ICT2203

Network Security

SINGAPORE INSTITUTE OF TECHNOLOGY

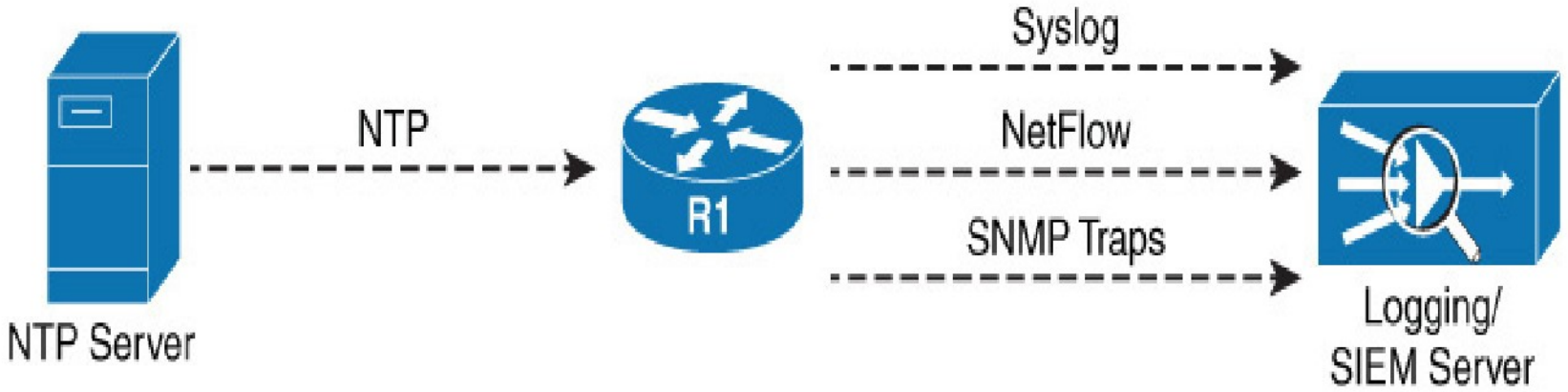
Lab 8 Notes:

Network Security Monitoring with NTP, Syslog and NetFlow

2021-2022 Trimester 3

Finally, with the various network security implemented, we are now ready to perform network security monitoring by logging all alerts to centralised servers for analysis and reporting.

Logging is commonly implemented using Syslog, SNMP and/or NetFlow protocols to centralised servers or SIEM (Security Information and Event Management) systems.



To facilitate analysis of log data, it is essential to synchronize time between all network devices, which can be achieved by using NTP (Network Time Protocol).

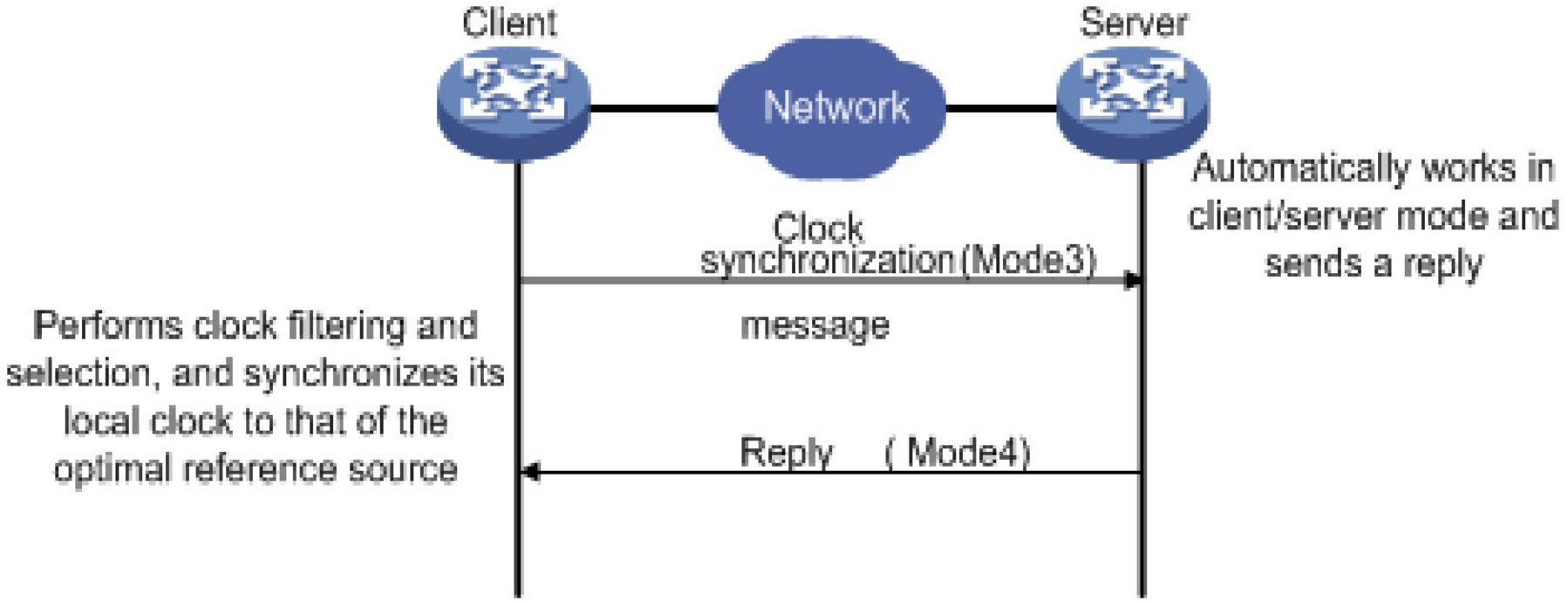
Lab Exercise 1A

SINGAPORE INSTITUTE OF TECHNOLOGY

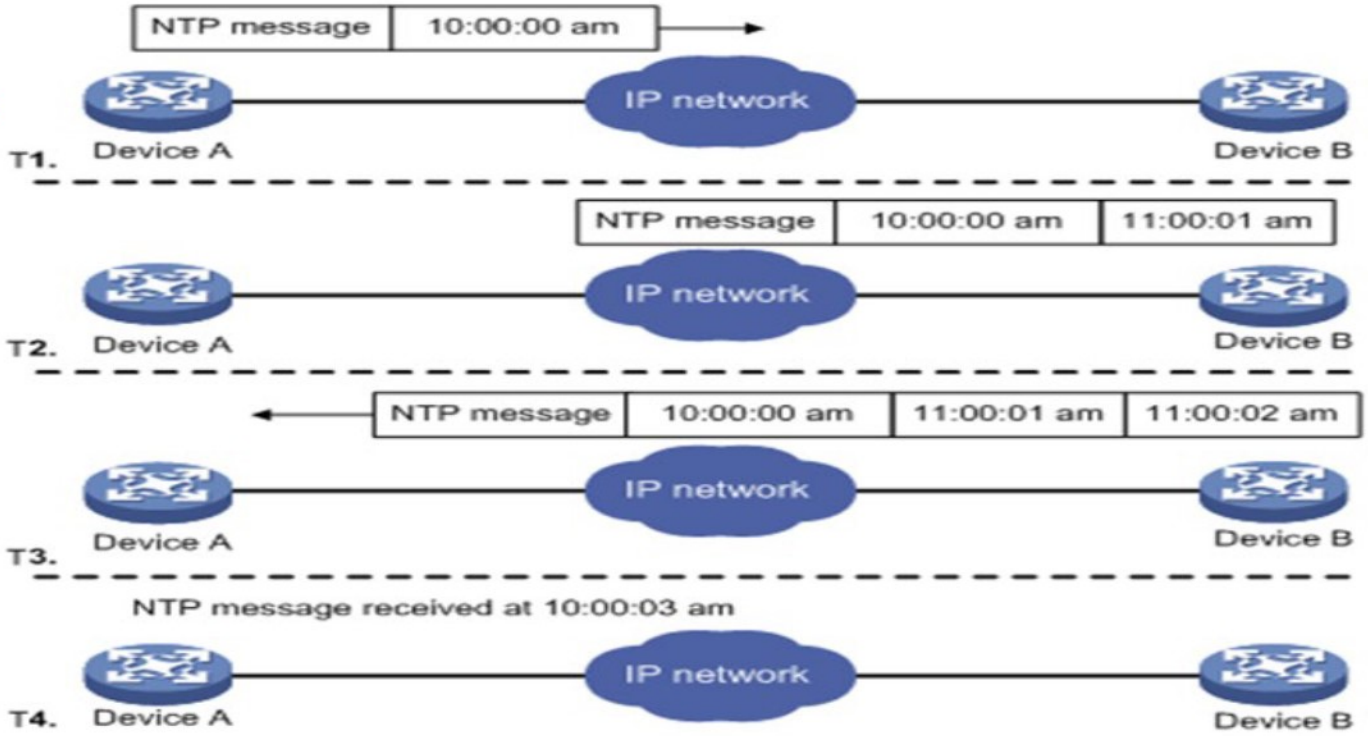
1.1 Configure NTP for synchronizing time between network devices to facilitate analysis of log data.

As defined in RFC 5905, there are 3 NTP operation modes – client/server, peer and broadcast. We'll discuss the common client/server mode to understand how NTP works.

NTP operates over UDP port 123. In client/server mode, the NTP client initiates a request to the NTP server, and receives a reply as shown:



In client/server mode, NTP client synchronises its clock by performing the following computation based on NTP reply:

