Nouvelles Technologies Réseau

(Enseignant : Mr Duris.)

Nagios™

David Imanache
Nicolas Joubert
Olivier Mayaud

Ingénieurs 2000
Informatique & Réseaux 3

Sommaire

Sommaire **2**
Introduction **3**
1 Les enjeux et les acteurs de la supervision **4**
1.1 La problématique des réseaux privés d'entreprises 4
1.2 Les différents aspects de la supervision réseaux 5
1.2.1 L'analyse de flux 5
1.2.2 La supervision SNMP : l'état des équipements 6
1.2.3 La supervision applicative : 'état des services 6
2 Rappels sur le protocole SNMP **7**
2.1 Généralités 7
2.2 Un peu de technique 7
2.2.1 Les OIDs 7
2.2.2 Les MIBS 8
2.3 Fonctionnement 9
3 La supervision par Nagios **12**
3.1 Présentation de l'application 12
3.1.1 Un fonctionnement par plugin : 12
3.1.2 Une interface web complète : 12
3.2 Mise en place de la solution dans un réseau privé d'entreprise 13
3.2.1 Net-SNMP et le client NSCA pour la gestion des traps : 13
3.2.2 Gestion des traps personnalisés : 16
3.2.3 Résumé des deux modes de fonctionnement de Nagios : 17
3.3 La configuration du système de supervision 18
3.4 Illustrations de l'interface et présentation des résultats 23
4 Conclusion **25**
Table des illustrations **26**

Introduction

Ce document présente une étude réalisée dans le cadre du cours de réseaux sur les nouvelles technologies sous la direction de Etienne Duris. Elle consiste à présenter l'application open source de supervision de réseaux et de services : *Nagios*. Nous nous intéresserons plus particulièrement aux nouveautés que présente Nagios par rapport aux autres solutions existantes. L'idée est de décrire le fonctionnement du logiciel dans une installation type d'un réseau privé d'entreprise.

Nagios présente l'intérêt d'être une solution de supervision de services réseaux entièrement compatible avec les outils de supervision classiques tels que SNMP. De ce fait, un chapitre est dédié à des rappels sur ce protocole afin de donner au lecteur les éléments nécessaires à la bonne compréhension de cette étude.

1 Les enjeux et les acteurs de la supervision

Nous allons présenter dans ce chapitre la notion de supervision de réseau. En quoi cela consiste il exactement et concrètement ?
Nous tenterons de répondre à cette question en présentant les deux grands axes de la supervision et leurs objectifs respectifs.

1.1 La problématique des réseaux privés d'entreprises

La notion de supervision s'applique essentiellement dans un contexte de réseau privé. L'idée maîtresse de la supervision est de contrôler la bonne santé du réseau et des services informatiques afin de mesurer la Qualité de Service (QoS). Cette notion de QoS n'est pas récente, mais elle est devenue le point d'intérêt central des stratégies de développement d'entreprise. La situation économique actuelle peu favorable exige des entreprises qu'elles contrôlent leurs dépenses. La tendance n'est pas à l'investissement dans de nouvelles technologies, mais à l'optimisation et au rendement de l'existant à moindre coût. De plus, l'évolution des méthodes de travail au sein même de l'entreprise veut qu'une part de plus en plus importante de l'activité de celle-ci sollicite les outils informatiques. Dans le cas des sociétés de services en informatique, c'est toute l'activité des ingénieurs et dirigeants qui repose intégralement sur la bonne disponibilité des services informatiques et réseaux. C'est dans cette volonté de contrôle que s'intègre la supervision de réseaux.

L'enjeu pour ces entreprises est donc primordial, puisque leurs chiffres dépendent directement des outils informatiques tels que le mail, le partage de fichier, l'intranet… Les utilisateurs doivent pouvoir accéder à ces services à tout instant. Malheureusement, un simple incident survenu sur une ligne ou un équipement peut rompre toute la chaîne. L'objectif des administrateurs d'un système d'information est donc de pouvoir connaître en permanence l'état du réseau, et d'être averti en temps réel des différents incidents pour réduire au maximum le délais d'intervention et de coupure du service.

La haute hiérarchie des entreprises est elle aussi concernée par les outils de supervisions. Un responsable de service informatique doit pouvoir informer sa Direction de la QoS de son système informatique en s'appuyant sur des chiffres concrets. Par exemple il doit pouvoir affirmer que le service de messagerie a été l'année précédente à 2 % inaccessible, 13% perturbé…

1.2 Les différents aspects de la supervision réseaux

Un utilisateur qui accède à service informatique, comme par exemple une simple consultation d'une page web de l'intranet, met en œuvre de nombreux mécanismes et doit réunir toutes les bonnes conditions.

Tout d'abord les lignes de communications ne doivent pas être engorgées. Les éléments actifs établissant le chemin ainsi que la machine fournissant le service doivent être opérationnels et le service concerné doit être disponible. Le schéma ci-dessous retrace les différents points clés, garants de l'accessibilité d'un service.

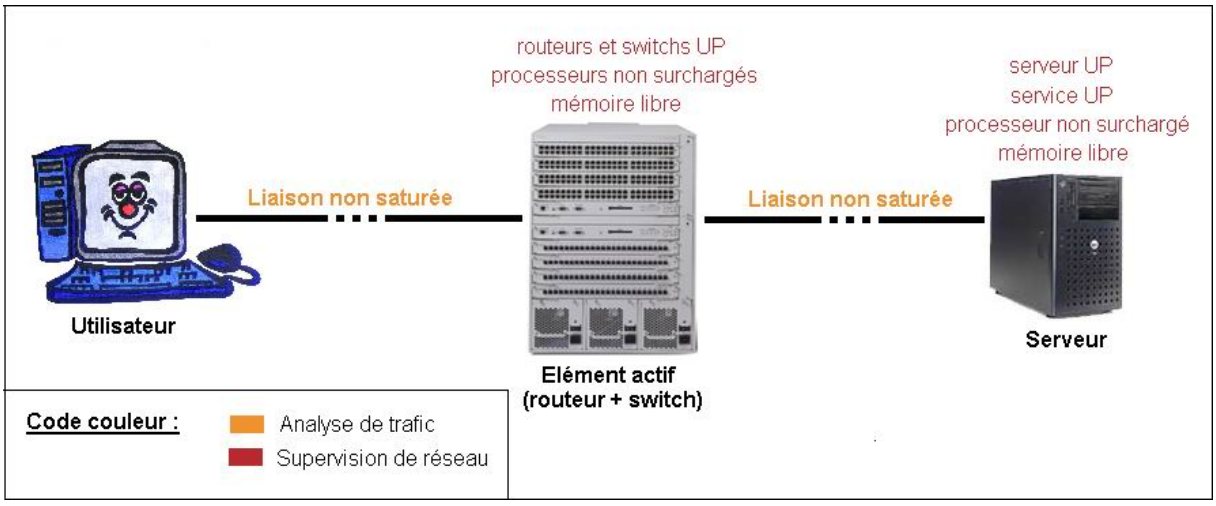

Figure 1 - Acces à un service distant

Nous voyons donc se distinguer deux axes majeurs de la supervision de réseau au sens général :

- L'analyse de flux
- la supervision des équipements et des services.

