UFR sciences et techniques
2, rue de la Houssinière
BP 92208
44322 Nantes Cedex 3

Stage du mois de juin 2005

Promotion : 2004 - 2005

Filière : Master 1 informatique

Tuteur : Attiobgé Christian

Rapport sur la mise en place d'une solution de supervision avec Nagios

ADJIDO Idjiwa

CEntre de Recherche Méthodologique d'Architecture (CERMA)
UMR CNRS 1563
EAN - Rue Massenet
BP 81931
44319 Nantes cedex 3, France
Tuteur entreprise : Thomas LEDUC

Sommaire

1. Ce que n'est pas ce document

Nous présentons dans ce document le contenu du stage (facultatif) réseau effectué sur une durée de un mois. Il présente dans son concept et ses éléments principaux la solution utilisée pour mettre en place une supervision de réseaux complémentaire à celle existante déjà. Ce document n'est donc pas un didacticiel étape par étape pour mettre en place ladite solution ; c'est une présentation de ce qu'il est conseillé de savoir ou d'avoir lu avant de se lancer dans sa mise en place. Les documentations spécifiques à la solution déployée allaient bien trop vite dans les manipulations techniques, nous avons donc souhaité ici différencier le « concept » du « technique ».

2. Objectifs

2.1. Cadre et besoins

Le stage se déroule dans un environnement comportant un parc d'une centaine de machines et de quelques serveurs. Il existe actuellement un plan de réseau semi dynamique, créé à partir d'informations sur les machines stockées dans des bases de données et à partir de requêtes SNMP[1], qui permet de déterminer sur quel port se situe telle machine. L'administrateur ne peut pas déterminer directement d'où vient le problème lorsqu'une machine est manquante sur le plan. Est-ce le port qui est désactivé, est ce la machine qui est éteinte ou l'interface réseau de cette machine qui est hors service ? L'administrateur ne peut pas non plus déterminer si un service particulier sur un serveur donné fonctionne correctement. Enfin, le simple fait de détecter qu'une anomalie existe derrière tel port de tel commutateur ne peut se faire sans regarder le plan de façon fréquente ou sans qu'un utilisateur ne déboule dans le bureau afin d'alerter l'administrateur. L'objet du stage est alors de trouver une solution afin de devenir « pro actif » face aux problèmes rencontrés, de pouvoir contrôler d'un simple coup d'œil l'état global du réseau, de déterminer l'origine d'un problème afin d'y remédier le plus rapidement possible. Ces quelques critères rentrent dans le domaine de la supervision de réseaux, essentiel pour assurer une disponibilité (souvent indispensable) du système d'information de l'entreprise. Nous définissons dans un premier temps ce que nous entendons par « supervision de réseaux » avant d'énoncer un comparatif des solutions actuelles du marché puis de nous pencher sur celle que nous avons choisie de mettre en place en présentant les notions l'entourant.

2.2. Cahier des charges

L'administrateur réseaux devra pouvoir surveiller son réseau via une interface web sécurisée. N'importe qui ne doit pas pouvoir accéder à cette interface, elle devra donc être protégée par un mot de passe. L'interface web devra comporter une cartographie du réseau, présentant les différents postes utilisateurs, les serveurs, les éléments actifs et imprimantes. Cette cartographie devra explicitement décrire l'état des ces machines ; permettre d'identifier l'état des ports des commutateurs sur ces mêmes machine serait un plus. L'interface devra permettre également, étant donné une machine, de vérifier que les services tournent correctement ou non. Des renseignements supplémentaires sur les différentes machines

[1] Simple Network Management Protocol

(charge CPU, espace disque, mémoire disponible, etc.) pourront être renseignés. Enfin, lorsque des problèmes surviendront, l'administrateur devra être notifié par un courriel dont le contenu indiquera le service et/ou la machine défectueuse. La solution devra bien entendu être la moins chère possible, nous pouvons comprendre gratuite. Nous nous sommes donc orienté vers une solution du monde libre : Nagios.

3. Supervision de réseaux

Qu'est ce que la supervision de réseau, à quoi elle sert et que nécessite-t-elle pour être mise en place ? C'est ce que nous tentons d'expliquer dans cette partie.

3.1. Définition

La supervision de réseaux peut être définie comme l'utilisation de ressources réseaux adaptées dans le but d'obtenir des informations (en temps réel ou non) sur l'utilisation ou la condition des réseaux et de leurs éléments afin d'assurer une bonne qualité et une répartition optimale de ceux-ci.

3.2. Objectif

L'objectif d'une supervision de réseaux peut ainsi être résumé en trois points :

- Etre réactif en alertant l'administrateur en cas de dysfonctionnement d'une partie du système d'information ;
- Etre pro actif en anticipant les incidents ;
- Cibler le problème dès son apparition afin d'agir rapidement de la façon la plus pertinente possible.

3.3. Problématique

La supervision de réseau nécessite des outils adaptés aux différents composants du réseau. Le parc est doté de machines sous Linux, Windows et Mac OS. Nous devons donc trouver, comme tout bon produit de supervision, un produit capable de superviser l'ensemble des équipements provenant de différents constructeurs et ayant des modes de gestion hétérogènes. Il s'agit alors de trouver un produit reposant sur un protocole ou un environnement normalisé afin de pouvoir servir de point d'entrée unique pour regrouper toutes les informations du réseau.

