INTERMEDIATE

Ethernet for Professionals

Joe R. Doupnik
Utah State University
jrd@cc.usu.edu

Novell

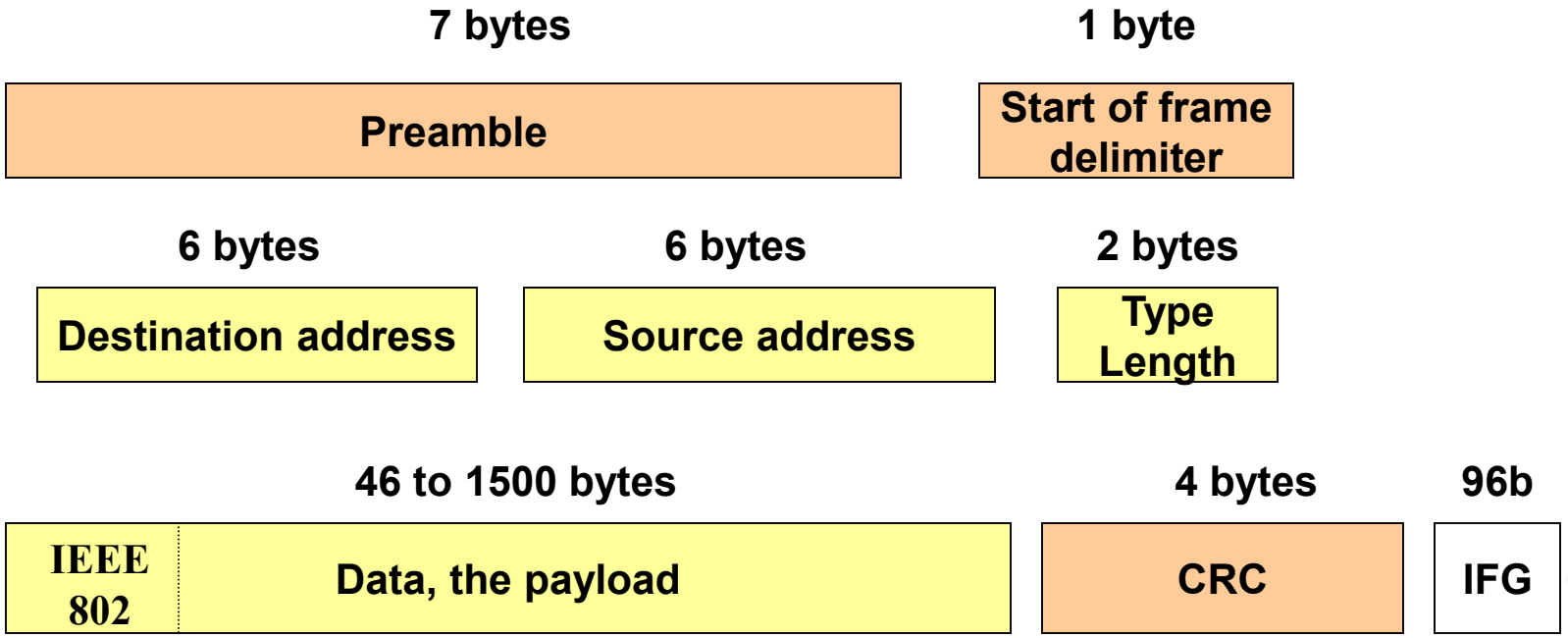
BRAINSHARE '99

Goals of this talk

- Review basics of Ethernet which we have either forgotten or never quite grasped
- Recognize common myths as fiction
- FDX versus HDX
- How to judge when a net is really busy

Ethernet frames, overview

- There are four frames, alas, but their general form looks like this:



Ethernet frames, Preamble

- Preamble bits are to synchronize the receiver's bit recovery electronics (the MAC layer material) to the precise bit rate of the transmitter
- The sync process involves a phaselock loop, and while synchronizing bits may be lost
- Preamble bit loss imposes limits on repeating frames, hubs in a row style

Ethernet frames, Preamble

- Preamble is a bit pattern of 1010 1010..
- Start of Frame Delimiter is a byte to demark when preamble has finished
- SFD has a bit pattern of 1010 1011
- The Ethernet controller chip becomes interested only after the SFD has been received

Ethernet frames, end of same

- The CRC calculation is run as each byte is received. It stops when the transmitted signal stops
- An Ethernet frame stops when the transmitted signal stops
- Length of a frame is implied by that end of signal condition

Ethernet frames, CRC

- CRC checking compares the calculated value with the received bit pattern at the time the signal stops.
- A CRC error is sent to the controller chip and the damaged frame is normally discarded. Some adapters may retain it for packet snoop programs, most don’t.

Ethernet frames, where none is

- After the end of signal is a 96 bit time period of silence
- This is the Inter Frame Gap
- Purpose is to let collision detection electronics recover from this packet
- No, folks don’t normally cheat by shortening the IFG, but they can and that is a source of urban legend

Ethernet frames versus speeds

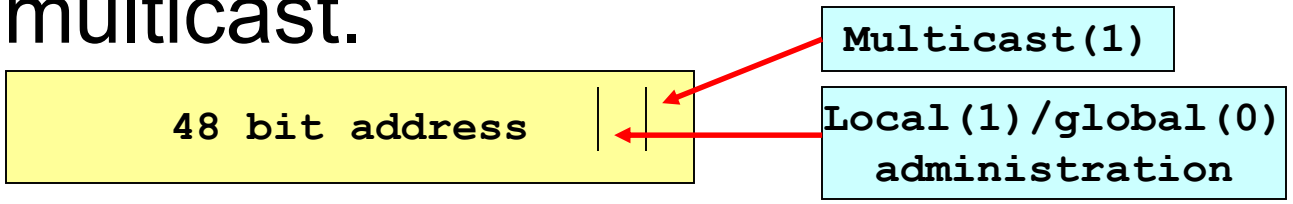
- 1Mbps, 10Mbps, 100Mbps Ethernet frames all look alike, except for clock rate
- 1Gbps Ethernet largely does too
  - Today Gbps Ethernet runs over fiber, full duplex
  - FDX is needed because shortest frames are too small to yield useful path lengths
  - And now Gbps Ethernet over copper has been approved as a standard

Ethernet frame kinds

- Destination and Source addresses are the Medium Access Control (MAC) specific identifications
- Purpose is to distinguish one lan adapter from another on the same broadcast medium
- These addresses are unrecognized off the local wire (but DHCP uses them)

Ethernet frame details

- MAC addresses are 6 bytes, divided into a leading vendor ident half and a trailing serial number half
- Multicast revises addresses to be protocol specific: high byte, least bit is “1” if multicast.



- DECnet revises addresses to be DECnet specific

Novell

Ethernet type/length, DIX

- The “Type/Length” field is the source of much confusion and misery
- In the beginning was Xerox Parc
- It begat Ethernet v1, a low speed (3Mbps) experiment, long since gone
- That soon lead to Ethernet v2, the implementation in use today
- They created “Ethernet” frames with a Type field; that was good

Ethernet type/length, used for

- The Type field is needed to inform the receiver which protocol is being carried in this frame, so the right protocol stack can be awakened to buffer the frame
- The length of an Ethernet frame is known from the controller, by when the signal stops. CRC checking needs this same indication.

Ethernet type/length, used for

- The IEEE 802.3 committee made a muddle of Ethernet while trying to placate IBM's Token Ring standard. TRN does things vastly differently.
- The goal was to create an abstract frame, IEEE 802.2 Logical Link Control form, which could be converted to Ethernet or TRN by translating bridges.

Ethernet type/length

- The committee replaced Type with an explicit Length value, reputedly to allow any protocol to know and discard MAC level padding.
- Recall, Ethernet has a 64B minimum length requirement, TRN does not.
- Reality (not invited to the committee sessions) says protocols know their data's length by internal means.

Ethernet type/length, 802

- So with “Type” replaced by redundant “Length” where will “Type” go?
- Answer: steal regular data bytes and create Ethernet_802.2 header:

But there’s no Type!

