Routers Industriales VITRIKO 2G/3G/4G SCRIPTS DE PROGRAMACIÓN

ÍNDICE

Contenido

1. Email según estado I/O Digital

- Objetivo

El siguiente script nos envía un email cuando las entradas de la interfaz ***I/O digital*** cambian de estado

- Estructura

```
EMAIL=ejemplo@VITRIKO.com
MESSAGE="BIN0 is active"
while true
do
 /usr/bin/io get bin0
 VAL=$?
 if [ "$VAL" != "$OLD" ]; then
  [ "$VAL" = "0" ] && /usr/bin/email -t $EMAIL -s "$MESSAGE"
  OLD=$VAL
 fi
 sleep 1
done
```

2. Backup Routes selected vía SMS

- Objetivo

Este script nos envía un SMS al número de teléfono especificado, cuando nuestra interfaz seleccionada como primaria en el apartado ***Backup Routes***, falle y otra interfaz pase a tener el rol de primaria

- Estructura

```
#!/bin/sh

old=0

mobile="your_mobile_number"

while true

do

sel=`ls -al  /var/backrt/ | awk '/bar-selected.inf ->/ { print  $NF }' | sed -n 's/.*bar-//;s/.inf.*//p'`

if [ $sel ]; then

  if [ "$sel" != "$old" ]; then

    old=$sel

    case $sel in

      mwan*) gsmsms $mobile "Mobil connection"

      ;;

      wifi*) gsmsms $mobile "WiFi connection"

      ;;

      eth1*) gsmsms $mobile "Secondary LAN connection"

      ;;

      pppo*) gsmsms $mobile "PPPoE connection"

      ;;

      eth0*) gsmsms $mobile "Primary LAN connection"

    esac
```

```
  fi
fi
sleep 10
done
```

3. VLAN (Virtual Local Area Network)

- Objetivo

El siguiente script nos permite la creación y edición de VLANs

- Estructura

1. Creamos la VLAN. Utilizaremos la interfaz ***ETH1.*** *Sin embargo, primero es necesario activar la interfaz sin una IP asignada, antes de que la interfaz pueda ser utilizada*

```
ifconfig eth1 0.0.0.0 up
```

2. Ahora pueden ser creadas las interfaces vlan 11 y vlan 12 en la interfaz ***ETH1.*** *Para crearlas utilizamos el comando* ***vconfig add***

```
vconfig add eth1 11
vconfig add eth1 12
```

3. *Podemos ver las interfaces VLAN utilizando el comando* ***ifconfig –a***

```
ifconfig -a
eth1.11 Link encap:Ethernet HWaddr 00:30:48:BF:4E:BD
BROADCAST MULTICAST MTU:1500 Metric:1
RX packets:0 errors:0 dropped:0 overruns:0 frame:0
TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:0
RX bytes:0 (0.0 b) TX bytes:0 (0.0 b)

eth1.12 Link encap:Ethernet HWaddr 00:30:48:BF:4E:BD
BROADCAST MULTICAST MTU:1500 Metric:1
RX packets:0 errors:0 dropped:0 overruns:0 frame:0
TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:0
RX bytes:0 (0.0 b) TX bytes:0 (0.0 b)
```

4. Asignamos las direcciones IP a las interfaces VLAN, utilizando el comando ***ifconfig***

```
ifconfig eth1.11 192.168.11.254 netmask 255.255.255.0 up
ifconfig eth1.12 192.168.12.254 netmask 255.255.255.0 up
```

5. Observamos que las direcciones IP han sido asignadas correctamente a las interfaces VLAN.

```
ifconfig eth1.11
eth1.11 Link encap:Ethernet HWaddr 00:30:48:BF:4E:BD
inet addr:192.168.11.254 Bcast:192.168.11.255 Mask:255.255.255.0
UP BROADCAST MULTICAST MTU:1500 Metric:1
RX packets:0 errors:0 dropped:0 overruns:0 frame:0
```

TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:0
RX bytes:0 (0.0 b) TX bytes:0 (0.0 b)

ifconfig eth1.12
eth1.12 Link encap:Ethernet HWaddr 00:30:48:BF:4E:BD
inet addr:192.168.12.254 Bcast:192.168.12.255 Mask:255.255.255.0
UP BROADCAST MULTICAST MTU:1500 Metric:1
RX packets:0 errors:0 dropped:0 overruns:0 frame:0
TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:0
RX bytes:0 (0.0 b) TX bytes:0 (0.0 b)

4. Reboot debido al fallo de un proceso

- Objetivo

El siguiente script lleva a cabo un ***reboot*** del router cuando alguno de los procesos que estén corriendo en el router falle

