

ANGD Mathrice 2009 / CIRM  
TP (Rev: 36 )  
SNMP / RRDTOOL

## Part I

# Net-SNMP

## 1 SNMPv1/SNMPv2c

### 1.1 Configurer le fichier snmpd.conf

Sous Debian/Ubuntu : /etc/snmp/snmpd.conf

Sous FreeBSD : /usr/local/etc/snmp/snmpd.conf

```
# First, map the community name (COMMUNITY) into a security name
# (local and mynetwork, depending on where the request is coming from):
#      sec.name  source          community
com2sec  paranoid default        public
#com2sec  readonly default        public
#com2sec  readwrite default       private

# Second, map the security names into group names:
#      sec.model  sec.name
group MyROSystem v1      paranoid
group MyROSystem v2c     paranoid
group MyROGroup  v1      readonly
group MyROGroup  v2c     readonly
group MyRWGroup  v1      readwrite
group MyRWGroup  v2c     readwrite

# Third, create a view for us to let the groups have rights to:
#      incl/excl subtree          mask
view all    included  .1          80
view system included  .iso.org.dod.internet.mgmt.mib-2.system

# Finally, grant the 2 groups access to the 1 view with different write permissions:
#      context sec.model sec.level match  read  write  notif
access MyROSystem ""      any      noauth  exact  system none  none
access MyROGroup  ""      any      noauth  exact  all    none  none
access MyRWGroup  ""      any      noauth  exact  all    all   none

syslocation ANGD MATHRICE CIRM
syscontact  admin@math.cnrs.fr
```

### 1.2 Configurer le fichier snmptrapd.conf

Sous Debian/Ubuntu : /etc/snmp/snmptrapd.conf

Sous FreeBSD : /usr/local/etc/snmp/snmptrapd.conf

```
authCommunity log,execute,net public
```

### 1.3 Configurer le fichier ~/.snmp/snmp.conf

```
defVersion 1
defCommunity public
```

## 2 SNMPv3

### 2.1 Configurer le fichier snmpd.conf

Sous Debian/Ubuntu : /etc/snmp/snmpd.conf

Sous FreeBSD : /usr/local/etc/snmp/snmpd.conf

```
createUser mathrice  SHA angdcirm AES marseille
createUser mathrice_rw SHA angdcirm AES marseille
# give read only access
rouser mathrice auth system
```

```
#rouser mathrice auth
# give read/write access
rwuser mathrice_rw auth system
```

## 2.2 Configurer le fichier snmptrapd.conf

Sous Debian/Ubuntu : /etc/snmp/snmptrapd.conf  
Sous FreeBSD : /usr/local/etc/snmp/snmptrapd.conf

```
createUser mathrice SHA angdcirm AES marseille
authUser log,execute,net mathrice
```

## 2.3 Configurer le fichier ~/.snmp/snmp.conf

```
# SNMPv3 username
defSecurityName mathrice
# MD5, SHA
defAuthType SHA
# noAuthNoPriv, authNoPriv, and authPriv
defSecurityLevel authPriv
# 8 caracteres minimum
defAuthPassphrase angdcirm
# DES, AES
defPrivType AES
# 8 caracteres minimum
# utile seulement si defSecurityLevel != { noAuthNoPriv, authNoPriv }
defPrivPassphrase marseille
# SNMPv3
defVersion 3
```

## 3 Démarrer le démon snmpd

```
$ sudo snmpd -f -Lf /tmp/mathrice_snmpd.log
```

## 4 Interrogation de l'agent SNMP local

### 4.1 snmpwalk

```
$ snmpwalk localhost
```

### 4.2 snmpget

```
$ snmpget localhost SNMPv2-MIB::sysName.0
```

### 4.3 Récupérer la consommation mémoire

En SNMPv1/SNMPv2c, dans le fichier snmpd.conf, il faut commenter la ligne "com2sec paranoid default public" et décommenter la ligne "com2sec readonly default public".

```
$ snmpwalk localhost UCD-SNMP-MIB::memory
```

```
UCD-SNMP-MIB::memIndex.0 = INTEGER: 0
UCD-SNMP-MIB::memErrorName.0 = STRING: swap
UCD-SNMP-MIB::memTotalSwap.0 = INTEGER: 6080560 kB
UCD-SNMP-MIB::memAvailSwap.0 = INTEGER: 6080132 kB
UCD-SNMP-MIB::memTotalReal.0 = INTEGER: 2061808 kB
UCD-SNMP-MIB::memAvailReal.0 = INTEGER: 74896 kB
UCD-SNMP-MIB::memTotalFree.0 = INTEGER: 6155028 kB
UCD-SNMP-MIB::memMinimumSwap.0 = INTEGER: 16000 kB
```

```
UCD-SNMP-MIB::memShared.0 = INTEGER: 0 kB
UCD-SNMP-MIB::memBuffer.0 = INTEGER: 185252 kB
UCD-SNMP-MIB::memCached.0 = INTEGER: 804512 kB
UCD-SNMP-MIB::memSwapError.0 = INTEGER: noError(0)
UCD-SNMP-MIB::memSwapErrorMsg.0 = STRING:
```

## 4.4 Récupérer la consommation disque

Dans le fichier `snmpd.conf`, ajouter une ligne `"disk /"`.