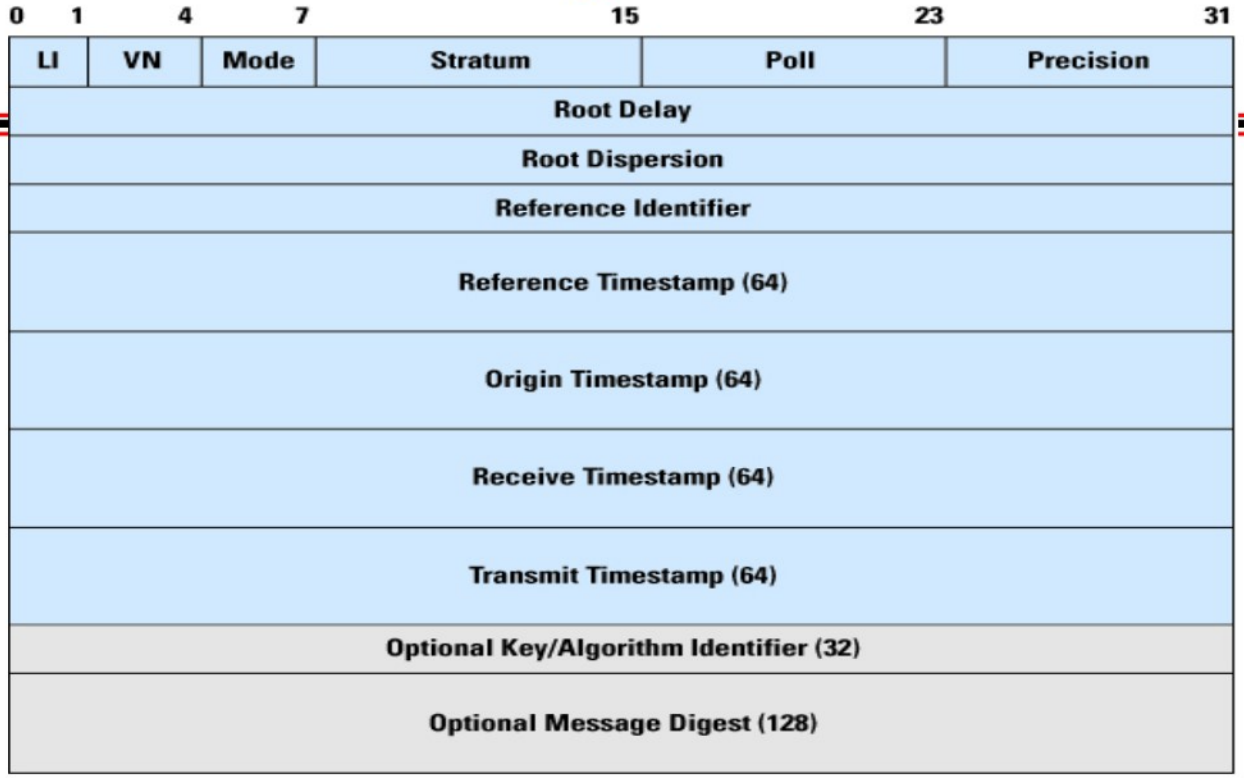


Offset of client clock from server clock: $\theta = \frac{(T2 - T1) + (T3 - T4)}{2}$ e.g. $\theta = \frac{1h1s + 59m59s}{2} = 1h$

Round-trip delay: $\delta = (T4 - T1) - (T3 - T2)$ e.g. $\delta = 3s - 1s = 2s$

(Clock filter and mitigation algorithms will then be used to determine the best and final offset to synchronize the system clock.)

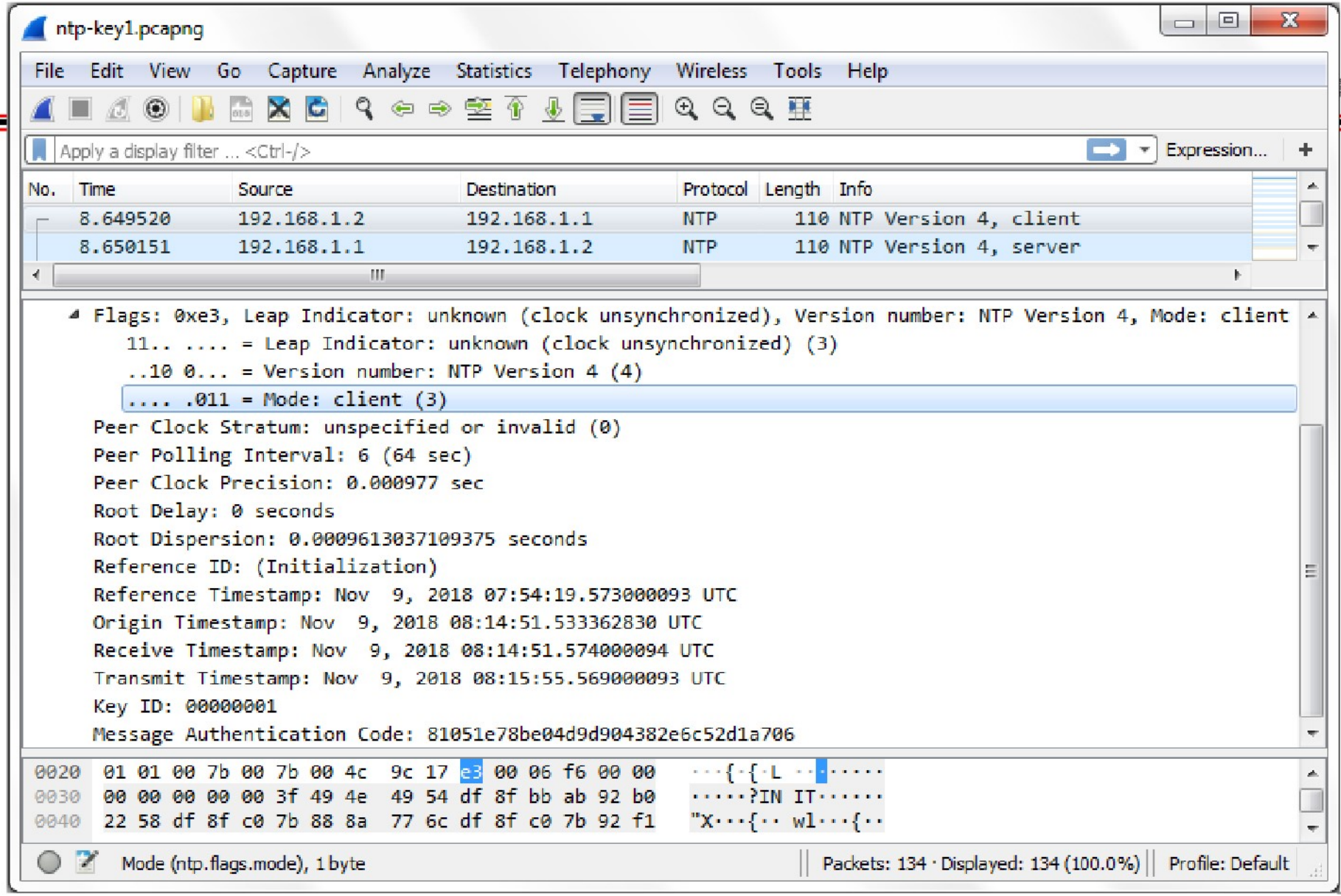
To support NTP computation, NTP client and server register its timestamps in the NTP message format as follows:



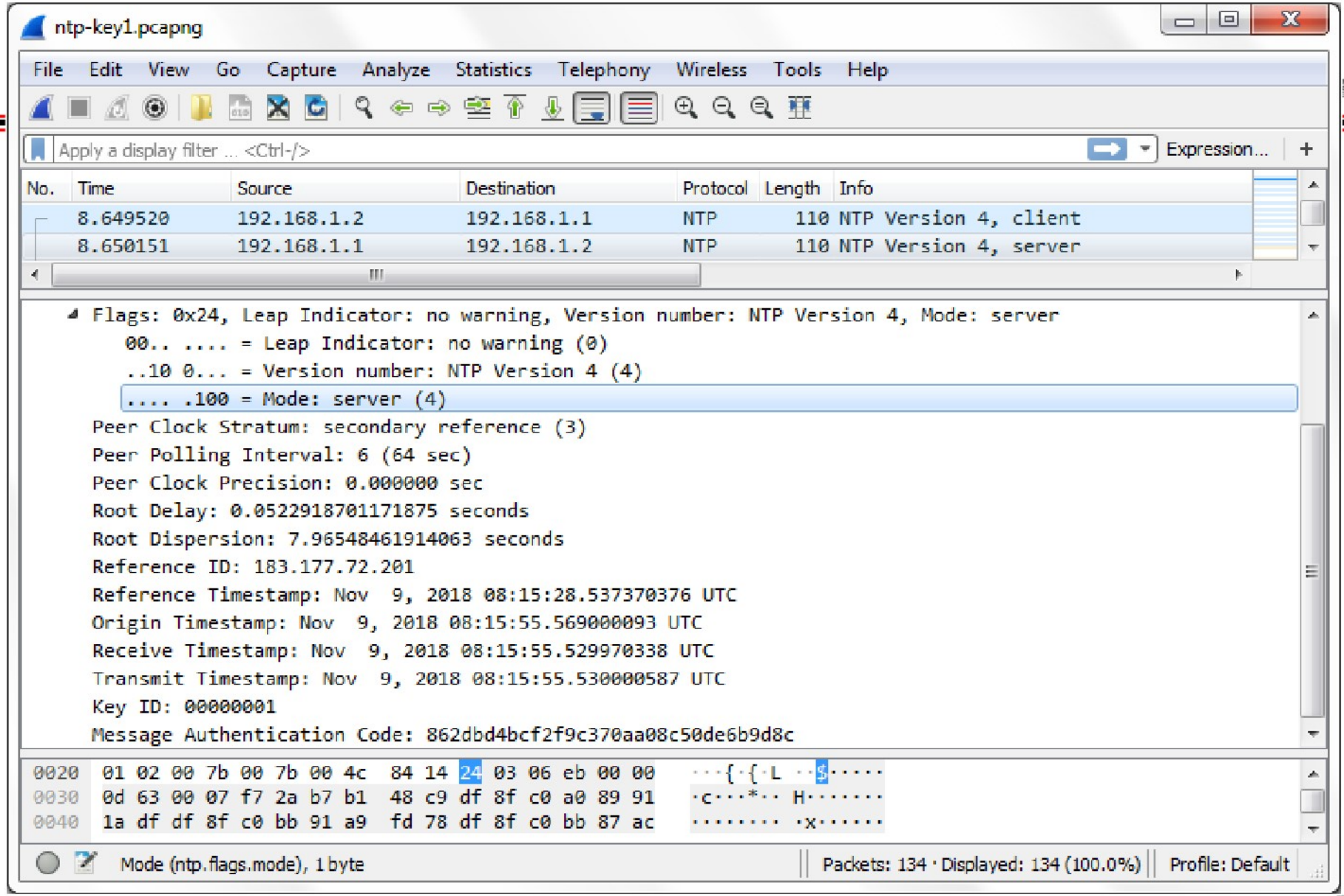
	NTP request	NTP reply
Origin timestamp:	ignore	T1
Receive timestamp:	ignore	T2
Transmit timestamp:	T1	T3

Reference timestamp: time when system clock was last updated

An example of NTP request sent by NTP client:

