Sommaire

Index des illustrations

Remerciements

Je tenais tout d'abord à remercier mon tuteur de stage monsieur François Chassaing pour m'avoir encadré tout au long de cet apprentissage du monde professionnel. Je remercie aussi l'ensemble des membres de Weborama pour m'avoir accueilli dans d'excellentes conditions et m'avoir permis de réaliser une expérience professionnelle enrichissante, complète et exemplaire. Un merci particulier à toute l'équipe technique avec laquelle j'ai travaillé et qui m'a supporté (pas toujours évident :-p) tout au long de ce stage : Julien (même si tu ne supportes pas la bonne équipe), Gilles, Frédéric, Pierre-Antoîne, Nicolas (merci pour la bobox !), Enric, Xavier et Sébastien.

En espérant leur avoir apporté autant qu'ils m'ont appris...

Je profite de cette page pour remercier tous les gens qui ont cru en moi, qui m'ont aidé et qui m'ont poussé à poursuivre mes études jusqu'à ce Master : Guillaume, Pierre, Vassili, Julien, Hugues, Lounes, Frédéric, les Ludos, Jean Philippe, Jean Baptiste, Francois Xavier, Hakim, Thomas, Félix, Leirn, Christian, Robby, Sylvain, Alexis, Laura, Florent, Stefan et Ryan. J'en oublie sûrement mais de toute façon cette page ne sera jamais lue par les intéressés :-D .

Un merci à tout mes différents collègues de promotion avec qui j'ai travaillé et un merci particulier à Pierre et Clément (Projet WallyGatoStar), Morgan, Julien, Kamel, Chloé, Karim et Thomas.

Je remercie également toute l'équipe pédagogique de Paris VI qui a su m'accompagner parfaitement durant les deux années où j'ai évolué à Jussieu.

A Thalie...

Summary

I achieved my work placement at Weborama. Weborama is a company that measures traffic and performance of web sites. My work project was to search, test and install a network and system supervision tool which might be able to replace the actual tool : BigBrother. There were two reasons to change the current tool : it was too limited (fonctionnalities, usage...) and Weborama was looking for something new, more powerful and with more fonctionnalities. After a theoritical evaluation and a comparison between several solutions, my choice did go towards Nagios. Nagios is an open source supervision tool, totally free, it is developped by Ethan Galstad. Beside, there is an important user's community, which is important to ensure maintainability. Nagios brings more fonctionnalities than BigBrother : email or sms problem notifications, possibility to disable a ressource check or to acknowledge a problem via the web interface, a really simple way to write and develop handmade plug-ins...

The specifications brought up several points that the tool was not able to accomplish alone : graph generation, checking storage results in a database and SNMP polling to obtain bandwidth load. Even if there are many solutions to answer to the needs described in the specifications, my choice has been oriented by easy means to bridge the gap with fully-compliant solution : Nagiosgraph for the graph generation, RRDTool to save data in a database (rrd base) and the plug-in Check_Traffic to determine the broadband activity.

Just before installing those set of softwares and plug-ins, I carried out a set of tests to adapt the control policy already used by BigBrother. The goal of this test session was to learn how to install all those set of softwares and to write an installation procedure. During this tests, my tutor (Mr Francois Chassaing) brought to my attention the fact that some plug-ins were not really adapted to obtain an optimized configuration of Nagios. This incites me to modify some plug-ins and to make them "smarter" than they were previously and totally autonomous (the plug-ins are able to determine all the thresholds on the host itself). When the plug-ins development was finished, my tutor and myself began the installation of this set of softwares and plug-ins.

As, the installation has not been planned just after the test session, I worked on others projects. The first one was to install a system to replicate MySQL servers. The first goal was to understand the technology and the mechanims of this kind of system. The second step was a test session which allowed me to
write an installation procedure of this system. This procedure will be adapted for each server where the replication is going to be installed. The second project was the elaboration of a backup policy for the user data. Before this project, backups were conduected heterogenously (not all users were not backing up their data... which others did not do it regularly). So I had to propose a solution to normalize this procedure which needed to be easily configurable for each user. My solution was a script developped in Batch DOS interacting directly with the Windows backup tool : Ntbackup. This script makes a data backup, on a distant Samba server, for each system init boot. Then a weekly backup was done on a DVD.

Moreover I did several administrative tasks such as mail account managing, configuration of printer servers... The work placement, beginning on April the 18th, took place at Paris (19 district). During this experience, I worked on a lot of different technologies, this is why it was a really interesting period. After being integrated into the technical team, a real exchange between the staff and myself. This relation brought to me part of the knowledge of my collegues, giving me the feeling of learning something every day.

Humanly, professionnaly and technicallly, I lived during this 6 months a really positive and enriching expereince. I especially had the satisfaction of bringing something to Weborama since all the projects I have conducted were totally accomplished !

I/ Présentation du stage

L'objectif premier de ce stage était l'étude et la mise en place d'une solution de supervision afin de pouvoir contrôler simplement et efficacement l'ensemble du réseau et des systèmes de Weborama. Avant ce stage, la supervision de la société reposait entièrement sur le logiciel BigBrother. Cette solution freeware et OpenSource permet de superviser les services réseaux suivants : FTP, HTTP, HTTPS, SMTP, POP3, DNS, Telnet, IMAP, NNTP et SSH. De plus elle assure le contrôle des ressources telles que la charge CPU, espace disque disponible... Big Brother est composé de scripts C et bash et repose sur une architecture client/serveur.
Toutes les informations récupérées sont affichées dans une page Web générée par BigBrother qui permet de visualiser facilement et rapidement l'état actuel du réseau par le biais d'un code de couleur : vert si tout est OK, orange dans le cas d'une alerte ou rouge si une panne a été détectée. Le fond d'écran reprend la couleur de l'état le plus alarmant.

Voici une capture d'écran de l'interface que BigBrother offre :

