How Packets Work

BUILD INTUTION 3

PACKETS ARE WORDS 3
READING THE MAP 3
FAILURE ANALYSIS 4
BITS ARE ANTS 4
BITS ARE WAVES ON A STRING 5
BITS ARE WAVES AT A FOOTBALL GAME 5
PACKETS ARE WORMS 5
PACKETS ARE TENNIS BALLS 5
Hubs 5
Cascaded Hubs 6
Bridges 6
Switches 6

ADDRESSES 7

PHYSICAL LAYER ADDRESSES 7
Introduction 7
Bridges 7
Switches 8
Broadcast 8
PROTOCOL LAYER ADDRESSES 8
Domain Name Servers (DNS) 8
IP addressing 8
Address Resolution Protocol (ARP) 8
IP Subnetting 9
Default Router 9
Sample Packets 9
Routing 9
Gateway-of-last-resort 11
Internet Routing 12

PERFORMANCE ANALYSIS 13

PACKET TRANSPORT FACTORS 13
Latency 13
Bandwidth 14
Contention 14
Collisions 14
Duplex 15
BOXES-WITH-BLINKING-LIGHTS – THE LINGO 15
Repeaters 15
Bridges 15

Routers *15*
Switches *16*
Layer 3 Switches *16*
Non-Blocking *16*
Wire-Speed *16*

NETWORK DESIGN 101 ..... **18**

ONE SERVER, FIFTY WORKSTATIONS 18
TWO SERVERS, FIFTY WORKSTATIONS 18
100MB HUBS VS 10MB SWITCHES 18
HUBS VS SWITCHES 19

REFERENCE **20**

OVERVIEW OF ETHERNET 20
PROPAGATION DELAY AND ITS RELATIONSHIP TO MAXIMUM CABLE LENGTH 20
ISSUES IN LAN SWITCHING AND MIGRATION FROM A SHARED LAN ENVIRONMENT 20
PERFORMANCE ANALYSIS 20
INTRODUCTION TO TCP-IP 21
NOTES 21

APPENDIX **22**

DHCP AND ARP PACKETS 22

BUILD INTUTION

In this section, we build physical and mental images of how bits and bytes and packets move around networks.

Packets are Words

How do computers talk to one another? I'm talking to you by throwing sentences, composed of words, at you. Your ears gather those words and forward them to your brains, which reassemble the words into sentences and which then derive meaning from what I've said. Computers do the same things – they conceive of an idea, they create sentences, fragment those sentences into packets, and throw them across wires to one another.

One of the driving limitations in computer communications is one that limits human communication — propagation volume. I can't shout loud enough to be heard on the next floor, let alone in a neighboring building, and neither can computers. Thus, we install boxes-with-blinking lights in the wiring closets on each floor; these boxes regenerate the electronic signal. The magic distance in the packet infrastructure business is 100 meters. Standard electronics using standard copper wire can't shout more than a 100 meters.

So, the boxes-with-blinking lights on each floor are signal regenerators – "repeaters", to use the industry lingo, also called "hubs" or "concentrators". When they receive a packet on one of their ports, they retransmit it on all their remaining ports, regenerating the signal and propagating it to everyone on the floor. We also put a box-with-blinking lights in the basement of each building. This box tends to be a fancier piece of machinery; in addition to regenerating signal, it forwards packets intelligently, sending them around the campus to their appropriate destinations, without needing to copy the packet to every single port. The industry calls these boxes "routers".

I visualize these packets as drops of water flowing through capillary-sized pipes at the speed of light. When a drop reaches a hub, the hub magically replicates the drop and blows a copy out every port.

Reading the Map

Let's look at the map. Starting with a bird's eye view, this map portrays the CNS-owned portions of the voice, video, and data transport infrastructure at the Hutch. We will focus on the bottom two panels, which describe the data infrastructure at the three primary campuses. Starting from the left, notice our locations: First Hill, Met Park, the SELU Outbuildings, Phase II, and Phase I.

The tall rectangles are routers, those boxes-with-blinking lights that sit in the basements of buildings. Ovals are repeaters, aka hubs, and the boxes-with-blinking lights that sit on each floor, boosting signals.

Let's trace the path which packets follow, using a PC on the first floor of the Phase I "A" Building talking to Spot as our example.

{Trace path of packets}

Let's trace the path packets take from a PC on the EPI token-ring to Spot.

Now, you know how to read the map.

Failure Analysis

Let's do some failure analysis on this particular path. How many packet infrastructure failure points are there? A1-hub, cf-rtr, cfbb-esx, df-b-rtr, df-esx for the first example; epi1, mp2t-hub, mp2t-rtr, mpbb-hub, mp-brg, {AT&T Local Services}, cf-brg, cfbb-esx, df-b-rtr, df-esx for the second example.

Bits are Ants

A minimum sized ethernet packet is 64 bytes or 512 bits long (8 x 64 = 512). Let's pretend every bit is an ant and that all these ants form a long line. A PC transmits this packet and as a result we can imagine 512 ants marching in a long line out from the PC's NIC and along a wire. At some point, they reach a hub in the closet. The first few ants enter the hub, vanish, and then start marching out the port on the other side of the hub. (Actually, the hub magically clones each ant and a clone of the first ant – followed by a clone of every other ant – pops out every single port on the hub, other than the original port which is receiving the line of ants. However, this cloning makes the visualization hard, so let's ignore it.) Thus, the hub introduces a latency of several ants – a trivial effect. (I do not know the precise number; I expect this number varies by manufacturer and model.)

Compare this behavior to that of a bridge. The line of ants leaves the PC, walk along the wire, and reach the bridge. They enter the bridge, every single one of them … but none have re-appeared out the other side. All the ants have vanished into the bridge. We wait. Suddenly, the first ant marches forth from the port on the other side of the bridge, followed by the rest of the line. The bridge introduces latency of 64 bytes, or 512 bits, or 512 ants … plus some more time spent thinking. (A traditional router replicates this experience.)

OK, let's move to a switch. Same process as for a bridge, except that as soon as the 512th ant has vanished into the switch, the first ant pops out the other side. The switch has introduced exactly 64 bytes, or 512 bits, or 512 ants' worth of delay, and nothing more. (Modern routers – aka Layer 3 switches – behave like this, too.) In general, the switch does not need to clone the ants and emit copies from every single port; the line of ants proceeds from the source port to the destination port without any cloning.

Bits are Waves on a String

Two people hold a long string between them. One person flicks a wrist. A wave propagates from this person along the string to the other person. We could think of a big wave as being a "1" in binary and a little wave as being a "0" in binary. Compare to Morse Code:
S = …
O = ---
S = …

If a "-" is like a one and a "." is like a zero, then we could send three short waves in succession, pause, then send three long waves in succession, pause, and send three short waves in succession, to produce "SOS".

Bits on a wire are like waves on a string. Though computers don't employ Morse Code; rather, they employ ASCII. Some of the ants are black ("1") and some are white ("0").

Bits are Waves at a Football Game

OK, everyone sits in a row. To emulate a bit, everyone stands up in order, creating a wave. In this visualization, each person is an electron, excited to a higher energy state as the voltage wave passes along the row of seats.