- SAP’s are Service Access Points
- SAPs identify a protocol within a host

Ethernet type/length, 802

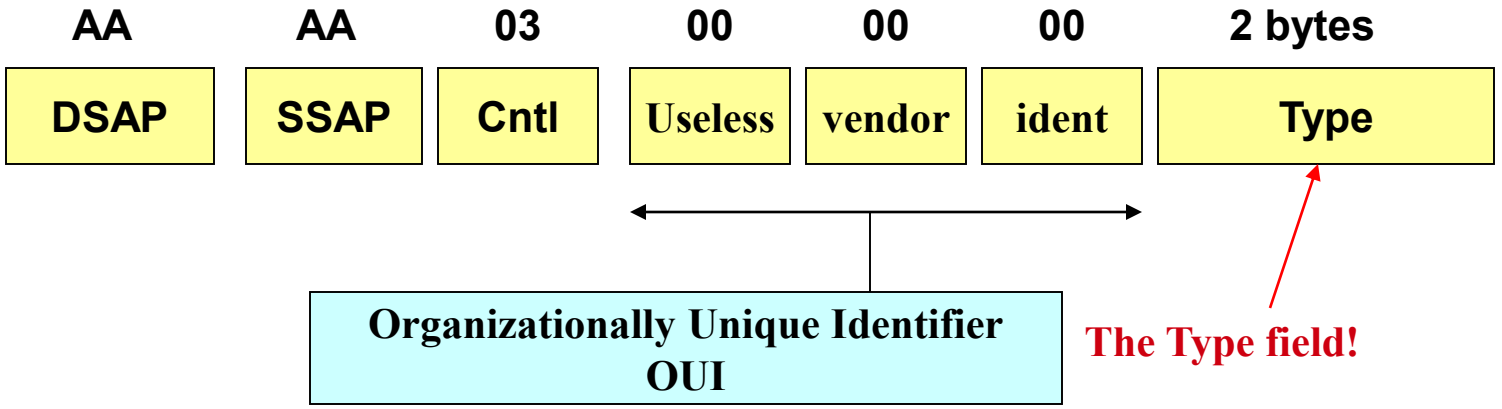
- SAPs come in two flavors: destination DSAP and source SSAP, just in case it's an international meeting, and each is only 8-bits (yields < 127 protocols)
- They are exceedingly difficult to get, so folks don't

Ethernet type/length, SNAP

- Enter the vendor community
- Vendors: we want more SAPs if that's what we are stuck with.
- IEEE: take a hike, and like it
- Vendors: Hmmmm. Give us one SAP value, pretty please
- IEEE: OK humble folks, you have 0xAA

Ethernet Type/length, SNAP

- Vendors: He he he. We fooled them.
- Create Ethernet_SNAP, sub-network access protocol, extended header in the data field



Novell

Ethernet type/length, Novell 802.3

- Alas, the Novell-exclusive Ethernet_802.3 frame missed the meeting and omitted all protocol identification. It is deceased but alive.
- Such frames use Length but no IEEE 802 interior. The IPX header starts where that would go. IPX checksum is in the DSAP and SSAP bytes so we can’t use IPX checksums

Novell

Ethernet type/length, Novell 802.3

- 802.2 framing says a DSAP field of all 1 bits is a broadcast.
- That byte is used by Novell's IPX checksum with Ethernet_802.3
- If the checksum is turned off that IPX field is all 1's
- The next byte, SSAP, is too but all 1's is not legal, thus we can detect old IPX if no checksums are used

Novell

Ethernet type/length, summary

- What can we say about the four types?
- Ethernet_SNAP has limited usage
- Ethernet_802.3 is a dinosaur (T.Rex)
- Ethernet_802.2 does the same job as below but with more work and more bytes (Ethertsm.nlm worries about this)
- Ethernet_II remains the lingua franca and frame of choice
- Tell them apart by Length < 1500 < Type

Ethernet_802.2 frame

Capture Buffer

No.	Source	Destination	Layer	Summary
57	00A0C9784E8D	000000000000	udp	Port:NETBIOS-NS —> NETBIOS-NS
58	0000C0234440	FFFFFFFFFFFF	arp	Req by 129.123.28.16 for 129.123.28.17
59	0000C0317E0A	030000000001	802.2	sap: NetBEUI -> NetBEUI
60	0060B076DFF4	FFFFFFFFFFFF	sap	Resp General; Server=0060B076DFF402
61	0000C0317E0A	00105A29F2B5	802.2	sap: NetBEUI -> NetBEUI
62	0000C0317E0A	00105A29F2B5	802.2	sap: NetBEUI -> NetBEUI
63	00805F2CEC65	080009BE6E55	802.2	sap: 0xC4 -> 0xB4
64	00906DE22400	09002B010001	ether	Protocol=0x8038
65	AA00040059E6	AB0004010101	ether	Protocol=DNALVC

```
Packet Number : 59                4:30:37 PM
Length : 65 bytes
802.3: ==================== IEEE 802.3 Datalink Layer ====================
       Station: 00-00-C0-31-7E-0A ----> 03-00-00-00-00-01
       Length: 47
802.2: ================== IEEE 802.2 Logical Link Control ================
       SSAP: NetBEUI      DSAP: NetBEUI
       Unnumbered Command: Unnumbered Information (UI)
Data:
   0: 2C 00 FF EF 0A 00 00 80 08 00 32 00 41 52 49 45 |,.........2.ARIE
  10: 4C 4C 45 20 20 20 20 20 20 20 20 20 50 48 41 53 |LLE         PHAS
  20: 45 52 49 49 20 20 20 20 20 20 20 00             |ERII        .
```

Length from hw

Multicast

```
   0: 03 00 00 00 00 01 00 00 C0 31 7E 0A 00 2F F0 F0 |.........1~../..
  10: 03 2C 00 FF EF 0A 00 00 00 00 00 32 00 41 52 49 |.,.........2.ARI
  20: 45 4C 4C 45 20 20 20 20 20 20 20 20 20 50 48 41 |ELLE         PHA
  30: 53 45 52 49 49 20 20 20 20 20 20 20 00          |SERII        .
```

Novell

Ethernet_802.3 (old IPX)

No.	Source	Destination	Layer	Summary
13	00A0C9657B56	FFFFFFFFFFFF	rip	Resp network=81 7B 3B 01; 3 hops
14	00A0C9657B56	FFFFFFFFFFFF	rip	Resp network=81 7B 3B 01; 3 hops
15	00A0C9657B56	FFFFFFFFFFFF	rip	Resp network=81 7B 01 1F; 2 hops
16	00A0C9657B56	FFFFFFFFFFFF	rip	Resp network=81 7B 01 1F; 2 hops
17	00A0C9657B56	FFFFFFFFFFFF	rip	Resp network=00 07 22 70; 3 hops
18	00A0C9657B56	FFFFFFFFFFFF	rip	Resp network=00 07 22 70; 3 hops
19	00400543B32A	FFFFFFFFFFFF	sap	Resp General; Server=SLS!!!!!!!!!!!!A
20	00400543B32A	FFFFFFFFFFFF	sap	Resp General; Server=SLS
21	00A0C9657B56	FFFFFFFFFFFF	rip	Resp network=5F 2C EC D2; 4 hop
22	00A0C9657B56	FFFFFFFFFFFF	rip	Resp network=5F 2C EC D2; 4 hop
23	00A0C9657B56	FFFFFFFFFFFF	sap	Resp General; Server=EDU-USU-

```
Packet Number : 15                5:13:48 PM
Length : 450 bytes
802.3: ==================== IEEE 802.3 Datalink Layer ====================
       Station: 00-A0-C9-65-7B-56 ----> FF-FF-FF-FF-FF-FF
       Length: 432
  ipx: ================== Internetwork Packet Exchange ==================
       Checksum: 0xFFFF
       Length: 432
       Hop Count:  0
       Packet Type: 1(RIP)
       Network: 81 7B 01 F0          ---->  81 7B 01 F0

   0: FF FF FF FF FF FF 00 A0 C9 65 7B 56 01 B0 FF FF |.........e{V....
  10: 01 B0 00 01 81 7B 01 F0 FF FF FF FF FF FF 04 53 |.....{.........S
```