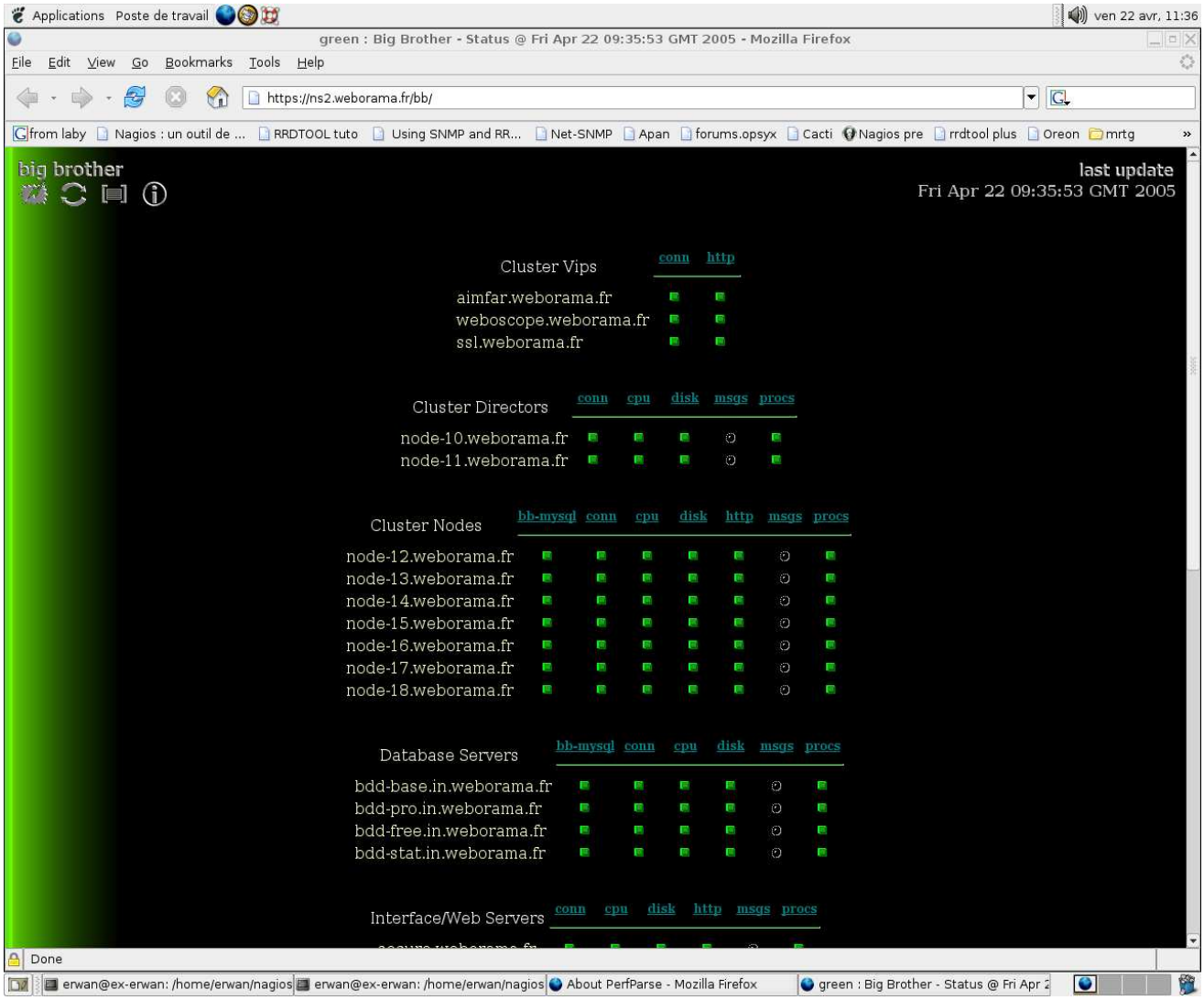

Fig. 1 : Capture d'écran de Big Brother (green is good !)

On peut voir l'état actuel des services et des ressources surveillées pour chaque serveur.

Cependant Big Brother reste trop rigide pour la création de nouveaux mécanismes de contrôle par exemple. De plus le paramétrage des notifications n'est pas vraiment complet ce qui empêche d'avoir une politique de notifications pertinente, efficace et adaptable (comme notifier uniquement si un service ne répond au bout de la troisième fois...). C'est donc pour ces diverses raisons que ce projet a vu le jour sous la forme d'un stage.
Le but est de proposer une solution qui regroupe les besoins suivants :

- la solution doit être en mesure d'offrir au moins les mêmes services que la solution actuellement déployée : BigBrother,
- permettre une meilleure réactivité vis à vis d'un problème, on parlera même de pro-activité dans le sens où l'administrateur doit pouvoir être averti des problèmes avant même que l'utilisateur ne s'en rende compte afin d'assurer un rétablissement plus rapide de la situation,
- permettre une politique de notifications complète et entièrement paramètrable,
- assurer une bonne gestion des logs et historiques avec possibilité de les sauver en base de données,
- possibilité de faire du polling SNMP en particulier pour contrôler les débits d'échanges de données au niveau de routeurs/switchs par exemple voire même d'un hôte,
- permettre la représentation graphique des données récoltées,
- et être gratuit !!!

Le stage ne durant que 6 mois, il est bien évident que cet objectif n'est pas le seul projet prévu. Suivant l'avancée du premier projet ; de nouveaux travaux, plus ou moins importants (tel que l'étude de la réplication de serveurs MySQL, la création d'un système de backups automatiques des données utilisateurs...), me seront confiés par la suite. Le stage est par conséquent très orienté systèmes et réseaux, et il me permettra donc d'aborder professionnellement l'ensemble du travail d'un administrateur.

Le stage se déroule intégralement au siège de Weborama situé à Paris où j'ai été intégré à l'équipe de techniciens et de développeurs et où je suis considéré comme un réel membre de cette équipe.

II/ Présentation de l'entreprise

2-1/ Présentation générale et historique

Weborama a été créé en 1998 par François Chassaing (Directeur Associé et Directeur technique) Sunny Paris (Directeur Associé et Directeur Recherche et Développement) et Rodolphe Rodrigues (Directeur Associé et Président de la société), scientifiques de formation et passionnés d'Internet dès la première heure.

En 1999, Weborama développe Weboscope™, l'outil de mesure d'audience des sites Internet. Cette solution "site centric" (dénomination à une méthodologie d'audit des sites Internet qui utilise des marqueurs invisibles directement placés sur les sites étudiés ce qui permet une réelle mesure du trafic Internet en contradiction avec les logs ou les méthodes "user centric"), devenue la référence sur le marché de la mesure d'audience, a connu de nombreuses évolutions. Weboscope™ est labellisé OJD (label délivré par Le Bureau Internet Multimédia qui certifie la fréquentation des sites Web sur la base du critère de la visite et du visiteur unique) depuis 2000. Aujourd'hui, Weboscope™ est une large gamme de produits qui répond à tous les besoins sur Internet, Intranet, Extranet et sur les nouveaux médias (I-mode, WAP, Web-TV, etc.) :

- Weboscope™ Audience : analyse de l'audience des sites Internet.
- Weboscope™ Intranet : analyse de l'audience des sites Intranet.
- Weboscope™ Sur Mesure : développement de solutions spécifiques.

Weborama a développé Weboscope™ Performance pour analyser l'efficacité et le R.O.I. (retour sur investissements) des opérations marketing sur Internet : stratégies e-commerce, campagnes de bannières, programmes d'affiliation ou de partenariats, référencements payants, achats de mots-clés, etc. Cette solution a été conçue en 2000 pour répondre aux attentes des professionnels du marketing, de la communication et de la publicité on-line qui souhaitent optimiser leurs stratégies e-marketing en termes d'efficacité et de retour sur investissement.

Weboscope™ Performance : analyse des opérations e-marketing.

En 2000 Weborama a obtenu de Start Up Avenue 530 000 € de financement. En 2001 l'ANVAR investit 150 000 € pour le développement de nouvelles technologies publicitaires : le projet Wousdat™.