Notre étude porte uniquement sur ce deuxième aspect de la supervision. Nagios ne faisant pas d'analyse de flux.

1.2.1 L'analyse de flux

L'analyse de flux consiste à connaître l'activité en temps réel sur les lignes réseaux. Les buts sont à la fois d'identifier les liaisons saturées et d'identifier l'origine de cette sur utilisation des ressources. De plus, au travers de statistiques, les dirigeants peuvent décider de l'augmentation des débits de lignes trop faibles. Tout cela joue un rôle déterminant dans la stratégie de développement de l'entreprise.

De nombreux outils permettent de faire cela. Ils utilisent tous le même principe de sondes qui « sniffent » le trafic sur une ligne précise.

Ces informations sont ensuite analysées selon les besoins des administrateurs. En général les informations recherchées sont des classements par nombre de paquets transmis par utilisateurs ou par application, par protocoles…
Ceci permet d'identifier les causes principales de la saturation constatée.

Nous pouvons citer par exemple les logiciels *rrdtool* ou *Ntop* qui sniffent au niveau des interfaces de machines ou routeurs, ou encore *fastnet* (payant) qui nécessite la mise en place d'une sonde.

1.2.2 La supervision SNMP : l'état des équipements

Comme nous allons le voir dans le deuxième chapitre de ce dossier, SNMP est le protocole de référence en matière de supervision. Il permet à un serveur central de communiquer avec tous les équipements et serveurs pour connaître l'état de très nombreux paramètres tels que la température dans le châssis de la machine, l'état des interfaces, des ports, le taux d'utilisation CPU… Ce protocole permet également au serveur central (appelé station d'administration SNMP), de récupérer toutes les alertes SNMP (les traps[1]) émises par les éléments actifs ou serveurs supervisés. Une interface permet généralement d'administrer l'ensemble des machines et de visualiser leurs états en temps réel.

1.2.3 La supervision applicative : 'état des services

La supervision applicative, longtemps ignorée, et aujourd'hui au cœur des préoccupations d'entreprises. Le principe de cette supervision est différent de celui vu avec SNMP. La supervision applicative consiste surveiller en testant régulièrement les services qui tournent sur les machines serveurs des réseaux. La nouveauté est de se placer du point de vue de l'utilisateur final du service. En effet, même si tous les liens réseau sont en bon état de fonctionnement, le programme responsable d'un service peut en revanche être interrompu ou perturbé. SNMP ne permet pas de savoir par exemple, si le FTP est ouvert, si le serveur web renvoie bien la page… C'est ce que propose de faire les outils de supervision applicative. La station d'administration va régulièrement testé et mesurer la qualité des services en les sollicitant comme le ferait un utilisateur. Nous pouvons connaître ainsi le temps que met le serveur web à nous retourner la page, savoir si le serveur DNS traite bien les requêtes qui lui parviennent, le temps de réponse aux pings des différents routeurs…

Là encore une interface permet d'administrer le tout, de visualiser l'état des services ou encore de remonter des alertes aux administrateurs…

[1] Messages d'alertes, cf. le sous chapitre suivant.

2 Rappels sur le protocole SNMP

2.1 Généralités

Créé en 1988 par l'IETF (**I**nternet **E**ngineering **T**ask **F**orce), SNMP était à l'origine destiné à l'administration distante des ponts et des routeurs. Aujourd'hui, SNMP est utilisé pour l'administration d'un large éventail de matériel informatique. Plus qu'un simple protocole de liaison, la mise en place d'une structure utilisant SNMP fait intervenir plusieurs concepts:

- La station d'administration :
  C'est la machine qui centralise toutes les données, c'est le cœur du système. C'est notamment cette station qui va dialoguer avec les différents matériels à administrer.

- Les agents SNMP :
  Ce sont des démons installés sur tous les éléments du réseau à superviser (machines serveurs ou éléments actifs). Ils envoient les traps SNMP et répondent aux requêtes de la station d'administration.

- Les MIBs (Management Information Base) :
  Ce sont des sortes de bases de données avec une topologie en arbre qui décrivent et sauvegardent des variables SNMP. Elles sont présentes sur toutes les machines, y compris la station d'administration.

- Les variables SNMP :
  Ce sont les feuilles de la MIB. Ces variables sont identifiées numériquement ou nominalement par des OID (Object Identifier). Comme nous le disions plus haut, le protocole est utilisé pour connaître à un instant donné l'état d'un matériel. Ainsi ce sont ces variables qui nous permettront de connaître par exemple le nombre de paquets entrants et sortants sur une interface donnée ou encore la température CPU d'un serveur….

2.2 Un peu de technique

2.2.1 Les OIDs

Ce sont les identifiants des variables contenues dans les MIB. Nous avons précisé plus haut que cet identifiant pouvait être numérique ou littéral. En effet, chaque nœud de l'arborescence est repéré non seulement par son nom (iso, mgmt, etc…) mais aussi par un index (iso(1), mgmt(2), etc…), soit deux façon de qualifier un nœud de MIB.

Ainsi chaque variable est identifiée par un chemin absolu (avec le '.' comme séparateur) qui se compose soit:

- *Numériquement* : des différent index associés à chaque nœud parcouru.
  Par exemple « .1.3.6.1.2.1 » représente l'entrée de la MIBII.
- *Littéralement :* des noms associés aux nœuds visités.
  Par exemple « .iso.org.dod.internet.mgmt » représente l'entrée de la branche de MIB contenant la MIBII

2.2.2 Les MIBS

Comme nous l'avons dit plus haut, les MIBS regroupent toutes les variables traitées par le protocole SNMP ; l'ensemble étant stocké dans un arbre. Nous avons mis le mot MIB au pluriel car il existe au sein de SNMP une petite subtilité à ce propos :

- L'arbre global contenant toutes les variables est souvent appelé **MIBI**.
- Néanmoins on parle aussi très souvent de la **MIBII**, qui est en fait la branche de la MIBI réservée à l'administration du matériel réseaux.

