

Lab 2

Secondary System Configuration & Operational Monitoring and Maintenance

Overview

This lab demonstrates typical secondary configuration tasks performed on JUNOS devices. The second section of this lab covers common operational monitoring and platform maintenance activities. In this lab, you monitor system, chassis, and interface operation, use network utilities, and perform system maintenance tasks.

All devices are connected to a common management network which facilitates access to the CLI. These exercises assume you already have some basic understanding of the JUNOS CLI interfaces or you have read the IJS documentation or similar.

Note that your *lab* login (password given to you separately) grants you all permissions needed to complete this lab; however, some restrictions have been made to prevent loss of connectivity to the devices. Please be careful, and have fun!

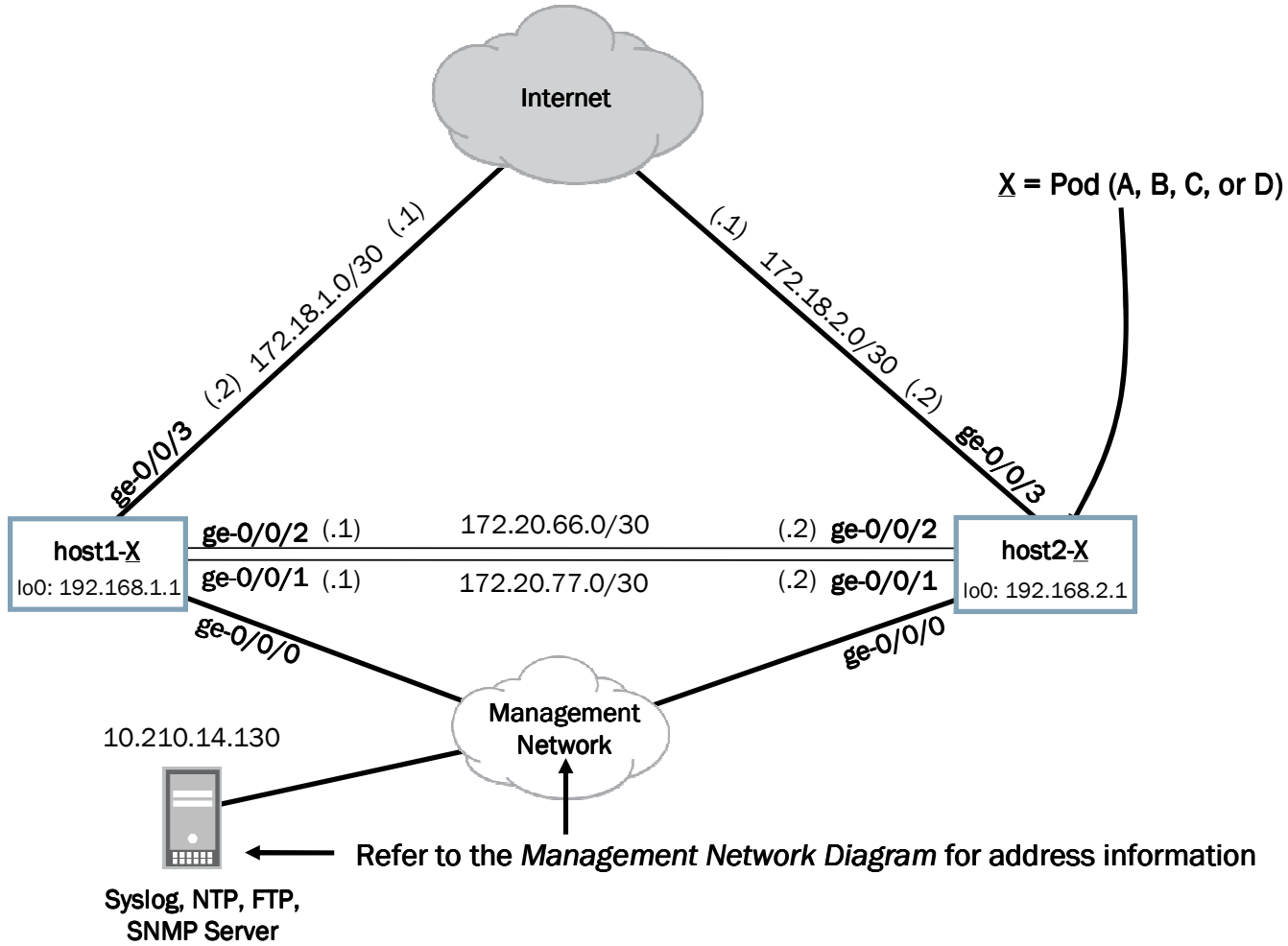
By completing this lab, you will perform the following tasks:

- Set up and verify proper operation of system logging (syslog)
- Configure and monitor NTP
- Enable and monitor the operation of SNMP
- Configure and monitor the configuration archival feature
- Monitor chassis, system, and interface operation
- Use network utilities

Please refer to the next page lab diagram to perform this exercise:

Lab Diagram

Network Diagram



Key Commands

Key *operational* mode commands used in this lab include the following:

```
file list
show log
show ntp associations
show system uptime
show interfaces terse
show interfaces extensive
show snmp statistics
show snmp walk
show system processes
show system storage
show system users
show chassis routing-engine
show chassis hardware
ping
monitor traffic interface
```

Part 1: Performing System Management Options

The objective of this lab part is to perform configuration of some common system management features. You will configure and monitor system logging, NTP, SNMP and configuration archival.

To do most of this lab you can use only one of the systems that have been assigned to you (either host1-x or host2-x). You only need to configure the other system in the Monitoring section to validate your configs and be able to ping the other end station.

