

# Security Monitoring



## ISP Workshops

# Managing and Monitoring IPv6 Networks

---

- ❑ SNMP Monitoring
- ❑ IPv6-Capable SNMP Management Tools
- ❑ NetFlow Analysis
- ❑ Syslog
- ❑ Keeping accurate time
- ❑ Intrusion Detection
- ❑ Managing the Security Configuration

# Using SNMP for Managing IPv6 Networks



# What is SNMP?

---

- ❑ SNMP – Simple Network Management Protocol
- ❑ Industry standard, hundreds of tools exist to exploit it
- ❑ Present on any decent network equipment
- ❑ Query/response based: GET / SET
- ❑ Monitoring generally uses GET
- ❑ Object Identifiers (OIDs)
- ❑ Keys to identify each piece of data
- ❑ Concept of MIB (Management Information Base)
- ❑ Defines a collection of OIDs

# What is SNMP?

---

- Typical queries
  - Bytes In/Out on an interface, errors
  - CPU load
  - Uptime
  - Temperature or other vendor specific OIDs
- For hosts (servers or workstations)
  - Disk space
  - Installed software
  - Running processes
  - ...
- Windows and UNIX have SNMP agents

# What is SNMP?

---

- UDP protocol, port 161
- Different versions
  - v1 (1988) – RFC1155, RFC1156, RFC1157
    - Original specification
  - v2 – RFC1901 ... RFC1908 + RFC2578
    - Extends v1, new data types, better retrieval methods (GETBULK)
    - Used is version v2c (simple security model)
  - v3 – RFC3411 ... RFC3418 (w/security)
- Typically we use SNMPv2 (v2c)



# SNMP roles

---

- Terminology:
  - Manager (the monitoring station)
  - Agent (running on the equipment/server)

# How does it work?

---

## □ Basic commands

- GET (manager → agent)
  - Query for a value
- GET-NEXT (manager → agent)
  - Get next value (e.g. list of values for a table)
- GET-RESPONSE (agent → manager)
  - Response to GET/SET, or error
- SET (manager → agent)
  - Set a value, or perform action
- TRAP (agent → manager)
  - Spontaneous notification from equipment (line down, temperature above threshold, ...)