Checksum where 802 LLC goes

Novell

Ethernet_SNAP

No.	Source	Destination	Layer	Summary
13	00A0C9657B56	FFFFFFFFFFFF	rip	Resp network=81 7B 3B 01; 3 hops
14	00A0C9657B56	FFFFFFFFFFFF	rip	Resp network=81 7B 3B 01; 3 hops
15	00A0C9657B56	FFFFFFFFFFFF	rip	Resp network=81 7B 01 1F; 2 hops
16	00A0C9657B56	FFFFFFFFFFFF	rip	Resp network=81 7B 01 1F; 2 hops
17	00A0C9657B56	FFFFFFFFFFFF	rip	Resp network=00 07 22 70; 3 hops
18	00A0C9657B56	FFFFFFFFFFFF	rip	Resp network=00 07 22 70; 3 hops
19	00400543B32A	FFFFFFFFFFFF	sap	Resp General; Server=SLS!!!!!!!!!!!A
20	00400543B32A	FFFFFFFFFFFF	sap	Resp General; Server=SLS
21	00A0C9657B56	FFFFFFFFFFFF	rip	Resp network=5F 2C EC D2; 4 hops
22	00A0C9657B56	FFFFFFFFFFFF	rip	Resp network=5F 2C EC D2; 4 hops
23	00A0C9657B56	FFFFFFFFFFFF	sap	Resp General; Server=EDU-USU-N

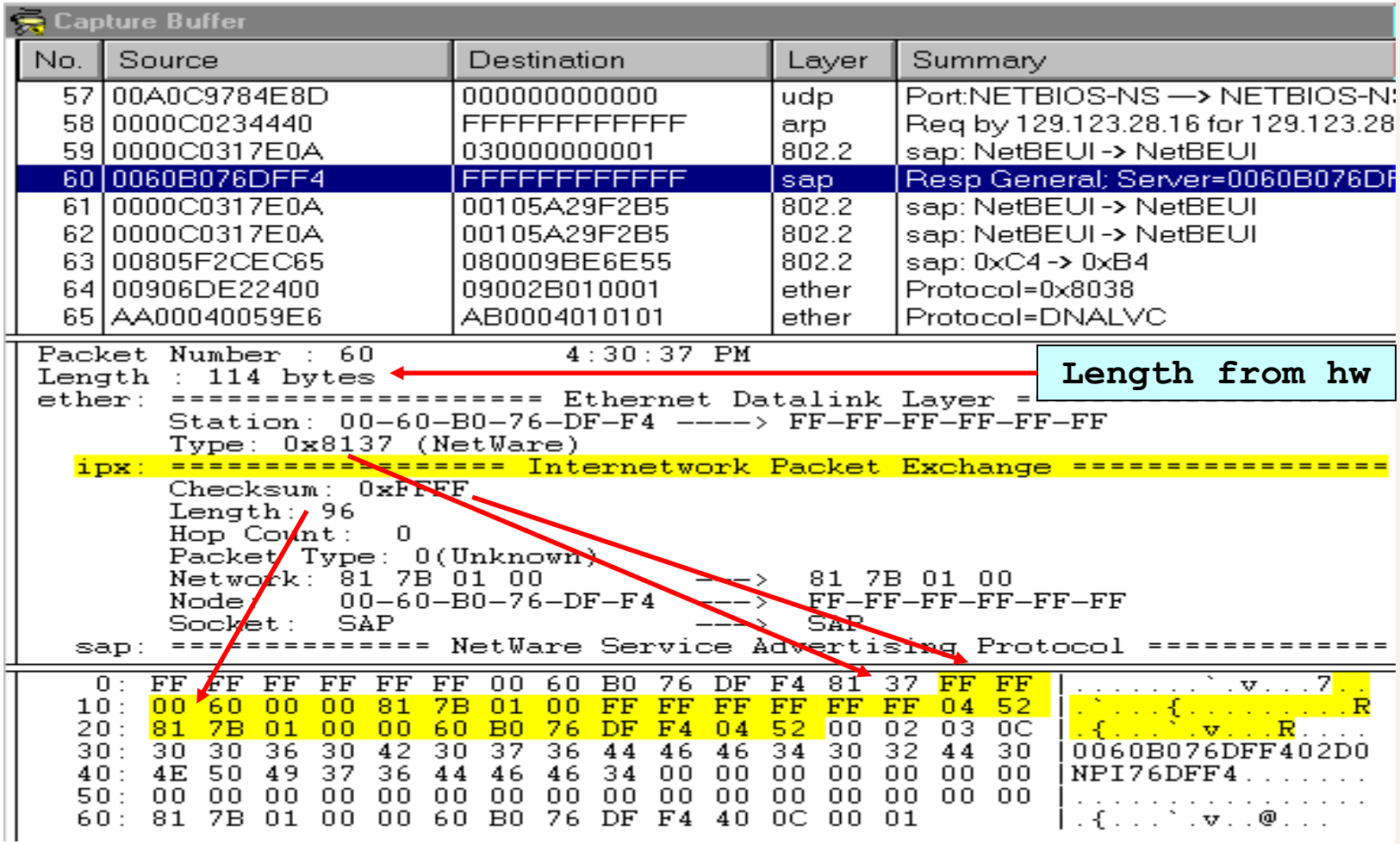
```
Packet Number : 16              5:13:48 PM
Length : 458 bytes
802.3: ==================== IEEE 802.3 Datalink Layer ====================
       Station: 00-A0-C9-65-7B-56 ----> FF-FF-FF-FF-FF-FF
       Length: 440
802.2: ================== IEEE 802.2 Logical Link Control ================
       SSAP: SNAP      DSAP: SNAP
       Unnumbered Command: Unnumbered Information (UI)
       SNAP Organization Code: 00 00 00
       SNAP Protocol Type: 0x8137 (NetWare)
  ipx: ==================== Internetwork Packet Exchange ==================
       Checksum: 0xFFFF
```

```
  0: FF FF FF FF FF FF 00 A0 C9 65 7B 56 01 B8 AA AA |.........e{V....
 10: 03 00 00 00 81 37 FF FF 01 B0 00 01 81 7B 01 F1 |.....7.......{..
 20: FF FF FF FF FF FF 04 53 81 7B 01 F1 00 A0 C9 65 |.......S.{.....e
```