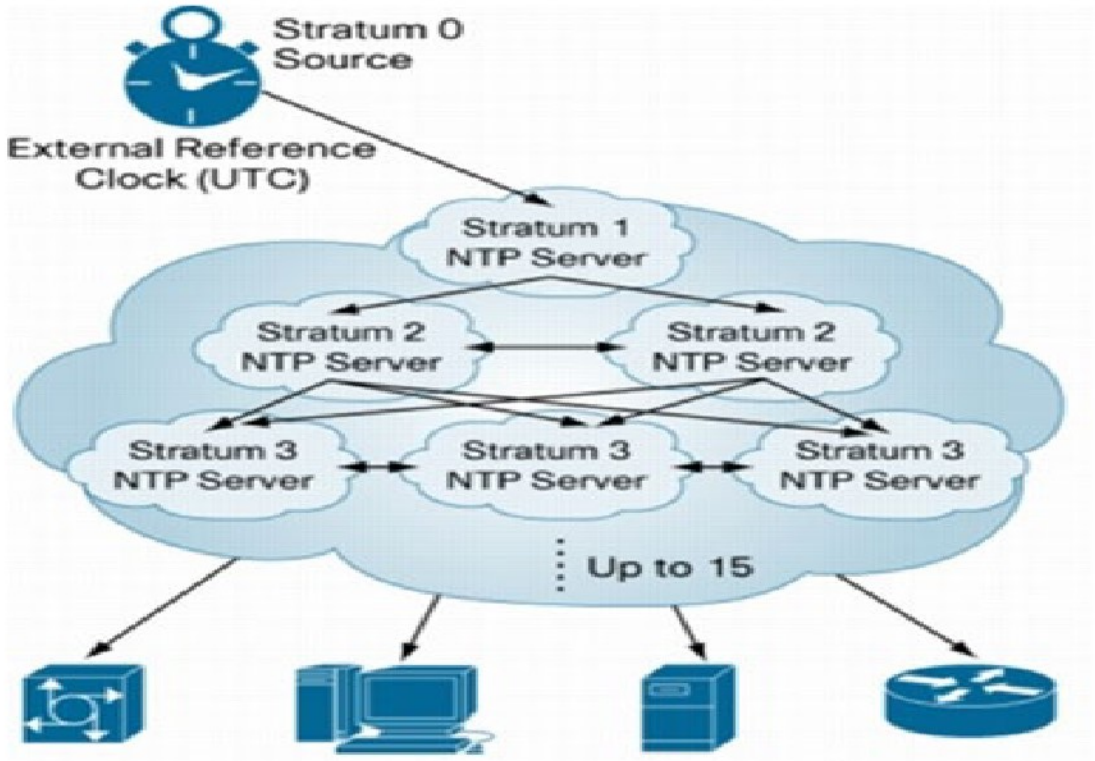


An example of NTP reply sent by NTP server:



To be scalable, NTP servers are typically setup in an hierarchical structure to support time synchronisation in practice.

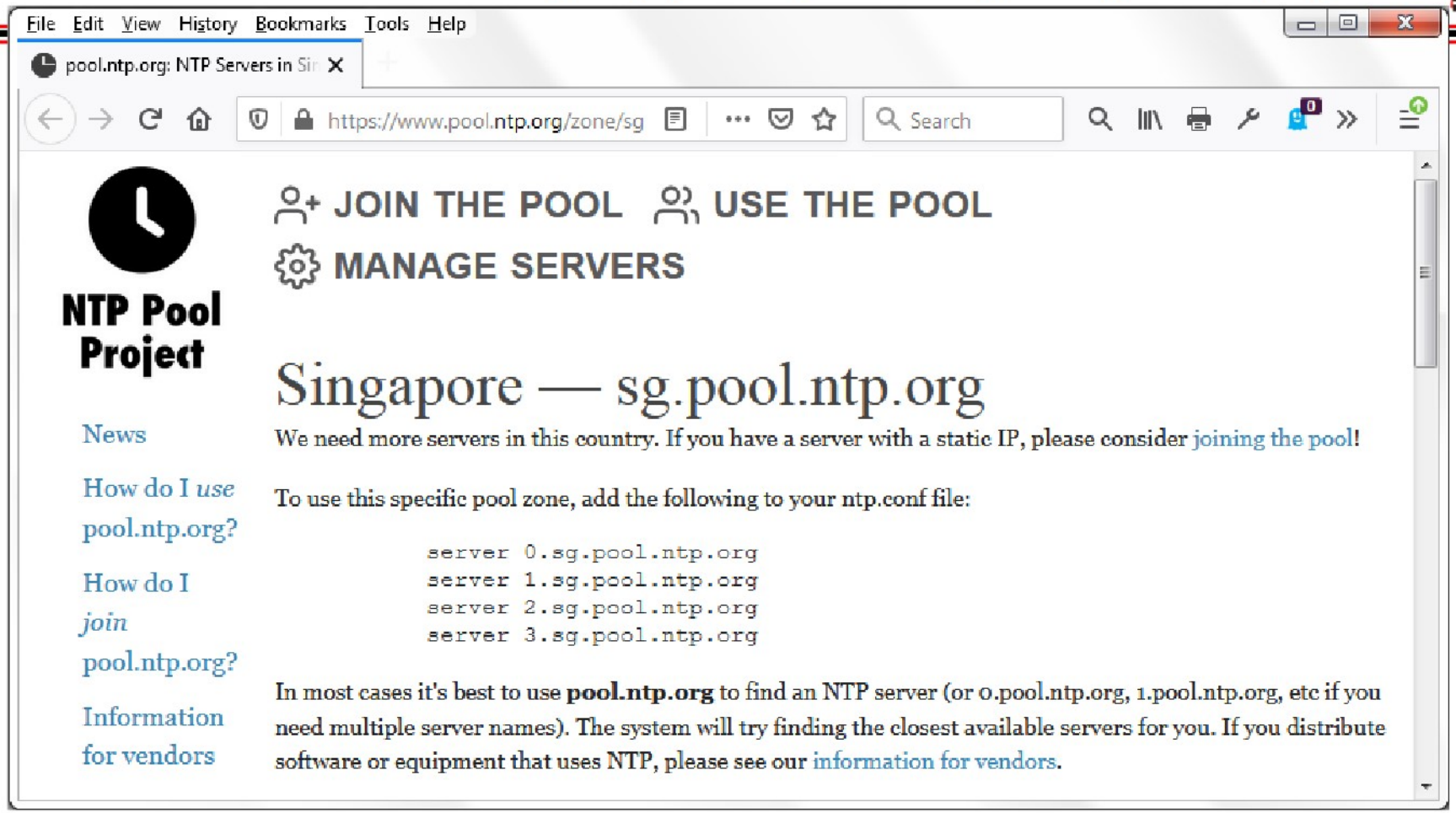
NTP uses the concept of stratum to describe how many NTP hops away a machine is from the most reliable authoritative time source at stratum 0.



As the stratum number increases, its accuracy degrades.

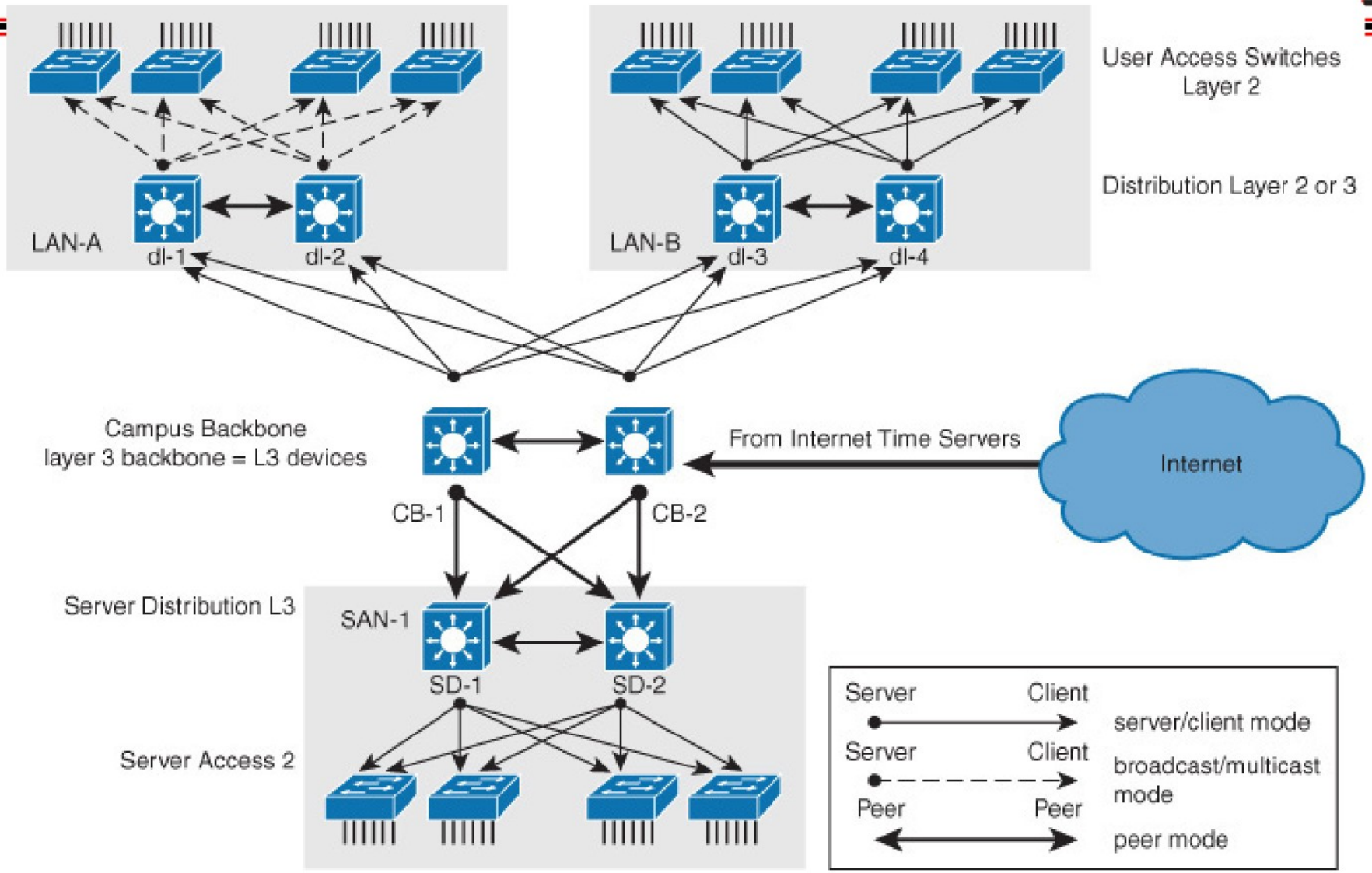
UTC (Coordinated Universal Time), aka GMT (Greenwich Mean Time), is the current worldwide standard for time and date.

An example is the NTP pool project which is maintaining a network of NTP servers that are being used by many systems around the world. ***https://www.pool.ntp.org/zone/sg***



To use the pool, simply configure your device to point to any of the provided domain name, each representing a random set of NTP servers.

After obtaining time from an NTP pool, an enterprise may implement time synchronization within its network in an hierarchical manner:



To be specific, we'll now discuss how to configure the network devices to synchronize time using NTP.

