

SNMP

Préambule

Comme son nom voudrait le faire croire, Simple Network Management Protocol est un protocole "simple" destiné à gérer des équipements informatiques, à distance ou non.

Ce n'est probablement pas un protocole primordial pour un petit réseau local, nous verrons tout de même qu'il peut rendre quelques services.

Il permet principalement de :

- visualiser une quantité pouvant être impressionnante d'informations concernant le matériel, les connexions réseaux, leur état de charge,
- modifier le paramétrage de certains composants,
- alerter l'administrateur en cas d'événements considérés comme grave,
- et d'autres choses encore...

Le tout à distance, via le réseau. Bref, grâce à ce protocole, un administrateur peut avoir la maîtrise de tout son réseau et de son parc informatique, sans quitter son bureau climatisé (les bureaux des administrateurs sont toujours climatisés, à cause du matériel informatique qui ne supporte pas bien la chaleur).

Miraculeux ? Pas vraiment. Simple ? Pas vraiment non plus.

Ce chapitre n'est pas destiné aux administrateurs qui souhaitent maîtriser ce protocole, mais plutôt aux curieux qui voudraient le découvrir et le mettre en oeuvre dans des situations simples.

Nous ne verrons pas comment écrire dans la MIB, ce qui, d'ailleurs ne pose pas de problèmes particuliers si l'on est autorisé à le faire, nous ne verrons pas non plus comment utiliser SNMP pour générer des alertes administratives.

Plan du chapitre

Préambule.....	1
Le principe : Présentation générale.....	4
Un agent sur chaque équipement.....	4
Un "manager" sur la station d'administration.....	5
Pratiquement.....	6
Le protocole.....	8
Les multiples versions.....	8
Le protocole lui-même.....	10
Installation des démons SNMP.....	12
Installation.....	12
Linux (Mandrake 9.1).....	12
Microsoft (Windows XP).....	12
Configuration.....	12
Linux (Mandrake 9.1).....	12
Microsoft (Windows XP).....	13
La MIB.....	16
C'est quoi la MIB ?.....	16
Structure de la MIB.....	16
Quelques exemples simples.....	18
Les communautés.....	19
Vérifications sur le réseau.....	20
Managers.....	22
Jawa Open Eyes.....	22
1 :Simple MIB Reference.....	22
2:Interrogation d'un noeud.....	23
A la recherche du bug.....	24
Recherche des interfaces : Internet.MGMT.MIB.Interface.....	24
Recherche des Ips : Internet.MGMT.MIB.Ip.ipAddrTable.....	26
Conclusions.....	27
Getif.....	27
MRTG intro.....	29
Qu'est-ce ?.....	29
De quoi a-t-on besoin ?.....	29
La manip proposée.....	29
Installation de MRTG.....	29
Le paquetage.....	29
Construire une configuration.....	30
Remarques.....	35
Une jolie page d'index pour MRTG.....	35
MRTG en production.....	37
Avertissement.....	37
La passerelle.....	37
Encore plus.....	38
La charge CPU.....	38
Qu'avons-nous fait ?.....	39

La mémoire disponible.....	43
Conclusions.....	44
Ressources.....	45
MRTG : index.html.....	45
MRTG : détails.....	46
MRTG avec 3 interfaces réseau.....	48
Charge active CPU %.....	49
Swap.....	51

Le principe : Présentation générale

Une machine informatique, quelle que soit sa fonction, a généralement beaucoup de choses à dire. Elle le dit le plus souvent discrètement, si bien que peu de gens l'entendent.

La plupart de ses utilisateurs ne l'écoute pas, sauf quand elle se met à hurler ou à faire la gueule, ce qui est tout de même assez rare. Elle hurle de diverses façons, une alarme sonore, par exemple, lorsqu'elle est en surchauffe, un écran bleu pour Windows, lorsque le noyau se plante irrémédiablement, elle fait la gueule, par exemple, lorsqu'elle refuse simplement de démarrer... Le reste du temps, elle chuchote divers indicateurs d'état, et certaines remarques, qu'elle consigne généralement dans les "logs", sorte de journaux intimes, que, finalement, assez peu de gens consultent.

Les administrateurs sont les seuls à s'y intéresser. Vous serez peut-être surpris de voir à quel point votre machine vous parle. Dans Windows (NTx), vous avez au moins quatre journaux, que vous pouvez rendre plus ou moins verbeux, mais c'est bien sûr sous Linux que vous découvrirez le mieux tout ce qu'une machine a à dire. L'objectif étant ici de parler de SNMP, nous n'entrerons pas plus dans ces détails, sachez, si ce n'est pas encore le cas, que sous Linux, vous avez deux répertoires fortement intéressants :

- /var/log qui contient tous les journaux tenus par votre machine,
- /proc qui est en fait un espace RAM dans lequel vous retrouverez une infinité de compteurs, d'indicateurs, de variables, drapeaux et autres choses encore.

Il y a enfin des informations dont le système ne se soucie pas obligatoirement, mais qui existent, remontées directement par le "firmware"¹, comme par exemple la température du processeur, la vitesse de rotation des ventilateurs.

Dans tout ça, il y a une foule d'informations qui intéressent les administrateurs et dont ils aimeraient disposer à distance via le réseau. Il est clair que lorsque le parc contient plusieurs centaines de machines, c'est tout de même plus agréable de disposer de toutes les informations en temps réel et de façon centralisée.

De plus, l'administrateur peut apprécier de pouvoir régler tel ou tel paramètre sans avoir à se déplacer sur le site de la machine concernée.

SNMP est exactement conçu pour répondre à tous ces besoins.

Un agent sur chaque équipement

Chaque équipement que l'on voudra "manager" à distance devra disposer d'un agent SNMP. Cet agent est un serveur, c'est à dire qu'il reste à l'écoute d'un port particulier : le port UDP 161.

La principale fonction de cet agent est de rester à l'écoute des éventuelles requêtes que l'administrateur lui enverra. Lorsqu'il recevra une requête, il y répondra, s'il y est autorisé. Plus

¹ **Firmware** : logiciel très proche du matériel, logé dans une mémoire non volatile : ROM, EPROM, etc. dont le but est de gérer basiquement ledit matériel, notamment son initialisation à la mise sous tension. Le "driver" sert à interfacer le système d'exploitation avec le "firmware" d'un composant périphérique, comme une carte réseau, une carte son...

Dans le cas d'une carte mère, le "firmware" peut être assimilé au BIOS, bien qu'à mon sens, le firmware n'en soit qu'un sous ensemble.

exactement, il répondra si la requête est émise par une entité autorisée. Autrement dit, cet agent est là pour écouter des requêtes et y répondre.

L'agent devra éventuellement pouvoir agir sur l'environnement local, si l'administrateur souhaite modifier un paramètre.

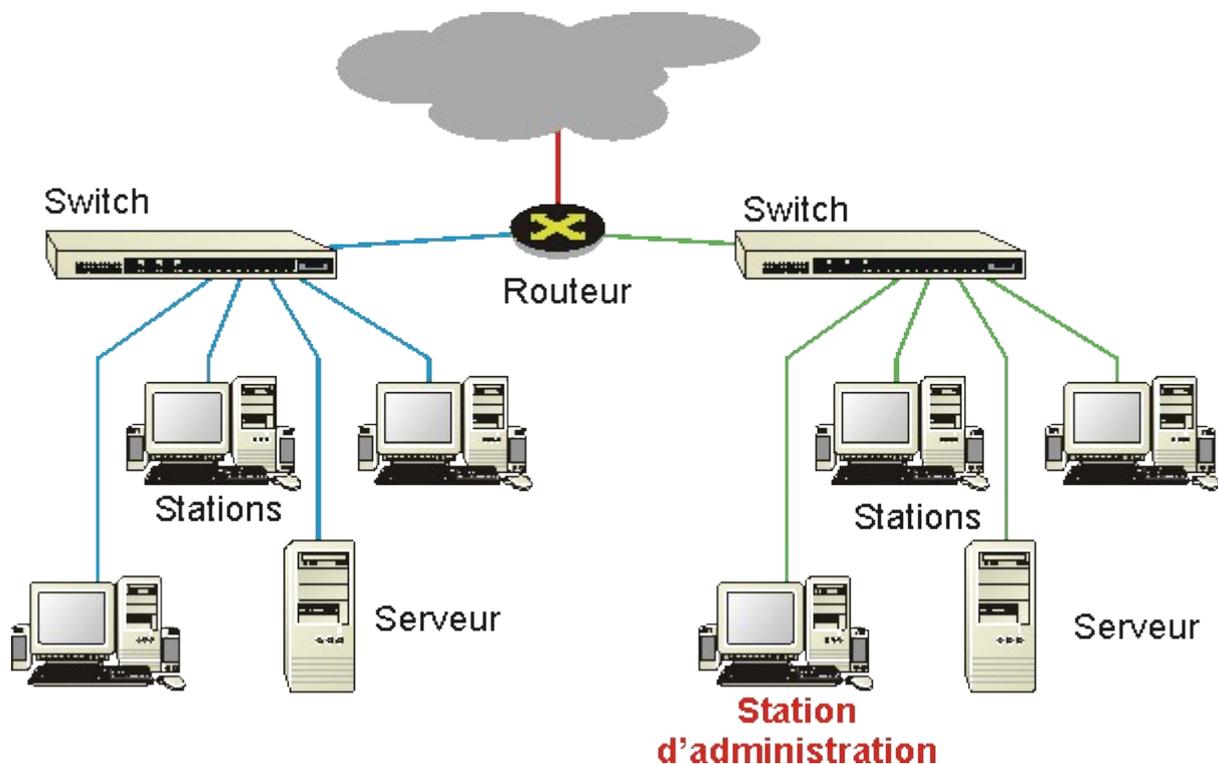
Par ailleurs, l'agent SNMP pourra éventuellement émettre des alertes de sa propre initiative, si il a été configuré pour ça. Par exemple, il pourra émettre une alerte si le débit sur une interface réseau atteint une valeur considérée par l'administrateur comme critique. Il peut y avoir une multitude d'alertes possibles, suivant la complexité de l'agent. La température du processeur, le taux d'occupation des disques durs, le taux d'occupation CPU...

On pourra trouver des agents SNMP sur des hôtes (ordinateurs) mais aussi sur des routeurs, des ponts, des switches...

Un "manager" sur la station d'administration

L'administrateur, sur sa machine d'administration, dispose d'un outil dit "manager". C'est avant tout un client, dans la mesure où c'est lui qui envoie les requêtes aux divers agents SNMP du réseau. Il devra aussi disposer d'une fonction serveur, car il doit rester à l'écoute des alertes que les divers équipements sont susceptibles d'émettre à tout moment.

Un petit dessin... Tous les équipements disposent d'un agent SNMP, la station d'administration dispose du "manager"



Dans un tel cas, l'administrateur peut, en théorie, observer le comportement de la totalité de son réseau depuis sa station d'administration. C'est vrai pour un LAN, c'est aussi vrai pour un WAN (entendez par là que l'équipement à administrer peut se trouver à des centaines de kilomètres).

Les "boîtes noires" (routeurs, switches...) sont équipés éventuellement d'un agent SNMP (c'est une

question de prix).

Pour les serveurs et les stations il existe probablement un logiciel à installer, quelque soit le système (sérieux). Pour Linux, c'est "NET-SNMP" (ou "UCD-SNMP"), Windows XP professionnel, Windows 2000 "pro" et "server" permettent d'installer un agent SNMP.

Le Manager dispose d'un serveur qui reste à l'écoute, sur le port UDP 162, des éventuels signaux d'alarme.

Pratiquement...

Juste pour donner un exemple, voyons ce que ça donne, si, depuis une machine Linux, nous interrogeons l'agent SNMP d'une station Windows XP.

Nous faisons ça avec un outil dont Linux a le secret, et que nous verrons plus loin. Sa particularité est de produire une sortie strictement illisible pour le non initié. Il s'agit de snmpwalk.

Le but n'est pas ici de tout comprendre mais juste de donner une idée des informations que l'on peut obtenir. La sortie faisant 1577 lignes, nous n'en verrons que quelques unes.

```
# Quelques infos sur la machine et son OS :
SNMPv2-MIB::sysDescr.0 = STRING: Hardware: x86 Family 15 Model 2 Stepping 4
      AT/AT COMPATIBLE - Software: Windows 2000 Version 5.1
      (Build 2600 Uniprocessor Free)
...
# Le type de carte réseau :
IF-MIB::ifDescr.2 = STRING: D-Link DFE-528TX PCI Adapter
      - Miniport d'ordonnancement de paquets
...
#Quelques infos sur la configuration IP :
IP-MIB::ipAdEntAddr.127.0.0.1 = IpAddress: 127.0.0.1
IP-MIB::ipAdEntAddr.192.168.0.10 = IpAddress: 192.168.0.10
IP-MIB::ipAdEntIfIndex.127.0.0.1 = INTEGER: 1
IP-MIB::ipAdEntIfIndex.192.168.0.10 = INTEGER: 2
IP-MIB::ipAdEntNetMask.127.0.0.1 = IpAddress: 255.0.0.0
IP-MIB::ipAdEntNetMask.192.168.0.10 = IpAddress: 255.255.255.0
...
RFC1213-MIB::ipRouteNextHop.0.0.0.0 = IpAddress: 192.168.0.250
...
#Les ressources de mémoire de masse :
HOST-RESOURCES-MIB::hrStorageDescr.2 = STRING: C:\ Label: Serial Number d803c0ad
HOST-RESOURCES-MIB::hrStorageDescr.3 = STRING: D:\
HOST-RESOURCES-MIB::hrStorageDescr.4 = STRING: E:\ Label:DATA Serial Number 10822ea3
HOST-RESOURCES-MIB::hrStorageDescr.5 = STRING: F:\
HOST-RESOURCES-MIB::hrStorageDescr.6 = STRING: G:\ Label:ZIP-100 Serial Number 1de00d03
...
# Les imprimantes installées :
HOST-RESOURCES-MIB::hrDeviceDescr.1 = STRING: HP DeskJet 600
HOST-RESOURCES-MIB::hrDeviceDescr.2 = STRING: HP DeskJet 550C
...
# Les services en cours d'exécution :
HOST-RESOURCES-MIB::hrSWRunName.1516 = STRING: "rundll32.exe"
HOST-RESOURCES-MIB::hrSWRunName.1520 = STRING: "OLFSNT40.EXE"
HOST-RESOURCES-MIB::hrSWRunName.1820 = STRING: "NAVAPW32.EXE"
HOST-RESOURCES-MIB::hrSWRunName.1836 = STRING: "spampal.exe"
HOST-RESOURCES-MIB::hrSWRunName.2016 = STRING: "rundll32.exe"
HOST-RESOURCES-MIB::hrSWRunName.2128 = STRING: "msimn.exe"
...
# Les applications installées !!!
HOST-RESOURCES-MIB::hrSWInstalledName.9 = STRING: "Ethereal 0.9.8"
HOST-RESOURCES-MIB::hrSWInstalledName.10 = STRING: "Exodus Jabber Client (remove only)"
HOST-RESOURCES-MIB::hrSWInstalledName.11 = STRING: "FileZilla (remove only)"
...
HOST-RESOURCES-MIB::hrSWInstalledName.79 = STRING: "SpamPal: BadWords plugin"
HOST-RESOURCES-MIB::hrSWInstalledName.80 = STRING: "SpamPal"
...
# Ainsi que leur date d'installation...
HOST-RESOURCES-MIB::hrSWInstalledDate.9 = STRING: 2002-12-17,10:29:48.0
```

```
HOST-RESOURCES-MIB::hrSWInstalledDate.10 = STRING: 2002-12-15,15:53:18.0
HOST-RESOURCES-MIB::hrSWInstalledDate.11 = STRING: 2003-1-9,10:50:28.0
...
HOST-RESOURCES-MIB::hrSWInstalledDate.79 = STRING: 2003-7-2,17:48:24.0
HOST-RESOURCES-MIB::hrSWInstalledDate.80 = STRING: 2003-6-30,17:38:58.0
...
```

Encore une fois, ce ne sont que des morceaux choisis, il y en a 1577 lignes et toutes ne sont pas si facilement compréhensibles. Nous verrons par la suite que snmpwalk peut être utilisé de façon plus fine, pour n'obtenir qu'un bout de branche.

Comme vous pouvez le remarquer, SNMP peut se révéler être plutôt du genre indiscret. Est-il nécessaire d'indiquer que, dans l'installation par défaut de l'agent SNMP de Windows XP, celui-ci racontera votre vie à n'importe qui ? N'installez pas SNMP sans savoir ce que vous faites et sans vérifier sa configuration par défaut...

Le protocole

Les multiples versions

Commençons par le plus désagréable. SNMP existe au moins dans les versions 1, 2c, 2 et 3. Comme pour tout protocole, les références sont des RFC. Si vous aimez ce genre de lecture :

Version	année	RFCs	Titre	Statut
v1	1990	1155	Structure and Identification of Management Information for TCP/IP-based Internets	standard
		1156	Management Information Base for network management of TCP/IP-based internets	historique
		1157	Simple Network Management Protocol (SNMP)	historique
v2c (classic)	1993	1441	Introduction to version 2 of the Internet-standard Network Management Framework	historique, proposé comme standard
		1442	Structure of Management Information for version 2 of the Simple Network Management Protocol (SNMPv2)	standard proposé, remplacé par RFC-1902
		1443	Textual Conventions for version 2 of the Simple Network Management Protocol (SNMPv2)	standard proposé, remplacé par RFC-1903
		1444	Conformance Statements for version 2 of the Simple Network Management Protocol (SNMPv2)	standard proposé, remplacé par RFC-1904
		1445	Administrative Model for version 2 of the Simple Network Management Protocol (SNMPv2)	historique
		1446	Security Protocols for version 2 of the Simple Network Management Protocol (SNMPv2)	historique
		1447	Party MIB for version 2 of the Simple Network Management Protocol (SNMPv2)	historique
		1448	Protocol Operations for version 2 of the Simple Network Management Protocol (SNMPv2)	standard proposé, remplacé par RFC-1905
		1449	Transport Mappings for version 2 of the Simple Network Management Protocol (SNMPv2)	standard proposé, remplacé par RFC-1906
		1450	Management Information Base for version 2 of the Simple Network Management Protocol (SNMPv2)	standard proposé, remplacé par RFC-1907
		1451	Manager-to-Manager Management Information Base	historique
		1452	Coexistence between version 1 and version 2 of the Internet-standard Network Management	standard proposé, remplacé par RFC-1908

			Framework	
v2	1996	1901	Introduction to Community-based SNMPv2	historique, proposé comme expérimental
		1902	Structure of Management Information for Version 2 of the Simple Network Management Protocol (SNMPv2)	standard, remplacé par RFC-2578
		1903	Textual Conventions for Version 2 of the Simple Network Management Protocol (SNMPv2)	standard, remplacé par RFC-2579
		1904	Conformance Statements for Version 2 of the Simple Network Management Protocol (SNMPv2)	standard, remplacé par RFC-2580
		1905	Protocol Operations for Version 2 of the Simple Network Management Protocol (SNMPv2) (3416)	standard, remplacé par RFC-3416
		1906	Transport Mappings for Version 2 of the Simple Network Management Protocol (SNMPv2) (3417)	standard, remplacé par RFC-3417
		1907	Management Information Base for Version 2 of the Simple Network Management Protocol (SNMPv2) (3418)	standard, remplacé par RFC-3418
		1908	Coexistence between Version 1 and Version 2 of the Internet-standard Network Management Framework (2576)	standard, remplacé par RFC-2576
v3	1999	2571	An Architecture for Describing SNMP Management Frameworks	standard, remplacé par RFC-3411
		2572	Message Processing and Dispatching for the Simple Network Management Protocol (SNMP)	standard, remplacé par RFC-3412
		2573	SNMP Applications	standard, remplacé par RFC-3413
		2574	User-based Security Model (USM) for version 3 of the Simple Network Management Protocol (SNMPv3)	standard, remplacé par RFC-3414
		2575	View-based Access Control Model (VACM) for the Simple Network Management Protocol (SNMP)	standard, remplacé par RFC-3415
v2 et v3	2000	2576	Coexistence between Version 1, Version 2, and Version 3 of the Internet-standard Network Management Framework	standard proposé

	2002	3411	An Architecture for Describing Simple Network Management Protocol (SNMP) Management Frameworks	standard
		3412	Message Processing and Dispatching for the Simple Network Management Protocol (SNMP)	standard
		3413	Simple Network Management Protocol (SNMP) Applications	standard
		3414	User-based Security Model (USM) for version 3 of the Simple Network Management Protocol (SNMPv3)	standard
		3415	View-based Access Control Model (VACM) for the Simple Network Management Protocol (SNMP)	standard
		3416	Version 2 of the Protocol Operations for the Simple Network Management Protocol (SNMP)	standard
		3417	Transport Mappings for the Simple Network Management Protocol (SNMP)	standard
		3418	Management Information Base (MIB) for the Simple Network Management Protocol (SNMP)	standard

D'une manière générale, reportez-vous au site [rfc-editor](http://www.rfc-editor.org/)² pour retrouver les RFC, avec les informations sur leur statut (expérimental, obsolète, actif...)