- Estructura

```
#!/bin/sh

SERVICE= vpnc

if ps ax | grep -v grep | grep $SERVICE > /dev/null

then

reboot
```

5. Reenvío de SMS recibidos por router a un móvil

- Objetivo

El siguiente script nos permite reenviar los SMS que hemos recibido en un router, a un móvil, para que podamos visualizarlos en el caso de no tener acceso al router

- Estructura

```
STARTUP=#!/bin/sh
STARTUP=#
STARTUP=# This script will be executed *after* all the other init scripts.
STARTUP=# You can put your own initialization stuff in here.
STARTUP=
STARTUP=# this is callback is called when sms is received
STARTUP=
STARTUP=PHONE1=+420123456789
STARTUP=
STARTUP=cat > /var/scripts/sms << EOF
STARTUP=#!/bin/sh
STARTUP=if [ "\$1" = "1" ] || [ "\$2" = "$PHONE1" ]; then
STARTUP=  logger "Received sms " \$3 " from " \$2
STARTUP=fi
STARTUP=EOF
STARTUP=
STARTUP=chmod +x /var/scripts/sms
STARTUP=
STARTUP=# this is reading daemon
STARTUP=
STARTUP=mkdir -p /var/smsrd/sms
STARTUP=
STARTUP=cat > /var/smsrd/smsrd << EOF
STARTUP=#!/bin/sh
STARTUP=while true
STARTUP=do
STARTUP=   gsmat "at+cmgf=1;+cmgl" | awk 'BEGIN { FOUND=0; FS="[:,]" }
FOUND==1 { gsub(/\r/, "", \$0); print PHONE" \""\$0"\"" > FILE; FOUND=0 }
/\+CMGL/ { gsub(/ /, "", \$2); FILE="/var/smsrd/sms/sms_"\$2; gsub(/\"/, "", \$4);
PHONE=\$4; FOUND=1 }'
STARTUP=
STARTUP=  for file in \$(ls /var/smsrd/sms); do
STARTUP=    SMS_FILE="/var/smsrd/sms/"\$file
STARTUP=
STARTUP=    read PHONE TEXT < \$SMS_FILE
STARTUP=
STARTUP=    /var/scripts/sms "0" \$PHONE "\$TEXT"
STARTUP=
STARTUP=    SMS_ID=\$(echo \$file | awk 'BEGIN {FS="_"} {print \$2}')
```

```
STARTUP=
STARTUP=    gsmat "at+cmgd=\$SMS_ID" > /dev/null
STARTUP=
STARTUP=    rm \$SMS_FILE
STARTUP=  done
STARTUP=
STARTUP=  sleep 5
STARTUP=done
STARTUP=EOF
STARTUP=
STARTUP=chmod +x /var/smsrd/smsrd
STARTUP=
STARTUP=/var/smsrd/smsrd &
```

6. Envío de la IP vía SMS, cuando la conexión PPP/WAN esté establecida

- Objetivo

El siguiente script nos envía por SMS la dirección IP así como el nombre del dispositivo, cuando la conexión PPP/WAN se haya establecido.

- Estructura

```
#!/bin/sh
#
# This script will be executed when PPP/WAN connection is established.
IP_PPP=$(cat /var/ppp/ip)
NAME="UR5v2" #name of router must be without space
TXT=$NAME"="$IP_PPP
gsmsms +420724342275 $TXT
exit 0
```

7. Envío de los SMS recibidos a un servidor FTP

- Objetivo

El siguiente script nos permite enviar los SMS recibidos de unos números de teléfono específicos, a un servidor FTP.

- Estructura

```
#!/bin/sh
#
# This script will be executed *after* all the other init scripts.
# You can put your own initialization stuff in here.

PHONE1=+420123456789
PHONE2=+420987654321
PHONE3=+420724342275
PHONE4=+420112233445
PHONE5=+420334455667

cat > /var/scripts/sms << EOF
#!/bin/sh
if [ "\$1" = "1" ] || [ "\$2" = "$PHONE1" ] || [ "\$2" = "$PHONE2" ] || [ "\$2" = "$PHONE3" ] || [ "\$2" = "$PHONE4" ] || [ "\$2" = "$PHONE5" ]; then
  text="Received SMS from \$2 Text: \$3 \$4 \$5 \$6 \$7 \$8"
  DATE=\`date +%Y-%m-%d_%H:%M:%S\`
  echo \$text > /root/textSMS_\$DATE.txt
  ftpput -u root -p root 192.168.1.8 /home/wasp/textSMS_\$DATE.txt /root/textSMS_\$DATE.txt
rm /root/textSMS_\$DATE.txt
fi
EOF
```