4. Solutions existantes

De nombreuses plateformes de supervision existent aujourd'hui. Certaines se contentent de connaître à tout instant l'état des nœuds du réseau, d'autres permettent également de connaître l'état des services sur ces nœuds, les derniers offrent la possibilité de ressortir de nombreuses statistiques du réseau permettant une analyse assez fine.

4.1. Les offres éditeurs

Depuis quelques années, conscient que la supervision est un marché porteur, les sociétés n'hésitent plus à investir dans un produit leur permettant de surveiller et mieux gérer leurs réseaux. Les éditeurs se sont alors lancés dans la course aux produits de supervision ; deux familles apparaissent, celle proposant des solutions généralistes supervisant le réseau, les serveurs, les applications, les sites web, etc. C'est le cas de *Patrol* ou *Mainview* (BMC), d'U*nicenter* (Computer Associate), de la gamme *openview* (HP), de *Tivoli* (IBM), de *BigBrother* pour ne citer que les plus connus. L'autre famille supervise des domaines plus spécifiques comme *Panorama* (Altaworks) qui gère uniquement l'aspect sécurité ou *PathWAI* (Candle) qui se penche principalement sur la supervision des applications. Toutes ces solutions ont en plus de spécificités les distinguant les unes des autres, un point commun : un prix élevé. Il faut compter près de 30 Keuros pour superviser un système d'information pour une entreprise de taille moyenne. Ce chiffre ne tient pas compte des formations du personnel travaillant avec ces solutions souvent complexes d'utilisation !

4.2. Les offres libres

Il existe des solutions de supervision libres qui sont professionnelles. Parmi les plus répandus, reconnus du moment nous pouvons citer *Nagios* (le successeur de *Netsaint*), *Zabbix*, *openNMS*. De ceux cités, *Nagios* est sans contexte le plus répandu et le plus suivi par la communauté des développeurs.

5. Solution sélectionnée : *Nagios*

5.1. Pourquoi Nagios ?

5.1.1. C'est une solution libre

Des solutions citées ci-dessus, *HPopenview*, *BigBrother* et *Nagios* sont les plus connues. *BigBrother*[2] est un superviseur de service fonctionnant sous windows NT. C'est une solution efficace mais qui ne permet de superviser qu'un nombre restreint de services. De plus, il n'est pas possible de rajouter des fonctionnalités ou de générer des alarmes par mail.
HPopenview est une solution modulaire très complète qui permet de cartographier automatiquement et dynamiquement le réseau, de collecter des informations de supervision, de les mettre en correspondance, d'envoyer des alarmes, de générer des comptes rendus graphiques...mais c'est également une solution payante, donc écartée de nos choix.

5.1.2. C'est une solution stable qui a fait ses preuves

Parmi les solutions libres, *Zabbix* et le projet *Oréon* ont été mis en concurrence pour notre choix avec le « célèbre » *Nagios*. Ce dernier est en effet réputé pour sa configuration fastidieuse mais également pour le fait qu'il soit tout aussi complet que la solution *HPopenview*. Le projet *Oréon*[3] est une couche au dessus de *Nagios* regroupant une interface de configuration web, une détection automatique du réseau et quelques fonctionnalités supplémentaires devant simplifier *Nagios*. Nous n'avons pas jugé le projet assez avancé pour la sélectionner comme solution stable. De plus, nous avons pensé que mettre en œuvre *Nagios* nous permettrait de mieux comprendre ce qui ce passe derrière *Oréon*. Enfin, *Zabbix*, désigné comme un potentiel concurrent à *Nagios* a été écarté pour l'orientation douteuse que prend le projet : l'auteur, bien que présentant son projet comme libre, ne veut pas que des développeurs touchent à son code, ce qui risque d'engendrer assez vite des clones de l'application rendant un support plus difficile. *Nagios* est stable, dispose d'une grande communauté de développeurs derrière elle et est utilisée par un grand nombre de fournisseurs d'accès ou de grands noms comme Air France, le CNRS[4] (taille de l'organisation : 26000 machines), l'IFSIC[5] (2500 machines) ou encore le modeste Ministère de l'Education National (130 000 machines).

[2] BigSister est libre, BigBrother n'étant libre que pour une utilisation personnelle.
[3] C'est un projet lancé par cinq étudiants d'Epitech
[4] Centre Nationale de la Recherche Scientifique
[5] Institut de Formation Supérieure en Informatique et Communication, Rennes

5.2. Présentation générale

5.2.1. Les fonctionnalités

L'ensemble des spécifications est disponible en ligne sur http://www.Nagios.org/about/
Nagios va permettre, entre autre, de superviser des services réseaux (SMTP, POP3, HTTP, DNS[6], etc.), de superviser les ressources systèmes (charge du processeur, processus en cours, etc.), de faire de la notification, classer les contacts à avertir par groupe de contacts, les machines par groupe de machines, de représenter par coloration les états des services et de leurs hôtes, de cartographier le réseau, de faire du « *reporting* », d'intégrer de nouveaux *plugins*, etc.

5.2.2. Le concept

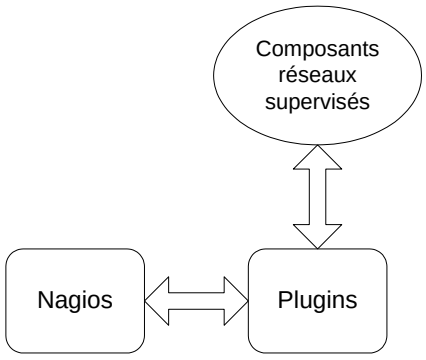
Nagios est un système de supervision de service développé pour des plateformes linux. Le concept consiste au lancement de contrôle des services et/ou stations définis à l'aide de « *plugins* » externes. Ces contrôles se font selon un intervalle défini durant la phase de configuration.