Comme vous le voyez, il règne la plus grande confusion dans la définition des versions 2 et 3 de SNMP.

La version 2, commencée à être définie en 1996, ne se voit réellement finalisée qu'en décembre 2002, après que la version 3 ait été définie. Ladite version 3, trop récente, n'est pas encore largement utilisée, si bien que c'est la version 1 qui se retrouve être supportée par tous, avec ses (graves) défauts, comme nous le verrons plus loin.

Pour que SNMP fonctionne, il n'y a pas qu'un protocole d'échange à définir. Il y a aussi une standardisation des informations que ce protocole peut transporter. C'est un protocole Internet, il doit être utilisable sur des plate-formes hétérogènes (matériel comme système d'exploitation).

C'est pour cette raison que l'on parlera de MIB (Management Information Base) et de SMI (Structure of Management Information).

Dans ce qui suit, nous nous appuyerons principalement sur SNMP v1. Commençons par le plus simple.

Le protocole lui-même

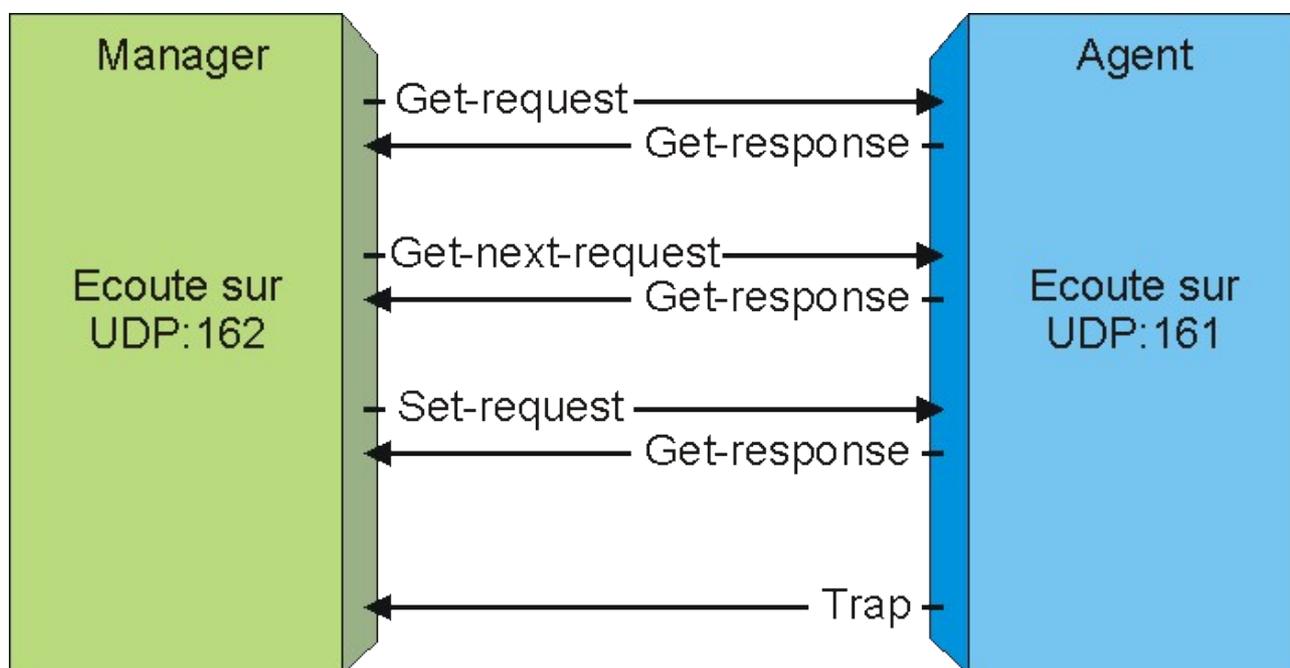
SNMP tire son "N" du fait qu'il s'appuie sur UDP d'une part, et qu'il ne propose qu'un nombre très restreint de commandes.

Les commandes sont les suivantes (version 1) :

² rfc-editor : <http://www.rfc-editor.org/cgi-bin/rfcsearch.pl>

Commande	Action
get-request	Le Manager SNMP demande une information à un agent SNMP
get-next-request	Le Manager SNMP demande l'information suivante à l'agent SNMP
set-request	Le Manager SNMP met à jour une information sur un agent SNMP
get-reponse	L'agent SNMP répond à un get-request ou a un set-request
trap	L'agent SNMP envoie une alarme au Manager

Jusque là, c'est on ne peut plus simple. L'agent utilise le port 161 et le manager, le port 162. Graphiquement, ça donne ceci :



Les commandes get-request, get-next-request et set-request sont toutes émises par le manager à destination d'un agent et attendent toutes une réponse get-response de la part de l'agent.

La commande trap est une alerte. Elle est toujours émise par l'agent à destination du manager, et n'attend pas de réponse.

Comme vous le voyez, jusque là, c'est extrêmement simple. Rassurez-vous, ça va nettement se compliquer avec la MIB et la SMI.

Installation des démons SNMP

L'installation, sous Linux comme sous Windows ne pose guère de problèmes. Il faudra juste faire un peu attention à ne pas laisser l'accès à n'importe qui sur n'importe quoi.

Installation

Linux (Mandrake 9.1)

Il suffit d'installer les paquets NET-SNMP et NET-SNMP UTILS, avec l'outil que vous préférez.

Microsoft (Windows XP)

Allez dans le panneau de configuration :

- Ajout/suppression de logiciels,
- ajouter ou supprimer des composants Windows,
- sélectionnez "Outils de gestion et d'analyse",
- cliquez sur "Détails",
- cochez "SNMP (Protocole simplifié de gestion de réseaux)".

Comme vous le voyez, c'est assez simple à installer, dans un cas comme dans l'autre.

Configuration

Là, ça va être un peu plus compliqué sous Linux.

Nous n'allons pas nous lancer dans une configuration très fine. Juste le minimum pour que :

- Toute machine du LAN puisse lire la totalité de la MIB, à travers la communauté "public",
- seule la machine locale puisse écrire, à travers la communauté "xPzrWf" (plus difficile à inventer que "private").

Linux (Mandrake 9.1)

Commencez par sauvegarder le fichier /etc/snmp/snmpd.conf qui a été créé avec l'installation. Il est largement documenté, même s'il n'est pas très compréhensible, il vous servira certainement un jour ou l'autre.

Créez ensuite un nouveau fichier /etc/snmp/snmpd.conf comme suit :

```
syscontact chris@gwl.maison.mrs
syslocation Le placard de l'entree

# 1° créer des relations entre les communautés et des noms de sécurité
#   nom.secu   source           communauté
com2sec Local localhost          xPzrWf
com2sec LocalNet 192.168.0.0/24 public

# 2° créer des relations entre des noms de groupes et les noms de sécurité
```

```
#      nom.groupe  version      nom.secu
group  RWGroup     v1           Local
group  ROGroup     v1           LocalNet

#3° Créer les diverses vues qui seront autorisées aux groupes
#
view   tout        included     .1

#4° Indiquee les accès aux vues suivant les groupes
#      nom.groupe  contexte  modele.secu  niveau.secu  prefixe  lecture  ecriture  notification
access ROGroup     ""        v1           noauth      exact    tout     none     none
access RWGroup     ""        v1           noauth      exact    tout     tout     none
```

Vous voyez comme c'est simple...

La communauté "public" pourra lire la totalité de la MIB depuis n'importe quelle machine du LAN (192.168.0.0/24), la communauté "xPzrWf" pourra lire et écrire partout où ce sera possible dans la MIB, uniquement depuis le poste local (localhost).

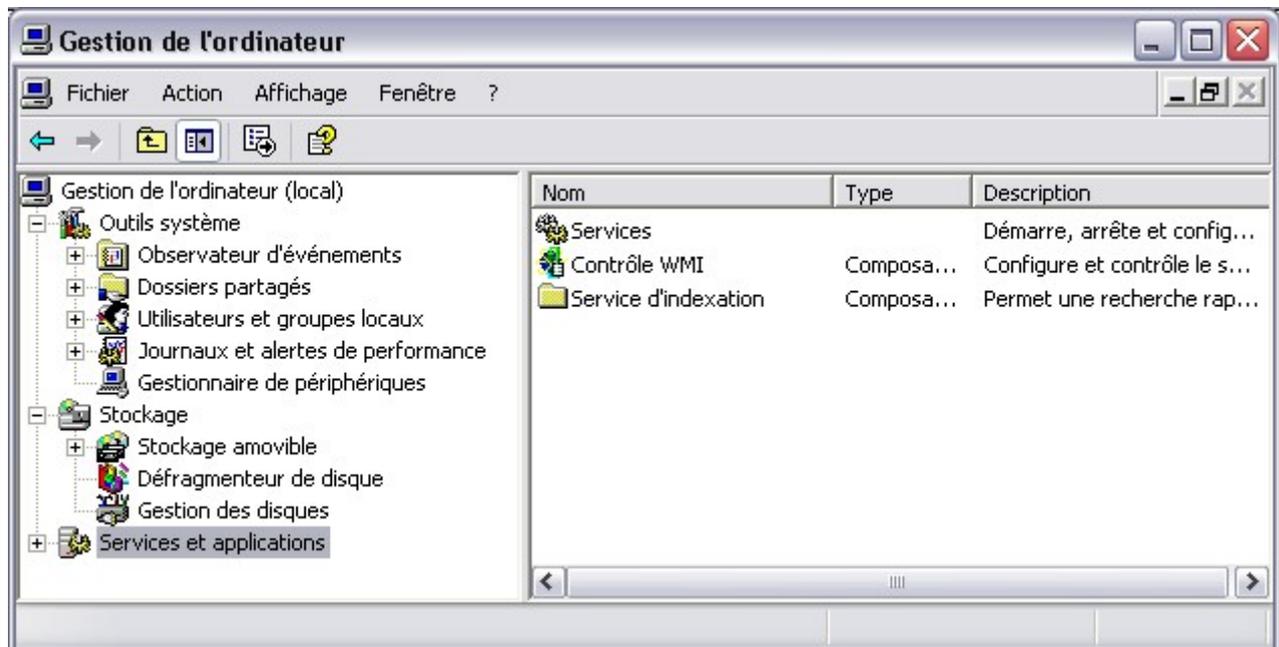
Je vous laisse donc vous débrouiller, si vous souhaitez faire plus compliqué, ce qui sera certainement nécessaire sur un réseau d'entreprise ou un réseau scolaire.

N'oubliez pas de relancer le démon par un :

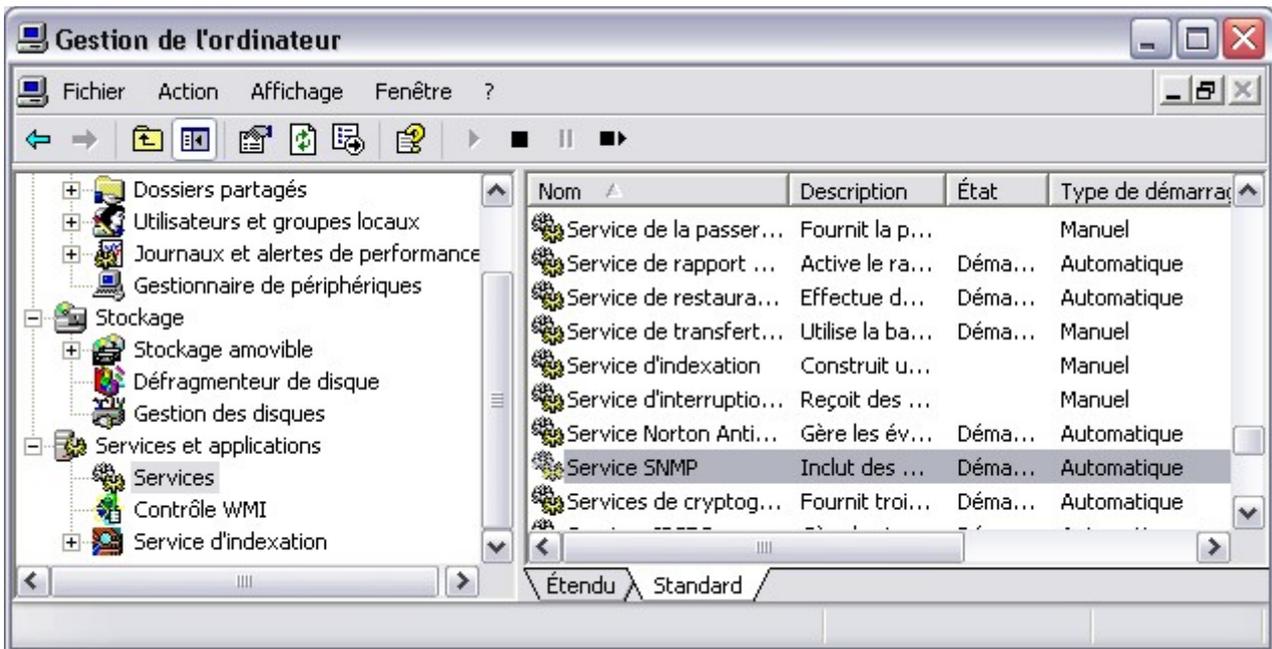
```
/etc/init.d/snmpd restart
```

Microsoft (Windows XP)

Gérez votre poste de travail, par exemple en cliquant du bouton droit sur "Poste de travail" dans le menu "Démarrer".



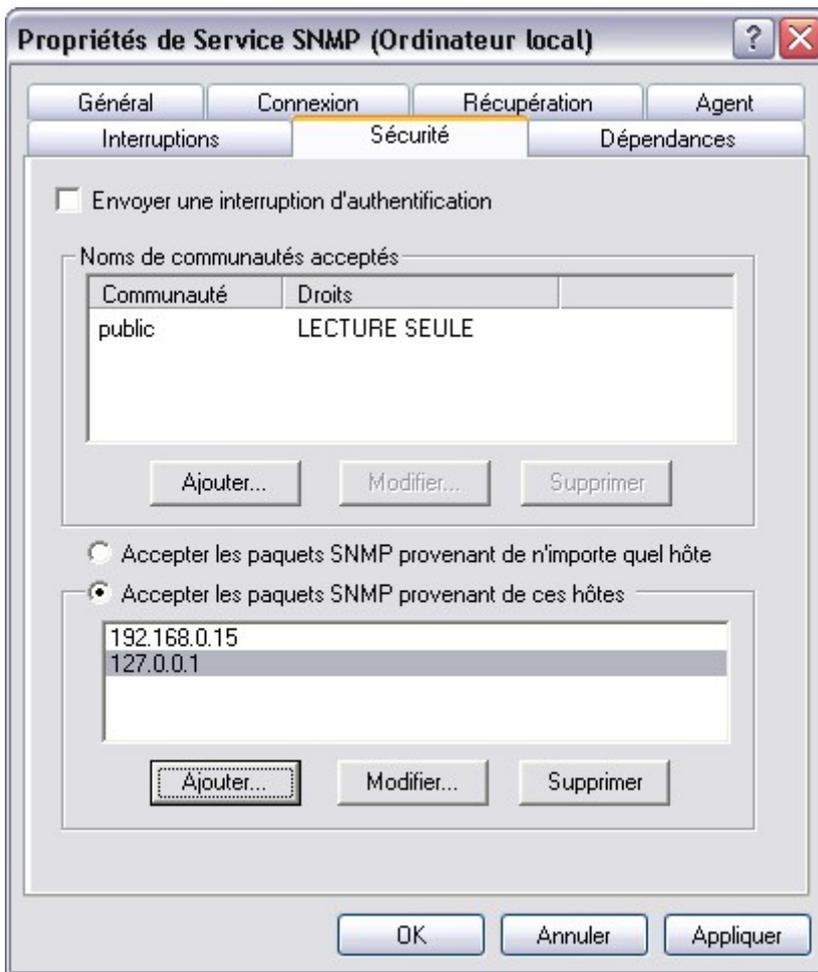
Développez "Service et applications" et repérez "Service SNMP" :



Double-cliquez dessus pour configurer l'agent.



Pour l'instant, c'est plutôt plus simple qu'avec Linux...



Mais là, il est facile de remarquer que pour être simple, ça en devient simpliste. Nous ne pouvons pas faire aussi fin qu'avec Linux, loin de là.

Nous ne créons donc qu'une communauté "public", qui n'aura que les droits de lecture depuis deux machines seulement.

La MIB

C'est quoi la MIB ?

C'est donc la base d'informations de gestion. Il y a, nous commençons à bien le comprendre :

- des informations à consulter,
- des paramètres à modifier,
- des alarmes à émettre...

Tout ça, en principe, de façon indépendante du matériel et du logiciel. Il faut donc que SNMP permette de retrouver ces informations et d'agir sur les paramètres de façon indépendante du matériel, comme du logiciel.

La MIB se présente comme une base de données normalisée, qui permettra de lire et d'écrire sur les équipements distants, de façon également normalisée. Ce sera à l'agent lui-même de faire la traduction entre les informations transmises par SNMP et la plate-forme.

Structure de la MIB

Elle est extrêmement simple pour un informaticien, et donc assez compliquée pour un cerveau humain "normal".