Packets are Worms

On a Vanilla Ethernet (10Mb Ethernet), a bit is twenty meters long. A minimum sized Vanilla Ethernet packet is 64 bytes or 512 bits long. This means that a minimum sized Ethernet packet is over ten thousand meters long! (512 x 20 meters = 10,240 meters ~= 6.2 miles) And a maximum sized ethernet packet (1518 bytes) is 140-150 miles long.

Let's visualize a ten thousand meter worm crawling forth from the NIC in a PC, across a hundred meters of cable to the ethernet hub in the closet, at which point the ethernet hub clones the worm and sends copies crawling out every port. Ten thousand meters is a long way. Every data cable on the floor is occupied with this worm for a long time. The latency introduced by the Ethernet hub is trivial (about 160 meters worth), in the overall picture.

When the worm reaches a bridge, a switch, or a router, all ten thousand meters must coil up inside the device before the head can emerge from another port.

Packets are Tennis Balls

HUBS

Visualize the table as a hub and the chairs sitting around the table as its ports. The people sitting in the chairs are NICs (PCs with NICs). (The cable between the NIC and the port is very short, in this model.) One chair at the table creates a packet – a tennis ball – and rolls it into the hub.

The hub must clone that tennis ball and roll a copy out every single port, into every single chair. Find a hub on the map.

CASCADED HUBS

Visualize two tables surrounded by chairs – each table is a hub, each chair is a port. Visualize a tube running from one chair on one table to another chair on another table – this tube is a cable connecting two hubs. Someone produces a tennis ball, rolls it onto the first table. The table clones the tennis ball and rolls a copy out each port … and one of those copies rolls through the tube connecting the chair of one table to the chair of the next table. Looks like the chairs at the next table receive the ball later than the chairs at the first table. And they do. But realize that here the model does not accurately predict behavior. Packets aren't really compact like balls – they are huge, like giant worms. Basically, everyone receives the packet during the same time window. Chairs at the nearer table may finish receiving the packet slightly before chairs at the far table, but the difference is slight, compared to the overall length of the worm (10,000 meters for the shortest possible worm.) Find a cascaded pair of hubs on the map.

BRIDGES

Visualize a person standing between the two tables, in the middle of the tube. Someone produces a tennis ball, rolls it into the middle of the first table. That table clones the ball and rolls a copy to every single person at that table, plus a copy through the tube. The person standing in the middle of the tube picks up the ball and examines it. This person has memorized who is sitting at what table. If the ball is destined for someone sitting at the far table, the person will roll the ball along the rest of the tube and onto the far table, where that table will clone the ball and roll a copy to each chair. But if the ball is destined for someone sitting at the first table … the person will drop the tennis ball on the floor, preventing that copy from reaching the far table. If the person doesn't know where the intended recipient is sitting… then just to be safe s/he rolls the ball to the far table. Find a bridge on the map.

SWITCHES

In this case the person who ran the bridge jumps into the middle of the table. Every time someone rolls a ball into the middle of the table, the person collects it and consults a list. If the recipient is on that list, the person rolls the ball only to that person, only to the chair holding that person. If the recipient isn't on the list … then the person standing in the middle of the table acts like a hub, cloning the tennis ball and sending a copy to every single chair. Find a switch on the map.

ADDRESSES

In this section we discuss addresses on packets, a concept key to understanding how packets are delivered to the appropriate destinations.

Physical Layer Addresses

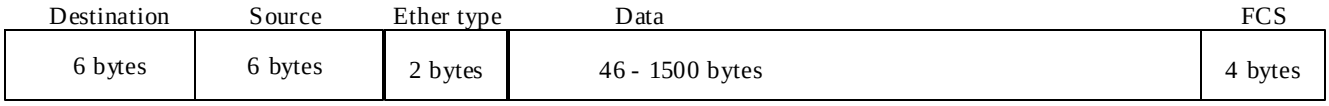
INTRODUCTION

In a simple shared environment, with one hub and a bunch of workstations, the table rolls the tennis ball to every single person. Every single person (NIC) reads the tennis ball. But each ball is addressed to only one of those people. How are balls addressed?

Every packet begins with the recipient's Media Access Control (MAC) or physical layer address, a six byte, unique address burned into a chip on every single NIC. Every person on that hub reads the tennis ball … but only as far as the first six bytes. If the first six bytes match that NIC's MAC address, then the NIC forwards the packet to software running in the OS on its PC. If it doesn't, then the NIC throws the packet away.

Immediately following these first six bytes (the recipient's MAC address) is another six bytes containing the sender's MAC address. This way, the recipient knows who sent the packet and to whom it should respond.

Ethernet Version 2 Frame



Destination = Destination MAC address
Source = Source MAC address
EtherType = Packet type (e.g. 0800 = IP, 0806 = ARP, 08137 = Novell IPX, 809B = AppleTalk)
Data = Packet payload, including upper layer protocol headers
FCS = Frame Check Sequence, aka CRC check on entire frame

BRIDGES

Now we can see how bridges figure out who sits where. When first turned on, they don't know – and must clone every tennis ball they receive. Each time a tennis ball appears at a port, the bridge reads the recipient's MAC address (to see if it knows specifically where to send the tennis ball or if it must clone it across every single port). However, the bridge also reads the **sender's** MAC address. And thus immediately knows where that person lives – in which chair that person sits – and adds this information to the list it keeps. This is called a "learning bridge", and these days all bridges (and their brethren, switches) learn. (I theorize that in the old day's bridges were programmed manually to know who sat where. But this pre-dates me.)

SWITCHES

Switches function just like bridges in this regard.

BROADCAST

The address "FF:FF:FF:FF:FF:FF", (or 11111111 11111111 : 11111111 11111111 : 11111111 11111111 : 11111111 11111111 : 11111111 11111111 : 11111111 11111111 in binary) is the MAC broadcast address. This is a special address. When a NIC reads a packet addressed to the broadcast address – all ones – it forwards the packet to software inside its OS. This address means, "everyone has to read me". There are occasionally times when a machine wants to talk to everyone. When a bridge or a switch receives a broadcast packet, it clones it and sends it to every single port ("flooding" to use the industry lingo). Routers, for reasons we'll see later, throw away broadcast packets.

Protocol Layer Addresses

DOMAIN NAME SERVERS (DNS)

When a user asks for "www.fhcrc.org", his or her PC cannot do anything useful with this. What the PC really wants to know is the MAC address of the NIC in the Web server at SELU. This turns out to be hard to figure out – that NIC is a long way away.

The first thing the PC does is to send a packet to a DNS server asking, "what is the IP address associated with this name?" The DNS server responds by sending the PC an IP address. The PC then employs various strategies, which we are about to discuss, in order to deliver that packet.

IP ADDRESSING

Each IP-aware device on the planet is assigned a unique IP address; this uniqueness is a requirement and allows packets to arrive at one and only one location. IP addresses consist of four fields, ranging from 1 to 254. Here is a typical address at FHCRC: 140.107.10.5. The first two fields "140.107" are owned by FHCRC – by convention, no one else can begin an IP address with these two fields. The second two are controlled internally – it is our responsibility to ensure that we deploy them uniquely.