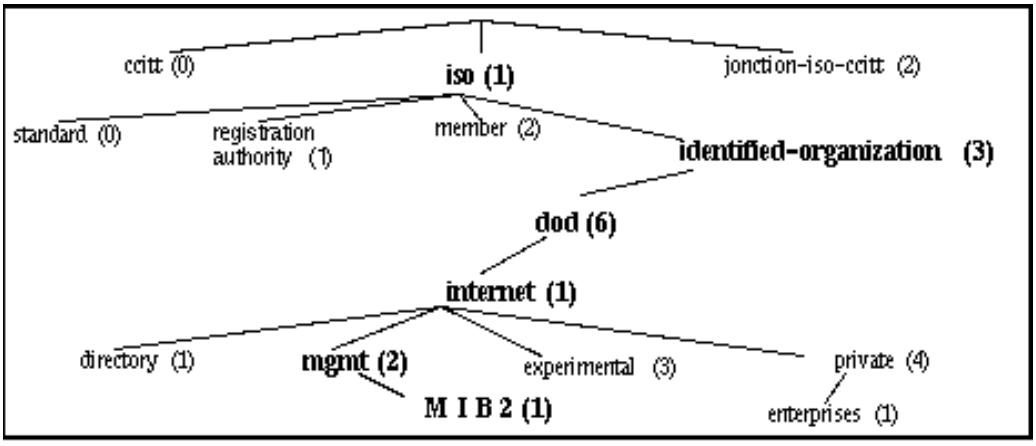

Figure 2 : Structure générale de la MIB SNMP

L'autre branche déterminante pour faire de l'administration est l'utilisation de la branche de MIB réservée aux entreprise (OID=.1.3.6.1.4.1). En outre une entreprise peut demander à obtenir sa propre branche de MIB (il suffit d'en faire la demande auprès de l'IANA, Internet Assigned Numbers Authority). Des grosses entreprises telles que Cisco ou Hewlett-Packard possèdent par exemple leurs propres branches de MIB dans lesquelles leurs matériels (au travers de l'agent SNMP qui y est installé) renseignent souvent des variables qui permettent d'avoir des renseignements tres précis sur le materiel (ex : la température interne d'un matériel.)

2.3 Fonctionnement

Ce protocole est basé sur un échange d'informations (par message) entre le matériel et la station de supervision. SNMP est un protocole de type client serveur, donc basé sur l'utilisation de démons (les agents SNMP).

C'est un protocole de la couche application (niveau 7 du modèle OSI) qui se base sur UDP pour l'acheminement des informations entre les différents agents (les agents écoutent les requêtes sur les ports UDP 161 et les alarmes sur le port 162). Chaque agent situé sur un noeud de réseau renseigne périodiquement les variables de MIB (délai paramétrable dans le fichier de configuration par défaut fixé à 30 secondes avec l'agent de net-snmp[2]).
L'accès aux variables d'un matériel réseau se fait grâce à un mot de passe particulier que l'on appelle la communauté :

- par défaut, la communauté en lecture a pour valeur 'public'
- par défaut, la communauté en écriture a pour valeur 'private'.

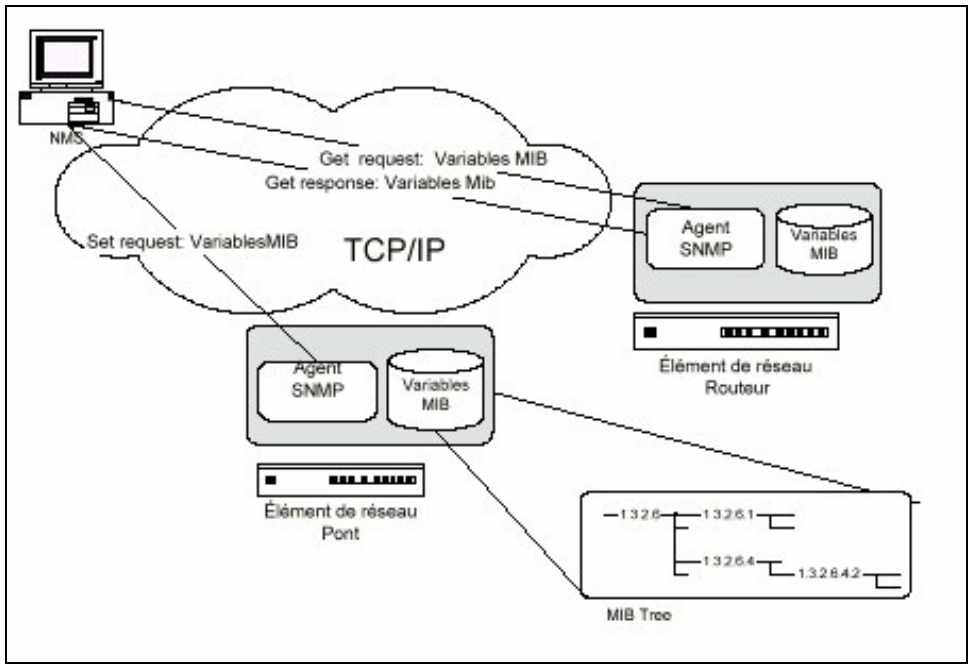

Figure 3 : Schéma de communication entre une NMS et le matériel (source : http://www.chez.com/marierenoir)

Le schéma ci-dessus nous montre les différentes étapes d'une connexion par le protocole SNMP entre une NMS (pour Network Management Station) et des noeuds de réseau. On peut voir apparaître notamment trois des cinq types de requête de basiques proposées par le protocole.

[2] Cf. la section consacrée à Net-SNMP (III.2.1)

- Les requêtes « get » servent à obtenir des informations
- Les requêtes « set » servent à modifier la valeur des variables des Mib.
- Les requêtes « get response » servent à obtenir la réponse de l'agent distant.

Ainsi on pourra également trouver des requêtes « trap » qui représentent des messages d'alerte envoyés spontanément par la machine administrée à la NMS (c'est un cas particulier car en principe le nœud ne fait que répondre à des requêtes du serveur). Enfin il faut citer les requêtes du type « get next » qui permettent de connaître la valeur de la variable se trouvant après celle qu'on lui passe en paramètre (permet notamment de parcourir toute une branche de MIB…)

D'une manière générale nous distinguons deux modes de fonctionnement complémentaires mais aux principes bien distincts :

- Le mode « **poll SNMP** » : La station d'administration demande à un agent SNMP de lui retourner la valeur de telle ou telle variable. Cela se passe donc en deux étapes, la requête du serveur pour interroger le client, et la réponse du client. Le serveur (ou station d'administration SNMP) peut ainsi connaître à des intervalles réguliers les informations qu'il souhaite. Nous venons de citer à titre d'exemple la température interne d'un élément actif, mais on peut aussi connaître le taux d'utilisation du processeur en temps réel, l'espace disque libre d'une machine… Le serveur est donc chargé d'interroger une liste d'informations bien précise pour connaître l'état de la machine. Le problème qui se pose est que l'on ne peut pas connaître à l'avance la nature des incidents. Ainsi, si l'un des deux processeur d'un élément actif est en panne, le second prend le relais automatiquement et personne ne s'aperçoit de rien, jusqu'au jour où le deuxième tombe en panne à son tour…

- Le mode « **Trap SNMP** » : un trap SNMP est un message informatif qui est envoyé par l'agent SNMP en cas d'incident à destination du serveur qui est toujours en écoute des ces messages. Pour reprendre notre exemple du processeur, si le premier venait à griller, un message indiquant la nature de l'incident est envoyé automatiquement à la station d'administration pour prévenir l'administrateur que le processeur n°1 de tel élément est hors service. La station d'administration se contente donc d'être à l'écoute de ces traps SNMP.

