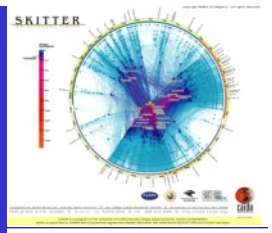
ISP and NSP Security Workshop

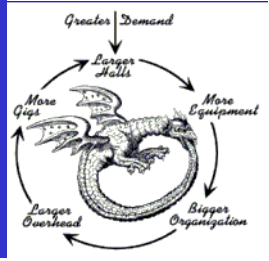
APRICOT 2013

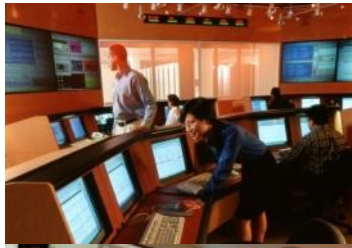
Day 3

Network Telemetry

Gain Visibility







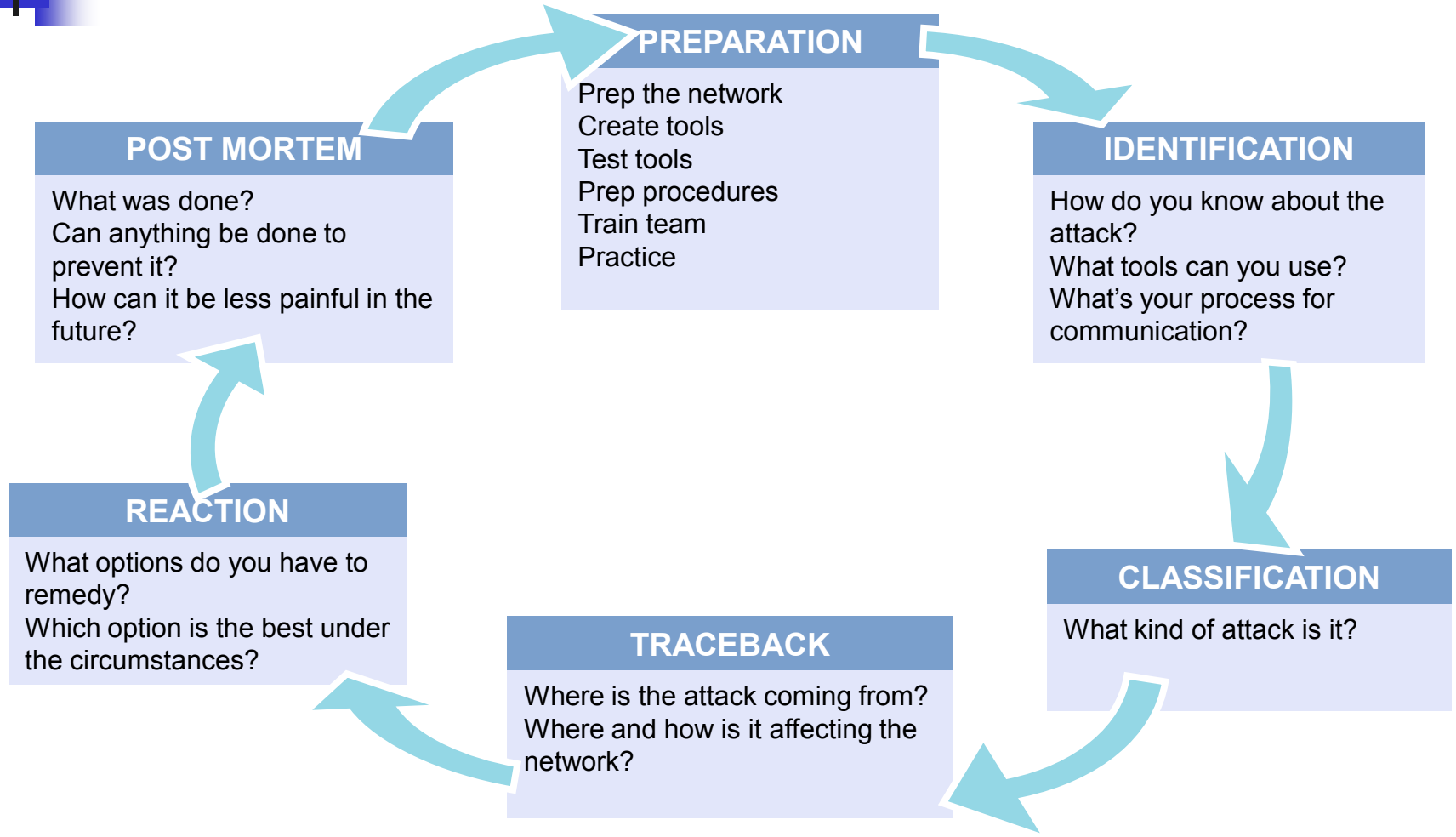
Total Visibility

- Network Telemetry: Why, What and Where
  - Why does one need to listen to the network?
  - What is one listening to?
  - Where does one gather data or information from?
- Network Telemetry: Tools, Techniques and Protocols
  - How to gather data or information?

Check List

- Check SNMP. Is there more you can do with it to pull down security information?
- Check RMON. Can you use it?
- Check Netflow. Are you using it, can you pull down more?
- See addendum for lots of links.

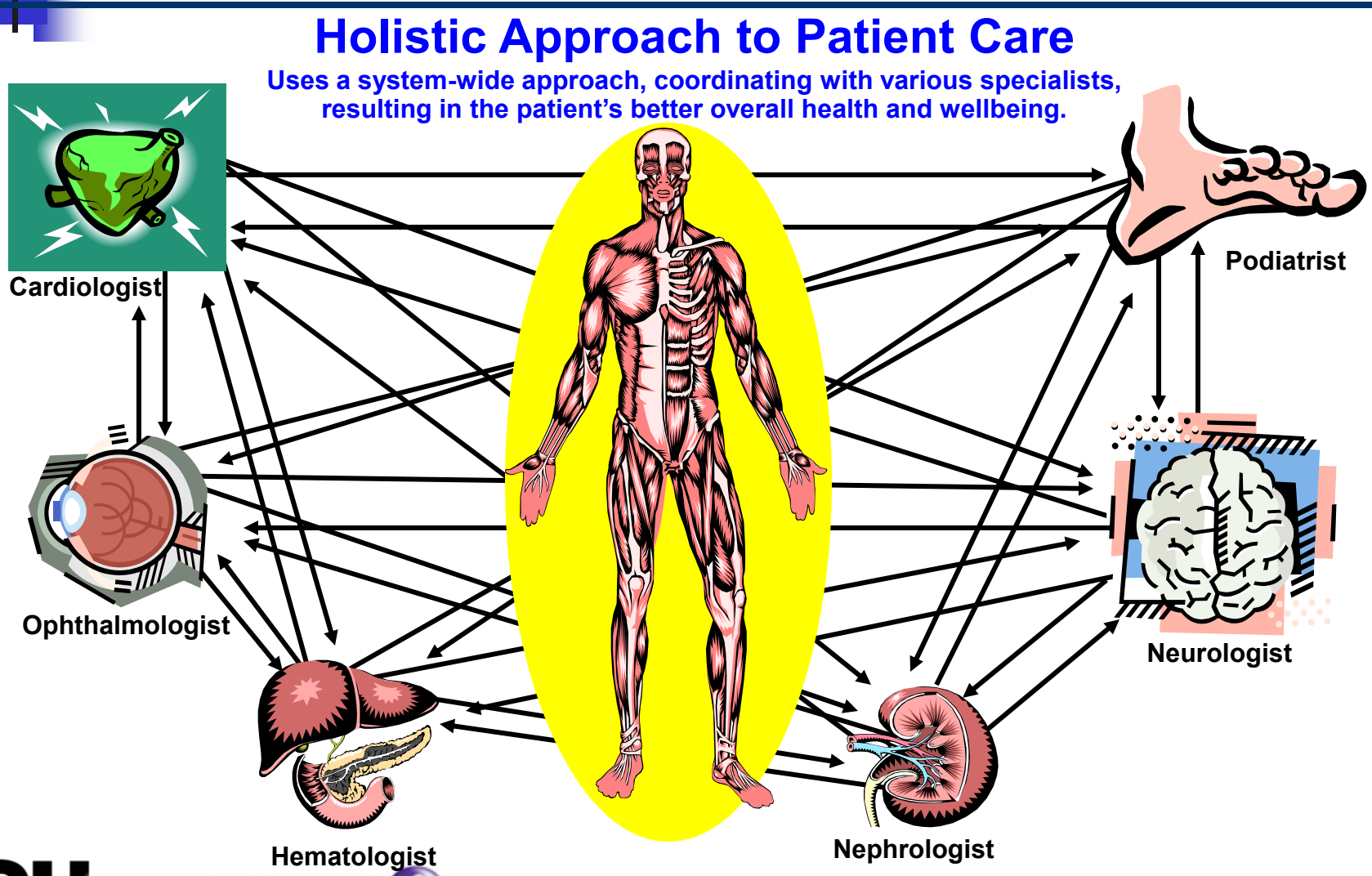
Review: Six Phases of Incident Response



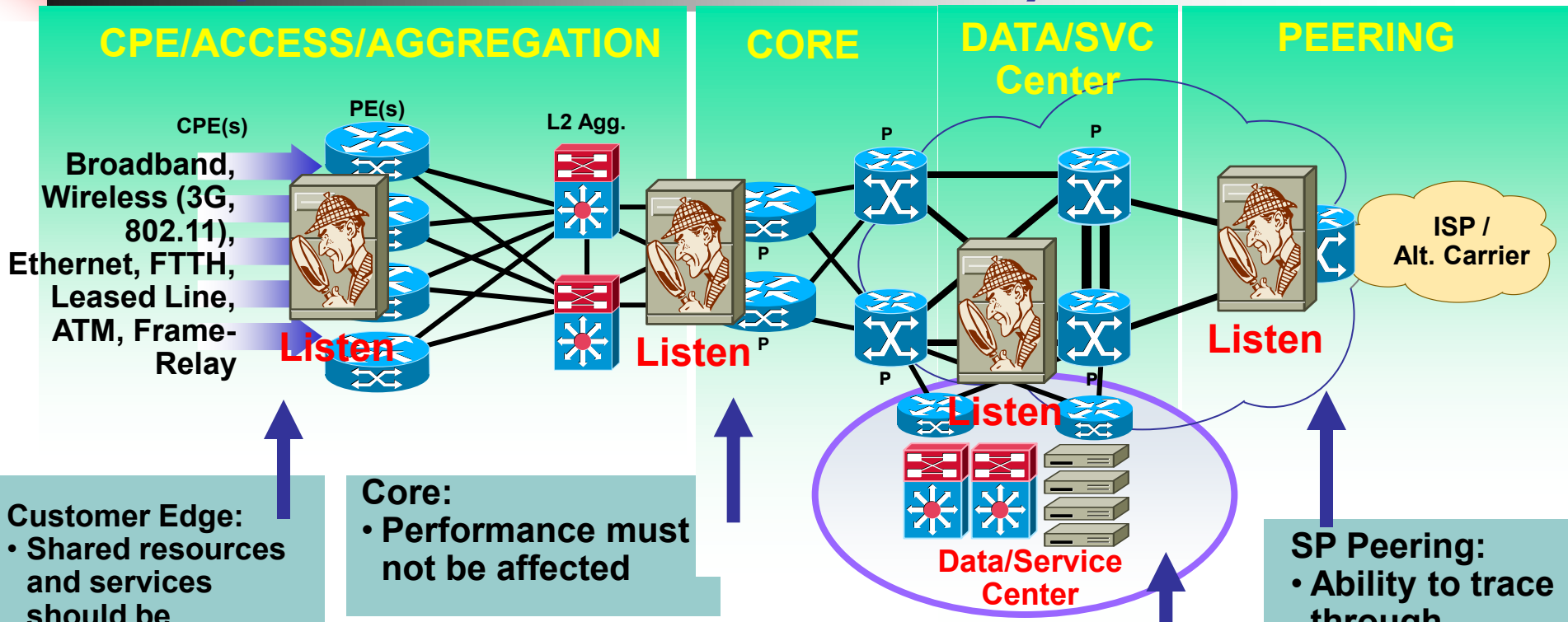
Why Does One Need to Listen to the Network?

- First and foremost
  - Helps with the other 5 steps [P-I-C-T-R-P]
- Helps to understand the network baseline and behavior
- To understand telemetry elements for information gathering
  - sources (data collection points),
  - protocols to use for data collection
  - telemetry tools
- In the event of security incident
  - to identify and know beforehand that what information is available for forensic work – audit trail
  - faster response time to restore availability when using telemetry during a security incident

Holistic Approach to System-Wide Telemetry



Holistic Approach to System-Wide Telemetry



Customer Edge:
- Shared resources and services should be available

Core:
- Performance must not be affected

Data Center:
- Inter as well as Intra Data Center traffic

SP Peering:
- Ability to trace through asymmetric traffic

Understand the Concept of Data Gathering

Risks and threats are NOT prevalent in one place ONLY…

Need to watch everywhere to avoid being eaten by thousand turkeys...

- Listening to a network element
  - Per device listening
  - Local data provide information about local threats
- Listening to Many
  - Correlation is a MUST
  - Intelligent analysis is a MUST

Listen

High CPU

- Spikes in CPU load on routers, switches, servers, and other devices is often an indication that an event is taking place. Such occurences should always be investigated.
- However, high CPU is not always an indicator of malicious activity. It is important to have both a baseline of historical CPU utilization statistics as well as an understanding of the various processes running on a given system, in order to determine the cause of CPU spikes.
- Correlating CPU utilization with other information such as network traffic statistics, routing-table changes, etc., is often required in order to gain an accurate understanding of the cause(s) and impact of an event.