Dans le fichier `/etc/fstab`, assurez-vous d'une entrée `"< device> / ufs rw 0 0"` ou `<device>` correspond à la valeur retournée par la commande `mount`.

### 4.4.1 Sans seuil minimum

```
$ snmpwalk localhost UCD-SNMP-MIB::dskEntry
```

```
UCD-SNMP-MIB::dskIndex.1 = INTEGER: 1
UCD-SNMP-MIB::dskPath.1 = STRING: /
UCD-SNMP-MIB::dskDevice.1 = STRING: /dev/sda1
UCD-SNMP-MIB::dskMinimum.1 = INTEGER: 10000
UCD-SNMP-MIB::dskMinPercent.1 = INTEGER: -1
UCD-SNMP-MIB::dskTotal.1 = INTEGER: 381597632
UCD-SNMP-MIB::dskAvail.1 = INTEGER: 284627616
UCD-SNMP-MIB::dskUsed.1 = INTEGER: 77738608
UCD-SNMP-MIB::dskPercent.1 = INTEGER: 21
UCD-SNMP-MIB::dskPercentNode.1 = INTEGER: 1
UCD-SNMP-MIB::dskErrorFlag.1 = INTEGER: noError(0)
UCD-SNMP-MIB::dskErrorMsg.1 = STRING:
```

### 4.4.2 Avec un seuil minimum

Modifier la ligne `"disk /"` en mettant une valeur numérique supérieur à la taille `UCD-SNMP-MIB::dskAvail.1`, par exemple `"disk / 300000000"`  
Relancer la commande précédente.

Que voyez vous?

## 4.5 Récupérer le status d'un processus

Sous Debian/Ubuntu : `apache2`

Sous FreeBSD : `httpd`

Dans le fichier `snmpd.conf`, ajouter une ligne `"proc apache2"` et une ligne `"proc httpd"`.

```
$ snmpwalk localhost UCD-SNMP-MIB::prTable
```

```
UCD-SNMP-MIB::prIndex.1 = INTEGER: 1
UCD-SNMP-MIB::prIndex.2 = INTEGER: 2
UCD-SNMP-MIB::prNames.1 = STRING: apache2
UCD-SNMP-MIB::prNames.2 = STRING: httpd
UCD-SNMP-MIB::prMin.1 = INTEGER: 0
UCD-SNMP-MIB::prMin.2 = INTEGER: 0
UCD-SNMP-MIB::prMax.1 = INTEGER: 0
UCD-SNMP-MIB::prMax.2 = INTEGER: 0
UCD-SNMP-MIB::prCount.1 = INTEGER: 0
UCD-SNMP-MIB::prCount.2 = INTEGER: 6
UCD-SNMP-MIB::prErrorFlag.1 = INTEGER: error(1)
UCD-SNMP-MIB::prErrorFlag.2 = INTEGER: noError(0)
UCD-SNMP-MIB::prErrorMessage.1 = STRING: No apache2 process running.
UCD-SNMP-MIB::prErrorMessage.2 = STRING:
UCD-SNMP-MIB::prErrFix.1 = INTEGER: noError(0)
UCD-SNMP-MIB::prErrFix.2 = INTEGER: noError(0)
UCD-SNMP-MIB::prErrFixCmd.1 = STRING:
UCD-SNMP-MIB::prErrFixCmd.2 = STRING:
```

Sous Debian/Ubuntu :

```
$ sudo apache2ctl stop
```

Sous FreeBSD :

```
$ sudo apachectl stop
```

```
$ snmpwalk localhost UCD-SNMP-MIB::prTable
```

```
UCD-SNMP-MIB::prIndex.1 = INTEGER: 1
UCD-SNMP-MIB::prIndex.2 = INTEGER: 2
UCD-SNMP-MIB::prNames.1 = STRING: apache2
UCD-SNMP-MIB::prNames.2 = STRING: httpd
UCD-SNMP-MIB::prMin.1 = INTEGER: 0
UCD-SNMP-MIB::prMin.2 = INTEGER: 0
UCD-SNMP-MIB::prMax.1 = INTEGER: 0
UCD-SNMP-MIB::prMax.2 = INTEGER: 0
UCD-SNMP-MIB::prCount.1 = INTEGER: 0
UCD-SNMP-MIB::prCount.2 = INTEGER: 0
UCD-SNMP-MIB::prErrorFlag.1 = INTEGER: error(1)
UCD-SNMP-MIB::prErrorFlag.2 = INTEGER: error(1)
UCD-SNMP-MIB::prErrorMessage.1 = STRING: No apache2 process running.
UCD-SNMP-MIB::prErrorMessage.2 = STRING: No httpd process running.
UCD-SNMP-MIB::prErrFix.1 = INTEGER: noError(0)
UCD-SNMP-MIB::prErrFix.2 = INTEGER: noError(0)
UCD-SNMP-MIB::prErrFixCmd.1 = STRING:
UCD-SNMP-MIB::prErrFixCmd.2 = STRING:
```

```
$ sudo apache2ctl start
```

