not relevant (here)

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 (DNS, TCP, UDP, ARP)

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 (DNS, TCP, UDP, ARP)
- we haven't even scratched the surface

TCP/IP Basics: Protocol Layers

Layer	Function
1. Link Layer	Network Hardware and device drivers
Physical Layer	Cable or physical medium
2. Network Layer	Basic communication, addressing, and routing
3. Transport Layer	Delivery of data to applications
4. Application Layer	End-User application programs

Examples of protocols for each layer:

1. Address Resolution Protocol (RFC 826; arp(4))
2. Internet Protocol (RFC 791; ip(4))
Internet Control Message Protocol (RFC 792; icmp(4))
3. Transmission Control Protocol (RFC 793, tcp(4))
User Datagram Protocol (RFC 768; udp(4))
4. Simple Mail Transfer Protocol (RFC 821)
Hypertext Transfer Protocol (RFC 2616)

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
```

```
Vlan16.cc.stevens-tech.edu (155.246.89.1) at 00:09:44:d1:64:00 on epic0  
guinness2.cs.stevens-tech.edu (155.246.89.6) at 00:e0:81:04:12:25 on epic0  
guinness.cs.stevens-tech.edu (155.246.89.8) at 00:50:45:5c:6e:00 on epic0  
murky.cs.stevens.edu (155.246.89.22) at 00:e0:29:6c:86:8b on epic0 permanent  
nirvana.phy.stevens-tech.edu (155.246.89.33) at 00:1e:68:0f:99:a2 on epic0  
tarantula.phy.stevens-tech.edu (155.246.89.41) at 00:50:45:5f:1c:d4 on epic0  
amstel.cs.stevens-tech.edu (155.246.89.68) at (incomplete) on epic0  
snoopy.cs.stevens-tech.edu (155.246.89.136) at 00:26:54:13:c0:27 on epic0  
dwarf.cs.stevens-tech.edu (155.246.89.207) at 00:04:75:84:04:3c on epic0
```

TCP/IP Basics: ARP

Ethernet Address Resolution Protocol

– or –

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

```
13:12:41.838940 arp who-has 204.90.78.43 tell 204.90.78.254
13:12:43.423212 arp who-has 166.84.6.205 tell 166.84.6.205
13:12:42.382758 arp reply 166.84.6.205 is-at e0:76:63:72:32:00
13:12:44.615252 arp who-has 204.29.154.251 tell 204.29.154.253
13:12:44.727658 arp who-has 204.90.78.43 tell 204.90.78.254
13:12:44.775306 arp who-has 204.29.154.44 tell 204.29.154.253
13:12:45.043106 arp who-has 204.90.78.78 tell 204.90.78.254
```

TCP/IP Basics: ICMP

Internet Control Message Protocol

```
$ ping -c 3 www.yahoo.com
PING www-real.wa1.b.yahoo.com (69.147.76.15): 56 data bytes
64 bytes from 69.147.76.15: icmp_seq=0 ttl=52 time=8.722 ms
64 bytes from 69.147.76.15: icmp_seq=1 ttl=53 time=9.382 ms
64 bytes from 69.147.76.15: icmp_seq=2 ttl=52 time=10.071 ms

----www-real.wa1.b.yahoo.com PING Statistics----
3 packets transmitted, 3 packets received, 0.0% packet loss
round-trip min/avg/max/stddev = 8.722/9.392/10.071/0.675 ms
$
```

TCP/IP Basics: ICMP

Internet Control Message Protocol

```
13:23:03.081954 IP 166.84.7.99 > 69.147.76.15: icmp 64: echo request seq 23
13:23:03.092153 IP 69.147.76.15 > 166.84.7.99: icmp 64: echo reply seq 23
13:23:04.081865 IP 166.84.7.99 > 69.147.76.15: icmp 64: echo request seq 24
13:23:04.090909 IP 69.147.76.15 > 166.84.7.99: icmp 64: echo reply seq 24
13:23:05.071735 IP 166.84.7.99 > 69.147.76.15: icmp 64: echo request seq 25
13:23:05.081368 IP 69.147.76.15 > 166.84.7.99: icmp 64: echo reply seq 25
```

TCP/IP Basics: ICMP

Internet Control Message Protocol

```
$ traceroute www.panix.com
 1 FE0-0-3.nav1.nyc.access.net (166.84.0.61)  1.315 ms  0.837 ms  1.073 ms
 2 FE0-0-128.nav2.nyc.access.net (166.84.66.5)  2.097 ms  1.580 ms  1.355 ms
 3 squid1.nyc.access.net (166.84.1.26)  0.959 ms  1.406 ms  2.497 ms
 4 w2.panix.com (166.84.62.253)  3.835 ms  3.217 ms  4.424 ms
$
```

TCP/IP Basics: ICMP

Internet Control Message Protocol