Before configuring NTP, you may wish to configure your device clock to current date/time:

```
R1# configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
R1(config)# clock timezone SGT 8
R1(config)# ^Z
R1#
R1# clock set 20:52:00 25 October 2017
*Oct 25 12:52:00.000: %SYS-6-CLOCKUPDATE: System clock has been updated from 20:36:38
  SGT Wed Oct 25 2015 to 20:52:00 SGT Wed Oct 25 2015, configured from console by
  console.
R1# show clock
20:52:55.051 SGT Wed Oct 25 2017
```

Any letters representing your timezone, e.g. SGT for Singapore Time, which is UTC + 8 hrs.

Based on the design of your network, configure network devices as NTP server, client, peer or broadcast:

- Server: provides time information to clients, only used if you are not able to reach reliable external NTP servers

```
R1(config)# ntp master [stratum]
```

- Client: synchronizes time with an NTP server

```
R1(config)# ntp server {ip | hostname} [version] [prefer]
```

- Peer: also called symmetric mode, which work both as server and client to synchronize, or be synchronized by each other

```
R1(config)# ntp peer {ip | hostname} [version] [prefer]
```

- Broadcast: pushes time announcements from NTP server to clients, used only when time accuracy is not a concern

Server to broadcast:

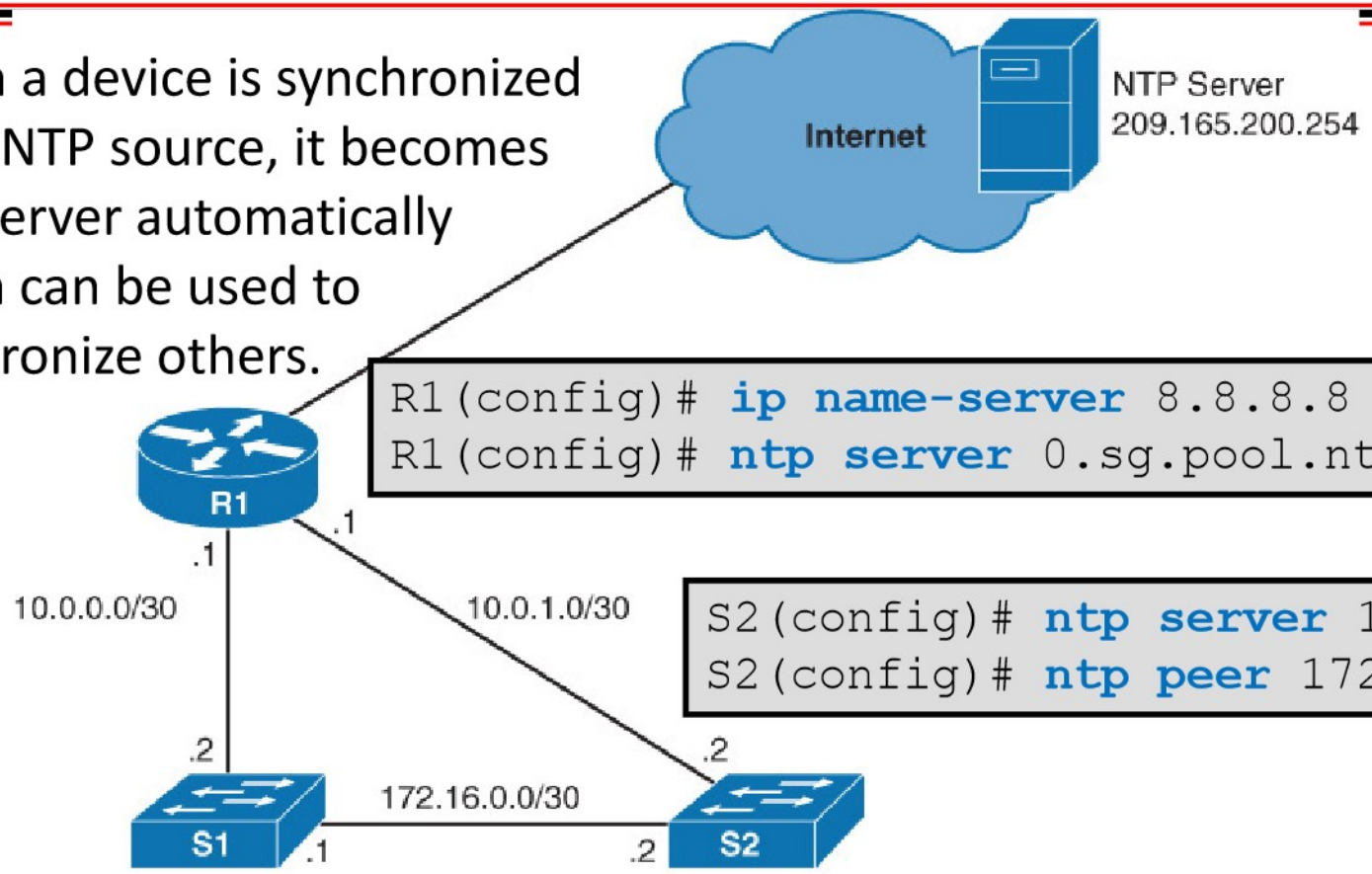
```
R1(config-if)# ntp broadcast [version]
```

Client to receive:

```
R1(config-if)# ntp broadcast client
```

An example of configuring NTP, with edge router R1 synchronizing with an external NTP time source, which in turn acts as NTP server for internal switches.

When a device is synchronized to an NTP source, it becomes NTP server automatically which can be used to synchronize others.




```
R1(config)# ip name-server 8.8.8.8
R1(config)# ntp server 0.sg.pool.ntp.org
```

```
S2(config)# ntp server 10.0.0.1
S2(config)# ntp peer 172.16.0.1
```

```
S1(config)# ntp server 10.0.0.1
S1(config)# ntp peer 172.16.0.2
```

To verify that the device's clock has indeed been synchronized with NTP, use the following show commands:

To verify which NTP server your device is synchronizing with, e.g.:

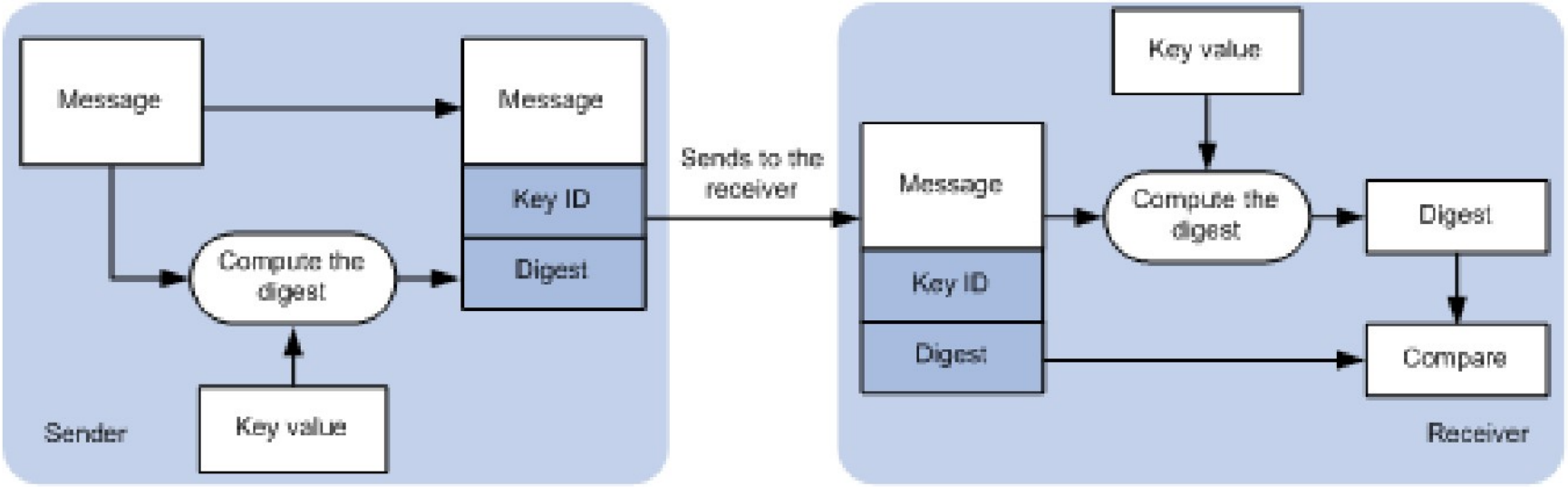
```
R1# show ntp associations

  address         ref clock       st   when   poll reach  delay  offset   disp
*~209.165.200.254 .LOCL.          1     24     64    17  1.000  -0.500  2.820
 * sys.peer, # selected, + candidate, - outlyer, x falseticker, ~ configured
```

By synchronizing with stratum 1 time source, your device automatically becomes stratum 2 time source as shown, e.g.:

```
R1# show ntp status
Clock is synchronized, stratum 2, reference is 209.165.200.254
nominal freq is 250.0000 Hz, actual freq is 250.0000 Hz, precision is 2**10
ntp uptime is 1500 (1/100 of seconds), resolution is 4000
reference time is D67E670B.0B020C68 (05:22:19.043 PST Mon Jan 13 2014)
clock offset is 0.0000 msec, root delay is 0.00 msec
root dispersion is 630.22 msec, peer dispersion is 189.47 msec
loopfilter state is 'CTRL' (Normal Controlled Loop), drift is 0.000000000 s/s
system poll interval is 64, last update was 5 sec ago.
```

Unfortunately, an attacker can inject false NTP replies to confuse time synchronization and log timings. Thus, it is advisable to enable NTP authentication to protect NTP clients.

Both NTP client and server can be configured with the same pre-shared key/password which is used to compute MD5 hash for authentication:



Using the pre-shared key, NTP server generates an MD5 hash and send it together with NTP message.

Using the same pre-shared key, NTP client re-computes the MD5 hash and verifies it with the received MD5 hash before accepting the NTP message.