## 4.6 Monitorer les capteurs du serveur

Sous Linux : lm-sensors

```
$ snmpwalk localhost LM-SENSORS-MIB::lmSensors
```

```
LM-SENSORS-MIB::lmTempSensorsIndex.1 = INTEGER: 0
LM-SENSORS-MIB::lmTempSensorsIndex.2 = INTEGER: 1
LM-SENSORS-MIB::lmTempSensorsDevice.1 = STRING: Core0 Temp
LM-SENSORS-MIB::lmTempSensorsDevice.2 = STRING: Core1 Temp
LM-SENSORS-MIB::lmTempSensorsValue.1 = Gauge32: 30000
LM-SENSORS-MIB::lmTempSensorsValue.2 = Gauge32: 35000
```

Sous OpenBSD : hw.sensors

```
$ snmpwalk localhost OPENBSD-SENSORS-MIB::sensors
```

Sous FreeBSD : pas d'implémentation de 'hw.sensors'

## 4.7 Mise en évidence des problèmes de sécurité liés aux communautés SNMP

```
$ sudo tcpdump -i lo -X src port snmp
```

```
$ snmpwalk localhost
```

Que trouvez-vous comme informations sur votre machine?

⇒ commenter sans tarder la ligne "com2sec readonly default public" dans le fichier snmpd.conf

## 5 Rendre un serveur réactif

### 5.1 Configurer le fichier snmpd.conf

Sous Debian/Ubuntu : /etc/snmp/snmpd.conf

Sous FreeBSD : /usr/local/etc/snmp/snmpd.conf

```
# to send SNMPv3 INFORM
trapsess      -v3 -Ci -u mathrice -l authPriv -a SHA -A angdcirm -x AES -X marseille localhost
# SNMPv3 user to use for internal lookup
agentSecName  mathrice
#defaultMonitors yes
# monitor proc objects
monitor       -r 60 -o prNames -o prErrMsg "process down"    prErrorFlag != 0
monitor       -r 60 -o prNames              "process up"      prErrorFlag == 0
```

### 5.2 Configurer le fichier snmptrapd.conf

Sous Debian/Ubuntu : /etc/snmp/snmptrapd.conf

Sous FreeBSD : /usr/local/etc/snmp/snmptrapd.conf

```
createUser mathrice SHA angdcirm AES marseille
authUser log,execute,net mathrice
```

### 5.3 Démarrer le démon snmptrapd

```
$ sudo snmptrapd -f -Le
```

### 5.4 Emettre une INFORM SNMPv3

```
$ snmptrap -Ci localhost 0 SNMPv2-MIB::coldStart.0
```

```
$ snmpinform localhost 0 SNMPv2-MIB::coldStart.0
```

### 5.5 Ajout d'un gestionnaire générique vous envoyant un email

Sous Debian/Ubuntu : /etc/snmp/snmptrapd.conf, /usr/bin/traptoemail

Sous FreeBSD : /usr/local/etc/snmp/snmptrapd.conf, /usr/local/bin/traptoemail

Dans le fichier snmptrapd.conf ajouter la ligne

```
traphandle default \
    /usr/bin/perl /usr/bin/traptoemail \
    -s smtp.<yourdoamin.tld> \
    -f tp.snmp@mathrice.org \
    mathrice
```

### 5.6 Ajout d'un gestionnaire spécifique à un OID

Sous Debian/Ubuntu : /etc/snmp/snmptrapd.conf, /usr/bin/traptoemail

Sous FreeBSD : /usr/local/etc/snmp/snmptrapd.conf, /usr/local/bin/traptoemail

Dans le fichier snmptrapd.conf ajouter la ligne

```
traphandle SNMPv2-MIB::coldStart.0 \
    /usr/bin/perl /usr/bin/traptoemail \
    -s smtp.<yourdoamin.tld> \
    -f cold.start@mathrice.org \
    mathrice
```