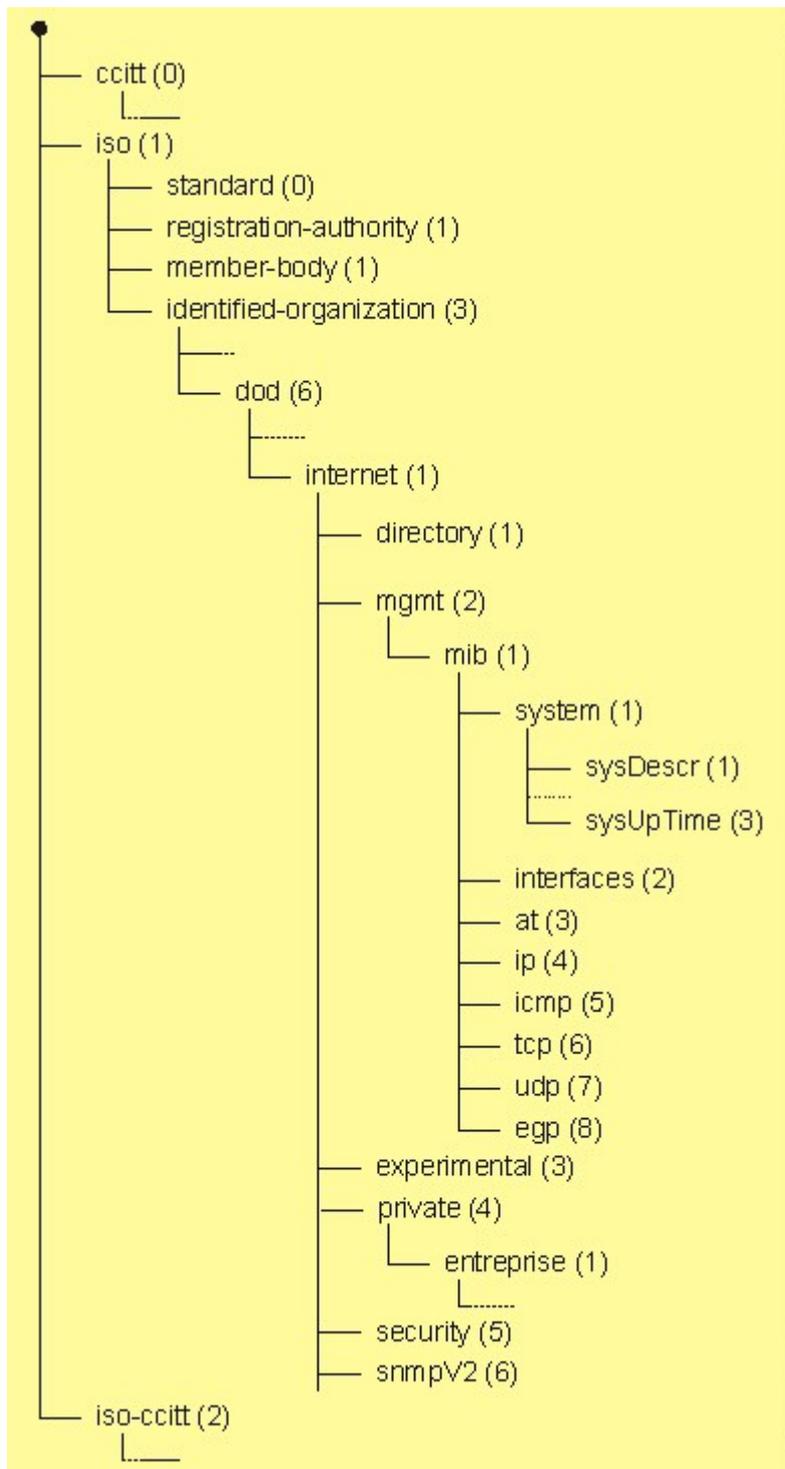
Elle est organisée hiérarchiquement, de la même façon que l'arborescence des domaines Internet.

Elle contient une partie commune à tous les agents SNMP en général, une partie commune à tous les agents SNMP d'un même type de matériel et une partie spécifique à chaque constructeur.

Elle peut contenir des scalaires (valeurs uniques) ou des tableaux de scalaires.

Non seulement la structure est normalisée, mais également les appellations des diverses rubriques. Ces appellations ne servent à rien, si ce n'est à rendre les choses plus lisibles (ce qui peut prêter à rire). En réalité, chaque niveau de la hiérarchie est repéré par un index numérique et SNMP n'utilise que cette façon de faire.

Mais voyons ça pratiquement :



Ceci, bien entendu, n'est pas une représentation exhaustive, les arbres, c'est bien connu, sont faits pour se développer.

Les branches qui n'aboutissent pas sont là pour indiquer qu'il y a le plus souvent des choses dessus, mais les branches qui semblent finies ne le sont pas forcément.

Les appellations en texte ("member-body" par exemple), peuvent varier légèrement d'une implémentation à l'autre. Rappelons-nous qu'elles ne sont là que pour rendre les choses plus compréhensibles pour le lecteur humain (informaticien). Ce qui ne varie jamais, en revanche, c'est l'index qui y correspond, placé entre parenthèses sur le dessin.

Nous pouvons imaginer à quel point ce système peut être étendu à divers domaines. Pour un réseau IP, c'est clairement le noeud "dod (6)" qui sera le plus utilisé et particulièrement le sous noeud "internet (1)".

Comme d'habitude, avec ce genre d'organisation, un paramètre sera accessible lorsque l'on connaîtra son chemin complet, depuis la racine de l'arbre, en haut sur le dessin. En informatique, les arbres ont toujours leurs racines en haut, parce que c'est un monde "underground".

Il s'agit ici de l'arbre représentant les divers paramètres. La valeur du paramètre (poétiquement la feuille de l'arbre), sera atteinte en y ajoutant un dernier index, qui commencera à 0 et y restera pour les scalaires et ira jusqu'à n pour les tableaux.

Comme cet arbre va dépendre de l'agent, il sera pratique de disposer d'un outil permettant de parcourir cet arbre pour en découvrir la structure. Cet outil se trouvera dans le manager SNMP.

Quelques exemples simples

Utilisons les outils de Linux, dont le manque d'ergonomie n'a d'égal que le fait qu'ils nous obligent à comprendre ce que l'on fait.

Quelque chose de très simple, pour commencer, obtenir " l'uptime " de la machine locale, le temps depuis lequel le système est en service. Dans l'arbre que nous avons juste dessus, nous voyons qu'il faut passer par les noeuds .1.3.6.1.2.1.1.3 . Nous ajoutons un zéro à cette suite, pour indiquer la feuille :

```
[root@gw mibs]# snmpget -v 1 -c public localhost .1.3.6.1.2.1.1.3.0
system.sysUpTime.0 = Timeticks: (23599841) 2 days, 17:33:18.41
```

Nous ne sommes pas encore en mesure de comprendre toute la finesse de cette commande, mais nous pouvons déjà observer deux choses :

- la ligne de commande se termine par le chemin complet d'accès à la variable souhaitée,
- ça fonctionne, puisque le système répond autrement que par un message d'erreur.

En effet, si nous nous étions trompés dans le chemin, nous aurions obtenu, soit autre chose que la valeur souhaitée, soit, plus probablement, un message d'erreur :

```
[root@gw mibs]# snmpget -v 1 -c public localhost .1.3.6.1.2.1.1.3.1
Error in packet
Reason: (noSuchName) There is no such variable name in this MIB.
Failed object: system.sysUpTime.1
```

Comme .1.3.6.1.2.1, autrement dit .iso.org.dod.internet.mgmt.mib est le début de chemin le plus souvent employé, il est possible de le sous-entendre de la façon suivante :

```
[root@gw mibs]# snmpget -v 1 -c public localhost 1.3.0
system.sysUpTime.0 = Timeticks: (23690312) 2 days, 17:48:23.12
```

Si le chemin indiqué commence par un point (.1.3.6.1.2.1.1.3.0) il s'agit d'un chemin absolu, s'il ne commence pas par un point (1.3.0), il est considéré comme relatif à ce qui manque, à savoir .1.3.6.1.2.1.

Enfin, il est possible de parler presque anglais :

```
[root@gw mibs]# snmpget -v 1 -c public localhost system.sysUpTime.0
system.sysUpTime.0 = Timeticks: (23721007) 2 days, 17:53:30.07
```

Attention aux lettres majuscules et minuscules. Bien entendu, le chemin complet fonctionne ici aussi, à la condition de connaître parfaitement la syntaxe du chemin :

```
[root@gw mibs]# snmpget -v 1 -c public localhost .iso.org.dod.internet.mgmt.mib-2.system.sysUpTime.0
system.sysUpTime.0 = Timeticks: (23878536) 2 days, 18:19:45.36
```

Amusez-vous un peu avec, et vous constaterez vite que, finalement, c'est plus facile avec les index.

Voyons tout de même l'outil "snmptranslate", sorte de couteau suisse de SNMP, qui permet, par exemple d'obtenir la traduction entre versions numérique et textuelle (et réciproquement) d'un chemin, mais qui permet également de retrouver la forme de l'arbre.

Ici, nous essayons de visualiser la branche qui part de "system" :

```
[root@gw mibs]# snmptranslate -Tp -IR system
+--system(1)
|
+-- -R-- String      sysDescr(1)
```

```

|       Textual Convention: DisplayString
|       Size: 0..255
+--- -R-- ObjID      sysObjectID(2)
+--- -R-- TimeTicks sysUpTime(3)
+--- -RW- String     sysContact(4)
|       Textual Convention: DisplayString
|       Size: 0..255
+--- -RW- String     sysName(5)
|       Textual Convention: DisplayString
|       Size: 0..255
+--- -RW- String     sysLocation(6)
|       Textual Convention: DisplayString
|       Size: 0..255
+--- -R-- INTEGER   sysServices(7)
|       Range: 0..127
+--- -R-- TimeTicks sysORLastChange(8)
|       Textual Convention: TimeStamp
|
+---sysORTable(9)
|
+---sysOREntry(1)
|   Index: sysORIndex
|
+--- ---- INTEGER   sysORIndex(1)
|       Range: 1..2147483647
+--- -R-- ObjID     sysORID(2)
+--- -R-- String    sysORDescr(3)
|       Textual Convention: DisplayString
|       Size: 0..255
+--- -R-- TimeTicks sysORUpTime(4)
|       Textual Convention: TimeStamp

```

En plus de l'arborescence (nom et index numérique), nous obtenons également le type de variable et son état (lecture seule ou lecture/écriture).

Pour en finir à ce niveau avec la MIB et les commandes de base associées sous Linux, disons ceci :

- Les outils ont besoin de connaître la version SNMP utilisée (option -v 1 pour SNMP v1),
- les outils ont besoin de connaître la "communauté" (à voir comme un mot de passe, nous y reviendrons plus bas). Ici la communauté autorisée en lecture s'appelle "public" (option -c public),
- les outils ont besoin de connaître la cible (adresse IP de l'agent SNMP interrogé), dans l'exemple : la machine locale,
- enfin, les commandes ont souvent besoin de savoir à quelle feuille de la MIB on s'intéresse.

Les communautés

Un agent SNMP est plus ou moins finement paramétrable, suivant le système. Il est possible, par exemple de créer des groupes de sécurité qui auront accès en lecture seule, d'autres en lecture/écriture, d'autres encore en lecture seule, mais sur certaines branches seulement...

Chaque groupe devra disposer d'une sorte de mot de passe, appelé "community". En général, la communauté "public" est celle qui a le droit de lecture sur les informations non sensibles.

L'inconvénient majeur est qu'avec SNMP v1, qui est actuellement la seule version vraiment stabilisée et reconnue par tous, ce mot de passe circule en clair sur le réseau, ce qui rend dans ce cas SNMP plus que dangereux.

Les versions suivantes de SNMP corrigent ce problème majeur.

Vérifications sur le réseau

Pour en finir avec cette page, un petit sniff sur le réseau pour vérifier tout ça. Voici la commande utilisée, avec sa réponse :

```
[root@gw mibs]# snmpget -v 1 -c public localhost .iso.org.dod.internet.mgmt.mib-2.system.sysUpTime.0
system.sysUpTime.0 = Timeticks: (24049155) 2 days, 18:48:11.55
```

Et voici ce que le sniffeur récupère. Deux paquets UDP, l'un pour la question :

```

Frame 1 (85 on wire, 85 captured)
  Arrival Time: Jul 22, 2003 12:40:55.435257000
  Time delta from previous packet: 0.000000000 seconds
  Time relative to first packet: 0.000000000 seconds
  Frame Number: 1
  Packet Length: 85 bytes
  Capture Length: 85 bytes
Ethernet II
  Destination: 00:00:00:00:00:00 (00:00:00_00:00:00)
  Source: 00:00:00:00:00:00 (00:00:00_00:00:00)
  Type: IP (0x0800)
Internet Protocol, Src Addr: localhost.localdomain (127.0.0.1), Dst Addr: localhost.localdomain (127.0.0.1)
  Version: 4
  Header length: 20 bytes
  Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
    0000 00.. = Differentiated Services Codepoint: Default (0x00)
    .... ..0. = ECN-Capable Transport (ECT): 0
    .... ...0 = ECN-CE: 0
  Total Length: 71
  Identification: 0x0000
  Flags: 0x04
    .1.. = Don't fragment: Set
    ..0. = More fragments: Not set
  Fragment offset: 0
  Time to live: 64
  Protocol: UDP (0x11)
  Header checksum: 0x3ca4 (correct)
  Source: localhost.localdomain (127.0.0.1)
  Destination: localhost.localdomain (127.0.0.1)
User Datagram Protocol, Src Port: 1867 (1867), Dst Port: snmp (161)
  Source port: 1867 (1867)
  Destination port: snmp (161)
  Length: 51
  Checksum: 0xd027 (correct)
Simple Network Management Protocol
  Version: 1
  Community: public
  PDU type: GET
  Request Id: 0x34339329
  Error Status: NO ERROR
  Error Index: 0
  Object identifier 1: 1.3.6.1.2.1.1.3.0 (SNMPv2-MIB::sysUpTime.0)
  Value: NULL

```

Et l'autre pour la réponse :

```

Frame 2 (89 on wire, 89 captured)
  Arrival Time: Jul 22, 2003 12:40:55.437704000
  Time delta from previous packet: 0.002447000 seconds
  Time relative to first packet: 0.002447000 seconds
  Frame Number: 2
  Packet Length: 89 bytes
  Capture Length: 89 bytes
Ethernet II
  Destination: 00:00:00:00:00:00 (00:00:00_00:00:00)
  Source: 00:00:00:00:00:00 (00:00:00_00:00:00)
  Type: IP (0x0800)
Internet Protocol, Src Addr: localhost.localdomain (127.0.0.1), Dst Addr: localhost.localdomain (127.0.0.1)
  Version: 4
  Header length: 20 bytes
  Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)

```

```
0000 00.. = Differentiated Services Codepoint: Default (0x00)
.... ..0. = ECN-Capable Transport (ECT): 0
.... ...0 = ECN-CE: 0
Total Length: 75
Identification: 0x0000
Flags: 0x04
  .1.. = Don't fragment: Set
  ..0. = More fragments: Not set
Fragment offset: 0
Time to live: 64
Protocol: UDP (0x11)
Header checksum: 0x3ca0 (correct)
Source: localhost.localdomain (127.0.0.1)
Destination: localhost.localdomain (127.0.0.1)
User Datagram Protocol, Src Port: snmp (161), Dst Port: 1867 (1867)
Source port: snmp (161)
Destination port: 1867 (1867)
Length: 55
Checksum: 0x4de4 (correct)
Simple Network Management Protocol
Version: 1
Community: public
PDU type: RESPONSE
Request Id: 0x34339329
Error Status: NO ERROR
Error Index: 0
Object identifier 1: 1.3.6.1.2.1.1.3.0 (SNMPv2-MIB::sysUpTime.0)
Value: Timeticks: (24049155) 2 days, 18:48:11.55
```

Vous avez noté le passage de la communauté en clair ? Ça veut dire qu'il est plus que dangereux d'utiliser ce protocole dans sa version 1 pour manager un système à distance. N'importe qui sur le réseau pouvant récupérer cette information pour ensuite, faire ce qu'il voudra.

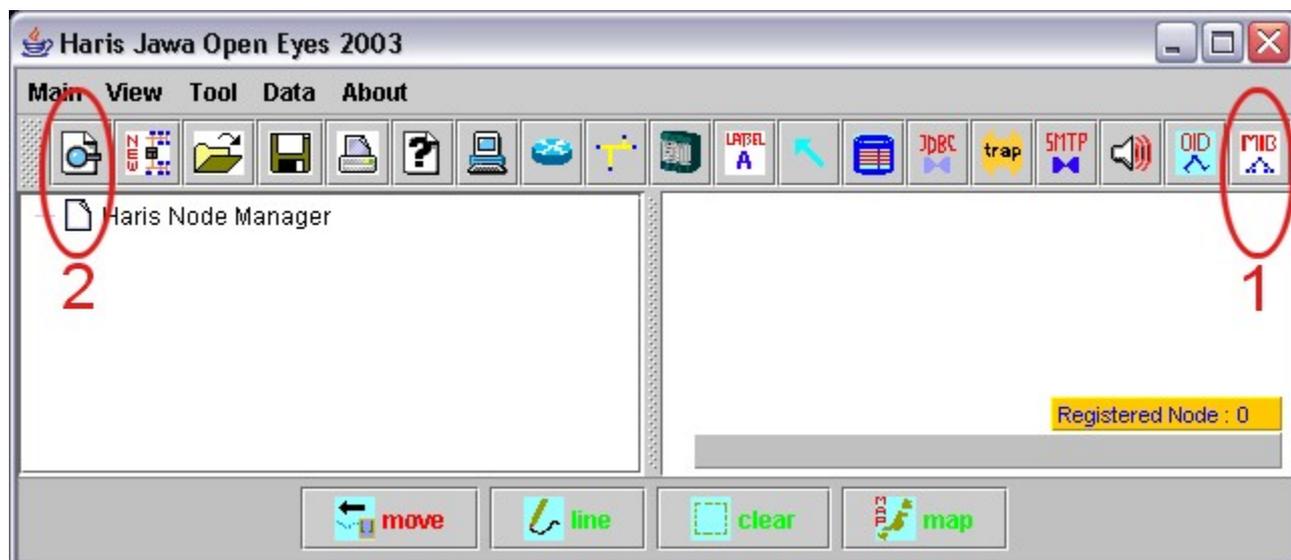
Managers

A la lumière de ce qui a été vu, il est clair que pour être exploitable, ce protocole nécessite d'être manipulé à travers une interface conviviale et ergonomique, encore que les administrateurs Unix, qui sont des spartiates modernes, prennent plaisir à manipuler des lignes de commandes dont la syntaxe reste franchement obscure. De fait, les outils en ligne de commande ont un réel intérêt, parce qu'ils sont facilement manipulables à partir de scripts, ce qui permet de se construire "facilement" des outils sur mesure, mais les "clickodromes", ont aussi du bon...

Le plus connu de ces clickodromes est un outil commercial de chez Hewlett Packard, intitulé "openview". Comme tout logiciel commercial professionnel, son prix en fait une merveille à lui tout seul.

Je n'ai pas trouvé d'équivalent vraiment convainquant en GPL sous Linux. Pour illustrer de façon un peu pratique ce qui a été vu jusqu'ici, nous allons utiliser un outil écrit en Java intitulé "open eyes"³. Pour des raisons beaucoup plus idéologiques que techniques, je n'aime pas trop utiliser Java, mais cet utilitaire offre l'avantage d'une certaine simplicité et peut, avantage de Java, fonctionner sur divers systèmes. Pour l'occasion, il est utilisé sous Windows XP, avec le "Java Runtime Environment"⁴.

Jawa Open Eyes



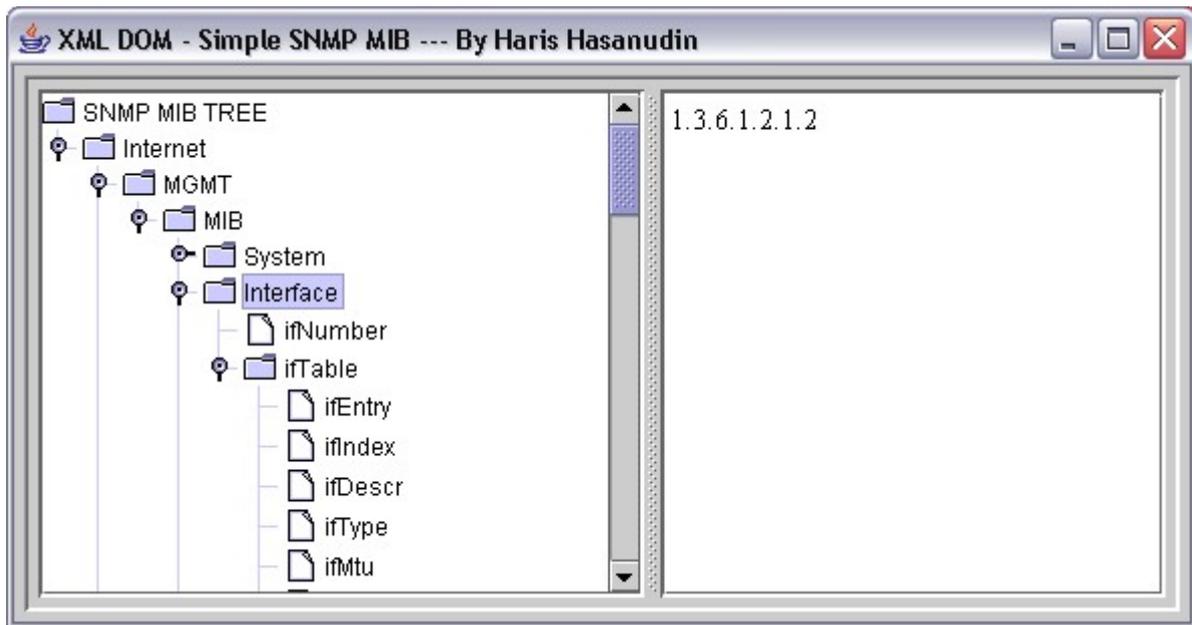
Nous allons d'abord nous intéresser à deux fonctions : un pseudo "mib browser" et l'interrogation d'un noeud IP.