Link-Flaps

- Link-flaps are also an indication that something is amiss.
- They're often a sign of misconfiguration, backhole incidents and the like - but they can also result from malicious activity, such as a DoS attack against a router causing a reload due to CPU spike, and hence a link-flap.
- Routers and switches can be configured to notify monitoring systems when link-flaps occur.
- Correlating link-flaps with other forms of information is often necessary in order to gain a complete understanding of an event.

Instrumentation

- Network instrumentation offers the most extensive and useful detection capabilities.
- This instrumentation is often coupled with dedicated analysis systems which collect, analyze, and correlate information from disparate sources in order to present a more complete view of events taking place within the network.
- There are several forms of instrumentation built into routers, switches, and other network devices. Instrumentation is also present in most modern general-purpose operating systems.
- There are a number of open source and commercial tools available which greatly enhance the utility of instrumentation.
- Getting started with network instrumentation is both inexpensive and relatively easy.

Example - sh proc c

```
7600>show proc c | e 0.00%__0.00%__0.00%
CPU utilization for five seconds: 38%/26%; one minute: 40%; five minutes: 43%
 PID Runtime(ms)   Invoked      uSecs   5Sec   1Min   5Min TTY Process
   5   192962596  13452649      14343  0.00%  0.52%  0.44%   0 Check heaps
  15   4227662201540855414        274  0.65%  0.50%  0.49%   0 ARP Input
  26   2629012683680473726         71  0.24%  0.29%  0.36%   0 Net Background
  50     9564564  11374799        840  0.08%  0.07%  0.08%   0 Compute load avg
  51    15291660    947844      16133  0.00%  0.03%  0.00%   0 Per-minute Jobs
  58    15336356  92241638        166  0.08%  0.02%  0.00%   0 esw_vlan_stat_pr
  67    10760516 506893631         21  0.00%  0.01%  0.00%   0 Spanning Tree
  68  31804659682556402094       1244  7.02%  7.04%  7.75%   0 IP Input
  69    25488912  65260648        390  0.00%  0.03%  0.00%   0 CDP Protocol
  73    16425564  11367610       1444  0.08%  0.02%  0.00%   0 QOS Stats Export
  81    12460616   1020497      12210  0.00%  0.02%  0.00%   0 Adj Manager
  82   442430400  87286325       5068  0.65%  0.73%  0.74%   0 CEF process
  83    68812944  11509863       5978  0.00%  0.09%  0.11%   0 IPC LC Message H
  95    54354632  98373054        552  0.16%  0.12%  0.13%   0 DHCPD Receive
  96    61891604  58317134       1061  1.47%  0.00%  4.43%   0 Feature Manager
 111        9420     12010        784  0.00%  0.23%  0.46%   0 Exec
 165   1817346481141817381        159  0.32%  0.57%  0.40%   0 IP SNMP
 166   117953648 573360040        205  0.00%  0.32%  0.26%   0 PDU DISPATCHER
 167   545931776 634808059        859  0.40%  1.37%  1.19%   0 SNMP ENGINE
 171    22376852 154770330        144  0.00%  0.02%  0.04%   0 IGMP Input
 175         680       263       2585  0.24%  0.21%  0.14%   1 SSH Process
 177   7481935235090724 14         21  0.08%  0.02%  0.03%   0 Standby (HSRP)
 178    14224288   2051379       6934  0.00%  0.02%  0.00%   0 BGP Scanner
```

CLI Pipes

Example - sh proc c

```
7600>sh proc c | e 0.00
CPU utilization for five seconds: 41%/26%; one minute: 46%; five minutes: 44%
 PID Runtime(ms)   Invoked      uSecs   5Sec   1Min   5Min TTY Process
  15   4227657321540854233        274  0.40%  0.39%  0.47%   0 ARP Input
  26   2629008963680468704         71  0.08%  0.36%  0.39%   0 Net Background
  50      9564512  11374786        840  0.08%  0.07%  0.08%   0 Compute load avg
  68   3180457804255618343O       1244  9.65%  8.49%  7.75%   0 IP Input
  69     25488888  65260576        390  0.32%  0.05%  0.01%   0 CDP Protocol
  82    442429604  87286223       5068  0.73%  0.73%  0.74%   0 CEF process
  :
 175          624        92       6782  0.57%  0.49%  0.16%   1 SSH Process
```

CLI Pipes allow clean and crisp output

IOS CLI - sh proc c (cont.)

- There are processes which are platform-specific - i.e., Feature Manager is found on the 6500/7600 only, while IPC CBus is 7500-specific.
- Aliasing the more complex sh proc c commands to a single-letter alias as part of the standard config is extremely useful when the box is under high load and it's hard to type on the console:
  - `Router(config)#alias exec p show proc c | e 0.00%__0.00%__0.00%`
- Understanding your platform(s), and what's normal - including periodically-run processes (BGP Scanner, for example) - is key
- On the 12000, one must either attach to a linecard or perform an execute command specifying a linecard in order to see its CPU load; on the 7500, one uses the if-con command to session to a VIP.

IOS CLI - sh int

- Sh int displays interface-level statistics, including throughput (pps) and bandwidth (bps)
- Typically, routers are set to use a 5-minute decaying average for interface statistics by default - changing this to 1 minute gives more granular statistics
- Looking for high input/output rates over a period of a minute or so can be very helpful
- Clear the counters first, otherwise it's much harder to determine which interfaces are receiving high rates of traffic

Example - sh int

```
GigabitEthernet3/13 is up, line protocol is up (connected)
  Hardware is C6k 1000Mb 802.3, address is 00d0.0136.000a (bia 00d0.0136.000a)
  Description: IP TELEPHONY
  Internet address is 10.89.254.130/26
  MTU 1500 bytes, BW 1000000 Kbit, DLY 10 usec,
     reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation ARPA, loopback not set
  Keepalive set (10 sec)
  Full-duplex mode, link type is autonegotiation, media type is SX
  output flow-control is unsupported, input flow-control is unsupported, 1000Mb/s
  Clock mode is auto
  input flow-control is off, output flow-control is off
  ARP type: ARPA, ARP Timeout 04:00:00
  Last input 00:00:00, output 00:00:00, output hang never
  Last clearing of "show interface" counters 1y39w
  Input queue: 0/75/15005/235 (size/max/drops/flushes); Total output drops: 1
  Queueing strategy: fifo
  Output queue :0/40 (size/max)
  5 minute input rate 4751000 bits/sec, 3006 packets/sec
  5 minute output rate 4499000 bits/sec, 2755 packets/sec
  L2 Switched: ucast: 19841909032 pkt, 3347755205145 bytes - mcast: 96885779 pkt, 5131184435 bytes
  L3 in Switched: ucast: 27282638229 pkt, 5095662463006 bytes - mcast: 94 pkt, 5191 bytes mcast
  L3 out Switched: ucast: 43107617667 pkt, 7275264441541 bytes
     47118207406 packets input, 9306459456266 bytes, 0 no buffer
     Received 83653389 broadcasts, 0 runts, 0 giants, 0 throttles
     0 input errors, 0 CRC, 0 frame, 649 overrun, 0 ignored
     0 input packets with dribble condition detected
     43210876182 packets output, 8089398934796 bytes, 0 underruns
     0 output errors, 0 collisions, 0 interface resets
     0 babbles, 0 late collision, 0 deferred
     0 lost carrier, 0 no carrier
     0 output buffer failures, 0 output buffers swapped out
```

Example - sh int

```
12000>sh int po1/1/0 | i 1 minute
  1 minute input rate 56616000 bits/sec, 18097
  packets/sec
  1 minute output rate 120609000 bits/sec, 24120
  packets/sec
12000>sh int po1/1/0 | i 1 minute
  1 minute input rate 59030000 bits/sec, 19171
  packets/sec
  1 minute output rate 111233000 bits/sec, 22365
  packets/sec
12000>sh int po1/1/0 | i 1 minute
  1 minute input rate 54307000 bits/sec, 17637
  packets/sec
  1 minute output rate 119223000 bits/sec, 23936
  packets/sec
```

IOS CLI - sh ip int

- sh ip int gives information about features configured on an interface
- It's useful to get the number or name of an ACL in order to check ACL counter hits (6500/7600 only shows ACL counters on Sup720 w/PFC3BXL)
- uRPF drop information is also available via sh ip int, shows information about spoofed and/or RTBH-dropped packets

Example - sh ip int

```
12000>sh ip int po1/1/0 | i veri
  IP verify source reachable-via ANY
   794407 verification drops
   1874428129 suppressed verification
  drops
12000>sh ip int po1/1/0 | i veri
  IP verify source reachable-via ANY
   794408 verification drops
   1874444463 suppressed verification
  drops
```

IOS CLI - sh ip traffic

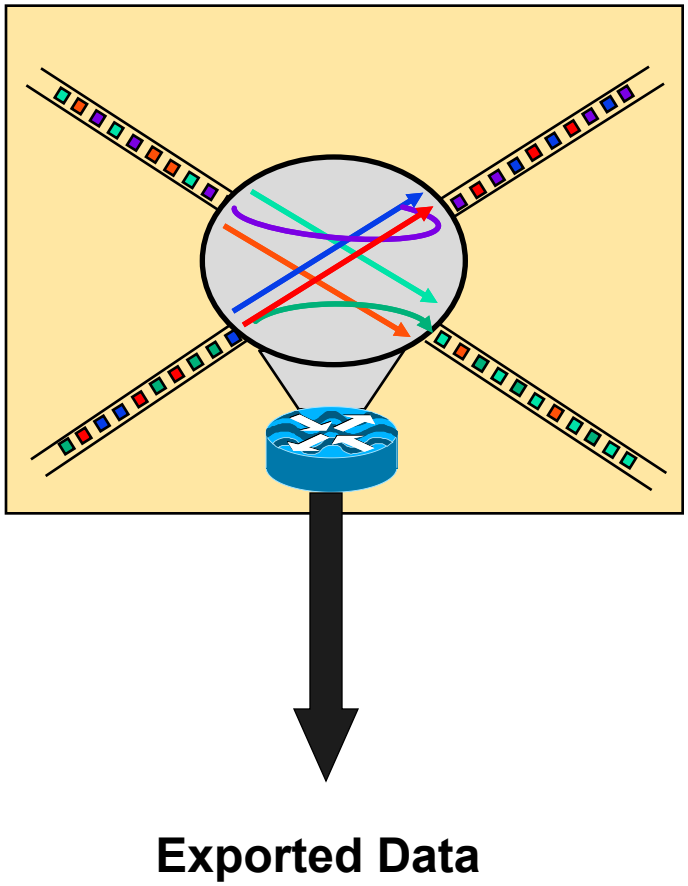
- Sh ip traffic provides a lot of useful global statistics, including per-protocol counts for ICMP, TCP, UDP, and multicast traffic
- Very useful for troubleshooting in general, as well as for spotting oddities
- Also shows global uRPF drop statistics

Example - sh ip traffic

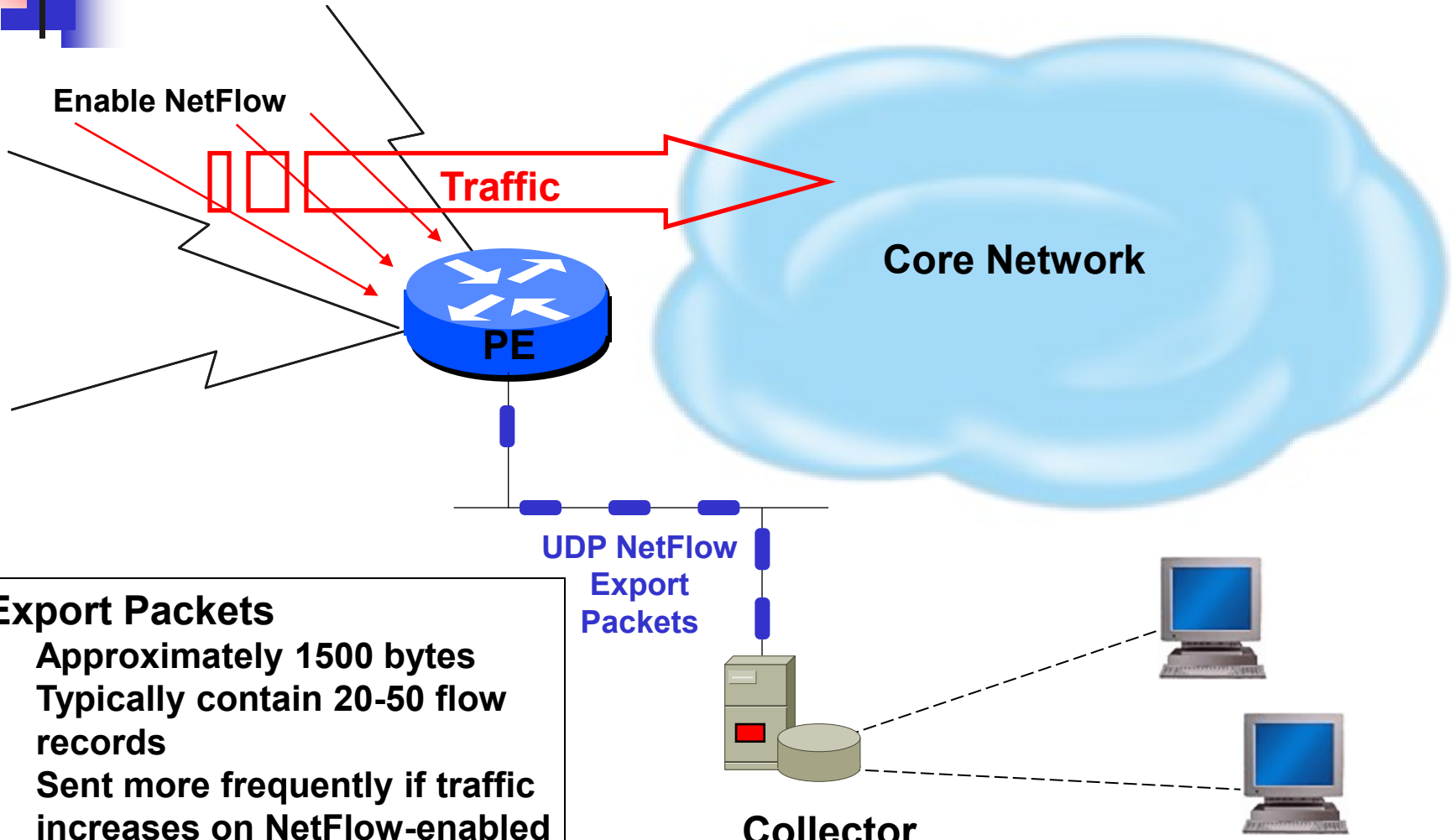
```
12000>sh ip traff | i RPF
         0 no route, 124780722 unicast RPF, 0 forced drop
12000>sh ip traff | i RPF
         0 no route, 124816525 unicast RPF, 0 forced drop
12000>sh ip traff | i RPF
         0 no route, 127777619 unicast RPF, 0 forced drop
12000>sh ip traff | i RPF
         0 no route, 135875095 unicast RPF, 0 forced drop
12000>sh ip traff | i RPF
         0 no route, 150883277 unicast RPF, 0 forced drop
```

What Is a Flow?