5.2.3. Le principe de fonctionnement

Les services de surveillance fournissent à l'application les résultats issus de l'analyse. Si ces résultats font remonter un problème, une notification est faite. Elle peut être effectuée par sms, messagerie instantanée ou mail. Dans le cadre du CERMA, nous nous contentons d'envoyer un mail à l'administrateur.

5.2.4. L'architecture

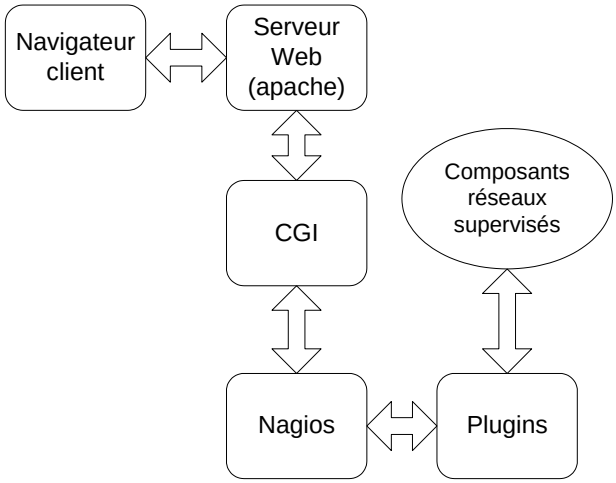
Nagios peut être considéré comme un noyau qui gère l'ordonnancement des vérifications (effectuées à l'aide de *plugins*) et les actions à prendre en fonction des incidents (alertes, actions correctives, etc.)



Afin de rendre plus exploitable les résultats, une interface web (nous utiliserons apache) basée sur les CGI[7] fournis par l'installation par défaut de Nagios a été rajoutée.

[6] *Simple Mail Transfert Protocol, Post Office Protocol, HyperText Transport Protocol, Domain Name Server*
[7] Common Gateway Interface

Nous obtenons alors l'architecture suivante :



Une base de donnée peut être couplée à *Nagios* lorsque le nombre de machines supervisées devient conséquent, ce qui n'est pas le cas du CERMA.

5.3. Les plugins

5.3.1. Les *plugins* de base

Nous entendons par *plugin* un programme exécutable ou un script (shell, perl, etc.) qui peut être lancé en ligne de commande pour tester une station ou un service. C'est le résultat de l'exécution du *plugin* qui est interprété par *Nagios* pour déterminer l'état du service ou de la station testée. Chacun peut donc définir son propre *plugin* et le rajouter aux *plugins* disponibles par défaut[8] ce qui permet un grand nombre de tests possibles. Il faut cependant respecter certaines conventions de code et de retour de fonction lors de l'écriture d'un *plugin* afin de coller aux *plugins* existants et d'être le mieux possible intégré à *Nagios*. Ainsi, le *plugin* devra être exécutable avec les droits de l'utilisateur *Nagios*, il devra afficher un message de préférence sur une seule ligne décrivant la situation du service (par exemple : « Temps de réponse OK : 0.586 secondes ») et posséder un code de retour[9] qui indique le statut du service :

- 0 : OK, le service fonctionne correctement ;
- 1 : Warning, le service est dégradé ;
- 2 : Critical, le service ne fonctionne plus ;
- 3 : Unknown, impossible de déterminer l'état du service.

Un nombre conséquent de *plugins* est fourni dans l'installation par défaut ; ainsi derrière le *plugin check_ping* se cache la commande */bin/ping* ou encore derrière le *plugin check_dns* se cache un simple *ping* sur ... *google.com* !

[8] La communauté *Nagios* offre aujourd'hui des bibliothèques de *plugins* assez complètes disponibles sur les sites http://www.Nagiosexchange.org, http://www.Nagios.org/download, http://sourceforge.net/projects/Nagiosplug/
[9] Ce que fait parfaitement la fonction *exit.*

5.3.2. NRPE, NSCA

Les créateurs du projet *Nagios* ont prévu des agents particuliers permettant de ne pas modifier les dispositions de sécurité existantes. Il s'agit principalement de *Nagios Remote Plugin Executor* et de *Nagios Service Check Acceptator*. Le premier constitue une méthode de surveillance active : un *plugin check_nrpe* permet à la station moniteur *Nagios* d'envoyer des instructions au démon NRPE situé sur la machine distante. Le second est une méthode de surveillance dite passive : un client NSCA est installé puis configuré sur chaque station supervisée afin qu'elle envoie d'elle-même les résultats de tests à la machine *Nagios*.