```
13:27:54.426132 IP 166.84.0.61 > 166.84.7.99: icmp 36:
    time exceeded in-transit
13:27:54.430212 IP 166.84.66.5 > 166.84.7.99: icmp 36:
    time exceeded in-transit
13:27:54.430212 IP 166.84.66.5 > 166.84.7.99: icmp 36:
    time exceeded in-transit
13:27:54.438744 IP 166.84.1.26 > 166.84.7.99: icmp 36:
    time exceeded in-transit
13:27:54.438744 IP 166.84.1.26 > 166.84.7.99: icmp 36:
    time exceeded in-transit
13:27:54.446457 IP 166.84.62.125 > 166.84.7.99: icmp 36:
    166.84.62.125 udp port 33444 unreachable
13:27:54.446457 IP 166.84.62.125 > 166.84.7.99: icmp 36:
    166.84.62.125 udp port 33444 unreachable
```

TCP/IP Basics: TCP

Transmission Control Protocol

```
$ telnet www.yahoo.com 80
```

TCP/IP Basics: TCP

Transmission Control Protocol

```
12:07:39.100673 IP panix.netmeister.org.60686 > f1.www.vip.re1.yahoo.com.http:
    S 3745952440:3745952440(0) win 32768 <mss 1460,nop,wscale 0,
      sackOK,nop,nop,nop,nop,timestamp 0 0>
12:07:39.115932 IP f1.www.vip.re1.yahoo.com.http > panix.netmeister.org.60686:
    S 3249386671:3249386671(0) ack 3745952441 win 65535 <mss
      1460,nop,wscale 1,nop,nop,timestamp 77355732 0,sackOK,eol>
12:07:39.115974 IP panix.netmeister.org.60686 > f1.www.vip.re1.yahoo.com.http:
    . ack 1 win 33580 <nop,nop,timestamp 0 77355732>
12:07:42.242725 IP panix.netmeister.org.60686 > f1.www.vip.re1.yahoo.com.http:
    P 1:17(16) ack 1 win 33580 <nop,nop,timestamp 7 77355732>
12:07:42.354215 IP f1.www.vip.re1.yahoo.com.http > panix.netmeister.org.60686:
    . ack 17 win 33296 <nop,nop,timestamp 77358905 7>
```

TCP/IP Basics: UDP

User Datagram Protocol