- Defined by seven unique keys:
  - Source IP address
  - Destination IP address
  - Source port
  - Destination port
  - Layer 3 protocol type
  - TOS byte (DSCP)
  - Input logical interface (ifIndex)



Creating Export Packets



Export Packets

- Approximately 1500 bytes
- Typically contain 20-50 flow records
- Sent more frequently if traffic increases on NetFlow-enabled interfaces

Key Concept—NetFlow Scalability

- Packet capture is like a *wiretap*
- NetFlow is like a *phone bill*
- This level of granularity allows NetFlow to scale for very large amounts of traffic

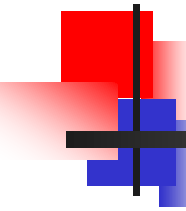
We can learn a lot from studying the phone bill!

Who's talking to whom, over what protocols & ports, for how long, at what speed, for what duration, etc.

NetFlow is a form of *telemetry* pushed from the routers/switches - each one can be a sensor!

NetFlow Versions: Clarifying the Version Myth

NetFlow Version	Comments
1	Original
5	Standard and most common
7	Specific to Cisco Catalyst 6500 and 7600 Series Switches Similar to Version 5, but does not include AS, interface, TCP Flag & TOS information
8	Choice of eleven aggregation schemes Reduces resource usage
9	Flexible, extensible file export format to enable easier support of additional fields & technologies; coming out now are MPLS, Multicast, & BGP Next-Hop
10	IETF standardized version of Flow export, known as IPFIX. Based on Netflow V9.



Why a New Version?

- Fixed formats (versions 1, 5, 7, and 8) are not flexible and adaptable
  - Cisco needed to build a new version each time a customer wanted to export new fields
- When new versions are created, partners need to reengineer to support the new export format

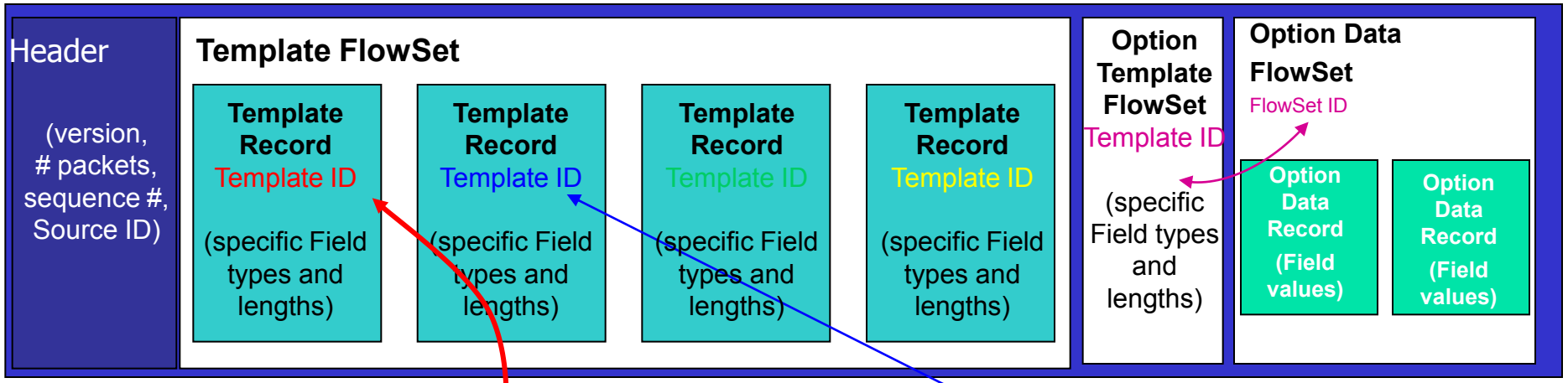
Solution: Build a flexible and extensible export format!

Netflow v9 Principles

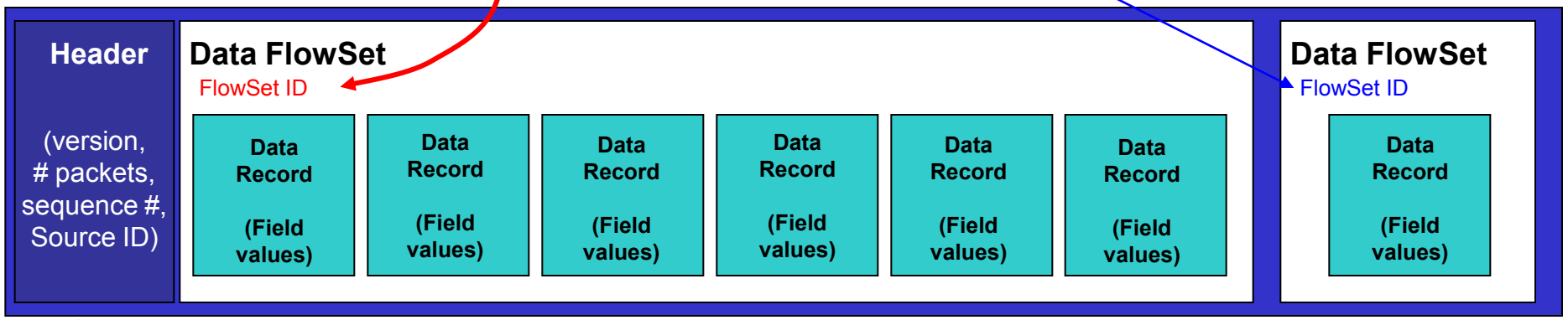
- Version 9 is an export format
- Works as a push model
- Send the template regularly (configurable)
- Independent of the underlying protocol, it is ready for any reliable protocol (e.g.,: TCP, SCTP)

NetFlow v9 Flexible Format

Example of Export Packet right after router boot or NetFlow configuration



Example of Export Packets containing mostly flow information



NetFlow v9 Export

Configuring Version 9 export

```
pamela(config)# ip flow-export version ?
  1
  5
  9
pamela(config)# ip flow-export version 9          .
```

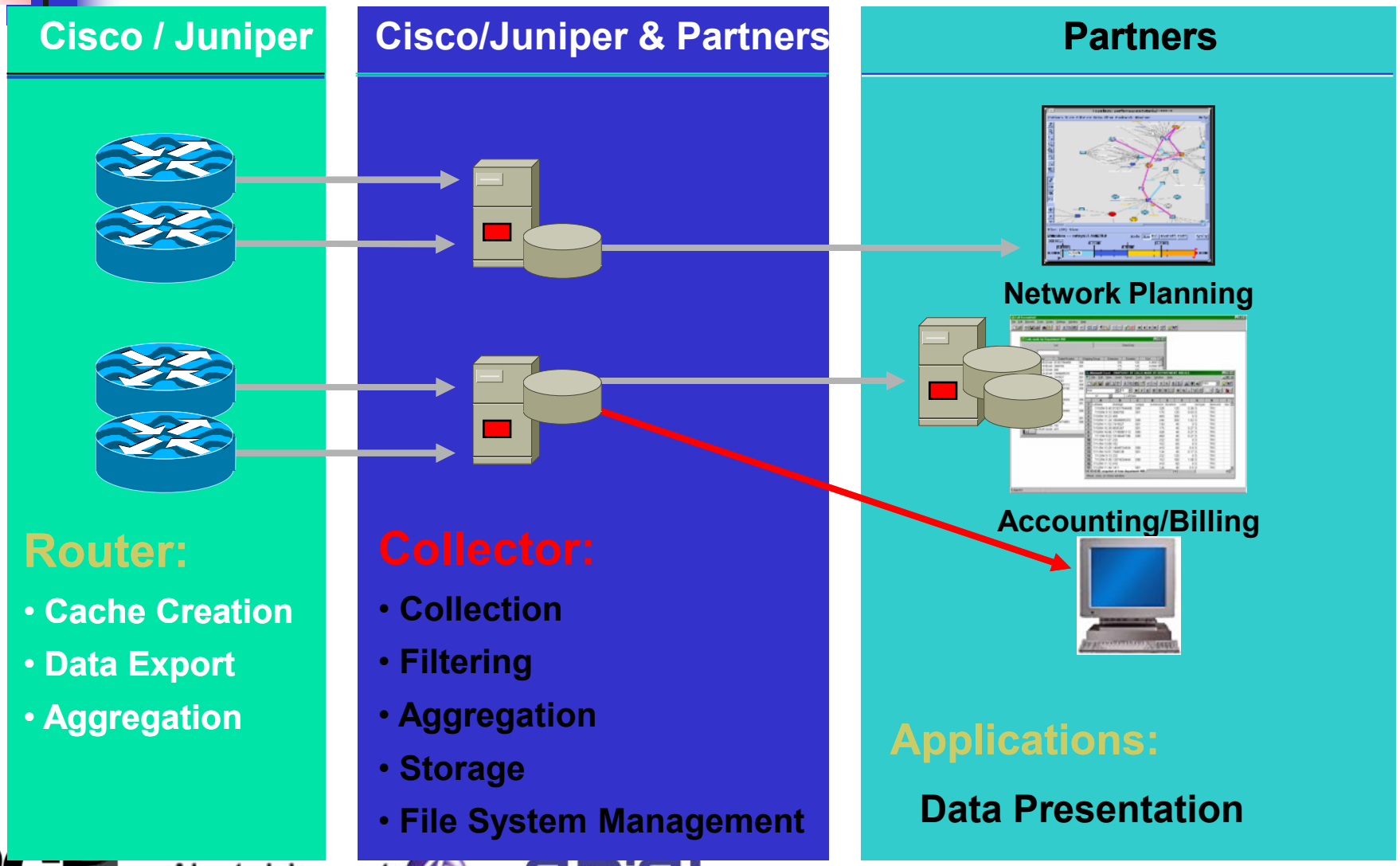
Export versions available for standard NetFlow flows

Configuring Version 9 export for an aggregation scheme

```
pamela(config)# ip flow-aggregation cache as
pamela(config-flow-cache)# enabled
pamela(config-flow-cache)# export ?
  destination Specify the Destination IP address
  version configure aggregation cache export version
pamela(config-flow-cache)# export version ?
  8  Version 8 export format
  9  Version 9 export format
pamela(config-flow-cache)# export version 9
```

Export versions available for aggregated NetFlow flows

NetFlow / jflow Infrastructure



Cisco 7200 NetFlow Example

```
7200>sh ip cache flow
IP packet size distribution (14952M total packets):
   1-32   64   96  128  160  192  224  256  288  320  352
  384  416  448  480
   .001 .325 .096 .198 .029 .014 .010 .010 .012 .003 .003
  .005 .003 .003 .002

    512  544  576 1024 1536 2048 2560 3072 3584 4096 4608
   .004 .005 .009 .043 .217 .000 .000 .000 .000 .000 .000

IP Flow Switching Cache, 4456704 bytes
  65527 active, 9 inactive, 2364260060 added
  4143679566 ager polls, 0 flow alloc failures
  Active flows timeout in 30 minutes
  Inactive flows timeout in 15 seconds
  last clearing of statistics never
```

Active flows

NetFlow Timeouts – tune to avoid the churn

Cisco 7200 NetFlow Example (Cont.)