## Part II

# RRDTOOL

## 6 Créer ses bases RRD

### 6.1 Créer une base RRD

Créer un script SHELL pour initialiser la base : `init.sh`

```
#!/bin/bash
# sur Linux :
#DATDEP='date +%s -d '2000-01-01''
DATDEP=946681200
# intervalle minimum de resolution = 1
DAT='expr $DATDEP - 1'
echo "Debut de RRD pour la date $DATDEP ($DAT)"

BASE=mathrice.rrd

[ -f $BASE ]&& rm -f $BASE
rrdtool create $BASE \
--start $DAT \
--step 10 \
DS:x:GAUGE:100:U:U \
DS:sinus:GAUGE:100:U:U \
RRA:LAST:0.5:1:8640 \
RRA:AVERAGE:0.5:6:1440 \
RRA:AVERAGE:0.5:30:288 \
RRA:AVERAGE:0.5:90:96 \
RRA:AVERAGE:0.5:180:48 \
RRA:MAX:0.5:1:8640 \
RRA:MAX:0.5:6:8640
```

Utiliser `rrdtool dump mathrice.rrd` pour retrouver les valeurs de la RRD créée.

Mot-clés : `ds`, `step`, `name`, `type`, `heartbeat`, `rra`, `cf`, `pdp_per_row`, `xff`, `database`, `row`

Indice : pour compter le nombre de valeurs consolidées, faire un `grep` sur le mot-cle 'row'

### 6.2 Alimenter la base RRD

Créer deux scripts pour alimenter (artificiellement) la base RRD :

- un script `update.py`
- exécuter : `update.py > update.sh`
- exécuter le script généré : `update.sh`

Script `update.py` :  
(sur Linux, la 1ere ligne est `#!/usr/bin/python` )

```
#!/usr/local/bin/python
# -*- coding: utf-8 -*-
from math import pi, sin
#pi = 3.14159
# nombre de mesures 1j = 8640 mesures toutes les 10sec.
N = 8640
# precision
n = 2 * pi / 100
x = 0
rrdfile = "mathrice.rrd"
debut = 946681200
```

```

fin = debut + N * 10
t = debut
# ecrite l'entete du script
print "#!/bin/bash"
print "#"
print "# 8640 mesures a intervalle de 10 sec."
print "# %d iterations de 10 sec. duree totale : %d sec." % (N,fin-debut)
print "# de %d a %d" % (debut,fin)
while (t <= fin):
    print "rrdtool update %s %d:%1.3f:%1.3f" % (rrdfile,t,x,sin(x))
    x = x + n
    t = t + 10

```

Utiliser `rrdtool dump mathrice.rrd` pour retrouver les valeurs entrées dans la RRD : vous devez constater que les `<raw>` ne sont plus vides (NaN).

### 6.3 Interroger la base RRD

Deux méthodes sont utilisées pour interroger la RRD database : `dump` et `fetch`.

Dump génère par défaut un fichier XML contenant toutes les données de la base, y compris l'entête et les paramètres des valeurs (DS) et des bases (RRA).

```
$ rrdtool dump mathrice.rrd
```

Fetch ne fait que l'extraction des données enregistrées dont on précise les limites :

```
$ rrdtool fetch mathrice.rrd LAST -s 946681200 -e s+1h
```

Le format de sortie est brut : 2 lignes d'entête (le nom des variables et une ligne vide) puis les données sous la forme :

```
timestamp: valeur1 valeur1
```

Ce format est très utile pour exporter les valeurs vers d'autres graphes, etc.

Le paramètre de résolution joue sur certaines fonctions de concaténations (`AVERAGE` par exemple) en modifiant le résultat fourni (par exemple pour moyenner sur une période différente). Mais il faut conserver la cohérence des paramètres : `-r RRR` doit rester un multiple des fonctions de consolidation des RAA déclarées.

Comparer :

```
$ rrdtool fetch mathrice.rrd AVERAGE -s 946681200 -e s+1h
```

et

```
$ rrdtool fetch mathrice.rrd AVERAGE -s 946681200 -e s+1h -r 900
```

Indice : regarder le timestamp de la sortie du fetch

### 6.4 Grapher les valeurs

#### 6.4.1 Choix des bonnes archives

Grapher la valeur sinus de `mathrice.rrd` sur une heure (par 1mn) :

```
\$ rrdtool graphv mathrice_minute1.png \
--title='Sinus par mn' \
--step 10 \
-s 946681220 -e 946684820 \
DEF:sinu=mathrice.rrd:sinus:AVERAGE \
LINE2:sinu#a00000:
```

Et avec cet exemple :

```
\$ rrdtool graphv mathrice_minute2.png \
--title='Sinus par mn' \
--step 10 \
-s 946681220 -e 946684820 \
DEF:sinu=mathrice.rrd:sinus:LAST \
LINE2:sinu#a00000:
```

Comparer les 2 graphes. Expliquer la différence.

