

Agente SNMP (sistema que está sendo **monitorado**)

1 - Instale os pacotes do SNMP

```
# dnf install net-snmp net-snmp-utils
```

2 - Edite o arquivo de configuração do SNMP:

```
# vim /etc/snmp/snmpd.conf
```

```
# First, map the community name "public" into a "security name"

#   sec.name source      community
##com2sec notConfigUser default    public
com2sec secTeste 127.0.0.1 mypublic
com2sec secTeste 192.168.4.0/24 mypublic
com2sec secTeste 10.3.1.0/28 mypublic

####
# Second, map the security name into a group name:

#   groupName securityModel securityName
##group notConfigGroup v1      notConfigUser
group groupTeste v2c secTeste

####
# Third, create a view for us to let the group have rights to:

# Make at least snmpwalk -v 1 localhost -c public system fast again.
#   name      incl/excl subtree      mask(optional)
##view systemview included .1.3.6.1.2.1.1
##view systemview included .1.3.6.1.2.1.25.1.1
view viewTeste included .1

####
# Finally, grant the group read-only access to the systemview view.

#   group      context sec.model sec.level prefix read  write notif
##access notConfigGroup "" any noauth exact systemview none none
access groupTeste "" any noauth exact viewTeste none none

# Configura o Trap para enviar para o endereço IP do NMS (gerente)
trap2sink 192.168.4.50 mypublic
```

3 - Reinicie o serviço do SNMP

```
# systemctl stop snmpd  
# systemctl start snmpd
```

4 - Crie os arquivos de monitoramento das interfaces de rede e monitoramento do HD

Arquivo interface-trap.sh (monitorea as interfaces da máquina):

```
#!/bin/bash  
  
COMMUNITY="mypublic"  
TRAP_RECEIVER="192.168.4.50"  
# Arquivo temporário para registrar o status da interface  
STATUS_FILE="/tmp/interface_status.txt"  
  
# Obtém todas as interfaces ativas (excluindo 'lo')  
INTERFACES=$(snmpwalk -v2c -c $COMMUNITY localhost .1.3.6.1.2.1.2.2.1.2 |  
awk -F': ' '{print $2}')  
  
echo "$INTERFACES"  
  
# Se não houver interfaces, sai do script  
if [ -z "$INTERFACES" ]; then  
    echo "Nenhuma interface encontrada via SNMP!"  
    exit 1  
fi  
  
# Criar arquivo de status caso não exista  
[ ! -f "$STATUS_FILE" ] && touch "$STATUS_FILE"  
  
echo "Teste 1"  
  
# Loop para verificar o status de cada interface  
while IFS= read -r INTERFACE; do  
    # Obtém o índice SNMP da interface  
    INDEX=$(snmpwalk -v2c -c $COMMUNITY localhost .1.3.6.1.2.1.2.2.1.2 | grep  
"$INTERFACE" | awk -F': ' '{print $2}' | awk '{print $1}')  
  
    # Se não encontrou o índice, pula para a próxima interface  
    if [ -z "$INDEX" ]; then  
        continue  
    fi
```

```

# Obtém o status operacional da interface
STATUS=$(snmpget -v2c -c $COMMUNITY localhost
.1.3.6.1.2.1.2.2.1.8.$INDEX | awk -F'=' '{print $2}' | grep -o '[0-9]\+')

# Verifica se a interface já tem um status registrado no arquivo, ^ no regex
significa "início da linha"
PREV_STATUS=$(grep "^$INTERFACE " "$STATUS_FILE" | awk '{print $2}')

# Se o status mudou, envia notificação e atualiza o arquivo
if [[ "$STATUS" == "2" && "$PREV_STATUS" != "2" ]]; then
    echo "Interface $INTERFACE caiu! Enviando trap..."
    snmptrap -v 2c -c $COMMUNITY $TRAP_RECEIVER "
.1.3.6.1.4.1.8072.2.3.0.1 .1.3.6.1.4.1.8072.2.3.2.2 s "🔴 Interface $INTERFACE
caiu!"
    sed -i "/^$INTERFACE /d" "$STATUS_FILE"
    echo "$INTERFACE 2" >> "$STATUS_FILE"

elif [[ "$STATUS" == "1" && "$PREV_STATUS" != "1" ]]; then
    echo "Interface $INTERFACE voltou! Enviando trap..."
    snmptrap -v 2c -c $COMMUNITY $TRAP_RECEIVER "
.1.3.6.1.4.1.8072.2.3.0.1 .1.3.6.1.4.1.8072.2.3.2.2 s "🟢 Interface $INTERFACE
está ativa novamente!"
    sed -i "/^$INTERFACE /d" "$STATUS_FILE"
    echo "$INTERFACE 1" >> "$STATUS_FILE"
fi

done <<< "$INTERFACES"

```

Arquivo hd-trap.sh (monitora a partição principal "/")

```

#!/bin/bash

COMMUNITY="mypublic"
TRAP_RECEIVER="192.168.4.50"
LAST_USAGE_FILE="/tmp/last_hd_usage"

# OID para obter o espaço usado do HD (pode variar dependendo da
configuração SNMP)
OID_USED=".1.3.6.1.2.1.25.2.3.1.6.42"
OID_TOTAL=".1.3.6.1.2.1.25.2.3.1.5.42"

# Obtém valores via SNMP
USED=$(snmpwalk -v 2c -c $COMMUNITY localhost $OID_USED | awk '{print
$4}')
TOTAL=$(snmpwalk -v 2c -c $COMMUNITY localhost $OID_TOTAL | awk '{print
$4}')