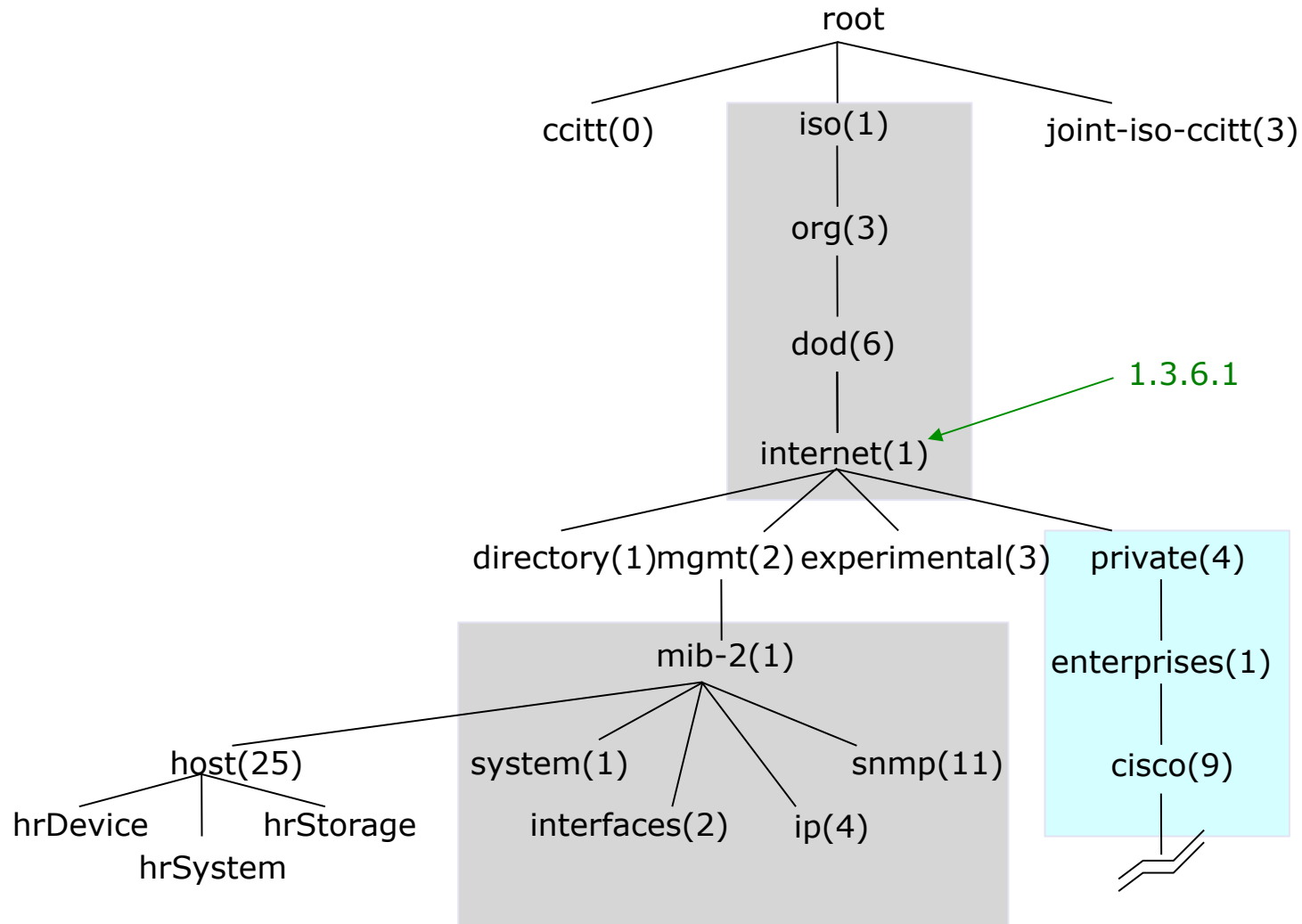


# OIDs and MIBs

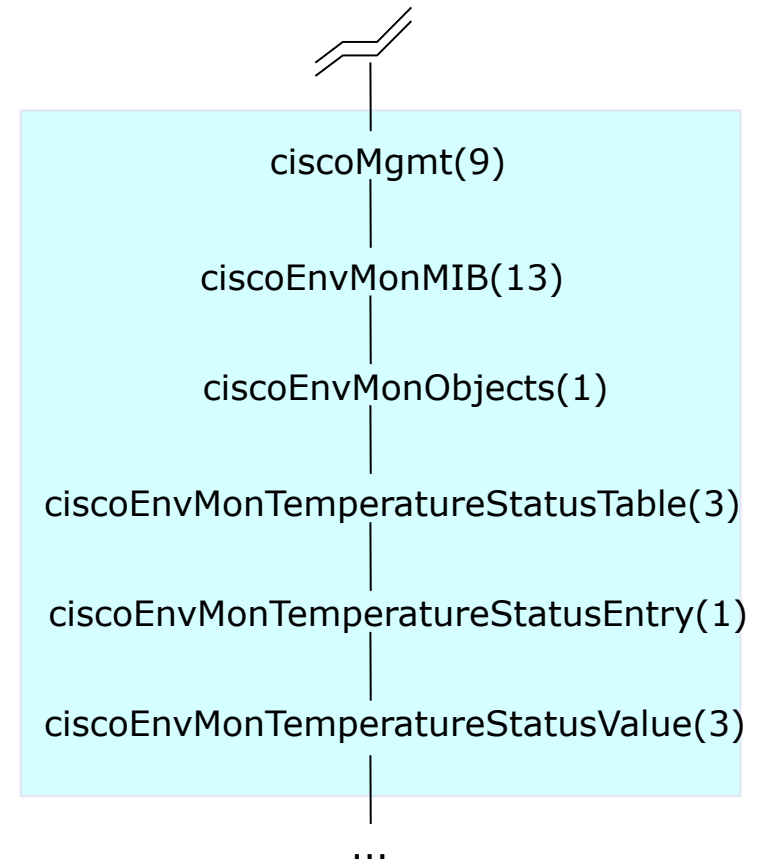
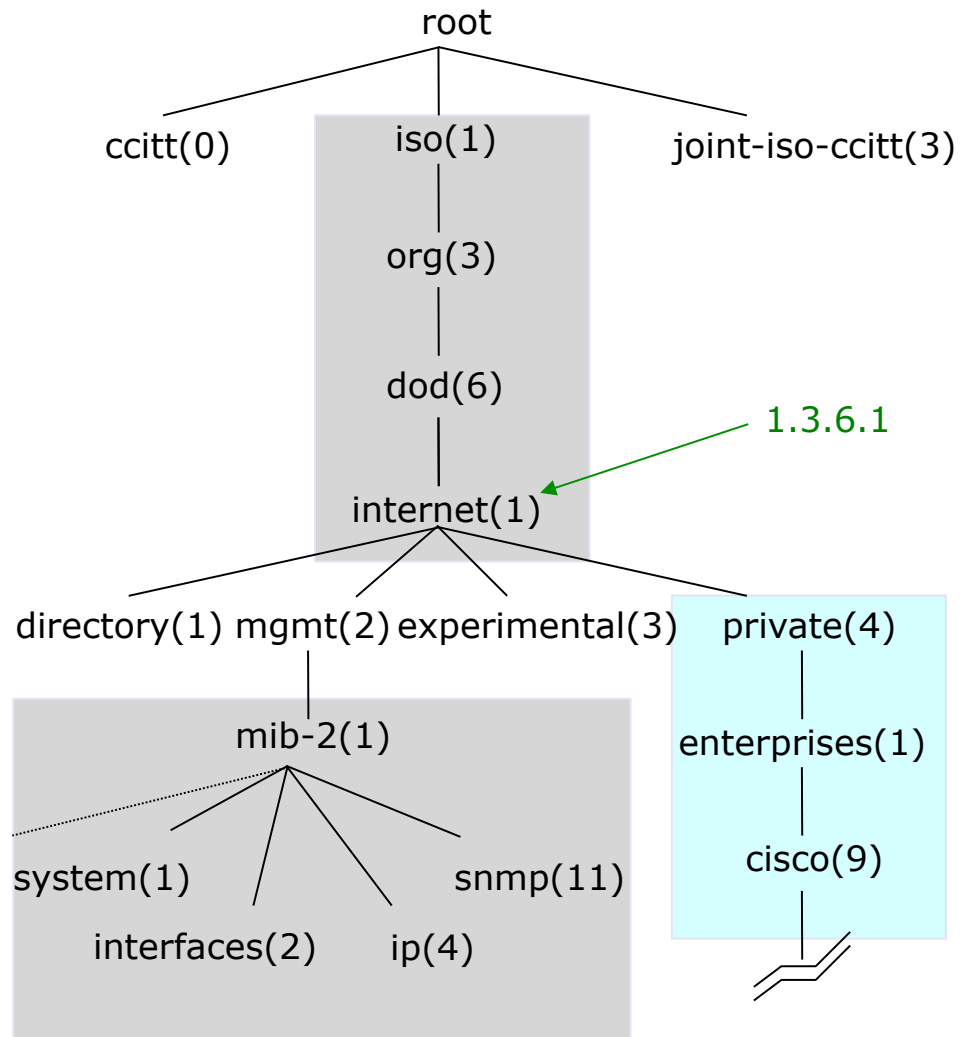
---

- **OID: Object Identifier**
  - A unique key to select a particular item of data in the device
  - The same piece of information is always found at the same OID. That's simple!
  - An OID is a variable-length string of numbers, e.g. 1.3.6.1.2.1.1.3
  - Allocated hierarchically in a tree to ensure uniqueness (similar to DNS)
- **MIB: Management Information Base**
  - A collection of related OIDs
  - A mapping of numeric OIDs to human-readable names

# The MIB Tree



# The MIB Tree



# If E-mail addresses were OIDs...

---

- `user@nsrc.org`
  - Would have been something like:
  - `user@nsrc.enterprises.private.internet.dod.org.iso`
  - `user@99999.1.4.1.6.3.1`
- Except that we write the top-most part at the left:
  - `1.3.6.1.4.1.99999.117.115.101.114`
- Don't worry about the deeply branched tree
  - What matters is that OIDs are unique.
- Ensures vendors don't have conflicting OIDs
- The numeric OID is what gets sent on the wire

# The Internet MIB

---

directory (1)	OSI directory
mgmt (2)	RFC standard objects *
experimental (3)	Internet experiments
private (4)	Vendor-specific *
security (5)	Security
snmpV2 (6)	SNMP internal

- \* Really only two branches of any interest:
  - 1.3.6.1.2.1 = Standard MIBs
  - 1.3.6.1.4.1 = Vendor-specific (proprietary) MIBs

# OIDs and MIBs

---

- Read from left to right
- OID components separated by '.'
  - 1.3.6.1.4.1.9. ...
- Each OID corresponds to a label
  - .1.3.6.1.2.1.1.5 → sysName
- The complete path:
  - .iso.org.dod.internet.mgmt.mib-2.system.sysName
- How do we convert from OIDs to Labels (and vice versa ?)
  - Use of MIBs files!

# MIB files

---

- MIB files define the objects that can be queried, including:
  - Object name
  - Object description
  - Data type (integer, text, list)
- MIB files are structured text, using ASN.1
- Standard MIBs include:
  - MIB-II (RFC1213) – a group of sub-MIBs
  - HOST-RESOURCES-MIB (RFC2790)

# MIBs – SAMPLE

---

```
sysUpTime OBJECT-TYPE
    SYNTAX  TimeTicks
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
        "The time (in hundredths of a second) since the network
        management portion of the system was last re-initialized."
    ::= { system 3 }
```

sysUpTime OBJECT-TYPE

**This defines the object called `sysUpTime`.**

SYNTAX TimeTicks

**This object is of the type `TimeTicks`. Object types are specified in the SMI we mentioned a moment ago**

ACCESS read-only

**This object can only be read via SNMP (i.e., `get-request`); it cannot be changed (i.e., `set-request`).**

STATUS mandatory

**This object must be implemented in any SNMP agent.**

DESCRIPTION

**A description of the object**

```
::= { system 3 }
```

**The `sysUpTime` object is the third branch off of the `system` object group tree.**



## MIB files - 2

---

- MIB files also make it possible to interpret a returned value from an agent
  - For example, the status for a fan could be 1,2,3,4,5,6 – what does it mean ?

# MIBs – SAMPLE

---

```
CiscoEnvMonState ::= TEXTUAL-CONVENTION
    STATUS current
    DESCRIPTION
        "Represents the state of a device being monitored.
        Valid values are:

        normal(1):          the environment is good, such as low
                           temperature.

        warning(2):        the environment is bad, such as temperature
                           above normal operation range but not too
                           high.

        critical(3):       the environment is very bad, such as
                           temperature much higher than normal
                           operation limit.

        shutdown(4):       the environment is the worst, the system
                           should be shutdown immediately.

        notPresent(5):     the environmental monitor is not present,
                           such as temperature sensors do not exist.

        notFunctioning(6): the environmental monitor does not
                           function properly, such as a temperature
                           sensor generates a abnormal data like
                           1000 C.
```

# Querying SNMP agent

---

- Some typical commands for querying:

`snmpget`

`snmpwalk`

`snmpstatus`

`snmptable`

- Syntax:

`snmpXXX -c community -v1 host [oid]`

`snmpXXX -c community -v2c host [oid]`

# Querying SNMP agent

---

## □ Let's take an example

```
snmpstatus -c NetManage -v2c 10.10.0.254
```

```
snmpget -c NetManage -v2c 10.10.0.254 ifNumber.0
```

```
snmpwalk -c NetManage -v2c 10.10.0.254 ifDescr
```

# Querying SNMP agent

---

- Community:
  - A "security" string (password) to define whether the querying manager will have RO (read only) or RW (read write) access
  - This is the simplest form of authentication in SNMP
- OID
  - A value, for example, .1.3.6.1.2.1.1.5.0
  - or its name equivalent: sysName.0
- Let's ask for the system's name (using the OID above)
  - Why the .0? What do you notice?