Type

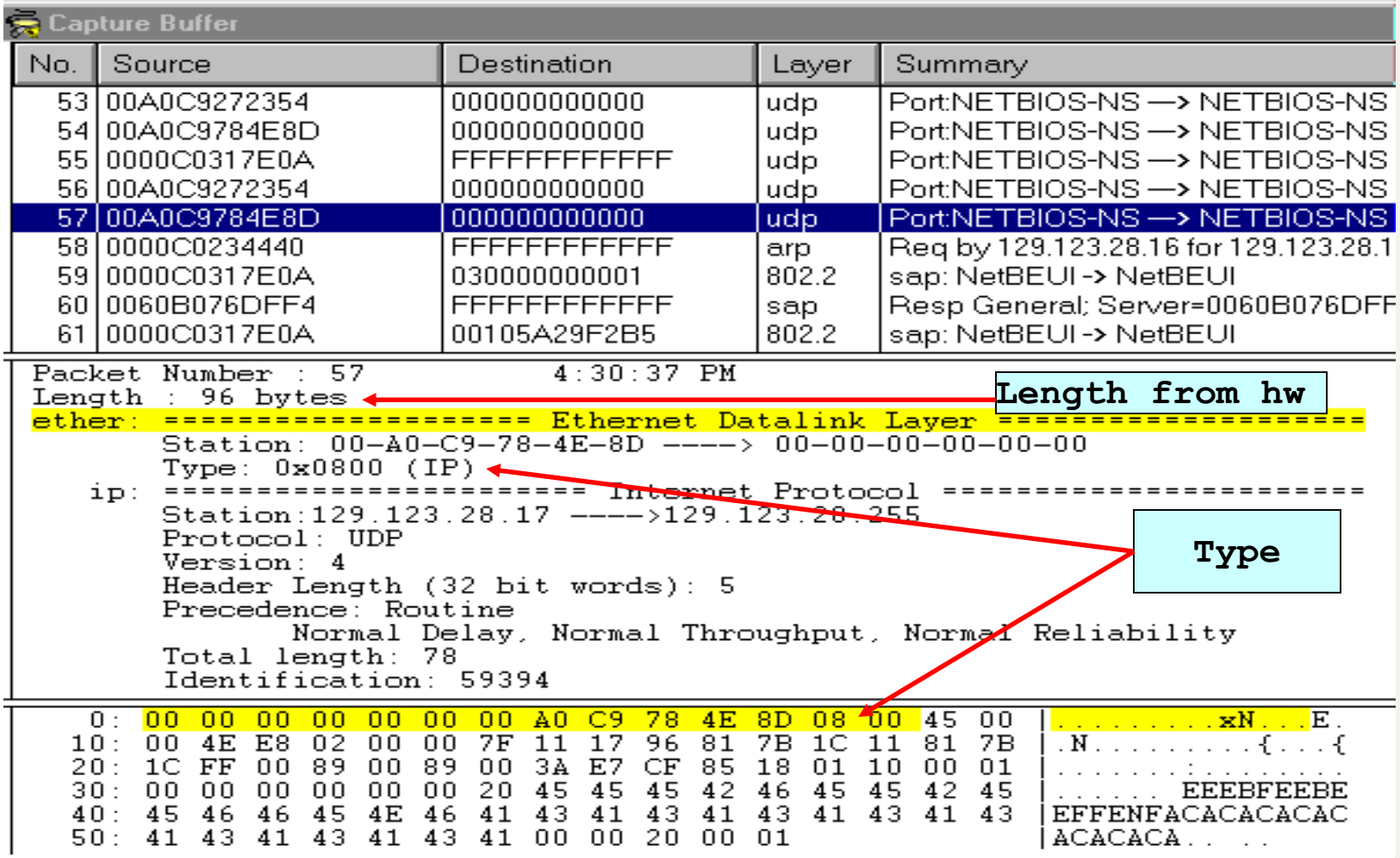
Novell

Ethernet_II (new IPX) frame



Novell

Ethernet_II frame (UDP)



Novell

Aloha experiment

- Radio links from islands to Honolulu computer center
- Idealized to fixed length frames
- Transmission occurs when user wishes without regard to other transmissions
- Collisions (garbles) cause packet loss
- This is NOT Ethernet (kill another urban legend)

Aloha Experiment

- Analysis of pure Aloha (pure anarchy) shows throughput to peak at 18% of capacity
- Slotted Aloha (all transmissions begin on a clock tick) doubles throughput to 36%. Here there be urban legends.
- Aloha does not “sense carrier” (does not listen before/while sending) and thus steps on existing transmissions

Ethernet

- Ethernet is not Aloha; it is CSMA/CD, a different technique
- Carrier sense says do not transmit while another station is doing so
- Politeness greatly increases throughput
- Sensing collisions rapidly terminates the contention interval without full (garbled) packets wasting time as with Aloha
- Ethernet throughput peaks above 90%

Ethernet, back off & retry strategy

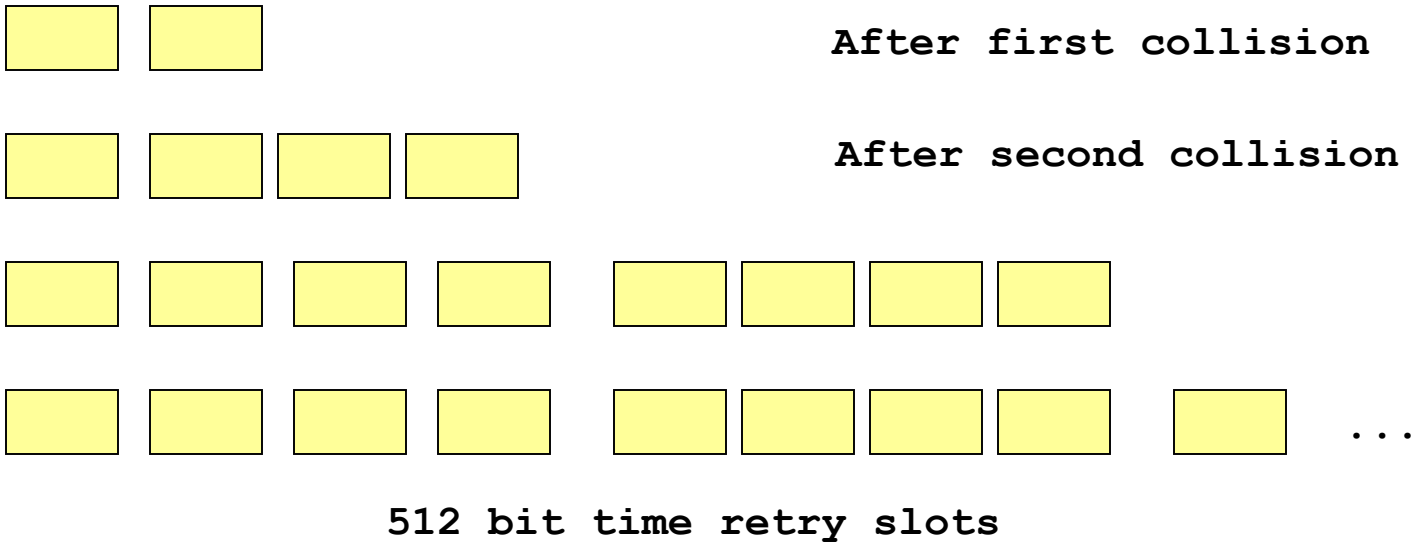
- Wait for wire to become quiet
- Transmit packet
- If collision then emit short jam and try again after random wait
- Random wait: wait for wire to become quiet, then choose transmit or wait one 512 bit slot time. 50-50 chance to wait
- Backing off randomly lets stations sort themselves out

Ethernet, back off & retry strategy

- If suffer a second collision then repeat exercise but choose amongst four slot times; similar doubling on each failure.
- 16 retries per packet are permitted, spread over as many as 1024 slots
- Back off and retry is done by Ethernet controller chip, no software is involved. That's fast (microsecond level).

Ethernet, retry slots

- The back off and retry process is a binary truncated exponential (doubling on each attempt, clipped at 1024 slots)



Novell

Ethernet collisions

- A collision is when two or more stations try to access the comms channel at the same time
- Collisions result in garbles
- Stations listen while transmitting and if a garble occurs they declare a collision and send a short jam gibberish string
- Stations declare a collision if they receive data while they are transmitting

Ethernet collisions, how long

- Time to discover a collision is less than propagation delay from one end of the cable to the far end, and back again
- That is the worst case of two stations talking to an empty wire and one goes first. 512 bits needed for max length net
- 100M cable -> time = 2 * distance / c
- 100M cable -> 1 microsec, 10 or 100 bits which puts most collisions into preamble and MAC header

Ethernet collisions, how long

- Ethernet minimum packet length of 512 bits (64 bytes) is to ensure the worst case of an end of cable to end of cable collision is heard by each transmitter while each is still sending
- 64B = 512 bits -> 5km / 2.5km cable
- Ethernet 64B minimum size is calculated starting at the destination MAC address and continues through the 4 byte CRC check

Novell

Ethernet collisions, late

- If cable is too long then transmitter thinks transmission was successful and discards its frame, yet the frame can be garbled and unusable. A late Collision
- Late collisions are nasty to locate and are often indicative of too many hubs in a row or cable which is too long
- Remember, the collision light is on for much longer than a collision