Traffic type

Protocol	Total	Flows	Packets	Bytes	Packets	Active(Sec)	Idle(Sec)
---	Flows	/Sec	/Flow	/Pkt	/Sec	/Flow	/Flow
TCP-Telnet	1398292	0.3	14	156	4.6	6.0	17.2
TCP-FTP	99569986	23.1	1	41	24.2	0.0	4.8
TCP-FTPD	185530	0.0	1	66	0.0	1.5	17.4
TCP-WWW	440235639	102.5	8	483	919.5	2.9	10.1
TCP-SMTP	18951357	4.4	21	629	94.1	6.4	20.0
TCP-X	11340	0.0	1	48	0.0	0.2	40.8
TCP-BGP	4018	0.0	2	51	0.0	7.5	12.5
TCP-NNTP	2701390	0.6	104	846	65.5	10.6	16.9
TCP-Frag	38932	0.0	11	407	0.1	1.9	17.2
TCP-other	403434143	93.9	7	444	688.2	6.9	18.6
UDP-DNS	65590214	15.2	1	114	24.0	1.6	17.7

Hint: How many TCP based applications you know have 1 pkt / flow?

Cisco 7200 NetFlow Example (Cont.)

Hint: What's going on here?

```
SrcIf          SrcIPaddress    DstIf          DstIPaddress    Pr
  SrcP DstP  Pkts
Fa0/1          10.66.74.46     Fa0/0          219.103.129.162 01
  0000 0800     1
Fa0/1          10.66.115.182   Fa0/0          194.22.114.198  01
  0000 0800     1
Fa2/1          10.66.74.46     Fa0/0          61.79.227.123   01
  0000 0800     1
Fa0/1          10.66.74.46     Fa0/0          211.167.105.242 01
  0000 0800     1
Fa0/0          129.42.184.35   Null           64.104.193.198  06
  2891 0019     3
Fa2/1          10.66.115.182   Fa0/0          202.20.138.184  01
  0000 0800     1
Fa2/1          10.66.115.182   Fa0/0          63.76.237.255   01
  0000 0800     1
```

Cisco Catalyst 6500 and 7600 Series Switches

```
6500>sh mls netflow ip detail
Displaying Netflow entries in Supervisor Earl
DstIP           SrcIP           Prot:SrcPort:DstPort   Src i/f:AdjPtr
---------------------------------------------------------------------
Pkts          Bytes       Age    LastSeen  Attributes
-----------------------------------------------

    QoS     Police Count Threshold    Leak     Drop Bucket  Use-Tbl Use-Enable
--------+-----------+--------+----------+----+------+-------+----------+
172.87.19.217   171.70.154.90   tcp :10112   :www      1023: 0
3             144           10     00:07:11   L3 - Dynamic
  0x0           0                 0         0        NO    48         NO        NO
171.101.24.123  171.69.89.39    tcp :1303    :139      400 : 0
0             0             39     00:06:42   L3 - Dynamic
  0x0           0                 0         0        NO    48         NO        NO
202.56.200.22   198.133.219.25  icmp:0       :0        1028: 0
26            2028          383    00:07:05   L3 - Dynamic
  0x0           0                 0         0        NO    78         NO        NO
```

Review the output

Cisco Catalyst 6500 and 7600 Series Switches (Cont.)

```
6500>sh mls netflow ip dest www.cisco.com det
Displaying Netflow entries in Supervisor Earl
DstIP           SrcIP           Prot:SrcPort:DstPort   Src i/f:AdjPtr
-----------------------------------------------------------------------
Pkts         Bytes       Age   LastSeen  Attributes
-----------------------------------------------------

    QoS     Police Count Threshold    Leak     Drop Bucket  Use-Tbl Use-Enable
----------+------------+---------+-----------+----+-------+-------+----------+
198.133.219.25  66.189.188.230  icmp:0      :0        1017: 0
1            60          28    00:16:36   L3 - Dynamic
  0x0          0               0        0       NO   60        NO       NO
198.133.219.25  142.32.208.231  tcp :9415   :www      1016: 0
34           1501        32    00:16:32   L3 - Dynamic
  0x0          0               0        0       NO   40        NO       NO
198.133.219.25  65.114.202.35   tcp :4936   :www      1017: 0
24           1099        24    00:16:40   L3 - Dynamic
  0x0          0               0        0       NO   40        NO       NO
198.133.219.25  80.202.170.129  icmp:0      :0        1017: 0
1            60          32    00:16:32   L3 - Dynamic
  0x0          0               0        0       NO   60        NO       NO
```

Review the output.

Versions

- Some releases are vendor/product specific
- What you need to know
  - Version 5
    - Widely supported
  - Version 8
    - Adds security to reporting stream (DES)
  - Version 9
    - Adds generalized formatting
    - Reduces need to upgrade tools between versions
  - Version 10 (IPFIX)
    - Increasing in availability across vendors
    - Supports multiple service types (L2, L3, NAT…)

cflowd Configuration Example

You *must* configure sampling for cflowd to work

```
forwarding-options {
    sampling {
        input {
            family inet {
                rate 1000;
                run-length 9;
            }
        }
        output {
            file filename sample.cfld files 20 size 1m;
            cflowd <address> {
                port <port>;
                version 5;
            }
        }
    }
}
```

```
interface FastEthernet0/0
 ip route-cache flow
interface FastEthernet0/1
ip route-cache flow

ip flow-export version 5
ip flow-export destination <ip_address> <port>
ip flow-export source FastEthernet0/0
```

Use address 169.223.142.3
Port 2x01 for Juniper
Port 2x02 for cisco
X = group number

```
forwarding-options {
    sampling {
        input {
            family inet {
                rate 1000;
                run-length 9;
            }
        }
        output {
            file filename sample.cfld
files 20 size 1m;
            cflowd <address> {
                port <port>;
                version 5;
            }
        }
    }

interfaces ge-0/1/0 {
        unit 0 {
            family inet {
                filter {
                    input all;
                    output all;
                }
        }
```

```
interface FastEthernet0/0
 ip route-cache flow
interface FastEthernet0/1
ip route-cache flow

ip flow-export version 5
ip flow-export destination <ip_address> <port>
ip flow-export source FastEthernet0/0
```

Use address 169.223.142.3
Port 2x01 for Juniper
Port 2x02 for cisco
X = group number

```
firewall {
    filter all {
        term all {
            then {
                sample;
                accept;
            }
        }
    }
}

}
```

cflowd Output Option

- cflowd is an output option under the sampling configuration
  - Each option discussed in detail

```
forwarding-options {
    sampling {
        input {
            family inet {
                rate 1000;
                run-length 9;
            }
        }
        output {
            file filename sample.cfld files 20 size 1m;
            cflowd 10.1.86.2 {
                port 2055;
                version 5;
            }
        }
    }
```

cflowd Aggregate Format

Viewing the local log file on the router

```
lab@R1> show log sampled
Jan  7 18:30:44    Start time of flow: 3812598
Jan  7 18:30:44    End time of flow: 3812598
Jan  7 18:30:44    Src port: 1088
Jan  7 18:30:44    Dst port: 1241
Jan  7 18:30:44    TCP flags: 0x0
Jan  7 18:30:44    IP proto num: 6
Jan  7 18:30:44    TOS: 0x0
Jan  7 18:30:44    Src AS: 64514
Jan  7 18:30:44    Dst AS: 64513
Jan  7 18:30:44    Src netmask len: 16
Jan  7 18:30:44    Dst netmask len: 24
Jan  7 18:30:44 v5 flow entry
Jan  7 18:30:44    Src addr: 192.168.46.101
Jan  7 18:30:44    Dst addr: 172.16.3.18
Jan  7 18:30:44    Nhop addr: 10.1.84.0
Jan  7 18:30:44    Input interface: 30
Jan  7 18:30:44    Output interface: 40
Jan  7 18:30:44    Pkts in flow: 1
Jan  7 18:30:44    Bytes in flow: 46
Jan  7 18:30:44    Start time of flow: 3812603
```

```
Jan  7 18:30:44    End time of flow: 3812603
Jan  7 18:30:44    Src port: 1029
Jan  7 18:30:44    Dst port: 20
Jan  7 18:30:44    TCP flags: 0x0
Jan  7 18:30:44    IP proto num: 6
Jan  7 18:30:44    TOS: 0x0
Jan  7 18:30:44    Src AS: 64514
Jan  7 18:30:44    Dst AS: 64513
Jan  7 18:30:44    Src netmask len: 16
Jan  7 18:30:44    Dst netmask len: 24
```

Remote cflowd Server

Files created on the remote cflowd server

```
ping# ls /usr/local/arts/data/cflowd/flows
10.1.83.1.flows.0       10.1.83.1.flows.4       10.1.83.1.flows.8
10.1.83.1.flows.1       10.1.83.1.flows.5       10.1.83.1.flows.9
10.1.83.1.flows.2       10.1.83.1.flows.6
10.1.83.1.flows.3       10.1.83.1.flows.7
```

FreeBSD server running CAIDA cflowd package

Raw Flows on the cflowd Server

Viewing the raw flows on the remote cflowd server

```
ping# flowdump 10.1.83.1.flows.0
FLOW
  index:           0xc7ffff
  router:          10.1.86.1
  src IP:          192.168.46.101
  dst IP:          172.16.3.18
  input ifIndex:   30
  output ifIndex:  40
  src port:        1029
  dst port:        20
  pkts:            1
  bytes:           46
  IP nexthop:      10.1.84.0
  start time:      Mon Jan 7 21:30:12 2002
  end time:        Mon Jan 7 21:30:12 2002
  protocol:        6
  tos:             0
  src AS:          64514
  dst AS:          64513
  src masklen:     16
  dst masklen:     24
  TCP flags:       0x0
  engine type:     0
  engine id:       0
```

FreeBSD server running CAIDA cflowd package

Principal NetFlow Benefits

SERVICE PROVIDER

- Peering arrangements
- SLA VPN user reporting
- Usage-based billing
- DoS/worm detection
- Traffic engineering
- Troubleshooting

ENTERPRISE

- Internet access monitoring (protocol distribution, traffic origin/destination)
- Associate cost of IT to departments
- More scalable than RMON
- DoS/worm detection
- Policy compliance monitoring

Open Source Tools for NetFlow Analysis —The OSU Flow-Tools

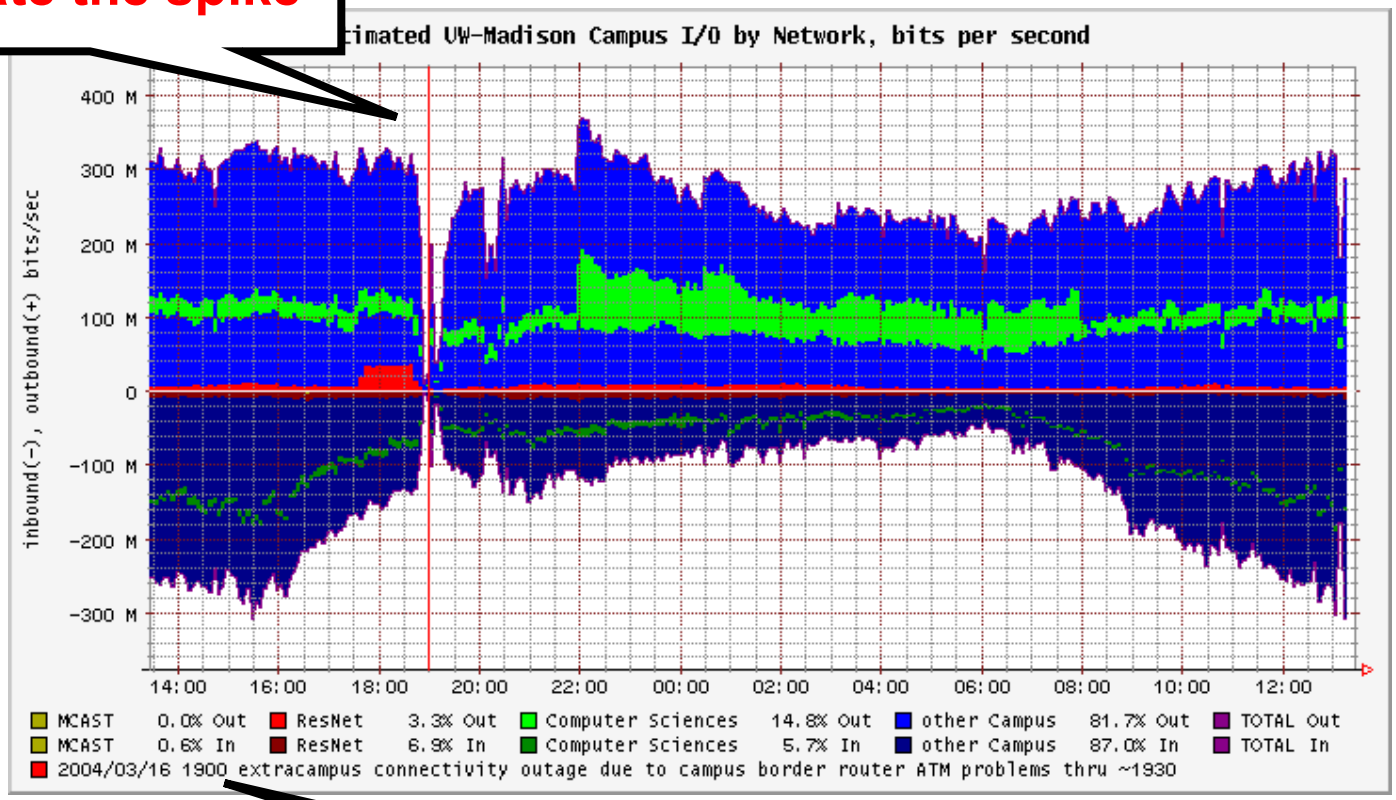
- Open source NetFlow collection and retrieval tools
- Developed and maintained by Mark Fullmer, available from http://www.splintered.net/sw/flow-tools/
- Runs on common *NIX platforms (Linux, FreeBSD, Mac OS/X, Solaris, etc.)
- Command-line tools allow for very display/sorting of specific criteria (source/dest IP, source/dest ASN, protocol, port, etc.)
- Data can be batched and imported into database such as Oracle, MySQL, Postgres, etc.
- Can be combined with other tools to provide visualization of traffic patterns
- Many other useful features - check it out today!