En 2002 Weborama met au point Wousdat™, l'outil de ciblage publicitaire. Développée avec le soutien de l'ANVAR, cette solution repose sur des technologies de profiling et d'adserving. Wousdat™ offre la possibilité d'augmenter considérablement l'efficacité des actions de publicité on-line et permet d'adapter en temps réel la diffusion du message publicitaire avec les cibles identifiées.

Voici une frise chronologique reprenant les évènements marquant de la société :

Fig. 2 : Chronologie de Weborama

2-2/ Le marché

Le développement d'Internet a conduit les professionnels à incorporer dans leur stratégie en ligne des procédés de plus en plus critiques pour leur business tels que la vente en ligne, le marketing direct, la publicité ainsi que beaucoup des services destinés aux clients (support, aide en ligne, etc...) .
Les professionnels ont aujourd'hui besoin d'améliorer l'analyse de l'ensemble de leurs actions en ligne. Un des exemples les plus évidents réside dans la difficulté qu'ont les professionnels à mesurer l'impact direct de leurs actions marketing et/ou de leurs campagnes publicitaires pour augmenter leur taux de conversion.

Ils ont donc besoin d'outils simples et ergonomiques qui traquent le comportement des visiteurs sur un site et qui mesure leurs réponses à des actions de communication ou de marketing, à des modifications de contenu ou d'offres commerciales.
Dans cette optique, Weborama propose une gamme complète de services e-marketing de mesure d'audience à destination du grand public (www.weborama.fr) et des professionnels de l'Internet (www.weboscope.com). Weborama propose également un outil de mesure de la performance des actions de communication on-line qui délivre des informations essentielles pour optimiser les stratégies e-business en termes d'efficacité et de retour sur investissement (www.weboscope.com). Forte d'une équipe R&D dédiée, la société a mis au point des technologies de profiling et des outils de ciblage publicitaire performants pour répondre à tous les besoins de ses clients. Aujourd'hui 3 brevets sont en cours : le méga panel, le profiling comportemental et le traitement parallèle de l'information.

2-3/ Descriptions des services et des outils

a/ Les solutions payantes

Weboscope™ Audience est la solution de référence pour mesurer l'audience des sites et faire gagner du temps dans la gestion et l'optimisation des sites. Weboscope™ mesure l'audience d'un site quelqu'en soit le nombre de pages. Labellisé par l'OJD (cf. partie 2-4/ Références), il fournit aux départements marketing et opérationnel une double analyse, quantitative et qualitative, de l'audience d'un site : volume de trafic, profil sociodémographique des visiteurs, répartition géographique, comportement de navigation des internautes, provenance en accès direct par les moteurs ou vos partenaires, motif des visites, etc..
Les déclinaisons de Weboscope™ Audience ont été conçues pour répondre aux différents besoins marketing, on distingue deux grands types d'offres :
"Trafic" : solution d'analyse professionnelle : Avec Trafic 5.0, le concentré essentiel de Weboscope™ Audience, dispose d'un outil de mesure d'audience simple, efficace.
"Entreprise" : solution d'expertise e-marketing : qui offre une supervision des sites plus complètes que l'offre précédente (comme la remontée d'alerte automatique, rapport d'audience entièrement personnalisable...) .

Weboscope™ Performance est la solution d'analyse et de reporting pour suivre les opérations on-line du clic à la conversion : sites e-commerce, campagnes de bannières, programmes de partenariat et d'affiliation, référencements payants, e-mailing, jeux-concours, etc. Elle apporte un éclairage en temps réel sur l'efficacité et le retour sur investissement de toutes les opérations : les coûts d'acquisition, les taux de conversion, les taux de transformation, le panier moyen des internautes, etc.
Pour ce produit il existe deux types d'offres :
L'offre Performance : permet de mesurer tous les clics et toutes les transformations qui ont été générés par les campagnes on-line qu'un client a mis en place.
L'offre E-business : cette offre intègre les indicateurs dédiés au commerce électronique et à l'achat de liens sponsorisés. Elle permet évidemment de mesurer les clics et les transformations mais elle permet surtout d'analyser en détail le retour sur investissement de chaque opération (chiffre d'affaires, panier moyen, etc...) .

Profiling et Ad-Serving :Wousdat™ - AIMFAR : Weborama met à disposition des centrales d'achat et des annonceurs un outil d'Ad-Serving de dernière génération; l'AIMFAR. Couplé à Wousdat™ - une technologie exclusivement développée par Weborama - cet outil d'Ad-Sserving est en mesure de cibler les internautes selon des segmentations de profils traditionnels (homme/femme, tranche d'âge, activité, etc.) à partir des sites visites...

b/ Les solutions gratuites

Weborama offre aussi des services gratuits, classiques du Web, en échange de bannières publicitaires (sous forme de pop-up ou de slide-in) :

Weboscope™ Free vous apporte une vue complète du site des adhérents grâce à des indicateurs pertinents : volume de trafic par page, circulation des internautes sur les pages, comparaison du trafic d'une période sur l'autre. Weboscope™ permet aussi d'analyser le référencement du site : les moteurs de recherche qui apportent le plus de trafic, les meilleures requêtes tapées pour accéder au site. Weboscope™ permet de plus de connaître le profil des internautes (exclusivité Weborama).

Promotion du site : Grâce aux "Classements" et aux "Nouvelles" les sites peuvent être visibles par l'ensemble de la communauté de Weborama.fr qui représente 43520 sites inscrits et plusieurs millions de visiteurs par mois. Weborama.fr est l'annuaire des meilleurs sites francophones. Les classements sont effectués par thématique selon la pertinence des sites, en termes de trafic et de popularité auprès des internautes.

Redirection / Nom de domaine gratuit : Weborama.fr met à disposition, de ses membres, gratuitement son service « Rue du Net » de nom de domaine. La redirection Web permet aux internautes de visiter un site avec une adresse Weborama.fr et non avec celle de l'hébergeur du site. Les url's sont originales, personnalisables, et disponibles immédiatement. Les serveurs de Weborama renvoient automatiquement toutes les requêtes des internautes vers les sites Web. De plus lorsque les membres changent d'hébergeur, les mises à jour s'effectueront immédiatement et automatiquement.

c/ Les solutions sur-mesure

Les entreprises des secteurs de la Banque et de l'Industrie ont par exemple des exigences de sécurité et de confidentialité très importantes auxquelles Weborama est parfaitement en mesure de répondre. En effet, la société a su développer des solutions personnalisées s'intégrant aux systèmes d'informations de ses clients : la solution internalisée ou « boîte noire ».
Les solutions de Weborama se basent sur des technologies qui obligent à avoir du trafic circulant entre les sites et les serveurs de Weborama. Mais pour respecter la contrainte de confidentialité, ce genre de technique ne peut être mise en place. Weborama a alors développé un système permettant d'internaliser tous les services et de les adapter entre autre au site intranet. Ce système est appelé la « boîte noire » qui regroupe les solutions d'analyse de trafic, de mesure d'audience et de performance. La « boîte noire » est une exclusivité Weborama qui permet de répondre aux exigences d'une certaine catégorie de clients comme les banques par exemple.