```

```

# Calcula percentual de uso
if [[ -n "$USED" && -n "$TOTAL" && "$TOTAL" -ne 0 ]]; then
    HD_USAGE=$(( 100 * USED / TOTAL ))

    # Lê o último valor armazenado (se existir)
    LAST_USAGE=0
    if [[ -f $LAST_USAGE_FILE ]]; then
        LAST_USAGE=$(cat $LAST_USAGE_FILE)
    fi

    if [[ "$HD_USAGE" -gt 80 && "$HD_USAGE" -gt "$LAST_USAGE" ]]; then
        echo "Uso do HD acima de 80%! Enviando trap..."
        snmptrap -v 2c -c $COMMUNITY $TRAP_RECEIVER " \
            .1.3.6.1.4.1.8072.2.3.0.1 \
            .1.3.6.1.4.1.8072.2.3.2.2 s " 🚨 Uso do HD acima de 80%! Uso atual:
        ${HD_USAGE}%"
    fi

    # Atualiza o arquivo com o novo valor
    echo "$HD_USAGE" > "$LAST_USAGE_FILE"
else
    echo "Erro ao obter dados SNMP."
fi

```

É importante verificar se as constantes `OID_USED` e `OID_TOTAL` estão apontando para as `OIDs` corretas de seu HD.

5 - Adicione as tarefas ao `crontab` para rodar os scripts num intervalo de tempo

```
# crontab -e
```

```

* * * * * /root/gerencia/trabalho-gerencia/interface-trap.sh
* * * * * /root/gerencia/trabalho-gerencia/hd-trap.sh

```

Úteis

Ativar uma interface de rede

```
# ip link set enp0s8 up
```

Desativar uma interface de rede

```
# ip link set enp0s8 down
```

Simular o uso do HD

Crie um arquivo grande para ocupar espaço no HD:

```
# fallocate -l 5G /tmp/simulacao_hd_cheio.img
```

Verifique o uso do HD:

```
# df -h
```

Ajuste o tamanho conforme necessário para atingir 80% de uso.

Entidade Gerenciadora (**monit**ora a rede)

1 - Instale os pacotes do SNMP

```
# dnf install -y net-snmp net-snmp-utils
```

2 - Edite o arquivo de configuração do SNMP TRAP

```
# vim /etc/snmp/snmptrapd.conf
```

```
# Example configuration file for snmptrapd
#
# No traps are handled by default, you must edit this file!
#
# authCommunity log,execute,net public
# traphandle SNMPv2-MIB::coldStart /usr/bin/bin/my_great_script cold

# Permitir que o daemon seja executado sem autenticação
disableAuthorization yes

# Comunidade SNMP permitida
authCommunity log,execute,net mypublic

# Definir um formato detalhado para a saída das traps
format2 %B

# Redirecionar todas as traps recebidas para um script
traphandle default /root/trabalho-gerencia/capture-trap.sh
```

3 - Crie o arquivo que irá ser chamado quando uma trap for recebida

```
# vim /root/trabalho-gerencia/capture-trap.sh
```

```
#!/bin/bash

LOG_FILE="/var/log/snmp_trap.log"

# Captura toda a entrada do snmptrapd e salva no log
echo "$(date): Recebido do snmptrapd" >> "$LOG_FILE"

# Ler a entrada linha por linha
```

```

while read line; do
    # Filtrar a linha que contém a mensagem
    if [[ "$line" ==
*"NET-SNMP-EXAMPLES-MIB::netSnmpExampleHeartbeatName"* ]]; then
        # Extrair a mensagem real (após o último espaço)
        MESSAGE=$(echo "$line" | awk -F
'NET-SNMP-EXAMPLES-MIB::netSnmpExampleHeartbeatName ' '{print $2}')
        echo "$(date): $MESSAGE" >> "$LOG_FILE"
    fi

    if [[ "$line" == *"Uso do HD acima de 80%"* ]]; then
        MESSAGE=$(echo "$line" | awk -F 'Uso do HD acima de 80%' '{print $2}')
        /root/gerencia/bot-telegram.sh "🔥 Uso do HD acima de
80%!n${MESSAGE}"
    fi
done

/root/trabalho-gerencia/bot-telegram.sh "$MESSAGE" >> /var/log/bot-telegram.log
2>&1

```

4 - Crie o arquivo que irá enviar a mensagem ao bot do Telegram

```
# vim /root/trabalho-gerencia/bot-telegram.sh
```

```

#!/bin/bash
#
# INICIO
# bot_script

#source ShellBot.sh
source /root/trabalho-gerencia/ShellBot.sh

# Token do bot
bot_token='5719350128:AAGZzfVOElqoL9jTsC9c6NpVISUQ3kY3p1E'

# Chat grupo
chat_id='-4691016164'

# Inicializando o bot
ShellBot.init --token "$bot_token"

# Imprime o username do bot.
ShellBot.username

```

```
# Envia mensagem para o chat informado
ShellBot.sendMessage --chat_id ${chat_id} \
                    --text "$1" \
                    --parse_mode markdown
```

5 - Crie ou baixe o arquivo que lida com a api de bots do Telegram

```
# vim /root/trabalho-gerencia/ShellBot.sh
```

Utilize o conteúdo do link: [ShellBot.sh](https://github.com/ShellBot/ShellBot.sh)