Ethernet collisions, where

- Near End Cross Talk (NEXT) on twisted pair must be low so a station does not hear its own transmission and think it is another station
- Twisted pair and fiber collisions occur “in the box,” not on the media!
- Coax collisions occur on the coax
- Collisions are normal healthy ways of sharing the common medium

Ethernet collisions, retry limit

- Software timeouts (if available) recover very very slowly compared to Ethernet controller retries
- Max collision rate before packet loss is 1600% (16 retries)
- One or two retries per packet is barely noticeable, yet that is a 100%-200% collision rate
- Exit urban legend of 10-30% util is full

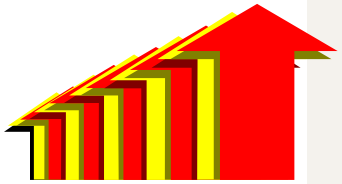
Ethernet Capture Effect

- Nothing succeeds like success
- Stations backing off from a collision are less likely to transmit now than a station which is not backing off
- Thus the station with the last successful transmission is more likely to be able to transmit now and be successful
- Leads to perceived “capture” of wire by one station

Ethernet Capture Effect

- Capture effect is broken when other stations get a frame in edgewise, as they will by chance
- Effect is highly visible between fast stations on the same wire
- TCP buffer sizes limit capture duration
- Competition amongst many stations reduces capture effect
- Capture effect with broadcasts = death

Ten little hubs, all in a row



- Remember those Preamble bits?
- Many are lost as a hub/repeater syncs on an incoming frame
- Lose too many and the SFD byte will not be seen, meaning no frame at all
- Bit shape is more jittery as bits are repeated, meaning poorer signal quality
- Bottom line: limit to two hubs in a row

Ten little switches, all in a row

- Bridges and above are store and forward class devices
- They read in a frame and then regenerate a complete new one, as if the box were two Ethernet boards back to back
- Bridges terminate a collision domain
- Store means delay, up to one whole packet time

Ten little switches, all in a row

- Bridges/switches learn which MAC addresses are on which ports, thus reducing repeating all traffic to all ports
- Bridges pass broadcast and multicast traffic, because no station has that kind of source address
- VLANs are bridges constructed from software, switches, and raw courage
- VLANs can leak broad/multicast traffic

Ten little switches, all in a row

- VLANs “tag” frames by adding two bytes of VLAN info where the Type / Length field was and slides down the original bytes. IEEE 802.1q, 802.1ac
- Bridges/switches can be cascaded to great depth because they store and forward each frame (preamble, bit jitter)
- Excessive broadcast/multicast traffic suggests adding routers selectively

Speed, the conversion problem

- Low speed to high is easy: buffer a low speed frame, repeat it at high speed when an opening occurs
- High speed to low can be difficult:
  - Buffering at the low speed output port means no feedback to the high speed originator
  - Buffering at the high speed input port is wisest, but there is a limit to the memory available
  - Feedback, back pressure is by an artificial collision

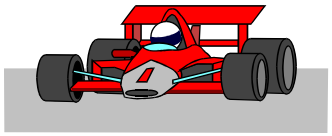
Spanning Tree

- Spanning tree algorithm is to prevent loops amongst bridges
- Packets are exchanged from bridge to bridge to develop a sink tree for the net and then to put selected bridges into standby mode
- DEC spanning tree algorithm works well and fast

Spanning Tree

- IEEE spanning tree algorithm is trouble
  - It stops all traffic transmission for about 20 seconds while spanning tree chatter reexamines the network topology
  - Turning on a desktop PC can trigger a rescan
  - Such deafness clobbers DHCP and NW logins and more, not to mention bothering everyone with comms outages
  - Use the quick convergence mode (Port Fast)

More speed stuff



- A modern PC can easily fill a 10Mbps Ethernet
- They can fill a Fast (100Mbps) Ethernet
- They cannot yet fill a Gigabit link. I got only 400Mbps out of regular PCs
- ISA bus boards are not good at 100Mbps and above; use PCI boards
- Turn off early transmit/receive(!)

FDX and auto-destruct mode

- The current sales appeal phrase is Full Duplex Ethernet. “It offers 200Mbps service” they say.
- Such is not the case, and one may readily be worse off than with Half Duplex (regular) work
- Full duplex lacks flow control, and what the IEEE has generated for that task is not necessarily supported well

Novell

FDX and auto-destruct mode

- FDX works fine when traffic is light. It does no good then, but it works fine.
- When traffic is heavy is when we want a boost. Alas lack of flow control under those conditions means overflow of buffers in relay boxes and lost packets
- Think of the Internet. It loses packets when overflows occur, and that hurts

HDX off the backbone

- HDX uses collisions to indicate a channel is busy, and that can cascade back to the point of origin.
- Retries after a collision are very fast, retries after a packet loss are very slow
- Most transfers are uni-directional: data one way, tinygram ACKs the other, no problem for HDX, no advantage from FDX. Backbones are bi-directional.

Novell

FDX for the backbone

- If, and only if, equipment is demonstrated to use IEEE FDX-busy signaling then consider FDX
- Autoconfiguration of boards and boxes often results in unwanted operating modes. Thus manually configure them for duplex and speed; do not trust the equipment to get this right.

Where to put the slow stuff

- A good practice on network design is to place congestion at the point of origin where it can be dealt with swiftly and effectively
- Thus do not put the slow/congested part of the net in the middle of the system
- Sharing a wire (hubs) may be better than faking missing capacity with a switch, and VLANs disguise congestion

Traffic, how big is big?

- We need quick rules of thumb to judge busy networks from idle ones.
- Keep in mind, the net is either sending bits or not, so the averaging time is important to specify when speaking of percentage utilization and so forth
- Ethernet is able to operate well at up to 90% utilization, but competition suffers

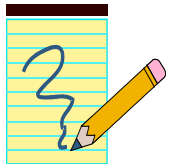
Bigness, thumb rules upon

- Packets/second cost machine resources to process, and router effort to direct (same for small and large packets)
- For 10Mbps Ethernet 1000 pkts/sec is a nice busy number, with headroom to spare, assumes mixed packet length
- For 100Mbps Ethernet 10000 pkts/sec is a busy wire

Bigness, thumb rules upon

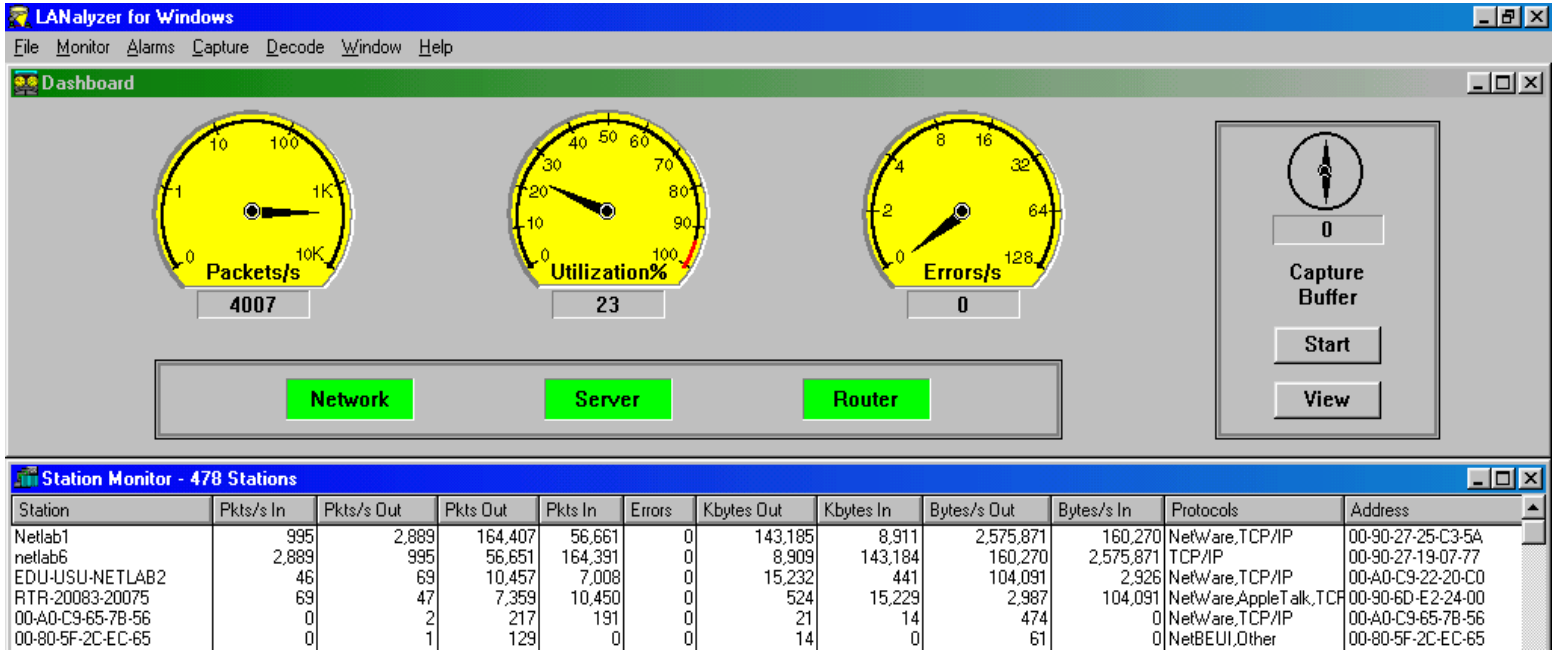
- Percentage utilization of the net requires we look more carefully. 20% will be a largely quiet net, 80% will seem sluggish or sticky
- Excessive broadcast traffic will slow down clients to yield a slow appearing network