2-4/ Références

- Weborama, reconnue société « innovante », est soutenue par l'ANVAR (Agence Nationale de Valorisation de la Recherche) qui fournit conseils, et moyens financiers pour favoriser la valorisation des résultats de la recherche scientifique, soutenir le développement industriel, la croissance et l'aboutissement de projets. L'ANVAR soutient et participe au développement de Weborama Wousdat™.
- Labellisé par Diffusion Contrôle : Les outils de mesure d'audience développés par Weborama sont labellisés par OJD-Diffusion Contrôle depuis 2000. Organisation professionnelle tripartite regroupant les principaux acteurs du marché Internet français (annonceurs, éditeurs et prestataires Internet), OJD a vocation de contrôler et certifier la mesure de la fréquentation. Cette mesure peut s'appliquer à tous les sites édités sur le territoire français, de tous les sites dont l'éditeur est français et de tous les sites francophones. Dans cette optique, OJD a défini en collaboration avec les principaux acteurs du marché, dont Weborama, un Cahier des charges que les outils de mesure de trafic de sites Internet doivent rigoureusement respecter. Pour être labellisés par OJD-Diffusion Contrôle, les outils de mesure d'audience doivent notamment respecter les points suivants :
  - Utiliser la méthodologie "site centric"
  - S'affranchir des biais induits par la mémoire cache (proxy, navigateur)
  - Passer au travers des Firewall d'entreprises
  - Exclure les robots
  - Filtrer les IP internes

La société continue de collaborer avec OJD afin d'améliorer le Cahier des charges notamment en y incluant les notions de visiteurs uniques et la certification des "newsletters" électroniques.

- Weborama est devenue un acteur majeur du marché européen des services marketing sur Internet. La société compte de nombreuses références clients qui lui renouvellent chaque année leur confiance. Weborama a mis en place un large réseau de partenaires en France et dans le monde : la société possède des clients dans plus de 25 pays et elle est plus particulièrement active en Grande-Bretagne, au Canada/Québec, au Maroc, en Italie, au Portugal, en Tunisie, et en Espagne. Avec plus de 40 000 sites audités à travers le monde, Weborama garantit une mesure fiable des sites Internet comme des campagnes promotionnelles, dans 5 langues et en temps réel.
- Noos, BNP Paribas, Société Générale, PSA Peugeot Citroën, Essilor, Arte, MSN sont les quelques grands noms (divers par leurs activités, ou leur structures) qui font partie des nombreuses références de Weborama.

2-5/ Résumé et Chiffres

Pour résumer, Weborama c'est :

- 15 salariés
- une société créée en 1998 qui est bénéficiaire depuis 2002
- plus de 300 clients et toutes offres confondues Weborama audite plus de 50 000 sites
- six ans d'expérience dans le domaine des nouvelles technologies. C'est ainsi que Weborama a acquis une haute expertise technologique pour répondre aux besoins marketing et technique de ses clients.
- forte d'une équipe R&D dédiée, la société a mis au point des technologies de profiling et des outils de ciblage publicitaire performants. Aujourd'hui 3 brevets sont en cours : le méga panel, le profiling comportemental et le traitement parallèle de l'information
- avec une progression de 31% sur 2003, le chiffre d'affaires s'établit en 2004 à 1,82 million d'euros. La société multiplie par deux ses revenus avec un bénéfice net de 700 K€ en 2004.

III/ Analyse du projet

3-1/ Rappel : les objectifs de la supervision

a/ Qu'est-ce que superviser ?

Superviser c'est contrôler et réviser un fait (définition du *Petit Larousse – 1991*).
Cette définition peut tout a fait s'appliquer au monde informatique. L'étude avancée de la conception d'un système informatique permet d'éviter la plupart des problèmes réseaux ou systèmes. Cependant il est impossible de certifier que le système est infaillible mais il est tout de même important de rester réactif face à un éventuel problème ! C'est donc dans cette optique que des outils de supervision ont été développés.

b/ Que peut-on superviser ?

A priori il est possible de superviser toutes sortes d'équipements :

- Le réseau et ses équipements
- Les serveurs
- Les périphériques
- Les applications
- Le trafic
- jusqu'à la machine à café si cette dernière est reliée au réseau !

La supervision informatique permet donc de superviser l'ensemble du Système d'Information d'une entreprise.

c/ Pourquoi supervise t on ?

L'informatique prend une place de plus en plus importante dans le monde de l'entreprise : on peut aisément comparer son importance à celui du coeur pour un être humain. Si le coeur cesse de battre, l'homme tragiquement meurt, de la même manière si le système informatique de l'entreprise tombe aucun mail ne peut circuler, aucune connexion vers les sites extérieurs (clients ou vendeurs) ne peut être réalisée... ce qui entraîne nécessairement une baisse notable de la performance et du chiffre d'affaires qui peut aboutir à la faillite de la société. Pour des entreprises comme Weborama, dont l'activité repose entièrement sur le net, superviser reste encore plus critique puisque le matériel doit être à tout moment accessible depuis l'extérieur. De plus avec l'accroissement incessant des réseaux d'entreprise, il est important d'avoir des outils capables de surveiller et d'alerter. Ces outils permettent donc aux équipes techniques d'avoir une réactivité plus forte lorsqu'un problème survient : dès qu'un contrôle remonte un état suspect, les techniciens peuvent directement agir et ne pas attendre que le problème soit découvert par un utilisateur par exemple.
Afin de bénéficier d'une surveillance permanente et efficace de l'ensemble de nos réseaux informatiques, il est nécessaire de confier la tâche de surveillance à un logiciel, qui aura pour tâche de relever à intervalles réguliers toutes les caractéristiques du système. Les administrateurs pourront ainsi bénéficier d'une surveillance permanente, et locale à chaque système informatique installé, nous permettant d'agir dès qu'un problème se présentera.
La supervision permet donc une gestion plus aboutie et complète d'un réseaux informatiques et de ses divers équipements. De plus une certaine crédibilité est à ajouter à l'équipe technique pour leur réactivité face aux problèmes !

3-2/ Les solutions possibles

Rappelons le cahier des charges pour le choix de l'outil de supervision :

- la solution doit être en mesure d'offrir au moins les mêmes services que la solution actuellement déployée : BigBrother,
- permettre une meilleure réactivité vis à vis d'un problème, on parlera même de pro-activité dans le sens où l'administrateur doit pouvoir être averti des problèmes avant même que l'utilisateur ne s'en rende compte afin d'assurer un rétablissement plus rapide de la situation,
- permettre une politique de notifications complètes et entièrement paramètrables,
- assurer une bonne gestion des logs et historiques avec possibilité de les sauvegarder en base de données,

- possibilité de faire du polling SNMP en particulier pour contrôler les débits d'échange de données au niveau de routeurs/switchs par exemple voire même d'un hôte,
- permettre la représentation graphique des données récoltées,
- et être gratuit !!!