### 6.4.2 Graphes à plusieurs courbes

Tracer le graphe à 2 courbes suivant :

```
\$ rrdtool graphv mathrice_double.png \
--title='Sinus par mn' \
--step 10 \
-s 946681220 -e 946684820 \
DEF:sinu=mathrice.rrd:sinus:LAST \
DEF:X=mathrice.rrd:x:LAST \
LINE2:X#00a000: \
LINE2:sinu#a00000:
```

Pour tracer des histogrammes, le mot-clé est AREA :

```
\$ rrdtool graphv mathrice_doubleArea.png \
--title='Sinus par mn' \
--step 10 \
-s 946681220 -e 946684820 \
DEF:sinu=mathrice.rrd:sinus:LAST \
DEF:X=mathrice.rrd:x:LAST \
AREA:X#00a000: \
LINE2:sinu#a00000:
```

Comparer avec :

```
\$ rrdtool graphv mathrice_doubleArea2.png \
--title='Sinus par mn' \
--step 10 \
-s 946681220 -e 946684820 \
DEF:sinu=mathrice.rrd:sinus:LAST \
DEF:X=mathrice.rrd:x:LAST \
LINE2:sinu#a00000: \
AREA:X#00a000:
```

Qu'en déduire ?

### 6.4.3 Légende

La ligne

```
--title='Sinus par mn'
```

indique le titre du graphique.

Au tracé du graphe mathrice\_double, effectuer les modifications suivantes :

- ajouter '-v Sin' à la ligne DEF:sinu...  
(noter que c'est la dernière ligne contenant -v qui prévaut : une seule légende sur Y)
- ajouter en fin de chaque ligne de tracé (LINE,AREA,...) une légende de courbe :
- ajouter en fin de graphe les lignes

```
GPRINT:sinu:LAST:"Actuel\ : %.11f -" \
GPRINT:sinu:MIN:"Mini\ : %.11f -" \
GPRINT:sinu:MAX:"Maxi\ : %.11f -" \
GPRINT:sinu:AVERAGE:"Moyen\ : %.11f"
```

Vous devez obtenir en final :

```
\$ rrdtool graphv mathrice_doubleTexte.png \
--title='Sinus par mn' \
--step 10 \
-s 946681220 -e 946684820 \
DEF:sinu=mathrice.rrd:sinus:LAST -v Sin \
DEF:X=mathrice.rrd:x:LAST \
LINE2:X#00a000:"X" \
LINE2:sinu#a00000:"sin(x)" \
GPRINT:sinu:LAST:"Actuel\ : %.11f -" \
GPRINT:sinu:MIN:"Mini\ : %.11f -" \
GPRINT:sinu:MAX:"Maxi\ : %.11f -" \
GPRINT:sinu:AVERAGE:"Moyen\ : %.11f"
```

#### 6.4.4 Tracés environnementaux

Au graphe précédent, ajouter :

- une ligne horizontale

```
HRULE:10#00a0a0
```

- une ligne verticale à 0h30

```
VRULE:946683000#00a0a0 \
```

- changer l'échelle verticale : forcer l'échelle verticale à [-5;50]

```
--lower-limit -5.0 \
--upper-limit 50.0 \
```

Vous devez obtenir :

```
\$ rrdtool graphv mathrice_doubleTexte.png \
--title='Sinus par mn' \
--step 10 \
-s 946681220 -e 946684820 \
--lower-limit -5.0 \
--upper-limit 50.0 \
DEF:sinu=mathrice.rrd:sinus:LAST -v Sin \
DEF:X=mathrice.rrd:x:LAST \
LINE2:X#00a000:"X" \
LINE2:sinu#a00000:"sin(x)" \
HRULE:10#00a0a0 \
VRULE:946683000#00a0a0 \
GPRINT:sinu:LAST:"Actuel\ : %.11f -" \
GPRINT:sinu:MIN:"Mini\ : %.11f -" \
GPRINT:sinu:MAX:"Maxi\ : %.11f -" \
GPRINT:sinu:AVERAGE:"Moyen\ : %.11f"
```

## 6.4.5 Calculs

Les calculs sont introduits par les lignes CDEF et VDEF.

Ajouter au graphe précédent les lignes correspondant aux valeurs suivantes :

$$\sin(2x), 5+\sin(5x), (2+\sin(x))/x$$

Soient les lignes de déclarations :

```
CDEF:sinuM2=sinu,2,* \
CDEF:cinqPsinuM5=5,sinu,5,*,+ \
CDEF:unS2Psin=X,sinu,2,+,/ \
```

et leurs tracés respectifs :

```
LINE:sinuM2#00a0c0:"sin*5" \
LINE:cinqPsinuM5#c00000:"5+sin(5x)" \
LINE:unS2Psin#0000c0:"sin(X)/X" \
```

Changer la taille du graphique généré :

```
--width 720 \
--height 440 \
```

Mettre  $\sin(x)/x$  en aire et changer sa couleur si elle dépasse 10

```
CDEF:Aok=unS2Psin,10,GT,0,unS2Psin,IF \
CDEF:Aover=unS2Psin,10,GT,unS2Psin,0,IF \
```

```
AREA:Aok#80c080:"OK" \
AREA:Aover#ff5050:"OVER" \
```