Open Source Tools for NetFlow Analysis Visualization—FlowScan

- Open source NetFlow graphing/visualization tools
- Developed and maintained by Dave Plonka, available from http://net.doit.wisc.edu/~plonka/FlowScan/
- Runs on common *NIX platforms (Linux, FreeBSD, Mac OS/X, Solaris, etc.)
- Makes use of NetFlow data collected via flow-tools to build traffic graphs
- Top-talkers by subnet, other types of reports supported
- Makes use of RRDTool for graphing
- Add-ons such as JKFlow module allow more detailed graphing

Open Source Tools for NetFlow Analysis Visualization—FlowScan

Investigate the spike

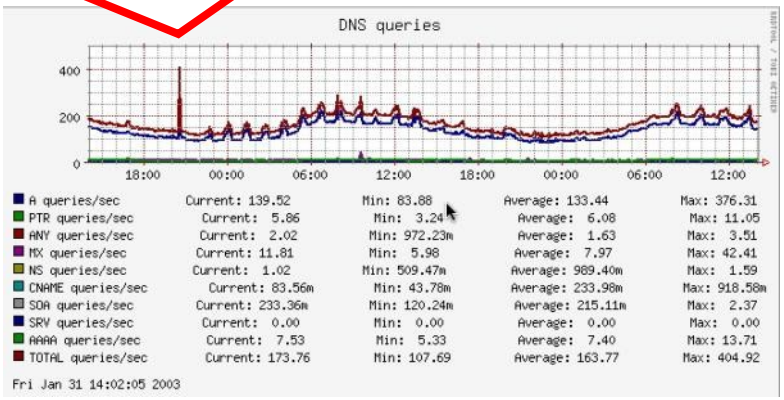


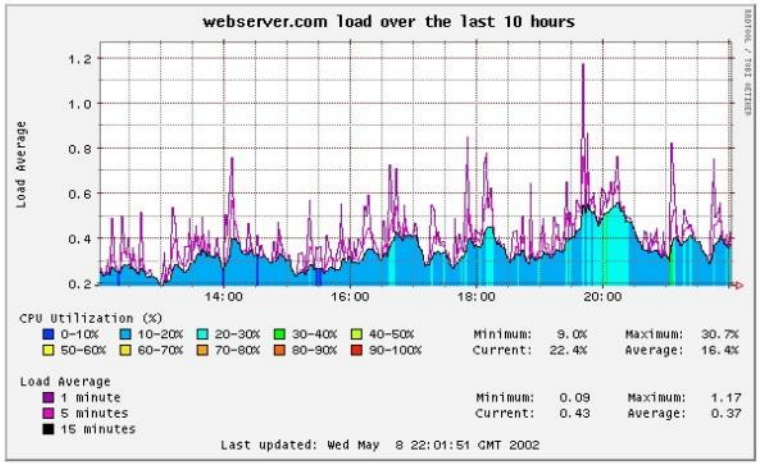
An identified cause of the outage

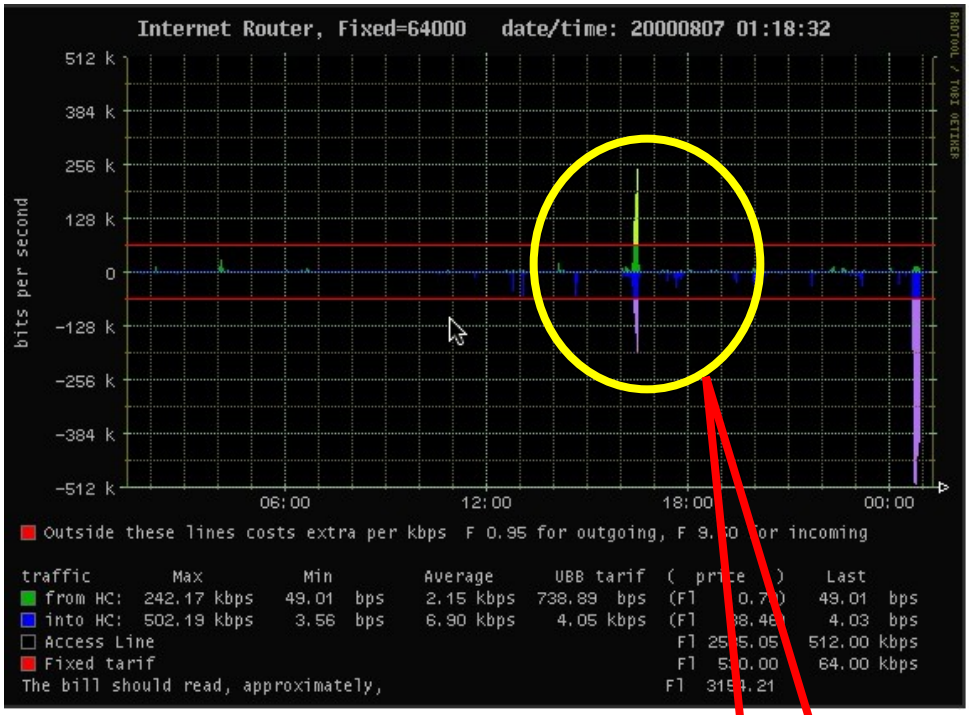
Source: University of Wisconsin

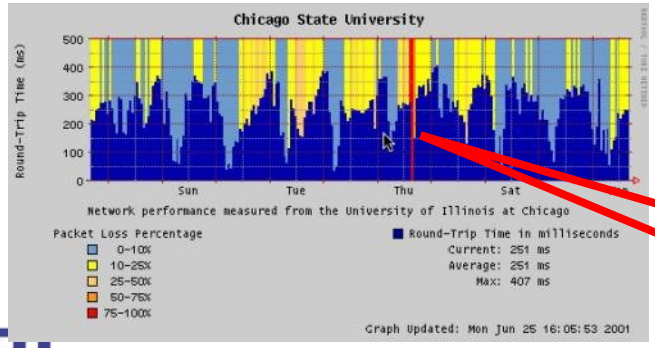
Other Visualization Techniques Using SNMP Data with RRDTool