De nombreux produits pour la supervision existent sur le marché : Tivoli d'IBM, OVO et NNM d'HP, Patrol de BMC, COABA d'Alcove ou encore Ganglia développé par l'université de Berkeley. Cependant parmi toutes ces solutions aucune ne répond entièrement aux besoins cités plus haut :

➔ Les solutions proposées par Alcove, IBM, HP ou Patrol sont toutes payantes (compter près de 9 500 euros pour la solution la moins chère).

➔ La solution Ganglia ne permet pas de gérer les notifications et n'assure pas la gestion d'un historique temps réel.

Le choix s'est donc porté naturellement sur Nagios qui correspond parfaitement aux besoins de Weborama et qui possède en plus de nombreuses documentations et d'une communauté importante d'utilisateurs et développeurs.

3-3/ La solution proposée : Nagios

a/ Description

Nagios est un outil de monitoring pour Linux. Il repose entièrement sur le moteur NetSaint, un ancien système de supervision. La première version de Nagios est parue en Mai 2002 et a été entièrement conçue par Ethan Galstad. Puis plusieurs versions se sont succédées jusqu'à la version 1.2 première version officielle de Nagios stable. Depuis peu de temps la version 2 est disponible mais toujours en version beta. Sur le site officiel de Nagios (www.nagios.org) la version 2.0b3 est téléchargeable depuis Mars dernier.

Voici un résumé imagé de la chronologie des versions de Nagios :

Fig. 3 : Chronologie de Nagios

Globalement Nagios est un simple point de centralisation, en effet il ne dispose pas de mécanismes internes pour vérifier l'état d'un service ou d'un hôte. Pour accomplir ce type de tâche Nagios doit utiliser des petits greffons appelés plug-ins. Les plug-ins sont des programmes externes (écrits en shell, perl, C...) qui peuvent être exécutés en ligne de commandes. Ils permettent de renvoyer à Nagios des résultats ou des états à propos d'un hôte ou d'un service particulier. Ils doivent respecter un certain format de réponse et d'affichage afin de faciliter l'intégration de ces derniers dans Nagios.

Nagios est donc un noyau (plus souvent appelé core) qui, une fois configuré, gère l'exécution des plug-ins et l'analyse des résultats.

Nagios offre les services suivants :

- Surveillance des services réseaux (SMTP, POP3, HTTP, NNTP, PING, etc.)
- Surveillance des ressources des hôtes (charge processeur, utilisation des disques, etc.)
- Système simple de plug-ins permettant aux utilisateurs de développer facilement leurs propres vérifications de services.
- Parallélisation de la vérifications des services.
- Possibilité de définir la hiérarchie du réseau en utilisant des hôtes "parents", ce qui permet la détection et la distinction entre les hôtes qui sont à l'arrêt et ceux qui sont injoignables.
- Notifications des contacts quand un hôte ou un service a un problème (via e-mail, pager, ou par une méthode définie par l'utilisateur)
- Possibilité de définir des gestionnaires d'évènements qui s'exécutent pour des évènements sur des hôtes ou des services, pour une résolution des problèmes pro-active
- Rotation automatique des fichiers log
- Support pour l'implémentation de la surveillance des hôtes de manière redondante
- Interface web, pour voir l'état actuel du réseau, notification et historique des problèmes, fichiers log, etc.

Nagios s'appuie sur un serveur Web (qui est un des pré requis) et des scripts CGIs qui permettent de représenter les analyses, les états des services ou des hôtes, les informations des notifications, les logs, la configuration... Son fonctionnement repose sur plusieurs fichiers de configuration qui regroupent les définitions des tests, des politiques, des contacts, des hôtes, etc... Tout ceci sera développé dans un prochain chapitre.

Les plug-ins permettent de faire des tests de services simples (comme les tests de services réseaux tel que HTTP, FTP...) comme le montre le schéma suivant :

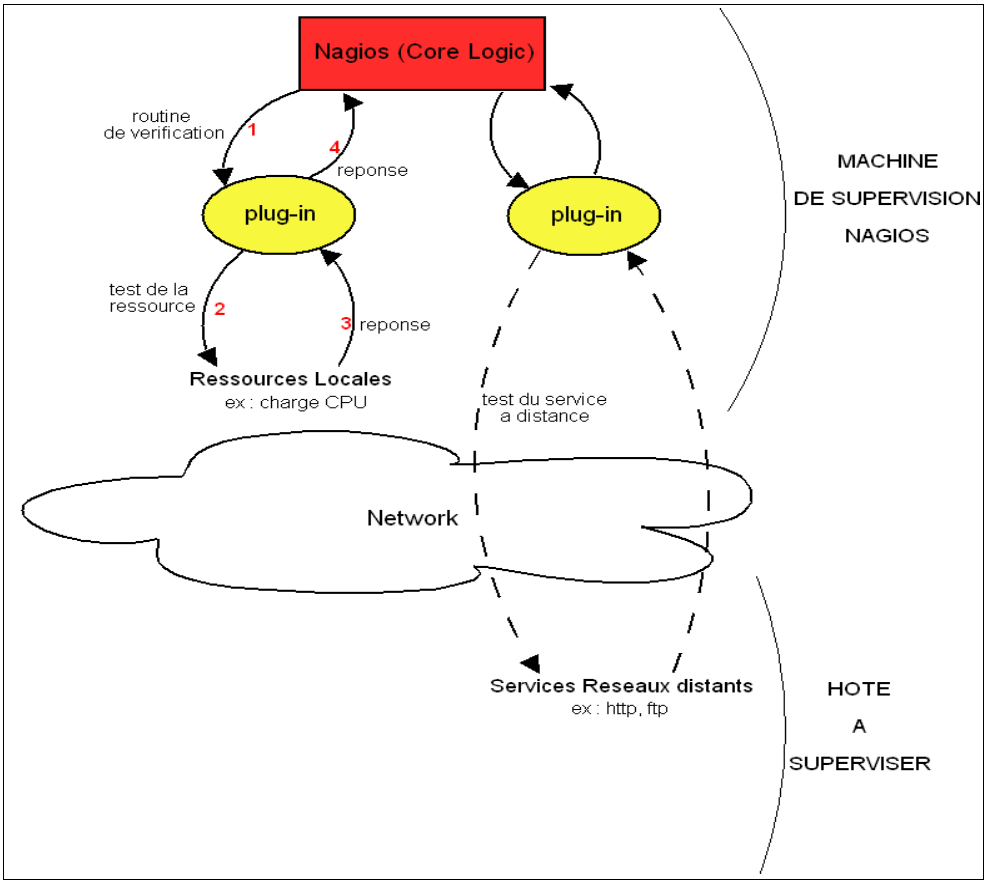

Fig. 4 : Fonctionnement de Nagios

Ainsi Nagios peut récupérer des informations sur l'état du serveur mail d'un hôte ou des ressources (espace disque, charge CPU...) de la machine sur laquelle Nagios a été installé.