8. Eliminar todos los SMS almacenados

- Objetivo

El siguiente script nos permite eliminar todos los SMS almacenados en el router

- Estructura

```
#!/bin/sh
gsmat at+cmgf=1 > /dev/null
num_sms=`gsmat "at+cmgl=\"ALL\"" | awk '/^\+CMGL/ { n++}; END {print n+0}'`
echo "Number of SMS is "$num_sms"."
pruch=0
if [ $num_sms -ne 0 ]; then
while [ $num_sms -gt 0 ]
do
  ind=`gsmat "at+cmgl=\"ALL\"" | awk '/^\+CMGL/ { print $2 }' | awk -F ,
NR==$num_sms'{ print $1 }'`
  echo "Deleting "$ind". SMS...."
  gsmat at+cmgd=$ind
  pruch=$(($pruch+1))
  num_sms=$(($num_sms-1))
done
echo "Deleted "$pruch" SMS."
else echo "No SMS was deleted."
fi
exit 0
```

9. Envío a través de SMS del estado de las interfaces binarias I/0 digital

- Objetivo

El siguiente script permite conocer el estado de cada una de las interfaces de la I/O digital, y enviar el estado de cada una de ellas vía SMS.

- Estructura

```
while true
do
io get bin4
BIN4=$?
if [ "$BIN4" != "$OLD4" ]; then
if [ "$BIN4" = "1" ]; then
gsmsms 123456789 "test SMS bin4-1"
OLD4=$BIN4
fi
if [ "$BIN4" = "0" ]; then
gsmsms 123456789 "test SMS bin4-0"
OLD4=$BIN4
fi
fi
sleep 2
io get bin3
BIN3=$?
if [ "$BIN3" != "$OLD3" ]; then
if [ "$BIN3" = "1" ]; then
gsmsms 123456789 "test SMS bin3-1"
OLD3=$BIN3
fi
if [ "$BIN3" = "0" ]; then
gsmsms 123456789 "test SMS bin3-0"
OLD3=$BIN3
fi
fi
sleep 2
done
```

10. Envío del nivel de voltaje a través de SMS

- Objetivo

Este script nos envía mediante un SMS, el nivel de voltaje de la entrada y salida digital en un instante determinado.

- Estructura

```
#!/bin/sh
mkdir -p /var/voltaged
Umin=16.3
Num=603231977
cat > /var/voltaged/voltaged << EOF
#!/bin/sh
old=0
while true
do
Uget=\`status sys | awk '/^Supply Voltage/ { print \$4 }'\`
if [ \$(echo "\$Uget $Umin" | awk '{print (\$1 < \$2)}') == 1 ]; then
  if [ \$old -eq 0 ]; then
    led on
    gsmsms $Num "Power is low!!! Power is only \$Uget V"
    old=1
  fi
else
  led off
  old=0
fi
sleep 5
done
EOF

chmod +x /var/voltaged/voltaged
/var/voltaged/voltaged &
```

11. REGLAS NAT CON IP TABLES

- Objetivo

Este script nos muestra cómo “hacer visible” mediante iptables, un servicio que se encuentra en una red privada, para los usuarios exteriores (utilizando DNAT – Destination NAT)

- Estructura

```
#!/bin/sh

#

# This script will be executed *after* all the other init scripts.

# You can put your own initialization stuff in here.


iptables -t nat -A PREROUTING -p tcp --dport 8081 -j DNAT --to
192.168.1.200:80

iptables -t nat -A PREROUTING -p tcp --dport 8082 -j DNAT --to
192.168.1.201:80
```

12. VALOR ACTUAL DEL PUERTO DE EXPANSIÓN CNT

- Objetivo

El siguiente script solicita al sistema operativo el valor actual del puerto de expansión CNT, y lo muestra aplicando el redondeo correspondiente.

- Estructura

```
#!/bin/sh
while true
do
  get_io=`io get an3`
  value=$((($get_io-38)*624)/100000))
  dec=$((($get_io-38)*624)%100000))
  valdec=$(($dec/10000))
  valcent=$((($dec%10000)/1000))

# rounding for one decimal place
if [ $valcent -ge 5 ]; then
  if [ $valdec -eq 9 ]; then
    value=$(($value+1))
  fi
  valdec=$(($valdec+1))
  valdec=$(($valdec%10))
fi

# print value
if [ $valdec -eq 0 ]; then
  echo "$value mA"
else
  echo "$value,$valdec mA"
fi
  sleep 1
done
```

13. SWITCHING ENTRE SIM CARD 1 Y SIM CARD 2

- Objetivo

El siguiente script hace un cambio de la ***SIM Card 1*** a la ***SIM Card 2*** cuando el estado de la ***I/O digital*** cambia de 1 a 0.