1 : Simple MIB Reference

Ce n'est pas un outil qui va afficher le contenu d'une MIB sur un noeud donné, mais il permet de retrouver la représentation numérique d'une clé de la MIB "générique".

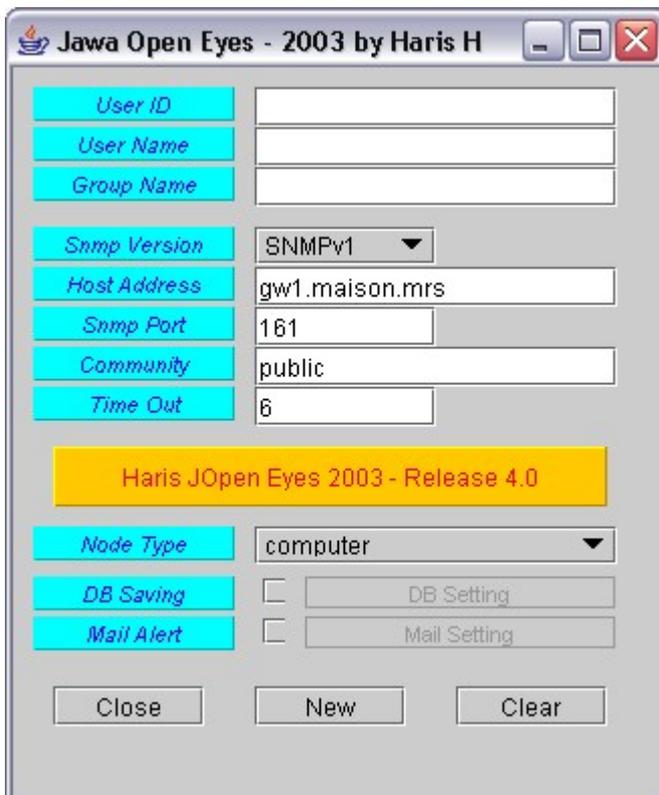
³ Open eyes : <http://sourceforge.net/projects/jopeneyes>

⁴ Java Runtime Environment : <http://java.sun.com/j2se/1.4.2/download.html>



Cet outil, nous allons le voir dans la suite, s'avère bien intéressant...

2: Interrogation d'un noeud

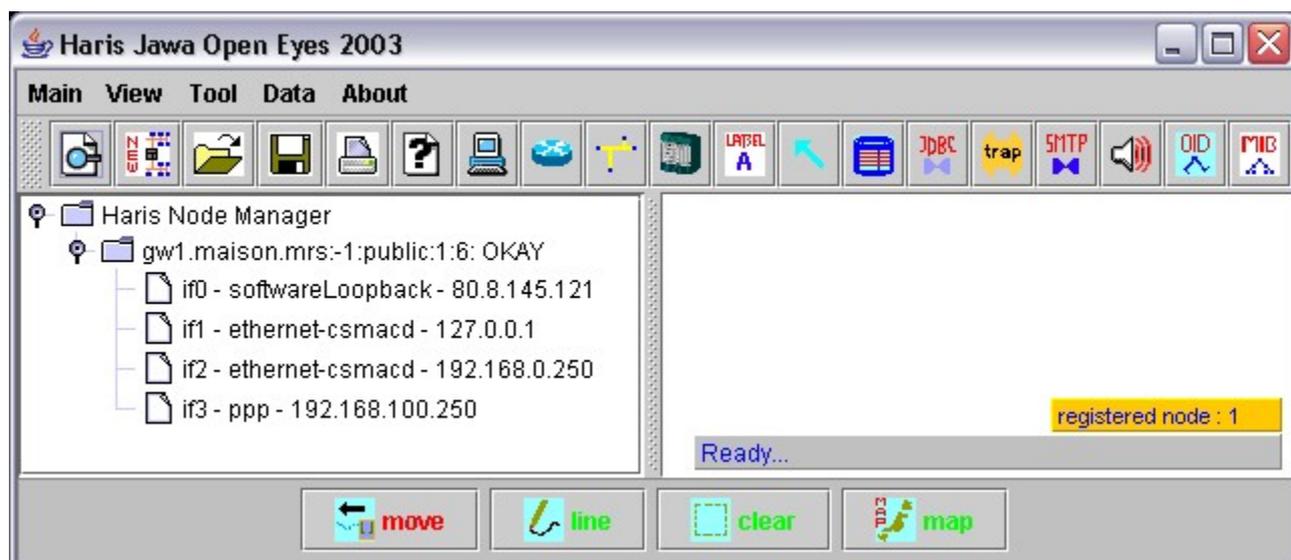


Et voilà le travail :

Voyons ce que cet outil peut nous donner comme informations, si on lui demande d'interroger ma passerelle Linux.

Pour information, elle dispose de quatre interfaces réseau :

- La "loopback" (lo : 127.0.0.1)
- Eth0, sur le réseau local : 192.168.0.250
- Eth1, vers le modem câble : 192.168.100.250
- ppp0, la connexion PPPoE qui, au moment des tests, dispose de l'adresse 80.8.145.121



Comme vous le voyez, c'est tout de même un peu mélangé... Et c'est tant mieux, parce que, lorsque tout fonctionne sans problèmes, on n'est pas obligé de chercher l'erreur ce qui est dommage, parce que chercher l'erreur oblige à chercher à comprendre le processus.

Nous avons donc ici de la chance, nous allons être obligés de chercher à comprendre !

A la recherche du bug

Tant qu'à mettre les mains dans le cambouis, autant utiliser les rudimentaires mais efficaces outils en ligne de commande de Linux, avec la commande "snmpwalk".

Recherche des interfaces : Internet.MGMT.MIB.Interface

Le "browser" de openeyes va nous servir. Nous avons vu que la représentation numérique de cette entrée de la MIB est .1.3.6.1.2.1.2 :

```
[root@gw1 root]# snmpwalk -OX -c public -v 1 localhost .1.3.6.1.2.1.2
interfaces.ifNumber.0 = 4

## Bon, nous avons bien 4 interfaces, et leur index sera :
interfaces.ifTable.ifEntry.ifIndex[1] = 1
interfaces.ifTable.ifEntry.ifIndex[2] = 2
interfaces.ifTable.ifEntry.ifIndex[3] = 3
interfaces.ifTable.ifEntry.ifIndex[4] = 4

## Leur nom :
interfaces.ifTable.ifEntry.ifDescr[1] = lo
interfaces.ifTable.ifEntry.ifDescr[2] = eth0
interfaces.ifTable.ifEntry.ifDescr[3] = eth1
interfaces.ifTable.ifEntry.ifDescr[4] = ppp0

# Leur type :
interfaces.ifTable.ifEntry.ifType[1] = softwareLoopback(24)
interfaces.ifTable.ifEntry.ifType[2] = ethernetCsmacd(6)
interfaces.ifTable.ifEntry.ifType[3] = ethernetCsmacd(6)
interfaces.ifTable.ifEntry.ifType[4] = ppp(23)

## Leur MTU :
interfaces.ifTable.ifEntry.ifMtu[1] = 16436
interfaces.ifTable.ifEntry.ifMtu[2] = 1500
interfaces.ifTable.ifEntry.ifMtu[3] = 1500
interfaces.ifTable.ifEntry.ifMtu[4] = 1492

## etc. Le reste est moins intéressant pour ce qui nous concerne.
```

```

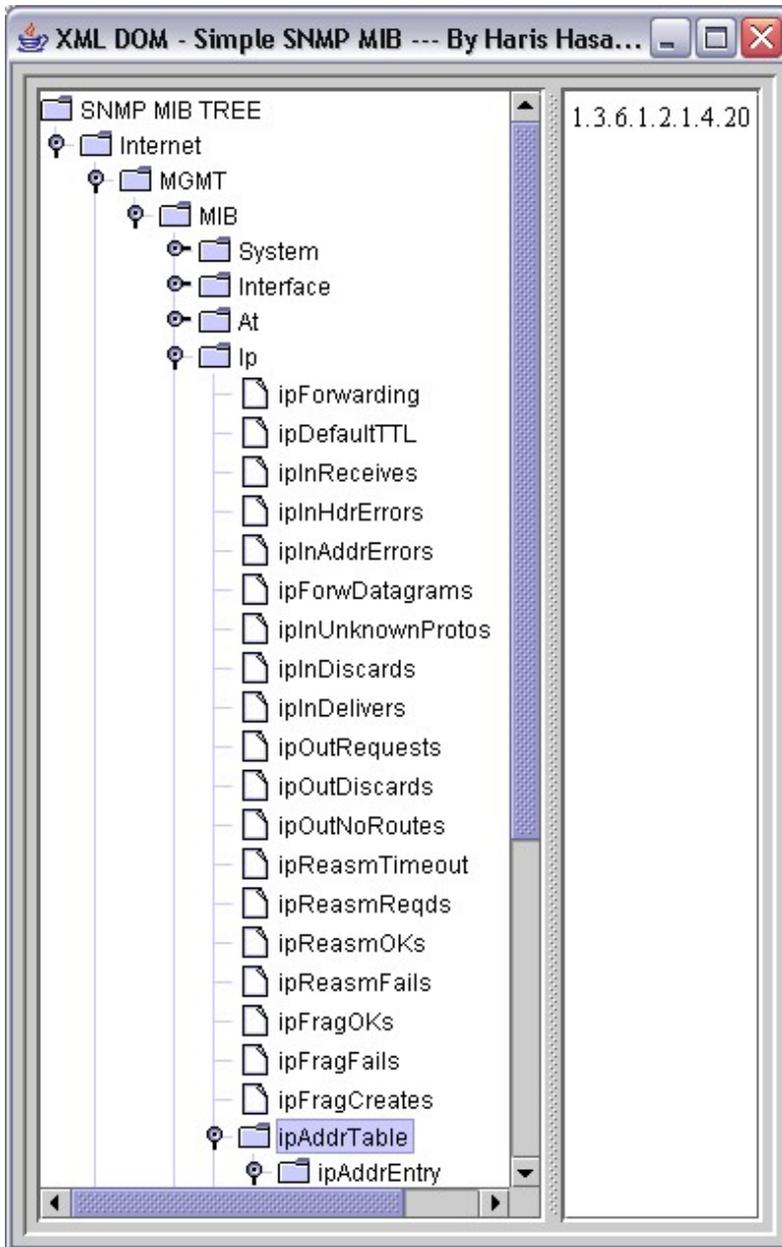
## je vous le laisse juste pour le "fun"...
interfaces.ifTable.ifEntry.ifSpeed[1] = Gauge32: 10000000
interfaces.ifTable.ifEntry.ifSpeed[2] = Gauge32: 10000000
interfaces.ifTable.ifEntry.ifSpeed[3] = Gauge32: 10000000
interfaces.ifTable.ifEntry.ifSpeed[4] = Gauge32: 0
interfaces.ifTable.ifEntry.ifPhysAddress[1] =
interfaces.ifTable.ifEntry.ifPhysAddress[2] = 0:20:18:2c:e:e9
interfaces.ifTable.ifEntry.ifPhysAddress[3] = 0:60:8c:50:f3:3e
interfaces.ifTable.ifEntry.ifPhysAddress[4] =
interfaces.ifTable.ifEntry.ifAdminStatus[1] = up(1)
interfaces.ifTable.ifEntry.ifAdminStatus[2] = up(1)
interfaces.ifTable.ifEntry.ifAdminStatus[3] = up(1)
interfaces.ifTable.ifEntry.ifAdminStatus[4] = up(1)
interfaces.ifTable.ifEntry.ifOperStatus[1] = up(1)
interfaces.ifTable.ifEntry.ifOperStatus[2] = up(1)
interfaces.ifTable.ifEntry.ifOperStatus[3] = up(1)
interfaces.ifTable.ifEntry.ifOperStatus[4] = up(1)
interfaces.ifTable.ifEntry.ifInOctets[1] = Counter32: 1262381
interfaces.ifTable.ifEntry.ifInOctets[2] = Counter32: 76146933
interfaces.ifTable.ifEntry.ifInOctets[3] = Counter32: 340396847
interfaces.ifTable.ifEntry.ifInOctets[4] = Counter32: 35138404
interfaces.ifTable.ifEntry.ifInUcastPkts[1] = Counter32: 10561
interfaces.ifTable.ifEntry.ifInUcastPkts[2] = Counter32: 941771
interfaces.ifTable.ifEntry.ifInUcastPkts[3] = Counter32: 1268973
interfaces.ifTable.ifEntry.ifInUcastPkts[4] = Counter32: 58952
interfaces.ifTable.ifEntry.ifInErrors[1] = Counter32: 0
interfaces.ifTable.ifEntry.ifInErrors[2] = Counter32: 0
interfaces.ifTable.ifEntry.ifInErrors[3] = Counter32: 0
interfaces.ifTable.ifEntry.ifInErrors[4] = Counter32: 0
interfaces.ifTable.ifEntry.ifOutOctets[1] = Counter32: 1264163
interfaces.ifTable.ifEntry.ifOutOctets[2] = Counter32: 242592651
interfaces.ifTable.ifEntry.ifOutOctets[3] = Counter32: 85618617
interfaces.ifTable.ifEntry.ifOutOctets[4] = Counter32: 3647223
interfaces.ifTable.ifEntry.ifOutUcastPkts[1] = Counter32: 10585
interfaces.ifTable.ifEntry.ifOutUcastPkts[2] = Counter32: 1079693
interfaces.ifTable.ifEntry.ifOutUcastPkts[3] = Counter32: 942638
interfaces.ifTable.ifEntry.ifOutUcastPkts[4] = Counter32: 55874
interfaces.ifTable.ifEntry.ifOutDiscards[1] = Counter32: 0
interfaces.ifTable.ifEntry.ifOutDiscards[2] = Counter32: 0
interfaces.ifTable.ifEntry.ifOutDiscards[3] = Counter32: 0
interfaces.ifTable.ifEntry.ifOutDiscards[4] = Counter32: 0
interfaces.ifTable.ifEntry.ifOutErrors[1] = Counter32: 0
interfaces.ifTable.ifEntry.ifOutErrors[2] = Counter32: 1
interfaces.ifTable.ifEntry.ifOutErrors[3] = Counter32: 0
interfaces.ifTable.ifEntry.ifOutErrors[4] = Counter32: 0
interfaces.ifTable.ifEntry.ifOutQLen[1] = Gauge32: 0
interfaces.ifTable.ifEntry.ifOutQLen[2] = Gauge32: 0
interfaces.ifTable.ifEntry.ifOutQLen[3] = Gauge32: 0
interfaces.ifTable.ifEntry.ifOutQLen[4] = Gauge32: 0
interfaces.ifTable.ifEntry.ifSpecific[1] = OID: .ccitt.zeroDotZero
interfaces.ifTable.ifEntry.ifSpecific[2] = OID: .ccitt.zeroDotZero
interfaces.ifTable.ifEntry.ifSpecific[3] = OID: .ccitt.zeroDotZero
interfaces.ifTable.ifEntry.ifSpecific[4] = OID: .ccitt.zeroDotZero

```

Bien. Si nous récapitulons, nous pouvons établir le tableau suivant :

Index	Nom	Type
1	lo	softwareLoopback(24)
2	eth0	ethernetCsmacd(6)
3	eth1	ethernetCsmacd(6)
4	ppp0	ppp(23)

Jusque là, c'est bien dans le même ordre que ce qu'affiche openeyes.

Recherche des Ips : Internet.MGMT.MIB.Ip.ipAddrTable

Petit retour dans le "browser de mib", pour rechercher où pourraient se trouver les adresses IP des interfaces.

Il y a des chances que ce soit dans Internet.MGMT.MIB.Ip.ipAddrTable, c'est à dire dans 1.3.6.1.2.1.4.20

Encore un petit coup de snmpwalk :

```
[root@gw1 root]# snmpwalk -OX -c public -v 1 localhost .1.3.6.1.2.1.4.20

# Les 4 adresses IP
ip.ipAddrTable.ipAddrEntry.ipAdEntAddr[80.8.145.121] = IpAddress: 80.8.145.121
ip.ipAddrTable.ipAddrEntry.ipAdEntAddr[127.0.0.1] = IpAddress: 127.0.0.1
ip.ipAddrTable.ipAddrEntry.ipAdEntAddr[192.168.0.250] = IpAddress: 192.168.0.250
ip.ipAddrTable.ipAddrEntry.ipAdEntAddr[192.168.100.250] = IpAddress: 192.168.100.250

## Et l'index qui correspond
ip.ipAddrTable.ipAddrEntry.ipAdEntIfIndex[80.8.145.121] = 4
ip.ipAddrTable.ipAddrEntry.ipAdEntIfIndex[127.0.0.1] = 1
ip.ipAddrTable.ipAddrEntry.ipAdEntIfIndex[192.168.0.250] = 2
ip.ipAddrTable.ipAddrEntry.ipAdEntIfIndex[192.168.100.250] = 3

## Le reste, quoi qu'intéressant, n'est d'aucune utilité pour
## la recherche du bug.
## Je vous le laisse encore pour le "fun"
```

```
ip.ipAddrTable.ipAddrEntry.ipAdEntNetMask[80.8.145.121] = IPAddress: 255.255.255.255
ip.ipAddrTable.ipAddrEntry.ipAdEntNetMask[127.0.0.1] = IPAddress: 255.0.0.0
ip.ipAddrTable.ipAddrEntry.ipAdEntNetMask[192.168.0.250] = IPAddress: 255.255.255.0
ip.ipAddrTable.ipAddrEntry.ipAdEntNetMask[192.168.100.250] = IPAddress: 255.255.255.0
ip.ipAddrTable.ipAddrEntry.ipAdEntBcastAddr[80.8.145.121] = 0
ip.ipAddrTable.ipAddrEntry.ipAdEntBcastAddr[127.0.0.1] = 1
ip.ipAddrTable.ipAddrEntry.ipAdEntBcastAddr[192.168.0.250] = 1
ip.ipAddrTable.ipAddrEntry.ipAdEntBcastAddr[192.168.100.250] = 1
```

Et voilà... L'auteur de openeyes n'avait pas prévu que les IPs pouvaient être données dans un ordre différent de celui des interfaces !

Les interfaces dans l'ordre 1, 2, 3, 4 et les IPs dans l'ordre 4, 1,2, 3

Si nous complétons le tableau précédent correctement, c'est à dire en tenant compte des index, nous obtenons :

Index	Nom	Type	Adresse IP
1	lo	softwareLoopback(24)	127.0.0.1
2	eth0	ethernetCsmacd(6)	192.168.0.250
3	eth1	ethernetCsmacd(6)	192.168.100.250
4	ppp0	ppp(23)	80.8.145.121

Ce qui est parfaitement juste. Openeye affiche les IP dans l'ordre où elles arrivent, sans s'occuper des index qui sont pourtant donnés juste après, erreur fatale !