ADDRESS RESOLUTION PROTOCOL (ARP)

Once the PC has discovered the IP address associated with the name the user has typed (e.g. "www.fhcrc.org"), how does it discover the MAC address of this machine? It needs to know the MAC address because it needs to write this on the very front of the tennis ball that it will roll to the hub. To discover the MAC address, the machine emits an ARP packet, a special kind of packet. See the Frame 27 in the Appendix. ARP packets are addressed to the broadcast address and contain a message saying "My MAC address is ABC, my IP address is XYZ, I want to talk to the person who owns the IP address DEF … but I don't know this person's MAC address. If you own the IP address DEF, please respond to this packet and please provide me with your

MAC address." If the machine "www.fhcrc.org (140.107.52.21)" exists at this table (on this hub, on this segment), then it will respond to the ARP packet, addressing it to DEF (the MAC address of the sender) along with its own MAC address (say, UVW). The sender will now know what the MAC address of the recipient is and will proceed to communicate by creating IP packets containing URLs and sending them to UVW. Everyone is happy.

IP SUBNETTING

This works fine for "local" communication. But in fact, www.fhcrc.org may not exist locally; it may sit far away. Routers throw away broadcast packets. (This is good – if routers forwarded ARP packets … then every machine on the planet would receive every ARP request … this turns out to be a solution which doesn't scale.) What to do?

The sending PC compares its IP address to its subnet mask to discover the segment on which it lives. Knowing this, it can determine whether or not the receiving station is local (on the same segment) or remote (behind a router).

PC's IP Address:	140.107.10.5	Server's IP Address:	140.107.52.21
PC's Subnet Mask:	255.255.255.0	Server's Subnet Mask:	255.255.255.0
PC's Default Router:	140.107.10.1	Server's Default Router:	140.107.52.1

IP addresses allow for hierarchy, because without hierarchy, a solution cannot scale. The IP subnet mask tells the box whether to ARP or to route, whether to ARP for the recipient's MAC address or whether to hand the packet to the default router (pass the buck).

DEFAULT ROUTER

If the recipient is behind a router, the sender doesn't bother to try to deliver the packet. Instead, it addresses the packet to the address of its "default router" (aka "default gateway"). The router receives the packet and is then responsible for delivering the packet. I call this "passing the buck".

SAMPLE PACKETS

Let's look at the DHCP packets in the Appendix. Notice how IP packets include the source and destination MAC addresses at the beginning and the source and destination IP addresses deeper into the packet.

ROUTING

How does the router know where to send the packet? Going back to our map, how does cf-rtr know to which floor to send a packet? Routers must be manually configured to know which interface is mapped to which segment. Here is a snippet from a Cisco router configuration file, a simplified version of cf-rtr, showing the interface which connects to the backbone (segment 10) and the interfaces feeding the floors in the A Building at Phase I.

Cisco Config File
Here is a snippet from cf-rtr's config file.

```
interface Ethernet0
 description Backbone
 ip address 140.107.10.3 255.255.255.0
!
interface Ethernet3
 description A1
 ip address 140.107.16.1 255.255.255.0
!
interface Ethernet4
 description A2
 ip address 140.107.18.1 255.255.255.0
!
interface Ethernet5
 description A3
 ip address 140.107.20.1 255.255.255.0
```

Routers exchange these configured lists with each other via "routing protocols", each building their own "route table", that router's personal view of how to get where. Thus, if a given router cannot itself deliver the packet, it ought to know about a router who can.

When a router has a packet to forward, it compares the packet's destination address with the contents of its routing table and decides which interface to send the packet through.

Cisco Routing Table
Here is an example of a routing table.

```
lv-rtr>show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate
default

Gateway of last resort is 140.107.10.4 to network 0.0.0.0

     140.107.0.0 255.255.255.0 is subnetted, 74 subnets
R       140.107.230.0 [120/1] via 140.107.47.56, 00:00:20, Ethernet6
D       140.107.240.0 [90/281856] via 140.107.10.3, 07:57:54, Ethernet5
D       140.107.204.0 [90/281856] via 140.107.10.3, 07:57:58, Ethernet5
D       140.107.203.0 [90/281856] via 140.107.10.3, 07:57:58, Ethernet5
D       140.107.202.0 [90/281856] via 140.107.10.3, 07:57:58, Ethernet5
D       140.107.201.0 [90/281856] via 140.107.10.3, 07:58:58, Ethernet5
D EX    140.107.133.0 [170/281856] via 140.107.10.14, 00:00:46, Ethernet5
                      [170/281856] via 140.107.44.241, 00:00:46, Ethernet2
D EX    140.107.131.0 [170/281856] via 140.107.10.14, 00:00:46, Ethernet5
                      [170/281856] via 140.107.44.241, 00:00:46, Ethernet2
D       140.107.129.0 [90/284160] via 140.107.10.14, 1d20, Ethernet5
D       140.107.98.0 [90/284160] via 140.107.10.14, 1d20, Ethernet5
D       140.107.97.0 [90/309760] via 140.107.10.14, 1d20, Ethernet5
D       140.107.96.0 [90/284160] via 140.107.10.14, 1d20, Ethernet5
D       140.107.110.0 [90/284160] via 140.107.10.14, 1d20, Ethernet5
D       140.107.108.0 [90/284160] via 140.107.10.14, 1d20, Ethernet5
D EX    140.107.118.0 [170/281856] via 140.107.10.14, 00:00:46, Ethernet5
                      [170/281856] via 140.107.44.241, 00:00:46, Ethernet2
D       140.107.115.0 [90/284160] via 140.107.10.14, 1d20,
C       140.107.47.0 is directly connected, Ethernet6
C       140.107.46.0 is directly connected, Ethernet9
C       140.107.45.0 is directly connected, Ethernet3
C       140.107.44.0 is directly connected, Ethernet2
C       140.107.43.0 is directly connected, Ethernet1
C       140.107.42.0 is directly connected, Ethernet0
D       140.107.41.0 [90/284160] via 140.107.10.14, 1d16, Ethernet5
C       140.107.40.0 is directly connected, Ethernet8
D       140.107.54.0 [90/284160] via 140.107.10.19, 13:22:25, Ethernet5
D       140.107.52.0 [90/284160] via 140.107.10.19, 13:22:25, Ethernet5
```

GATEWAY-OF-LAST-RESORT

Routers themselves have a concept of a “default router” – a pass-the-buck kind of concept. Routers call this particularly knowledgeable sibling the “gateway-of-last-resort”. If a router stares at its routing table and cannot figure out where to forward a packet, then it forwards the packet to the gateway-of-last-resort. In the Hutch environment, this is slushy-rtr, the router sitting on the edge of our DMZ. Slushy-rtr knows how to forward packets to the various entities that connect to our DMZ – Ovid, CIS, SHMC, and CHMC. If the packet isn’t headed to any of

those destinations, then slushy-rtr engages its own “gateway-of-last-resort”. In fact, it has two: internap-a-rtr and internap-b-rtr.