```
$ nslookup www.yahoo.com
```

```
Server:          166.84.67.2
```

```
Address:         166.84.67.2#53
```

```
Non-authoritative answer:
```

```
www.yahoo.com    canonical name = www.wa1.b.yahoo.com.
```

```
www.wa1.b.yahoo.com    canonical name = www-real.wa1.b.yahoo.com.
```

```
Name:   www-real.wa1.b.yahoo.com
```

```
Address: 69.147.76.15
```

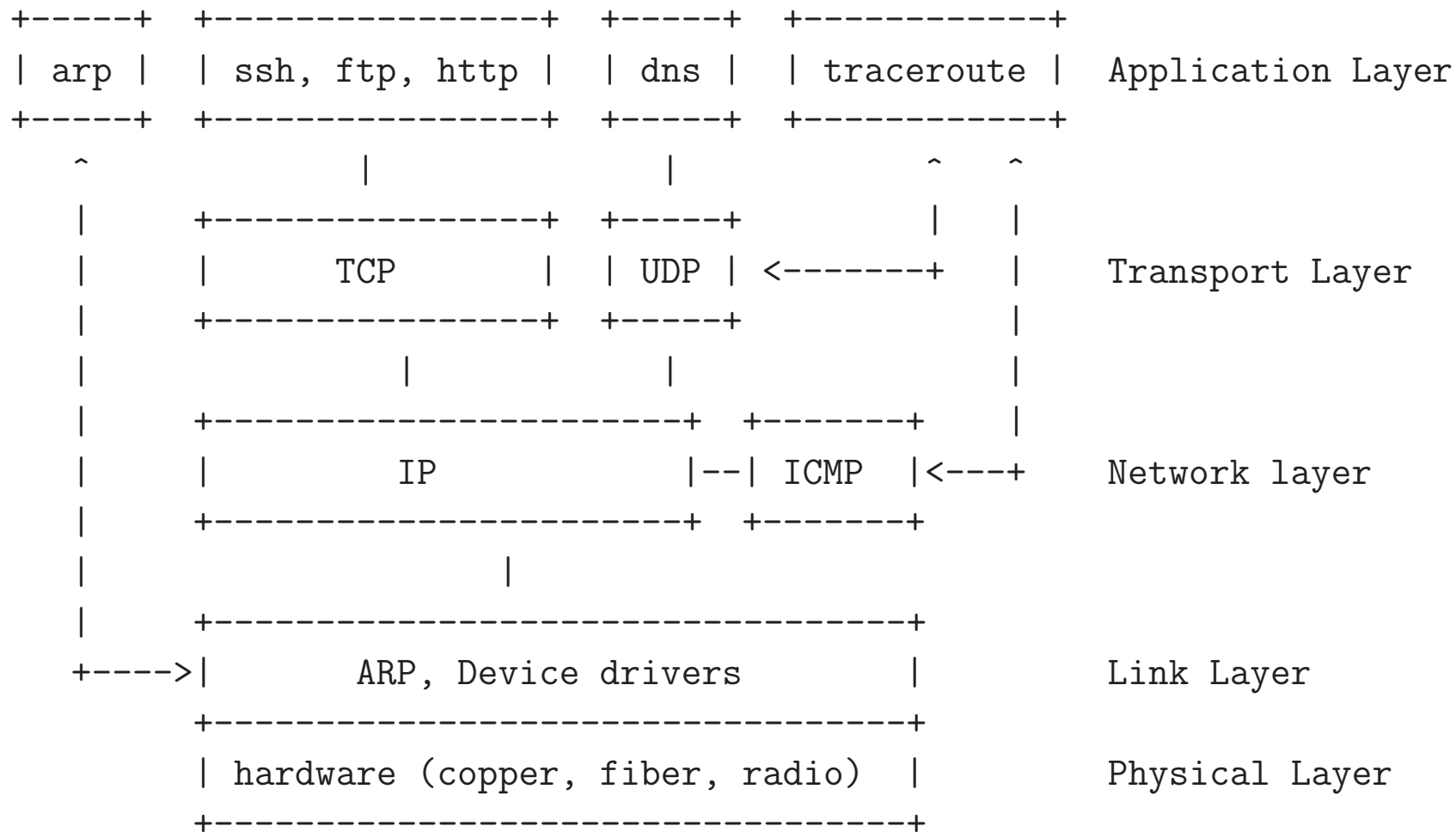
```
$
```

TCP/IP Basics: UDP

User Datagram Protocol

```
12:07:39.072263 IP panix.netmeister.org.53243 > cache2.ns.access.net.domain:
                59127+ AAAA? www.yahoo.com. (31)
12:07:39.082518 IP cache2.ns.access.net.domain > panix.netmeister.org.53243:
                59127 2/1/0 CNAME[|domain]
12:07:39.082744 IP panix.netmeister.org.53242 > cache2.ns.access.net.domain:
                59128+ A? www.yahoo.com. (31)
12:07:39.100074 IP cache2.ns.access.net.domain > panix.netmeister.org.53242:
                59128 3/2/1 CNAME[|domain]
```

TCP/IP Basics: Putting it all together



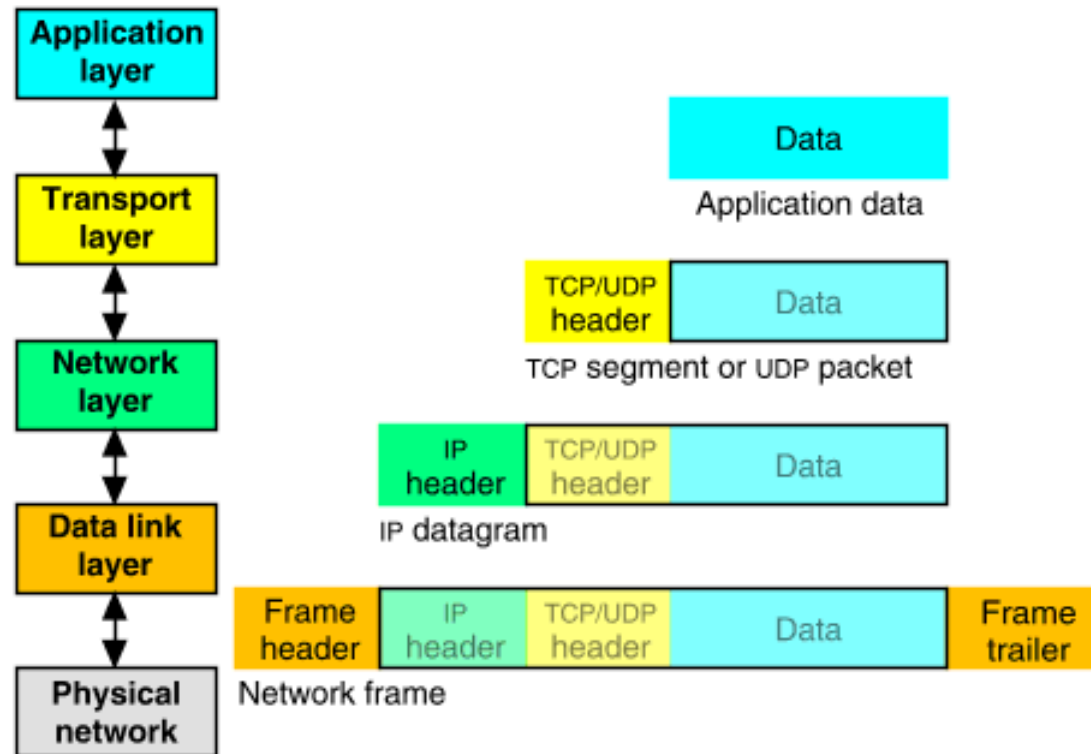
TCP/IP Basics: Data Encapsulation



TCP/IP Basics: Data Encapsulation



TCP/IP Basics: Data Encapsulation



IP Basics

10011011111101100101100100010110

IPv4 addresses are 32-bit numbers

IP Basics

1001101111110110 0101100100010110

IPv4 addresses are divided into a *network part* and a *host part*.

IP Basics

10 01101111110110 0101100100010110

There are three different *classes* of IPv4 networks.

IP Basics

10011011 11110110 01011001 00010110

Each IPv4 address consists of four octets.

IP Basics

10011011 11110110 01011001 00010110

155 . 246 . 89 . 22

Each IPv4 address consists of four octets.

Subnets