Note

Please do NOT delete interface ge-0/0/0 as this is your management interface which provides access to your session and the J-Web !!

Do NOT delete either the security section of your configurations. This allows your system to allow any traffic in/out.

Note 2

During this lab, your access through the management network will be affected. Ensure that you use the console connection to access your assigned station. Using the console connection ensures persistent connectivity even when the management network access is unavailable.

Note 3

Remember that the exercise proposed in this documentation is generic and the examples given here apply only to one particular pod of devices. Please adapt the example to your assigned set of devices (host1-a & host2-a, or host1-b & host2-b, or host1-c & host2-c, or host1-d & host2-d).

Look at you lab diagram and mind the pod of systems that you have been assigned!

Step 1.1

Log in to the system with the username *lab* using the password given to you. Please use the console connection to access your system.

Display the configuration's system syslog hierarchy.

```
host1-a (tty2)

login: lab
Password:

--- JUNOS 9.6R1.13 built 2009-08-01 09:23:09 UTC

lab@host1-a> configure
Entering configuration mode

[edit]
lab@host1-a# show system syslog

user *{
    any emergency;
}
file messages {
    any any;
    authorization info;
}
file interactive-commands {
    interactive-commands any;
}

[edit]
lab@host1-a#
```

- ◆ Based on the display, what facilities and severity levels currently log to the messages log file?

- ❖ In the sample output, the messages file shows the `any` and `authorization` facilities using the `any` and `info` severities respectively.

- ◆ What is the purpose of specifying a facility of `any`?

- ❖ This option logs all facility levels.

Step 1.2

Navigate to the `[edit system syslog]` hierarchy and configure a new system log file named `config-changes`. Specify a facility of `change-log` and a severity of `info`.

```
[edit]
lab@host1-a# edit system syslog

[edit system syslog]
lab@host1-a# set file config-changes change-log info

[edit system syslog]
lab@host1-a#
```

Step 1.3

Configure your system to send logs to a remote server running the standard syslog utility. Refer to your management network diagram for the server address. Choose the correct facility that logs access attempts on the system. (Hint: The current messages log file is already utilizing this facility.) Use a severity level of `info`. Commit your changes and exit configuration mode using the **commit and-quit** command.

```
[edit system syslog]
lab@host1-a# set host 10.210.14.130 authorization info

[edit system syslog]
lab@host1-a# commit and-quit

commit complete
Exiting configuration mode

lab@host1-a>
```

Step 1.4

Using the `file list /var/log/` command, verify the creation of a log file named `config-changes`.

```
lab@host1-a> file list /var/log/

/var/log/:
.snap/
__jsrpd_commit_check__
authd_libstats
authd_profilelib
authd_sdb.log
autod
chassisd
config-changes
cosd
cscript.log
dateiname.txt
dateiname.txt.0.gz
```

```
dcd
dfwc
dfwd
eccd
gres-tp
hostname-cached
... TRIMMED...
```

Note

The files stored in the `/var/log/` directory might vary between each system.

- ◆ What other log files from your system's configuration does this directory store?

- ❖ Although the files in the `/var/log/` directory might vary on each system, the *messages* and *interactive-commands* log files should be present on all systems.

Step 1.5

Configure the system to synchronize its clock with an NTP server. Refer to the Management network diagram for the server's IP address.

```
lab@host1-a> configure
Entering configuration mode
[edit]
lab@host1-a# set system ntp server 10.210.14.130
[edit]
lab@host1-a#
```

Step 1.6

Use the same server IP address used in the previous step and configure an NTP boot server. Commit the configuration and return to operational mode.

```
[edit]
lab@host1-a# set system ntp boot-server 10.210.14.130
[edit]
lab@host1-a# show system ntp

boot-server 10.210.14.130;
server 10.210.14.130;
```

```
[edit]
lab@host1-a# commit and-quit

commit complete
Exiting configuration mode

lab@host1-a>
```

Step 1.7

View the *config-changes* log and verify the logging of the latest configuration changes.

```
lab@host1-a> show log config-changes

Jul 17 12:12:23 host1-a mgd[37121]: UI_CFG_AUDIT_SET: User 'lab' set: [system ntp
boot-server] "10.210.14.130 -> "10.210.14.130"

lab@host1-a>
```

Step 1.8

Manually force synchronization with the NTP server by issuing the **set date ntp** operational mode command.

```
lab@host1-a> set date ntp

17 Jul 12:14:03 ntpdate[38392]: step time server 10.210.14.130 offset -0.000248
sec
```

Step 1.9

Verify synchronization with the NTP server by using the **show ntp associations** command. The system is synchronized with the NTP server if you see the server address in the *remote* column with an asterisk (*) next to it. Check the current system time using the **show system uptime** command

Note

It might take a few minutes for the system's time to synchronize with the NTP server.

```
lab@host1-a> show ntp associations

      remote          refid          st t when poll reach  delay  offset  jitter
=====
*10.210.14.130    131.211.84.189    2 -   33   64    3   1.021   0.077   1.183
```

```
lab@host1-a> show system uptime

Current time: 2010-07-17 12:16:20 UTC
System booted: 2010-06-19 11:22:37 UTC (4w0d 00:53 ago)
Protocols started: 2010-06-22 14:27:36 UTC (3w3d 21:48 ago)
Last configured: 2010-07-17 12:12:35 UTC (00:03:45 ago) by lab
12:16PM up 28 days, 54 mins, 1 user, load averages: 3.02, 3.05, 3.01
```

- ◆ What does the asterisk (*) next to the NTP server address signify?

- ❖ The asterisk (*) represents the peer chosen for synchronization. When you define multiple NTP peers, the system selects only a single NTP peer.