- Estructura

```
# Default is when input port is 1. It is primary SIM.
# Secondar SIM is chosen when input port is 0.
/sbin/service ppp stop
/usr/bin/io get bin0
VAL=$?
if [ "$VAL" = "0" ]; then
        /bin/sed -e "s/\(PPP_DEFAULT_SIM=\).*/\12/" -i /etc/settings.ppp
        /bin/sed -e "s/\(PPP_BACKUP_SIM=\).*/\11/" -i /etc/settings.ppp
else
        /bin/sed -e "s/\(PPP_DEFAULT_SIM=\).*/\11/" -i /etc/settings.ppp
        /bin/sed -e "s/\(PPP_BACKUP_SIM=\).*/\12/" -i /etc/settings.ppp
fi
LAST=$VAL
/sbin/service ppp start
sleep 20
while true
do
  /usr/bin/io get bin0
  VAL=$?

  if [ $VAL != $LAST ]; then
    LAST=$VAL
      if [ "$VAL" = "0" ]; then
        /sbin/service ppp stop
        /bin/sed -e "s/\(PPP_DEFAULT_SIM=\).*/\12/" -i /etc/settings.ppp
        /bin/sed -e "s/\(PPP_BACKUP_SIM=\).*/\11/" -i /etc/settings.ppp
        /sbin/service ppp start
      fi
      if [ "$VAL" = "1" ]; then
        /sbin/service ppp stop
        /bin/sed -e "s/\(PPP_DEFAULT_SIM=\).*/\11/" -i /etc/settings.ppp
        /bin/sed -e "s/\(PPP_BACKUP_SIM=\).*/\12/" -i /etc/settings.ppp
        /sbin/service ppp start
      fi
    fi
  sleep 1
done
```

14. MONITORIZACIÓN DE UNA INTERFAZ

- Objetivo

El siguiente script realiza un *ping a una dirección IP* de una interfaz ***para conocer el estado de ésta***. ***Si el ping es correcto*** y existe respuesta por parte la interfaz, ***se envía un mensaje a través de SMS***.

- Estructura

```
#!/bin/sh
IP=192.168.1.2   #Monitored IP address
NTP=5     #Number of trasmitted packets
NRP=3     #Number of received packets
TPO=5     #Time of Power off (s)
TAR=20    #Time after reboot (s)
TBP=10    #Time between pings (s)
MNUMBER=123456789  #Mobile number
TEXTSMS="RESTART"  #Text of SMS
while true
do
ping -c $NTP $IP > /root/ping.txt
PNG=`cat /root/ping.txt |awk '/packet/{print $4}'`
if [ $PNG -lt $NRP ]; then
/usr/bin/io set out0 1
gsmsms $MNUMBER $TEXTSMS
sleep $TPO
/usr/bin/io set out0 0
sleep $TAR
fi
/usr/bin/io set out0 0
```

```
sleep $TBP

done
```

15. ESTADO DE IPSEC MEDIANTE I/O DIGITAL.

- Objetivo

El siguiente script nos permite conocer a través del valor de la ***I/0 Digital***, si el túnel IPsec está establecido. Si el túnel está establecido, tendremos una salida **Out = 0** y si el túnel no lo está, el valor **Out = 1.**

- Estructura

```
num=1 # number of IPSec [1,2,3,4]

while true

do

  whack --status | awk "/ipsec$num/ && /STATE_QUICK.*/ && /IPsec SA/" |
grep established

  sts=$?

  if [ "$sts" = "0" ]; then

    io set out0 1

  else

    io set out0 0

  fi

  sleep 5

done
```

16. OPEN VPN – SITE TO SITE BRIDGED VPN ENTRE 2 ROUTERS.

- ### Objetivo

El siguiente script nos permite establecer **un túnel VPN site-to-site** entre 2 routers, teniendo uno de ellos el rol de servidor, y el otro de cliente.