For configuring of NTP authentication, the additional commands required are as follows:

- Define one or more NTP key(s), each number specifies a unique NTP key (password):

```
R1(config)# ntp authentication-key key-number md5 string
```

- Specify which key to enable for authentication:

```
R1(config)# ntp trusted-key key-number
```

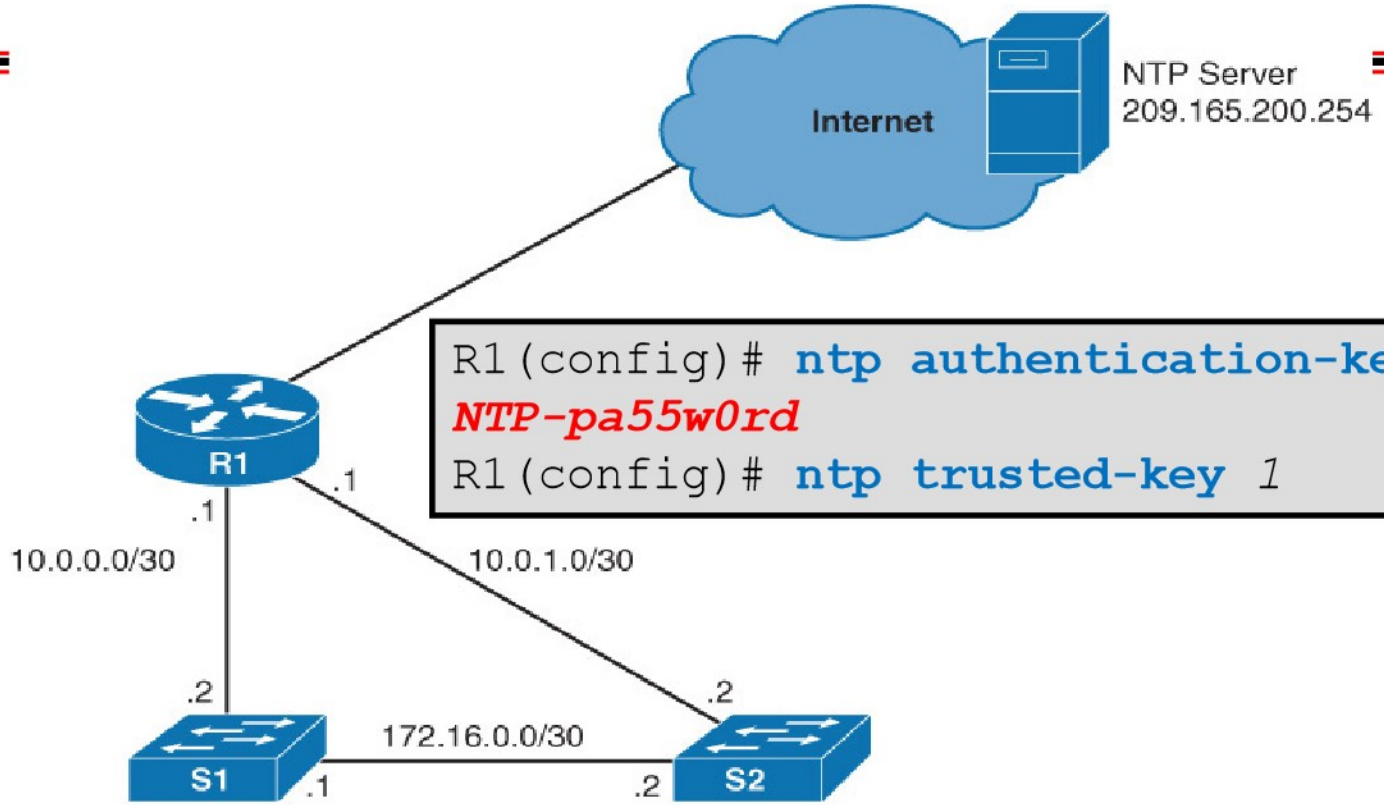
- Enable NTP authentication:

```
R1(config)# ntp authenticate
```

- For NTP client to specify which key to use for authentication:

```
R1(config)# ntp server {ip | hostname} key key-number
```

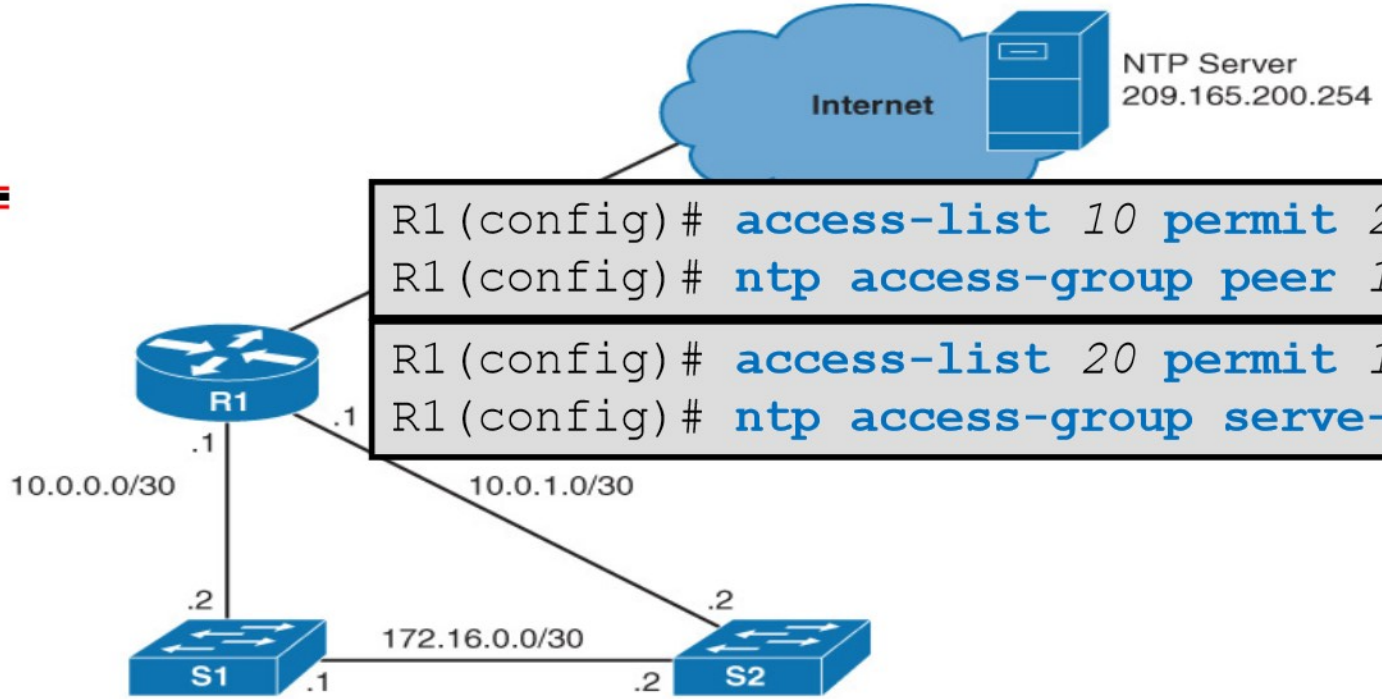
An example of configuring NTP authentication to prevent NTP clients from synchronising with attacker's time source.




```
R1(config)# ntp authentication-key 1 md5
NTP-pa55w0rd
R1(config)# ntp trusted-key 1
```

```
S1(config)# ntp authentication-key 1 md5 NTP-pa55w0rd
S1(config)# ntp authenticate
S1(config)# ntp trusted-key 1
S1(config)# ntp server 10.0.0.1 key 1
```

In addition, NTP can be protected by ACLs.




```
R1(config)# access-list 10 permit 209.165.200.254
R1(config)# ntp access-group peer 10
```

```
R1(config)# access-list 20 permit 10.0.0.0 0.0.0.7
R1(config)# ntp access-group serve-only 20
```

Command for configuring ACL:

```
ntp access-group {query-only | serve-only | serve | peer}
access-list-number
```

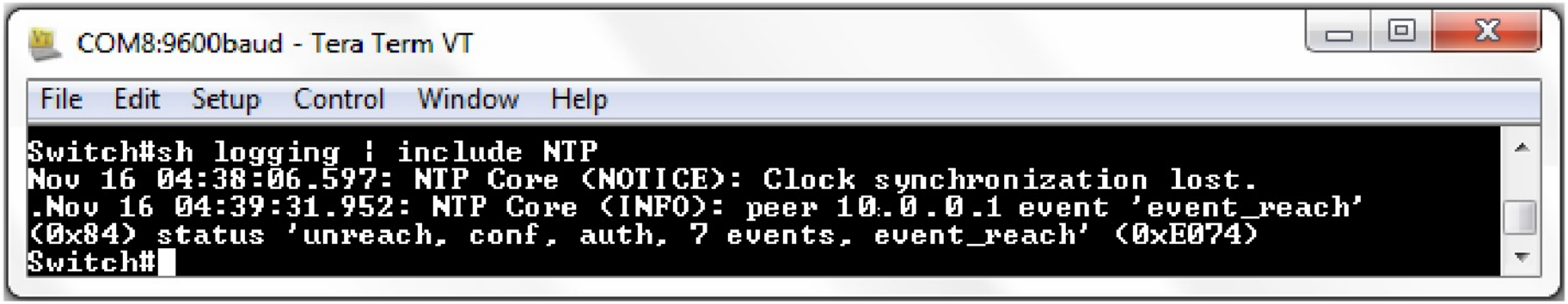
- peer: accept NTP reply and also respond to NTP request
- serve-only: for server to respond to NTP request only
- serve: similar as 'serve-only', but also accept NTP control queries
- query-only: only accept NTP control queries such as SNMP

Since NTP is important, it is also recommended to enable NTP logging to log significant NTP events.

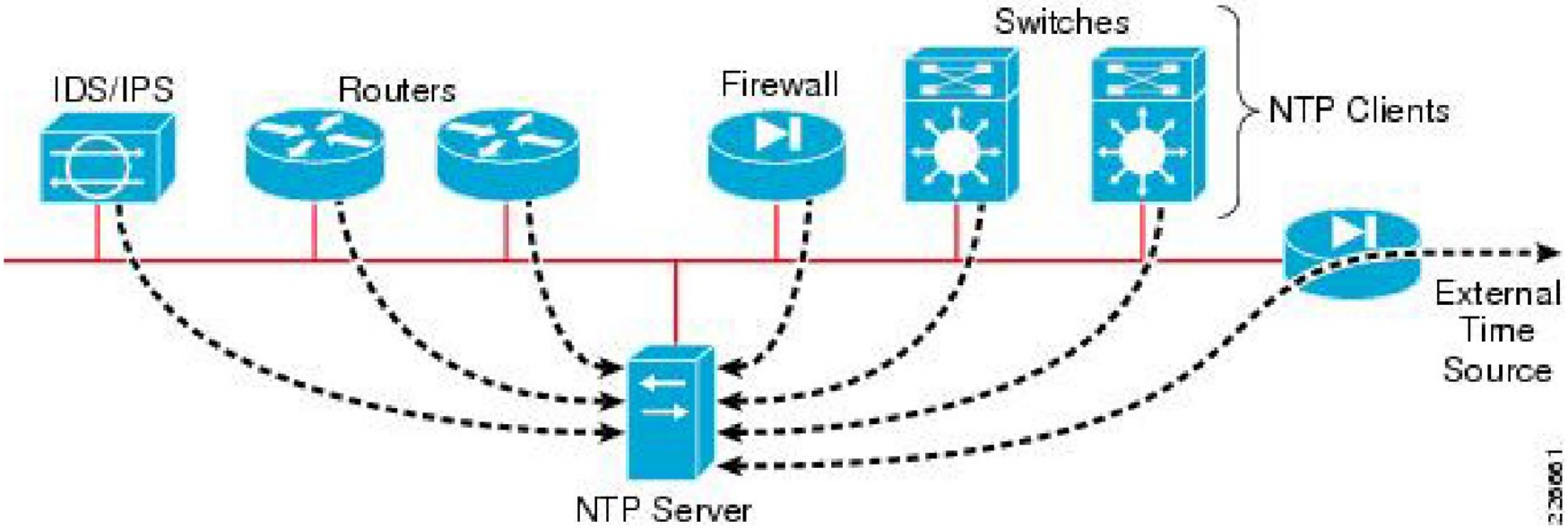
Enabling NTP logging (logging centrally will be discussed shortly):

```
Switch(config)# ntp logging
```