Gateway-of-last-resort Routing Table

```
slushy-rtr>show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate
default
       U - per-user static route, o - ODR

Gateway of last resort is 206.253.199.227 to network 0.0.0.0

S    207.123.155.0/24 [1/0] via 206.253.199.234
S    192.168.1.0/24 [1/0] via 206.253.199.237
S    192.168.18.0/24 [1/0] via 206.253.199.235
{…}
D       140.107.26.0/24 [90/307200] via 140.107.10.3, 3d15h, Ethernet1
D       140.107.24.0/24 [90/307200] via 140.107.10.3, 3d15h, Ethernet1
     206.253.199.0/24 is variably subnetted, 2 subnets, 2 masks
C       206.253.199.224/28 is directly connected, Ethernet0
S    199.98.88.0/24 [1/0] via 206.253.199.231
D*EX 0.0.0.0/0 [170/281856] via 206.253.199.227, 1w4d, Ethernet0
               [170/281856] via 206.253.199.226, 1w4d, Ethernet0
```

INTERNET ROUTING

Once the packet reaches our ISP, the same process continues. Routing behavior within ISPs and across the Internet backbone is beyond the scope of this discussion. And beyond my personal expertise, as well.

PERFORMANCE ANALYSIS

Performance analysis is complex and requires the use of a variety of tools, from a packet analyzer to a process watcher to a source debugger. In the interest of brevity, we grossly oversimplify the problem, focussing on a few sample issues from the packet transport portion of the issue.

Packet Transport Factors

Here are some of the packet transport factors, which affect performance.

LATENCY

The packet infrastructure world uses the term "latency" to refer to the gap between the time the sender sends the packet and the time the receiver receives the packet. Four factors contribute to latency:[1]

Propagation delay	The length of time that information takes to travel the distance of the line. Propagation delay is mostly determined by the speed of light; therefore, the propagation delay factor is not affected by the networking technology in use.
Transmission delay	The length of time a packet takes to cross the given media. The speed of the media and the size of the packet determine transmission delay.
Store-and-forward delay	The length of time an internetworking device (such as a switch, bridge, or router) takes to send a packet that it has received.
Processing delay	The time required by a networking device for looking up the route, changing the header, and other routing/switching tasks.

Propagation delay is driven by the speed of light and is not easily mutable. Transmission delay is driven by the size of the packet (and eventually by the number of packets): users tend to transmit more and more data as time goes on. Store-and-forward delay is driven by the size of the packet. And processing delay goes to zero in modern boxes.

Looking at a typical Ethernet environment, we see that bits travel from the originating PC across copper wire — moving at the speed of light — until they reach the hub in the local wiring closet. Hubs introduce latency — they must read at least a byte of the incoming packet before beginning to retransmit it. This latency is trivial, however, compared to the latency introduced by most

[1] http://www.cisco.com/univercd/cc/td/doc/cisintwk/idg4/nd2013.htm

other packet infrastructure devices (bridges, switches, routers, gateways), which must read the entire packet — 64 to 1518 bytes in length — and then think about it for a while before retransmitting it.

BANDWIDTH

The industry talks about 10 Mb Ethernet — which I call Vanilla Ethernet — 100 Mb Ethernet (Fast Ethernet), and 1000 Mb Ethernet (Gigabit Ethernet). Vanilla Ethernet gear transmits at the rate of ten million bits per second; Fast Ethernet gear transmits at the rate of one hundred million bits per second; and Gigabit Ethernet gear transmits at the rate of a billion bits per second. Thus, in some sense, one can buy ethernet cards and closet electronics, which are "faster" than the Vanilla Ethernet versions we own. I try to avoid the term "faster" because in fact the bits travel at the same speed on Vanilla Ethernet as they do on Fast Ethernet and on Gigabit Ethernet: the speed of light is the speed of light and no vendor, no matter how much marketing muscle they have, can change that reality. On the other hand, the bits are becoming shorter. On a Vanilla Ethernet, a bit is about 20 meters long. On a Fast Ethernet segment, 2 meters long. And on a Gigabit Ethernet segment, 20 centimeters long. As a result, boxes can transmit more information per unit of time on a Gigabit Ethernet than they could on a Vanilla Ethernet. Rather than saying that ethernet is becoming faster, I prefer to say that ethernet is acquiring more bandwidth. On a Vanilla Ethernet, a computer can transmit about one megabyte of actual data per second. On a Fast Ethernet, that number jumps to ten megabytes of actual data. And, on a Gigabit Ethernet, to one hundred megabytes per second.

CONTENTION

The packet transport industry uses the term "contention" to describe the condition in which two or more packets want to be transmitted across a single pipe. In general, pipes in the packet infrastructure business are capillary-sized – they can only carry a single packet a time. If two PCs want to talk simultaneously across the same wire … one of them has to go first, the other has to wait. These two PCs are said to be "contending" for the wire.

In a shared ethernet environment – the environment that dominates the Hutch today – only one packet at a time can be travelling on the entire floor. While that packet is travelling, everyone else has to wait.

COLLISIONS

In a shared ethernet environment, collisions occur when two or more stations attempt to transmit at the same time. This event creates an ethernet "collision", smashing both packets and requiring both stations to retransmit. Typically, the collision process does not cost either station a substantial amount of time, as the entire experience is measured in microseconds. However, under degenerate circumstances – bad electronics or bad cabling or density of transmitting stations – collisions can impact application performance.

DUPLEX

In a Vanilla Ethernet environment, transmission is "half-duplex", meaning that a station is either transmitting or receiving … but it cannot do both at the same time. The first pair of wires in a 10BaseT cable is used to transmit while the second two are used to receive … but in a Vanilla Ethernet environment, the two pairs cannot be employed simultaneously. More recently, the Ethernet community has moved toward "full-duplex" behavior, in which NICs and closet electronics can send and receive simultaneously, sending on the first pair of wires and receiving on the second pair. This behavior requires newer NICs and, more importantly, switches (not hubs) in the closet.

Boxes-with-blinking-lights – The Lingo

REPEATERS

In a shared Ethernet environment — created by repeaters (aka hubs and concentrators) — everyone shares the wire on that floor. One person can talk at a time, period. When a station transmits, the local hub accepts those bits, and after a delay of a byte or so, starts replicating those bits out every single port it owns. Repeaters are fundamentally half-duplex devices – only one station can talk at a time. All the other devices described here come in half-duplex and full-duplex flavors. Repeaters are simple devices that function flawlessly even if their brains fry and typically take less than a second to boot.

BRIDGES

Bridges are little used in the Hutch network, but I'll spend a few moments describing them because their newer cousins – switches – will become popular at the Hutch under NetImp00. Bridges typically contain two ports. They read a packet on one port and make a simple forwarding decision about whether or not to replicate the packet on its other port. Basically, they learn which PCs exist on their left side and on their right side and only forward a packet if it is intended for a PC on their far side. Bridges are used as a simple mechanism for reducing traffic volume in a shared Ethernet environment or for converting between dissimilar media – ethernet to ATM or FDDI, for instance. Bridges require brains in order to function and typically take less than a minute to boot.

ROUTERS

Routers read the entire packet, think about it for a while, and then forward the packet out the appropriate port. Older routers can experience contention on their backplane, but that threshold is far higher than that of a hub, which can't handle more than a single packet at a time. For example, the Cisco AGS+, which sits at the core of the Hutch data net, is rated at 40,000 packets per second, real world traffic (mixed packet sizes), crossing its backplane. The Cisco Catalyst 5000 with Route Switch Modules, which sit in the basements of the Phase II and Met Park buildings, are rated at 170,000 packets per second (mixed packet sizes) across its backplane. The Catalyst 6506s CNS purchased under NetImp99 are rated at six million packets per second. Compare these to the maximum capacity of a 10Mb ethernet segment: 14,400 packets per second

of minimum size packets (64 byte). Routers are complex devices, require brains to function, and require several minutes to boot.