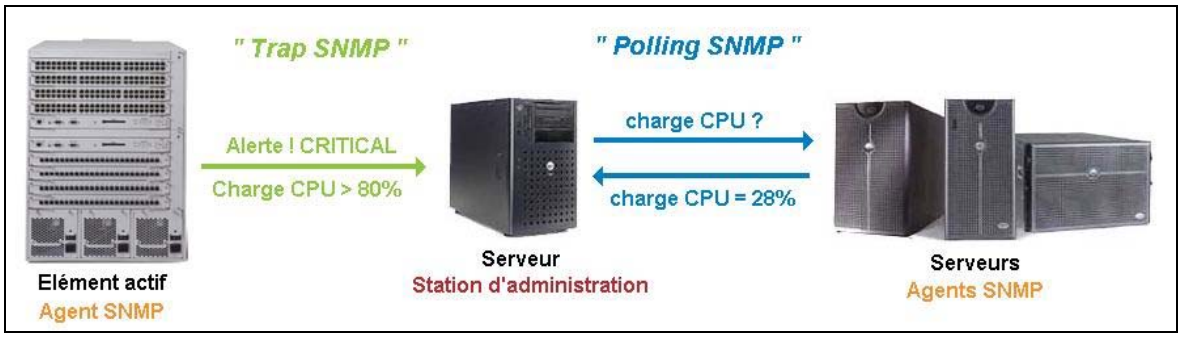

Figure 4 - Les deux méthodes de supervision SNMP

Ces deux modes permettent donc de superviser l'état des différents équipements du réseau. C'est ici que se trouve la limite de ce protocole SNMP. Il ne peut pas connaître les applications qui tournent sur la machine. Le protocole SNMP ne supervise que les premières couches du modèle OSI. On peut ainsi connaître l'état d'une machine, si elle est active et en « bonne santé » ou non, mais on ne sait pas si les services eux sont bien actifs et accessibles. Or, cela n'est rien moins que le point de vue de l'utilisateur, qui est au centre des intérêts de ce projet de supervision.

3 La supervision par Nagios

3.1 Présentation de l'application

Nagios est anciennement connu sous le nom de Netsaint. Plus exactement, Nagios est une évolution de Netsaint auquel a été ajouté, entre autre, la gestion du protocole SNMP.

Nagios est un outil développé principalement par Ethan Galstad, il est conçu pour fonctionner sous un système d'exploitation Linux (compatible UNIX). Cet outil propose de superviser les machines et les services d'un réseau via des plugins indépendants, chacun responsable d'un test particulier.

3.1.1 Un fonctionnement par plugin :

Le principe de fonctionnement de Nagios est simple, il dispose de nombreux plugins (des scripts exécutables) qui lui permettent de réaliser des tests sur les machines. Tout ce qu'il doit faire est renseigné dans des fichiers de configuration. Il connaît ainsi toutes les informations sur les machines ainsi que les tests qu'il doit effectuer et le comportement qu'il doit adopter en fonction des résultats. De nombreux plugins existent et ils permettent d'effectuer des taches courante tel que : l'ouverture d'une connexion SSH, ou la récupération une page Web en HTTP…
Le plugin **snmp_check** permet d'interroger les différents équipements en SNMP via la commande « snmp get ». Ceci constitue un aspect de la fonctionnalité SNMP de Nagios.

Bien évidemment, de part ce système de plugins, nous pouvons développer nos propres tests. Ceci permet à Nagios de superviser tout type d'application, y compris celles entièrement internes et propriétaires de l'entreprise.

3.1.2 Une interface web complète :

Nagios fournit une interface web en PHP, permettant aux administrateurs (login+password) de consulter l'état du réseau et des services et de visualiser toutes les alertes en coup d'œil.
Cette interface permet par ailleurs d'agir directement sur la configuration de Nagios via des scripts CGI. L'administrateur peut par exemple mettre en pause tous les tests d'une machine qu'il vient de couper pour des raisons de maintenance. L'autre avantage de cette interface est l'accès à toutes les statistiques générées par des scripts CGI. Ils sont entièrement paramétrables et permettent d'avoir une vue globale sur la qualité de service des machines.

3.2 Mise en place de la solution dans un réseau privé d'entreprise

Nous allons à présent imaginer sa mise en place dans un réseau privé contenant des éléments actifs (switchs, routeurs…) et des machines serveurs fournissant des services standards (FTP, HTTP, SSH, DNS…) et plus exotiques comme le système de messagerie par exemple…

3.2.1 Net-SNMP et le client NSCA pour la gestion des traps :

Nous avons déjà vu comment Nagios peut interroger des agents SNMP via un plugin. En revanche pour que notre application de supervision soit complète, il nous faut encore intégrer la gestion des traps SNMP. Pour cela une solution, prévue par Nagios, consiste à communiquer avec la station d'administration du réseau, celle qui reçoit ces fameux traps. Cette machine peut être distante ou non.

Nagios fournit deux petits programmes, le client send_nsca et le serveur **nsca**. Il s'agit de deux binaires et de deux fichiers de configuration très basiques. Le serveur ncsa doit être installer sur le serveur Nagios. Nagios ouvre un « pipe » dans lequel le serveur nsca peut écrire les résultats qu'il reçoit des clients send_nsca qui eux sont installés sur les machines distantes. Nous installerons ce client sur la station d'administration. Ce mécanisme assurera la liaison entre le serveur Nagios et la station d'administration qui reçoit les traps snmp.

Il nous reste encore à définir la communication entre la station de base, qui reçoit le trap SNMP émis par un des éléments du réseau, et le programme client de Nagios.
Nous utiliserons **Net-SNMP**, qui peut s'installer sous forme de package Linux. Net-SNMP fournit un ensemble de programmes SNMP, tels que les commandes snmpget, snmpwalk, snmptranslate et snmptrap, et deux programme démons snmpd et snmptrapd.

Snmptrapd est le programme chargé de récupérer les traps reçus et de déclencher une action en fonction de l'OID du trap. Nous pouvons ainsi écrire un script shell qui sera exécuté à la réception des traps qui permettra de faire remonter les informations à Nagios via le client NSCA de Nagios.

Voici un exemple de configuration du démon snmptrapd (snmptrapd.conf)

```
# LINK DOWN :
traphandle
.iso.org.dod.internet.snmpV2.snmpModules.snmpMIB.snmpMIBObje
cts.snmpTraps.linkDown /usr/local/nagios/NoAction

#LINK UP :
traphandle
.iso.org.dod.internet.snmpV2.snmpModules.snmpMIB.snmpMIBObje
cts.snmpTraps.linkUp /usr/local/nagios/NoAction

[...]

# Alimentation DOWN :
traphandle .1.3.6.1.4.1.2272.1.21.0.6
/usr/local/nagios/traitement-traps 6

# Alimentation UP :
traphandle .1.3.6.1.4.1.2272.1.21.0.14
/usr/local/nagios/traitement-traps 14

[...]

# trap perso : .1.9 utilise pour la gestion des espaces
disques des machines AIX
traphandle .1.9 /usr/local/nagios/trapperso

# trap bidon : .1.9.9 utilise pour la gestion de la mail
queue des machines AIX
traphandle .1.9.9 /usr/local/nagios/trapperso

[...]

# Traitement par défaut :
traphandle default /usr/local/nagios/traitement-traps 0
```

A la réception d'un trap, on lance un script shell selon l'OID du trap. Si le trap ne correspond à aucun de ceux dans la liste, alors le démon exécutera le script indique derrière le mot clé 'default' (dernière ligne).