Examples of NTP logs:



If out-of-band (OOB) management network is available, it is more direct and secure to perform NTP time synchronization over OOB, e.g.:



If VRF is used: (note that command may be slightly different for different devices)

```
R1(config)# ntp server vrf Mgmt-vrf 10.0.0.1
```

```
S1(config)# ntp server 10.0.0.1 use-vrf Mgmt-vrf
```

Lab Exercise 1B

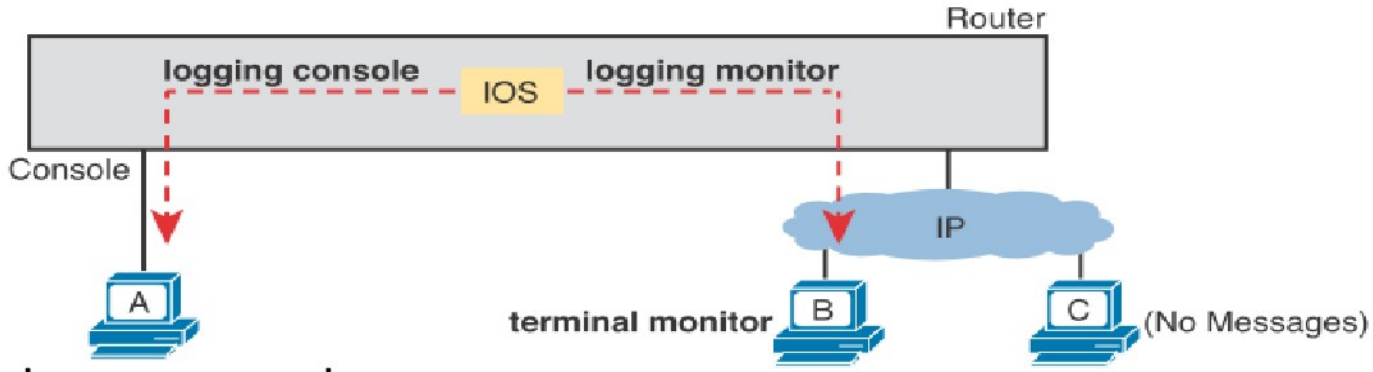
SINGAPORE INSTITUTE OF TECHNOLOGY

1.2 Configure Syslog which is one of the most commonly implemented method for logging.

Syslog (System message logging) (RFC 5424) is a standard protocol for event notification messages which is widely adopted by the industry.

While configuring Cisco devices in the labs, you must have seen or even be annoyed by the syslog messages that appeared, e.g.

```
*Dec 18 17:10:15.079: %LINEPROTO-5-UPDOWN: Line protocol on Interface
  FastEthernet0/0, changed state to down
```



Disable syslog on console:

```
R1(config)# no logging console
```

Prevent syslog from interrupting command entry:

```
R1(config)# logging synchronous
```

Enable syslog over Telnet/SSH:

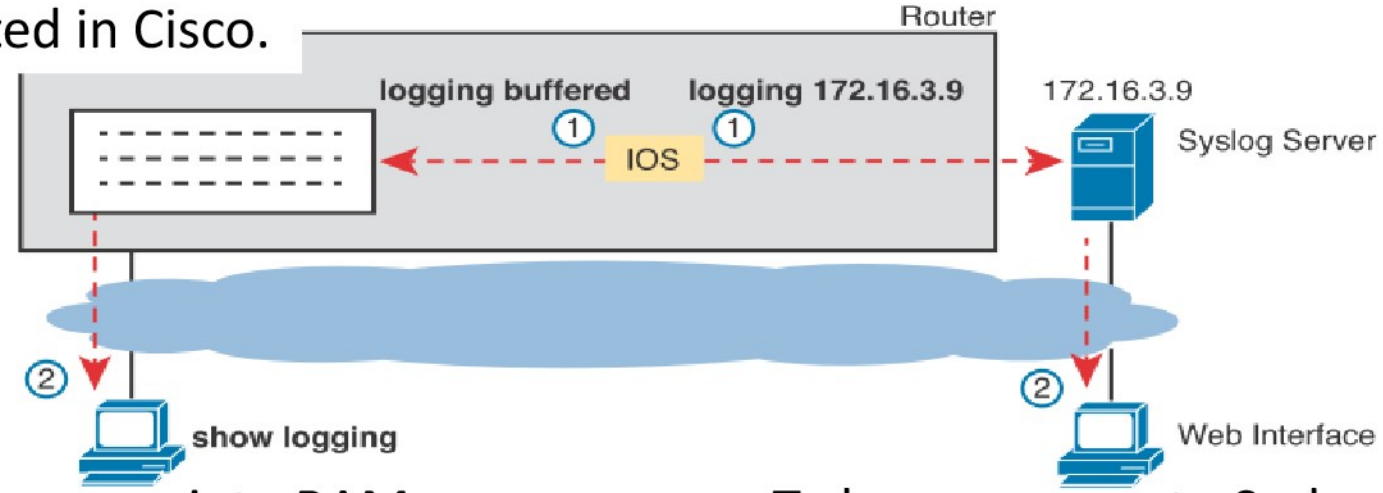
```
R1(config)# logging monitor
```

In addition, at B:

```
R1# terminal monitor
```

For monitoring of network security, it is more useful to store log messages in the device RAM, or even better to a centralised Syslog server.

By default, Syslog messages are transported over UDP port 514 (RFC 5426) or more securely over TLS over TCP port 6514 (RFC 5425) – may not be supported in Cisco.



To log messages into RAM:

```
R1(config)# logging buffered
```

In addition:

```
R1# show logging
```

To log messages to Syslog server:

```
R1(config)# logging
172.16.3.9
R1(config)# logging trap 4
```

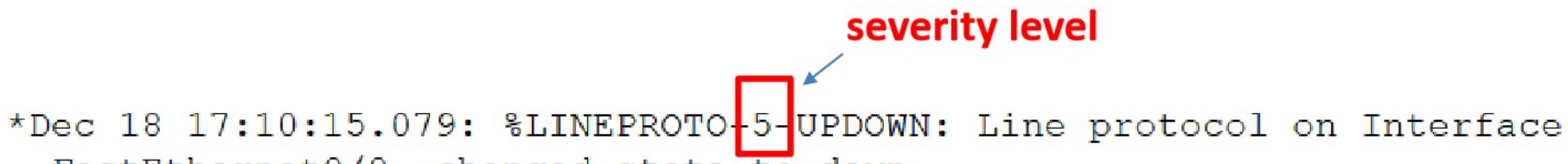
All errors 4 or below (more serious) are logged.

To prevent cluttering the log with too much data, Syslog has a severity level rating which you can use it to control which messages to log.

Keyword	Numeral	Description	
Emergency	0	System unusable	Severe
Alert	1	Immediate action required	
Critical	2	Critical Event (Highest of 3)	Impactful
Error	3	Error Event (Middle of 3)	
Warning	4	Warning Event (Lowest of 3)	
Notification	5	Normal, More Important	Normal
Informational	6	Normal, Less Important	
Debug	7	Requested by User Debug	Debug

Service	To Set Message Levels
Console	logging console *level-name* | *level-number*
Monitor	logging monitor *level-name* | *level-number*
Buffered	logging buffered *level-name* | *level-number*
Syslog	logging trap *level-name* | *level-number*

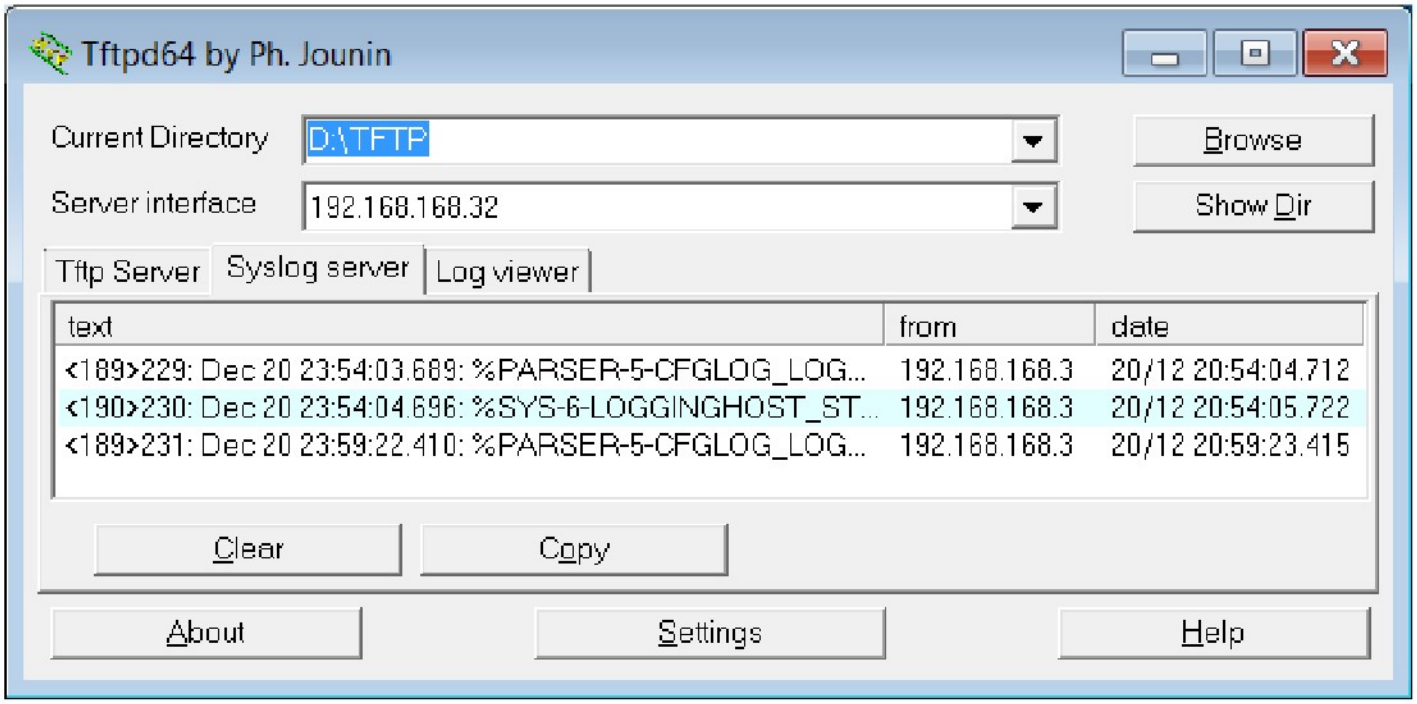
An example of Syslog message which shows the date/time of occurrence, the severity level and the description.