Conclusions

Ce petit exemple nous a forcé à regarder de près comment les informations sont organisées dans la MIB et comment on peut y accéder avec des outils simples.

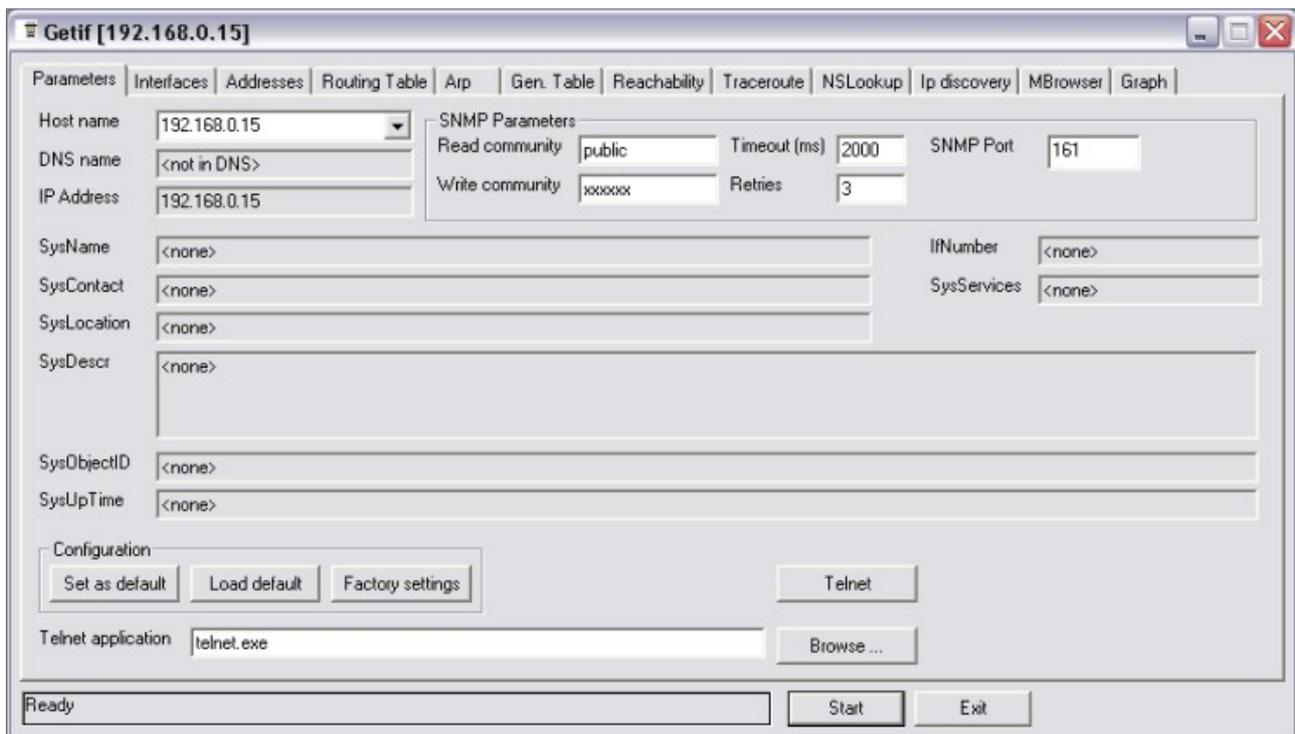
Nous pourrions aller encore un peu plus loin en utilisant la commande `snmpget` qui, plutôt que d'afficher toute une branche de la MIB, se bornera à une seule feuille, je vous laisse jouer avec si le coeur vous en dit.

A la lueur de ce que nous avons vu, il est clair que pour être vraiment utile, SNMP doit être manipulé à travers un outil apte à lire périodiquement certaines informations, à en tirer des graphes, des statistiques etc. Dans la dernière partie de cet exposé, nous verrons MRTG, très souvent utilisé pour afficher la charge d'interfaces réseau.

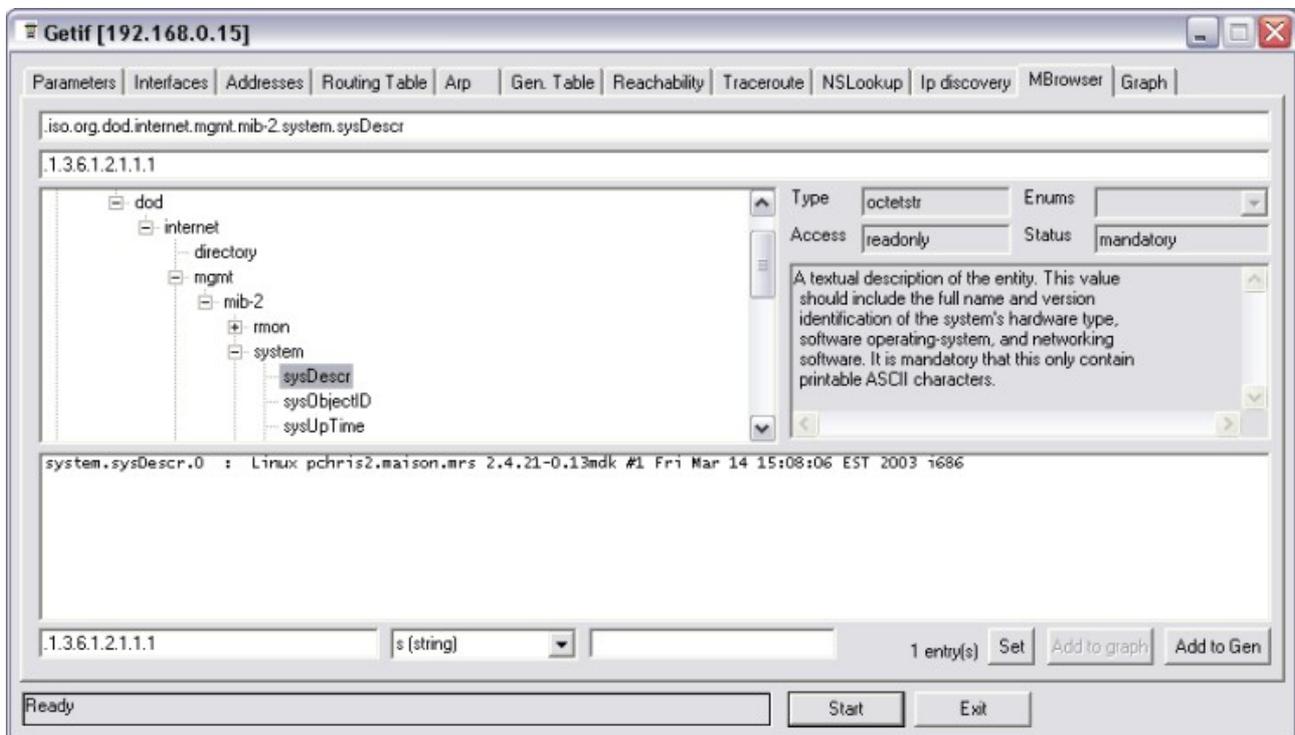
Getif

Les curieux peuvent faire un tour sur cette page⁵ qui présente un outil gratuit sous Windows : "getif". Il a l'apparence suivante :

⁵ Getif : <http://www.wtcs.org/snmp4tpc/getif.htm>



Après avoir indiqué l'adresse IP de la machine à observer, ainsi que la (les) communauté(s) en lecture (et en écriture), on clique sur start. On peut voir pas mal de choses avec Getif, mais ce qui reste le plus intéressant pour nous est le "mib browser" :



Avec lequel nous pourrons faire les mêmes choses qu'avec snmpwalk, snmpget, snmpset et snmptranslate, d'une façon un petit peu plus ergonomique. Je n'ai pas trouvé sous Linux d'équivalent à cet utilitaire.

MRTG intro

Qu'est-ce ?

Multi Router Traffic Grapher

Le but de ce logiciel est de construire des pages HTML avec des jolis graphes, représentant diverses choses que l'on extrait généralement d'une machine via SNMP.

De quoi a-t-on besoin ?

MRTG se contente de construire des pages HTML. Il vous faudra donc disposer de plusieurs choses :

- Un agent SNMP bien sûr, sur l'hôte dont vous voulez auditer les paramètres,
- un serveur HTTP (Apache, par exemple) quelque part sur le réseau. C'est dessus que l'on installera MRTG,
- quelques paquetages de perl. Mais si, comme nous allons le faire ici, MRTG est installé depuis un paquetage RPM, les dépendances seront gérées automatiquement.

La manip proposée

Commençons simplement. Sur une machine Mandrake 9.1 équipée d'une seule interface réseau, nous avons déjà un serveur Apache. Nous allons installer le paquetage MRTG, avec les outils Mandrake habituels.

Nous configurerons alors MRTG pour qu'il affiche le trafic généré sur l'unique interface réseau.

Nous verrons ensuite comment aller plus loin. LA machine sur laquelle la manip est faite s'appelle pchris2.maison.mrs et dispose de l'adresse 192.168.0.15.

Installation de MRTG

Le paquetage

MRTG ne figure pas sur les CD ROM de la distribution. Il faut aller le chercher, par exemple, sur <http://fr2.rpmfind.net>. A l'heure où je rédige ces lignes, le paquetage disponible est mrtg-2.9.27-1mdk.i586.rpm. Voyez sur le site les dépendances et téléchargez et installez ce qui vous est nécessaire.

Le paquetage installe une somme impressionnante de fichiers. Ceux qui sont les plus remarquables dans un premier temps sont dans :

- /var/www/html/
C'est le répertoire par défaut du serveur APACHE. La paquetage ajoute un répertoire nommé mrtg, dans lequel vous trouverez pas mal de documentation.

- /usr/bin/

Les exécutable. Nous y trouverons :

- mrtg lui-même, nous verrons quoi en faire,
- cfmaker, un script qui nous permettra de démarrer,
- indexmaker, également utile pour démarrer.

Parce que si vous croyez qu'après avoir installé le paquetage, vous allez tout de suite voir de beaux graphes bien colorés, vous allez être déçus...

A part l'installation des fichiers nécessaires, le paquetage s'est contenté d'ajouter dans /etc/crontab une ligne du genre :

```
0-59/5 * * * * root /usr/bin/mrtg /var/www/html/mrtg/mrtg.cfg
```

On n'est pas là pour gloser de crond. Si vous ne comprenez pas le sens de cette ligne, sachez qu'elle va faire que mrtg sera exécuté toutes les 5 minutes, en utilisant la configuration décrite dans /var/www/html/mrtg/mrtg.cfg.

Ce fichier existe bien, certes, mais il est vide. Plutôt, il ne contient que des commentaires, qui ne serviront à rien pour mrtg. Il nous faudra donc construire un fichier de configuration adapté à ce que l'on veut visualiser.

Construire une configuration

Comme MRTG est bien fait, le script cfmaker va nous aider à construire cette configuration.

Comme MRTG est très bien fait, il nous suffit d'aller avec notre butineur sur le site web de notre machine (celle où nous installons MRTG) et de faire :

<http://pchris2.maison.mrs/mrtg/cfmaker.html>, dans notre exemple. (Sur une distribution Debian, la documentation HTML est placée dans /usr/share/doc/mrtg/html).

Lisez bien cette page. Si vous la comprenez du premier coup, alors vous êtes un **vrai** informaticien...

Sinon, on va essayer de s'en sortir quand même. Mais avant de commencer, il faut que je vous confesse quelque chose. Ma machine pchris2 dispose de deux cartes Ethernet, mais une seule est active. A preuve :

```
[root@pchris2 root]# ifconfig
eth0 Lien encap:Ethernet HWaddr 00:05:5D:48:2B:84
inet adr:192.168.0.15 Bcast:192.168.0.255 Masque:255.255.255.0
UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
RX packets:234851 errors:0 dropped:0 overruns:0 frame:0
TX packets:1692 errors:0 dropped:0 overruns:0 carrier:0
collisions:17 lg file transmission:100
RX bytes:58112272 (55.4 Mb) TX bytes:454788 (444.1 Kb)
Interruption:10 Adresse de base:0xe000

lo Lien encap:Boucle locale
inet adr:127.0.0.1 Masque:255.0.0.0
UP LOOPBACK RUNNING MTU:16436 Metric:1
RX packets:684 errors:0 dropped:0 overruns:0 frame:0
TX packets:684 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 lg file transmission:0
RX bytes:150267 (146.7 Kb) TX bytes:150267 (146.7 Kb)
```

Mais SNMP va voir la seconde et va indiquer sa présence, tout en indiquant qu'elle est "down".

Bien. Premier essai simplissime, avec cfmaker. Comme nous allons le voir tout de suite, ce script va se contenter de sortir une configuration sur l'écran :

```
## Nous nous contentons d'invoquer cfmaker en lui indiquant la "communauté"@l'hôte"
[root@pchris2 root]# cfmaker public@localhost

## Et l'affichage de donner dans un premier temps les infos SNMP:
--base: Get Device Info on public@localhost:
--base: Vendor Id:
--base: Populating confcache
--snpo: confcache public@localhost: Descr lo --> 1
--snpo: confcache public@localhost: Descr eth0 --> 2
--snpo: confcache public@localhost: Descr eth1 --> 3
--snpo: confcache public@localhost: Ip 127.0.0.1 --> 1
--snpo: confcache public@localhost: Ip 192.168.0.15 --> 2
--snpo: confcache public@localhost: Type 24 --> 1
--snpo: confcache public@localhost: Type 6 --> 2
--snpo: confcache public@localhost: Type 6 --> 3 (duplicate)
--snpo: confcache public@localhost: Eth --> 1
--snpo: confcache public@localhost: Eth 00-05-5d-48-2b-84 --> 2
--snpo: confcache public@localhost: Eth 00-20-af-07-1a-3d --> 3
--base: Get Interface Info
--base: Walking ifIndex
--base: Walking ifType
--base: Walking ifSpeed
--base: Walking ifAdminStatus
--base: Walking ifOperStatus

## Puis la configuration proprement dite :
# Created by
# /usr/bin/cfmaker public@localhost

### Global Config Options

# for UNIX
# WorkDir: /home/http/mrtg

# or for NT
# WorkDir: c:\mrtgdata

### Global Defaults

# to get bits instead of bytes and graphs growing to the right
# Options[_]: growright, bits

#####
# System: pchris2.maison.mrs
# Description: Linux pchris2.maison.mrs 2.4.21-0.13mdk #1 Fri Mar 14 15:08:06 EST 2003 i686
# Contact: Root <root@localhost> (configure /etc/snmp/snmp.local.conf)
# Location: Unknown (edit /etc/snmp/snmpd.conf)
#####

### Interface 1 >> Descr: 'lo' | Name: '' | Ip: '127.0.0.1' | Eth: '' ###
### The following interface is commented out because:
### * it is a Software Loopback interface
#
# Target[localhost_1]: 1:public@localhost:
# SetEnv[localhost_1]: MRTG_INT_IP="127.0.0.1" MRTG_INT_DESCR="lo"
# MaxBytes[localhost_1]: 1250000
# Title[localhost_1]: Traffic Analysis for 1 -- pchris2.maison.mrs
# PageTop[localhost_1]: <H1>Traffic Analysis for 1 -- pchris2.maison.mrs</H1>
# <TABLE>
# <TR><TD>System:</TD> <TD>pchris2.maison.mrs in Unknown (edit /etc/snmp/snmpd.conf)</TD></TR>
# <TR><TD>Maintainer:</TD> <TD>Root &lt;root@localhost&gt; (configure
# /etc/snmp/snmp.local.conf)</TD></TR>
# <TR><TD>Description:</TD><TD>lo </TD></TR>
# <TR><TD>ifType:</TD> <TD>softwareLoopback (24)</TD></TR>
# <TR><TD>ifName:</TD> <TD></TD></TR>
# <TR><TD>Max Speed:</TD> <TD>1250.0 kBytes/s</TD></TR>
# <TR><TD>Ip:</TD> <TD>127.0.0.1 (localhost)</TD></TR>
# </TABLE>
```

```
### Interface 2 >> Descr: 'eth0' | Name: '' | Ip: '192.168.0.15' | Eth: '00-05-5d-48-2b-84' ###
Target[localhost_2]: 2:public@localhost:
SetEnv[localhost_2]: MRTG_INT_IP="192.168.0.15" MRTG_INT_DESCR="eth0"
MaxBytes[localhost_2]: 1250000
Title[localhost_2]: Traffic Analysis for 2 -- pchris2.maison.mrs
PageTop[localhost_2]: <H1>Traffic Analysis for 2 -- pchris2.maison.mrs</H1>
<TABLE>
<TR><TD>System:</TD> <TD>pchris2.maison.mrs in Unknown (edit /etc/snmp/snmpd.conf)</TD></TR>
<TR><TD>Maintainer:</TD> <TD>Root &lt;root@localhost&gt; (configure
/etc/snmp/snmp.local.conf)</TD></TR>
<TR><TD>Description:</TD><TD>eth0 </TD></TR>
<TR><TD>ifType:</TD> <TD>ethernetCsmacd (6)</TD></TR>
<TR><TD>ifName:</TD> <TD></TD></TR>
<TR><TD>Max Speed:</TD> <TD>1250.0 kBytes/s</TD></TR>
<TR><TD>Ip:</TD> <TD>192.168.0.15 (pchris2.maison.mrs)</TD></TR>
</TABLE>

### Interface 3 >> Descr: 'eth1' | Name: '' | Ip: '' | Eth: '00-20-af-07-1a-3d' ###
### The following interface is commented out because:
### * it is administratively DOWN
### * it is operationally DOWN
#
# Target[localhost_3]: 3:public@localhost:
# SetEnv[localhost_3]: MRTG_INT_IP="" MRTG_INT_DESCR="eth1"
# MaxBytes[localhost_3]: 1250000
# Title[localhost_3]: Traffic Analysis for 3 -- pchris2.maison.mrs
# PageTop[localhost_3]: <H1>Traffic Analysis for 3 -- pchris2.maison.mrs</H1>
# <TABLE>
# <TR><TD>System:</TD> <TD>pchris2.maison.mrs in Unknown (edit /etc/snmp/snmpd.conf)</TD></TR>
# <TR><TD>Maintainer:</TD> <TD>Root &lt;root@localhost&gt; (configure
/etc/snmp/snmp.local.conf)</TD></TR>
# <TR><TD>Description:</TD><TD>eth1 </TD></TR>
# <TR><TD>ifType:</TD> <TD>ethernetCsmacd (6)</TD></TR>
# <TR><TD>ifName:</TD> <TD></TD></TR>
# <TR><TD>Max Speed:</TD> <TD>1250.0 kBytes/s</TD></TR>
# </TABLE>
```

Je ne détaille pas ce fichier, nous allons essayer de faire mieux en utilisant les options de cfmaker.

Déjà, nous constatons que le répertoire de travail de MRTG est resté en commentaire. Ce répertoire nécessite peut-être quelques mots d'explication.