5.4. L'installation

Quelque soit la documentation[10] que vous pourrez obtenir sur *Nagios*, elle vous indiquera qu'il faut s'armer de patience. *Nagios* peut s'installer de deux façons, soit manuellement (c'est la partie annoncée comme fastidieuse, nous confirmons) en compilant les sources disponibles sur le site officiel, soit en utilisant les nombreux paquetages mis à disposition selon la distribution que vous utiliserez. Personnellement, la phase manuelle nous à permis, après cependant quelques heures de compilation, de recherche de librairies pour gérer les dépendances, de bien comprendre ou était installé tel ou tel fichier. La phase automatique, via l'utilisation d'un RPM (ou d'utilitaires tel que *yum* ou *apt-get*) se charge des dépendances entre les différents fichiers et librairies, une dizaine de minutes suffisent alors pour une installation minimale bien moins « prise de tête ». Il faut peut être noter que les répertoires suggérés lors de l'installation manuelle ne sont pas ceux utilisés lors des installations via paquetages et que les fichiers n'ont pas forcément les même noms selon les versions installées. Un document détaillant la mise en œuvre Nagios mise en œuvre pour le CERMA est fourni en annexe.

5.5. Le mode de déploiement

Nagios, en tant que bonne solution de supervision peut être déployée de trois manières différentes : centralisée, hiérarchisée ou distribuée.

5.5.1. Déploiement centralisé

Toute la supervision se fait à l'aide d'une seule machine. L'inconvénient immédiat est bien-sur que toute la gestion repose sur le bon état de la station de supervision. De plus, la machine doit être capable de gérer l'ensemble des données de supervision lui arrivant du réseau.

5.5.2. Déploiement hiérarchisé

Un serveur de supervision dialogue avec d'autres serveurs de supervision ; chacun d'eux étant affecté à un unique segment de réseau. Ces mêmes serveurs peuvent être à la fois client et serveurs de supervision, c'est-à-dire qu'ils peuvent également avoir d'autres serveurs sous leur responsabilité. Ce type de déploiement est plus complexe à mettre en place mais offre une meilleure tolérance à la panne. Un inconvénient supplémentaire est également un temps de

[10]Une documentation officielle française est disponible derrière http://www.Nagios.org/docs/, nous recommandons vivement de la suivre au moins pour savoir ce que font les différents paquetages ensuite.

réponse plus long du à une synchronisation nécessaire entres les serveurs pour la remontée d'informations.

5.5.3. Déploiement distribué

C'est un déploiement qui combine les deux précédents. Chaque station de supervision tient à jour une base de données et toutes les stations échangent entre elles toutes les données de supervision. Certaines stations peuvent être spécialisées ; la difficulté de ce déploiement consiste à mettre en place une parfaite coopération des stations.

5.5.4. Dans le cadre du CERMA

Nous rappelons que le réseau est un réseau modeste d'une centaine de machines. Nous pouvons nous permettre de centraliser la supervision sur une seule machine. Mettre en place un déploiement distribué serait du luxe inutile dans notre cas ; en revanche, le réseau étant segmenté en VLANs, nous pourrions très bien imaginer faire un déploiement hiérarchique. Mais ce serait prévoir plusieurs machines affectées à la supervision ce que ne permet pas les moyens du parc. Nous mettons donc en place une seule station *Nagios* sur laquelle reposera toute la gestion de la supervision. Nous utiliserons principalement le protocole SNMP en association avec *Nagios* évitant ainsi l'installation puis la configuration des agents NRPE, NSCA sur les stations.

5.6. La configuration de base

Configurer *Nagios* sans utiliser d'outils[11] particulier revient à éditer des fichiers textes. Nous présentons ici les principaux fichiers qui constituent la configuration de notre solution de supervision. Bien que certains outils nous permettent d'ignorer la présence de ces fichiers, il est bon de savoir qui ils sont et ce qu'ils contiennent ou encore comment ils sont reliés entre eux. Les noms des fichiers indiqués sont ceux par défaut, ils pourraient très bien être nommés autrement, le tout étant de savoir ce que l'on met et dans quel fichier.

5.6.1. Définition des commandes

C'est ici que l'on se sert des *plugins*. Afin d'en simplifier l'utilisation, nous créons une sorte d'alias : une commande composée d'un nom qui identifie la ligne de commande à laquelle elle correspond. Ces commandes sont placées dans un fichier *commands.cfg* et seront appelés dans les autres fichiers de configurations pour pouvoir tester stations et services selon la commande et le *plugin* utilisé.

5.6.2. Définition des contacts

Lorsque des notifications sont faites par *Nagios*, elles peuvent être transmises à un contact par mail, par sms, etc. C'est dans un fichier *contacts.cfg* que l'on définit le contact. Un contact doit alors avoir un nom, éventuellement un alias, les plages de fonctionnement auquel il peut être notifié et la manière qu'il aura de l'être. C'est à ce niveau que le mail doit être renseigné si nécessaire. Le contact peut alors appartenir à un groupe particulier au même titre que

[11] Certains outils comme Nagat, Nagios-admin (pour ne citer qu'eux !) sont disponibles sur http://www.Nagiosexchange.org/Configuration.40.0.html et permettent une configuration plus aisée de ces fichiers textes.

d'autres contacts. Ces groupes de contacts sont définis dans un fichier nommé *contactgroup.cfg* qui contient le nom du groupe et les membres qui y appartiennent.

5.6.3. Définition des stations

Le fichier *hosts.cfg* comporte la déclaration de toutes les stations. Nous y retrouvons les noms, les alias, les adresses des machines et également divers paramètres tels que les intervalles de tests, le nombre de tentatives, etc. Il y a bien sur possibilité de faire une déclaration générique. Ainsi, nous pouvons définir un *host* générique comportant une certaine configuration des paramètres. Toutes les stations déclarées utilisant le paramètre *use « le nom de la station générique »* hériterons alors de ses propriétés, cela permet d'alléger le fichier et une configuration plus claire. De la même façon que la définition des contacts plusieurs stations peuvent faire partie d'un même groupe. Par exemple, le CERMA possède quatre commutateurs qui sont déclarés individuellement mais qui font partie d'un même groupe commutateurs. C'est le fichier *hostgroup.cfg* qui comporte ces déclarations.