- ### Estructura

Para la creación del túnel VPN:

- **OpenVPN Server:**

1. Creamos un archivo al que llamamos ***server.ovpn*** y lo copiamos en una carpeta que pertenezca al administrador principal ***(folder/root).*** Generamos los certificados y claves (dh1024.pem,ca.crt, server.key, server.crt), y los copiamos en la misma carpeta.

```
port 1194
proto udp
dev tap0
server-bridge 172.16.194.254 255.255.255.0 172.16.194.100
172.16.194.200
comp-lzo
keepalive 10 60
persist-tun
dh /root/dh1024.pem
ca /root/ca.crt
tls-server
key /root/server.key
cert /root/server.crt
verb 3
comp-lzo
```

2. Escribimos el script en la ***Startup Script:***

```
Startup Script:
#!/bin/sh
#
# This script will be executed *after* all the other init scripts.
# You can put your own initialization stuff in here.
. /etc/settings.eth
# Define Bridge Interface
br="br0"
```

```
# Define TAP interfaces to be bridged,
# for example tap="tap0".
tap="tap0"
# Define physical ethernet interface to be bridged
# with TAP interface(s) above.
eth="eth0"
eth_ip=$ETH_IPADDR
eth_netmask=$ETH_NETMASK
killall dhcpd
/sbin/ip link set eth0 down
sleep 5
/sbin/ip link set eth0 up
/usr/sbin/openvpnd --mktun --dev $tap
/usr/sbin/brctl addbr $br
/usr/sbin/brctl addif $br $eth
/usr/sbin/brctl addif $br $tap
/sbin/ifconfig $tap 0.0.0.0 promisc up
/sbin/ifconfig $eth 0.0.0.0 promisc up
/sbin/ifconfig $br $eth_ip netmask $eth_netmask up
```

3. Escribimos los siguientes comandos en ***Up/Down Script***

Up Script:

```
#!/bin/sh
#
# This script will be executed when PPP/WAN connection is
established.
/usr/sbin/openvpnd --syslog --config /root/server.ovpn &
```

Down Script:

```
#!/bin/sh
#
# This script will be executed when PPP/WAN connection is lost.
killall openvpnd
```

4. Hacemos reboot del router.

- **OpenVPN Client:**

1. Creamos un archivo llamado ***client.ovpn*** y lo copiamos en una carpeta que pertenezca al administrador principal ***(folder/root).*** Copiamos todos los certificados y claves (ca.crt, er75i.key, er75i.crt) en la misma carpeta.

The file client.ovpn:

```
remote 10.0.5.160
```

```
port 1194
proto udp
dev tap
comp-lzo
keepalive 10 60
tls-client
persist-tun
ca /root/ca.crt
key /root/er75i.key
cert /root/er75i.crt
verb 3
```

2. Escribimos el script en la ***Startup Script:***

StartUp Script:

```
#!/bin/sh
#
# This script will be executed *after* all the other init scripts.
# You can put your own initialization stuff in here.
. /etc/settings.eth
# Define Bridge Interface
br="br0"
# Define TAP interfaces to be bridged,
# for example tap="tap0".
tap="tap0"
# Define physical ethernet interface to be bridged
# with TAP interface(s) above.
eth="eth0"
eth_ip=$ETH_IPADDR
eth_netmask=$ETH_NETMASK
killall dhcpd
/sbin/ip link set eth0 down
sleep 5
/sbin/ip link set eth0 up
/usr/sbin/openvpnd --mktun --dev $tap
/usr/sbin/brctl addbr $br
/usr/sbin/brctl addif $br $eth
/usr/sbin/brctl addif $br $tap
/sbin/ifconfig $tap 0.0.0.0 promisc up
/sbin/ifconfig $eth 0.0.0.0 promisc up
/sbin/ifconfig $br $eth_ip netmask $eth_netmask up
```

3. Escribimos los siguientes comandos en ***Up/Down Script***

Up Script:

```
#!/bin/sh
```

#
 # This script will be executed when PPP/WAN connection is established.
/usr/sbin/openvpnd --syslog --config /root/client.ovpn &
Down Script:
#!/bin/sh
#
 # This script will be executed when PPP/WAN connection is lost.
killall openvpnd

4. Hacemos ***reboot*** del router.

17. IPROUTES

- Objetivo

El siguiente script nos indica cómo configurar rutas estáticas, mediante el comando ***iproute add***