```
10011011  11110110  01011001  00010110
11111111  11111111  00000000  00000000
```

A *netmask* splits the IPv4 address into *network* and *host* parts.

Subnets

```
10011011  11110110  01011001  00010110
11111111  11111111  11111111  00000000
```

A *netmask* splits the IPv4 address into *network* and *host* parts.

Subnets

```
$ ipcalc -n 155.246.89.22/16
```

```
Address: 155.246.89.22      10011011.11110110. 01011001.00010110
Netmask: 255.255.0.0 = 16  11111111.11111111. 00000000.00000000
Wildcard: 0.0.255.255     00000000.00000000. 11111111.11111111
=>
Network: 155.246.0.0/16    10011011.11110110. 00000000.00000000
HostMin: 155.246.0.1      10011011.11110110. 00000000.00000001
HostMax: 155.246.255.254  10011011.11110110. 11111111.11111110
Broadcast: 155.246.255.255 10011011.11110110. 11111111.11111111
Hosts/Net: 65534          Class B
```

Subnets

```
$ ipcalc -n 155.246.89.22/24
Address: 155.246.89.22      10011011.11110110.01011001. 00010110
Netmask: 255.255.255.0 = 24 11111111.11111111.11111111. 00000000
Wildcard: 0.0.0.255        00000000.00000000.00000000. 11111111
=>
Network: 155.246.89.0/24   10011011.11110110.01011001. 00000000
HostMin: 155.246.89.1     10011011.11110110.01011001. 00000001
HostMax: 155.246.89.254   10011011.11110110.01011001. 11111110
Broadcast: 155.246.89.255 10011011.11110110.01011001. 11111111
Hosts/Net: 254             Class B
```

IPv6

```
001000000000000001
0000010011111000
000000000000000100
000000000000000111
0000001011100000
1000000111111111
1111111001010010
1001101001101011
```

IPv6 addresses are 128 bits.

IPv6

2001:04f8:0004:0007:02e0:81ff:fe52:9a6b

IPv6 addresses are 128 bits.

IPv6

2001:4f8:4:7:2e0:81ff:fe52:9a6b

IPv6 addresses are 128 bits.

IPv6

:::1

IPv6 addresses can be compressed.

IPv6

IPv6 accounts for less than 1% of internet traffic in any given country...

IPv6

...even though we'll likely run out of IPv4 addresses in early 2010.

Routing

How do I get from here to my destination?

Routing Table