Step 1.10

Return to configuration mode and configure the system to allow SNMP access using a community value of *junos*. The system should allow processing of SNMP messages only when it receives them from the NMS server's IP address. Refer to the management network diagram for the server's IP address.

```
lab@host1-a> configure
Entering configuration mode

[edit]
lab@host1-a# set snmp community junos clients 10.210.14.130

[edit]
lab@host1-a#
```

Step 1.11

Configure an SNMP trap group to send traps to the NMS server. The SNMP trap group should send traps whenever an interface transitions to a down state. Name the trap group *interfaces*. Commit your configuration when ready

```
[edit]
lab@host1-a# set snmp trap-group interfaces targets 10.210.14.130

[edit]
lab@host1-a# set snmp trap-group interfaces categories link

[edit]
lab@host1-a# show snmp

community junos {
  clients {
    10.210.14.130/32;
  }
}
trap-group interfaces {
  categories {
    link;
  }
  targets {
    10.210.14.130;
  }
}
```

```
[edit]
lab@host1-a# commit
commit complete
```

- ◆ What trap category would you enable to receive traps for an over-temperature condition?

- ❖ You would enable the *chassis* category to send traps for an over-temperature condition.

Note

In subsequent steps you will disable the *ge-0/0/1* interface. Ensure the terminal session to your system uses the console connection.

Step 1.12

To test your SNMP configuration, temporarily disable the *ge-0/0/1* interface using the **set interfaces ge-0/0/1 disable** command. Commit the new setting and verify that the interface is down using the **run show interfaces ge-0/0/1 terse** command. Next, re-enable the interface by issuing the **delete interfaces ge-0/0/0 disable** command (alternatively you could do instead a **rollback 1**)

Commit the change and return to operational mode.

```
[edit]
lab@host1-a# set interfaces ge-0/0/1 disable
```

```
[edit]
lab@host1-a# commit
commit complete
```

```
[edit]
lab@host1-a# run show interfaces ge-0/0/1 terse
```

Interface	Admin	Link	Proto	Local	Remote
ge-0/0/1	down	down			

```
[edit]
lab@host1-a# delete interfaces ge-0/0/1 disable
```

```
[edit]
lab@host1-a# commit and-quit
commit complete
Exiting configuration mode
```

```
lab@host1-a>
```

Step 1.13

Verify that the interface transition resulted in the sending of a trap by viewing the `messages` log. Use the pipe symbol (`|`) and match on the `ge-0/0/1` interface and the keyword `snmp` to parse the `messages` log output. Next, issue the `show snmp statistics` command and confirm that the `Traps` value in the `Output` section is not zero.

```
lab@host1-a> show log messages | match ge-0/0/1 | match snmp
```

```
Jul 15 12:34:18 host1-a rpd[6949]: EVENT <SNMP Index> ge-0/0/1.0 index 64 <Up
Broadcast Multicast> address #0 0.26.88.e9.d2.81
Jul 15 12:34:18 host1-a mib2d[971]: SNMP_TRAP_LINK_UP: ifIndex 197, ifAdminStatus
up(1), ifOperStatus up(1), ifName ge-0/0/1.0
Jul 17 12:28:53 host1-a mib2d[971]: SNMP_TRAP_LINK_DOWN: ifIndex 120,
ifAdminStatus down(2), ifOperStatus down(2), ifName ge-0/0/1
Jul 17 12:29:30 host1-a mib2d[971]: SNMP_TRAP_LINK_UP: ifIndex 120, ifAdminStatus
up(1), ifOperStatus up(1), ifName ge-0/0/1
Jul 17 12:29:53 host1-a mgd[37121]: UI_CMDLINE_READ_LINE: User 'lab', command
'run show log messages | match ge-0/0/1 | match snmp '
```

```
lab@host1-a> show snmp statistics
```

```
SNMP statistics:
```

```
Input:
```

```
Packets: 0, Bad versions: 0, Bad community names: 0,
Bad community uses: 0, ASN parse errors: 0,
Too bigs: 0, No such names: 0, Bad values: 0,
Read onlys: 0, General errors: 0,
Total request varbinds: 0, Total set varbinds: 0,
Get requests: 0, Get nexts: 0, Set requests: 0,
Get responses: 0, Traps: 0,
Silent drops: 0, Proxy drops: 0, Commit pending drops: 0,
Throttle drops: 0, Duplicate request drops: 0
```

```
V3 Input:
```

```
Unknown security models: 0, Invalid messages: 0
Unknown pdu handlers: 0, Unavailable contexts: 0
Unknown contexts: 0, Unsupported security levels: 0
Not in time windows: 0, Unknown user names: 0
Unknown engine ids: 0, Wrong digests: 0, Decryption errors: 0
```

```
Output:
```

```
Packets: 4, Too bigs: 0, No such names: 0,
Bad values: 0, General errors: 0,
Get requests: 0, Get nexts: 0, Set requests: 0,
Get responses: 0, Traps: 4
```

◆ Does the `messages` log show trap entries associated with the interface status change?

❖ Yes, you should see log entries for the status change for both the physical and the

logical interfaces.

- ◆ Does the **show snmp statistics** command list a non-zero value for outgoing traps?

- ❖ Yes, you should see a non-zero value for the output traps counter. In the sample output, you can see a value of 4. Your counter's value might vary.