Vous devez obtenir :

```
\$ rrdtool graphv matrice_doubleTexte.png \
--width 720 \
--height 440 \
--title='Sinus par mn' \
--step 10 \
-s 946681220 -e 946684820 \
--lower-limit -5.0 \
DEF:sinu=matrice.rrd:sinus:LAST -v Sin \
DEF:X=matrice.rrd:x:LAST \
CDEF:sinuM2=sinu,2,* \
CDEF:cinqPsinuM5=5,sinu,5,*,+ \
CDEF:unS2Psin=X,sinu,2,+,/ \
CDEF:Aok=unS2Psin,10,GT,0,unS2Psin,IF \
CDEF:Aover=unS2Psin,10,GT,unS2Psin,0,IF \
HRULE:10#00a0a0 \
VRULE:946683000#00a0a0 \
AREA:Aok#80c080:"OK" \
AREA:Aover#ff5050:"OVER" \
LINE2:X#00a000:"X" \
LINE2:sinu#a00000:"sin(x)" \
LINE:sinuM2#00a0c0:"sin*5" \
LINE:unS2Psin#0000c0:"sin(X)/X" \
GPRINT:sinu:LAST:"Actuel\ : %.11f -" \
GPRINT:sinu:MIN:"Mini\ : %.11f -" \
GPRINT:sinu:MAX:"Maxi\ : %.11f -" \
GPRINT:sinu:AVERAGE:"Moyen\ : %.11f"
```

Mettre  $\sin(x)/x$  en aire et changer sa couleur si elle dépasse 10  
mais uniquement la partie qui dépasse 10

```
CDEF:Aover=unS2Psin,10,GT,unS2Psin,10,-,0,IF \
CDEF:AoverB=unS2Psin,10,GT,10,0,IF \
```

```
AREA:Aover#80c080:"" \
STACK:AoverB#ff0000:"OVER" \
```

Vous devez obtenir :

```

\$$ rrdtool graphv mathrice_doubleTexte.png \
--width 720 \
--height 440 \
--title='Sinus par mn' \
--step 10 \
-s 946681220 -e 946684820 \
--lower-limit -5.0 \
DEF:sinu=mathrice.rrd:sinus:LAST -v Sin \
DEF:X=mathrice.rrd:x:LAST \
CDEF:sinuM2=sinu,2,* \
CDEF:cinqPsinuM5=5,sinu,5,*,+ \
CDEF:unS2Psin=X,sinu,2,+,/ \
CDEF:Aok=unS2Psin,10,GT,0,unS2Psin,IF \
CDEF:Aover=unS2Psin,10,GT,10,0,IF \
CDEF:AoverB=unS2Psin,10,GT,unS2Psin,10,-,0,IF \
HRULE:10#00a0a0 \
VRULE:946683000#00a0a0 \
AREA:Aok#80c080:"OK" \
AREA:Aover#80c080:"" \
STACK:AoverB#ff0000:"OVER" \
LINE2:X#00a000:"X" \
LINE2:sinu#a00000:"sin(x)" \
LINE:sinuM2#00a0c0:"sin*5" \
LINE:unS2Psin#0000c0:"sin(X)/X" \
GPRINT:sinu:LAST:"Actuel\ : %.11f -" \
GPRINT:sinu:MIN:"Mini\ : %.11f -" \
GPRINT:sinu:MAX:"Maxi\ : %.11f -"\
GPRINT:sinu:AVERAGE:"Moyen\ : %.11f"

```

D'autres possibilités :

- mettre une échelle des ordonnées en log (-o)
- changer les grille d'alignements (-x-grid XXX ; -y-grid YYYY)
- changer les couleurs de fond
- etc.

## 7 Utiliser des bases d'outils. Exemple de collectd

### 7.1 Lancement de collectd

Pour le TP, l'outil est déjà lancé.

Outil standard de collecte de données locales à but de supervision/monitoring.

L'outil est essentiellement un moteur de plugins.

Il se configure depuis un fichier texte : *collectd.conf*

situé :

- pour Linux dans */etc/collectd/collectd.conf*,
- pour la machine VM du TP dans */usr/local/collectd.conf*

Ce fichier contient :

```

#
# Config file for collectd(1).
# Please read collectd.conf(5) for a list of options.
# http://collectd.org/
#

Hostname      "localhost"

```

```

FQDNLookup true
BaseDir "/var/db/collectd"
PIDFile "/var/run/collectd.pid"
PluginDir "/usr/local/lib/collectd"
TypesDB "/usr/local/lib/collectd/types.db"
Interval 10
ReadThreads 5

# mandatory
LoadPlugin rrdtool

# optionnal
LoadPlugin cpu
LoadPlugin df
LoadPlugin interface
LoadPlugin load
LoadPlugin memory
LoadPlugin swap

```

Pour lancer le démon :

- sur Linux : `/etc/init.d/collectd start`
- sur BSD : `/usr/local/etc/rc.d/collectd start`  
(éventuellement créer auparavant le répertoire `/var/db/collectd`)

Pour les exercices suivants, on prendra les données extraites des bases de collectd qui effectue des relevés sur la machine de TP.