SWITCHES

The term "switch" is a marketing word for a multi-port bridge. Like bridges, they do not (generally) need to retransmit the packet on every single port they own, only on the port that contains the receiving computer. They generally do not spend any time making this forwarding decision – they make the decision while the packet is being read into the device and thus start forwarding as soon as they have received the last bit on the input port. Switches generally have high backplane capacities, reducing the chances of contention there. Switches are complex devices, require brains to function, and require several minutes to boot.

LAYER 3 SWITCHES

Newer routers behave like switches in that they do not introduce processing delay while making forwarding decisions; like switches, they perform this work while the packet is being received and thus start retransmitting the packet as soon as the last bit of the incoming packet has been read. [Furthermore, an entire group of ports can be configured to belong to a single subnet, collectively responding to a single IP address; thus, ports can, under administrative control, rather fluidly look like the Layer 2 ports of a conventional switch or like the Layer 3 ports of a conventional router. For the purposes of this discussion, the key point is the performance doesn't change – the speed with which the box forwards packets using Layer 2 mechanisms is the same speed with which the box forwards packets using Layer 3 mechanisms.] I prefer to refer to Layer 3 switches as "routers" when they are performing Layer 3 functions and as "switches" when they are performing Layer 2 functions.

NON-BLOCKING

The recent popularity of "non-blocking" switches and routers reduces the opportunity for latency introduced by these devices. Non-blocking switches and routers still must accept the entire packet before transmitting it, but their backplanes cannot experience contention — they have sufficient bandwidth to handle full-throttle streams of packets from all their ports simultaneously.

WIRE-SPEED

Vendors may refer to a bridge, switch, or router as a "wire-speed" device, meaning that the device – once it has read the entire packet – can start forwarding the packet without any delay. Unlike a traditional bridge or router – which only starts to think about where to forward a packet after it has read the entire packet – a wire-speed device figures out where to forward the packet while it is reading the packet. Most bridges, switches, and routers sold today are wire-speed devices. I find this term deceptive, because wire-speed devices still incur store-and-forward delay, i.e. they must read the entire packet before beginning to propagate it from another port. Compare this to a hub, which only needs to read the first byte or so of a packet before beginning to propagate. Or, compare this to a piece of wire, which doesn't introduce any store-and-forward

delay whatsoever. Plain old copper wire carries packets much faster than any so-called wire-speed device. To illustrate this, imagine two columns of ants, each identical in length (512 ants long), each marching at the same speed, and each starting at the start line. The flag goes down, the two columns of ants start marching. The first one must pass through a wire-speed device; the second faces no such obstacle. The first column reaches the wire-speed device (visualize a cardboard box) and must enter the box and coil up inside it, waiting for the very last ant to enter before the first one can exit on the other side. Meanwhile, the second column of ants keeps marching forward. After passing the box, the second column of ants will be exactly one column length (512 ants) ahead of the first.

NETWORK DESIGN 101

In this section we design small networks in order to play with the concepts covered in this seminar. In all cases the actual answers to each of the following scenarios are: "It depends on lots of other factors". However, for the purposes of the exercise, we limit the universe to the material we have covered in this seminar, applying what we have learned to develop ways of thinking about network design problems.

One Server, Fifty Workstations

Let's apply these concepts and this lingo to analyzing performance in a simple network design. Let us assume that we have two sites, each identical. Each site contains a bunch of workstations – let's say fifty – plus one server. Nothing else. Let's put a 10Mb hub in the middle of one site and a 10Mb switch in the middle of the other. At which site will users experience snappier application behavior?[2]

Two Servers, Fifty Workstations

OK, let's add a second server to each site. Does this change the result?[3]

100Mb hubs vs 10Mb switches

Now let's swap out the 10Mb hub for a 100Mb hub. Does this change the result?[4]

---

[2] The 10Mb hub site – though, only the most sensitive of user would have a chance of noticing the difference. Surprising? Remember, the file server is receiving and sending all traffic, in this example. Thus, in the switch environment, all packets headed to the file server are queued at the port heading to the file server and must wait their turns, to be spat out one by one on the wire headed to the file server. The switch offers zero benefit over the hub; in fact, it introduces a tiny amount of delay, because it must read in the entire packet before forwarding it, whereas the hub only needs to read in the first byte or so before beginning its forwarding operation. Naturally, the real world, not all traffic would run from the users to the file server and back again – there would likely be at least one printer somewhere, not to mention other resources. Furthermore, the use of full-duplex NICs would change this experience as well, making the switch a more attractive option.

[3] Yup. Now, the switch may well deliver better performance than the hub, as the presence of the switch would allow two simultaneous conversations to occur (workstation A talking to file server A; workstation B talking to file server B).

[4] Nope, doesn't change the game at all, for the purposes of this exercise.

Hubs vs switches

Looking at the current LAN/MAN Map, let us compare the current MAN design with one in which the Xylan boxes – m-brg, mp-brg, and cf-mrg – were replaced with hubs. Which would deliver snappier application behavior?[5]

[5] The hub design, for reason described in the first exercise. In addition, the presence of the spanning tree across the TLS also means that hubs would deliver better overall throughput, as hubs would not need to create a spanning tree.

REFERENCE

Overview of Ethernet

http://wwwhost.ots.utexas.edu/ethernet/ethernet-home.html

Breyer, Robert and Sean Riley. Switched, Fast, and Gigabit Ethernet. Part II Repeating, Bridging, Switching, and Layer 3 Switching. MacMillan Technical Publishing, 1999.

Propagation Delay and its Relationship to Maximum Cable Length

http://www.optimized.com/COMPENDI/EN-Propa.htm

You may know that the minimum frame size in an Ethernet network is 64 bytes or 512 bits, including the 32 bit CRC. You may also know that the maximum length of an Ethernet cable segment is 500 meters for 10BASE5 thick cabling and 185 meters for 10BASE2 thin cabling. It is, however, a much less well-known fact that these two specifications are directly related. In this essay, we will discuss the relationship between minimum frame size and maximum cable length.

Propagation Delay

Before we discuss frame size and cable length, an understanding of signal propagation in copper media is necessary. Electrical signals in a copper wire travel at approximately 2/3 the speed of light. This is referred to as the propagation speed of the signal. Since we know that Ethernet operates at 10Mbps or 10,000,000 bits per second, we can determine that the length of wire that one bit occupies is approximately equal to 20 meters or 60 feet via the following math:

speed of light in a vacuum = 300,000,000 meters/second
speed of electricity in a copper cable = 200,000,000 meters/second
(200,000,000 m/s) / (10,000,000 bits / s) = 20 meters per bit

We can further determine that a minimum size Ethernet frame consisting of 64 bytes or 512 bits will occupy 10,240 meters of cable.