5.6.4. Définition des services

Le fichier *services.cfg* comporte la déclaration des services à superviser. De la même façon que précédemment une déclaration générique d'un service particulier peut être faite. Le paramètre *use* permet ensuite d'affecter à un autre service les caractéristiques de ce service générique. A chaque service peut être affecté une ou plusieurs stations.

5.6.5. Définition des plages de fonctionnement pour les différents services

Il est possible de définir des types d'intervalles de temps permettant de gérer le temps de fonctionnement de services, les périodes de présence ou d'absence des administrateurs, etc. C'est le fichier *timeperiods.cfg* qui héberge ces définitions. Chaque plage de fonctionnement est nommée pour être ainsi appelé lors des déclarations d'intervalles par exemple indiqués dans les fichiers de configuration présentés plus haut.

6. Nagios couplé à SNMP

Simple Network Management Protocol est un des protocoles (simples) de gestion de réseaux les plus utilisés, devenu de fait un standard quasi incontournable dans le domaine. Il s'appuie sur le protocole TCP/IP[12] ce qui peut expliquer le fait qu'il se soit propagé si rapidement. Trois versions se sont succédées : SNMPv1, SNMPv2 et SNMPv3. Plusieurs RFC[13] définissent ce standard, nous pouvons citer la RFC 1155SMI (Structure of Management Information), RFC 1156MIB (Management Information Base), RFC 1157SNMP Protocol, etc. SNMP est conçu pour répondre aux besoins pratiques et quotidiens de l'administrateur. Ce dernier a souvent besoin de connaître un certain nombre d'informations sur ses équipements. Sur un parc conséquent se déplacer peut vite devenir sportif, voire épuisant. Régler des paramètres de son bureau, tranquillement assis sur une chaise est donc particulièrement intéressant. Les systèmes utilisant SNMP possèdent deux éléments clés. Un agent logiciel, comprenant une base d'objets supervisés et de variables, qui fonctionne dans les stations gérées et une station de gestion (*manager*) qui contient le protocole et les

[12] *Transmission Control Protocol / Internet Protocol*
[13] *Request For Comments*, documents spécifiant les normes, standards ou protocoles.

applications de gestion. Cette station permet de récolter, d'analyser les données relatives aux équipements supervisés. Le protocole est asynchrone, de type question/réponse. Il peut faire à lui tout seul l'objet d'une étude, nous le présentons succinctement et ne nous lancerons donc pas dans les détails ; tout comme nous n'aborderons pas les questions de sécurité des différentes versions.

6.1. Le principe de SNMP

6.1.1. L'agent SNMP

Chaque équipement sur lequel intervient l'administrateur via SNMP doit disposer d'un agent SNMP. Il s'agit d'un serveur qui reste à l'écoute d'un port particulier : l'UDP 161. Le serveur répond aux requêtes qu'il reçoit (d'une entité autorisée). Soit pour agir localement sur un paramètre que l'administrateur souhaite modifier, soit pour retourner l'information demandée. L'agent pourra également, s'il a été configuré pour, envoyer de lui-même des alertes. Ces agents SNMP peuvent être placés sur des équipements terminaux (ordinateurs) mais également sur des équipements intermédiaires (routeurs, switchs, ponts, etc). De plus en plus d'équipement intermédiaire dispose d'agents SNMP. En revanche, l'agent doit être installé sur les stations (et serveurs). Sous Windows (2000 Pro, Server et XP) l'agent est incorporé, sous Unix existe net-SNMP ou ucd-SNMP. Les agents SNMP peuvent être paramétrés (la finesse dépendant du système) afin de créer des groupes de sécurité qui ont un accès en lecture seule sur les informations. D'autres en lecture/ écriture ou encore en écriture seules et sur certains paramètres seulement.

6.1.2. Station d'administration

L'administrateur dispose sur sa machine d'un outil appelé « manager ». L'outil fait office à la fois de serveur (puisqu'il reçoit les alertes) et de client (puisqu'il envoi des requêtes). Le « manager » reste à l'écoute sur le port UDP 162. L'administrateur peut donc en théorie observer le comportement de l'ensemble du réseau (que ce soit un LAN ou un WAN) depuis sa station. Des managers existent en version graphique (*openview*, *open eyes*, etc.) ou en version commande en ligne (sous unix, *SNMPwalk* est un des plus connus et celui que nous avons utilisé principalement durant nos tests). Des versions plus élaborées pouvant tirer des graphes, des statistiques, afficher la charge d'interfaces réseaux existent également. L'application *getif* gratuite sous Windows rassemble par exemple a elle seule de façon plus ergonomique, les fonctionnalités de SNMPwalk, SNMPget, SNMPset et SNMPtranslate disponible sous unix avec le *package* net-SNMP.

6.2. Le protocole SNMP

6.2.1. Commandes du protocole dans sa première version

La version 1 comporte 5 commandes principales :

- get-request : le manager demande une information à l'agent ;
- get-next-request : le manager demande l'information suivante à l'agent ;
- set-request : le manager met à jour une information sur un agent ;
- get-reponse : l'agent répond à un get-request ou a un set-request ;
- trap : l'agent envoie une alarme au manager.