Bigness, test results



- The pictures which follow are from Novell's Lanalyzer for Windows (LZFW)
- 100Mbps hub, switches are slower
- Traffic was generated artificially
- Real traffic is erratic when averaged over more than a few seconds. Allow headroom for long duration bursts by increasing bandwidth

Bigness, test results (watch dials)

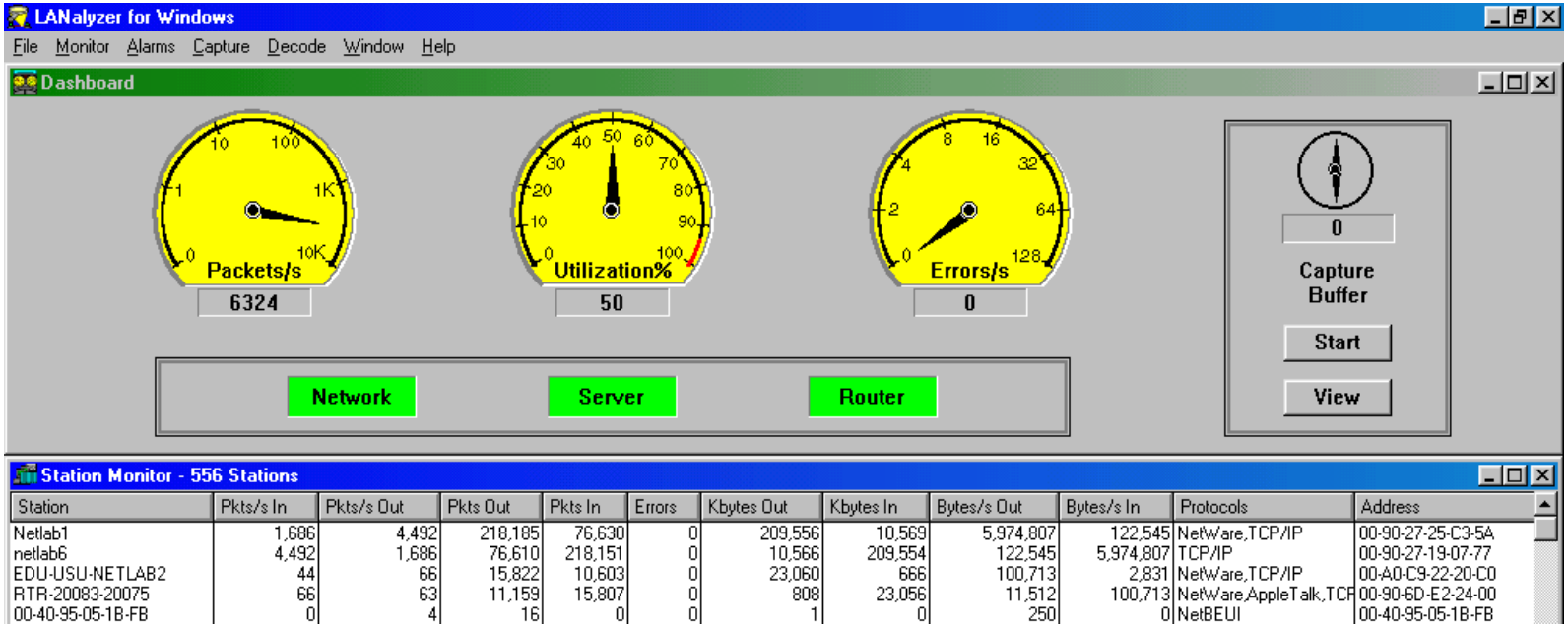


Station	Pkts/s In	Pkts/s Out	Pkts Out	Pkts In	Errors	Kbytes Out	Kbytes In	Bytes/s Out	Bytes/s In	Protocols	Address
Netlab1	995	2,889	164,407	56,661	0	143,185	8,911	2,575,871	160,270	NetWare,TCP/IP	00-90-27-25-C3-5A
netlab6	2,889	995	56,651	164,391	0	8,909	143,184	160,270	2,575,871	TCP/IP	00-90-27-19-07-77
EDU-USU-NETLAB2	46	69	10,457	7,008	0	15,232	441	104,091	2,926	NetWare,TCP/IP	00-A0-C9-22-20-C0
RTR-20083-20075	69	47	7,359	10,450	0	524	15,229	2,987	104,091	NetWare,AppleTalk,TCP	00-90-6D-E2-24-00
00-A0-C9-65-7B-56	0	2	217	191	0	21	14	474	0	NetWare,TCP/IP	00-A0-C9-65-7B-56
00-80-5F-2C-EC-65	0	1	129	0	0	14	0	61	0	NetBEUI,Other	00-80-5F-2C-EC-65

Mixture of large but mostly small packets
About 2.5MB/sec (20Mbps), 4000 pkts/sec