Step 1.11

Perform an SNMP MIB walk with the JUNOS Software CLI using the **show snmp mib walk jnxOperatingDescr** command. Note that the resolved object identifier (OID) of *jnxOperatingDescr* is case sensitive. The OID is variable; we are simply using this OID as an example.

```
lab@host1-a> show snmp mib walk jnxOperatingDescr
jnxOperatingDescr.1.1.0.0 = midplane
jnxOperatingDescr.4.1.0.0 = SRX240 PowerSupply fan 1
jnxOperatingDescr.4.2.0.0 = SRX240 PowerSupply fan 2
jnxOperatingDescr.4.3.0.0 = SRX240 CPU fan 1
jnxOperatingDescr.4.4.0.0 = SRX240 CPU fan 2
jnxOperatingDescr.7.1.0.0 = FPC: FPC @ 0/*/*
jnxOperatingDescr.8.1.1.0 = PIC: 16x GE Base PIC @ 0/0/*
jnxOperatingDescr.9.1.0.0 = Routing Engine
jnxOperatingDescr.9.1.1.0 = USB Hub
```

Note

JUNOS Software accepts both the dotted-decimal notation and alpha-numeric notation of SNMP MIB OIDs. The previous example polls the Juniper Networks Chassis MIB for a mapping of component OIDs. This tool is helpful for deciphering what component might be initiating an SNMP trap when your NMS station reports the OID in only a dotted-decimal notation. You do not need to configure SNMP to perform SNMP polling from within JUNOS Software.

- ◆ What OID associates with the Routing Engine (RE) for your system?

- ❖ The RE associates with the 9.1.0.0 OID leaf. This leaf is merely one leaf in the MIB tree and does not represent the full OID string.

Step 1.15

Enter configuration mode and configure your system to archive its configuration to a remote FTP server whenever a commit operation occurs. You should configure the `archive-sites` as `"ftp://ftp@10.210.14.130:/archive"` including the quotation marks, where `10.210.x.y` represents the server's IP address as specified on the management network diagram. You should configure the password as `ftp`. You perform this configuration under the `[edit system archival configuration]` hierarchy level. Commit your configuration and return to operational mode.

```
lab@host1-a> configure
Entering configuration mode

[edit]
lab@host1-a# edit system archival configuration

[edit system archival configuration]
lab@host1-a# set archive-sites "ftp://ftp@10.210.14.130:/archive" password ftp

[edit system archival configuration]
lab@host1-a# set transfer-on-commit

[edit system archival configuration]
lab@host1-a# commit and-quit
commit complete
Exiting configuration mode

lab@host1-a>
```

Step 1.16

Verify that the configuration successfully transferred to the remote FTP server using the `show log messages | match transfer` command.

```
lab@host1-a> show log messages | match transfer

Jul 17 19:42:01 host1-a mgd[40478]: UI_CFG_AUDIT_SET: User 'lab' set: [system
archival configuration] <unconfigured> -> "transfer-on-commit"
Jul 17 19:42:01 host1-a mgd[40478]: UI_CMDLINE_READ_LINE: User 'lab', command
'set transfer-on-commit '
Jul 17 19:42:29 host1-a mgd[40478]: UI_CMDLINE_READ_LINE: User 'lab', command
'show log messages | match transfer '
Jul 17 19:42:38 host1-a logger: transfer-file: Transferred
/var/transfer/config/host1-a_juniper.conf.gz_20100717_194217
Jul 17 19:44:11 host1-a mgd[40478]: UI_CMDLINE_READ_LINE: User 'lab', command
'show log messages | match transfer '
```

Note

Even when using the `transfer-on-commit` option with configuration archival, the transfer is cyclical and uses a short time interval. If you do not see the transfer in your log, wait a minute or two and look again.

- ◆ What do the numbers at the end of the transferred filename represent?

- ❖ The configuration file contains the current date and UTC time according to the system clock.

Part 2: Monitoring System and Chassis Operation

In this lab part, you will use key commands within the CLI to monitor system and chassis operation.

Step 2.1

Issue the `show system processes extensive` command to check the status of the routing protocol process (rpd). Alternatively, issue the `show system processes extensive | match "PID | rpd"` command to parse the output. The use of two pipes (|) in this command allows you to make multiple matches. In this case it matches `rpd` for the routing protocol process as well as `PID` to view the column headers.

```
lab@host1-a> show system processes extensive | match "PID | rpd"
  PID USERNAME      THR PRI NICE   SIZE    RES STATE  C   TIME   WCPU COMMAND
  6949 root              1   4    0 34872K 18120K kqread 0   6:35  0.00% rpd
```

- ◆ What is the weighted CPU usage of rpd?

- ❖ The answer can vary. In the sample output taken from `host1-a`, the weighted CPU usage is 0%. The weighted CPU column represents the CPU usage over a period of time.

Step 2.2

Issue the `show system statistics` command to view protocol statistics related to your device.

```
lab@host1-a> show system statistics
Tcp:
  24850 packets sent
    22799 data packets (4266126 bytes)
    47 data packets retransmitted (6757 bytes)
    0 resends initiated by MTU discovery
    1808 ack only packets (627 packets delayed)
    0 URG only packets
    31 window probe packets
    0 window update packets
```

```
188 control packets
37004 packets received
21775 acks(for 4265961 bytes)
1344 duplicate acks
0 acks for unsend data
16453 packets received in-sequence(358689 bytes)
630 completely duplicate packets(6 bytes)
...TRIMMED...
```

- ◆ How many TCP packets did your assigned device send since the last clearing of the system statistics?