## SNMP failure: no response?

---

- ❑ The device might be offline or unreachable
- ❑ The device might not be running an SNMP agent
- ❑ The device might be configured with a different community string
- ❑ The device might be configured to refuse SNMP queries from your IP address
- ❑ In all of these cases you will get no response

# IPv6-Capable SNMP Management Tools – MRTG



# MRTG: Multi Router Traffic Grapher

---

- ❑ Tool to monitor traffic load on network links
- ❑ Generates HTML pages with PNG images
- ❑ Almost live visual representation of traffic
- ❑ Available at <http://oss.oetiker.ch/mrtg/>.
- ❑ MRTG is ubiquitous...
- ❑ Uses simple SNMP queries on a regular interval to generate graphs.





# MRTG

---

- ❑ External MRTG readers to interpret data as needed
- ❑ Can build graphs of anything with SNMP MIB like CPU load, disk availability, temperature, etc.
- ❑ Data sources can be anything that provides a counter or gauge value – not necessarily SNMP.
- ❑ For example, graphing round trip times
- ❑ MRTG can be extended to work with RRDTool



## MRTG: Issues

---

- ❑ Generates a new graph every 5 minutes...  
Lots of overhead if lots of graphs.
- ❑ Very few customizable graphing options.
- ❑ Disk space can be an issue.
- ❑ MRTG management is tedious.

# Using MRTG

---

- ❑ Get the required packages
- ❑ Compile and install the packages
- ❑ Make cfg files for router interfaces with cfmaker
- ❑ Create html pages from the cfg files with indexmaker
- ❑ Trigger MRTG periodically from cron or run it in daemon mode

# RRDtool

---

- ❑ Round Robin Database for time series data storage
- ❑ Command line based
- ❑ From the author of MRTG
- ❑ Made to be faster and more flexible
- ❑ Includes CGI and Graphing tools, plus APIs
- ❑ Solves the Historical Trends and Simple Interface problems as well as storage issues



- ❑ Find RRDtool here: <http://oss.oetiker.ch/rrdtool/>

# Defining the Output (Archives)

---

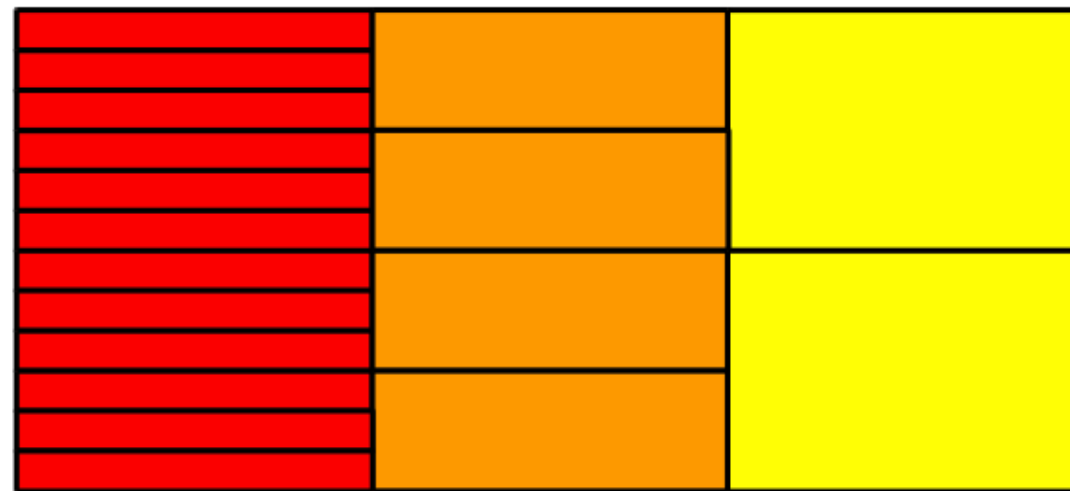
- ❑ RRA:AVERAGE:0.5:1:24
- ❑ RRA:AVERAGE:0.5:6:10
  - RRA = Round Robin Archive
  - AVERAGE = consolidation function
  - 0.5 = up to 50% of consolidated points may be UNKNOWN
  - 1:24 = this RRA keeps each sample (average over one 5 minute primary sample), 24 times (which is 2 hours worth)
  - 6:10 = one RRA keeps an average over every six 5 minute primary samples (30 minutes), 10 times (which is 5 hours worth)
  - Clear as mud!
  - All depends on original step size which defaults to 5 minutes

# RRDtool Database Format

Recent data stored once every 5 minutes for the past 2 hours (1:24)

Old data averaged to one entry per day for the last 365 days (288:365)

--step  
300  
(5 minute  
input step  
size)



RRA  
1:24

RRA  
6:10

RRA  
288:365

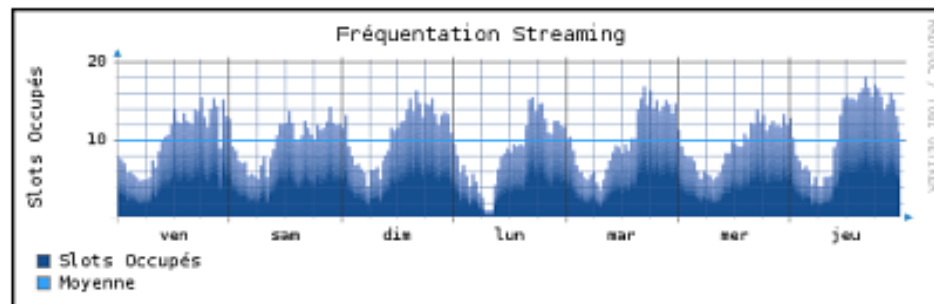
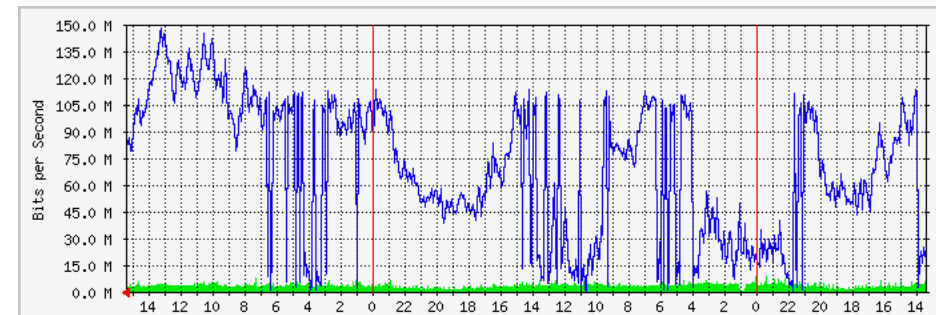
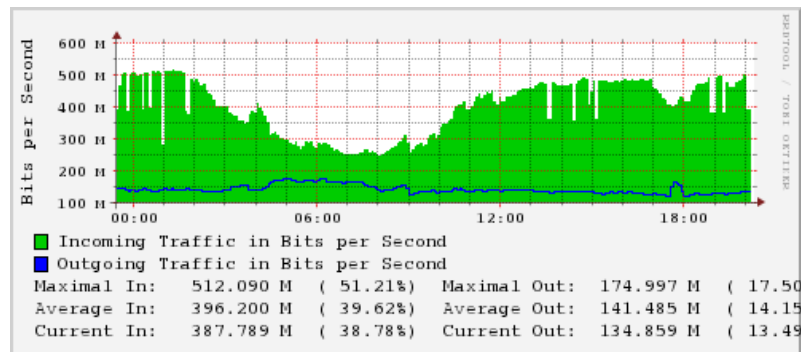
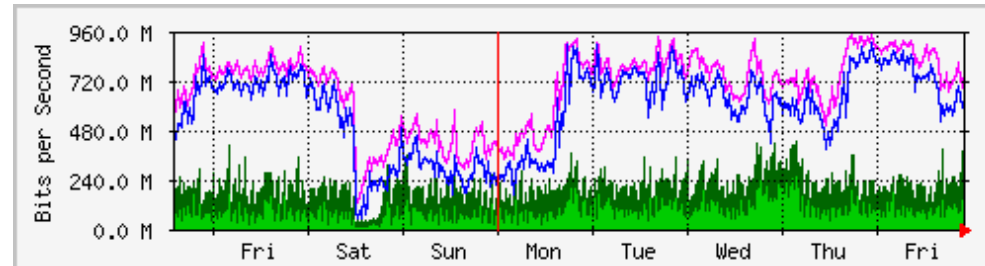
Medium length data averaged to one entry per half hour for the last 5 hours (6:10)