Novell

Bigness, test results



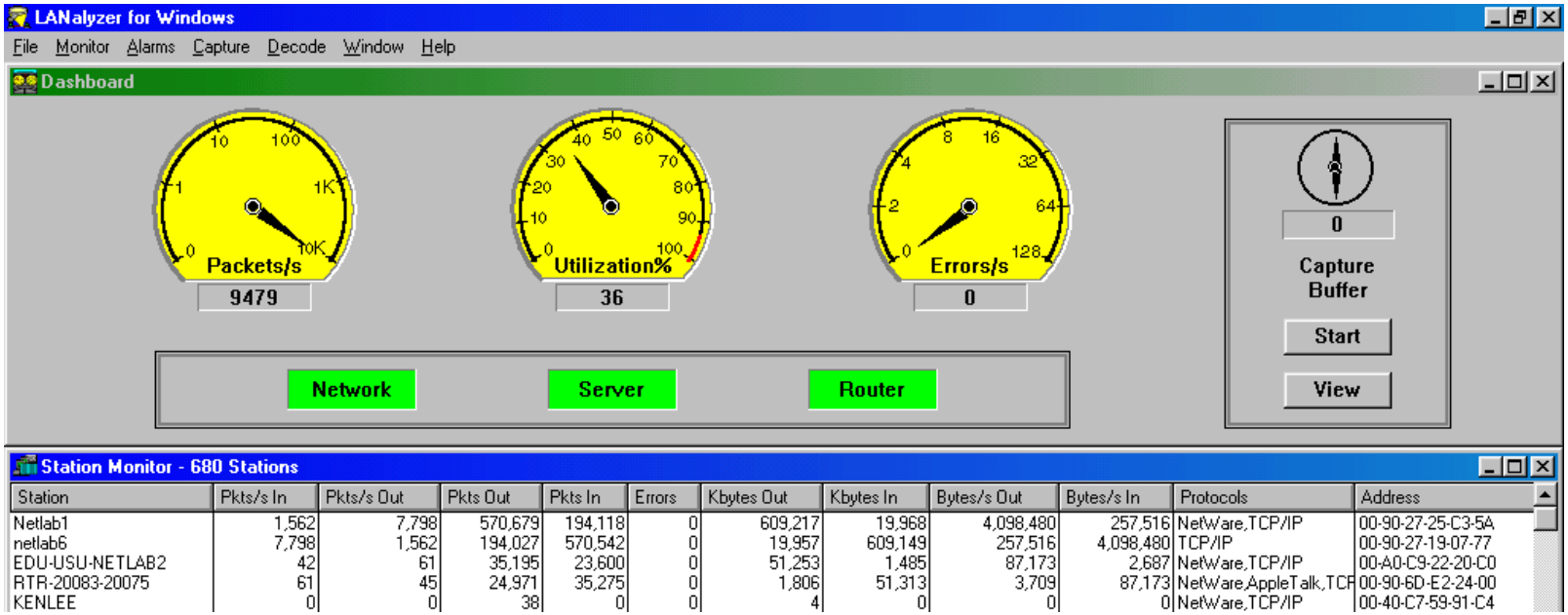
Station	Pkts/s In	Pkts/s Out	Pkts Out	Pkts In	Errors	Kbytes Out	Kbytes In	Bytes/s Out	Bytes/s In	Protocols	Address
Netlab1	1,686	4,492	218,185	76,630	0	209,556	10,569	5,974,807	122,545	NetWare,TCP/IP	00-90-27-25-C3-5A
netlab6	4,492	1,686	76,610	218,151	0	10,566	209,554	122,545	5,974,807	TCP/IP	00-90-27-19-07-77
EDU-USU-NETLAB2	44	66	15,822	10,603	0	23,060	666	100,713	2,831	NetWare,TCP/IP	00-A0-C9-22-20-C0
RTR-20083-20075	66	63	11,159	15,807	0	808	23,056	11,512	100,713	NetWare,AppleTalk,TCP	00-90-6D-E2-24-00
00-40-95-05-1B-FB	0	4	16	0	0	1	0	250	0	NetBEUI	00-40-95-05-1B-FB

Mostly large packets

About 5.9MB/sec (48Mbps), 6300 pkts/sec

Novell

Bigness, test results



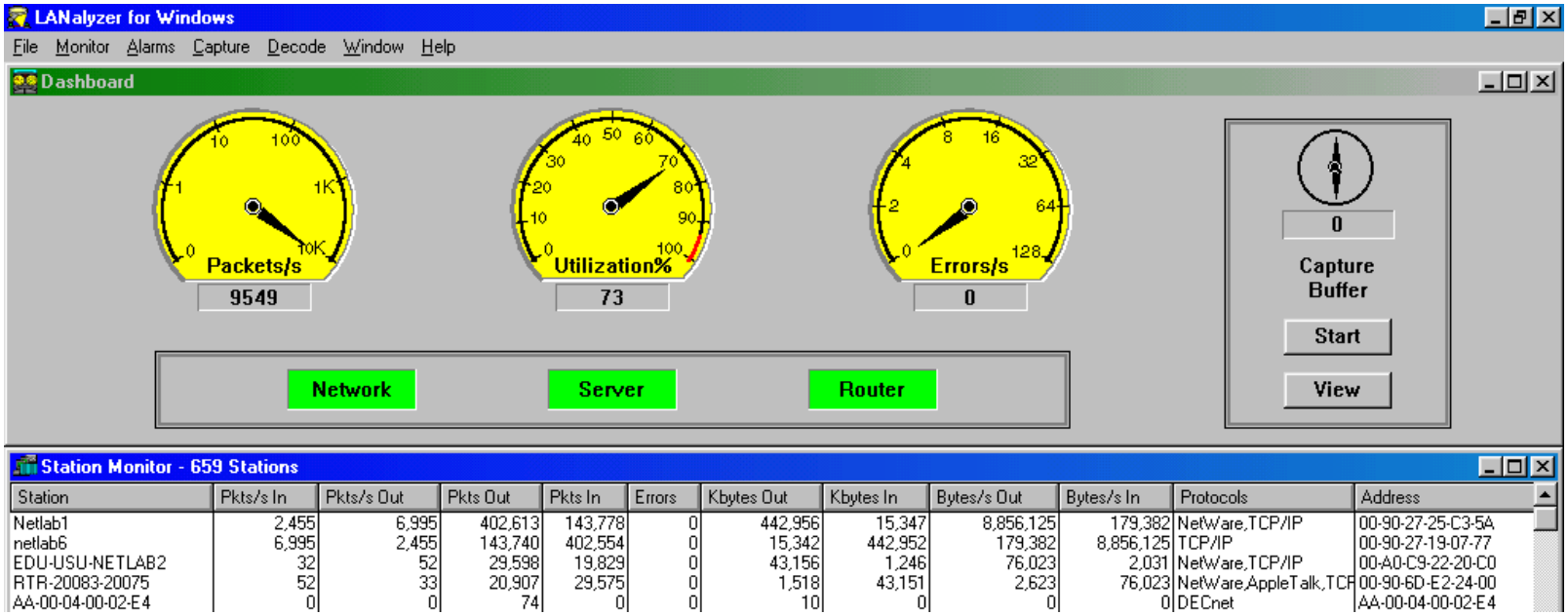
Station	Pkts/s In	Pkts/s Out	Pkts Out	Pkts In	Errors	Kbytes Out	Kbytes In	Bytes/s Out	Bytes/s In	Protocols	Address
Netlab1	1,562	7,798	570,679	194,118	0	609,217	19,968	4,098,480	257,516	NetWare,TCP/IP	00-90-27-25-C3-5A
netlab6	7,798	1,562	194,027	570,542	0	19,957	609,149	257,516	4,098,480	TCP/IP	00-90-27-19-07-77
EDU-USU-NETLAB2	42	61	35,195	23,600	0	51,253	1,485	87,173	2,687	NetWare,TCP/IP	00-A0-C9-22-20-C0
RTR-20083-20075	61	45	24,971	35,275	0	1,806	51,313	3,709	87,173	NetWare,AppleTalk,TCP	00-90-6D-E2-24-00
KENLEE	0	0	38	0	0	4	0	0	0	NetWare,TCP/IP	00-40-C7-59-91-C4