Mais comment Nagios peut il récupérer des informations sur les ressources des hôtes distants ? Pour ce faire il va falloir utiliser un agent, qui devra être installé sur chaque hôte où l'on souhaite superviser les ressources. L'un des agents que l'on peut utiliser est NRPE (Nagios Remote Plug-in Executor) qui se présente sous la forme d'un plug-in et d'un démon (ou daemon). Le démon est un petit programme à installer sur les hôtes à superviser qui communiquera directement avec le plug-in de la station Nagios. Le plug-in *check_nrpe* tourne sur la machine Nagios et est utilisé pour envoyer les requêtes d'exécution de plug-in à *l'agent NRPE* de la machine distante. *L'agent NRPE* exécutera le plug-in approprié (c.a.d. correspondant à la référence envoyée via le plug-in *check_nrpe*) sur la machine distante et retournera les données de sortie et le code de retour au plugin check_nrpe de la machine Nagios. Le plug-in *check_nrpe* envoie la sortie du plug-in distant et le code de retour à Nagios comme si c'était le sien. Cela permet d'exécuter les plug-ins de manière transparente sur les machines distantes. Le mode de communication est dit actif car c'est la machine Nagios qui va initier l'exécution d'un test. Par opposition l'agent NCSA réalise le même travail de façon passive : hôtes renvoient eux-même le résultat de l'exécution d'un plug-in régulièrement.

Le schéma suivant, plus complet, montre du mode de fonctionnement de Nagios avec NRPE :

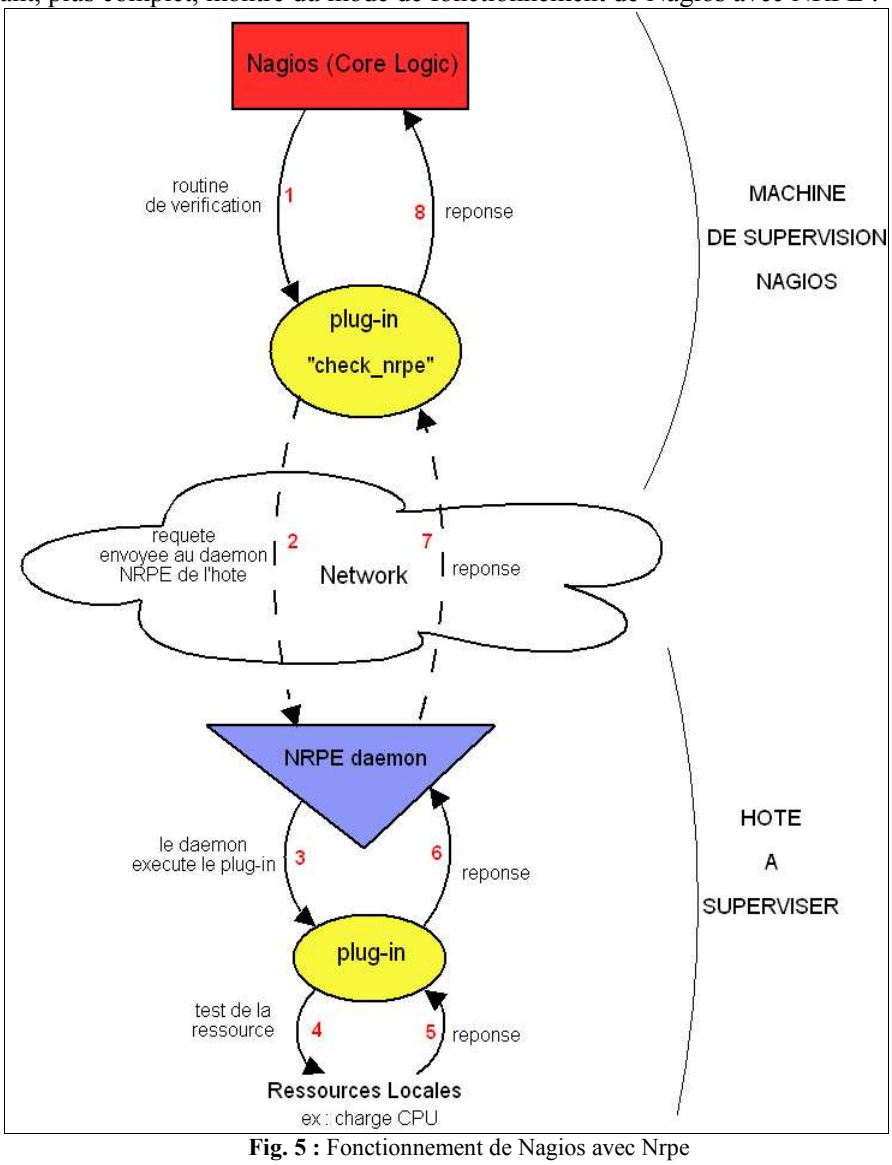

Fig. 5 : Fonctionnement de Nagios avec Nrpe

Concrètement le plug-in *check_nrpe* envoie au démon de l'hôte la référence du test à effectuer. Une fois ce message reçu le démon lance l'exécution du plug-in correspondant à la référence reçue.

A ce stade là si l'on combine uniquement Nagios avec NRPE et la suite de plug-ins officiels de Nagios on peut effectuer exactement les mêmes contrôles que la solution déjà installée BigBrother. De plus des fonctionnalités avancées seront disponibles comme la politique de notifications par exemple.

b/ Nagiosgraph et RRDTool

L'un des points importants du cahier des charges est de pouvoir générer des graphes à partir des données collectées via Nagios. Plusieurs outils réalisant cette tâche existe :

- ➔ Apan
- ➔ Cacti ou MRTG
- ➔ Nagiosgraph

Le choix c'est porté sur Nagiosgraph pour plusieurs raisons :

- Les données sont sauvegardées dans des base rrd (round robin database). Ces bases sont créées via RRDTool qui est une sorte de gestionnaire de base de données permettant entre autre leur représentation graphique. Les bases rrd ont la particularité de ne jamais dépasser leur taille de départ car elles sont créées pour contenir un nombre défini d'échantillons. Ce nombre est précisé lors la création de la base. Une fois le nombre maximum d'échantillons atteints, les nouveaux échantillons seront enregistrés par dessus les plus anciens. Ce système garantit que les bases rrd n'exploseront jamais et permet une plus grande souplesse dans leur utilisation et leur gestion. RRDTool est donc un pré-requis de Nagiosgraph.
- Nagiosgraph permet de grapher n'importe quel service ou ressource ce qui n'est pas le cas de Cacti ou MRTG qui ne génère des graphes que pour le pollling SNMP.
- Simplicité d'utilisation et d'installation. Si le nombre de sites traitant d'Apan est si important c'est davantage du à sa complexité d'installation qu'à son efficacité. Nagiosgraph se limite à l'utilisation de scripts Perl et CGI ce qui le rend très léger à installer et à exploiter.

Nagiosgraph se décompose en plusieurs fichiers :

- *nagiosgraph.conf* : qui est le fichier de configuration regroupant diverses informations sur la localisation des applications (RRDTool par exemple), les répertoires où seront sauvegardés les données...
- *insert.pl* : est le script Perl qui permet d'insérer les données dans les bases rrd qu'il aura créé.
- *show.cgi* : est le script CGI utilisé pour la génération des graphes à partir des bases rrd.
- *map* : qui est le fichier regroupant les expressions régulières des services et des ressources que l'on souhaite pouvoir grapher. C'est donc grâce à ce fichier que tout service ou ressource peuvent être graphé : il suffit de définir une expression régulière correspondant !