Anomaly for DNS Queries







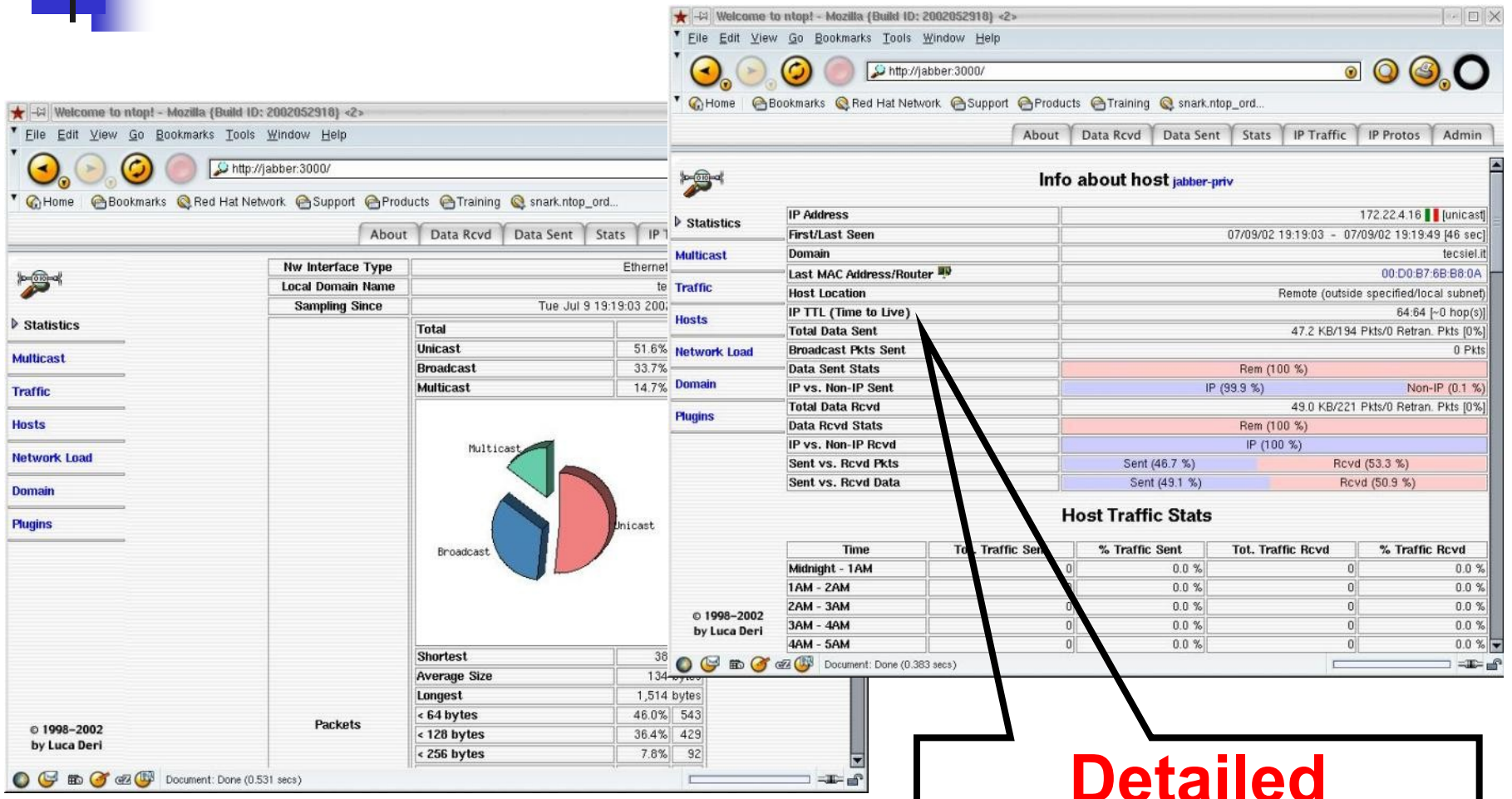


Thru'put Spike

RTT Spike

Source: http://people.ee.ethz.ch/~oetiker/webtools/rrdtool/

Displaying RMON—ntop Examples



Source: http://www.ntop.org

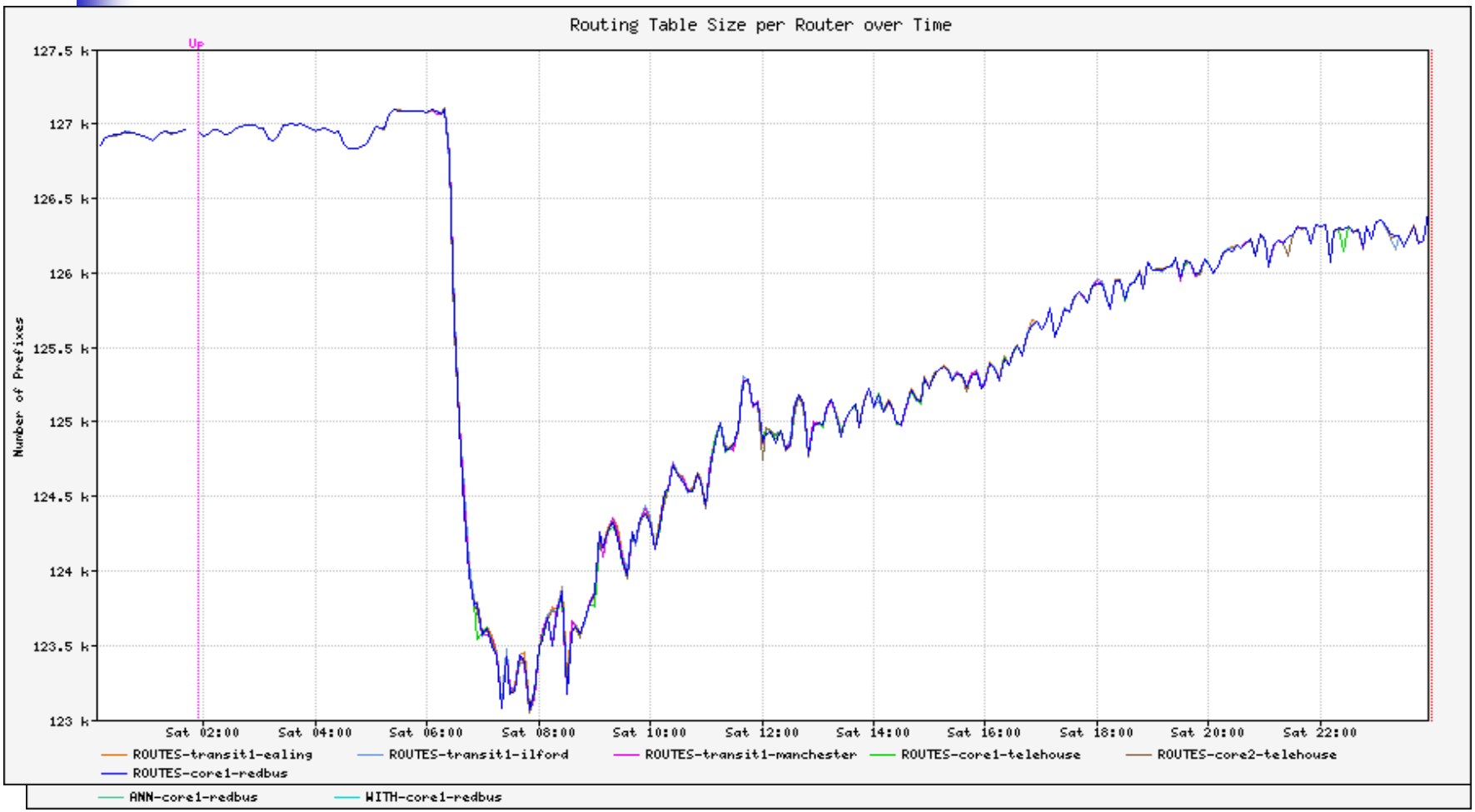
Detailed Analysis i.e. TTL

Packet Clearing House

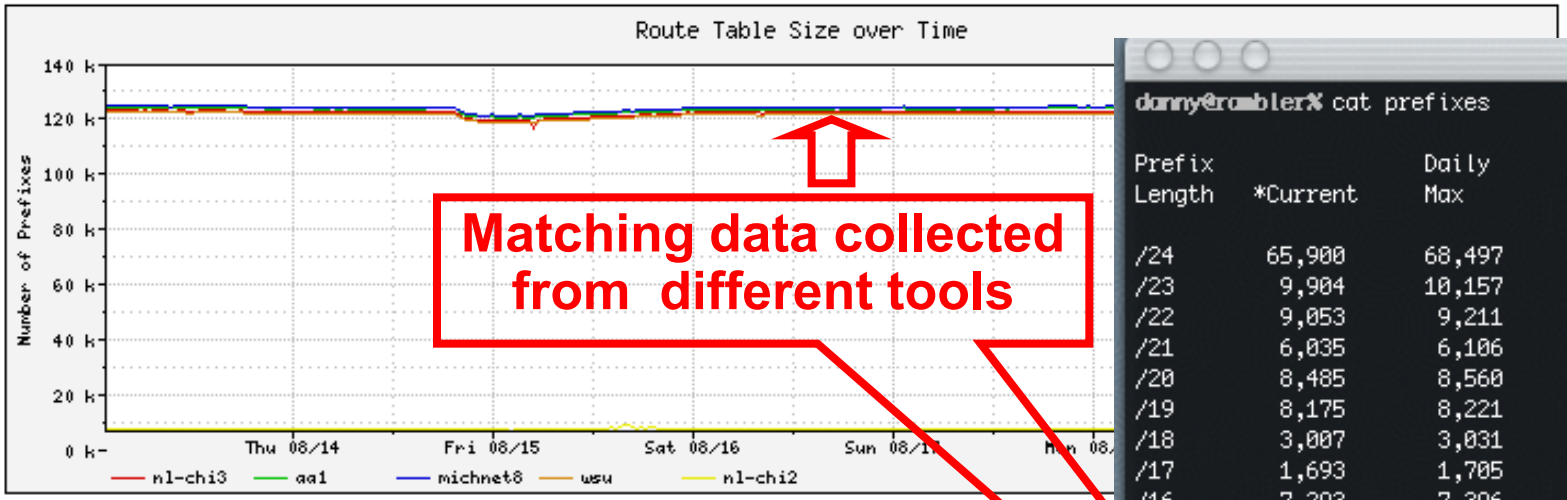
Alcatel·Lucent

SUBISU

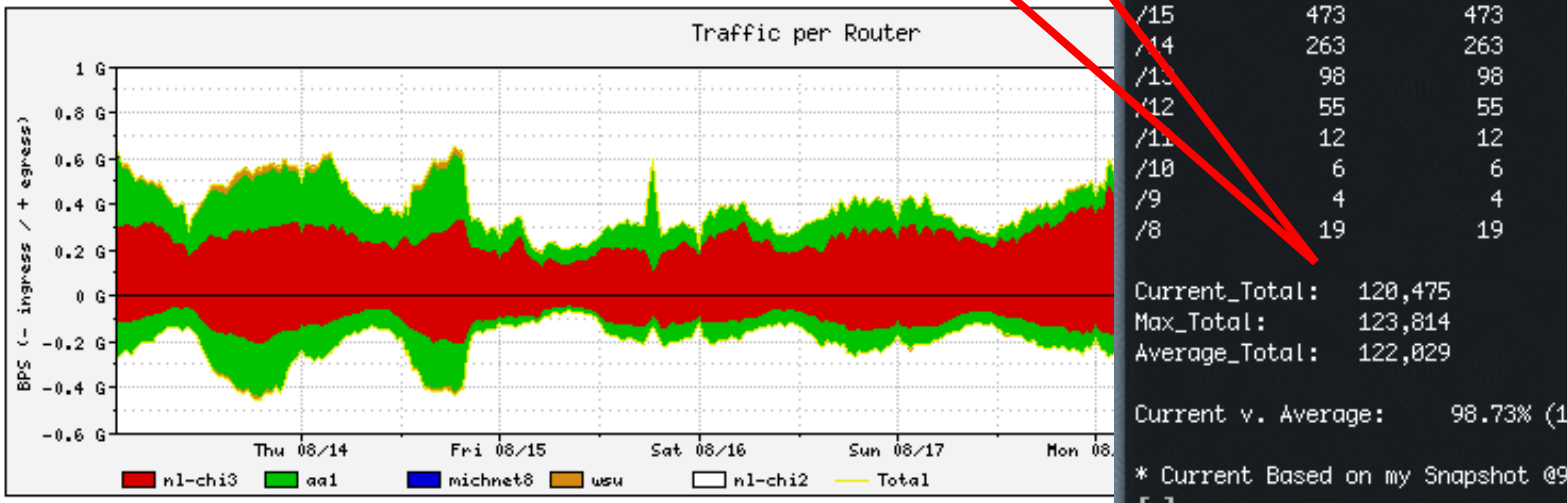
BGP Example—SQL Slammer



Correlating NetFlow and Routing Data



Matching data collected from different tools



tcsh — tcsh

```
donny@rambler% cat prefixes

Prefix                Daily       Daily
Length   *Current     Max         Average

/24        65,900      68,497      67,259
/23         9,904      10,157      10,027
/22         9,053       9,211       9,110
/21         6,035       6,106       6,045
/20         8,485       8,560       8,487
/19         8,175       8,221       8,161
/18         3,007       3,031       3,005
/17         1,693       1,705       1,690
/16         7,293       7,396       7,326
/15           473         473         469
/14           263         263         262
/13            98          98          97
/12            55          55          54
/11            12          12          11
/10             6           6           5
/9              4           4           3
/8             19          19          18

Current_Total:   120,475
Max_Total:       123,814
Average_Total:   122,029

Current v. Average:     98.73% (1554 prefixes)

* Current Based on my Snapshot @9P MDT 8.14.2003
[~]
donny@rambler%
```

Syslog

- De facto logging standard for hosts, network infrastructure devices, supported in all Cisco routers and switches
- Many levels of logging detail available—choose the level(s) which are appropriate for each device/situation
- Logging of ACLs is generally contraindicated due to CPU overhead—NetFlow provides more info, doesn't max the box
- Can be used in conjunction with Anycast and databases such as MySQL (http://www.mysql.com) to provide a scalable, robust logging infrastructure
- Different facility numbers allows for segregation of log info based upon device type, function, other criteria
- Syslog-ng from http://www.balabit.com/network-security/syslog-ng/ adds a lot of useful functionality—HOW-TO located at http://www.campin.net/newlogcheck.html

Local Log Files

- Local log files are useful for:
  - Detecting problems on the router
  - Monitoring the system usage by friendly users
  - Monitoring normal events
- Local log files are not useful for:
  - Monitoring the activity of attackers who have compromised your system

A good attacker will erase the evidence of their activity from local log files!

Remote Log Files

- Logging to a remote host has several advantages:
  - Initial attacker activity is available in the log
    - Remote logging is difficult to stop prior to the compromise
    - Attacker can stop remote logging once the system is compromised
    - The lack of remote logging can be an indication of a problem
  - Remote logs from multiple systems can be consolidated
    - You can monitor and coordinate events between multiple systems

What is Syslog?

- Operating systems and applications generate a multitude of log messages about a variety of things
  - Syslog was developed as a generic logging server to accept, categorize, and record log messages
  - As systems became more complex, a method was needed to forward log messages to a remote syslog server and consolidate messages from multiple hosts
  - BSD Syslog Protocol
    - Outlined in RFC 3164
    - Specifies the format and content of remote syslog messages

Syslog Facilities (1 of 2)

- Each message has a *facility* used to categorize the type of message generated
- The router specifies the facility to which each message belongs

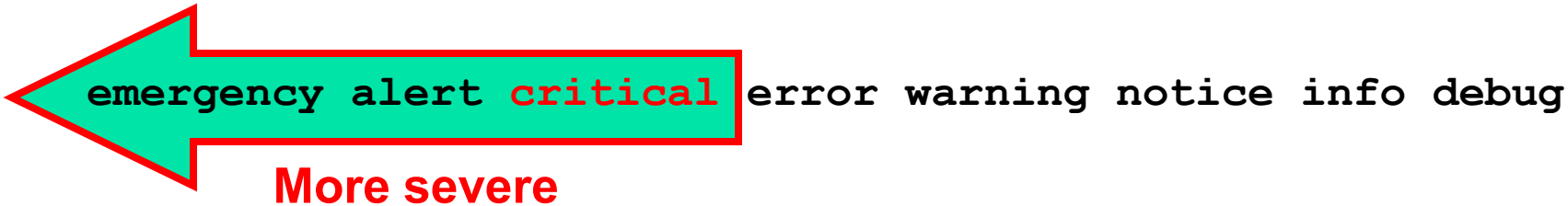
Facility	Description
Any	Any facility
Authorization	Any authorization attempt
Change-log	Any change to the configuration
Conflict-log	Messages generated when configuration conflicts with the hardware
Cron	Cron daemon
Daemon	Various system daemons

Syslog Facilities (2 of 2)

Facility	Description
Firewall	Firewall filtering subsystem
Interactive-commands	Commands executed in the CLI
Kernel	Messages generated by the JUNOS software kernel
PFE	Messages generated by the PFE
User	Messages from user processes
Local0 – Local7	*Local-use* facilities

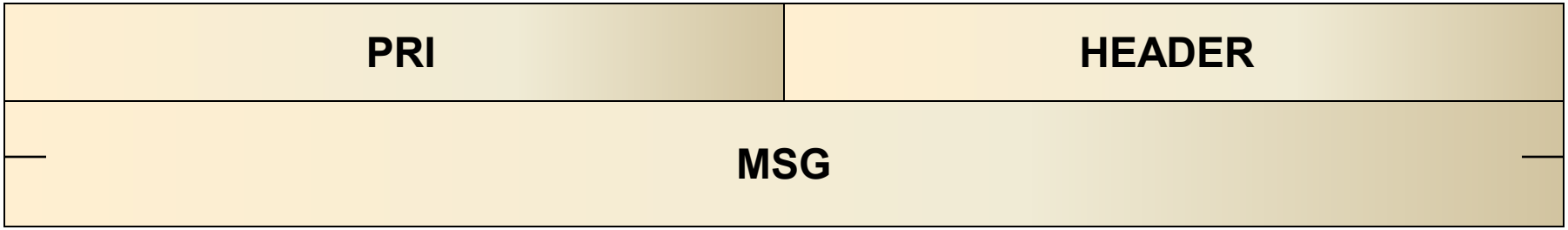
Syslog Severity

- Each message has a *severity* used to prioritize its importance
  - Setting a facility and severity level causes the router to log all messages for that severity at the specified level and above
    - For example, logging at the `critical` level also causes `alert` and `emergency` messages to be logged



Syslog Packet Format

- Format and content of messages defined in RFC 3164
  - Messages sent on UDP port 514
  - No minimum size
  - Maximum size is 1024 bytes
- Messages consist of three text strings
  - PRI (Priority)
  - HEADER
  - MSG (Message)



Overriding the Remote Facility

- By default, syslog messages are sent with their normal BSD-specified facility and various local facilities
  - You can override the message facility

```
[edit system]
lab@R1# show
syslog {
    host 10.1.10.2 {
        authorization info;
        change-log info;
        interactive-commands info;
        facility-override local7;
        log-prefix Security;
    }
}
```

Security Cautions

- Caution:
  - Syslog messages can contain sensitive information in cleartext
    - User authentication messages when logging the authorization facility
    - Passwords entered into the configuration when logging the interactive-commands facility
  - Consider sending syslog messages only on the out-of-band management network
  - Compromise of the remote syslog server might give an attacker enough information to compromise the router!

Good things to log

- All login attempts
  - Successful or not
- All commands typed
  - So you know who did what and when
  - Helps Identifying "training issues" as well ☺
- Availability issues
  - Interface status change (not dialup)
  - More critical for core routers
  - BGP peering changes
  - OSPF neighbor changes?

NTP

Benefits of Deploying NTP

- Very valuable on a global network with network elements in different time zones
- Easy to correlate data from a global or a sizable network with a consistent time stamp
- NTP based timestamp allows to trace security events for chronological forensic work
- Any compromise or alteration is easy to detect as network elements would go out of sync with the main 'clock'
- Did you there is an NTP MIB? Some think that we may be able to use "NTP Jitter" to watch what is happening in the network.