# So simple...

---

- ❑ `rrdtool create /var/nagios/rrd/host0_load.rrd -s 600 DS:1MIN-Load:GAUGE:1200:0:100 DS:5MIN-Load:GAUGE:1200:0:100 DS:15MIN-Load:GAUGE:1200:0:100 RRA:AVERAGE:0.5:1:50400 RRA:AVERAGE:0.5:60:43800`
- ❑ `rrdtool create /var/nagios/rrd/host0_disk_usage.rrd -s 600 DS:root:GAUGE:1200:0:U DS:home:GAUGE:1200:0:U DS:usr:GAUGE:1200:0:U DS:var:GAUGE:1200:0:U RRA:AVERAGE:0.5:1:50400 RRA:AVERAGE:0.5:60:43800`
- ❑ `rrdtool create /var/nagios/rrd/apricot-INTL_Ping.rrd -s 300 DS:ping:GAUGE:600:0:U RRA:AVERAGE:0.5:1:50400 RRA:AVERAGE:0.5:60:43800`
- ❑ `rrdtool create /var/nagios/rrd/host0_total.rrd -s 300 DS:IN:COUNTER:1200:0:U DS:OUT:COUNTER:600:0:U RRA:AVERAGE:0.5:1:50400 RRA:AVERAGE:0.5:60:43800`

# What it looks like...







# Questions?

---

?

# NetFlow Analysis



# What is a Network Flow?

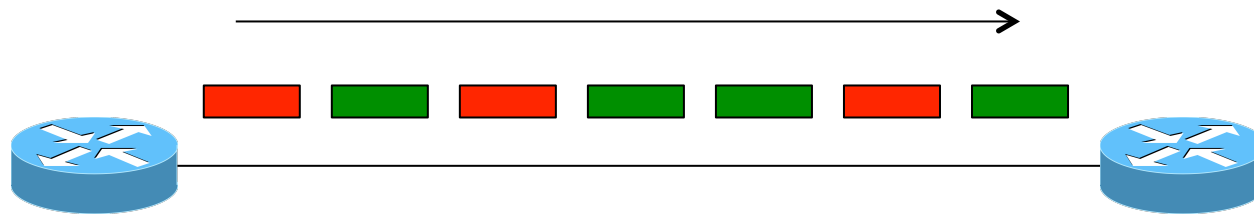
---



- A set of related packets
- Packets that belong to the same transport connection. e.g.
  - TCP, same src IP, src port, dst IP, dst port
  - UDP, same src IP, src port, dst IP, dst port
  - Some tools consider “bidirectional flows”, i.e.  $A \rightarrow B$  and  $B \rightarrow A$  as part of the same flow

[http://en.wikipedia.org/wiki/Traffic\\_flow\\_\(computer\\_networking\)](http://en.wikipedia.org/wiki/Traffic_flow_(computer_networking))

# Simple flows

---



-  = Packet belonging to flow X
-  = Packet belonging to flow Y

# Cisco IOS Definition of a Flow

---

- Unidirectional sequence of packets sharing:
  1. Source IP address
  2. Destination IP address
  3. Source port for UDP or TCP, 0 for other protocols
  4. Destination port for UDP or TCP, type and code for ICMP, or 0 for other protocols
  5. IP protocol
  6. Ingress interface (SNMP ifIndex)
  7. IP Type of Service

# IOS: which of these six packets are in the same flows?

---

	<i>Src IP</i>	<i>Dst IP</i>	<i>Protocol</i>	<i>Src Port</i>	<i>Dst Port</i>
A	1.2.3.4	5.6.7.8	6 (TCP)	4001	80
B	5.6.7.8	1.2.3.4	6 (TCP)	80	4001
C	1.2.3.4	5.6.7.8	6 (TCP)	4002	80
D	1.2.3.4	5.6.7.8	6 (TCP)	4001	80
E	1.2.3.4	8.8.8.8	17 (UDP)	65432	53
F	8.8.8.8	1.2.3.4	17 (UDP)	53	65432

# IOS: which of these six packets are in the same flows?

---

	<i>Src IP</i>	<i>Dst IP</i>	<i>Protocol</i>	<i>Src Port</i>	<i>Dst Port</i>
A	1.2.3.4	5.6.7.8	6 (TCP)	4001	80
B	5.6.7.8	1.2.3.4	6 (TCP)	80	4001
C	1.2.3.4	5.6.7.8	6 (TCP)	4002	80
D	1.2.3.4	5.6.7.8	6 (TCP)	4001	80
E	1.2.3.4	8.8.8.8	17 (UDP)	65432	53
F	8.8.8.8	1.2.3.4	17 (UDP)	53	65432

*What about packets "C" and "D"?*

# Flow Accounting

---

- A summary of all the packets seen in a flow (so far):
  - Flow identification: protocol, src/dst IP/port...
  - Packet count
  - Byte count
  - Start and end times
  - Maybe additional info, e.g. AS numbers, netmasks
- Records traffic volume and type but not content



# Uses and Applications

---

- You can answer questions like:
  - Which user / department has been uploading / downloading the most?
  - Which are the most commonly-used protocols on my network?
  - Which devices are sending the most SMTP traffic, and to where?
- Identification of anomalies and attacks
- More fine-grained visualisation (graphing) than can be done at the interface level

# Working with flows

---

1. Configure device (e.g. router) to generate flow accounting records
2. Export the flows from the device (router) to a collector (PC)
  - Configure protocol version and destination
3. Receive the flows, write them to disk
4. Analyse the flows