MRTG peut générer une multitude de pages différentes. Nous pouvons par exemple auditer plusieurs machines du réseau à partir de la même installation de MRTG. Il suffit de le lancer plusieurs fois, avec des fichiers de configuration différents ou, plus simplement, définir dans un même fichier de configuration, des représentations de paramètres issus de plusieurs machines. Nous allons ici créer un répertoire pour la machine locale : /var/www/html/mrtg/local.

```
[root@pchris2 root]# md /var/www/html/mrtg/local
```

Et nous allons utiliser cfmaker, avec l'option qui permet de spécifier ce répertoire de travail :

```
[root@pchris2 root]# cfmaker --global 'WorkDir: /var/www/html/mrtg/local' public@localhost
--base: Get Device Info on public@localhost:

## Le début ne change pas...
--base: Vendor Id:
--base: Populating confcache
...

# Created by
# /usr/bin/cfmaker --global 'WorkDir: /var/www/html/mrtg/local' public@localhost

### Global Config Options

# for UNIX
# WorkDir: /home/http/mrtg
```

```
# or for NT
# WorkDir: c:\mrtgdata

### Global Defaults

# to get bits instead of bytes and graphs growing to the right
# Options[_]: growright, bits

WorkDir: /var/www/html/mrtg/local

#####
# System: pchris2.maison.mrs
## Le reste ne change pas non plus.
...
```

Mais nous pouvons constater que la directive "WorkDir" figure maintenant dans la configuration. A ce stade, nous pourrions déjà utiliser ce fichier de configuration, qui montrerait le trafic présent sur eth0, mais ça donnerait des pages pas très belles :

- En anglais,
- Avec des descriptions d'interfaces pas très parlantes :
Traffic Analysis for 2 -- pchris2.maison.mrs...

Nous allons essayer de faire mieux.

```
[root@pchris2 root]# cfmaker --global 'WorkDir: /var/www/html/mrtg/local' --ifdesc=descr
public@localhost
...
### Interface 2 >> Descr: 'eth0' | Name: '' | Ip: '192.168.0.15' | Eth: '00-05-5d-48-2b-84' ###

Target[localhost_2]: 2:public@localhost:
SetEnv[localhost_2]: MRTG_INT_IP="192.168.0.15" MRTG_INT_DESCR="eth0"
MaxBytes[localhost_2]: 1250000
Title[localhost_2]: eth0 -- pchris2.maison.mrs
PageTop[localhost_2]: <H1>eth0 -- pchris2.maison.mrs</H1>
<TABLE>
<TR><TD>System:</TD> <TD>pchris2.maison.mrs in Unknown (edit /etc/snmp/snmpd.conf)</TD></TR>
<TR><TD>Maintainer:</TD> <TD>Root &lt;root@localhost>; (configure
/etc/snmp/snmp.local.conf)</TD></TR>
<TR><TD>Description:</TD><TD>eth0 </TD></TR>
<TR><TD>ifType:</TD> <TD>ethernetCsmacd (6)</TD></TR>
<TR><TD>ifName:</TD> <TD></TD></TR>
<TR><TD>Max Speed:</TD> <TD>1250.0 kBytes/s</TD></TR>
<TR><TD>Ip:</TD> <TD>192.168.0.15 (pchris2.maison.mrs)</TD></TR>
</TABLE>
...

Title[localhost_2]: eth0 -- pchris2.maison.mrs
```

C'est tout de même plus compréhensible que :

```
Title[localhost_2]: Traffic Analysis for 2 -- pchris2.maison.mrs
```

Bien. Pour finir, nous allons dire à cfmaker que nous voulons du français et aussi un comptage en bits par seconde. En effet, par défaut, le comptage se fait en octets par seconde.

Attention, l'astuce de l'anti-slash (\) permet de saisir une ligne de commande longue en plusieurs morceaux :

```
[root@pchris2 root]# cfmaker \
> --global 'WorkDir: /var/www/html/mrtg/local' \
> --global 'Language: french' \
> --global 'Options[_]: bits,growright' \
> --ifdesc=descr public@localhost
...
WorkDir: /var/www/html/mrtg/local
Language: french
```

```
Options[_]: bits,growright
...
```

Notez bien qu'il est parfaitement possible de reprendre à la main le fichier de configuration, pour arriver au même résultat. Cette méthode des options n'est vraiment utile que si vous devez créer des fichiers de configuration du même style pour de nombreux noeuds de votre réseau.

Parce que bien entendu, je peux créer sur pchris2 des pages d'audit pour ma passerelle qui s'appelle gw1.maison.mrs :

```
[root@pchris2 root]# cfgmaker public@gw1.maison.mrs
--base: Get Device Info on public@gw1.maison.mrs:
--base: Vendor Id:
--base: Populating confcache
--snpo: confcache public@gw1.maison.mrs: Descr lo --> 1
--snpo: confcache public@gw1.maison.mrs: Descr eth0 --> 2
--snpo: confcache public@gw1.maison.mrs: Descr eth1 --> 3
--snpo: confcache public@gw1.maison.mrs: Descr ppp0 --> 4
--snpo: confcache public@gw1.maison.mrs: Ip 80.8.146.171 --> 4
--snpo: confcache public@gw1.maison.mrs: Ip 127.0.0.1 --> 1
--snpo: confcache public@gw1.maison.mrs: Ip 192.168.0.250 --> 2
--snpo: confcache public@gw1.maison.mrs: Ip 192.168.100.250 --> 3
--snpo: confcache public@gw1.maison.mrs: Type 24 --> 1
--snpo: confcache public@gw1.maison.mrs: Type 6 --> 2
--snpo: confcache public@gw1.maison.mrs: Type 6 --> 3 (duplicate)
--snpo: confcache public@gw1.maison.mrs: Type 23 --> 4
--snpo: confcache public@gw1.maison.mrs: Eth --> 1
--snpo: confcache public@gw1.maison.mrs: Eth 00-20-18-2c-0e-e9 --> 2
--snpo: confcache public@gw1.maison.mrs: Eth 00-60-8c-50-f3-3e --> 3
--snpo: confcache public@gw1.maison.mrs: Eth --> 4 (duplicate)
...
```

Bien. Revenons à notre cas local et finissons par créer le fichier de configuration dans /var/www/html/mrtg/local, nous l'appellerons local.cfg :

```
[root@pchris2 root]# cfgmaker \
> --global 'WorkDir: /var/www/html/mrtg/local' \
> --global 'Language: french' \
> --global 'Options[_]: bits,growright' \
> --ifdesc=descr public@localhost \
> --output /var/www/html/mrtg/local/local.cfg

--base: Get Device Info on public@localhost:
--base: Vendor Id:
--base: Populating confcache
--snpo: confcache public@localhost: Descr lo --> 1
--snpo: confcache public@localhost: Descr eth0 --> 2
--snpo: confcache public@localhost: Descr eth1 --> 3
--snpo: confcache public@localhost: Ip 127.0.0.1 --> 1
--snpo: confcache public@localhost: Ip 192.168.0.15 --> 2
--snpo: confcache public@localhost: Type 24 --> 1
--snpo: confcache public@localhost: Type 6 --> 2
--snpo: confcache public@localhost: Type 6 --> 3 (duplicate)
--snpo: confcache public@localhost: Eth --> 1
--snpo: confcache public@localhost: Eth 00-05-5d-48-2b-84 --> 2
--snpo: confcache public@localhost: Eth 00-20-af-07-1a-3d --> 3
--base: Get Interface Info
--base: Walking ifIndex
--base: Walking ifType
--base: Walking ifSpeed
--base: Walking ifAdminStatus
--base: Walking ifOperStatus
--base: Writing /var/www/html/mrtg/local/local.cfg
```

Nous pouvons vérifier que ce fichier existe bien et qu'il ressemble à ce que nous avons vu. Il nous reste au moins une chose à faire, si vous avez suivi, il faut aller modifier la ligne dans la crontab comme suit :

```
0-59/5 * * * * root /usr/bin/mrtg /var/www/html/mrtg/local/local.cfg
```

Pour que MRTG s'exécute avec le bon fichier de configuration.

Après avoir attendu au moins 5 minutes (si vous êtes pressé, vous pouvez toujours utiliser la commande `mrtg` manuellement dans une console), nous trouvons dans le répertoire `/var/www/html/mrtg/local` :

```
[root@pchris2 root]# cd /var/www/html/mrtg/local
[root@pchris2 local]# ls -l
total 140
-rw-r--r-- 1 root root 3608 sep 6 20:14 local.cfg
-rw-r--r-- 1 root root 1771 sep 7 09:30 localhost_2-day.png
-rw-r--r-- 1 root root 8907 sep 7 09:30 localhost_2.html
-rw-r--r-- 1 root root 49102 sep 7 09:30 localhost_2.log
-rw-r--r-- 1 root root 1481 sep 7 09:00 localhost_2-month.png
-rw-r--r-- 1 root root 49113 sep 7 09:25 localhost_2.old
-rw-r--r-- 1 root root 1621 sep 7 09:00 localhost_2-week.png
-rw-r--r-- 1 root root 1679 sep 6 20:20 localhost_2-year.png
-rw-r--r-- 1 root root 0 sep 7 09:30 local.ok
-rw-r--r-- 1 root root 538 août 11 2002 mrtg-l.png
-rw-r--r-- 1 root root 414 août 11 2002 mrtg-m.png
-rw-r--r-- 1 root root 1759 août 11 2002 mrtg-r.png
```

Les trois derniers fichiers sont des icônes qui apparaissent en bas des pages générées par MRTG. Il faut les recopier manuellement depuis le répertoire `/var/www/html/mrtg`.

Il me suffit d'aller maintenant sur http://pchris2.maison.mrs/mrtg/local/localhost_2.thml⁶ pour voir le résultat.

Remarques

- Ces graphes ne sont pas parlants. C'est normal, pour qu'ils présentent un intérêt, il faudrait que la machine tourne 24h/24 pendant plusieurs jours. MRTG serait intéressant, par exemple sur la passerelle, qui dispose de plusieurs interfaces réseau et qui tourne effectivement 24/24, c'est du reste ce que nous allons faire plus loin,
- il y a quelques mots anglais qui traînent dans la description du système. C'est assez simple d'y remédier, en éditant à la main le fichier `local.cfg`,
- les informations concernant la "System" et le "maintainer" sont issues de SNMP. Il est possible d'agir sur ces valeurs en modifiant `snmp.conf`,
- on aimerait bien que dans le répertoire local, il y ait une page d'index, surtout si l'on dispose de plusieurs interfaces. En effet, MRTG va créer une page par interface active. Par chance, MRTG est fourni avec un script qui va réaliser cette page d'index, ce que nous allons voir tout de suite.

Une jolie page d'index pour MRTG

Le script `indexmaker` est là pour ça.

```
[root@pchris2 local]# indexmaker /var/www/html/mrtg/local/local.cfg
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<HTML>
<HEAD>
<TITLE>MRTG Index Page</TITLE>
<META HTTP-EQUIV="Refresh" CONTENT="300">
<META HTTP-EQUIV="Cache-Control" content="no-cache">
<META HTTP-EQUIV="Pragma" CONTENT="no-cache">
<META HTTP-EQUIV="Expires" CONTENT="Sun, 07 Sep 2003 08:06:57 GMT">
```

⁶ Voir les ressources, p 46

```

</HEAD>

<BODY bgcolor="#ffffff" text="#000000" link="#000000" vlink="#000000" alink="#000000">
<!-- commandline was: /usr/bin/indexmaker /var/www/html/mrtg/local/local.cfg -->

<H1>MRTG Index Page</H1>

<TABLE BORDER=0 CELLPADDING=0 CELLSPACING=10>
<TR>
<TD><DIV><B>eth0 -- pchris2.maison.mrs</B></DIV>
<DIV><A HREF="localhost_2.html">
<IMG BORDER=1 ALT="localhost_2 Traffic Graph" SRC="localhost_2-day.png"></A><BR>
<SMALL><!--#flastmod file="localhost_2.html" --></SMALL></DIV>
</TD></TR>
</TABLE>

<BR>
<TABLE BORDER=0 CELLSPACING=0 CELLPADDING=0>
<TR>
<TD WIDTH=63><A
  HREF="http://www.ee.ethz.ch/~oetiker/webtools/mrtg/"><IMG
  BORDER=0 SRC="mrtg-l.png" WIDTH=63 HEIGHT=25 ALT="MRTG"></A></TD>
<TD WIDTH=25><A
  HREF="http://www.ee.ethz.ch/~oetiker/webtools/mrtg/"><IMG
  BORDER=0 SRC="mrtg-m.png" WIDTH=25 HEIGHT=25 ALT=""></A></TD>
<TD WIDTH=388><A
  HREF="http://www.ee.ethz.ch/~oetiker/webtools/mrtg/"><IMG
  BORDER=0 SRC="mrtg-r.png" WIDTH=388 HEIGHT=25
  ALT="Multi Router Traffic Grapher"></A></TD>
</TR>
</TABLE>
<TABLE BORDER=0 CELLSPACING=0 CELLPADDING=0>
<TR VALIGN=top>
<TD WIDTH=88 ALIGN=RIGHT><FONT FACE="Arial,Helvetica" SIZE=2>
  version 2.9.21</FONT></TD>
<TD WIDTH=388 ALIGN=RIGHT><FONT FACE="Arial,Helvetica" SIZE=2>
  <A HREF="http://www.ee.ethz.ch/~oetiker/">Tobias Oetiker</A>
  <A HREF="mailto:oetiker@ee.ethz.ch">&lt;oetiker@ee.ethz.ch&gt;</A>
  and&nbsp;<A HREF="http://www.bungi.com/">Dave&nbsp;&Rand</A>&nbsp;&
  <A HREF="mailto:dlr@bungi.com">&lt;dlr@bungi.com&gt;</A></FONT>
</TD>
</TR>
</TABLE>
</BODY>
</HTML>

```

La documentation installée dans le répertoire mrtg vous éclairera sur les paramètres que l'on peut utiliser pour aménager la présentation.

Voici une solution possible :

```

[root@pchris2 local]# indexmaker --columns=1 --sort=descr \
> --sidebyside /var/www/html/mrtg/local/local.cfg \
> --output=/var/www/html/mrtg/local/index.html

```

Ceci va créer dans <http://pchris2.maison.mrs/mrtg/local/> une page index.html dont voici la représentation⁷.

Et voilà, nous avons une installation MRTG fonctionnelle, quoique peu utile sur une station de travail.

Nous allons refaire la même chose sur une passerelle entre un LAN et l'Internet. Ce sera tout de même plus intéressant.

⁷ Voir les ressources, p 45

MRTG en production

Avertissement

N'oublions pas que l'objet de ce chapitre est SNMP et non MRTG...

Les exemples qui suivent sont destinés à montrer comment utiliser SNMP et pas à montrer toutes les finesses de MRTG. Pour ça, vous avez la documentation fournie avec :)

La passerelle

En l'occurrence, il s'agit d'une machine Mandrake 9.0 sur laquelle MRTG 2.9.21 a été installé, comme vu plus haut.

Cette machine est équipée de deux interfaces Ethernet, l'une pour le LAN et l'autre pour le modem-câble. Par dessus, nous trouvons un lien PPP (PPPoE oblige). Ce qui nous donne ceci :

```
[root@gw1 root]# ifconfig
eth0 Lien encap:Ethernet HWaddr 00:20:18:2C:0E:E9
inet adr:192.168.0.250 Bcast:192.168.0.255 Masque:255.255.255.0
UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
RX packets:2157836 errors:0 dropped:0 overruns:0 frame:0
TX packets:2075111 errors:1 dropped:0 overruns:0 carrier:1
collisions:1840 lg file transmission:100
RX bytes:189444177 (180.6 Mb) TX bytes:691138390 (659.1 Mb)
Interruption:10 Adresse de base:0xe000

eth1 Lien encap:Ethernet HWaddr 00:60:8C:50:F3:3E
inet adr:192.168.100.250 Bcast:192.168.100.255 Masque:255.255.255.0
UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
RX packets:2788786 errors:0 dropped:0 overruns:0 frame:0
TX packets:2266845 errors:0 dropped:0 overruns:0 carrier:0
collisions:2250 lg file transmission:100
RX bytes:991395133 (945.4 Mb) TX bytes:228369369 (217.7 Mb)
Interruption:3 Adresse de base:0x300

lo Lien encap:Boucle locale
inet adr:127.0.0.1 Masque:255.0.0.0
UP LOOPBACK RUNNING MTU:16436 Metric:1
RX packets:28457 errors:0 dropped:0 overruns:0 frame:0
TX packets:28457 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 lg file transmission:0
RX bytes:3438607 (3.2 Mb) TX bytes:3438607 (3.2 Mb)

ppp0 Lien encap:Protocole Point-à-Point
inet adr:80.8.146.171 P-t-P:80.8.144.1 Masque:255.255.255.255
UP POINTOPOINT RUNNING NOARP MULTICAST MTU:1492 Metric:1
RX packets:401251 errors:0 dropped:0 overruns:0 frame:0
TX packets:523520 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 lg file transmission:3
RX bytes:180973212 (172.5 Mb) TX bytes:41086532 (39.1 Mb)
```

Nous utilisons, comme d'habitude le script cfmaker :

```
[root@gw1 root]# cfmaker \  
> --global 'WorkDir: /var/www/html/mrtg/local' \  
> --global 'Language: french' \  
> --global 'Options[_]: bits,growright' \  
> --ifdesc=descr public@localhost \  
> --output /var/www/html/mrtg/local/local.cfg
```

Seulement voilà, nous rencontrons un problème sur ppp0 :

...

```
### Interface 4 >> Descr: 'ppp0' | Name: '' | Ip: '80.8.146.171' | Eth: '' ###
### The following interface is commented out because:
### * has a speed of 0 which makes no sense
#
# Target[localhost_4]: 4:public@localhost:
# SetEnv[localhost_4]: MRTG_INT_IP="80.8.146.171" MRTG_INT_DESCR="ppp0"
# MaxBytes[localhost_4]: 0
# Title[localhost_4]: ppp0 -- gw1.maison.mrs
# PageTop[localhost_4]: <H1>ppp0 -- gw1.maison.mrs</H1>
# <TABLE>
# <TR><TD>System:</TD> <TD>gw1.maison.mrs in Le local technique</TD></TR>
# <TR><TD>Maintainer:</TD> <TD>Root &lt;root@localhost&gt; (configure /etc/snmp/snmp.conf)</TD></TR>
# <TR><TD>Description:</TD><TD>ppp0 </TD></TR>
# <TR><TD>ifType:</TD> <TD>ppp (23)</TD></TR>
# <TR><TD>ifName:</TD> <TD></TD></TR>
# <TR><TD>Max Speed:</TD> <TD>0.0 bits/s</TD></TR>
# <TR><TD>Ip:</TD> <TD>80.8.146.171 (ca-marseille-19-171.w80-8.abo.wanadoo.fr)</TD></TR>
# </TABLE>
```

SNMP ne donne pas la vitesse maxi de ppp0 et MRTG ne sait pas comment calculer les graphes. Nous allons devoir modifier tout ça à la main. Fort heureusement, ce n'est pas trop compliqué.

La vitesse maxi, nous la connaissons, elle dépend de l'abonnement. Ici, 512 kilo bits par seconde. Nous prendrons 640 kilo bits par seconde, soit 80 kilo octets par seconde. Nous modifions comme suit :

```
# MaxBytes[localhost_4]: 80000
...
# <TR><TD>Max Speed:</TD> <TD>640.0 Kbits/s</TD></TR>
```

De plus, laisser l'IP telle qu'elle a été trouvée au moment de la construction du fichier n'a aucun sens puisque cette adresse va changer à chaque nouvelle session PPPoE (sauf si vous disposez d'une IP fixe, bien entendu). Nous modifions donc la dernière ligne du tableau :

```
# <TR><TD>Ip:</TD> <TD>Adresse dynamique</TD></TR>
```

Ça devrait suffire à obtenir quelque chose de propre. Bien entendu, il ne faudra pas oublier de décommenter toutes les lignes nécessaires.

Il ne nous reste plus qu'à construire la page d'index :

```
[root@gw1 root]# indexmaker --columns=1 --sort=descr \
> --sidebyside /var/www/html/mrtg/local/local.cfg \
> --output=/var/www/html/mrtg/local/index.html
```

Et voilà le travail...⁸

Il est clair que les graphes représentant ppp0 et eth1 vont être très similaires, puisque ppp0 passe sur eth1. Ce ne sera probablement pas la peine de garder les deux, du moins dans ma configuration simple.

Encore plus...