Le fonctionnement est relativement simple. Il faut pouvoir configurer Nagios pour qu'à chaque collecte d'informations il y ait une exécution automatique du script insert.pl. Ce script va récupérer la sortie texte des plug-ins et les comparer avec les expressions régulières du fichier map. Si une entrée correspond le script *insert.pl* va générer la base rrd. Si cette dernière existe déjà il se contentera de la mettre à jour.

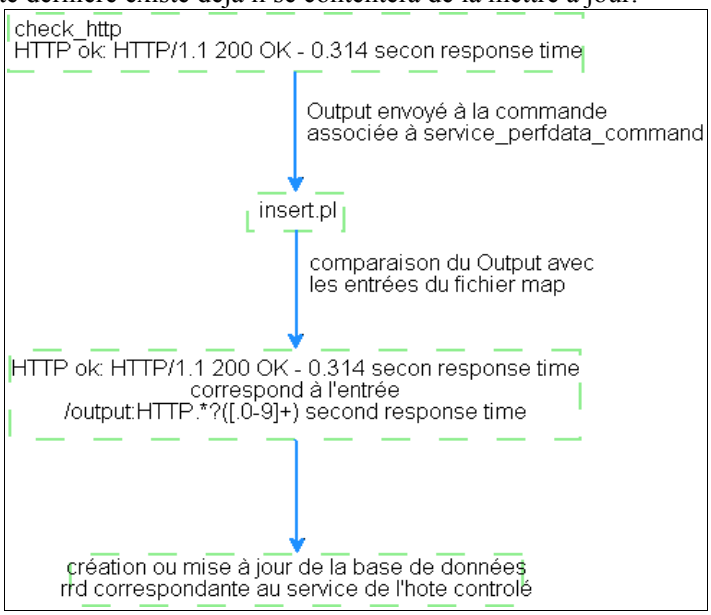

Fig. 6 : Fonctionnement de Nagiosgraph

Lorsqu'un utilisateur souhaite afficher le graphe correspondant il exécutera le script *show.cgi* en passant en paramètre le nom de l'hôte et le service. Le script interagit directement avec RRDTool et les bases rrd pour générer le graphe.

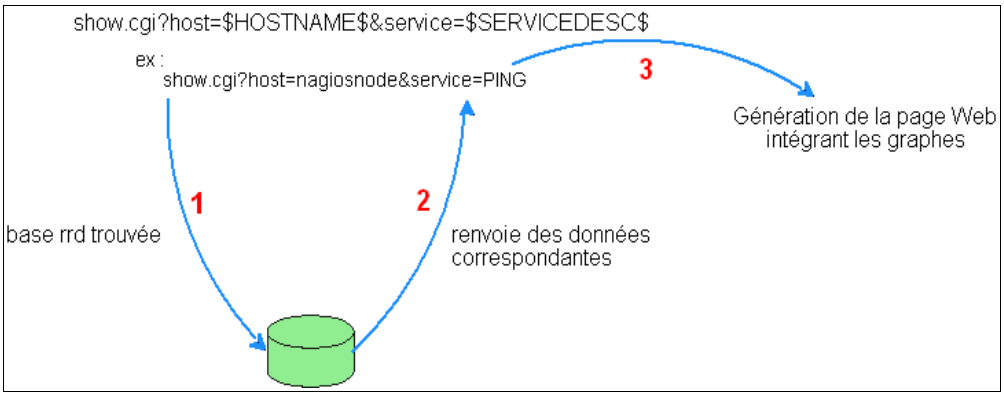

Fig. 7 : Fonctionnement de show.cgi

c/ Check_traffic

Check_traffic est un plug-in écrit en Perl qui permet de faire du polling SNMP et de sauvegarder ses résultats dans des bases de données rrd. Il ne fait pas partie de l'ensemble de plug-ins de bases. Pour fonctionner il est nécessaire d'avoir les outils de polling suivant : *snmpwalk* et *snmpget*. Ces outils font partie de la gamme *Net-SNMP* qui regroupe divers utilitaires utilisant le protocole SNMP.

Le détail de son fonctionnement sera développé plus loin dans le rapport.
D'autres mécanismes sont disponibles pour réaliser cette tâche (tel que *check_iftraffic*) mais le choix s'est fait sur le plug-in le plus complet et le plus simple à paramétrer.

3-4/ Bilan

Pour respecter entièrement le cahier des charges, nous avons donc décidé de travailler avec la liste des logiciels, scripts et plug-ins suivants :

- ➔ Nagios v1.2
- ➔ Nagios-plugins v1.3.1
- ➔ NRPE v2.0
- ➔ Nagiosgraph v0.3
- ➔ RRDTool v1.0.48
- ➔ Check_traffic v0.90b

Les versions pour ce projet ont été choisies principalement pour leur stabilité de fonctionnement. Le schéma suivant récapitule comment les différentes parties du projet interagissent entre elles :