- **A timestamp:** *Dec 18 17:10:15.079
- **The facility on the router that generated the message:** %LINEPROTO
- **The severity level:** 5
- **A mnemonic for the message:** UPDOWN
- **The description of the message:** Line protocol on Interface FastEthernet0/0, changed state to down

There are many syslog servers available. An example is the free open source tftpd64 server which can also function as syslog server that you can try in the lab.

https://pjo2.github.io/tftpd64/



Alternatively, you may also try other Syslog servers like Kiwi Syslog which offers a free edition supporting up to 5 clients.

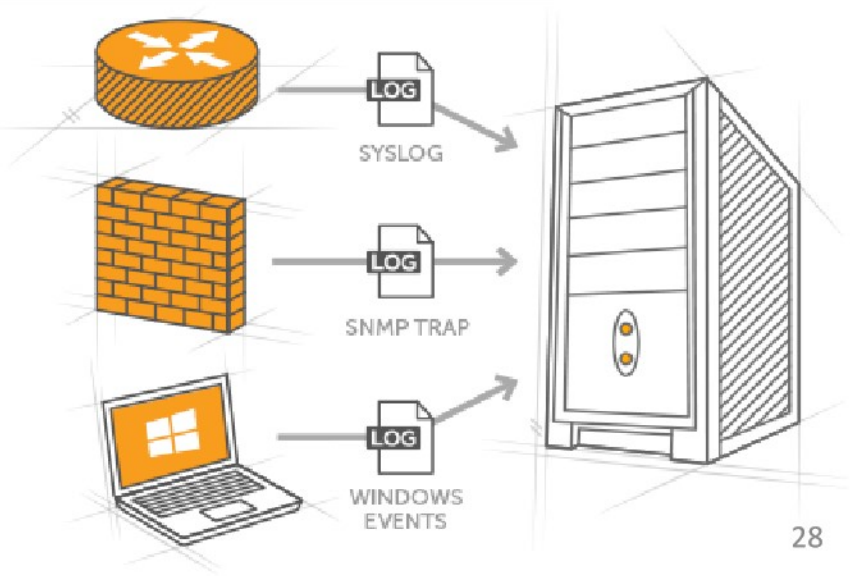
https://www.kiwisyslog.com/free-tools/kiwi-free-syslog-server

KIWI SYSLOG WEB ACCESS v1.5.1

Events | Filters | Highlighting | Settings | Admin

Filter: [All Kiwi Syslog Server Events]

Date	Time	Facility	Level	Host Name	Message Text
2015-03-20	14:05:00	Cron	Warning	10.111.201.39	Logon failed
2015-03-20	14:05:00	Local1	Warning	10.111.201.39	Logoff Successful
2015-03-20	14:05:00	Local7	Critical	10.111.201.39	This is a test message generated by Kiwi SyslogGen
2015-03-20	14:05:00	Local1	Alert	10.111.201.39	Logon successful
2015-03-20	14:04:59	Cron	Error	10.111.201.39	Logon failed
2015-03-20	14:04:59	System1	Critical	10.111.201.39	Logoff Successful
2015-03-20	14:04:59	Syslog	Error	10.111.201.39	This is a test message
2015-03-20	14:04:59	System0	Critical	10.111.201.39	Logon successful
2015-03-20	14:04:58	Cron	Warning	10.111.201.39	Logon failed
2015-03-20	14:04:58	System2	Info	10.111.201.39	Logoff Successful
2015-03-20	14:04:58	User	Notice	10.111.201.39	This is a test message
2015-03-20	14:04:58	News	Debug	10.111.201.39	Logon successful
2015-03-20	14:04:57	Cron	Info	10.111.201.39	Logon failed



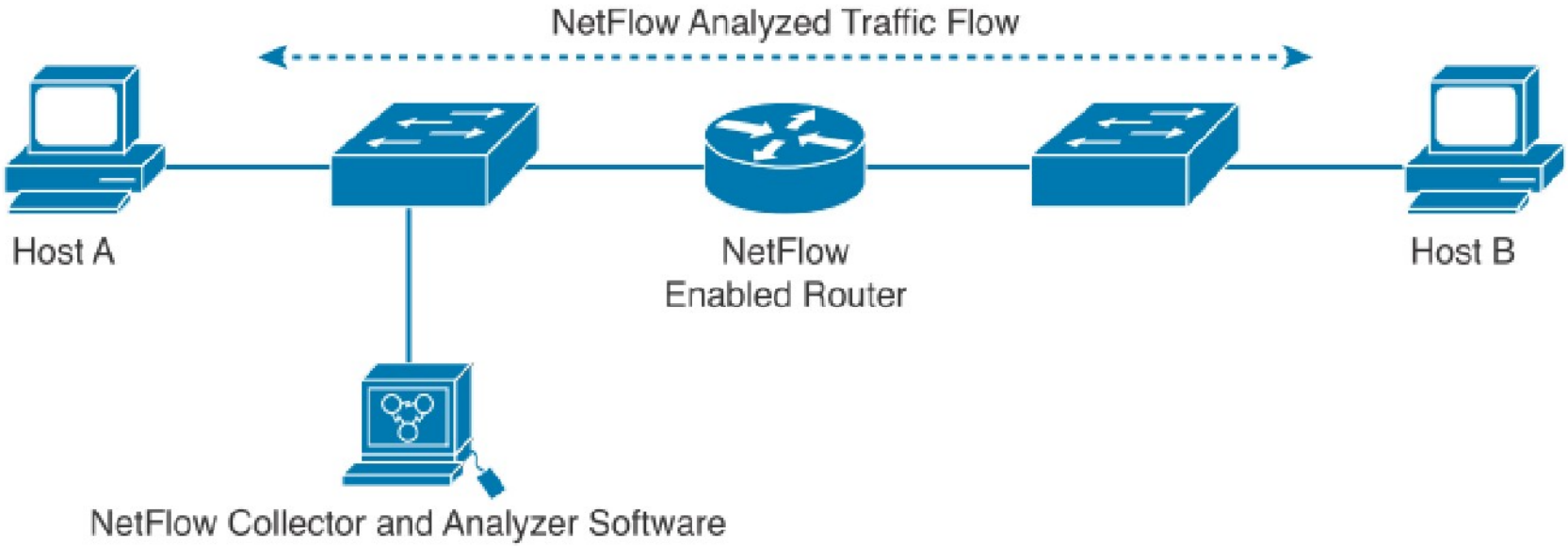
Lab Exercise 2

SINGAPORE INSTITUTE OF TECHNOLOGY

2.1 Configure NetFlow to provide statistical records of different IP packets that flow through a router.

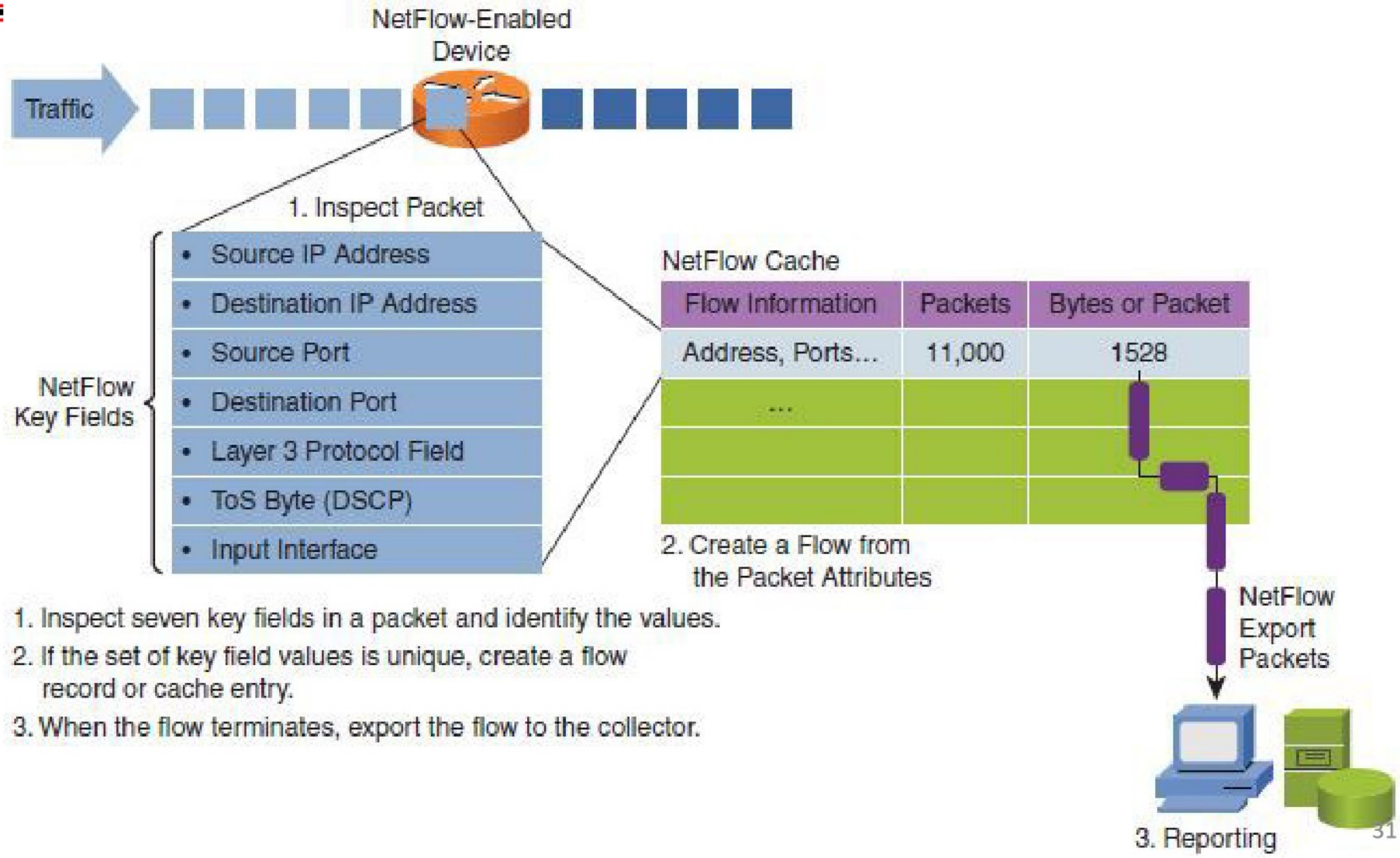
To support network traffic monitoring, Cisco has developed NetFlow (RFC 3954) to collect statistics on IP packets flowing through network devices.