- Estructura

```
STARTUP=# This script will be executed *after* all the other init scripts.
STARTUP=# You can put your own initialization stuff in here.
STARTUP=
STARTUP=[ -h /usr/sbin/dhcp2d ] || ln -s /usr/sbin/dhcpd /usr/sbin/dhcp2d
STARTUP=
STARTUP=cat > /var/dhcp/dhcpd2.conf << EOF
STARTUP=option routers 192.168.2.1;
STARTUP=option domain-name-servers 192.168.2.1;
STARTUP=default-lease-time 600;
STARTUP=max-lease-time 86400;
STARTUP=subnet 192.168.2.0 netmask 255.255.255.0 { range 192.168.2.2
192.168.2.254; }
STARTUP=EOF
STARTUP=
STARTUP=touch /var/dhcp/dhcpd2.leases
STARTUP=/usr/sbin/dhcp2d -q -cf /var/dhcp/dhcpd2.conf -lf
/var/dhcp/dhcpd2.leases -pf /var/dhcp/dhcp2d.pid eth1 2> /dev/null &
STARTUP=
STARTUP=mkdir -p /etc/iproute2
STARTUP=mkdir /var/iproute2
STARTUP=
STARTUP=cat > /var/iproute2/rt_tables << EOF
STARTUP=# reserved values
STARTUP=#
STARTUP=255    local
STARTUP=254     main
STARTUP=253     default
STARTUP=0       unspec
STARTUP=#
STARTUP=# local
STARTUP=#
STARTUP=#1      inr.ruhep
STARTUP=2 first_module
STARTUP=3 second_module
```

```
STARTUP=EOF
STARTUP=
STARTUP=[ -h /etc/iproute2/rt_tables ] || ln -s /var/iproute2/rt_tables
/etc/iproute2/rt_tables
STARTUP=
STARTUP=iptables -t nat -A POSTROUTING -o usb1 -j MASQUERADE
STARTUP=iptables -t nat -A POSTROUTING -o usb0 -j MASQUERADE
STARTUP=
STARTUP=cat > /var/dev_check << EOF
STARTUP=#!/bin/sh
STARTUP=
STARTUP=. /etc/settings.eth
STARTUP=
STARTUP=killall dnsmasq
STARTUP=
STARTUP=touch /var/resolv/resolv1.conf
STARTUP=touch /var/resolv/resolv2.conf
STARTUP=
STARTUP=/usr/sbin/dnsmasq -u root -h -a \$ETH_IPADDR -z -r
/var/resolv/resolv1.conf &
STARTUP=/usr/sbin/dnsmasq -u root -h -a \$ETH2_IPADDR -z -r
/var/resolv/resolv2.conf &
STARTUP=
STARTUP=OLD_USB0=1
STARTUP=OLD_USB1=1
STARTUP=
STARTUP=while true
STARTUP=do
STARTUP=
STARTUP=ls /var/ppp | grep -q "iface-m0.inf" > /dev/null
STARTUP=VAL=\$?
STARTUP=
STARTUP=if [ "\$VAL" != "\$OLD_USB0" ]; then
STARTUP=  OLD_USB0=\$VAL
STARTUP=  export \`ipcalc -p \$ETH_NETWORK \$ETH_NETMASK\`
STARTUP=  if [ "\$VAL" = "0" ]; then
STARTUP=    logger "usb0 found"
STARTUP=    DNS=\`awk 'NR==4 {print}' /var/ppp/iface-m0.inf\`
STARTUP=    ip route add default dev usb0 table first_module &
STARTUP=    ip rule add from \$ETH_NETWORK/\$PREFIX table
first_module
STARTUP=    ip route add \$ETH_NETWORK/\$PREFIX dev eth0 table
first_module
STARTUP=    ip route add \$ETH2_NETWORK/\$PREFIX dev eth1 table
first_module
STARTUP=    ip route add default via \`ifconfig usb0 | awk '/inet/ {
gsub("addr:",""); print \$2}'\` dev usb0 table first_module
STARTUP=    echo "nameserver \$DNS" > /var/resolv/resolv1.conf
STARTUP=    ip route add \$DNS dev usb0
STARTUP=  else
STARTUP=    logger "usb0 lost"
```

```
STARTUP=    ip route del default dev usb0 table first_module &
STARTUP=    ip rule del from \$ETH_NETWORK/\$PREFIX table
first_module
STARTUP=    ip route del \$ETH_NETWORK/\$PREFIX dev eth0 table
first_module
STARTUP=    ip route del \$ETH2_NETWORK/\$PREFIX dev eth1 table
first_module
STARTUP=  fi
STARTUP=
STARTUP=fi
STARTUP=
STARTUP=ls /var/ppp | grep -q "iface-m1.inf" > /dev/null
STARTUP=VAL=\$?
STARTUP=
STARTUP=if [ "\$VAL" != "\$OLD_USB1" ]; then
STARTUP=  OLD_USB1=\$VAL
STARTUP=  export \`ipcalc -p \$ETH2_NETWORK \$ETH2_NETMASK\`
STARTUP=  if [ "\$VAL" = "0" ]; then
STARTUP=    logger "usb1 found"
STARTUP=    DNS=\`awk 'NR==4 {print}' /var/ppp/iface-m1.inf\`
STARTUP=    ip route add default dev usb1 table second_module &
STARTUP=    ip rule add from \$ETH2_NETWORK/\$PREFIX table
second_module
STARTUP=    ip route add \$ETH2_NETWORK/\$PREFIX dev eth1 table
second_module
STARTUP=    ip route add \$ETH_NETWORK/\$PREFIX dev eth0 table
second_module
STARTUP=    ip route add default via \`ifconfig usb1 | awk '/inet/ {
gsub("addr:",""); print \$2}'\` dev usb1 table second_module
STARTUP=    echo "nameserver \$DNS"  > /var/resolv/resolv2.conf
STARTUP=    ip route add \$DNS dev usb1
STARTUP=    iptables -t mangle -D block -j ACCEPT > /dev/null
STARTUP=    iptables -t mangle -I block -j ACCEPT > /dev/null
STARTUP=  else
STARTUP=    logger "usb1 lost"
STARTUP=    ip route del default dev usb1 table second_module &
STARTUP=    ip rule del from \$ETH2_NETWORK/\$PREFIX table
second_module
STARTUP=    ip route del \$ETH2_NETWORK/\$PREFIX dev eth1 table
second_module
STARTUP=    ip route del \$ETH_NETWORK/\$PREFIX dev eth0 table
second_module
STARTUP=  fi
STARTUP=fi
STARTUP=
STARTUP=sleep 1
STARTUP=done
STARTUP=
STARTUP=EOF
STARTUP=
STARTUP=chmod +x /var/dev_check
```

```
STARTUP=
STARTUP=/var/dev_check &
IP_UP=#!/bin/sh
IP_UP=#
IP_UP=# This script will be executed when PPP/WAN connection is
established.
IP_UP=
IP_UP=route del default gw 192.168.254.254
IP_DOWN=#!/bin/sh
IP_DOWN=#
IP_DOWN=# This script will be executed when PPP/WAN connection is
lost.
```

