# PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ

## FACULTAD DE CIENCIAS E INGENIERÍA

# EXTENSIÓN DE UN GESTOR DE REDES BASADO EN SOFTWARE LIBRE PARA UN OPERADOR MÓVIL

**ANEXO A**

**Tesis para optar el Título de Ingeniero de las Telecomunicaciones, que presenta el bachiller:**

**HERNÁN ROMANO CURO**

**ASESOR: MG. ANTONIO OCAMPO ZÚÑIGA**

**Lima, julio de 2013**

# *Anexo A: Monitoreo de usuarios telnet conectados*

En el presente anexo se realizarán los pasos seguidos para monitorear la cantidad de sesiones telnet activas en un router cisco siguiendo el diseño planteado.

## A.1 Descripción del sistema

La implementación de esta prueba local se realizó con ayuda del software GNS3, el cual se encuentra bajo una licencia libre y permite la simulación de entornos de red; además, en el gestor se usó el sistema operativo Linux en su distribución Ubuntu versión 12.04 en el cual se instaló el paquete de herramientas NET-SNMP alojado en una máquina virtual, Oracle VM VirtualBox.

El demonio snmpd se encuentra implementado en el NMS; el cual se encargará de ejecutar un script que permitirá extraer la información del router utilizando el protocolo telnet. De esta manera, hacemos el proceso transparente para la plataforma cacti.

## A.2 Configuración del sistema

La tarea principal de esta implementación es demostrar el diseño propuesto en la presente tesis para la extracción de parámetros de red de distintos equipos que soporten acceso remoto vía telnet o SSH, sin importar la marca del fabricante del equipo.

Para ello, relacionamos los OID's de los objetos a monitorear con el script correspondiente dentro del archivo de configuración snmpd.conf. (/etc/snmp/snmpd.conf)

**Tabla A.1: Ejemplo de modificación en el archivo de configuración snmpd**

| |
|---|
| pass .1.3.6.1.4.1.7000.1.1.0 /home/kuro/Desktop/scripts/bashs/sesionesTelnet.sh |

El script realiza la tarea de extraer la información, conectándose vía telnet con la ayuda de la herramienta expect del sistema Ubuntu, ejecuta el comando *show line* propio del dispositivo monitoreadoy devuelve la cantidad de sesiones activas telnet del router, finalmente el script se encarga de filtrar la información obtenida y devolver la información

de forma ordenada junto con el OID y el tipo de dato obtenido. Cabe recalcar que no se toma en cuenta la sesión telnet realizada por el mismo NMS hacia el router.

**Tabla A.2: Script para obtener el número de sesiones telnet activas**

```
#!/bin/bash -f

PLACE=".1.3.6.1.4.1.7000.1.1.0"

rm -f /home/kuro/Desktop/scripts/expects/sesionesTelnet.log
touch /home/kuro/Desktop/scripts/expects/sesionesTelnet.log

/home/kuro/Desktop/scripts/expects/sesionesTelnet.exp

value=$(grep "*"
/home/kuro/Desktop/scripts/expects/logeo_tes1.log | grep "VTY"
-c)

echo .1.3.6.1.4.1.7000.1.1.0

echo "Integer32"
echo $(($value-1))
```

**Tabla A.3: Script que realiza el acceso remoto vía telnet**

```
#!/usr/bin/expect -f

log_user 0
log_file -a
/home/kuro/Desktop/scripts/expects/sesionesTelnet.log
set timeout -1
set ip "192.168.0.19"
set pass "admin"

spawn telnet $ip

expect "*?assword:*"
send "$pass\r"

expect "*?R1?*"
send "enable\r"

expect "*?assword:*"
send "$pass\r"

expect "*?R1#*"
send "show line\r"
```

```
expect "*?R1#*"
send "exit\r"

expect eof
```

Luego, se utiliza el commando snmpget en dirección al localhost; en el cual se encuentra el demonio snmpd; el cual finalmente devolverá la cantidad de sesiones telnet activas en el router en tiempo real como lo muestra la figura A.1.

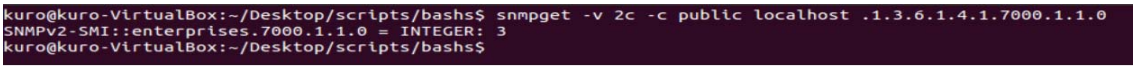


**Figura A.1: Verificación de las sesiones telnet activas mediante el comando snmpget**

# PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ

## FACULTAD DE CIENCIAS E INGENIERÍA

# EXTENSIÓN DE UN GESTOR DE REDES BASADO EN SOFTWARE LIBRE PARA UN OPERADOR MÓVIL

**ANEXO B**

**Tesis para optar el Título de Ingeniero de las Telecomunicaciones, que presenta el bachiller:**

**HERNÁN ROMANO CURO**

**ASESOR: MG. ANTONIO OCAMPO ZÚÑIGA**

**Lima, julio de 2013**

# *Anexo B: Monitoreo de procesos en un Router*

En el presente anexo se realizarán los pasos seguidos para monitorear los procesos en un router cisco siguiendo el diseño planteado.