Many tools available, both free and commercial



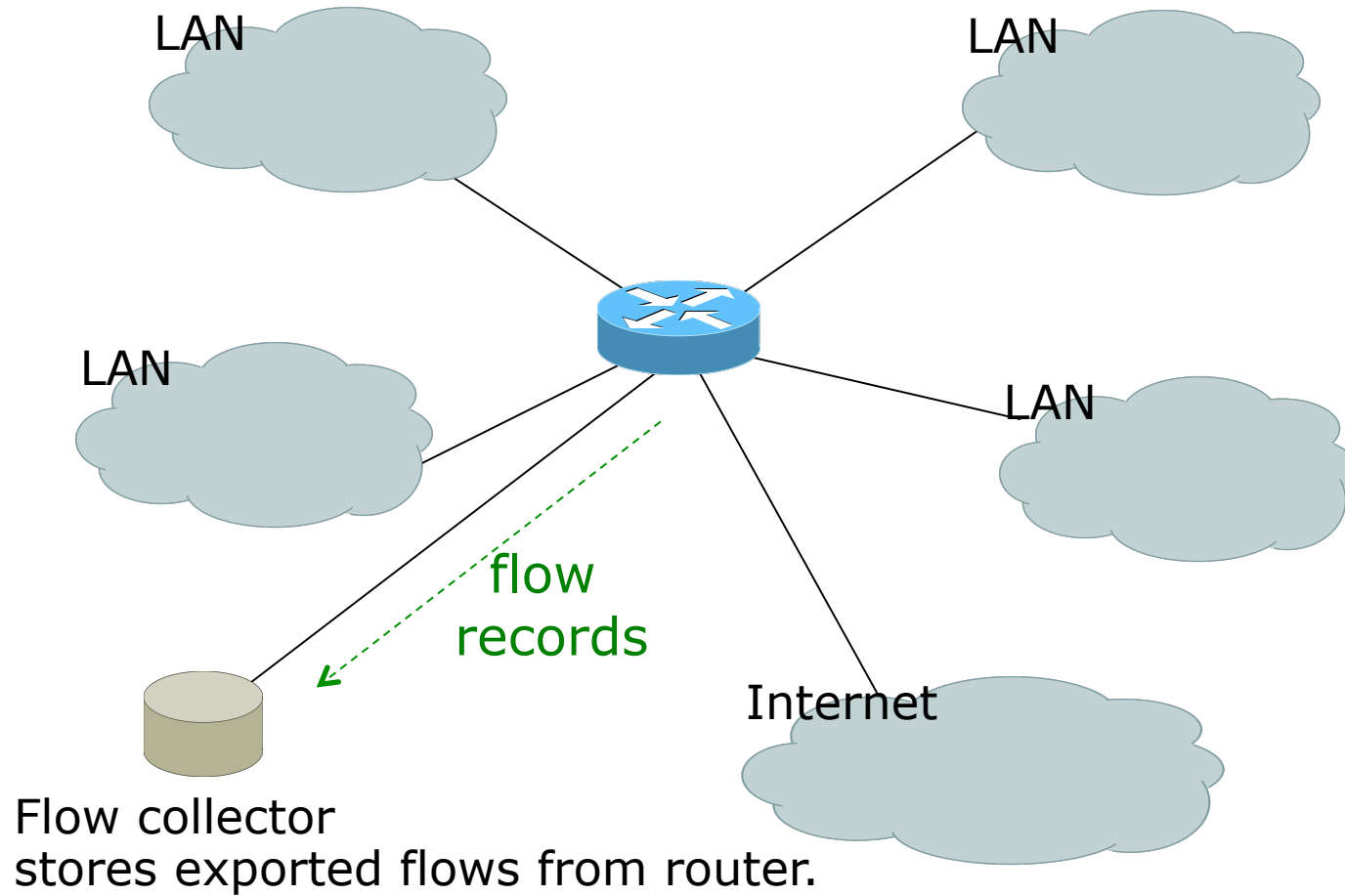
# Where to generate flow records

---

1. On a router or other network device
  - If the device supports it
  - No additional hardware required
  - Might have some impact on performance
2. Passive collector (usually a Unix host)
  - Receives a copy of every packet and generates flows
  - Requires a mirror port
  - Resource intensive

# Router Collection

---

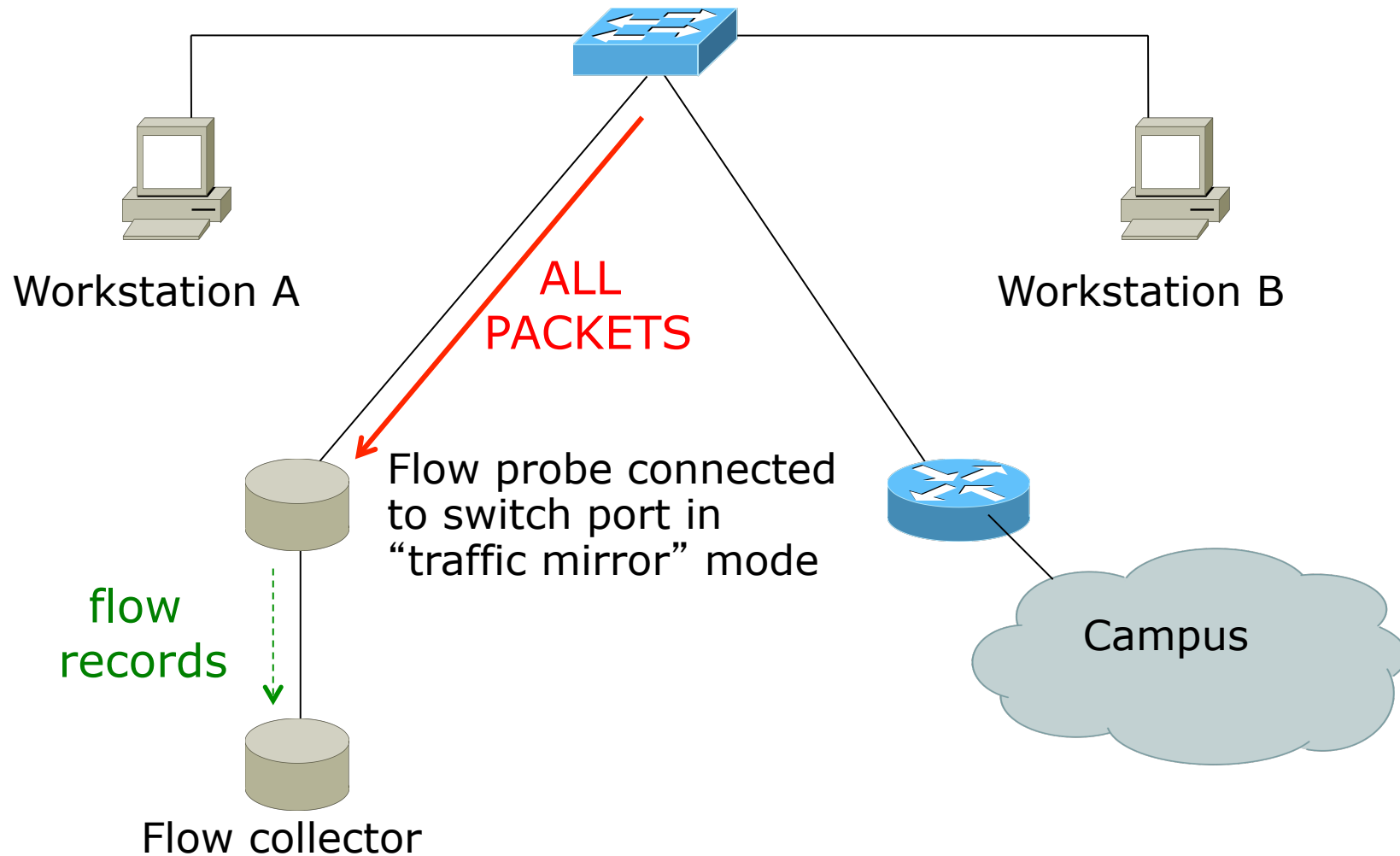


# Router Collection

---

- ❑ All flows through router can be observed
- ❑ Router overhead to process & export flows
- ❑ Can select which interfaces Netflow collection is needed and not activate it on others
- ❑ If router on each LAN, Netflow can be activated on them to reduce load on core router

# Passive Monitor Collection



# Passive Collector

---

- Examples
  - softflowd (Linux/BSD)
  - pfflowd (BSD)
  - ng\_netflow (BSD)
- Collector sees all traffic through the network point it is connected on and generates flows
- Relieves router from processing traffic, creating flows and exporting them

## Passive Collector (cont.)

---

- Useful on links:
  - With only one entry into the network
  - Where only flows from one section of the network are needed
- Can be deployed in conjunction with an IDS



## A thought:

---

- Your network probably already has a device which is keeping track of IP addresses and port numbers of traffic flowing through it.