18. ENVÍO DE UN ARCHIVO DE LOG A UNA SERIE DE NÚMEROS DE TELÉFONO VÍA SMS.

- Objetivo

El siguiente script nos permite, dados unos números de teléfono (indicados en el archivo ***list2.txt)***, enviar un archivo de logging a un teléfono móvil vía SMS.

- Estructura:

Parte 1:

```
#!/bin/sh
. /root/list2.txt
num=`cat /root/list2.txt`
echo $num
i=1
while [ $i -le 10 ];
do
  echo "Telephone number $i is "$((phone$i))
  i=$(($i+1));
done
```

```
#!/bin/sh
if [ -e /root/log.txt ]; then
  echo -e "Removing log file ....\n"
  rm -f /root/log.txt
else
  echo -e "Nothing for removing.\n"
fi
for i in `cat /root/list.txt`
do
  echo -e "Sending SMS to number $i"
  sms $i "Hello world"
  err=$?
  d=`date`
  if [ "$err" = "0" ]; then
    echo -e "O.K: SMS was sent to number $i\n"
    echo "$d O.K: SMS was sent to number $i" >> /root/log.txt
  else
```

```
    echo -e "ERROR: SMS wasn't sent to number $i!!!!\n"
    echo "$d ERROR: SMS wasn't sent to number $i!!!!" >> /root/log.txt
  fi
done
```

19. AUTOMOUNT

- ### Objetivo

Este script nos indica los comandos (Linux) necesarios, para montar las unidades extraíbles que conectemos al router y que el S.O. del router las reconozca

- ### Estructura

Parte 1:

```
#!/bin/sh
#
LAST=0
i=0
while true
do
flsh=`cat /proc/diskstats |awk '/8\x20\x20\x20\x201/ {print $3}'`
if [ $flsh ]; then
  i=1
else
  i=0
fi
if [ $LAST != $i ]; then
  LAST=$i
  if [ $i = 1 ]; then
    echo "Mount flash disk."
      if [ -d /mnt/flash ]; then
        mount /dev/$flsh /mnt/flash
```

```
      else
        mkdir /mnt/flash
        mount /dev/$flsh /mnt/flash
      fi
  else
    echo "UMOUNT flash disk."
    umount /mnt/flash
    rmdir /mnt/flash
  fi
fi
sleep 2
done
```