Issues in LAN Switching and Migration from a Shared LAN Environment

http://wwwhost.ots.utexas.edu/ethernet/pdf/techrept14.pdf

Performance Analysis

Nemzow, Martin. Fast Ethernet Implementation and Migration Solutions. Chapter 2 Bandwidth versus Latency. McGraw Hill 1997.

Introduction to TCP-IP

The Association for Computing Machinery's series of introductory articles.
http://www.acm.org/crossroads/xrds1-1/tcpjmy.html

Hill Associates educational series.
http://www.hill.com/library/staffpubs/tcpip.html

ISP Webopedia – collection of links to Web-based TCP-IP tutorials.
http://isp.webopedia.com/TERM/T/TCP_IP.html

TCP/IP Connections (tutorials)
http://aci.mta.ca/Tutorials/TCPIP/Pages/IPaddrs.html

Notes

Number packets sequentially, ack receipt. Implement flow control (credits, windows).
Connection-oriented vs connection-less.

Application	Application
Transport	Transport
Internetworking	Internet
Subnetworks	Network

APPENDIX

DHCP and ARP Packets

```
- - - - - - - - - - - - - - - - - - Frame 1 - - - - - - - - - - - - - - - - - -


DLC:  ----- DLC Header -----
DLC:
DLC:  Frame 1 arrived at  15:45:30.3602; frame size is 342 (0156 hex) bytes.
DLC:  Destination = BROADCAST FFFFFFFFFFFF, Broadcast
DLC:  Source      = Station 3Com  703759
DLC:  Ethertype   = 0800 (IP)
DLC:
IP: ----- IP Header -----
IP:
IP: Version = 4, header length = 20 bytes
IP: Type of service = 00
IP:       000. .... = routine
IP:       ...0 .... = normal delay
IP:       .... 0... = normal throughput
IP:       .... .0.. = normal reliability
IP: Total length    = 328 bytes
IP: Identification  = 0
IP: Flags           = 0X
IP:       .0.. .... = may fragment
IP:       ..0. .... = last fragment
IP: Fragment offset = 0 bytes
IP: Time to live    = 128 seconds/hops
IP: Protocol        = 17 (UDP)
IP: Header checksum = 39A6 (correct)
IP: Source address      = [0.0.0.0]
IP: Destination address = [255.255.255.255]
IP: No options
IP:
UDP: ----- UDP Header -----
UDP:
UDP: Source port      = 68 (Bootpc/DHCP)
UDP: Destination port = 67 (Bootps/DHCP)
UDP: Length           = 308
UDP: Checksum         = 9464 (correct)
UDP: [300 byte(s) of data]
UDP:
DHCP: ----- DHCP Header -----
DHCP:
DHCP: Boot record type        = 1 (Request)
DHCP: Message type            = 1 DHCP Discover
DHCP: Hardware address type   = 1 (10Mb Ethernet)
DHCP: Hardware address length = 6 bytes
DHCP:
DHCP: Hops                    = 0
DHCP: Transaction id          = C301C301
DHCP: Elapsed boot time       = 0 seconds
DHCP: Flags                   = 0000
```

```
DHCP:    0... ....  .... .... = No broadcast
DHCP: Client self-assigned IP address   = [0.0.0.0]
DHCP: Client IP address                 = [0.0.0.0]
DHCP: Next Server to use in bootstrap   = [0.0.0.0]
DHCP: Relay Agent                       = [0.0.0.0]
DHCP: Client hardware address           = 3Com  703759
DHCP:
DHCP: Host name      = ""
DHCP: Boot file name = ""
DHCP:
DHCP: Vendor Information tag = 63825363
DHCP: Message Type                  = 1 (DHCP Discover)
DHCP: Client identifier             = 010020AF703759
DHCP: HostName                      = "skend98"
DHCP: Parameter Request List: 8 entries
DHCP:   Request option code = 1
DHCP:   Request option code = 3
DHCP:   Request option code = 6
DHCP:   Request option code = 15
DHCP:   Request option code = 44
DHCP:   Request option code = 46
DHCP:   Request option code = 47
DHCP:   Request option code = 57
DHCP:

ADDR  HEX                                                ASCII
0000  FF FF FF FF FF FF 00 20  AF 70 37 59 08 00 45 00  ....... .p7Y..E.
0010  01 48 00 00 00 00 80 11  39 A6 00 00 00 00 FF FF  .H......9.......
0020  FF FF 00 44 00 43 01 34  94 64 01 01 06 00 C3 01  ...D.C.4.d......
0030  C3 01 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
0040  00 00 00 00 00 00 00 20  AF 70 37 59 00 00 00 00  ....... .p7Y....
0050  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
0060  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
0070  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
0080  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
0090  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00A0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00B0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00C0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00D0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00E0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00F0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
0100  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
0110  00 00 00 00 00 00 63 82  53 63 35 01 01 3D 07 01  ......c.Sc5..=..
0120  00 20 AF 70 37 59 0C 08  73 6B 65 6E 64 39 38 00  . .p7Y..skend98.
0130  37 08 01 03 06 0F 2C 2E  2F 39 FF 00 00 00 00 00  7.....,./9......
0140  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
0150  00 00 00 00 00 00                                ......
```

```
- - - - - - - - - - - - - - - - - - Frame 3 - - - - - - - - - - - - - - - - - -

DLC:  ----- DLC Header -----
DLC:
DLC:  Frame 3 arrived at  15:45:30.8987; frame size is 349 (015D hex) bytes.
DLC:  Destination = Station 3Com  703759
DLC:  Source      = Station Cisco 00F74F
DLC:  Ethertype   = 0800 (IP)
DLC:
IP: ----- IP Header -----
IP:
IP: Version = 4, header length = 20 bytes
IP: Type of service = 00
IP:       000. .... = routine
IP:       ...0 .... = normal delay
IP:       .... 0... = normal throughput
IP:       .... .0.. = normal reliability
IP: Total length    = 335 bytes
IP: Identification  = 13585
IP: Flags           = 4X
IP:       .1.. .... = don't fragment
IP:       ..0. .... = last fragment
IP: Fragment offset = 0 bytes
IP: Time to live    = 254 seconds/hops
IP: Protocol        = 17 (UDP)
IP: Header checksum = F6B5 (correct)
IP: Source address      = [140.107.10.112]
IP: Destination address = [140.107.44.144]
IP: No options
IP:
UDP: ----- UDP Header -----
UDP:
UDP: Source port      = 67 (Bootps/DHCP)
UDP: Destination port = 68 (Bootpc/DHCP)
UDP: Length           = 315
UDP: Checksum         = 7A69 (correct)
UDP: [307 byte(s) of data]
UDP:
DHCP: ----- DHCP Header -----
DHCP:
DHCP: Boot record type        = 2 (Reply)
DHCP: Message type            = 2 DHCP Offer
DHCP: Hardware address type   = 1 (10Mb Ethernet)
DHCP: Hardware address length = 6 bytes
DHCP:
DHCP: Hops                    = 0
DHCP: Transaction id          = C301C301
DHCP: Elapsed boot time       = 0 seconds
DHCP: Flags                   = 0000