Local System Time

- In a security situation, you must have a consistent concept of time across the network (it does not have to be the correct time, just consistent)
- Choose UTC/GMT or Head office Time Zone
- NTP was developed to synchronize large numbers of network devices to a consistent, accurate time reference
- Local and remote log files are stamped with the local system time
  - Event correlation is easier if all devices are synchronized
  - Law enforcement officials might need copies of these logs

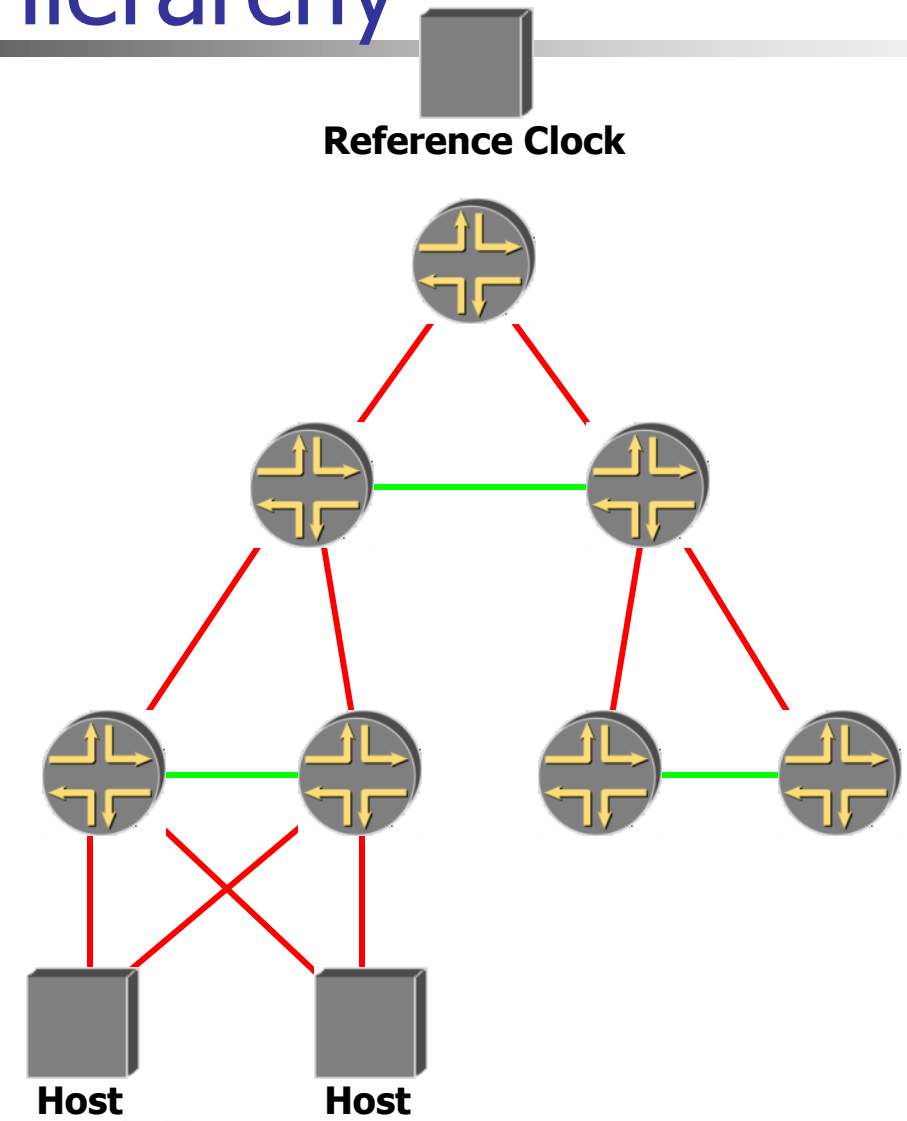
Network Time Protocol

- NTPv3: Network Time Protocol (Version 3) Specification, Implementation and Analysis (RFC 1305—March 1992)
  - Defines a protocol to keep accurate, synchronized time between network devices
    - Uses UDP port 123
  - Additional features incorporated in NTPv4
    - DES encryption
    - Not an IETF standard, but widely supported
    - Backwards compatible with NTP v3

Three NTP Modes

- Three modes:
  - Client mode
    - Client synchronizes local time one way to remote server
  - Symmetric active mode
    - Equal peer systems synchronize each other's local time
  - Broadcast mode
    - Server sends periodic broadcast/multicast messages on broadcast-capable media
    - Clients receives broadcast/multicast messages and synchronize local time

NTP Hierarchy



Client Mode

Symmetric Active Mode

Broadcast Mode

Stratum

Stratum	Min accuracy
1	1.0×10^{-11}
2	1.6×10^{-8}
3	4.6×10^{-6}

NTP Security

- NTP security
  - NTP relies on the number of connected hosts to:
    - Receive accurate time information
    - Isolate participants whose clock is incorrect
  - NTP supports MD5 and DES authentication
  - NTP without authentication on the public network is subject to spoofing
    - Create the appropriate filters to block incoming unsolicited information
    - Consider using the management network for NTP traffic

NTP Boot Server

- NTP particulars:
  - NTP will not synchronize with a peer whose time is very different
    - Tiny offsets are adjusted normally
    - Small offsets are *slewed* (adjusted slowly)
    - Larger offsets are *stepped* (set anew)
    - Huge offsets are rejected outright
  - To synchronize the initial time:
    - Use an NTP boot server
    - When the router is booted a request is issued to the boot server to get the initial reference time

```
[edit system]
lab@R1# show
ntp {
    boot-server 10.1.10.2;
}
```

Client Configuration

```
[edit system]
lab@R1# show
ntp {
    boot-server 10.1.10.2;
    server 10.1.10.2 version 3 prefer;
    server 10.1.9.2;
}
```

Symmetric Active Mode Configuration

```
[edit system]
lab@R1# show
ntp {
    boot-server 10.1.10.2;
    peer 10.1.10.2 version 3 prefer;
    peer 10.1.9.2;
}
```

Broadcast Mode Configuration

```
[edit system]
lab@R1# show
ntp {
    boot-server 10.1.10.2;
    server 10.1.10.2 version 3 prefer;
    peer 10.1.9.2;
    broadcast 224.0.1.1;
    broadcast 10.1.2.255 version 3;
}
```

Broadcast Client Configuration

```
[edit system]
lab@R1# show
ntp {
    boot-server 10.1.10.2;
    broadcast-client
}

[edit system]
lab@R1# show
ntp {
    boot-server 10.1.10.2;
    multicast-client
}
```

Authentication

- Authentication of time synchronization
  - All NTP modes can use authenticated connections
    - Prevents spoofing
  - Supports two types of encrypted/hashed authentication algorithms
    - DES
    - MD5
  - Easy to configure
    - No real reason not to use authentication

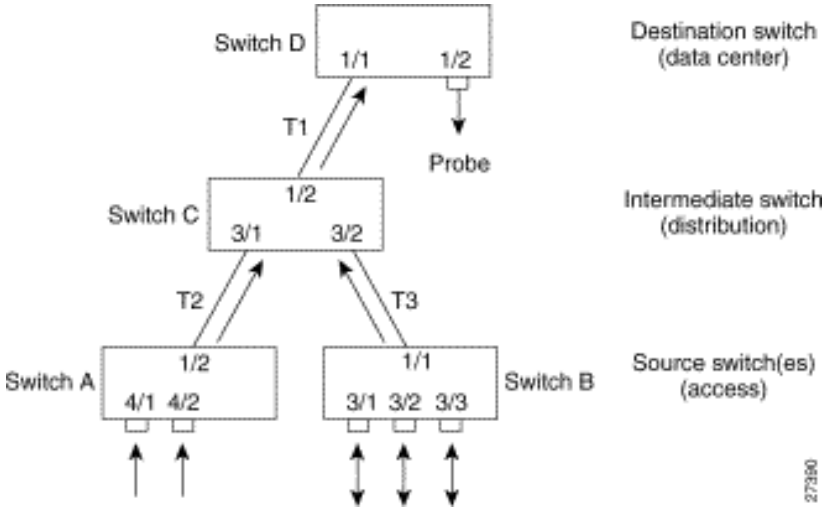
Utilizing Packet Capture

- SPAN/RSPAN (6500/7600, 4K, 2900,), copy/capture VACLs (6500/7600), IP Traffic Export (software-based routers) are all used to get packets to analysis systems
- SPAN/RSPAN and copy/capture VACLs do not have measurable performance impact; IP Traffic Export can delay processing of traffic outbound from the router, based upon the volume of traffic to be replicated
- A *NIX box running tcpdump is a common method of capturing packets, with analysis performed offline using additional open-source tools such as Ethereal
- The Cisco NAM-2 captures packets via SPAN/RSPAN or copy/capture VACLs on the 6500/7600; it can perform basic on-board analysis, but captures are typically saved and downloaded for use in Ethereal, Network General Sniffer, etc.
- Packet capture is generally undertaken after a macro-level indication of an issue via SNMP, NetFlow, etc.

Utilizing Packet Capture (cont.)

- Packet capture should take place at key points in the topology such as distribution gateways, IDC switch meshes, desktop access switch meshes, and in some cases, the core
- It is important to be as specific as possible when capturing packets; at high rates of speed, the amount of information can be overwhelming
- There's lots of garbage out there - 'weird' packets are often perfectly explicable, in context
- It's extremely important to ensure that traffic is captured bidirectionally - or, if this isn't possible, the observer must know about the unidirectionality of the capture and take it into account when analyzing the captured traffic
- Conversely, it's important to avoid capturing duplicate traffic, especially in complex topologies

Packet Capture Example - CatOS RSPAN

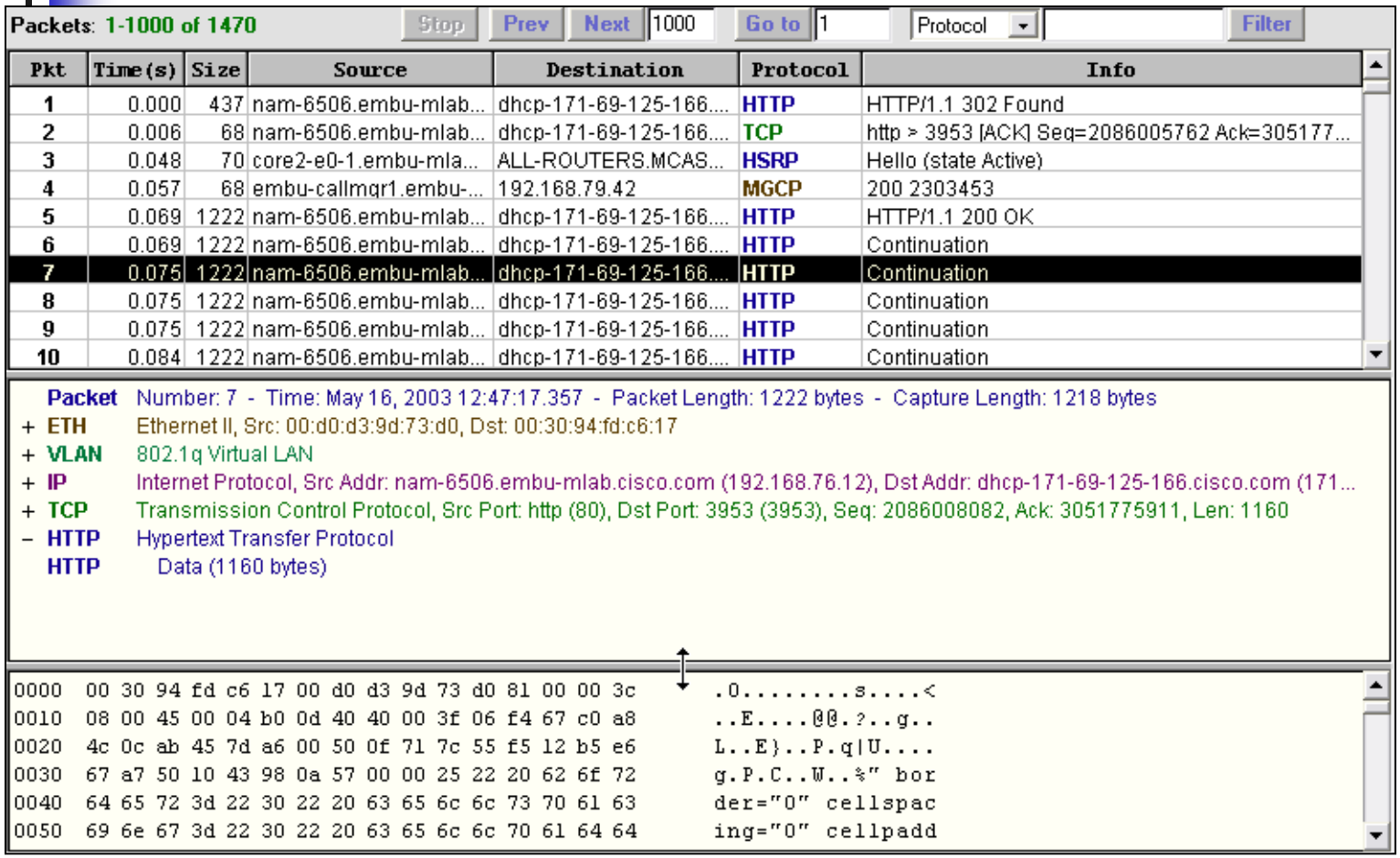


Switch	Ports	RSPAN VLAN	Direction	RSPAN CLI Commands
A (source)	4/1, 4/2	901	Ingress	**set rspan source 4/1-2 901 rx**
B (source)	3/1, 3/2, 3/3	901	Bidirectional	**set rspan source 3/1-3 901**
C (intermediate)	-	901	-	No RSPAN CLI command needed
D (destination)	1/2	901	-	**set rspan destination 1/2 901**