When enabled, NetFlow keeps statistics of different IP packets flowing through a router in a NetFlow cache, which can then be exported to a NetFlow Collector for centralized logging and analysis.



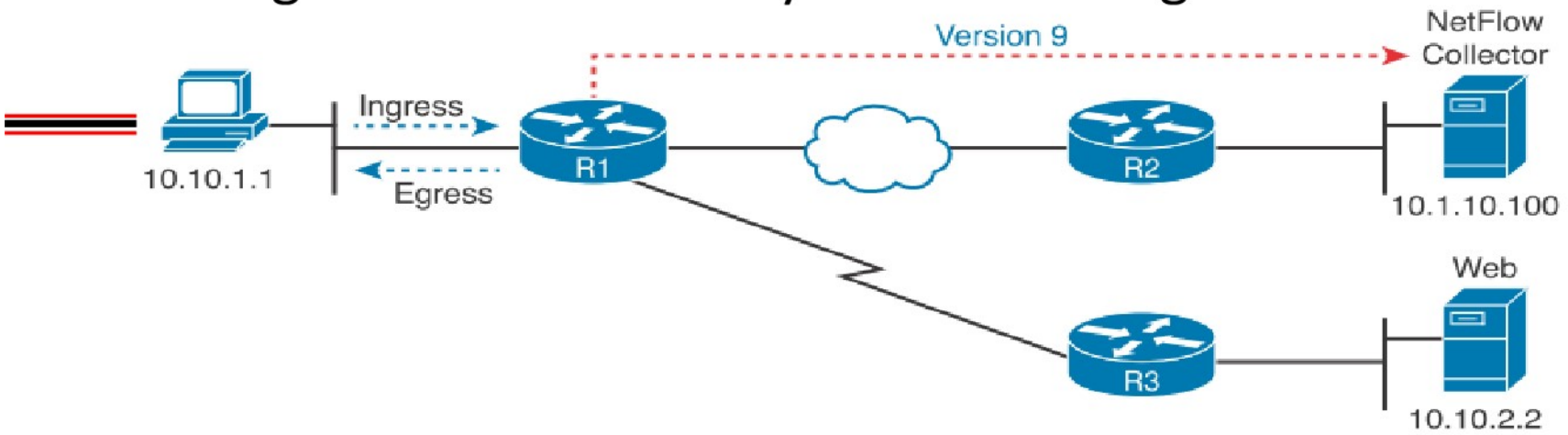
NetFlow has proven valuable and today different variants have been implemented, e.g. IPFIX (RFC 7011, derived from NetFlow), sFlow (RFC 3176) and JFlow (Juniper).

A flow is defined as a unidirectional sequence of packets that pass through a network device having the same values for certain key fields in the packet headers, examples:



1. Inspect seven key fields in a packet and identify the values.
2. If the set of key field values is unique, create a flow record or cache entry.
3. When the flow terminates, export the flow to the collector.

The simplest way to enable NetFlow is to rely on default matching rules and thus only need to configure the followings:




```
R1(config)# interface fastethernet0/0
R1(config-if)# ip flow ingress
R1(config-if)# ip flow egress
R1(config-if)# exit
R1(config)# ip flow-export destination 10.1.10.100 99
R1(config)# ip flow-export version 9
R1(config)# ip flow-export source loopback 0
R1(config)# end
```

Monitor incoming packets

Monitor outgoing packets

Port number

current version

For easier identification of router, use loopback 0 IP address to send NetFlow packets to Collector.

NetFlow is transported over UDP (port not standardized in RFC).

To verify NetFlow configuration, use the following commands:

```
R1# show ip flow interface
FastEthernet0/0
  ip flow ingress
  ip flow egress

R1# show ip flow export
Flow export v9 is enabled for main cache
  Export source and destination details :
  VRF ID : Default
   Source(1)        1.1.1.1 (Loopback0)
   Destination(1)   10.1.10.100 (99)
  Version 9 flow records
  0 flows exported in 0 udp datagrams
  0 flows failed due to lack of export packet
  0 export packets were sent up to process level
  0 export packets were dropped due to no fib
  0 export packets were dropped due to adjacency issues
  0 export packets were dropped due to fragmentation failures
  0 export packets were dropped due to encapsulation fixup failures
R1#
```

To view statistics stored locally in the NetFlow cache, use the command:

```
R1# show ip cache flow
IP packet size distribution (5727 total packets):
   1-32   64   96  128  160  192  224  256  288  320  352  384  416  448  480
   .000 .147 .018 .700 .000 .001 .001 .001 .001 .011 .009 .001 .002 .000 .001

    512  544  576 1024 1536 2048 2560 3072 3584 4096 4608
   .001 .001 .097 .000 .000 .000 .000 .000 .000 .000 .000

  .
  .
  .
Protocol          Total    Flows   Packets Bytes  Packets Active(Sec) Idle(Sec)
--------          Flows     /Sec     /Flow  /Pkt     /Sec       /Flow     /Flow
TCP-Telnet            4      0.0        27    43      0.2         5.0      15.7
TCP-WWW             104      0.2        14   275      3.4         2.1       1.5
ICMP                  4      0.0      1000   100      8.8        27.9      15.4

SrcIf         SrcIPaddress    DstIf         DstIPaddress    Pr SrcP DstP  Pkts
  .
  .
  .
```

Instead of relying on default matching rules, NetFlow version 9 allows the flexibility to configure matching rules and collection statistics, hence also called flexible NetFlow.

(1) Example of configuring matching rules and collection statistics using flow record:

```
R1(config)# flow record my-record
R1(config-flow-record)# match ipv4 source address
R1(config-flow-record)# match ipv4 destination address
R1(config-flow-record)# match transport source-port
R1(config-flow-record)# match transport destination-port
R1(config-flow-record)# match ipv4 protocol
R1(config-flow-record)# match ipv4 tos
R1(config-flow-record)# match interface input
R1(config-flow-record)# collect counter bytes
R1(config-flow-record)# collect counter packets
```

Next, apply the specific matching rules and collection statistics configured onto the interface of the router as follows:

(2) Configuring flow exporter:

```
R1(config)# flow exporter my-exporter
R1(config-flow-exporter)# destination 10.1.10.100
R1(config-flow-exporter)# transport udp 99
R1(config-flow-exporter)# source lo0
R1(config-flow-exporter)# export-protocol netflow-v9
```

(3) Configuring flow monitor:

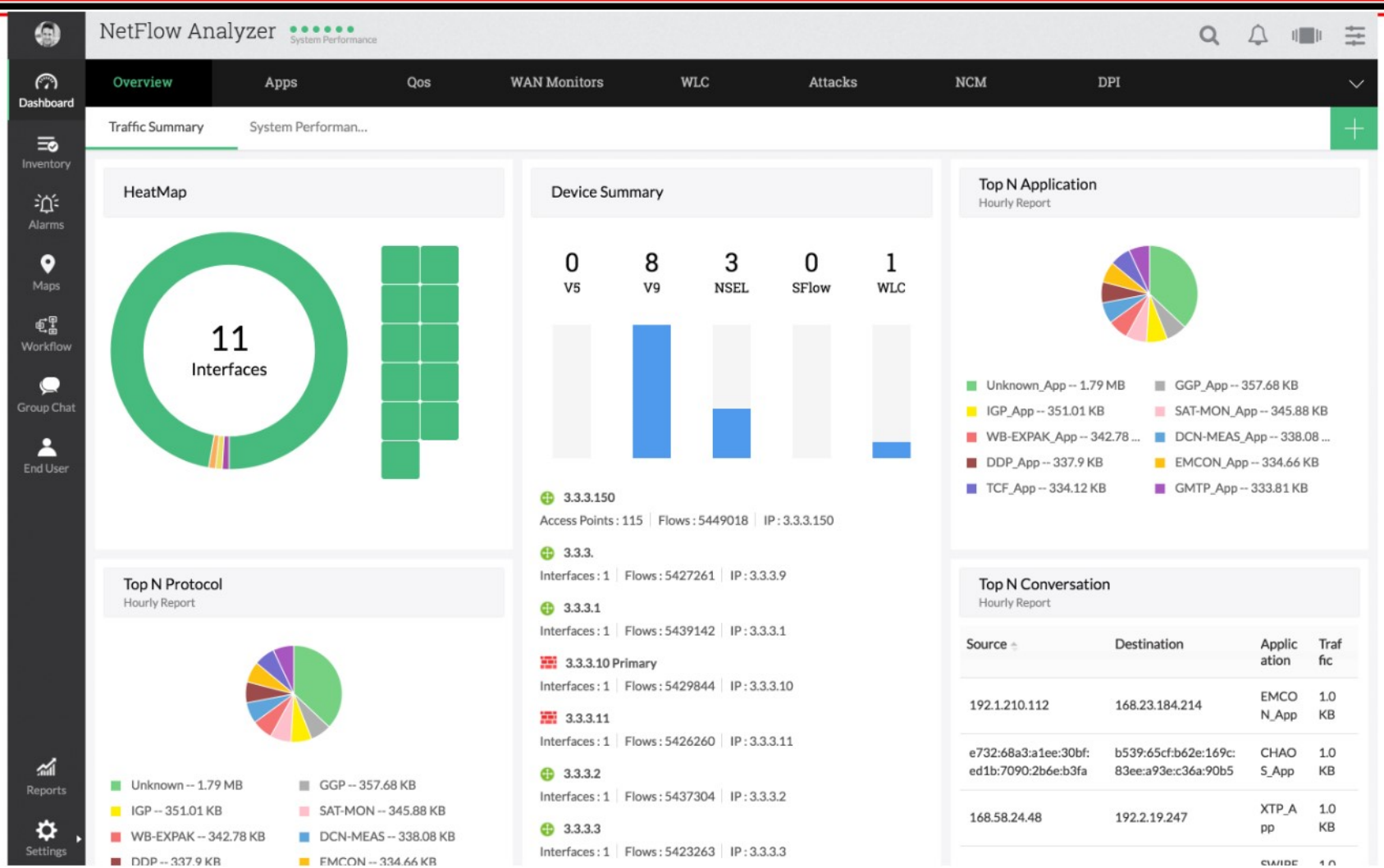
```
R1(config)# flow monitor my-monitor
R1(config-flow-monitor)# exporter my-exporter
R1(config-flow-monitor)# record my-record
```

(4) Applying flow monitor onto router interface:

```
R1(config)# interface g0/0
R1(config-if)# ip flow monitor my-monitor {input|output}
```

An example of NetFlow collector is the NetFlow Analyzer by ManageEngine which you can try in your team project.

https://www.manageengine.com/products/netflow/download-free.html



Today, instead of separate logs, SIEM (Security Information and Event Management) solution is getting common to provide unified real-time correlation of all logs and threat alerting.

Figure 1: Magic Quadrant for Security Information and Event Management