DHCP:    0... ....  .... .... = No broadcast
DHCP: Client self-assigned IP address   = [0.0.0.0]
DHCP: Client IP address                 = [140.107.44.144]
DHCP: Next Server to use in bootstrap   = [140.107.10.112]
DHCP: Relay Agent                       = [140.107.44.1]
DHCP: Client hardware address           = 3Com  703759
DHCP:
DHCP: Host name      = ""
```

```
DHCP: Boot file name = ""
DHCP:
DHCP: Vendor Information tag = 63825363
DHCP: Message Type                  = 2 (DHCP Offer)
DHCP: Server IP address           = [140.107.10.112]
DHCP: Request IP address lease time = 1209600 (seconds)
DHCP: Subnet mask = [255.255.255.0]
DHCP: Gateway address            = [140.107.44.1]
DHCP: Domain Name Server address = [140.107.10.112]
DHCP: Domain Name Server address = [140.107.10.111]
DHCP: Domain Name Server address = [140.107.10.110]
DHCP: Domain name                     = "fhcrc.org"
DHCP: NetBIOS Server address     = [140.107.10.40]
DHCP: NetBIOS Server address     = [140.107.10.50]
DHCP: NetBIOS node type             =2 (P-node)
DHCP:

ADDR  HEX                                                ASCII
0000  00 20 AF 70 37 59 00 00  0C 00 F7 4F 08 00 45 00  . .p7Y.....O..E.
0010  01 4F 35 11 40 00 FE 11  F6 B5 8C 6B 0A 70 8C 6B  .O5.@......k.p.k
0020  2C 90 00 43 00 44 01 3B  7A 69 02 01 06 00 C3 01  ,..C.D.;zi......
0030  C3 01 00 00 00 00 00 00  00 00 8C 6B 2C 90 8C 6B  ...........k,..k
0040  0A 70 8C 6B 2C 01 00 20  AF 70 37 59 00 00 00 00  .p.k,.. .p7Y....
0050  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
0060  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
0070  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
0080  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
0090  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00A0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00B0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00C0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00D0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00E0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00F0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
0100  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
0110  00 00 00 00 00 00 63 82  53 63 35 01 02 36 04 8C  ......c.Sc5..6..
0120  6B 0A 70 33 04 00 12 75  00 01 04 FF FF FF 00 03  k.p3...u........
0130  04 8C 6B 2C 01 06 0C 8C  6B 0A 70 8C 6B 0A 6F 8C  ..k,....k.p.k.o.
0140  6B 0A 6E 0F 0A 66 68 63  72 63 2E 6F 72 67 00 2C  k.n..fhcrc.org.,
0150  08 8C 6B 0A 28 8C 6B 0A  32 2E 01 02 FF           ..k.(.k.2....
```

```
- - - - - - - - - - - - - - - - - - Frame 4 - - - - - - - - - - - - - - - - - -

DLC:  ----- DLC Header -----
DLC:
DLC:  Frame 4 arrived at  15:45:30.8992; frame size is 342 (0156 hex) bytes.
DLC:  Destination = BROADCAST FFFFFFFFFFFF, Broadcast
DLC:  Source      = Station 3Com  703759
DLC:  Ethertype   = 0800 (IP)
DLC:
IP: ----- IP Header -----
IP:
IP: Version = 4, header length = 20 bytes
IP: Type of service = 00
IP:       000. .... = routine
IP:       ...0 .... = normal delay
IP:       .... 0... = normal throughput
IP:       .... .0.. = normal reliability
IP: Total length    = 328 bytes
IP: Identification  = 512
IP: Flags           = 0X
IP:       .0.. .... = may fragment
IP:       ..0. .... = last fragment
IP: Fragment offset = 0 bytes
IP: Time to live    = 128 seconds/hops
IP: Protocol        = 17 (UDP)
IP: Header checksum = 37A6 (correct)
IP: Source address      = [0.0.0.0]
IP: Destination address = [255.255.255.255]
IP: No options
IP:
UDP: ----- UDP Header -----
UDP:
UDP: Source port      = 68 (Bootpc/DHCP)
UDP: Destination port = 67 (Bootps/DHCP)
UDP: Length           = 308
UDP: Checksum         = DA84 (correct)
UDP: [300 byte(s) of data]
UDP:
DHCP: ----- DHCP Header -----
DHCP:
DHCP: Boot record type        = 1 (Request)
DHCP: Message type            = 3 DHCP Request
DHCP: Hardware address type   = 1 (10Mb Ethernet)
DHCP: Hardware address length = 6 bytes
DHCP:
DHCP: Hops                    = 0
DHCP: Transaction id          = C301C301
DHCP: Elapsed boot time       = 0 seconds
DHCP: Flags                   = 0000
DHCP:    0... ....  .... .... = No broadcast
DHCP: Client self-assigned IP address   = [0.0.0.0]
DHCP: Client IP address                 = [0.0.0.0]
DHCP: Next Server to use in bootstrap   = [0.0.0.0]
DHCP: Relay Agent                       = [0.0.0.0]
DHCP: Client hardware address           = 3Com  703759
DHCP:
DHCP: Host name      = ""
```

```
DHCP: Boot file name = ""
DHCP:
DHCP: Vendor Information tag = 63825363
DHCP: Message Type                  = 3 (DHCP Request)
DHCP: Client identifier             = 010020AF703759
DHCP: Request specific IP address   = [140.107.44.144]
DHCP: Server IP address           = [140.107.10.112]
DHCP: HostName                      = "skend98"
DHCP: Parameter Request List: 8 entries
DHCP:   Request option code = 1
DHCP:   Request option code = 3
DHCP:   Request option code = 6
DHCP:   Request option code = 15
DHCP:   Request option code = 44
DHCP:   Request option code = 46
DHCP:   Request option code = 47
DHCP:   Request option code = 57
DHCP:

ADDR  HEX                                                ASCII
0000  FF FF FF FF FF FF 00 20  AF 70 37 59 08 00 45 00  ....... .p7Y..E.
0010  01 48 02 00 00 00 80 11  37 A6 00 00 00 00 FF FF  .H......7.......
0020  FF FF 00 44 00 43 01 34  DA 84 01 01 06 00 C3 01  ...D.C.4........
0030  C3 01 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
0040  00 00 00 00 00 00 00 20  AF 70 37 59 00 00 00 00  ....... .p7Y....
0050  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
0060  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
0070  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
0080  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
0090  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00A0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00B0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00C0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00D0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00E0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00F0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
0100  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
0110  00 00 00 00 00 00 63 82  53 63 35 01 03 3D 07 01  ......c.Sc5..=..
0120  00 20 AF 70 37 59 32 04  8C 6B 2C 90 36 04 8C 6B  . .p7Y2..k,.6..k
0130  0A 70 0C 08 73 6B 65 6E  64 39 38 00 37 08 01 03  .p..skend98.7...