- ❖ The answer can vary. In the previous example taken from *host1-a*, the device sent 24850 TCP packets

Step 2.3

Issue the **show system storage** command to view information regarding the device storage space.

```
lab@host1-a> show system storage
```

Filesystem	Size	Used	Avail	Capacity	Mounted on
/dev/da0s1a	898M	251M	575M	30%	/
devfs	1.0K	1.0K	0B	100%	/dev
devfs	1.0K	1.0K	0B	100%	/dev/
/dev/md0	477M	477M	0B	100%	/junos
/cf	898M	251M	575M	30%	/junos/cf
devfs	1.0K	1.0K	0B	100%	/junos/dev/
procfs	4.0K	4.0K	0B	100%	/proc
/dev/bo0s1e	24M	32K	22M	0%	/config
/dev/md1	168M	13M	141M	8%	/mfs
/dev/da0s1f	61M	5.4M	51M	10%	/cf/var/log
/cf/var/jail	898M	251M	575M	30%	/jail/var
devfs	1.0K	1.0K	0B	100%	/jail/dev
/dev/md2	39M	4.0K	36M	0%	/mfs/var/run/utm

- ◆ How much free space is available on your device?

- ❖ The answer can vary. In the sample output, 575 Megabytes are available.

Step 2.4

Issue the **show system uptime** command to view the current system time.

```
lab@host1-a> show system uptime
Current time: 2010-07-17 20:07:15 UTC
System booted: 2010-06-19 11:22:37 UTC (4w0d 08:44 ago)
Protocols started: 2010-06-22 14:27:36 UTC (3w4d 05:39 ago)
Last configured: 2010-07-17 19:42:16 UTC (00:24:59 ago) by lab
8:07PM up 28 days, 8:45, 1 user, load averages: 3.13, 3.07, 3.01
```

- ◆ When was your device last booted?

- ❖ The answer will vary. In the example taken from *host1-a*, you can see that the system booted about 4weeks and 8 hours ago.

Step 2.5

Open another terminal window to the jumpserver and use Telnet to access your system's management IP address. If needed, refer to the management network diagram. Log in one more time with the username **lab** and the password **lab123** (or the one being assigned to you)

```
[luis@js2 ~]$ telnet 10.210.14.131
Trying 10.210.14.131...
Connected to srx1-a.lab2.cavellgroup.com (10.210.14.131).
Escape character is '^]'.
host1-a (ttyp0)
login: lab
Password:
--- JUNOS 9.6R1.13 built 2009-08-01 09:23:09 UTC
lab@host1-a>
```

Return to the console session and issue the **show system users** command to view information about users logged in to your team's device.

```
lab@host1-a> show system users
8:13PM up 28 days, 8:51, 2 users, load averages: 3.09, 3.09, 3.02
USER      TTY      FROM          LOGIN@      IDLE WHAT
lab       u0       -             8:11PM      -  -cli (cli)
lab       p0       10.210.14.130 8:12PM      1  -cli (cli)
```

- ◆ How many times the user *lab* is logged in?

- ❖ Two times: one using the console (TTY *u0*) and a second one using a telnet session (TTY *p0*)

- ◆ What is the source IP address of the telnet session established by the user *lab*?

- ❖ In the following example taken from *host1-a*, the source IP address of the telnet session established by the user *lab* is 10.210.14.130.

Step 2.6

Issue the **request system logout user *lab*** command to force a log out for the user *lab* coming from the telnet session. Be careful here and kick out the telnet session (TTY *p0*) and not the one coming from the console. Next, issue the **show system users** command to verify the user session for *walter* has been terminated.

```
lab@host1-a> request system logout user lab terminal p0
```

```
lab@host1-a> show system users
```

```
8:20PM up 28 days, 8:58, 1 user, load averages: 3.00, 3.02, 3.00
USER      TTY      FROM          LOGIN@      IDLE WHAT
lab       u0       -             8:11PM      -   -cli (cli)
```

- ◆ Was the user Telnet session for *lab* properly closed?

- ❖ As shown in the sample output, the *p0* Telnet session for the user *lab* should now be closed

Step 2.7

Check the environmental status of your team's device by issuing the **show chassis environment** command.

```
lab@host1-a> show chassis environment
```

Class	Item	Status	Measurement
Temp	Routing Engine	OK	42 degrees C / 107 degrees F
Fans	SRX240 PowerSupply fan 1	OK	Spinning at high speed
	SRX240 PowerSupply fan 2	OK	Spinning at high speed
	SRX240 CPU fan 1	OK	Spinning at high speed
	SRX240 CPU fan 2	OK	Spinning at high speed
	SRX240 IO fan 1	OK	Spinning at high speed
	SRX240 IO fan 2	OK	Spinning at high speed
Power	Power Supply 0	OK	

- ◆ What is the temperature and status of the routing engine (RE)?

- ❖ Your details might vary. The sample capture shows a temperature of 42 degrees Celsius and a status of OK

- ◆ Name another **show chassis** command that displays the RE temperature. (Hint: Use the ?.)

- ❖ As the following capture shows, the **show chassis routing-engine** command displays the RE temperature as well as other RE-specific details.

```
lab@host1-a> show chassis routing-engine
```

```
Routing Engine status:
```

```
Temperature          42 degrees C / 107 degrees F
CPU temperature      39 degrees C / 102 degrees F
DRAM                1024 MB
Memory utilization   82 percent
CPU utilization:
  User              4 percent
  Background        0 percent
  Kernel            2 percent
  Interrupt         0 percent
  Idle              94 percent
Model                RE-SRX240-POE
Serial ID            AAAX5451
Start time           2010-06-19 11:22:37 UTC
Uptime               28 days, 9 hours, 20 minutes, 56 seconds
Last reboot reason   0x1:power cycle/failure
Load averages:      1 minute   5 minute   15 minute
                   0.02         0.07         0.07
```

Step 2.8

Issue the **show chassis temperature-thresholds** command.

```
lab@host1-a> show chassis temperature-thresholds
```

Item	Fan speed		Yellow alarm		Red alarm	
	Normal	High	Normal	Bad fan	Normal	Bad fan
Chassis default	48	54	65	55	75	65
Routing Engine	55	60	75	65	85	70

- ◆ At what temperature is a red alarm generated for the RE?