Nous exécutons ici deux scripts différents. Traitement-trap prend en paramètre un chiffre qui lui permet de distinguer le trap reçu. Ce script, réalise les actions de log, du parsage dans la mib pour récupérer les infos du trap (commande snmptranslate de Net-snmp…). C'est ce script qui fait appel au client send_nsca en lui passant différents paramètres :

```
# envoi des infos a Nagios :
/bin/echo -e "$ipaddress\t$service\t$state\t$descr\n" |
/usr/local/nagios/nsca/client/send_nsca localhost -c
/usr/local/nagios/nsca/client/send_nsca.cfg
```

La syntaxe est définie dans la documentation de nsca :

- *$ipaddress :* adresse ip de l'élément emetteur du trap (routeur, serveurs..). Cette information se trouve à la première ligne du trap.

- *$service :* nom du service (« SNMPtraps » ici) associé dans le fichier de configuration du serveur Nagios, services.cfg. Nous y reviendrons plus tard.

- *$state :* entier indiquant à Nagios l'état de l'information du test.
  0 = OK
  1 = WARNING
  2 = CRITICAL
  3 = UNKNOWN

- *$descr* : chaîne de caractère de taille limitée. Nous utilisons cette chaîne pour retourner à Nagios la description du trap dans la mib récupérée par la commande snmptranslate de Net-snmp.

Nous pouvons distinguer plusieurs types de traps. Ceux connus dans les mibs standards (linkUp et linkDown), ceux contenus dans les mibs privées (ici celles de Nortel : …2272.1.21…) et ceux personnalisés crées par nous même avec NET-SNMP (définis dans snmpd.conf). Ces derniers vont nous permettre de superviser de façon passive l'espace disque et la mail queue de certains serveurs AIX. Le démon snmpd permet aussi de superviser directement l'espace disque des machines Linux. Cette autre méthode consiste simplement à insérer les lignes suivantes dans le fichier snmd.conf :

```
disk /var/log/nagios 15%
disk /usr/local 15%
```

Cela permet de veiller à ce que les partitions /var/log/nagios et /usr/local dispose d'au moins 15% d'espace disque disponible.

3.2.2 Gestion des traps personnalisés :

Nous disposons du programme snmptrap fournit par Net-SNMP. Il permet d'envoyer des traps sur demande. Un trap est identifié par son OID, or, nous avons la possibilité de personnaliser une partie de la MIB de Net-SNMP. Nous pouvons alors y définir de nouveaux traps spécifiques à nos besoins.

Prenons l'exemple d'un serveur de messagerie sur lequel on voudrait contrôler le nombre de mail dans la file d'attente (la mail queue). Nagios fournit un plugin (check_mailq) qui se propose de faire cela. Notre problème pourrait être que l'application de messagerie n'est pas standard et ne réponde pas aux requêtes de ce plugin, ou qu'il y ait un « viruswall » en amont qui bloque les requêtes… Nous pouvons alors, soit en développer un, dans ce cas Nagios ferrait ses tests de façon active (reuqête/réponse), soit utiliser les traps SNMP. Il suffit pour cela de placer un script shell en crontab qui vérifie le nombre de messages en attente et envoie au besoin un trap d'OID personnalisé via la commande snmptrap. A la réception, le démon snmptrad de la station d'administration SNMP, reconnaît l'OID du trap et son contenu, et peut ainsi lancer le client Nagios pour communiquer l'alerte au serveur Nagios, comme nous venons de le voir.

Les traps personnalisés créés dans la mib ne disposent pas de description de base. On peut aller en placer une à la main, mais la syntaxe des mibs n'est pas très intuitive. Nous utilisons ces traps à des buts bien précis, nous en connaissons la description à communiquer à Nagios comme vu ci-dessus. De plus, un trap possède une partie réservée à de l'information. Nous allons donc utiliser cet emplacement pour y insérer en dur la description du trap. La station d'administration (le démon snmptrapd) se chargera d'extraire le message du trap. C'est ce que fait le script trapperso pour les trap .1.9 et .1.9.9 vu dans l'exemple précédent. Ensuite il se comporte comme le script traitement-trap, mais envoie le message extrait du trap via le client nsca plutôt que d'aller consulter la mib.

3.2.3 Résumé des deux modes de fonctionnement de Nagios :

Nous pouvons distinguer deux modes de fonctionnement complémentaires de Nagios : le mode actif, ou de polling et le mode passif ou de traps.

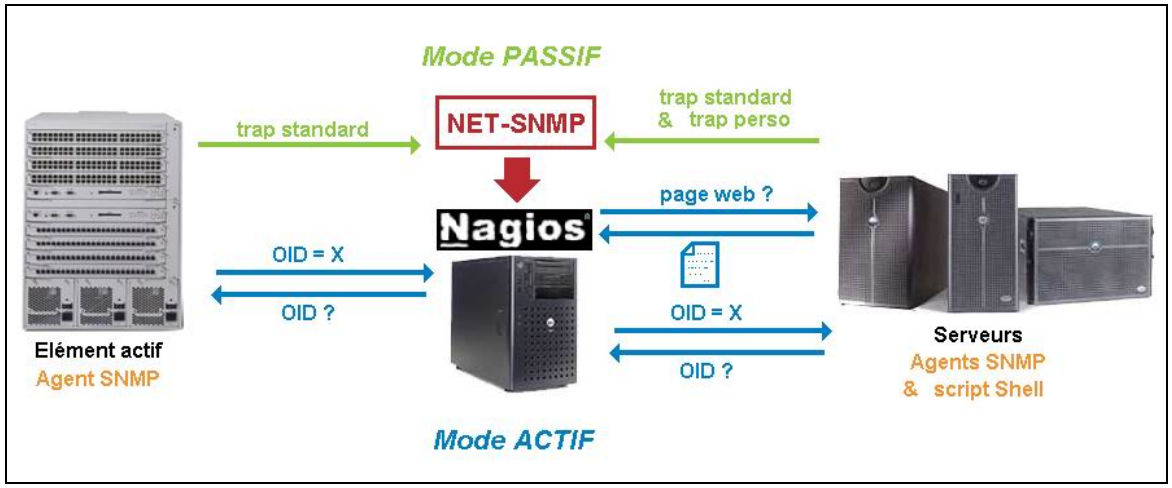

Figure 5 - Les deux modes de fonctionnement de Nagios

- En **mode polling**, Nagios exécute un plugin pour réaliser un test à des intervalles de temps réguliers. Il analyse ensuite la réponse et adopte un comportement en fonction de celle-ci. Ce mode de fonctionnement entraîne une génération du trafic sur le réseau.