0140  06 0F 2C 2E 2F 39 FF 00  00 00 00 00 00 00 00 00  ..,./9..........
0150  00 00 00 00 00 00                                ......
```

```
- - - - - - - - - - - - - - - - - - Frame 5 - - - - - - - - - - - - - - - - - -

DLC:  ----- DLC Header -----
DLC:
DLC:  Frame 5 arrived at  15:45:30.9549; frame size is 349 (015D hex) bytes.
DLC:  Destination = Station 3Com  703759
DLC:  Source      = Station Cisco 00F74F
DLC:  Ethertype   = 0800 (IP)
DLC:
IP: ----- IP Header -----
IP:
IP: Version = 4, header length = 20 bytes
IP: Type of service = 00
IP:       000. .... = routine
IP:       ...0 .... = normal delay
IP:       .... 0... = normal throughput
IP:       .... .0.. = normal reliability
IP: Total length    = 335 bytes
IP: Identification  = 13586
IP: Flags           = 4X
IP:       .1.. .... = don't fragment
IP:       ..0. .... = last fragment
IP: Fragment offset = 0 bytes
IP: Time to live    = 254 seconds/hops
IP: Protocol        = 17 (UDP)
IP: Header checksum = F6B4 (correct)
IP: Source address      = [140.107.10.112]
IP: Destination address = [140.107.44.144]
IP: No options
IP:
UDP: ----- UDP Header -----
UDP:
UDP: Source port      = 67 (Bootps/DHCP)
UDP: Destination port = 68 (Bootpc/DHCP)
UDP: Length           = 315
UDP: Checksum         = 7769 (correct)
UDP: [307 byte(s) of data]
UDP:
DHCP: ----- DHCP Header -----
DHCP:
DHCP: Boot record type        = 2 (Reply)
DHCP: Message type            = 5 DHCP Ack
DHCP: Hardware address type   = 1 (10Mb Ethernet)
DHCP: Hardware address length = 6 bytes
DHCP:
DHCP: Hops                    = 0
DHCP: Transaction id          = C301C301
DHCP: Elapsed boot time       = 0 seconds
DHCP: Flags                   = 0000
DHCP:    0... ....  .... .... = No broadcast
DHCP: Client self-assigned IP address   = [0.0.0.0]
DHCP: Client IP address                 = [140.107.44.144]
DHCP: Next Server to use in bootstrap   = [140.107.10.112]
DHCP: Relay Agent                       = [140.107.44.1]
DHCP: Client hardware address           = 3Com  703759
DHCP:
DHCP: Host name      = ""
```

```
DHCP: Boot file name = ""
DHCP:
DHCP: Vendor Information tag = 63825363
DHCP: Message Type                  = 5 (DHCP Ack)
DHCP: Server IP address           = [140.107.10.112]
DHCP: Request IP address lease time = 1209600 (seconds)
DHCP: Subnet mask = [255.255.255.0]
DHCP: Gateway address            = [140.107.44.1]
DHCP: Domain Name Server address = [140.107.10.112]
DHCP: Domain Name Server address = [140.107.10.111]
DHCP: Domain Name Server address = [140.107.10.110]
DHCP: Domain name                    = "fhcrc.org"
DHCP: NetBIOS Server address     = [140.107.10.40]
DHCP: NetBIOS Server address     = [140.107.10.50]
DHCP: NetBIOS node type             =2 (P-node)
DHCP:

ADDR  HEX                                                ASCII
0000  00 20 AF 70 37 59 00 00  0C 00 F7 4F 08 00 45 00  . .p7Y.....O..E.
0010  01 4F 35 12 40 00 FE 11  F6 B4 8C 6B 0A 70 8C 6B  .O5.@......k.p.k
0020  2C 90 00 43 00 44 01 3B  77 69 02 01 06 00 C3 01  ,..C.D.;wi......
0030  C3 01 00 00 00 00 00 00  00 00 8C 6B 2C 90 8C 6B  ...........k,..k
0040  0A 70 8C 6B 2C 01 00 20  AF 70 37 59 00 00 00 00  .p.k,.. .p7Y....
0050  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
0060  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
0070  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
0080  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
0090  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00A0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00B0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00C0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00D0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00E0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00F0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
0100  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
0110  00 00 00 00 00 00 63 82  53 63 35 01 05 36 04 8C  ......c.Sc5..6..
0120  6B 0A 70 33 04 00 12 75  00 01 04 FF FF FF 00 03  k.p3...u........
0130  04 8C 6B 2C 01 06 0C 8C  6B 0A 70 8C 6B 0A 6F 8C  ..k,....k.p.k.o.
0140  6B 0A 6E 0F 0A 66 68 63  72 63 2E 6F 72 67 00 2C  k.n..fhcrc.org.,
0150  08 8C 6B 0A 28 8C 6B 0A  32 2E 01 02 FF           ..k.(.k.2....
```

```
- - - - - - - - - - - - - - - - - - Frame 27 - - - - - - - - - - - - - - - - - -

DLC:  ----- DLC Header -----
DLC:
DLC:  Frame 27 arrived at  15:45:35.0895; frame size is 60 (003C hex) bytes.
DLC:  Destination = BROADCAST FFFFFFFFFFFF, Broadcast
DLC:  Source      = Station 3Com  703759
DLC:  Ethertype   = 0806 (ARP)
DLC:
ARP: ----- ARP/RARP frame -----
ARP:
ARP: Hardware type = 1 (10Mb Ethernet)
ARP: Protocol type = 0800 (IP)
ARP: Length of hardware address = 6 bytes
ARP: Length of protocol address = 4 bytes
ARP: Opcode 1 (ARP request)
ARP: Sender's hardware address = 3Com  703759
ARP: Sender's protocol address = [140.107.44.144]
ARP: Target hardware address   = 000000000000
ARP: Target protocol address   = [140.107.44.1]
ARP:

ADDR  HEX                                                ASCII
0000  FF FF FF FF FF FF 00 20  AF 70 37 59 08 06 00 01  ....... .p7Y....
0010  08 00 06 04 00 01 00 20  AF 70 37 59 8C 6B 2C 90  ....... .p7Y.k,.
0020  00 00 00 00 00 00 8C 6B  2C 01 01 01 01 01 01 01  .......k,.......
0030  01 01 01 01 01 01 01 01  01 01 01 01              ............



- - - - - - - - - - - - - - - - - - Frame 28 - - - - - - - - - - - - - - - - - -

DLC:  ----- DLC Header -----
DLC:
DLC:  Frame 28 arrived at  15:45:35.0905; frame size is 60 (003C hex) bytes.
DLC:  Destination = Station 3Com  703759
DLC:  Source      = Station Cisco 00F74F
DLC:  Ethertype   = 0806 (ARP)
DLC:
ARP: ----- ARP/RARP frame -----
ARP:
ARP: Hardware type = 1 (10Mb Ethernet)
ARP: Protocol type = 0800 (IP)
ARP: Length of hardware address = 6 bytes
ARP: Length of protocol address = 4 bytes
ARP: Opcode 2 (ARP reply)
ARP: Sender's hardware address = Cisco 00F74F
ARP: Sender's protocol address = [140.107.44.1]
ARP: Target hardware address   = 3Com  703759
ARP: Target protocol address   = [140.107.44.144]
ARP:

ADDR  HEX                                                ASCII
0000  00 20 AF 70 37 59 00 00  0C 00 F7 4F 08 06 00 01  . .p7Y.....O....
0010  08 00 06 04 00 02 00 00  0C 00 F7 4F 8C 6B 2C 01  ...........O.k,.
0020  00 20 AF 70 37 59 8C 6B  2C 90 00 00 00 00 00 00  . .p7Y.k,.......
0030  00 00 00 00 00 00 00 00  00 00 00 00              ............
```