```
$ netstat -nr
Routing tables
```

```
Internet:
```

Destination	Gateway	Flags	Refs	Use	Mtu	Interface
default	155.246.89.1	UGS	14	12406507	-	epic0
127/8	127.0.0.1	UGRS	0	0	33196	lo0
127.0.0.1	127.0.0.1	UH	16	3247590	33196	lo0
155.246.89/24	link#2	UC	13	0	-	epic0
155.246.89.1	00:09:44:d1:64:00	UHLc	1	0	-	epic0
155.246.89.20	00:1e:68:2f:40:da	UHLc	3	111293857	-	epic0
155.246.89.22	00:e0:29:6c:86:8b	UHLc	0	177	-	lo0
155.246.89.41	00:50:45:5f:1c:d4	UHLc	0	294455	-	epic0
155.246.89.68	link#2	UHLc	2	2383	-	epic0
155.246.89.69	00:04:76:3b:66:b6	UHLc	1	50	-	epic0
155.246.89.153	link#2	UHLc	0	8	-	epic0

IPv6 Routing Table

```

Internet6:
Destination          Gateway              Flags    Refs      Use    Mtu  Interface
::/104               ::1                  UGRS     0         0      -   lo0 =>
::/96                ::1                  UGRS     0         0      -   lo0
::1                  ::1                  UH       13        2368  33192 lo0
::127.0.0.0/104     ::1                  UGRS     0         0      -   lo0
::224.0.0.0/100     ::1                  UGRS     0         0      -   lo0
::255.0.0.0/104     ::1                  UGRS     0         0      -   lo0
::ffff:0.0.0.0/96   ::1                  UGRS     0         0      -   lo0
2001:db8::/32       ::1                  UGRS     0         0      -   lo0
2002::/24           ::1                  UGRS     0         0      -   lo0
2002:7f00::/24      ::1                  UGRS     0         0      -   lo0
2002:e000::/20      ::1                  UGRS     0         0      -   lo0
2002:ff00::/24      ::1                  UGRS     0         0      -   lo0
fe80::/10           ::1                  UGRS     0         0      -   lo0
fe80::%lo0/64       fe80::1%lo0         U        0         0      -   lo0
fe80::1%lo0         link#1              UHL     0         0      -   lo0
fe80::%xennet0/64   link#2              UC      0         0      -   xennet0
fe80::e276:63ff:fe72:3900%xennet0 e0:76:63:72:39:00 UHL     0         0      -   lo0
ff01:1::/32        ::1                  UC      0         0      -   lo0
ff01:2::/32        link#2              UC      0         0      -   xennet0
ff02::%lo0/32      ::1                  UC      0         0      -   lo0
ff02::%xennet0/32   link#2              UC      0         0      -   xennet0

```

Routing

If no mapping found, broadcast request to every host on subnet.

Networking Gear: Hub



Networking Gear: Hub



Networking Gear: Switch



Networking Gear: Switch



Networking Gear: Router



Networking Gear: Router



Networking Gear: Router



Integrating a machine into your network

Collect some information:

- IP address
- subnet mask
- broadcast address
- routing information
- DNS server
- hostname

Practical Example

The network configuration is defined in a set of text configuration files.

- `/etc/rc.conf` – `rc.conf(5)` specifies system services, including the network services, to be automatically started at system initialisation.
- `/etc/hosts` – `hosts(5)` the most basic hostname/IP map.
- `/etc/myname` – the fully qualified hostname (unless set via `hostname` in `/etc/rc.conf`).
- `/etc/mygate` – default gateway. Usually better put as *defaultrouter* in `/etc/rc.conf`, or you can run `routed(8)`.
- `/etc/ifconfig.{IF}`. – the definition of network interface `IF`, used by `/etc/rc.d/network` at system initialisation, to configure a network interface. See `ifconfig.if(5)`. An alternative is setting `ifconfig_{IF}="..."` in `/etc/rc.conf`.
- `/etc/nsswitch.conf` – name-service switch configuration file

Practical Example

- `/etc/ifaliases` – `ifaliases(5)` a single interface can be aliased to more than one IP number. The aliases are specified here, and used by `/etc/rc.d/network` at system initialisation.
- `/etc/resolv.conf` – `resolv.conf(5)` specifies how to resolve IP numbers to their hostnames. Most commonly, one or more IPs of the nameservers to query:

```
search subdomain.yourdomain.tld yourdomain.tld
nameserver 192.168.253.10
nameserver 192.168.253.11
```

Beware, this file is overwritten by `dhclient(8)`!

Reading

- tcpdump(8)
- ktrace(1)
- various RFCs
- route(8)
- tcp(4)/ip(4)
- netstat(1)