## B.1 Descripción del sistema

La implementación de esta prueba local se realizó con ayuda del software GNS3, el cual se encuentra bajo una licencia libre y permite la simulación de entornos de red; además, en el gestor se usó el sistema operativo Linux en su distribución Ubuntu versión 12.04 en el cual se instaló el paquete de herramientas NET-SNMP alojado en una máquina virtual, Oracle VM VirtualBox.

El demonio snmpd se encuentra implementado en el NMS; el cual se encargará de ejecutar scripts que permitirán extraer la información del router utilizando el protocolo telnet. De esta manera, hacemos el proceso transparente para la plataforma cacti.

## B.2 Configuración del sistema

La tarea principal de esta implementación es demostrar el diseño propuesto en la presente tesis para la extracción de parámetros de red de distintos equipos que soporten acceso remoto vía telnet o SSH, sin importar la marca del fabricante del equipo.

Para ello, relacionamos los OID's de los objetos a monitorear con el script correspondiente dentro del archivo de configuración snmpd.conf. (/etc/snmp/snmpd.conf)

**Tabla B.1: Ejemplo de modificación en el archivo de configuración snmpd**

| |
|---|
| pass .1.3.6.1.4.1.8072.2.7000.1 /home/kuro/Desktop/scripts/bashs/procesos.sh |

Los scripts realizan la tarea de extraer la información, conectándose vía telnet con la ayuda de la herramienta expect del sistema Ubuntu, ejecuta el comando *show processes cpu sorted 5min* propio del dispositivo monitoreado y devuelve la cantidad

de sesiones activas telnet del router, finalmente el script se encarga de filtrar la información obtenida y devolver la información de forma ordenada junto con el OID y el tipo de dato obtenido.

**Tabla B.2: Script para la extracción del ID, nombre y valor del proceso**

```
#!/bin/sh -f

PLACE1=".1.3.6.1.4.1.8072.2.7000.1.1"  # OID Raiz de ID
PLACE2=".1.3.6.1.4.1.8072.2.7000.1.2"  # OID Raíz Nombre
PLACE3=".1.3.6.1.4.1.8072.2.7000.1.3"  # OID Raíz Valor
REQ="$2"                               # OID Solicitado

/home/kuro/Desktop/scripts/expects/procesos.exp

grep '^ [1-9]\|^  [1-9]\|^   [1-9]'
/home/kuro/Desktop/scripts/expects/logeo_procesos.log | awk '{
s= ""; for (i = 9; i < NF; i++) s = s " " $i; print $1, $7, s}'
>>
/home/kuro/Desktop/scripts/expects/logeo_procesos_limpios.log

numeroDeProcesos=$(awk 'max=="" || $1 > max {max=$1} END{ print
max}' FS=" "
/home/kuro/Desktop/scripts/expects/logeo_procesos_limpios.log)

#
#  Solicitudes GETNEXT - Aquí se valida las siguientes
instancias
#
if [ "$1" = "-n" ]; then

      for i in `seq 1 $numeroDeProcesos`;
         do
      aux_2=$(($i-1))
      if [ $REQ = $PLACE1 ] || [ $REQ = $PLACE1.$aux_2 ]; then
      case "$REQ" in
      $PLACE1.*|      \
      $PLACE1)    RET=$PLACE1.$i ;
      REQ=$RET;
           echo "$RET"
           echo "integer";
           cat
/home/kuro/Desktop/scripts/expects/logeo_procesos_limpios.log |
awk '$1 ~ /^'$i'$/'        | awk '{ print $'1'}';;
      *) echo "string";     echo "ack... $RET $REQ"
      esac
      fi

     if [ $REQ = $PLACE2 ] || [ $REQ = $PLACE2.$aux_2 ]; then
```

```
      case "$REQ" in
      $PLACE2.*|      \
      $PLACE2)    RET=$PLACE2.$i ;
      REQ=$RET;
            echo "$RET"
            echo "string";
            cat
/home/kuro/Desktop/scripts/expects/logeo_procesos_limpios.log |
awk '$1 ~ /^'$i'$/'        | awk '{ s= ""; for (z = 3; z <= NF;
z++) s = s " " $z; print s}';;
      *) echo "string";     echo "ack... $RET $REQ"
      esac
      fi

     if [ $REQ = $PLACE3 ] || [ $REQ = $PLACE3.$aux_2 ]; then
      case "$REQ" in
      $PLACE3.*|      \
      $PLACE3)    RET=$PLACE3.$i ;
      REQ=$RET;
            echo "$RET"
            echo "integer";
            cat
/home/kuro/Desktop/scripts/expects/logeo_procesos_limpios.log |
awk '$1 ~ /^'$i'$/'        | awk '{sub(/%/, "", $2); print
$2*100}';;
      *) echo "string";     echo "ack... $RET $REQ"
      esac

      fi

      done
else
#
#  Solicitudes GET - Aquí se valida las siguientes instancias
#
      for i in `seq 1 $numeroDeProcesos`;
         do
      if [ $REQ = $PLACE1.$i.0 ]; then
      RET=$REQ;
            echo "$RET";
            echo "integer";
            cat
/home/kuro/Desktop/scripts/expects/logeo_procesos_limpios.log |
awk '$1 ~ /^'$i'$/'        | awk '{ print $'1'}'
      fi

     if [ $REQ = $PLACE2.$i.0 ]; then
      RET=$REQ;
            echo "$RET";
            echo "string";
            cat
/home/kuro/Desktop/scripts/expects/logeo_procesos_limpios.log |
```

```
awk '$1 ~ /^'$i'$/' |
     awk '{ s= ""; for (z = 3; z <= NF; z++) s = s " " $z;
print s}'
     fi

     if [ $REQ = $PLACE3.$i ]; then
     RET=$REQ;
          echo "$RET";
          echo "integer";
          cat
/home/kuro/Desktop/scripts/expects/logeo_procesos_limpios.log |
awk '$1 ~ /^'$i'$/'        | awk '{sub(/%/, "", $2); print
$2*100}'
     fi

     done
fi
```