- ❖ Assuming the fans are operational, `host1-a` raises a red alarm when the RE reaches 85 degrees Celsius (sample output is from an SRX210). These threshold values might vary between different JUNOS devices. J4350 platforms, for example, have a threshold value of 90 degrees Celsius for the RE when fans are operational.

Step 2.9

View details about your system's hardware components using the **show chassis hardware** command.

```
lab@host1-a> show chassis hardware
```

Hardware inventory:

Item	Version	Part number	Serial number	Description
Chassis			AH4109AA0026	SRX240-poe
Routing Engine	REV 35	750-021794	AAAX5451	RE-SRX240-POE
FPC 0				FPC
PIC 0				16x GE Base PIC
Power Supply 0				

- ◆ What is the chassis serial number for your team's device?

- ❖ The answer will vary depending on your assigned device. In the example, the chassis serial number is AH4109AA0026.

Step 2.10

Refer to the network diagram for this lab and configure the listed interfaces. Use logical unit 0 on all specified interfaces and a /30 as a subnet mask for each of the interfaces, except from the lo0 (/32)

```
[edit]
lab@host1-a# edit interfaces

[edit interfaces]
lab@host1-a# set ge-0/0/1 unit 0 family inet address 172.20.77.1/30

[edit interfaces]
lab@host1-a# set ge-0/0/2 unit 0 family inet address 172.20.66.1/30

[edit interfaces]
lab@host1-a# set ge-0/0/3 unit 0 family inet address 172.18.1.2/30

[edit interfaces]
lab@host1-a# set lo0 unit 0 family inet address 192.168.1.1/32

[edit interfaces]
lab@host1-a#
```

Activate the configuration and return to operational mode.

```
[edit interfaces]
lab@host1-a# commit and-quit

commit complete
Exiting configuration mode

lab@host1-a>
```

Note

Please log into the other device assigned to you (host1-b) and configure the relevant interfaces too!

Step 2.11

Issue the **show interfaces terse** CLI command to verify the state of the configured interfaces.

```
lab@host1-a> show interfaces terse
```

Interface	Admin	Link	Proto	Local	Remote
ge-0/0/0	up	up			
ge-0/0/0.0	up	up	inet	10.210.14.131/27	
...TRIMMED...					
ge-0/0/1	up	up			
ge-0/0/1.0	up	up	inet	172.20.77.1/30	
ge-0/0/2	up	up			
ge-0/0/2.0	up	up	inet	172.20.66.1/30	
ge-0/0/3	up	up			
ge-0/0/3.0	up	up	inet	172.18.1.2/30	
ge-0/0/4	up	up			
ge-0/0/5	up	down			
ge-0/0/6	up	down			

```
ge-0/0/7          up    down
ge-0/0/8          up    down
ge-0/0/9          up    down
ge-0/0/10         up    down
ge-0/0/11         up    down
ge-0/0/12         up    down
ge-0/0/13         up    down
ge-0/0/14         up    down
ge-0/0/15         up    down
gre               up    up
ipip              up    up
lo0               up    up
lo0.0             up    up    inet    192.168.1.1    --> 0/0
...TRIMMED ...
```

- ◆ What is the Admin and Link state of the recently configured interfaces?

- ❖ All configured interfaces should show an Admin and Link state of up, as shown in the sample capture.

Step 2.12

Issue the **show interfaces ge-0/0/0 extensive** command and answer the questions that follow:

```
lab@host1-a> show interfaces ge-0/0/0 extensive
Physical interface: ge-0/0/0, Enabled, Physical link is Up
Interface index: 131, SNMP ifIndex: 118, Generation: 134
Description: MGMT Interface - DO NOT DELETE
Link-level type: Ethernet, MTU: 1514, Link-mode: Full-duplex, Speed: 1000mbps,
BPDU Error: None, MAC-REWRITE Error: None, Loopback: Disabled,
Source filtering: Disabled, Flow control: Enabled, Auto-negotiation: Enabled,
Remote fault: Online
Device flags      : Present Running
Interface flags: SNMP-Traps Internal: 0x0
Link flags       : None
CoS queues       : 8 supported, 8 maximum usable queues
Hold-times       : Up 0 ms, Down 0 ms
Current address: 00:26:88:e9:d2:80, Hardware address: 00:26:88:e9:d2:80
Last flapped    : 2010-06-22 15:32:18 UTC (3w4d 05:22 ago)
Statistics last cleared: 2010-06-22 16:07:39 UTC (3w4d 04:47 ago)
Traffic statistics:
Input bytes      :          101217679          232 bps
Output bytes     :           6056153           0 bps
Input packets    :          1645482           0 pps
Output packets   :           23501           0 pps
Input errors:
Errors: 0, Drops: 0, Framing errors: 0, Runts: 0, Policed discards: 72584,
```

Introduction to JUNOS Software & Routing Essentials

```
L3 incompletes: 0, L2 channel errors: 0, L2 mismatch timeouts: 0,
FIFO errors: 0, Resource errors: 0
Output errors:
  Carrier transitions: 0, Errors: 0, Drops: 0, Collisions: 0, Aged packets: 0,
...TRIMMED...
Logical interface ge-0/0/0.0 (Index 68) (SNMP ifIndex 119) (Generation 133)
Flags: SNMP-Traps Encapsulation: ENET2
Traffic statistics:
  Input bytes   :           101217799
  Output bytes  :           5308773
  Input packets:           1645484
...TRIMMED...
```

- ◆ What is the SNMP `ifIndex` for `ge-0/0/0`? What about for `ge-0/0/0.0`?