Les bases sont dans les sous-répertoires définis par l'étiquette `BaseDir` de `collectd.conf`, par défaut :

- sur Linux : `/var/lib/collectd/rrd/localhost/plugin`
- sur BSD : `/var/db/collectd/localhost/plugin`

## 7.2 Structure des bases

Relever la structure des bases suivantes :

- `cpu-0` : `cpu-idle.rrd` `cpu-system.rrd` `cpu-user.rrd`
- `interface` : `if_octets-bge0.rrd` `if_octets.bge1` `if_packets-bge0.rrd` `if_packets.bge1`

On pourra utiliser le script suivant pour extraire les méta-données du XML :

```

#!/bin/bash
# getRRDmetaxml.sh v1.0.0 obf131109
#
# extrait les datas d'un xml
# formatte au sens RRD : DS et RAA
#
# A lancer sur le .xml issu de : rrdtool dump {fichier.rrd} {fichier.xml}
# Reprendre en final les valeurs des compteurs de lignes dans chaque RARA respectives
###
#
# traite un fichier xml
FIC=$1
FLTRSED="s/^\.*<(.*)>\(.*\)<\/.*>.*\/1:\2/"
#
echo "# releve de la structure de $1"
echo "#"
#
# variables generales
#

```

```

STEP=$(cat $FIC | grep step | sed $FLTRSED | sed "s:/=/")
echo "--$STEP"
echo ""
#
# variables liees a DS
#
LSTDS=$(cat $FIC | grep -v row | grep -e "name" -e "type" -e "heartbeat" -e "min" -e "max" \
| sed "$FLTRSED" \
| paste - - - - \
)
echo "$LSTDS" | awk '{printf("DS:%s:%s:%s:U:U\n", $2, $4, $6);}'
NBDS=$(echo "$LSTDS" | wc -l)
echo "# ON A TROUVE $NBDS DS"
echo ""
#
# variables liees a RRA
#
LSRRA=$(cat $FIC | grep -e "cf" -e "pdp_per_row" -e "xff" | sed "$FLTRSED" | paste - - - )
RRADEF=$(echo "$LSRRA" | awk '{printf("RRA:%s:%s:%s:xxxx\n", $2, $6, $4);}')
echo "$RRADEF"
NBRRA=$(echo "$RRADEF" | wc -l)
echo "# ON A TROUVE $NBRRA RAA"
echo ""

# comptage des lignes "row"
# plusieurs groupes possibles ...

LSTR=$(cat $FIC | grep -e '<row>' -e '<database>' | sed "s/^.*\(\.\*\)>.*\/1/" | sed "s/.*<//" | sed "s/>.*\/")
echo "# valeurs xxxx des RAA respectives"
echo "$LSTR" | awk ' \
BEGIN{ndr=0 ; cptr[0]=0; c=0} \
{ if($1 == "database"){if(cptr[ndr]!=0)printf("#RRA %d : %d rows\n", ndr, cptr[ndr]);ndr++; cptr[ndr]=0; c=1} \
else if(c==1 && $1 == "row"){cptr[ndr]++}\
} END { printf("#RRA %d : %d rows\n", ndr, cptr[ndr])}'

#.
```

Extraire les métadonnées des bases RRD et donner une interprétation des DS et RAA de ces bases pour une des bases cpu.

Vous devriez obtenir :

- cpu-idle (-step=10)

```

DS:value:COUNTER:20:U:U
RAA:AVERAGE:0.1:1:1200
RAA:MIN:0.1:1:1200
RAA:MAX:0.1:1:1200
RAA:AVERAGE:0.1:7:1235
RAA:MIN:0.1:7:1235
RAA:MAX:0.1:7:1235
RAA:AVERAGE:0.1:50:1210
RAA:MIN:0.1:50:1210
RAA:MAX:0.1:50:1210
RAA:AVERAGE:0.1:223:1202
RAA:MIN:0.1:223:1202
RAA:MAX:0.1:223:1202
RAA:AVERAGE:0.1:2635:1201
RAA:MIN:0.1:2635:1201
RAA:MAX:0.1:2635:1201
```

- les autres cpu ont la même structure
- les bases if\_octets-bge1.rrd et if\_packets-bge1.rrd ont deux sources : 'tx' et 'rx' mais ont les mêmes échelles de temps.

Interpréter ces données pour trouver les échelles de temps conservées dans l'archive.