- En **mode passif**, Nagios reste à l'écoute de tout ce qu'on peut lui dire. Pour communiquer avec lui, il suffit d'installer le programme client send_nsca sur les serveurs à superviser et de faire tourner le démon nsca sur le serveur Nagios. Dans notre configuration, c'est le démon snmptrapd de Net-SNMP qui utilise ce programme client via le script 'traitement-trap'.

Quelque soit le mode de fonctionnement, Nagios remonte des alertes aux administrateurs définis dans ses fichiers de configuration, que soit par mail,sms, pagger… Nagios met aussi en permanence à jour sont interface web qui reflète donc en temps réel l'état du réseau et des services.

3.3 La configuration du système de supervision

Nagios se configure via des fichiers textes où sont définis toute sorte d'objet au sens programmation. Nous définissons des objets représentants des machines, des services, des tests, des plages horaires, des administrateurs, des comportements, des objets génériques…

Cette phase de configuration de Nagios est relativement longue et fastidieuse si l'on souhaite gérer plusieurs centaines de machines et services. Il faut écrire plusieurs centaines de lignes dans plusieurs fichiers… De plus tous les tests doivent être très finement calibrés (seuils de criticité) car nous souhaitons que toutes les alertes nous soient communiquées par mail. Un nombre trop important de mail noie l'information importante. Seuls les alertes vraiment gênantes doivent êtres remontées aux administrateurs.

Prenons l'exemple des pings, on définit différents seuils de criticité qui détermine l'état d'accessibilité de la machine :

- OK
- WARNING
- CRITICAL
- DOWN

Attention à ne confondre ces états avec ceux des services tests remontés par les différents tests. Les temps de réponses dépendent de plusieurs paramètres comme l'engorgement de la ligne, l'éloignement de la machine… ainsi il n'est pas toujours possible définir des seuils standards. Les administrateurs de Nagios doivent donc passer du temps à observer le comportement du système de supervision pour affiner les nombreux réglages possibles pour ne conserver que les informations vraiment indispensables.

Nous allons présent lister les principaux fichiers de configuration de Nagios. Ils ne sont pas tous mentionnés, seulement les plus importants. Ces fichiers se trouvent dans le répertoire etc/ du répertoire d'installation de Nagios.

Fichier	Description
cgi.cfg	Configuration du site web et des cgi (authorization…).
checkcommands.cfg	Définition des tests.
contactgroups.cfg	Définition des groupes d'administrateurs.
contatcs.cfg	Définition des administrateurs (droits, adresse mail, nature des alertes…)
hostextinfo.cfg	Définissions complémentaires des machines pour la cartographie du réseau par les cgi de l'interface web (icône, emplacement…)
hostgroups.cfg	Définition des groupes de machines.
hosts.cfg	Définition des machines
miscommands.cfg	Définition des commandes. Notamment celle d'envoie par mail (host-notify-by-email)
nagios.cfg	Fichier de configuration principal (emplacement des fichiers, gestion des logs, user et group, comportement général…).
resource.cfg	Définition des variables. Notamment $USER1 = chemin d'accès aux plugins)
serviceextinfo.cfg	Définissions complémentaire des services. Utilisé par l'interface…
services.cfg	Définition des services à superviser. C'est le plus gros fichier à écrire. On y renseigne tous les services de toutes les machines que Nagios devra gérer.
timeperiods.cfg	Définitions des plages horaires (7j/7-24h/24 ou workhours..)

Nous allons à présent voir à titre d'exemple quelques extraits choisis de ces fichiers de configuration. Le but est aussi pédagogique puisqu'il va nous permettre de concrétiser ce que nous avons vu depuis le début.

Le fichier *nagios.cfg* est bien commenté est ne présente pas vraiment d'intérêt technique, nous ne nous attarderons pas, mais il est incontournable de le parcourir de A à Z.

Exemple de définition d'administrateur :

```
# 'toto' contact definition
define contact{
 contact_name                    toto
 alias                           Super ToTo
 service_notification_period     24x7
 host_notification_period        24x7
 service_notification_options    w,u,c,r
 host_notification_options       d,u,r
 service_notification_commands   notify-by-email
 host_notification_commands      host-notify-by-email
 email                           toto@mail.fr
}
```

On y indique entre autres les alertes qui lui seront communiquées. Ici il sera informer par mail de tous les types d'alertes (host :DOWN, UNREACHABLE, RECOVERY et service : WARNING, UNKNOWN, CRITICAL, RECOVERY)

Il sera alerté 24h/24 et 7j/7 grâce à l'objet 24x7 définit dans timeperiods.cfg :

```
# '24x7' timeperiod definition
define timeperiod{
 timeperiod_name 24x7
 alias           24 Hours A Day, 7 Days A Week
 sunday          00:00-24:00
 monday          00:00-24:00
 tuesday         00:00-24:00
 wednesday       00:00-24:00
 thursday        00:00-24:00
 Friday          00:00-24:00
 saturday        00:00-24:00
}
```

Il sera alerté par mail pour les services grâce à l'objet notify-by-email définit dans miscommands.cfg :

```
# 'notify-by-email' command definition
define command{
 command_name    notify-by-email
 command_line    /usr/bin/printf "%b" "***** Nagios 1.0
*****\n\nNotification Type: $NOTIFICATIONTYPE$\n\nService:
$SERVICEDESC$\nHost: $HOSTALIAS$\nAddress:
$HOSTADDRESS$\nState: $SERVICESTATE$\n\nDate/Time:
$DATETIME$\n\nAdditional Info:\n\n$OUTPUT$" | /usr/bin/mail
-s "** $NOTIFICATIONTYPE$ alert - $HOSTALIAS$/$SERVICEDESC$
is $SERVICESTATE$ **" $CONTACTEMAIL$
}
```

Nous pouvons ensuite définir nos services et nos machines. Nous avons sélectionnés des extraits en rapport avec les spécificités que nous avons décrites plus haut.