Parte 2:

```
#!/bin/sh
#
LAST=0
i=0
while true
do
flsh=`cat /proc/diskstats |awk '/8\x20\x20\x20\x20\x20\x20\x201/ {print $3}'`
if [ $flsh ]; then
  i=1
else
  i=0
fi
if [ $LAST != $i ]; then
```

```
 LAST=$i
 if [ $i = 1 ]; then
  echo "Mount flash disk."
   if [ -d /mnt/flash ]; then
    mount /dev/$flsh /mnt/flash
   else
    mkdir /mnt/flash
    mount /dev/$flsh /mnt/flash
   fi
 else
  echo "UMOUNT flash disk."
  umount /mnt/flash
  rmdir /mnt/flash
 fi
fi
sleep 2
done
```

Parte 3:

```
#!/bin/sh
#
LAST=0
i=0
while true
do
```

```
flsh=`cat /proc/diskstats |awk '/179\x20\x20\x20\x20\x20\x20\x201/ {print $3}'`
if [ $flsh ]; then
  i=1
else
  i=0
fi
if [ $LAST != $i ]; then
  LAST=$i
  if [ $i = 1 ]; then
    echo "Mount flash disk."
      if [ -d /mnt/flash ]; then
        mount /dev/$flsh /mnt/flash
      else
        mkdir /mnt/flash
        mount /dev/$flsh /mnt/flash
      fi
  else
    echo "UMOUNT flash disk."
    umount /mnt/flash
    rmdir /mnt/flash
  fi
fi
sleep 2
done
```

20. CORRIENTE DEL ROUTER

- ### Objetivo

  El siguiente script nos muestra la corriente que circula por el router.

- ### Estructura

```
#!/bin/sh
while true
do
  get_io=`io get an3`
  value=$((($get_io-38)*624)/100000))
  dec=$((($get_io-38)*624)%100000))
  valdec=$(($dec/10000))
  valcent=$((($dec%10000)/1000))

# rounding for one decimal place
if [ $valcent -ge 5 ]; then
  if [ $valdec -eq 9 ]; then
    value=$(($value+1))
  fi
  valdec=$(($valdec+1))
  valdec=$(($valdec%10))
fi

# print value
if [ $valdec -eq 0 ]; then
  echo "$value mA"
else
  echo "$value,$valdec mA"
fi
  sleep 1
done
```

21. VOLTAJE DEL ROUTER

- ### Objetivo

El siguiente script nos muestra el voltaje del router

- ### Estructura

```
#!/bin/sh
mkdir -p /var/voltaged
Umin=16.3
Num=603231977
cat > /var/voltaged/voltaged << EOF
#!/bin/sh
old=0
while true
do
Uget=\`status sys | awk '/^Supply Voltage/ { print \$4 }'\`
if [ \$(echo "\$Uget $Umin" | awk '{print (\$1 < \$2)}') == 1 ]; then
  if [ \$old -eq 0 ]; then
    led on
    gsmsms $Num "Power is low!!! Power is only \$Uget V"
    old=1
  fi
else
  led off
  old=0
fi
sleep 5
done
EOF

chmod +x /var/voltaged/voltaged
/var/voltaged/voltaged &
```

22. ACTIVACIÓN DE LA SALIDA DIGITAL A TRAVÉS DE TELÉFONO MÓVIL

- Objetivo

Este script nos permite activar vía teléfono móvil, la salida digital durante X segundos. (En este caso en particular, 5 segundos).

- Estructura

```
PHONE=+34600600600
cat > /var/scripts/sms << EOF
#!/bin/sh
if [ "\$1" = "1" ] || [ "\$2" = "$PHONE" ]; then
  if [ "\$3" = "IMPULSE" ]; then
    /usr/bin/io set out1 1
    sleep 5
    /usr/bin/io set out1 0
  fi
fi
EOF
```

23. GUARDAR ESTADO I/0 DIGITAL

- ### Objetivo

  Este script almacena el estado de ***"out 0"*** y lo restaura después de hacer reboot al router.

- ### Estructura

Parte 1:

```
#!/bin/sh
#
# This script will be executed *after* all the other init scripts.
# You can put your own initialization stuff in here.
. /var/data/settings.io

if [ $LAST_IO_STATE = 1 ]; then
  /usr/bin/io set out0 1
else
 /usr/bin/io set out0 0
fi
LAST=$LAST_IO_STATE
while true
do
  STATE=`/usr/bin/io get out0`
    if [ $STATE != $LAST ]; then
      if [ $STATE = 0 ]; then
        echo "LAST_IO_STATE=0" > /var/data/settings.io
        LAST=0
      else
        echo "LAST_IO_STATE=1" > /var/data/settings.io
        LAST=1
      fi
    fi
  sleep 1
done
```