La charge CPU

Afficher le trafic réseau, c'est bien, mais SNMP donne bien plus d'informations. Ce serait par exemple intéressant de tracer le graphe de la charge CPU.

Pour y arriver, c'est assez simple, il suffit de lire le tutoriel de net-snmp sur

⁸ Voir les ressources, p 48

<http://net-snmp.sourceforge.net/> et d'adapter le modèle à notre configuration :

```
LoadMIBs: /usr/share/snmp/mibs/UCD-SNMP-MIB.txt
### Attention, les trois lignes qui suivent doivent en fait n'en faire qu'une seule
Target[gw1.cpusum]: ssCpuRawUser.0&ssCpuRawUser.0:public@localhost
+ ssCpuRawSystem.0&ssCpuRawSystem.0:public@localhost
+ ssCpuRawNice.0&ssCpuRawNice.0:public@localhost
RouterUptime[gw1.cpusum]: public@localhost
MaxBytes[gw1.cpusum]: 100
Title[gw1.cpusum]: CHARGE CPU
PageTop[gw1.cpusum]: <H1>Charge Active CPU %</H1>
Unscaled[gw1.cpusum]: ymwd
ShortLegend[gw1.cpusum]: %
YLegend[gw1.cpusum]: Utilisation CPU
Legend1[gw1.cpusum]: CPU Actif en % (Charge)
Legend2[gw1.cpusum]:
Legend3[gw1.cpusum]:
Legend4[gw1.cpusum]:
LegendI[gw1.cpusum]: Actif
LegendO[gw1.cpusum]:
Options[gw1.cpusum]: growright,nopercent
```

Qu'avons-nous fait ?

Il nous faut aller dans la MIB, chercher les bonnes valeurs. Comme nous utilisons une représentation textuelle des feuilles à lire, il nous faut charger le fichier qui permettra d'interpréter cette représentation. Avec une représentation numérique, ça n'aurait pas été nécessaire.

L'outil `snmptranslate` va nous servir à repérer la partie de la MIB qui nous intéresse ici. Pour éviter de faire trop long, cherchons tout de suite dans la branche "private" :

```
[root@gw1 root]# snmptranslate -Tp -IR private
+--private(4)
|
+--enterprises(1)
|
+--ucdavis(2021)
|
+--prTable(2)
|
+--prEntry(1)
|   | Index: prIndex
|   |
|   +-- -R-- Integer32 prIndex(1)
|   |   | Range: 0..65535
|   +-- -R-- String prNames(2)
|   |   | Textual Convention: DisplayString
|   |   | Size: 0..255
|   +-- -R-- Integer32 prMin(3)
|   +-- -R-- Integer32 prMax(4)
|   +-- -R-- Integer32 prCount(5)
|   +-- -R-- Integer32 prErrorFlag(100)
|   +-- -R-- String prErrMsg(101)
|   |   | Textual Convention: DisplayString
|   |   | Size: 0..255
|   +-- -RW- Integer32 prErrFix(102)
|   +-- -R-- String prErrFixCmd(103)
|   |   | Textual Convention: DisplayString
|   |   | Size: 0..255
|
+--memory(4)
|
+-- -R-- Integer32 memIndex(1)
+-- -R-- String memErrorName(2)
|   | Textual Convention: DisplayString
|   | Size: 0..255
+-- -R-- Integer32 memTotalSwap(3)
+-- -R-- Integer32 memAvailSwap(4)
+-- -R-- Integer32 memTotalReal(5)
+-- -R-- Integer32 memAvailReal(6)
+-- -R-- Integer32 memTotalSwapTXT(7)
+-- -R-- Integer32 memAvailSwapTXT(8)
```

```

|   +--- -R-- Integer32 memTotalRealTXT(9)
|   +--- -R-- Integer32 memAvailRealTXT(10)
|   +--- -R-- Integer32 memTotalFree(11)
|   +--- -R-- Integer32 memMinimumSwap(12)
|   +--- -R-- Integer32 memShared(13)
|   +--- -R-- Integer32 memBuffer(14)
|   +--- -R-- Integer32 memCached(15)
|   +--- -R-- Integer32 memSwapError(100)
|   +--- -R-- String    memSwapErrorMsg(101)
|                   Textual Convention: DisplayString
|                   Size: 0..255
|
|   +---extTable(8)
|   |
|   |   +---extEntry(1)
|   |   |
|   |   |   Index: extIndex
|   |   |
|   |   |   +--- -R-- Integer32 extIndex(1)
|   |   |   |
|   |   |   |   Range: 0..65535
|   |   |   |
|   |   |   |   +--- -R-- String    extNames(2)
|   |   |   |   |
|   |   |   |   |   Textual Convention: DisplayString
|   |   |   |   |   Size: 0..255
|   |   |   |   |
|   |   |   |   |   +--- -R-- String    extCommand(3)
|   |   |   |   |   |
|   |   |   |   |   |   Textual Convention: DisplayString
|   |   |   |   |   |   Size: 0..255
|   |   |   |   |
|   |   |   |   |   +--- -R-- Integer32 extResult(100)
|   |   |   |   |   +--- -R-- String    extOutput(101)
|   |   |   |   |   |
|   |   |   |   |   |   Textual Convention: DisplayString
|   |   |   |   |   |   Size: 0..255
|   |   |   |   |
|   |   |   |   |   +--- -RW- Integer32 extErrFix(102)
|   |   |   |   |   +--- -R-- String    extErrFixCmd(103)
|   |   |   |   |   |
|   |   |   |   |   |   Textual Convention: DisplayString
|   |   |   |   |   |   Size: 0..255
|   |   |
|   |   |
|   |   +---dskTable(9)
|   |   |
|   |   |   +---dskEntry(1)
|   |   |   |
|   |   |   |   Index: dskIndex
|   |   |   |
|   |   |   |   +--- -R-- Integer32 dskIndex(1)
|   |   |   |   |
|   |   |   |   |   Range: 0..65535
|   |   |   |   |
|   |   |   |   |   +--- -R-- String    dskPath(2)
|   |   |   |   |   |
|   |   |   |   |   |   Textual Convention: DisplayString
|   |   |   |   |   |   Size: 0..255
|   |   |   |   |   |
|   |   |   |   |   |   +--- -R-- String    dskDevice(3)
|   |   |   |   |   |   |
|   |   |   |   |   |   |   Textual Convention: DisplayString
|   |   |   |   |   |   |   Size: 0..255
|   |   |   |   |   |
|   |   |   |   |   |   +--- -R-- Integer32 dskMinimum(4)
|   |   |   |   |   |   +--- -R-- Integer32 dskMinPercent(5)
|   |   |   |   |   |   +--- -R-- Integer32 dskTotal(6)
|   |   |   |   |   |   +--- -R-- Integer32 dskAvail(7)
|   |   |   |   |   |   +--- -R-- Integer32 dskUsed(8)
|   |   |   |   |   |   +--- -R-- Integer32 dskPercent(9)
|   |   |   |   |   |   +--- -R-- Integer32 dskPercentNode(10)
|   |   |   |   |   |   +--- -R-- Integer32 dskErrorFlag(100)
|   |   |   |   |   |   +--- -R-- String    dskErrorMsg(101)
|   |   |   |   |   |   |
|   |   |   |   |   |   |   Textual Convention: DisplayString
|   |   |   |   |   |   |   Size: 0..255
|   |   |   |
|   |   |   |
|   |   |   +---laTable(10)
|   |   |   |
|   |   |   |   +---laEntry(1)
|   |   |   |   |
|   |   |   |   |   Index: laIndex
|   |   |   |   |
|   |   |   |   |   +--- -R-- Integer32 laIndex(1)
|   |   |   |   |   |
|   |   |   |   |   |   Range: 0..3
|   |   |   |   |   |
|   |   |   |   |   |   +--- -R-- String    laNames(2)
|   |   |   |   |   |   |
|   |   |   |   |   |   |   Textual Convention: DisplayString
|   |   |   |   |   |   |   Size: 0..255
|   |   |   |   |   |   |
|   |   |   |   |   |   |   +--- -R-- String    laLoad(3)
|   |   |   |   |   |   |   |
|   |   |   |   |   |   |   |   Textual Convention: DisplayString
|   |   |   |   |   |   |   |   Size: 0..255
|   |   |   |   |   |   |
|   |   |   |   |   |   |   +--- -R-- String    laConfig(4)
|   |   |   |   |   |   |   |
|   |   |   |   |   |   |   |   Textual Convention: DisplayString
|   |   |   |   |   |   |   |   Size: 0..255
|   |   |   |   |   |
|   |   |   |   |   |   +--- -R-- Integer32 laLoadInt(5)
|   |   |   |   |   |   +--- -R-- Opaque    laLoadFloat(6)

```

```

|         |         Textual Convention: Float
|         |         Size: 7
|         +--- -R-- Integer32 laErrorFlag(100)
|         +--- -R-- String    laErrMsg(101)
|         |         Textual Convention: DisplayString
|         |         Size: 0..255
|
+---systemStats(11)
|
| +--- -R-- Integer32 ssIndex(1)
| +--- -R-- String    ssErrorName(2)
|         |         Textual Convention: DisplayString
|         |         Size: 0..255
| +--- -R-- Integer32 ssSwapIn(3)
| +--- -R-- Integer32 ssSwapOut(4)
| +--- -R-- Integer32 ssIOSent(5)
| +--- -R-- Integer32 ssIOReceive(6)
| +--- -R-- Integer32 ssSysInterrupts(7)
| +--- -R-- Integer32 ssSysContext(8)
| +--- -R-- Integer32 ssCpuUser(9)
| +--- -R-- Integer32 ssCpuSystem(10)
| +--- -R-- Integer32 ssCpuIdle(11)
| +--- -R-- Counter  ssCpuRawUser(50)
| +--- -R-- Counter  ssCpuRawNice(51)
| +--- -R-- Counter  ssCpuRawSystem(52)
| +--- -R-- Counter  ssCpuRawIdle(53)
| +--- -R-- Counter  ssCpuRawWait(54)
| +--- -R-- Counter  ssCpuRawKernel(55)
| +--- -R-- Counter  ssCpuRawInterrupt(56)
| +--- -R-- Counter  ssIORawSent(57)
| +--- -R-- Counter  ssIORawReceived(58)
| +--- -R-- Counter  ssRawInterrupts(59)
| +--- -R-- Counter  ssRawContexts(60)
|
+---ucdInternal(12)
+---ucdExperimental(13)
|
| +---ucdDlmodMIB(14)
|         +--- -R-- Integer32 dlmodNextIndex(1)
|         |
|         +---dlmodTable(2)
|                 |
|                 +---dlmodEntry(1)
|                         |         Index: dlmodIndex
|                         |
|                         +--- ---- Integer32 dlmodIndex(1)
|                                 |         Range: 1..65535
|                         +--- -RW- String    dlmodName(2)
|                                 |         Textual Convention: DisplayString
|                                 |         Size: 0..255
|                         +--- -RW- String    dlmodPath(3)
|                                 |         Textual Convention: DisplayString
|                                 |         Size: 0..255
|                         +--- -R-- String    dlmodError(4)
|                                 |         Textual Convention: DisplayString
|                                 |         Size: 0..255
|                         +--- -RW- EnumVal  dlmodStatus(5)
|                                 |         Values: loaded(1), unloaded(2), error(3), load(4), unload(5),
create(6), delete(7)
+---ucdDemoMIB(14)
|
| +---ucdDemoMIBObjects(1)
|         |
|         +---ucdDemoPublic(1)
|                 |
|                 +--- -RW- Integer32 ucdDemoResetKeys(1)
|                         |         Range: 0..2147483647
|                 +--- -RW- String    ucdDemoPublicString(2)
|                         |         Size: 0..1024
|                 +--- -R-- String    ucdDemoUserList(3)
|                 +--- -R-- String    ucdDemoPassphrase(4)
|
+---fileTable(15)
|
| +---fileEntry(1)

```

```

|       | Index: fileIndex
|       |
|       | +--- -R-- Integer32 fileIndex(1)
|       | |       Range: 0..2147483647
|       | +--- -R-- String   fileName(2)
|       | |       Textual Convention: DisplayString
|       | |       Size: 0..255
|       | +--- -R-- Integer32 fileSize(3)
|       | +--- -R-- Integer32 fileMax(4)
|       | +--- -R-- EnumVal  fileErrorFlag(100)
|       | |       Textual Convention: TruthValue
|       | |       Values: true(1), false(2)
|       | +--- -R-- String   fileErrorMsg(101)
|       | |       Textual Convention: DisplayString
|       | |       Size: 0..255
|
+---version(100)
|
| +--- -R-- Integer32 versionIndex(1)
| +--- -R-- String   versionTag(2)
| |       Textual Convention: DisplayString
| |       Size: 0..255
| +--- -R-- String   versionDate(3)
| |       Textual Convention: DisplayString
| |       Size: 0..255
| +--- -R-- String   versionCDate(4)
| |       Textual Convention: DisplayString
| |       Size: 0..255
| +--- -R-- String   versionIdent(5)
| |       Textual Convention: DisplayString
| |       Size: 0..255
| +--- -R-- String   versionConfigureOptions(6)
| |       Textual Convention: DisplayString
| |       Size: 0..255
| +--- -RW- Integer32 versionClearCache(10)
| +--- -RW- Integer32 versionUpdateConfig(11)
| +--- -RW- Integer32 versionRestartAgent(12)
| +--- -RW- Integer32 versionDoDebugging(20)
|
+---snmperrs(101)
|
| +--- -R-- Integer32 snmperrIndex(1)
| +--- -R-- String   snmperrNames(2)
| |       Textual Convention: DisplayString
| |       Size: 0..255
| +--- -R-- Integer32 snmperrErrorFlag(100)
| +--- -R-- String   snmperrErrMsg(101)
| |       Textual Convention: DisplayString
| |       Size: 0..255
|
+---mrTable(102)
|
| +---mrEntry(1)
| |       Index: mrIndex
| |       |
| |       | +--- -R-- ObjID    mrIndex(1)
| |       | +--- -R-- String  mrModuleName(2)
| |       | |       Textual Convention: DisplayString
| |       | |       Size: 0..255
|
+---ucdSnmAgent(250)
|
| +---hpux9(1)
|
| +---sunos4(2)
|
| +---solaris(3)
|
| +---osf(4)
|
| +---ultrix(5)
|
| +---hpux10(6)
|
| +---netbsd1(7)
|
| +---freebsd(8)

```

```

| |
| +---irix(9)
| |
| +---linux(10)
| |
| +---bsdi(11)
| |
| +---openbsd(12)
| |
| +---unknown(255)
| |
+---ucdTraps(251)
|
| +---ucdStart(1)
| +---ucdShutdown(2)

```

Comme l'indique le tutoriel NET-SNMP, le système fournit trois compteurs :User, System et Nice. Si l'on souhaite, comme c'est le cas ici, afficher la charge totale, il nous faut additionner ces trois valeurs.

De plus, MRTG trace les graphes par deux sur le même diagramme. Nous traçons ici deux fois le même.

Pour le reste, il faudra regarder de plus près dans le tutoriel de MRTG si l'on souhaite comprendre les options qui sont spécifiées.

Quelques jours plus tard, voici ce que l'on peut observer⁹...

Notez la pointe d'activité autour des 4h du matin, due à l'activité générée par logrotate.

La mémoire disponible

Là encore SNMP peut nous informer, mais voyons d'abord un peu avec la commande top.

Ici, sur la passerelle, équipée de 128 Mo de RAM :

```

10:06am up 11 days, 22:04, 4 users, load average: 0,40, 0,24, 0,14
81 processes: 80 sleeping, 1 running, 0 zombie, 0 stopped
CPU states: 64,9% user, 0,8% system, 0,0% nice, 34,1% idle
Mem: 127360K av, 117440K used, 9920K free, 0K shrd, 13752K buff
Swap: 401584K av, 27408K used, 374176K free 24072K cached

```

Linux a tendance à prendre ses aises dans la RAM disponible (il n'est pas le seul système à le faire). Deux informations sont ici utiles :

- la mémoire RAM disponible (9920 Ko),
- la mémoire swap disponible (374176 Ko).

Voyons ce que sait nous dire SNMP :

```

[root@gw1 root]# snmpwalk -v 1 localhost -c public private.enterprises.ucdavis.memory
enterprises.ucdavis.memory.memIndex.0 = 0
enterprises.ucdavis.memory.memErrorName.0 = swap
enterprises.ucdavis.memory.memTotalSwap.0 = 401584
enterprises.ucdavis.memory.memAvailSwap.0 = 374176
enterprises.ucdavis.memory.memTotalReal.0 = 127360
enterprises.ucdavis.memory.memAvailReal.0 = 8872
enterprises.ucdavis.memory.memTotalFree.0 = 383048
enterprises.ucdavis.memory.memMinimumSwap.0 = 16000
enterprises.ucdavis.memory.memShared.0 = 0
enterprises.ucdavis.memory.memBuffer.0 = 15356
enterprises.ucdavis.memory.memCached.0 = 27312
enterprises.ucdavis.memory.memSwapError.0 = 0
enterprises.ucdavis.memory.memSwapErrorMsg.0 =

```

⁹ Voir les ressources, p 49

Nous pourrions par exemple afficher le swap total et le swap disponible, la mémoire réelle (RAM) totale et la mémoire réelle disponible. Nous allons le faire pour le swap.

```
LoadMIBs: /usr/share/snmp/mibs/UCD-SNMP-MIB.txt
Target[gw1.swap]: memAvailSwap.0&memTotalSwap.0:public@gw1.maison.mrs
Options[gw1.swap]: nopercent,growright,gauge,noinfo
Title[gw1.swap]: Swap
PageTop[gw1.swap]: <H1>Swap.</H1>
MaxBytes[gw1.swap]: 1000000000
kMG[gw1.swap]: k,M,G,T,P,X
Ylegend[gw1.swap]: Octets
ShortLegend[gw1.swap]: octets
LegendI[gw1.swap]: Swap dispo
LegendO[gw1.swap]: Swap total
Legend1[gw1.swap]: Swap disponible
Legend2[gw1.swap]: Swap total
```

Là encore, je vous laisse regarder dans la documentation de MRTG pour les détails de la configuration.

Quelques temps après, nous obtenons ceci¹⁰ (ce qui n'a ici, rien de passionnant, il faut bien le dire).

Conclusions

Juste un mot pour dire que MRTG, non seulement sait exploiter SNMP, mais qu'il peut aussi afficher des informations issues de scripts, ce qui en augmente encore la portée. Bien entendu, nous ne verrons pas comment faire ici.

Faites un "top" et observez les ressources consommées lorsque MRTG se met au travail... C'est une constante de la mesure : tout instrument de mesure perturbe le milieu dans lequel il opère. Aussi je vous conseille, si vous voulez observer quelque chose de relativement fiable d'installer MRTG sur une machine indépendante de celles dont vous voulez auditer les ressources, surtout en termes de CPU et de mémoire.

Dans le cadre d'un petit réseau domestique, SNMP/MRTG pour visualiser le trafic sur la passerelle, c'est déjà le grand luxe.