**What is it?**

# Flow Export Protocols

---

- Cisco Netflow, different versions
  - v5: widely deployed
  - v9: newer, extensible, includes IPv6 support
- IP Flow Information Export (**IPFIX**):
  - IETF standard, based on Netflow v9
- **sFlow**: Sampling-based, commonly found on switches
- **jFlow**: Juniper
- We use Netflow, but many tools support multiple protocols

# Cisco Netflow

---

- Unidirectional flows
- IPv4 unicast and multicast
  - (IPv6 in Netflow v9)
- Flows exported via UDP
  - Choose a port. No particular standard, although 2055 and 9996 are commonly used
- Supported on IOS, ASA and CatOS platforms
  - But with different implementations

# Cisco IOS Configuration

---

- ❑ Configured on each input interface
  - modern IOS allows both input and output
- ❑ Define the version
- ❑ Define the IP address and port of the collector (where to send the flows)
- ❑ Optionally enable aggregation tables
- ❑ Optionally configure flow timeout and main (v5) flow table size
- ❑ Optionally configure sample rate

# Configuring Netflow: the old way

---

## □ Enable CEF

```
ip cef
ipv6 cef
```

## □ Enable Netflow on each interface (IPv4)

```
ip route-cache flow
```

 (pre IOS 12.4)

OR

```
ip flow ingress
ip flow egress
```

(IOS 12.4 onwards)

## □ Enable Netflow on each interface (IPv6)

```
ipv6 flow ingress
ipv6 flow egress
```

(IOS 12.4 and 12.4T only)

# Configuring Netflow: the old way

---

## □ Configuring top-talkers:

```
ip flow-top-talkers
  top 50
  sort-by bytes
  match input-interface <interface>
```

## □ Showing top flows:

```
show ip flow top-talkers
```

## □ Exporting Flows to a collector

```
ip flow-export version [5|9] [origin-as|peer-as]
ip flow-export destination <x.x.x.x> <udp-port>
```



## "Flexible Netflow": the new way

---

- ❑ Only way to monitor IPv6 flows on modern IOS
- ❑ Start using it now – IPv6 is coming / here
- ❑ Many mind-boggling options available, but basic configuration is straightforward

# Flexible netflow configuration

---

- Define one or more exporters

```
flow exporter EXPORTER-1
  destination 192.0.2.99
  transport udp 9996
  source Loopback0
  template data timeout 300
```

- Define one or more flow monitors

```
flow monitor FLOW-MONITOR-V4
  exporter EXPORTER-1
  cache timeout active 300
  record netflow ipv4 original-input
flow monitor FLOW-MONITOR-V6
  exporter EXPORTER-1
  cache timeout active 300
  record netflow ipv6 original-input
```



# Flexible netflow configuration

---

- Apply flow monitors to interface

```
interface GigabitEthernet0/0/0
 ip flow monitor FLOW-MONITOR-V4 input
 ip flow monitor FLOW-MONITOR-V4 output
 ipv6 flow monitor FLOW-MONITOR-V6 input
 ipv6 flow monitor FLOW-MONITOR-V6 output
```

# "Top-talkers"

---

- You can summarize flows directly on the router, e.g.

```
show flow monitor FLOW-MONITOR-V4 cache aggregate ipv4  
source address ipv4 destination address sort counter  
bytes top 20
```

- Yes, that's one long command!
- Old command "show ip flow top-talkers" sadly gone, but you could make an alias

```
config terminal  
alias exec top-talkers show flow..
```



Questions?

---

# System Logging – Syslog



# Syslog basics

---

- ❑ Uses UDP protocol, port 514
- ❑ Syslog messages have two attributes (in addition to the message itself):

Facility		Level	
Auth	Security	Emergency	(0)
Authpriv	User	Alert	(1)
Console	Syslog	Critical	(2)
Cron	UUCP	Error	(3)
Daemon	Mail	Warning	(4)
FTP	NTP	Notice	(5)
Kern	News	Info	(6)
LPR		Debug	(7)
Local0...Local7			



# Log Management and Monitoring

- ❑ Keep your logs in a secure place where they can be easily inspected.
- ❑ Watch your log files.
- ❑ They contain important information:
  - Lots of things happen and someone needs to review them.
  - It's not practical to do this manually.

# Log Management and Monitoring

---

## □ On your routers and switches

```
Sep  1 04:40:11.788 INDIA: %SEC-6-IPACCESSLOGP: list 100 denied tcp
79.210.84.154(2167) -> 169.223.192.85(6662), 1 packet

Sep  1 04:42:35.270 INDIA: %SYS-5-CONFIG_I: Configured from console by pr on
vty0 (203.200.80.75)

%CI-3-TEMP: Overtemperature warning

Mar  1 00:05:51.443: %LINK-3-UPDOWN: Interface Serial1, changed state to down
```

## □ And, on your servers

```
Aug 31 17:53:12 ubuntu nagios3: Caught SIGTERM, shutting down...

Aug 31 19:19:36 ubuntu sshd[16404]: Failed password for root from
169.223.1.130 port 2039 ssh2
```