Mostly small packets, more efficient transfer

About 4MB/sec (32Mbps), 9500 pkts/sec

Novell

Bigness, test results

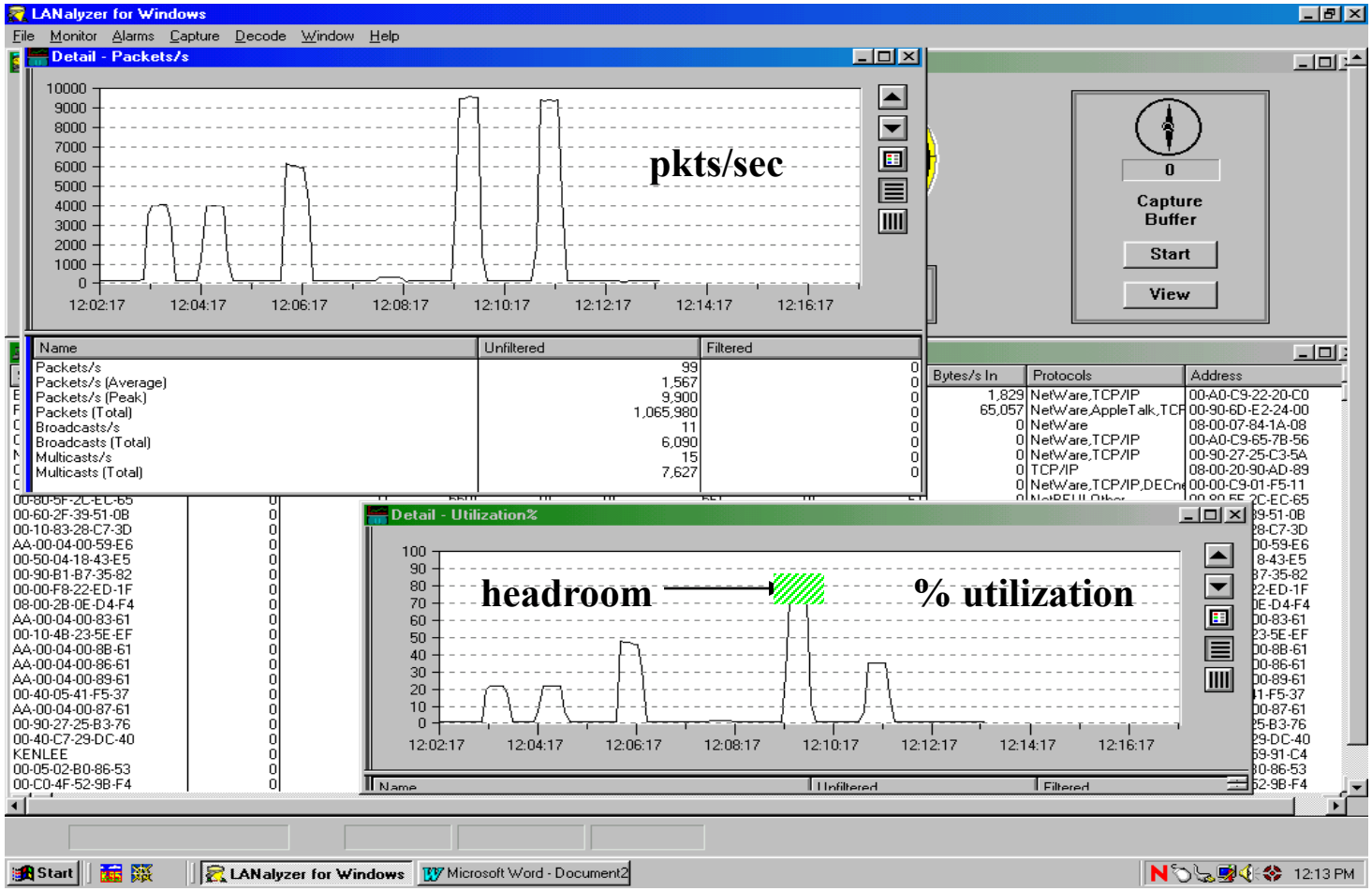


Large packets, more efficient transfer,

About 8.9MB/sec (70Mbps), 9500 pkts/sec

Novell

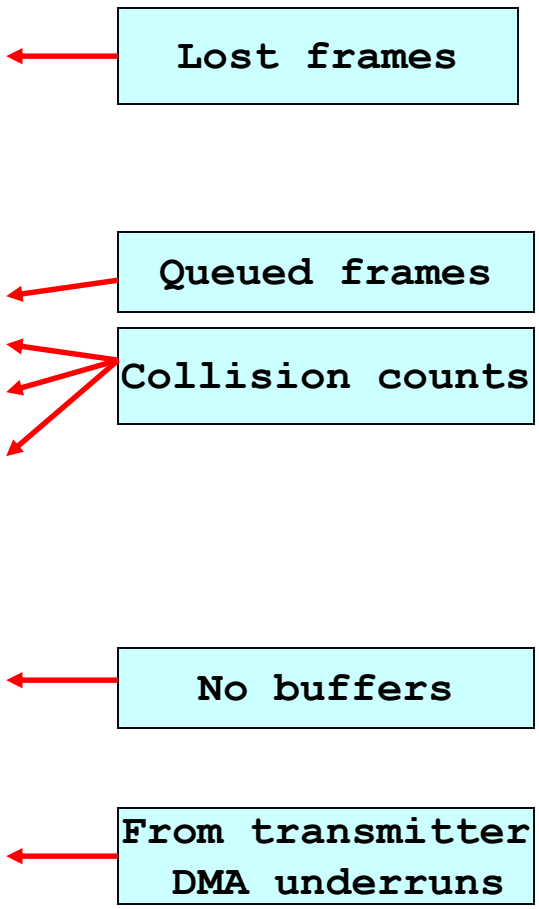
Big, overall rates from tests



Novell

NetWare Monitor, Lan Statistics

Tx Good Frames	1,540,477
Tx Maximum Collisions	7
Tx Late Collisions	0
Tx DMA Underruns	0
Tx Lost Carrier Sense	0
Tx Frames Deferred	3,590
Tx OK With Single Collisions	5,672
Tx OK With Multiple Collisions	5,346
Tx Total Collisions	20,536
Rx Good Frames	2,973,059
Rx CRC Errors	0
Rx Alignment Errors	0
Rx No Resource Errors	15
Rx DMA Overrun Errors	0
Rx Collision Detect Errors	0
Rx Short Frame Errors	19



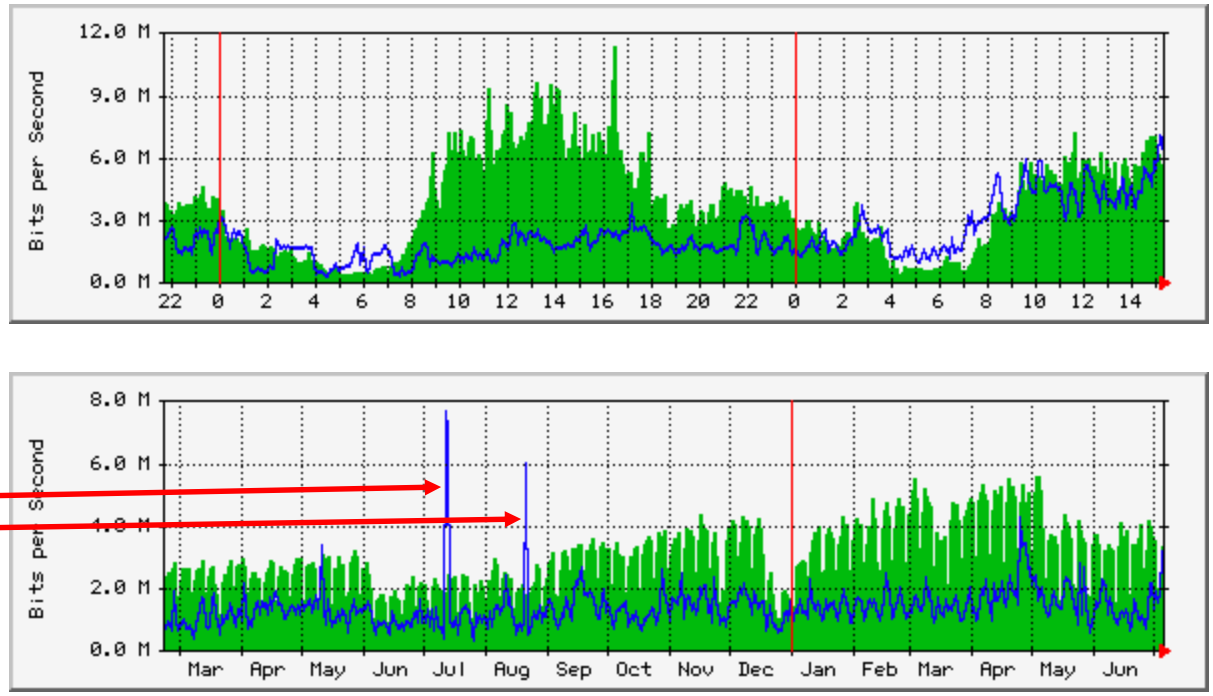
Novell

MRTG web graphics (free)

- 5 minute and daily link averages

in

out



DoS Attacks

http://ee-staff.ethz.ch/~oetiker

Novell

Novell