Exemple de machine serveur classique :

```
# 'heracles' host definition
define host{
 use        generic-host    ; Name of host template to use
 host_name                  heracles
 alias                      Heracles
 address                    192.168.251.13
 check_command              check-host-alive
 max_check_attempts         10
 notification_interval      120
 notification_period        24x7
 notification_options       d,u,r
}
```

Nous y incluons un objet générique définis dans ce même fichier :

```
# Generic host definition template
define host{
 name                       generic-host
 notifications_enabled      1    ; notifications enabled
 event_handler_enabled      1    ; event handler is enabled
 flap_detection_enabled     1    ; Flap detection enabled
 process_perf_data          1    ; Process performance data
 retain_status_information  1
 retain_nonstatus_information    1
 register   0
}
```

Définition d'un service classique (PING) :

```
define service{
use   generic-service ; Name of service template to use

host_name                  heracles
service_description        PING
is_volatile                0
check_period               24x7
max_check_attempts         3
normal_check_interval      5
retry_check_interval       1
contact_groups             nagios-admins
notification_interval      120
notification_period        24x7
notification_options       w,u,c,r
check_command              check_ping!100.0,20%!500.0,60%
}
```

La commande associée à ce service est définie dans le fichier checkcommands.cfg :

```
# 'check_ping' command definition
define command{
command_name     check_ping
command_line     $USER1$/check_ping -H $HOSTADDRESS$ -w
$ARG1$ -c $ARG2$ -p 5
}
```

Pour connaître la syntaxe d'un plugin, il suffit soit de lire la doc, soit d'exécuter le plugin à la main pour voir afficher une aide rapide.

Définition du service de gestion des traps SNMP.(SNMPtrap)

```
define service{
 host_name  heracles
 service_description        SNMPtraps
 is_volatile                     1
 active_checks_enabled      0
 passive_checks_enabled          1
 check_period               24x7
 max_check_attempts         1
 normal_check_interval      5
 retry_check_interval       2
 contact_groups             nagios-admins
 notification_interval      240
 notification_period        24x7
 notification_options       w,u,c,r
 check_command              check_none
}
```

Nous pouvons noter la directive passive_check_enabled = 1 pour indiquer qu'il s'agit d'un service test en mode passif. La commande (check_none) associée à ce service ne fait absolument rien.

Lorsqu'un trap sera reçu, snmptrapd exécutera le client nsca qui communiquera les informations au serveur nsca qui lui écrira les résultats dans un fichier (un pipe). Il est donc important que le nom de service envoyé par le client nsca soit bien déclaré dans le fichier de configuration.

3.4 Illustrations de l'interface et présentation des résultats

Nous allons à présent voir quelques exemples de pages web du site de Nagios pour donner un aperçu de son interface.

Voici une page, la status overview qui regroupe les machines selon les groupes définis dans hostgroups.cfg :

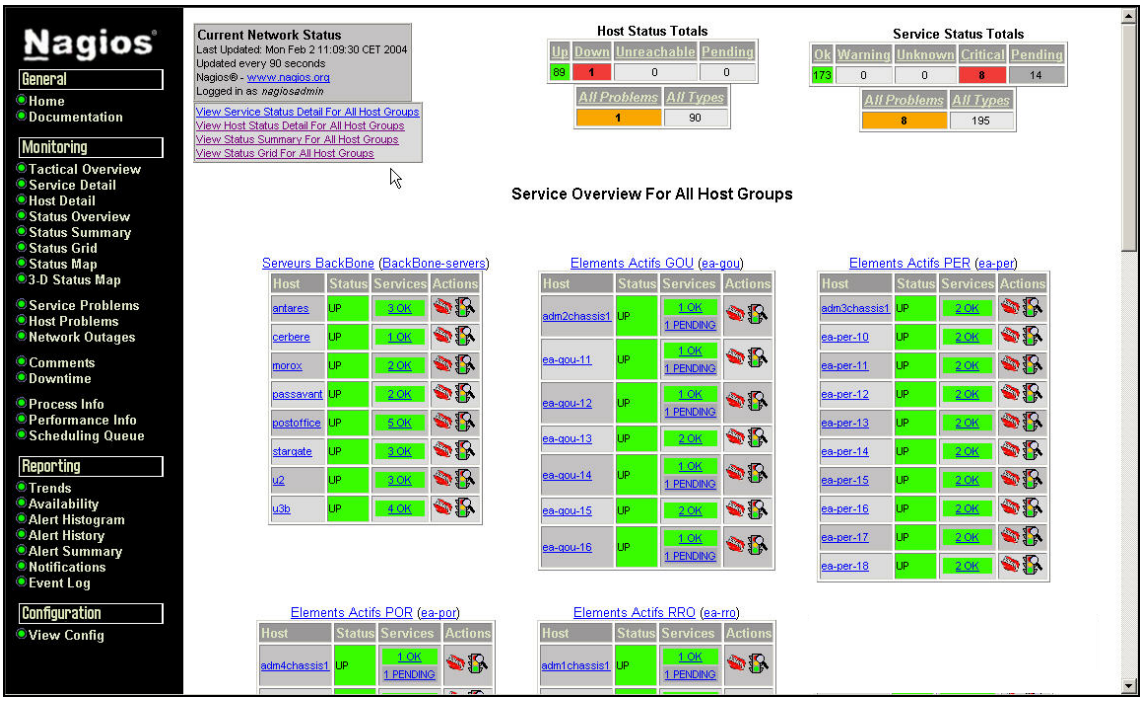

Figure 6 - Interface Nagios : "status overview"

La page suivante présente des statistiques sur le service d'impression d'un serveur. On y retrouve les informations sur la qualité de service. (Cette illustration est issue du site officiel de Nagios).

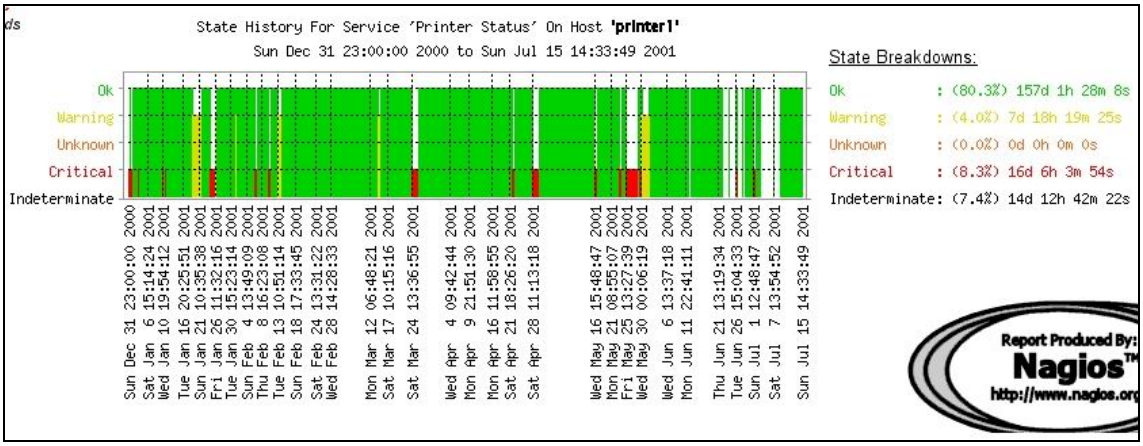

Figure 7 - Interface Nagios : Visualisation de statistique

Cette page retrace un historique complet des évènements, qui sont par stockés dans des fichiers de logs.