# Log Management

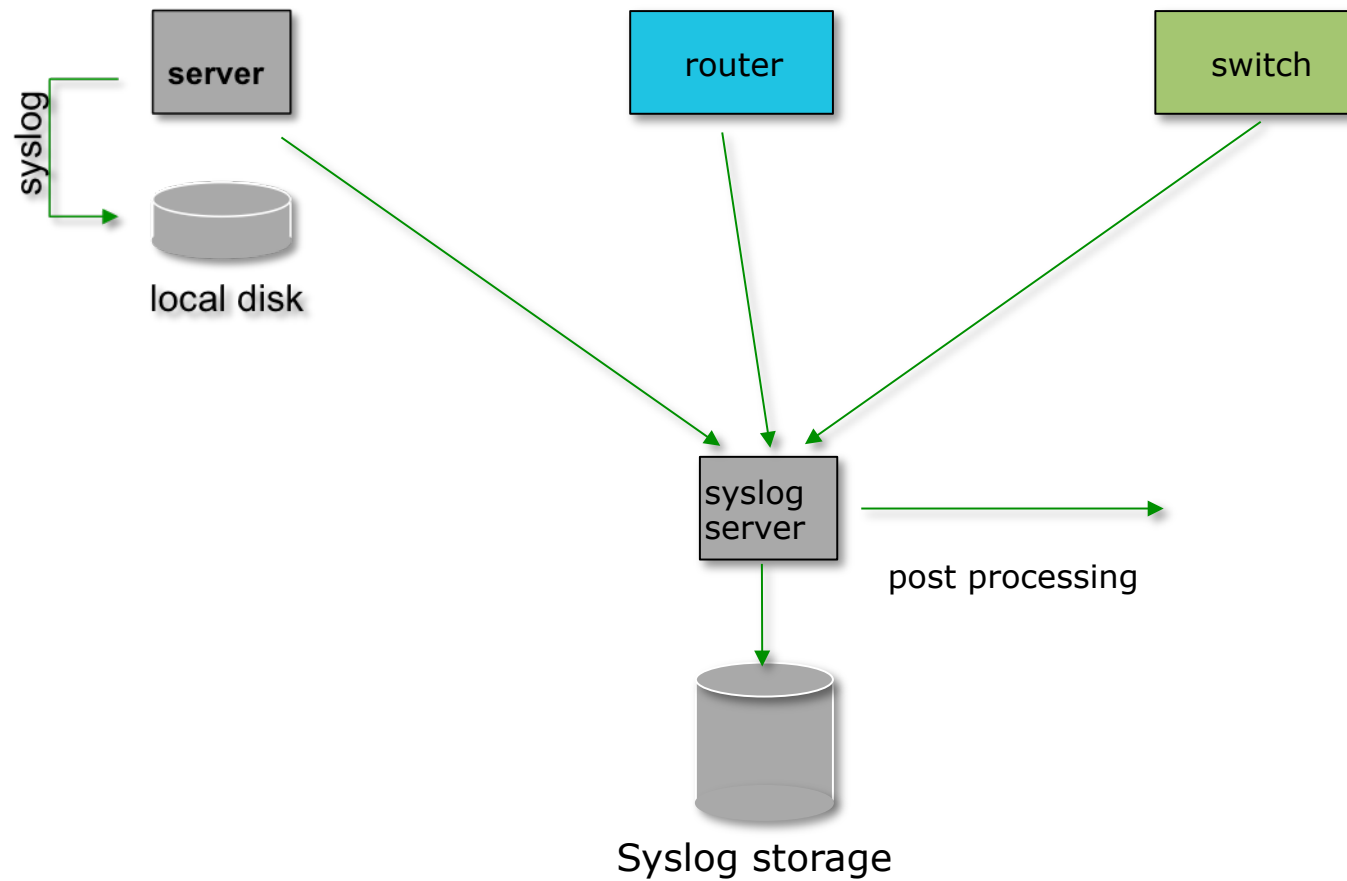
---

- ❑ Centralize and consolidate log files
- ❑ Send all log messages from your routers, switches and servers to a single node – a log server.
- ❑ All network hardware and UNIX/Linux servers can be monitored using some version of syslog.
- ❑ Windows can, also, use syslog with extra tools.
- ❑ Save a copy of the logs locally, but, also, save them to a central log server.



# Centralized logging

---



# Configuring centralized logging

---

- Cisco hardware

- At a minimum:

- logging ip.of.logging.host

- Unix and Linux nodes

- In /etc/syslog.conf, add:

- ```
*.* @ip.of.log.host
```

- Restart syslogd

- Other equipment have similar options

- Options to control facility and level

# Receiving syslog messages

---

- ❑ Identify the facility that the equipment is going to use to send its messages.
- ❑ Reconfigure syslogd to listen to the network.
  - Ubuntu: add "-r" to `/etc/defaults/syslogd`
- ❑ Add an entry to syslogd where messages are going to be written:  

```
local7.* /var/log/routers
```
- ❑ Create the file  

```
touch /var/log/routers
```
- ❑ Restart syslogd  

```
/etc/init.d/syslogd restart
```

# Grouping logs

---

- ❑ Using facility and level you can group by category in distinct files.
- ❑ With software such as syslog-ng you can group by machine, date, etc. automatically in different directories.
- ❑ You can use grep to review logs.
- ❑ You can use typical UNIX tools to group and eliminate items that you wish to filter:  

```
egrep -v '(list 100 denied|logging rate-limited)' mylogfile
```
- ❑ Is there a way to do this automatically?

# Benefits of Accurate Time



# Network Time Protocol

---

- ❑ If you want to cross compare logs from network devices, you need to synchronize the time on all the devices
- ❑ Use NTP
  - From external time source
    - ❑ Upstream ISP, Internet, GPS, atomic clock
  - From internal time source
    - ❑ Router can act as stratum 1 time source
- ❑ Operators of network infrastructure crossing timezones tend to:
  - Operate network in one timezone
  - Or set all system clocks to GMT (UTC)

# Network Time Protocol

---

- ❑ **Set timezone**

```
clock timezone <name> [+/-hours [mins]]
```

- ❑ **Router as source**

```
ntp master 1
```

- ❑ **External time source (master)**

```
ntp server a.b.c.d
```

- ❑ **External time source (equivalent)**

```
ntp peer e.f.g.h
```

# Network Time Protocol

---

## □ Example Configuration:

```
clock timezone AEST 10
!  
ntp update-calendar  
ntp source loopback0  
ntp server <other time source>  
ntp peer <other time source>  
ntp peer <other time source>
```



# Securing NTP

---

- ❑ NTP needs to be protected (like any other protocol)
  - Defaults are for NTP to listen to the world (for synchronisation as well as command/control functions)
- ❑ Command/control channels return large amounts of information from one simple query
  - Potential of serious DOS attack for very little effort
- ❑ Best Practice:
  - Allow synchronisation from trusted devices (or world)
  - Allow command/control from no one (or NOC only)

# Secured NTP: Example

---

- Access control lists:
  - One ACL to block access to the command/control channels
  - Another ACL to allow access to the time server
- Applying ACLs to ntp configuration
  - `ntp access-group [ipv4|ipv6] peer`
    - The ntp devices we give **full** access to
  - `ntp access-group [ipv4|ipv6] serve`
    - The ntp devices we give server and query access to
  - `ntp access-group [ipv4|ipv6] serve-only`
    - The ntp devices we give server access to
  - `ntp access-group [ipv4|ipv6] query-only`
    - The ntp devices we give query access to

# Secured NTP: IPv4

---

- Example IPv4 Configuration:

```
ip access-list standard ntp-block
  remark Utility ACL to block everything
  deny any
ip access-list standard ntp-servers
  remark NTP peers/servers we sync with
  permit 192.168.1.254
  deny any
!
ntp access-group peer ntp-servers
ntp access-group serve ntp-block
ntp access-group serve-only ntp-block
ntp access-group query-only ntp-block
ntp server 192.168.1.254
```

# Secured NTP: IPv6

---

- Example IPv6 Configuration:

```
ipv6 access-list v6ntp-block
  remark Utility ACL to block everything
  deny any any
ipv6 access-list v6ntp-servers
  remark NTP peers/servers we sync with
  permit 2001:db8::1/128 any
  deny any any
!
ntp access-group ipv6 peer v6ntp-servers
ntp access-group ipv6 serve v6ntp-block
ntp access-group ipv6 serve-only v6ntp-block
ntp access-group ipv6 query-only v6ntp-block
ntp server 2001:db8::1
```

# Using Intrusion Detection and Prevention Systems



# Intrusion detection

---

- What is intrusion detection ?
  - Technically, any method that allows you to discover if someone has penetrated or is attempting intrusion into your network, host, services.
- What is intrusion ?
  - Unlawfully gaining access to systems, resources
  - The access itself, or the methods used, may be unlawful
  - There may not be a “break-in”
  - The result is the same
    - Someone is accessing something they are not allowed to...
- Just because the door was open doesn't mean I am allowed to walk in
  - You still have an intruder!



# What is an IDS ?

---

- An IDS is a device, or group of devices, which look for specific patterns in network traffic, for the purpose of detecting malicious intent

# Snort ?

---

- ❑ Snort is an open source IDS, and one of the oldest ones
- ❑ Hundreds of thousands of users
- ❑ Active development of rules by the community make Snort up to date, and often more so than commercial alternatives
- ❑ Snort is fast! It can run at Gbit/s rates with the right hardware and proper tuning



# Where to place Snort ?

---

- Snort will need to be close to the “choke point” (the point where all traffic flows through on the way in or out of your network)
  - Inside of the border router or firewall, for example

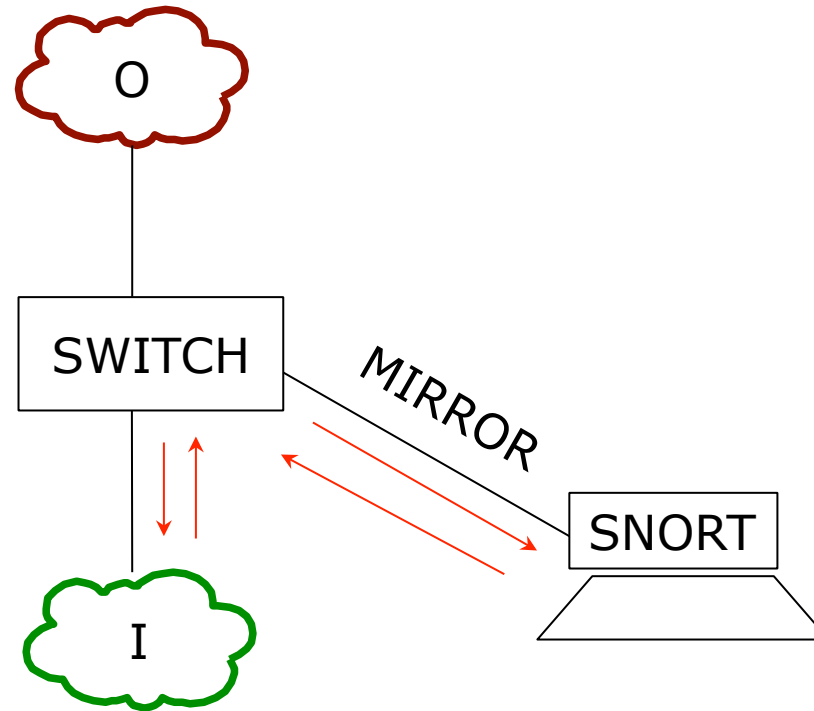
# Getting Snort to see the network

---

- You could run Snort in multiple ways
  - As a device “in line” behind or after the firewall/router
    - But this adds one more element that can fail in your connectivity
  - Or you could use a span/mirror port to send traffic to Snort
  - Or you can use an “optical splitter” to “mirror” or “tap into” traffic from a fiber optic link
    - This method and the previous are the most recommended