Nous rappelons que l'agent utilise le port 161 et le manager le port 162.

6.2.2. Nécessité d'une standardisation des informations

Comme on le constate le protocole est simple. Cependant, étant un protocole Internet, il doit être utilisable sur des plates formes hétérogènes (matériel et systèmes d'exploitations !). Il a donc fallu lui associer des standards pour la gestion, le stockage des informations transportées et utilisées par le protocole. Un élément indissociable de SNMP est donc la MIB pour Management Information Base, une base d'informations de gestion.

6.3. Management Information Base

SNMP doit permettre de retrouver les informations et d'agir sur les paramètres des équipements de façon indépendant du matériel et du logiciel. La MIB se présente comme cette base de donnée normalisée qui permet de lire et d'écrire sur les équipements distants (de façon également standard). C'est à l'agent lui-même de traduire les informations entre SNMP et la plate forme de supervision.

6.3.1. Syntaxe des informations de gestion (SMI)

Cette syntaxe définit comment chaque élément est représenté dans la base d'information de gestion. C'est en fait un sous ensemble de la norme ASN.1[14] : seuls quatre types de données sont utilisées :

- Integer : ce sont des valeurs entières ;
- Octect String : c'est une suite de zéro ou de plusieurs octets pouvant accepter des valeurs comprises entre 0 et 255 ;
- Object Identifier : suite de numéros référençant un objet enregistré par une autorité compétente ;
- Null.

Il y a également deux types de données structurées utilisables que sont les listes et les tables à deux dimensions.

6.3.2. Structure de la MIB

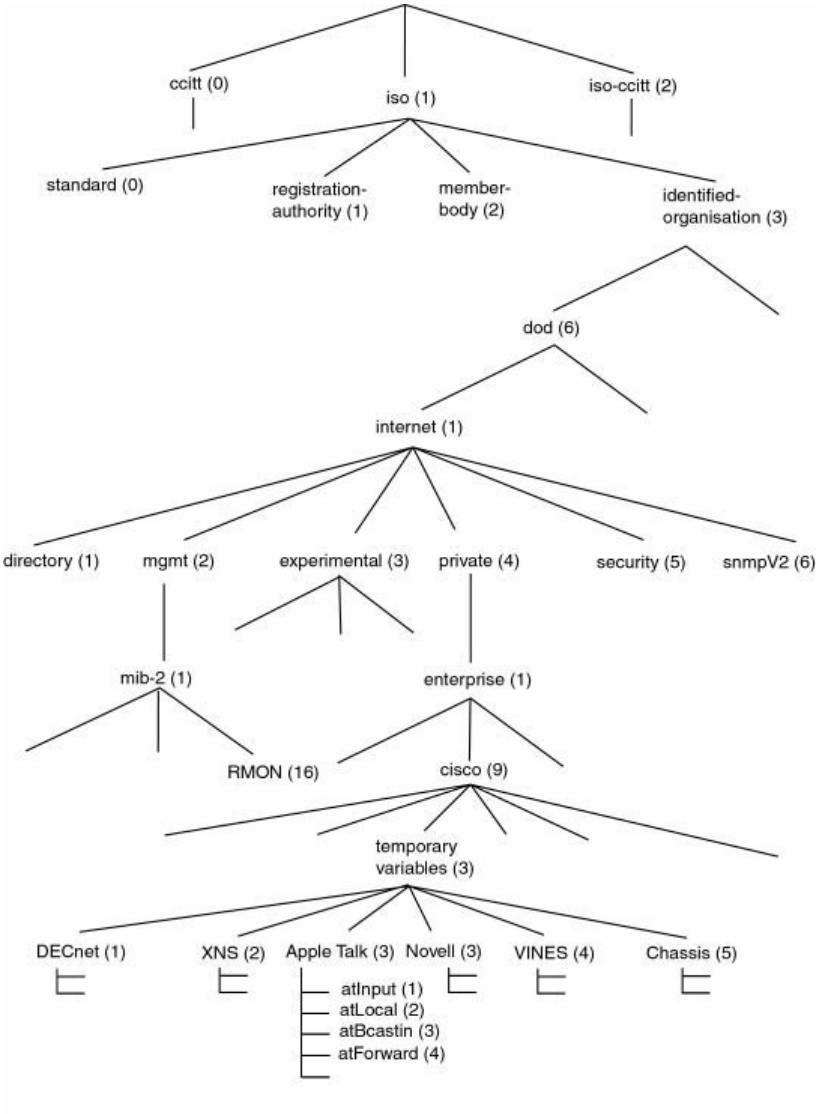
La *Management Information Base* contient l'ensemble des variables supervisées. Elle met en œuvre un jeu de variable traitant à la fois du matériel et du logiciel. Les noms des éléments de la base suivent une hiérarchie définie par l'ISO[15] pour le nommage des objets réseaux ; les objets de la base sont ainsi classés en une structure de classes d'objets. Un premier niveau de classes d'objets de la base (MIB 1) comprend, entre autres, les groupes suivants :

- *system* : informations relatives au nœud lui-même ;
- *interfaces* : pour les ports et les interfaces réseaux ;
- *address translation* : pour la traduction des adresses ip ;

Seuls certains groupes d'objets peuvent être implémentés suivants le type d'équipement à gérer. De la même façon, des MIB propriétaires ont vu le jour ; elles comportent des objets spécifiques.