### 7.3 Contenu des bases

Le dump d'une base rrd donne les dates de stockage des données :

- en tête du fichier (lastupdate)
- en commentaire des lignes de valeurs (row), avec ou sans ('NaN') donnée pour chaque RRA

Vous pouvez en déduire la plage horaire qui contient des données.

### 7.4 Grapher les valeurs

Exemple de bases : CPU et interface réseau

#### 7.4.1 Aire colorée

Exemple de COUNTER : interfaces

Les fichiers if\_octets-bge0.rrd et if\_packets-bge0.rrd ont deux sources : tx (émission) et rx (réception)

Tracer les courbes "à la MRTG" avec 'rx' en histogramme vert et 'tx' en courbe bleue pour les 2 bases en octets et en paquets, sur la dernière heure.

Vous devez obtenir :

```
\$ cd interface
\$ rrdtool graphv /tmp/interface_bge0-InOut-lin.png \
  --title=' ' \
  --step 10 \
  -s e-1h \
  DEF:in_o=if_octets-bge0.rrd:rx:AVERAGE \
  DEF:out_o=if_octets-bge0.rrd:tx:AVERAGE \
  DEF:in_p=if_packets-bge0.rrd:rx:AVERAGE \
  DEF:out_p=if_packets-bge0.rrd:tx:AVERAGE \
  AREA:in_o#00c000:"Oct-In" -v 'Bits/Pack. per Sec' \
  LINE4:in_p#c0c0c0:"Pk-In" \
  LINE:out_o#0000f0:"Oct-Out" \
  LINE:out_p#ff0000:"Pk-Out"
```

Pour plus de lisibilité des petites valeurs, utiliser une échelle logarithmique :

```
\$ rrdtool graphv /tmp/interface_bge0-InOut-log.png \
  --title=' ' \
  -o \
  --step 10 \
  -s e-1h \
  DEF:in_o=if_octets-bge0.rrd:rx:AVERAGE \
  DEF:out_o=if_octets-bge0.rrd:tx:AVERAGE \
  DEF:in_p=if_packets-bge0.rrd:rx:AVERAGE \
  DEF:out_p=if_packets-bge0.rrd:tx:AVERAGE \
  AREA:in_o#00c000:"Oct-In" -v 'Bits/Pack. per Sec' \
  LINE4:in_p#c0c0c0:"Pk-In" \
  LINE:out_o#0000f0:"Oct-Out" \
  LINE:out_p#ff0000:"Pk-Out"
```

## 7.4.2 Pile colorée

Autre exemple de COUNTER : cpu-0

Les fichiers cpu-system.rrd, cpu-user.rrd et cpu-idle.rrd ont une source : value  
Tracer les courbes des 3 valeurs sur les 20 dernières minutes.

Vous devez obtenir :

```
\$ cd ../cpu-0
\$ rrdtool graphv /tmp/cpu0-20mn-courbes.png \
  --title=' ' \
  --step 10 \
  -s e-20m \
  DEF:sys=cpu-system.rrd:value:AVERAGE \
  DEF:user=cpu-user.rrd:value:AVERAGE \
  DEF:idle=cpu-idle.rrd:value:AVERAGE \
  LINE:idle#c0c0f0:"Idle" \
  LINE:sys#c00000:"System" -v '% CPU-0' \
  LINE:user#00c000:"User"
```

Tracer ces courbes en histogrammes empilés (AREA puis STACK) sur 1 heure.

Vous devez obtenir :

```
\$ rrdtool graphv /tmp/cpu0-1h.png \
  --title=' ' \
  --step 10 \
  -s e-1h \
  DEF:sys=cpu-system.rrd:value:AVERAGE \
  DEF:user=cpu-user.rrd:value:AVERAGE \
  DEF:idle=cpu-idle.rrd:value:AVERAGE \
  AREA:sys#c00000:"System" -v '% CPU-0' \
  STACK:user#00c000:"User" \
  STACK:idle#c0c0f0:"Idle"
```

Pourquoi l'histogramme n'est pas toujours plat alors que les valeurs sont complémentaires ?

Comment lisser artificiellement l'aspect pour que Idle arrive toujours à 100% ?  
(Indice : faire un calcul)

Vous devez obtenir :

```
\$ rrdtool graphv /tmp/cpu0-1h-mod.png \
  --title=' ' \
  --step 10 \
  -s e-1h \
  DEF:sys=cpu-system.rrd:value:AVERAGE \
  DEF:user=cpu-user.rrd:value:AVERAGE \
  DEF:idle=cpu-idle.rrd:value:AVERAGE \
  CDEF:idd=100,user,sys,+,- \
  AREA:sys#c00000:"System" -v '% CPU-0' \
  STACK:user#00c000:"User" \
  STACK:idd#c0c0f0:"Idle"
```

Pourquoi le dernier histogramme ne montre pas la même réponse que le précédent ?

Pouvez-vous le vérifier ? Comment ?