Packet Capture Example - tcpdump

```
tcpdump -lllvvvnxxXX -s 1500 -i en1
tcpdump: listening on en1, link-type EN10MB (Ethernet), capture size 1500 bytes
..
07:10:25.740130 IP (tos 0x0, ttl  64, id 15460, offset 0, flags [none], length: 59) 10.25.7.122.58607
    > 172.17.168.183.53: [udp sum ok]  15197+ A? delta.mac.com. (31)
        0x0000:  0005 31a0 3414 000d 93f0 c5bc 0800 4500  ..1.4.........E.
        0x0010:  003b 3c64 0000 4011 d8bd 0a19 077a ab46  .;<d..@......z.F
        0x0020:  a8b7 e4ef 0035 0027 bfb9 3b5d 0100 0001  .....5.'..;]....
        0x0030:  0000 0000 0000 0564 656c 7461 036d 6163  .......delta.mac
        0x0040:  0363 6f6d 0000 0100 01                   .com.....
07:10:25.829524 IP (tos 0x0, ttl  56, id 14524, offset 0, flags [DF], length: 256) 172.17.168.183.53
    > 10.25.7.122.58607: [udp sum ok]  15197 q: A? delta.mac.com. 2/4/4 delta.mac.com. CNAME
    idisk.mac.com., idisk.mac.com. A 17.250.248.77 ns: mac.com. NS nserver4.apple.com., mac.com. NS
    nserver.apple.com., mac.com. NS nserver2.apple.com., mac.com. NS nserver3.apple.com. ar:
    nserver.apple.com. A 17.254.0.50, nserver2.apple.com. A 17.254.0.59, nserver3.apple.com. A
    17.112.144.50, nserver4.apple.com. A 17.112.144.59 (228)
        0x0000:  000d 93f0 c5bc 0005 31a0 3414 0800 4500  ........1.4...E.
        0x0010:  0100 38bc 4000 3811 a3a0 ab46 a8b7 0a19  ..8.@.8....F....
        0x0020:  077a 0035 e4ef 00ec c78e 3b5d 8180 0001  .z.5......;]....
        0x0030:  0002 0004 0004 0564 656c 7461 036d 6163  .......delta.mac
        0x0040:  0363 6f6d 0000 0100 01c0 0c00 0500 0100  .com............
        0x0050:  0006 ea00 0805 6964 6973 6bc0 12c0 2b00  ......idisk...+.
        0x0060:  0100 0100 000d da00 0411 faf8 4dc0 1200  ............M...
        0x0070:  0200 0100 0222 ab00 1108 6e73 6572 7665  ....."....nserve
        0x0080:  7234 0561 7070 6c65 c016 c012 0002 0001  r4.apple........
        0x0090:  0002 22ab 000a 076e 7365 7276 6572 c058  .."....nserver.X
        0x00a0:  c012 0002 0001 0002 22ab 000b 086e 7365  ........"....nse
        0x00b0:  7276 6572 32c0 58c0 1200 0200 0100 0222  rver2.X........"
        0x00c0:  ab00 0b08 6e73 6572 7665 7233 c058 c06c  ....nserver3.X.l
        0x00d0:  0001 0001 0001 86fa 0004 11fe 0032 c082  .............2..
        0x00e0:  0001 0001 0001 86fa 0004 11fe 003b c099  .............;..
        0x00f0:  0001 0001 0001 86fa 0004 1170 9032 c04f  ...........p.2.O
        0x0100:  0001 0001 0002 9995 0004 1170 903b       ...........p.;
```

Packet Capture Example - Wireshark



Packets: 1-1000 of 1470 | Stop | Prev | Next | 1000 | Go to | 1 | Protocol | Filter

Pkt	Time(s)	Size	Source	Destination	Protocol	Info
1	0.000	437	nam-6506.embu-mlab...	dhcp-171-69-125-166....	HTTP	HTTP/1.1 302 Found
2	0.006	68	nam-6506.embu-mlab...	dhcp-171-69-125-166....	TCP	http > 3953 [ACK] Seq=2086005762 Ack=305177...
3	0.048	70	core2-e0-1.embu-mla...	ALL-ROUTERS.MCAS...	HSRP	Hello (state Active)
4	0.057	68	embu-callmgr1.embu-...	192.168.79.42	MGCP	200 2303453
5	0.069	1222	nam-6506.embu-mlab...	dhcp-171-69-125-166....	HTTP	HTTP/1.1 200 OK
6	0.069	1222	nam-6506.embu-mlab...	dhcp-171-69-125-166....	HTTP	Continuation
7	0.075	1222	nam-6506.embu-mlab...	dhcp-171-69-125-166....	HTTP	Continuation
8	0.075	1222	nam-6506.embu-mlab...	dhcp-171-69-125-166....	HTTP	Continuation
9	0.075	1222	nam-6506.embu-mlab...	dhcp-171-69-125-166....	HTTP	Continuation
10	0.084	1222	nam-6506.embu-mlab...	dhcp-171-69-125-166....	HTTP	Continuation

```
  Packet  Number: 7 - Time: May 16, 2003 12:47:17.357 - Packet Length: 1222 bytes - Capture Length: 1218 bytes
+ ETH     Ethernet II, Src: 00:d0:d3:9d:73:d0, Dst: 00:30:94:fd:c6:17
+ VLAN    802.1q Virtual LAN
+ IP      Internet Protocol, Src Addr: nam-6506.embu-mlab.cisco.com (192.168.76.12), Dst Addr: dhcp-171-69-125-166.cisco.com (171...
+ TCP     Transmission Control Protocol, Src Port: http (80), Dst Port: 3953 (3953), Seq: 2086008082, Ack: 3051775911, Len: 1160
- HTTP    Hypertext Transfer Protocol
  HTTP      Data (1160 bytes)
```

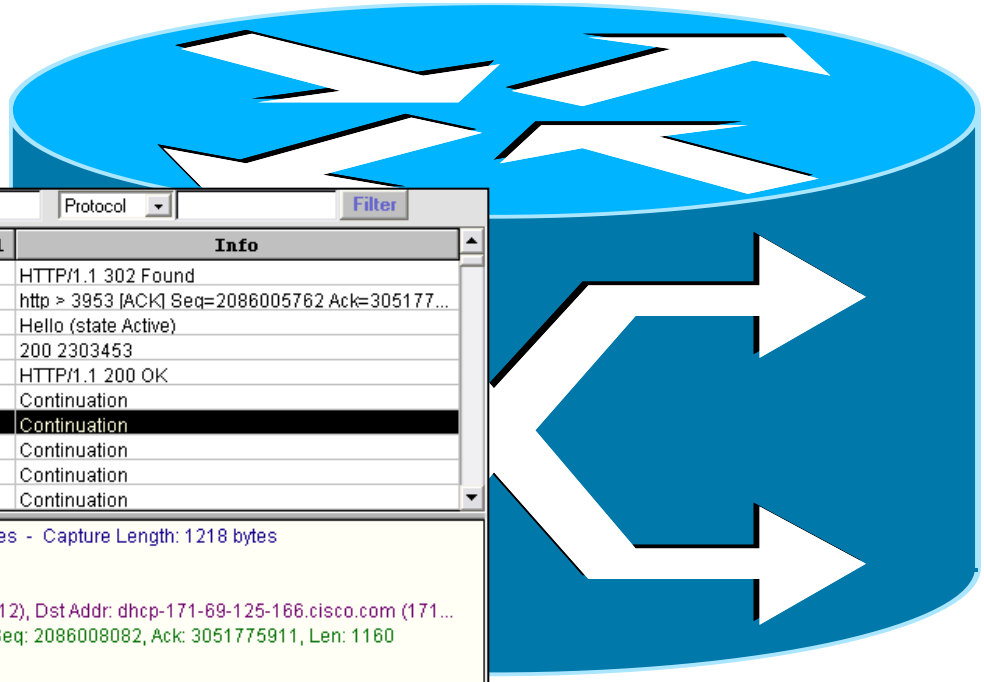
```
0000  00 30 94 fd c6 17 00 d0 d3 9d 73 d0 81 00 00 3c    .0........s....<
0010  08 00 45 00 04 b0 0d 40 40 00 3f 06 f4 67 c0 a8    ..E....@@.?..g..
0020  4c 0c ab 45 7d a6 00 50 0f 71 7c 55 f5 12 b5 e6    L..E}..P.q|U....
0030  67 a7 50 10 43 98 0a 57 00 00 25 22 20 62 6f 72    g.P.C..W..%" bor
0040  64 65 72 3d 22 30 22 20 63 65 6c 6c 73 70 61 63    der="0" cellspac
0050  69 6e 67 3d 22 30 22 20 63 65 6c 6c 70 61 64 64    ing="0" cellpadd
```

Source: http://www.wireshark.org

References

- DoS detection:
  - "Tackling Network DoS on Transit Networks": David Harmelin, DANTE, March 2001 (Describes a detection method based on NetFlow) [http://archive.dante.net/upload/pdf/DiP42.pdf]
  - "Inferring Internet Denial-of-Service Activity": David Moore et al, May 2001; (Described a new method to detect dos attacks, based on the return traffic from the victims, analysed on A /8 network; very interesting reading) [http://www.caida.org/outreach/papers/backscatter/index.xml]
  - "The Spread of the Code Red Worm": David Moore, CAIDA, July 2001 (Using the above to detect how this worm spread across the Internet) [http://www.caida.org/analysis/security/code-red/]
- DoS tracing:
  - "Tracing Spoofed IP Addresses": Rob Thomas, Feb 2001; (Good technical description of using netflow to trace back a flow) [http://www.cymru.com/Documents/tracking-spoofed.html]

Packet Capture Examples



Packets: 1-1000 of 1470 | Stop | Prev | Next | 1000 | Go to | 1 | Protocol | Filter

Pkt	Time(s)	Size	Source	Destination	Protocol	Info
1	0.000	437	nam-6506.embu-mlab...	dhcp-171-69-125-166....	HTTP	HTTP/1.1 302 Found
2	0.006	68	nam-6506.embu-mlab...	dhcp-171-69-125-166....	TCP	http > 3953 [ACK] Seq=2086005762 Ack=305177...
3	0.048	70	core2-e0-1.embu-mla...	ALL-ROUTERS.MCAS...	HSRP	Hello (state Active)
4	0.057	68	embu-callmgr1.embu-...	192.168.79.42	MGCP	200 2303453
5	0.069	1222	nam-6506.embu-mlab...	dhcp-171-69-125-166....	HTTP	HTTP/1.1 200 OK
6	0.069	1222	nam-6506.embu-mlab...	dhcp-171-69-125-166....	HTTP	Continuation
7	0.075	1222	nam-6506.embu-mlab...	dhcp-171-69-125-166....	HTTP	Continuation
8	0.075	1222	nam-6506.embu-mlab...	dhcp-171-69-125-166....	HTTP	Continuation
9	0.075	1222	nam-6506.embu-mlab...	dhcp-171-69-125-166....	HTTP	Continuation
10	0.084	1222	nam-6506.embu-mlab...	dhcp-171-69-125-166....	HTTP	Continuation

```
Packet  Number: 7 - Time: May 16, 2003 12:47:17.357 - Packet Length: 1222 bytes - Capture Length: 1218 bytes
+ ETH   Ethernet II, Src: 00:d0:d3:9d:73:d0, Dst: 00:30:94:fd:c6:17
+ VLAN  802.1q Virtual LAN
+ IP    Internet Protocol, Src Addr: nam-6506.embu-mlab.cisco.com (192.168.76.12), Dst Addr: dhcp-171-69-125-166.cisco.com (171...
+ TCP   Transmission Control Protocol, Src Port: http (80), Dst Port: 3953 (3953), Seq: 2086008082, Ack: 3051775911, Len: 1160
- HTTP  Hypertext Transfer Protocol
HTTP     Data (1160 bytes)
```

```
0000  00 30 94 fd c6 17 00 d0 d3 9d 73 d0 81 00 00 3c    .0........s....<
0010  08 00 45 00 04 b0 0d 40 40 00 3f 06 f4 67 c0 a8    ..E....@@.?..g..
0020  4c 0c ab 45 7d a6 00 50 0f 71 7c 55 f5 12 b5 e6    L..E}..P.q|U....
0030  67 a7 50 10 43 98 0a 57 00 00 25 22 20 62 6f 72    g.P.C..W..%" bor
0040  64 65 72 3d 22 30 22 20 63 65 6c 6c 73 70 61 63    der="0" cellspac
0050  69 6e 67 3d 22 30 22 20 63 65 6c 6c 70 61 64 64    ing="0" cellpadd
```

Wealth of information, L1-L7 raw data for analysis

Source: http://www.ethereal.com, Cisco Systems, Inc.

Tell Me Where to Start From?

1. Flow enablement on the network elements
2. Flow data correlation and analysis
3. SNMP
    1. CPU / Memory utilization
    2. Link usage and display with MRTG
    3. Collect SNMP traps
4. SysLog collection and analysis
5. Monitoring to Routing, DNS queries, etc. [BGP, DNS]
6. Local and remote packet capture facility [Most have it today with sniffer, wireshark]

Homework from Total Visibility

- Define telemetry strategy—ASAP
  - Local and remote
- Need to start deployment today where the most bang for the buck is offered. However, the end goal is to achieve the holistic view
- Telemetry: Deploy, Understand and Practice
  - For any security event – Proactive telemetry during the incident, if 'SECOPS' trained then they can use it with familiarity of 'back of their hand'
- Telemetry builds foundation to be successful with all the other 5 of 6 steps methodology