[14] *Abstract Syntax Notation*
[15] *International Organisation for Standardization*

L’organisation hiérarchique se fait de la même façon que les domaines d’Internet. Elle contient une partie commune à tous les agents SNMP, à tous les agents d’un même type de matériel ensuite, une partie spécifique aux constructeurs enfin. Les appellations des diverses rubriques qui composent la MIB sont également normalisées et permettent théoriquement une meilleure lecture pour l’œil humain. En effet, SNMP ne se sert que de l’indexation faite de ces appellations. Chaque niveau hiérarchique est indexé numériquement.



Pour un réseau IP c'est le nœud dod et le sous nœud Internet qui seront utilisés. Il s'agit donc de passer par les nœuds « .1.3.6.1 » ce qui correspond à « *.iso.org.dod.internet* ». Pour désigner une feuille on rajoute un 0 pour indiquer le scalaire (la valeur), n si il s'agit d'un tableau de scalaires. La notion de chemin absolu/relatif est retrouvée par le «**.**» placé avant la suite de nœuds. « **.1.3… »** désigne un chemin absolu, **« 1.3… »** désigne un chemin relatif. Passer de la version numérique à la version textuelle parfois plus parlante peut se faire avec des outils comme SNMPtranslate ou même des options de SNMPwalk.

6.4. SNMP et Nagios

Comme nous l'avons expliqué dans les rubriques ci-dessus, SNMP peut être un outil intégré à Nagios. Nous avons vu que des *plugins* spécifiques *NRPE, NSCA* fourni par *Nagios* permettent de ne pas utiliser SNMP. Cependant, l'utilisation de ces *plugins* impose l'installation d'un agent spécifique (les agents SNMP étant par défaut sur tous les commutateurs) sur chaque machine supervisée ! Des *plugins* utilisant SNMP sont également fournis dans l'installation par défaut de Nagios. Nous pouvons citer *check_snmp* ou *check_ifoperstatus* par exemple. Le second va nous permettre par exemple de remonter l'état, le statut (up ou down) d'un port précis d'un commutateur donné. Ce dernier *plugin* par exemple utilise la commande *snmpget* sur l'OID « .1.3.6.1.2.1.2.2.1.8.x » ou x est le numéro du port à considérer.

7. L'utilisation de Nagios par le CERMA

7.1. Les services monitorés

Le service de base utilisé sur l'ensemble du parc est le test de leur présence sur le réseau. Un ping, via le *plugin check_ping* est effectué à intervalles réguliers. Sur certaines machines, les serveurs linux notamment, nous pouvons tester également que les services ftp, http, ssh soient fonctionnels. Plusieurs *plugins* offert dans la distribution par défaut sont des *plugins* locaux. Ainsi tester l'espace disque restant où la mémoire utilisée en utilisant ces *plugins* se fait localement ; le test ne peut donc pas être effectué de la machine de supervision par défaut. Nous n'avons pas souhaité installer les *plugins NRPE* ou *NSCA* dans un premier temps. Une autre solution sera d'écrire un *plugin* spécifique utilisant SNMP (activé sur les machines concernées) afin de récupérer ces informations.

7.2. La génération des fichiers de configurations

La génération manuelle des fichiers de configuration est fastidieuse. Il faut définir chaque station. Définir le groupe auquel elle appartient. Définir ensuite les services de cette station. Et ce, pour toutes les stations du parc à superviser ! Un utilitaire existe nmap2nagios[16] qui permet de créer des fichiers de configuration à partir d'un fichier xml généré par Nmap. Nmap génère le fichier xml en scannant le réseau. Problème avec le réseau du CERMA : les machines sont la plupart du temps éteintes ! Seules une vingtaine de machines ont donc été scannées lors de l'utilisation du script, les serveurs linux. Cette solution n'est donc pas adaptée aux conditions pratiques du parc du CERMA. Il a fallu trouver un autre moyen de générer le fichier xml puis dans la foulée nous avons personnalisé la génération des fichiers de configuration. Pour ce faire, l'administrateur du CERMA a adapté un script perl écrit pour le

[16] Disponible sur http://www.nagiosexchange.org

plan textuel existant afin de fournir le fichier xml et nous avons développé quelques lignes de code pour générer à partir du xml produit par son script les six fichiers de configuration directement lié à la gestion du parc.

7.3. L'accès aux informations

L'accès aux informations fournies par Nagios se fait classiquement par l'interface web qui est fourni avec l'installation par défaut décrite en détail (rubrique par rubrique) dans la documentation. En lisant cette documentation ou en manipulant quelques secondes l'interface nous découvrons comment n'afficher que les informations par groupe de machines (seulement les données concernant les imprimantes ou les serveurs linux) ou par services, etc. La cartographie et autres informations sont donc accessibles via un navigateur. L'accès à ces données est protégé par un mot de passe. L'authentification se fait en utilisant l'annuaire LDAP.

7.4. Affichage des ports

Nous avons utilisé une astuce pour afficher les ports sur la cartographie. Chaque port est défini pour Nagios comme une station normale, sans adresse spécifique. Nagios afin de permettre un meilleur affichage offre la notion de parent pour une machine donnée. Ainsi une machine connecté à un commutateur aura pour parent ce dernier. De cette façon, l'affichage montre clairement que la machine dépend du commutateur. Nous avons donc utilisé cette fonctionnalité en insérant le port (considéré comme une station par Nagios) entre la machine et le commutateur du port auquel elle est connectée. Ainsi, la machine a pour parent un port qui a pour parent son commutateur. Le plan permet donc de savoir graphiquement quelle machine est derrière quel port.