- ❖ The SNMP `ifIndex` values vary between student devices. In the example, the SNMP `ifIndex` for `ge-0/0/0` and `ge-0/0/0.0` are 118 and 119, respectively.

- ◆ What is the current hardware address for the `ge-0/0/0` interface?

- ❖ The current hardware address for the `ge-0/0/0` interface varies between student devices. In the example, the current hardware address is `00:26:88:e9:d2:80`

- ◆ Does the `ge-0/0/0` interface show any input errors?

- ❖ Although it is possible input errors exist, the answer to this question should typically be no.

- ◆ Does the `ge-0/0/0` interface show input and output traffic statistics? How are those statistics counted?

- ❖ The interface should show input and output traffic statistics. The system counts traffic statistics as both bytes and packets as shown in the sample capture.

Step 2.13

Issue the `clear interfaces statistics ge-0/0/0` command followed by the `show interfaces ge-0/0/0 extensive | find "traffic"` command.

```
lab@host1-a> clear interfaces statistics ge-0/0/0
```

```
lab@host1-a> show interfaces ge-0/0/0 extensive | find "traffic"
```

```
Traffic statistics:
```

```
Input bytes : 0 0 bps
Output bytes : 0 0 bps
Input packets: 0 0 pps
Output packets: 0 0 pps
```

- ◆ Were the statistics for the ge-0/0/0 interface successfully cleared?

- ❖ Although your statistics might not show all zeros, as the sample capture

Part 3: Using Network Utilities and Monitoring Traffic

In this lab part, you will use network utilities within the CLI and monitor local system traffic.

Step 3.1

Refer to the management network diagram and ascertain the IP address of the server attached to your management network. Ping the server. Specify a data size of 500 bytes. Ensure that the ping is continuous.

```
lab@host1-a> ping 10.210.14.130 size 500
```

```
PING 10.210.14.130 (10.210.14.130): 500 data bytes
508 bytes from 10.210.14.130: icmp_seq=0 ttl=64 time=3.147 ms
508 bytes from 10.210.14.130: icmp_seq=1 ttl=64 time=1.098 ms
508 bytes from 10.210.14.130: icmp_seq=2 ttl=64 time=1.090 ms
508 bytes from 10.210.14.130: icmp_seq=3 ttl=64 time=7.299 ms
...TRIMMED...
```

- ◆ Which command option do you use to make the ping continuous?

- ❖ As shown in the sample output, you do not need an extra command option to make the ping continuous. Echo requests send continuously by default. You can use the **count** option to send a defined amount of packets.

Note

You can stop the ping operation by using the Ctrl+c keystroke combination. You should, however, let the ping operation continue at this time for the subsequent monitoring step!!

Step 3.2

Open another terminal window to the jumpserver and use Telnet to access your system's management IP address. If needed, refer to the management network diagram. Log in one more time with the username **lab** and the password **lab123** (or the one being assigned to you)

```
[luis@js2 ~]$ telnet 10.210.14.131
Trying 10.210.14.131...
Connected to srx1-a.lab2.cavellgroup.com (10.210.14.131).
Escape character is '^]'.
host1-a (ttyp0)
login: lab
Password:
--- JUNOS 9.6R1.13 built 2009-08-01 09:23:09 UTC
lab@host1-a>
```

Step 3.3

Use the **monitor traffic interface ge-0/0/0** command to begin monitoring the ge-0/0/0 management interface.

Note

You can stop the monitoring operation by using the Ctrl+c keystroke combination. You can also increase the capture size using the **size** option to avoid truncated packets.

```
lab@host1-a> monitor traffic interface ge-0/0/0
verbose output suppressed, use <detail> or <extensive> for full protocol decode
Address resolution is ON. Use <no-resolve> to avoid any reverse lookup delay.
Address resolution timeout is 4s.
Listening on ge-0/0/0, capture size 96 bytes
```

Reverse lookup for 10.210.14.131 failed (check DNS reachability).
Other reverse lookup failures will not be reported.
Use <no-resolve> to avoid reverse lookups on IP addresses.

```
22:07:15.196093 Out IP truncated-ip - 235 bytes missing! 10.210.14.131.telnet >
10.210.14.130.44151: P 4185293323:4185293566(243) ack 3714343467 win 33304
<nop,nop,timestamp 2444575565 3099134743>
22:07:15.196920 In IP 10.210.14.130.44151 > 10.210.14.131.telnet: . ack 243 win
501 <nop,nop,timestamp 3099134778 2444575565>
22:07:15.621065 Out IP truncated-ip - 24 bytes missing! 10.210.14.131 >
10.210.14.130: ICMP echo request, id 40834, seq 135, length 64
22:07:15.627114 In IP 10.210.14.130 > 10.210.14.131: ICMP echo reply, id 40834,
seq 135, length 64
22:07:15.724033 In STP 802.1w, Rapid STP, Flags [Learn, Forward], bridge-id
8000.00:23:9c:11:f2:00.8216, length 43
22:07:16.204778 Out IP truncated-ip - 174 bytes missing! 10.210.14.131.telnet >
10.210.14.130.44151: P 243:425(182) ack 1 win 33304 <nop,nop,timestamp 2444576568
3099134778>
22:07:16.209258 In IP 10.210.14.130.44151 > 10.210.14.131.telnet: . ack 425 win
501 <nop,nop,timestamp 3099135753 2444576568>
22:07:16.209359 Out IP truncated-ip - 672 bytes missing! 10.210.14.131.telnet >
10.210.14.130.44151: P 425:1105(680) ack 1 win 33304 <nop,nop,timestamp 2444576573
3099135753>
22:07:16.215427 In IP 10.210.14.130.44151 > 10.210.14.131.telnet: . ack 1105 win
501 <nop,nop,timestamp 3099135756 2444576573>
22:07:16.627570 Out IP truncated-ip - 24 bytes missing! 10.210.14.131 >
10.210.14.130: ICMP echo request, id 40834, seq 136, length 64
22:07:16.628507 In IP 10.210.14.130 > 10.210.14.131: ICMP echo reply, id 40834,
seq 136, length 64
...TRIMMED...
```

- ◆ Does the capture display ICMP traffic?