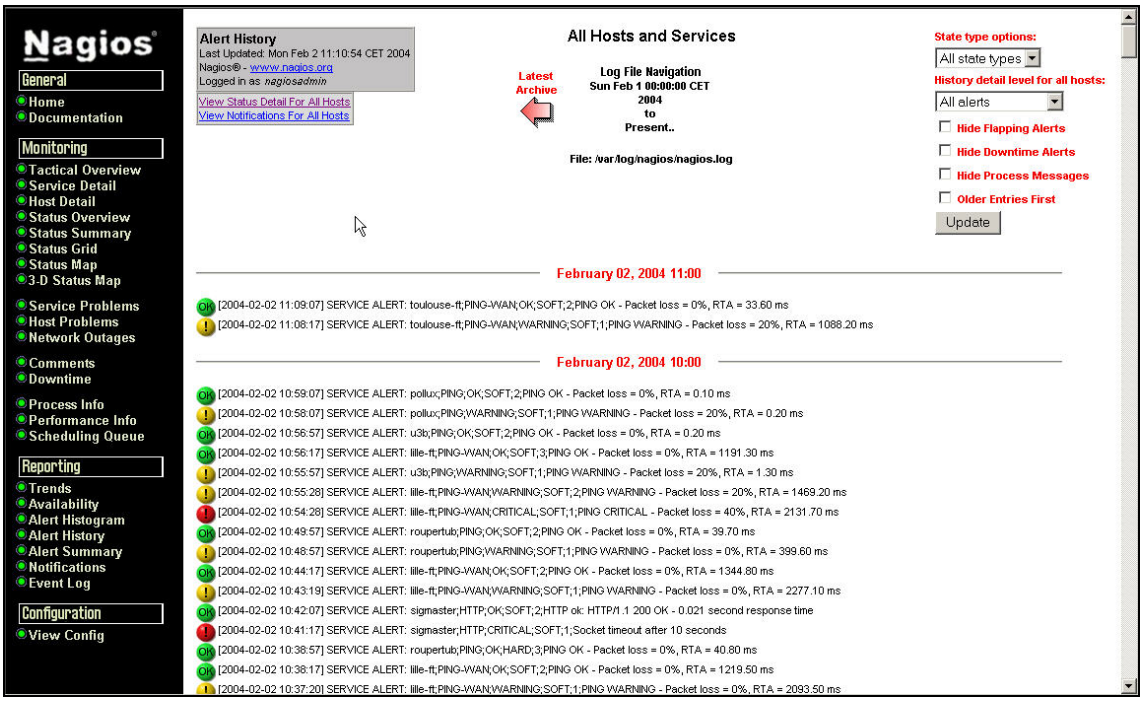

Figure 8 - Interface Nagios : Historique des événements

Cette dernière page suivante présente l'ensemble des services « CRITICAL ». On y retrouve principalement les traps SNMP :

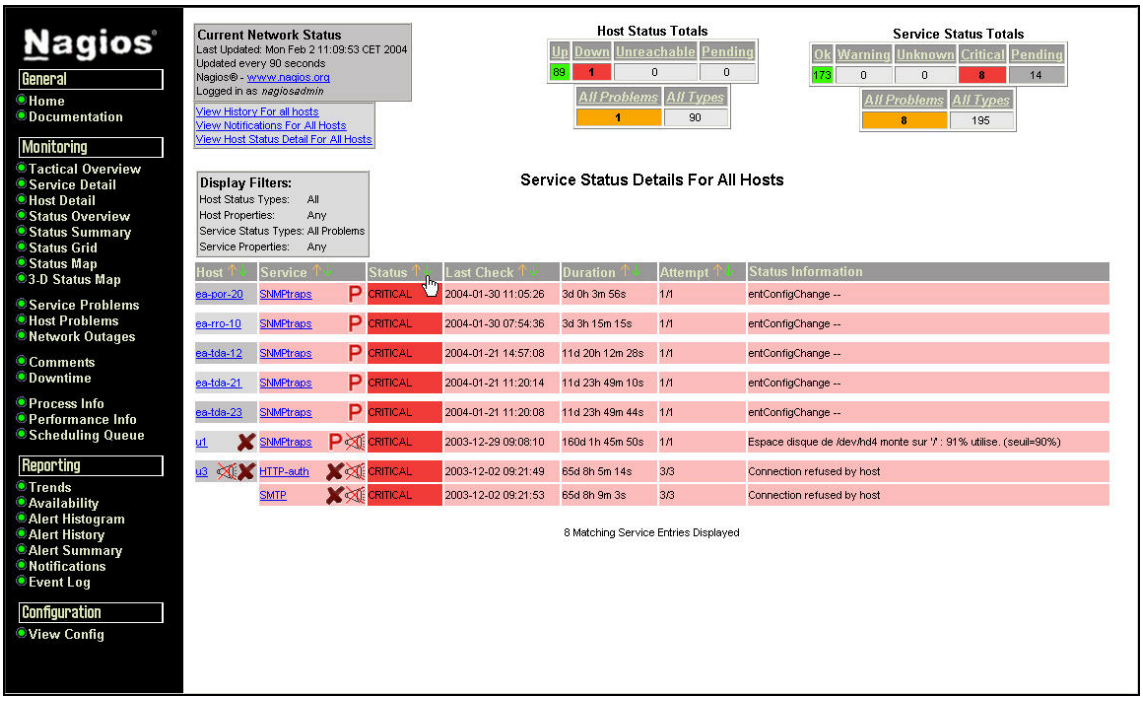

Figure 9 - Interface Nagios : Les services "critiques"

4 Conclusion

Nous rappelons que le but de ce dossier n'est pas de présenter les fonctionnalités exhaustives de Nagios. Nous avons voulu montrer ce que l'on peut faire avec cet outil de supervision et expliquer sa mise en place dans un réseau. Un autre point important de ce dossier et la présentation de l'utilisation du protocole SNMP avec Nagios, notamment la méthode de gestion de traps.

Les scripts ne sont pas présentés en détails pour ne pas alourdir d'avantage la lecture de ce document, mais nous en avons extrait les informations qui nous semblés les plus importantes à présenter.

Une documentation très complète existe déjà sur la configuration de Nagios, et sur l'utilisation de ces nombreux plugins. Elle est disponible sur le site de Nagios.

Nous avons voulu compléter, au travers de ce dossier, cette documentation qui peut éviter aux futurs administrateurs de passer trop de temps à comprendre comment s'emboîtent les différents outils pour mener une politique efficace de supervision. Nagios est tellement personnalisable, qu'il est quasiment possible de tout superviser avec, cela demande un peu de recherche de développement et de temps...

Table des illustrations

Figure 1 - Acces à un service distant 5
Figure 2 : Structure générale de la MIB SNMP 8
Figure 3 : Schéma de communication entre une NMS et le matériel 9
Figure 4 - Les deux méthodes de supervision SNMP 11
Figure 5 - Les deux modes de fonctionnement de Nagios 17
Figure 6 - Interface Nagios : "status overview".....23
Figure 7 - Interface Nagios : Visualisation de statistique 23
Figure 8 - Interface Nagios : Historique des événements.....24
Figure 9 - Interface Nagios : Les services "critiques" 24