8. Bilan personnel

8.1. Difficultés rencontrées

La principale difficulté a d'abord été le système d'exploitation. Nous ne connaissions linux que de façon superficielle, les commandes de base tel que *ls, cd,* etc ; et uniquement quelques fichiers comme le *.bashrc*, *lilo.conf,* etc. Il nous a donc fallu passer une barrière en nous habituant au nouvel environnement, comprendre comment était organisé les fichiers, ou se trouvait les fichiers de configurations, les fichiers de logs, les scripts de lancement de services... prendre nos marques sur la distribution. Les premiers problèmes ont donc été les outils avec lesquels nous travaillions. Une fois nos raccourcis placés, l'environnement mieux maîtrisé, nous avons après avoir essuyé quelques essais laborieux de compilation manuelle de Nagios découvert puis compris le fonctionnement des utilitaires tels que *yum*, *synaptic*, *apt-get*. A partir de ce moment là seulement, l'outil linux n'a plus été un problème et nous ne perdions plus de temps à gérer les dépendances : nous pouvions nous concentrer sur la configuration de Nagios.

8.2. Compétences utilisées

Connaître le principe de configuration par fichier texte et savoir lire une documentation (en ligne ou non) auront été un plus. Nos quelques connaissances en Shell nous ont permis de faciliter nos démarches dans l'automatisation de certaines tâches récurrentes. Les acquis de la formation Java nous ont permis de développer quelques lignes de codes permettant de générer automatiquement les fichiers de configurations de Nagios. Enfin, un soupçon de bon sens, de persévérance et de patience ont été les derniers ingrédients pour cette recette.

8.3. Compétences acquises

Sans se poser de question : linux. Nous pouvons maintenant dire sans complexe que nous avons de bonnes bases sous linux après un mois d'utilisation quotidienne du système d'exploitation ; nous avons testé deux distributions différentes qui sont la Fédora Core 2 et la Ubuntu. Quelques astuces de programmation (shell et java 1.5) nous ont également été fournies généreusement. Nos connaissances théoriques sur le protocole SNMP ont été améliorées et la vie quotidienne en entreprise au côté d'un administrateur réseau nous a confronté à des cas pratiques de problèmes/solutions qui n'ont fait qu'augmenter notre expérience. Nous avons également appris à livrer une solution de déploiement à partir d'une installation par défaut. La solution Nagios mise en œuvre pour le CERMA peut être installée en tapant trois commandes. Une personne ne connaissant pas Nagios peut ainsi aisément mettre en place la solution. Un autre avantage d'un déploiement efficace est que ces trois commandes sont également ce qu'il faudrait faire pour monter en quelques minutes une autre machine de supervision en cas de défaillance de la première. Enfin, accessoirement, nous avons également intégré après avoir quelque peu participé à la migration de commutateurs du parc combien la pose professionnelle du câblage était importante et nécessaire dans un réseau☺.

9. Conclusion

Nous avons appris bien des choses durant ce mois passé très vite. Cependant, nous avons l'impression en sortant du stage que tout reste à faire ! La solution livrée est fonctionnelle, elle permet de visualiser en temps réel l'état des stations et des quelques services mis en place sur certaines machines. Il y a cependant un problème qui n'a pas encore été réglé. Certaines machines n'ont pas d'adresse ip définies lors de la génération des fichiers de configurations, il s'agit des machines dont l'adresse est affectée dynamiquement. Les outils standard comme le *ping* fournis avec Nagios, à la base créée pour la supervision de serveurs ayant donc des adresses fixes, ne fonctionnent pas sans adresse ip. Nous avons alors utilisée une astuce qui fonctionne sur quelques machines : utiliser le nom netbios. Pour toutes les machines référencées, la résolution d'adresse se fait bien, le ping est alors fonctionnel. En revanche, l'état actuel de la solution est incapable de superviser les machines non référencées et sans adresse ip, Nagios les considère logiquement comme en état critique puisqu'elles ne répondent pas au *ping*. Une solution serait alors d'écrire (ou de trouver) un *plugin* qui détermine dynamiquement l'adresse ip d'une machine donnée à partir de son nom ou de son adresse mac en utilisant *rarp* par exemple. Une autre fonctionnalité intéressante serait d'utiliser *nagios graph*[17] et des *plugins* comme *check_traffic* ou autre afin de récupérer des informations sur les débits circulants sur les différents ports des commutateurs ; il serait alors possible d'avoir en temps réel des courbes indiquant des données sur le trafic du réseau. Ce sont principalement les deux points que nous aurions développés sur un stage plus long.

10. Annexe

Sont fournis en annexe avec ce rapport :

- un document orienté installation étape par étape de Nagios présentant la mise en œuvre de la solution pour le CERMA ;
- les sources de l'application java générant automatiquement les fichiers de configurations et la documentation associée ;
- un Makefile et une archive permettant une installation aisée : l'utilisateur tape simplement « *make install* » après l'installation de Nagios par défaut, elle-même effectuée à l'aide de deux commandes à partir d'installeur tel que *yum* ou *apt-get*.

[17] http://sourceforge.net/projects/nagiosgraph/