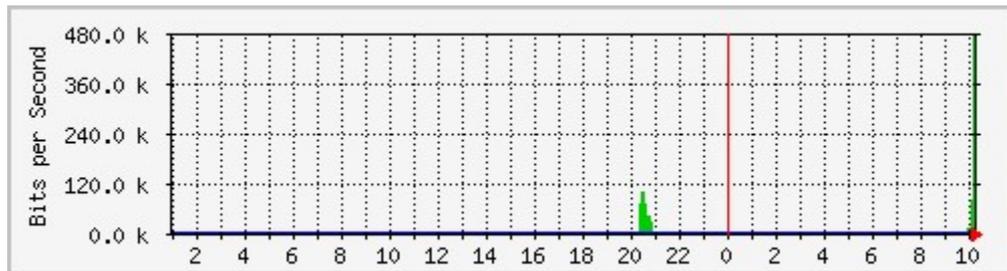
¹⁰ Voir les ressources, p 51

Ressources

MRTG : index.html

MRTG Index Page

eth0 --
pchris2.maison.
mrs



MRTG MULTI ROUTER TRAFFIC GRAPHER

version 2.9.21

Tobias Oetiker <oetiker@ee.ethz.ch>
and Dave Rand <dlr@bungie.com>

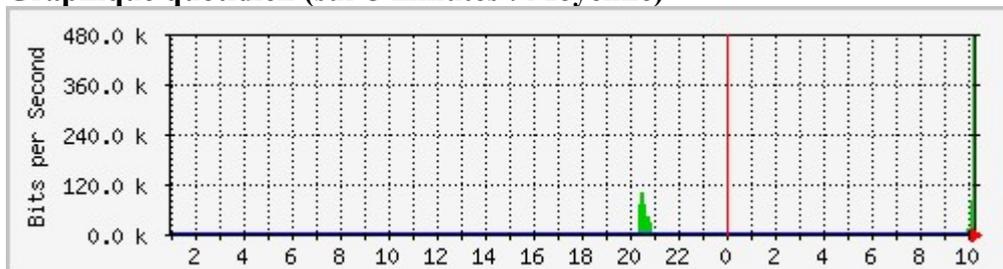
MRTG : détails

eth0 -- pchris2.maison.mrs

System: pchris2.maison.mrs in Unknown (edit /etc/snmp/snmpd.conf)
Maintainer: Root <root@localhost> (configure /etc/snmp/snmp.local.conf)
Description: eth0
ifType: ethernetCsmacd (6)
ifName:
Max Speed: 10.0 Mbits/s
Ip: 192.168.0.15 (pchris2.maison.mrs)

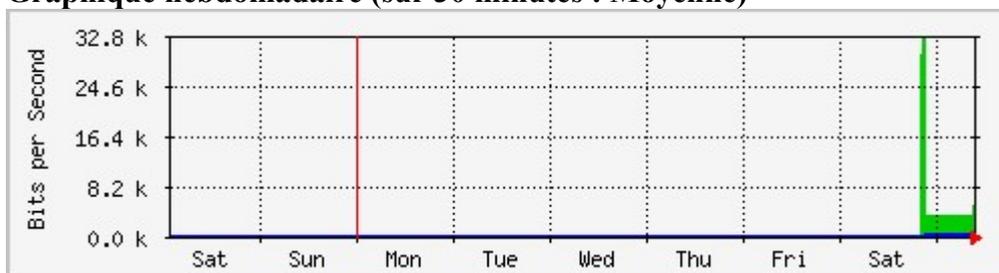
Les statistiques ont été mises à jour le **Dimanche 7 Septembre 2003 à 9:40**,
'pchris2.maison.mrs' était alors en marche depuis **0:40:47**.

Graphique quotidien (sur 5 minutes : Moyenne)



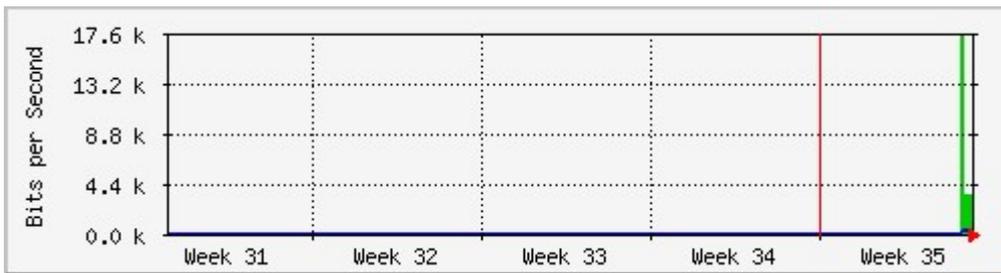
Max **Entrée:** 104.0 kb/s (1.0%) Moyenne **Entrée:** 5832.0 b/s (0.1%) Actuel **Entrée:** 640.0 b/s (0.0%)
Max **Sortie:** 2744.0 b/s (0.0%) Moyenne **Sortie:** 424.0 b/s (0.0%) Actuel **Sortie:** 144.0 b/s (0.0%)

Graphique hebdomadaire (sur 30 minutes : Moyenne)



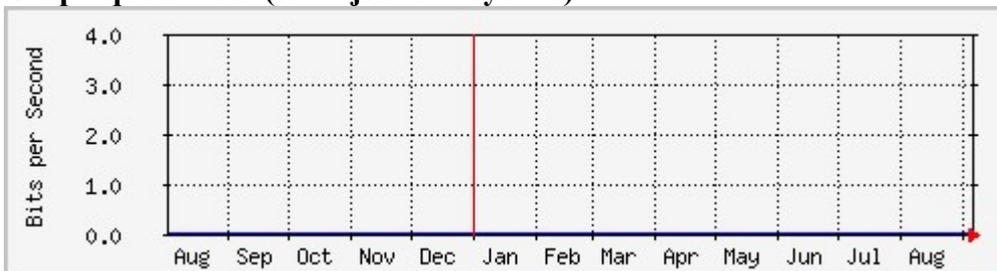
Max **Entrée:** 32.6 kb/s (0.3%) Moyenne **Entrée:** 5736.0 b/s (0.1%) Actuel **Entrée:** 5392.0 b/s (0.1%)
Max **Sortie:** 968.0 b/s (0.0%) Moyenne **Sortie:** 416.0 b/s (0.0%) Actuel **Sortie:** 968.0 b/s (0.0%)

Graphique mensuel (sur 2 heures : Moyenne)



Max **Entrée:** 17.4 kb/s (0.2%) Moyenne **Entrée:** 5928.0 b/s (0.1%) Actuel **Entrée:** 3632.0 b/s (0.0%)
Max **Sortie:** 408.0 b/s (0.0%) Moyenne **Sortie:** 392.0 b/s (0.0%) Actuel **Sortie:** 408.0 b/s (0.0%)

Graphique annuel (sur 1 jour : Moyenne)



Max **Entrée:** 0.0 b/s (0.0%) Moyenne **Entrée:** 0.0 b/s (0.0%) Actuel **Entrée:** 0.0 b/s (0.0%)
Max **Sortie:** 0.0 b/s (0.0%) Moyenne **Sortie:** 0.0 b/s (0.0%) Actuel **Sortie:** 0.0 b/s (0.0%)

VERT ### Trafic d'entrée en Bits par seconde
BLEU ### Trafic de sortie en Bits par seconde

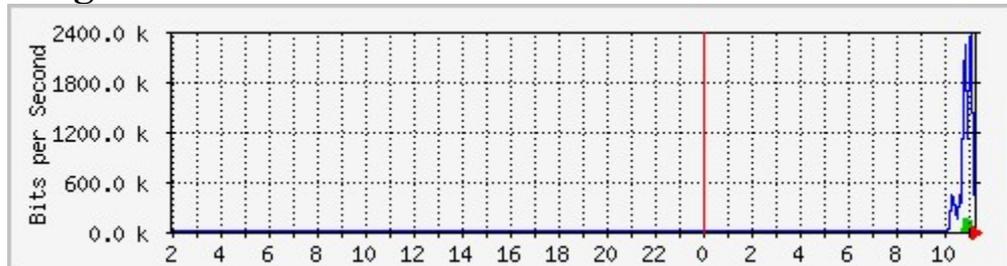
MRTG MULTI ROUTER TRAFFIC GRAPHER
version 2.9.21 Tobias Oetiker <oetiker@ee.ethz.ch> et Dave Rand <dlr@bungui.com>

Localisation effectuée par Fabrice Prigent <fabrice.prigent@univ-tlse1.fr>

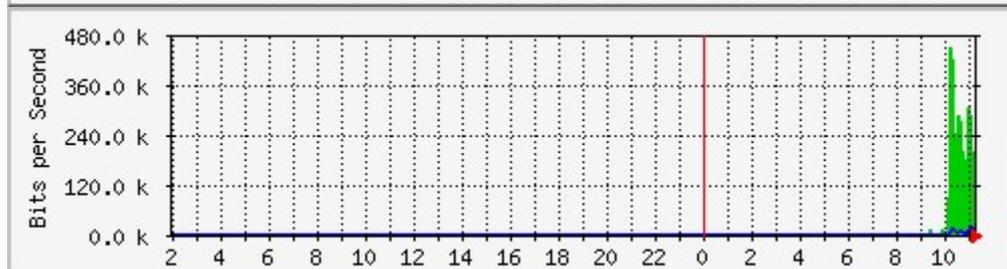
MRTG avec 3 interfaces réseau

MRTG Index Page

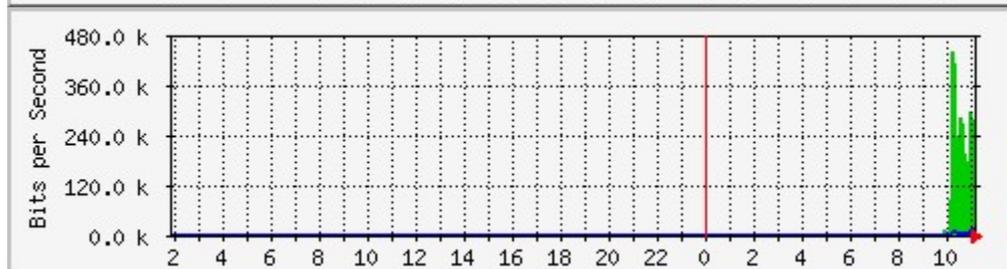
**eth0 --
gw1.maison.mrs**



**eth1 --
gw1.maison.mrs**



**ppp0 --
gw1.maison.mrs**



MRTG MULTI ROUTER TRAFFIC GRAPHER

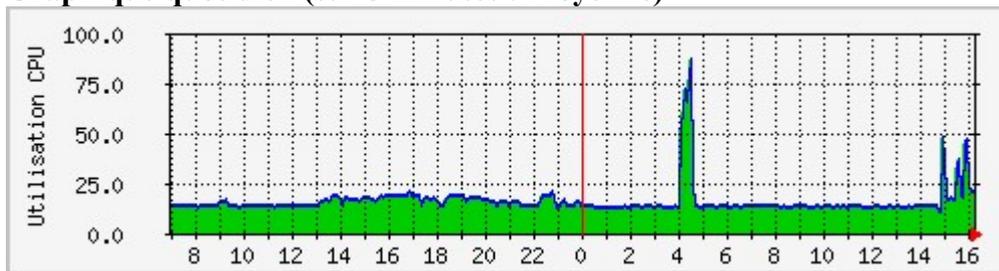
version 2.9.21

Tobias Oetiker <oetiker@ee.ethz.ch>
and Dave Rand <dlr@bungie.com>

Charge active CPU %

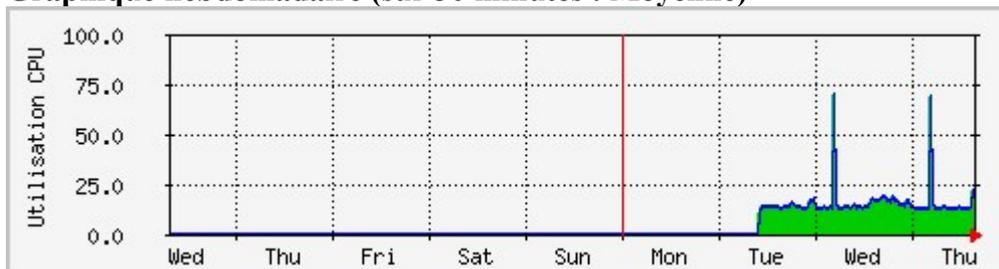
Les statistiques ont été mises à jour le **Judi 11 Septembre 2003 à 16:15**,
'gw1.maison.mrs' était alors en marche depuis **8 days, 20:35:13**.

Graphique quotidien (sur 5 minutes : Moyenne)



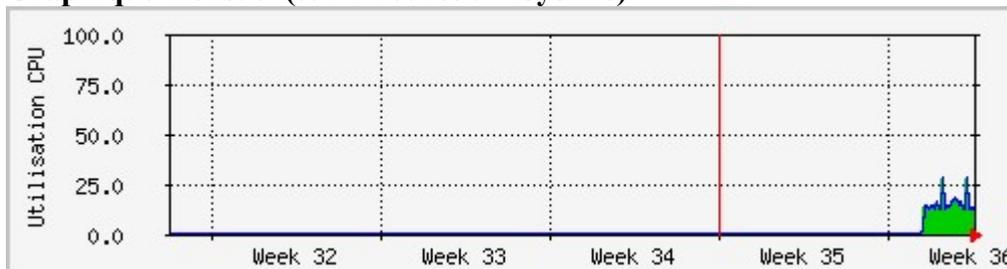
Max Actif 87.0 % Moyenne Actif 16.0 % Actuel Actif 21.0 %

Graphique hebdomadaire (sur 30 minutes : Moyenne)



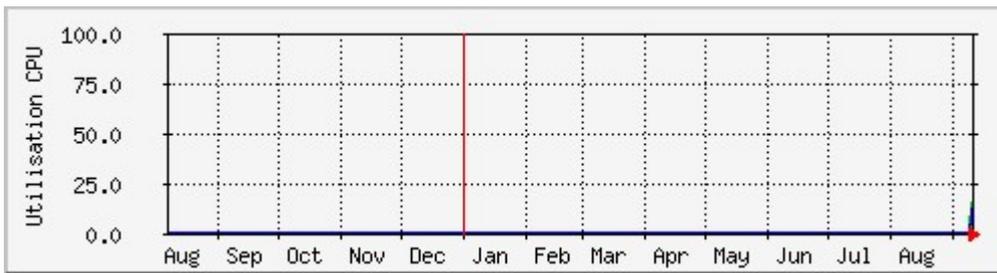
Max Actif 70.0 % Moyenne Actif 15.0 % Actuel Actif 19.0 %

Graphique mensuel (sur 2 heures : Moyenne)



Max Actif 28.0 % Moyenne Actif 15.0 % Actuel Actif 13.0 %

Graphique annuel (sur 1 jour : Moyenne)



Max Actif 16.0 % Moyenne Actif 13.0 % Actuel Actif 16.0 %

VERT ### CPU Actif en %

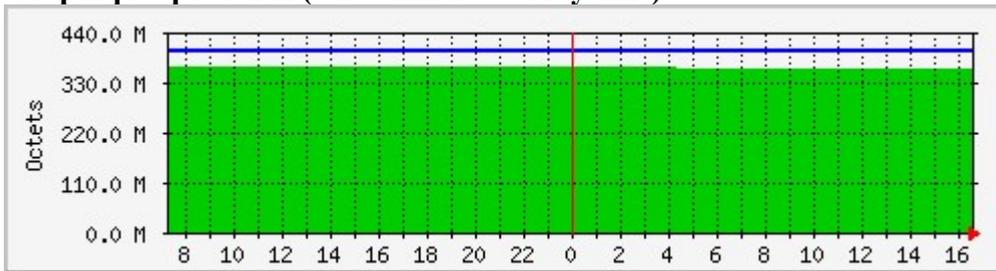
MRTG MULTI ROUTER TRAFFIC GRAPHER
version 2.9.21 Tobias Oetiker <oetiker@ee.ethz.ch> et Dave Rand
<dlr@bungicom.com>

Localisation effectuée par Fabrice Prigent <fabrice.prigent@univ-tlse1.fr>

Swap

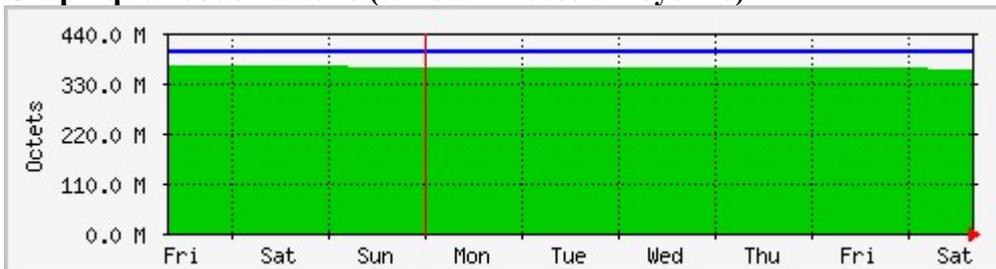
Les statistiques ont été mises à jour le **Samedi 20 Septembre 2003 à 16:35**

Graphique quotidien (sur 5 minutes : Moyenne)



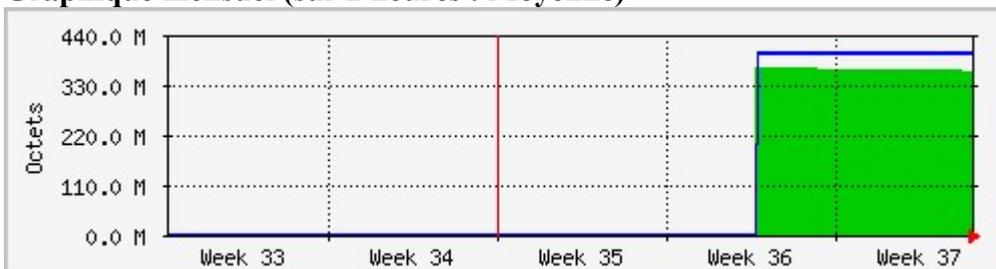
Max **Swap dispo** 366.3 Moctets Moyenne **Swap dispo** 365.7 Moctets Actuel **Swap dispo** 364.9 Moctets
Max **Swap total** 401.6 Moctets Moyenne **Swap total** 401.6 Moctets Actuel **Swap total** 401.6 Moctets

Graphique hebdomadaire (sur 30 minutes : Moyenne)



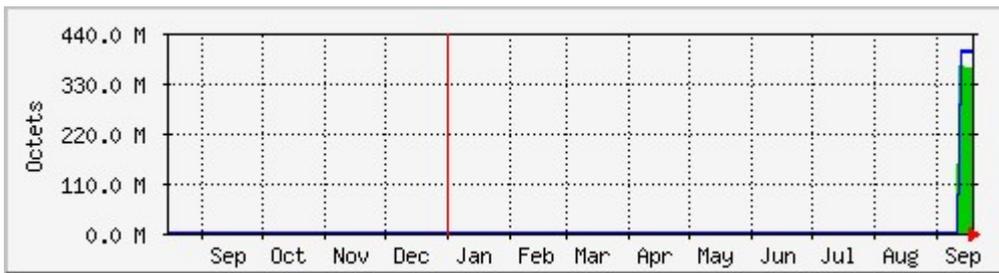
Max **Swap dispo** 370.7 Moctets Moyenne **Swap dispo** 368.3 Moctets Actuel **Swap dispo** 364.9 Moctets
Max **Swap total** 401.6 Moctets Moyenne **Swap total** 401.6 Moctets Actuel **Swap total** 401.6 Moctets

Graphique mensuel (sur 2 heures : Moyenne)



Max **Swap dispo** 371.2 Moctets Moyenne **Swap dispo** 368.5 Moctets Actuel **Swap dispo** 364.9 Moctets
Max **Swap total** 401.6 Moctets Moyenne **Swap total** 401.6 Moctets Actuel **Swap total** 401.6 Moctets

Graphique annuel (sur 1 jour : Moyenne)



Max **Swap dispo** 370.7 Moctets Moyenne **Swap dispo** 342.1 Moctets Actuel **Swap dispo** 368.0 Moctets
Max **Swap total** 401.6 Moctets Moyenne **Swap total** 372.3 Moctets Actuel **Swap total** 401.6 Moctets

VERT ### Swap disponible

BLEU ### Swap total

MRTG MULTI ROUTER TRAFFIC GRAPHER
version 2.9.21 Tobias Oetiker <oetiker@ee.ethz.ch> et Dave Rand
<dlr@bung.com>

Localisation effectuée par Fabrice Prigent <fabrice.prigent@univ-tlse1.fr>