- ❖ Yes, you should see ICMP echoes and replies from your ping operation, amongst other traffic.

- ◆ How can you filter the output to show only the ICMP traffic?

- ❖ Use the matching option to filter by header information in the output:

```
lab@host1-a> monitor traffic interface ge-0/0/0 matching icmp
```

verbose output suppressed, use <detail> or <extensive> for full protocol decode
Address resolution is ON. Use <no-resolve> to avoid any reverse lookup delay.

Introduction to JUNOS Software & Routing Essentials

```
Address resolution timeout is 4s.
Listening on ge-0/0/0, capture size 96 bytes

Reverse lookup for 10.210.14.131 failed (check DNS reachability).
Other reverse lookup failures will not be reported.
Use <no-resolve> to avoid reverse lookups on IP addresses.

22:11:24.215955 Out IP truncated-ip - 24 bytes missing! 10.210.14.131 >
10.210.14.130: ICMP echo request, id 40834, seq 382, length 64
22:11:24.216918 In IP 10.210.14.130 > 10.210.14.131: ICMP echo reply, id 40834,
seq 382, length 64
22:11:25.222321 Out IP truncated-ip - 24 bytes missing! 10.210.14.131 >
10.210.14.130: ICMP echo request, id 40834, seq 383, length 64
22:11:25.223346 In IP 10.210.14.130 > 10.210.14.131: ICMP echo reply, id 40834,
seq 383, length 64
22:11:26.228769 Out IP truncated-ip - 24 bytes missing! 10.210.14.131 >
10.210.14.130: ICMP echo request, id 40834, seq 384, length 64
22:11:26.229680 In IP 10.210.14.130 > 10.210.14.131: ICMP echo reply, id 40834,
seq 384, length 64
^C

19 packets received by filter
0 packets dropped by kernel

lab@host1-a>
```

- ◆ What command option allows you to view source and destination MAC addresses for the captured packets?

-
-
- ❖ Include the **layer2-headers** option to view Layer 2 header information, including the source and destination MAC addresses as shown:

```
lab@host1-a> monitor traffic interface ge-0/0/0 matching icmp layer2-headers
verbose output suppressed, use <detail> or <extensive> for full protocol decode
Address resolution is ON. Use <no-resolve> to avoid any reverse lookup delay.
Address resolution timeout is 4s.
Listening on ge-0/0/0, capture size 96 bytes

Reverse lookup for 10.210.14.131 failed (check DNS reachability).
Other reverse lookup failures will not be reported.
Use <no-resolve> to avoid reverse lookups on IP addresses.

22:14:14.306530 Out 0:26:88:e9:d2:80 > 0:c:29:cd:d5:84, ethertype IPv4 (0x0800),
length 74: truncated-ip - 24 bytes missing! 10.210.14.131 > 10.210.14.130: ICMP
echo request, id 40834, seq 551, length 64
22:14:14.307436 In PFE proto 2 (ipv4): 10.210.14.130 > 10.210.14.131: ICMP echo
reply, id 40834, seq 551, length 64
22:14:15.312960 Out 0:26:88:e9:d2:80 > 0:c:29:cd:d5:84, ethertype IPv4 (0x0800),
length 74: truncated-ip - 24 bytes missing! 10.210.14.131 > 10.210.14.130: ICMP
echo request, id 40834, seq 552, length 64
```

Introduction to JUNOS Software & Routing Essentials

```
22:14:15.313826 In PFE proto 2 (ipv4): 10.210.14.130 > 10.210.14.131: ICMP echo
reply, id 40834, seq 552, length 64
22:14:16.319483 Out 0:26:88:e9:d2:80 > 0:c:29:cd:d5:84, ethertype IPv4 (0x0800),
length 74: truncated-ip - 24 bytes missing! 10.210.14.131 > 10.210.14.130: ICMP
echo request, id 40834, seq 553, length 64
22:14:16.320371 In PFE proto 2 (ipv4): 10.210.14.130 > 10.210.14.131: ICMP echo
reply, id 40834, seq 553, length 64
22:14:17.325947 Out 0:26:88:e9:d2:80 > 0:c:29:cd:d5:84, ethertype IPv4 (0x0800),
length 74: truncated-ip - 24 bytes missing! 10.210.14.131 > 10.210.14.130: ICMP
echo request, id 40834, seq 554, length 64
22:14:17.326894 In PFE proto 2 (ipv4): 10.210.14.130 > 10.210.14.131: ICMP echo
reply, id 40834, seq 554, length 64
22:14:18.332435 Out 0:26:88:e9:d2:80 > 0:c:29:cd:d5:84, ethertype IPv4 (0x0800),
length 74: truncated-ip - 24 bytes missing! 10.210.14.131 > 10.210.14.130: ICMP
echo request, id 40834, seq 555, length 64
22:14:18.333326 In PFE proto 2 (ipv4): 10.210.14.130 > 10.210.14.131: ICMP echo
reply, id 40834, seq 555, length 64
^C

28 packets received by filter
0 packets dropped by kernel

lab@host1-a>
```

Note

The **monitor traffic** command only captures packets that are local to the device. It does not capture transit packets.

Step 3.4

Stop both the ping and monitor operations using the Ctrl+c keystroke combination



You have completed Lab 2 !