# Getting Snort to see the network

---





# Getting Snort to see the network

---

- ❑ Be careful not to overload your switch port
  - If you mirror a gigabit port to another gigabit port, the monitoring port (the receiving port) can drop packets if the total traffic exceeds 1 Gbit/s
- ❑ We'll illustrate this...

# Monitoring Port...

---

- ❑ On Cisco Catalyst, this is a “SPAN” port
- ❑ You can SPAN one port to another, a group of ports to one port, or an entire VLAN to a port
- ❑ Sample config:

```
interface FastEthernet 0/1
# port monitor FastEthernet 0/2
```
- ❑ This would copy any packet received on F0/2 to F0/1

# Monitoring Port...

---

- ❑ Other equipment vendors have different syntax
- ❑ HP calls it a “mirror port”

# Snort configuration file

---

- ❑ By default, `/etc/snort/snort.conf`
- ❑ It's a long file – 900+ lines
- ❑ If you browse it, you will notice many “preprocessor” entries
- ❑ Snort has a number of “preprocessors” which will analyze the network traffic and possibly clean it up before passing it to the rules

# Snort rules

---

- ❑ Snort rules are plain text files
- ❑ Adding new rules to snort is as simple as dropping the files into `/etc/snort/rules/`
- ❑ Groups of rules can be loaded from `snort.conf` using the “include” statement
- ❑ Rules can match anything
  - Technical – web attacks, buffer overflow, portscan, etc...
  - Policy/user oriented – URL filtering, keyword, forbidden applications, etc...



# Tailoring the rules

---

- ❑ Not all rules will make sense in your network
- ❑ You will want to customize which rules you want to run
- ❑ Otherwise you will get many false positives, which will lead you to ignore Snort, or simply turn it off...
  - It doesn't help to have logs full of junk alerts you don't want
  - To avoid this, rules can be suppressed (disabled)

# Updating Snort rules

---

- ❑ The commercially maintained snort rules are available for free with a 30 day delay from:
  - <http://www.snort.org/start/rules>
- ❑ Other rules are maintained by some volunteers at emerging threats:
  - <http://rules.emergingthreats.net/open/>
- ❑ The updating of rules can be automated with a tool called “Pulled Pork”, which is located at:
  - <http://code.google.com/p/pulledpork/>

# Sample rules

---

These signatures are not enabled by default as they may generate false positive alarms on networks that do mysql development.

```
alert tcp $EXTERNAL_NET any -> $SQL_SERVERS 3306
(msg:"MYSQL root login attempt"; flow:to_server,established;
content:"|0A 00 00 01 85 04 00 00 80|root|00|"; classtype:protocol-
command-decode; sid:1775; rev:2;)
```

```
alert tcp $EXTERNAL_NET any -> $SQL_SERVERS 3306
(msg:"MYSQL show databases attempt"; flow:to_server,established;
content:"|0F 00 00 00 03|show databases"; classtype:protocol-
command-decode; sid:1776; rev:2;)
```

```
alert tcp $EXTERNAL_NET any -> $SQL_SERVERS 3306
(msg:"MYSQL 4.0 root login attempt"; flow:to_server,established;
content:"|01|"; within:1; distance:3; content:"root|00|"; within:5;
distance:5; nocase; classtype:protocol-command-decode; sid:3456;
rev:2;)
```

# Reporting and logging

---

- ❑ Snort can be made to log alerts to an SQL database, for easier searching
- ❑ A web front-end for Snort, BASE, allows one to browse security alerts graphically

# References and documentation

---

- Snort preprocessors:
  - <http://www.informit.com/articles/article.aspx?p=101148&seqNum=2>
- Snort documentation
  - <http://www.snort.org/docs>
- An install guide for Ubuntu 10.04:
  - <http://www.snort.org/assets/158/014-snortinstallguide292.pdf>

# Managing the Security Configuration



# What is RANCID?

---

- ❑ The "Really Awesome New Cisco config Differ" – Really!



- ❑ A configuration management tool:
  - Keeps track of changes in the configs of your network equipment (Cisco, HP, Juniper, Foundry, etc.)
- ❑ Works on routers and switches

# What is RANCID?

---

- ❑ Automates retrieval of the configurations and archives them
- ❑ Functions as:
  - Backup tool – “woops, my router burned”
  - Audit tool – “how did this error get in?”
  - Blame allocation :) – “who did it?”
- ❑ The data is stored in a VCS (Version Control System) – supported are:
  - CVS (Concurrent Versions Systems)
  - SVN (SubVersioN)





# What is Version Control?

---

- Three basic principles:
  - Keep a record and history of changes
  - Give public access to the information
  - To maintain different versions from the same data set
- What types of data?
  - Source code
  - Documentation
  - Configuration files
  - Generally, any type of data...

# How does RANCID work?

---

- Run (manually or automated)
- Lookup list of groups
  - For each device in each list of groups
    - Connect to the equipment (telnet, ssh, ...)
    - Run "show" commands – config, inventory, ...
    - Collect, filter/format data
    - Retrieve the resulting config files
    - CVS/SVN check-in the changes
    - Generate a diff from the previous version
    - E-mail the diff to a mail address (individual or group)

# What to use it for?

---

- ❑ Track changes in the equipment configuration
- ❑ Track changes in the hardware (S/N, modules)
- ❑ Track version changes in the OS (IOS, CatOS versions)
- ❑ Find out what your colleagues have done without telling you!
- ❑ Recover from accidental configuration errors (anyone have stories?)



# Post processing

---

- ❑ Run traditional filtering commands on your configs (grep, sed, for information)
- ❑ Re-use the automated login tools to build your own batch tools or do interactive login
- ❑ On large configurations, you can parallelize operations



## Other applications

---

- ❑ Automated checks (verify configs for strange/inconsistent setup)
- ❑ Generate DNS file from equipment list
- ❑ Use IP address adjacency to produce a graph of your network

# References

---

- ❑ RANCID Project Home Page  
<http://www.shrubbery.net/rancid/>
- ❑ Subversion (SVN) Home Page  
<http://subversion.apache.org/>
- ❑ Good, Short RANCID Presentation  
<http://www.shrubbery.net/rancid/NANOG29/>
- ❑ RANCID HowTo's  
[http://www.linuxhomenetworking.com/wiki/index.php/Quick\\_HOWTO%3A\\_Ch1%3A\\_Network\\_Backups\\_With\\_Rancid](http://www.linuxhomenetworking.com/wiki/index.php/Quick_HOWTO%3A_Ch1%3A_Network_Backups_With_Rancid)  
[http://gentoo-wiki.com/HOWTO\\_Rancid](http://gentoo-wiki.com/HOWTO_Rancid)  
<http://homepage.mac.com/duling/halfdozen/RANCID-Howto.html>

# Security Monitoring



ISP Workshops