**TABLA B.3: Script que realiza el acceso remoto vía telnet**

```
#!/usr/bin/expect -f

log_user 0
log_file -a
/home/kuro/Desktop/scripts/expects/logeo_procesos.log
set timeout -1
set ip "192.168.0.19"
set pass "admin"

spawn telnet $ip

expect "*?assword:*"
send "$pass\r"

expect "*?R1?*"
send "enable\r"

expect "*?assword:*"
send "$pass\r"

expect "*?R1#*"
send "terminal length 0\r"

expect "*?R1#*"
send "show processes cpu sorted 5min\r"

expect "*?R1#*"
send "exit\r"
```

```
expect eof
```

Luego, se utilizará el comando snmpwalk en dirección al localhost; en el cual se encuentra el demonio snmpd; y finalmente devolverá los procesos en el router en tiempo real como lo muestra la figura B.1.

```
NET-SNMP-EXAMPLES-MIB::netSnmpExamples.7000.1.2.103 = INTEGER: 0
NET-SNMP-EXAMPLES-MIB::netSnmpExamples.7000.1.2.104 = INTEGER: 0
NET-SNMP-EXAMPLES-MIB::netSnmpExamples.7000.1.2.105 = INTEGER: 0
NET-SNMP-EXAMPLES-MIB::netSnmpExamples.7000.1.2.106 = INTEGER: 0
NET-SNMP-EXAMPLES-MIB::netSnmpExamples.7000.1.2.107 = INTEGER: 0
NET-SNMP-EXAMPLES-MIB::netSnmpExamples.7000.1.2.108 = INTEGER: 0
NET-SNMP-EXAMPLES-MIB::netSnmpExamples.7000.1.3.1 = STRING: " Chunk Manager"
NET-SNMP-EXAMPLES-MIB::netSnmpExamples.7000.1.3.2 = STRING: " Load Meter"
NET-SNMP-EXAMPLES-MIB::netSnmpExamples.7000.1.3.3 = STRING: " CEF Scanner"
NET-SNMP-EXAMPLES-MIB::netSnmpExamples.7000.1.3.4 = STRING: " Check heaps"
NET-SNMP-EXAMPLES-MIB::netSnmpExamples.7000.1.3.5 = STRING: " Pool Manager"
NET-SNMP-EXAMPLES-MIB::netSnmpExamples.7000.1.3.6 = STRING: " Timers"
NET-SNMP-EXAMPLES-MIB::netSnmpExamples.7000.1.3.7 = STRING: " Serial Backgroun"
NET-SNMP-EXAMPLES-MIB::netSnmpExamples.7000.1.3.8 = STRING: " Environmental mo"
NET-SNMP-EXAMPLES-MIB::netSnmpExamples.7000.1.3.9 = STRING: " HC Counter Timer"
NET-SNMP-EXAMPLES-MIB::netSnmpExamples.7000.1.3.10 = STRING: " ATM Idle Timer"
```

**Figura B.1: Resultado del comando Snmpwalk**