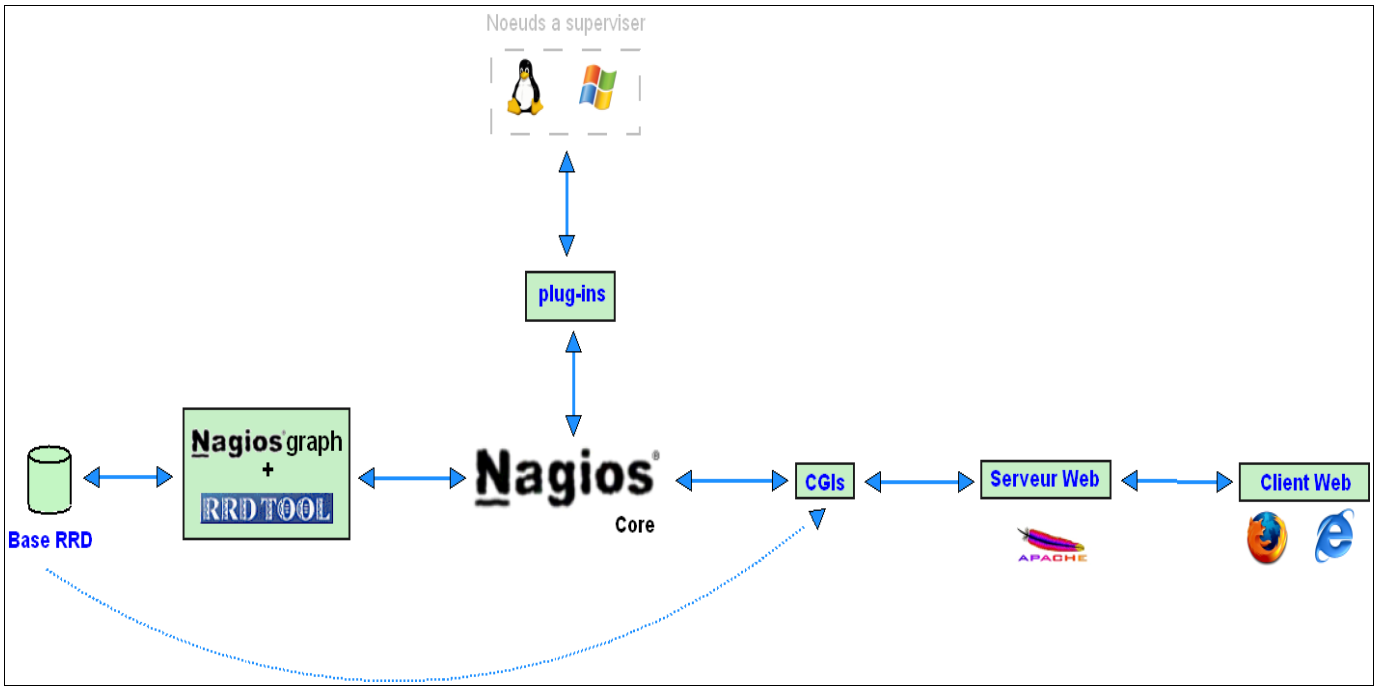

Fig. 8 : Bilan fonctionnel de Nagios et de ses composants

IV/ Politique de supervision

4-1 / Définition

Après avoir établie une liste, non figée, d'outils à utiliser, la deuxième étape consiste à établir la politique de supervision qui sera déployée sur le réseau. Une politique de supervision est l'ensemble des définitions des procédures de contrôles à effectuer et des hôtes à surveiller. La conception de la politique est le point crucial de la mise en place d'une solution de supervision. De cette phase découlera la configuration des outils, tout dépendra donc des choix faits lors de cette phase théorique.

Matériellement Weborama repose sur plusieurs serveurs différents qu'il sera nécessaire de surveiller :

- serveurs d'infrastructures (serveur mail, dns, ntp...)
- serveurs web
- serveur de base de données
- ainsi que sur un cluster de 9 machines.

Le but est donc de proposer une solution permettant de surveiller toutes ces machines (malgré leurs différences matérielle, de services ou de systèmes d'exploitation) qui sont indispensables à l'activité professionnelle de Weborama.

Cependant la politique, à élaborer pour Weborama, n'allait pas être si compliquée à évaluer puisqu'elle reprendra en partie la politique de BigBrother.

En m'inspirant de ce qui était déjà établi, j'ai pu construire une matrice de politique (cf. **Fig. 9 :** Matrice de politique) qui évoluera et sera complétée tout au long de ce projet...

Hotes	Adresses IP	Groupes	Services et systèmes à vérifier													
			Services Réseaux								Systèmes					
			Ping (Is Alive)	Trafic	Http	Dns	Imap	Pop	Smtp	Ntp	MySQL	Charge CPU	Espace disque	Surveillande de "crond"
aimfar.weborama.fr	217.117.154.2	Cluster VIP												
weboscope.weborama.fr	217.117.154.3													
ruedunet.weborama.fr	217.117.154.4													
ssl.weborama.fr	217.117.154.6													
node-10	217.117.154.10	Cluster Directors												
node-11	217.117.154.11													
node-12	217.117.154.12	Cluster Nodes												
node-13	217.117.154.13													
node-14	217.117.154.14													
node-15	217.117.154.15													
node-16	217.117.154.16													
node-17	217.117.154.17													
node-18	217.117.154.18													
bdd-base	10.10.10.100	Data Base Servers												
bdd-pro	10.10.10.101													
bdd-free	10.10.10.102													
bdd-stat	10.10.10.106													
bdd-calcul	10.10.10.103													
bdd-slave	10.10.10.107													
secure	217.117.154.31	Web Servers												
web	217.117.154.30													
ns2	10.10.10.50	Infrastructure Servers												
ns1	217.117.154.40													
storage	10.10.10.120													
ns-cache	10.10.10.40													
brv2	217.117.154.66													
devel	217.117.154.37													
ns1.fr.telecity.net	217.117.145.36	Dns Secondaries												
ns3.fr.telecity.net	217.117.155.28													
www.google.fr	www.google.fr	Connectivity												
www.yahoo.fr	www.yahoo.fr													
ns6.gandi.net	ns6.gandi.net													
switch-ext1	10.10.13.1													
switch-ext2	10.10.13.2													
switch-int1	10.10.13.3													
switch-int2	10.10.13.4													
switch-san	10.10.13.5													
gateway	217.117.154.1													

.... Non pollable ....

Fig. 9 : Matrice de politique

Après avoir conçu la matrice de la politique de contrôle du réseau de Weborama, il est ensuite de rigueur de préciser les seuils critiques, les procédures de notifications, les personnes à contacter... pour chacun des services et des ressources supervisées.

Les questions à se poser, pour compléter la matrice de politique, sont les suivantes :

- ➔ Quelle sera la fréquence de contrôle ? Est ce que la ressource est vraiment critique et doit-on la contrôler toutes les minutes ? Est il possible de la contrôler un peu moins régulièrement ?
- ➔ A partir de quel seuil le service (ou la ressource) sera-il considéré en état WARNING ou en état CRITICAL ? Sur quelle base sont-ils définis ?
- ➔ Quand une notification sera-t-elle envoyée ? A partir de combien de contrôles la notification devra être émise ?
- ➔ Les notifications sont elles nécessaires pour tous les hôtes ?
- ➔ A qui envoyer les notifications ? Quelles sont les personnes concernées par le problème et qui est en mesure de le résoudre ?
- ➔ Au bout de combien de temps une nouvelle notification doit elle être envoyée si l'état persiste ?
- ➔ Est il intéressant d'obtenir des graphes des résultats du contrôle ?
- ➔ Quel plug-ins parmi ceux disponibles dans la suite nagios-plugins-1.3.1 sont nécessaires pour réaliser tous ces contrôles ?

En répondant à ces questions, il sera possible de configurer complètement Nagios aux besoins de Weborama.

4-2/ Politique pour les services réseaux

a/ Questions communes

- *Quelle sera la fréquence de contrôle ?*
  Sauf précisions, le service réseau d'un hôte sera contrôlé toutes les minutes.
- *A partir de quel seuil le service sera-t-il considéré en état WARNING ou en état CRITICAL ?*
  Les tests des services réseaux reposent quasiment tous (voir plus bas pour de plus amples informations) sur un test de connexion ou l'exécution d'une simple requête, typique du service supervisé. Deux cas se présentent : soit le service répond et on considère un état OK ; soit le service ne répond pas et donc on considère la génération d'une alarme avec un état CRITICAL. Bien évidemment pour certains contrôles, plusieurs précisions seront nécessaires pour rendre plus fine et fiable la supervision.
- *Quand une notification sera-t-elle envoyée ?*
  Il faut considérer qu'un réseau n'est jamais totalement parfait et que de ce fait certains problèmes peuvent survenir : perte ou corruption de paquets, temps de réponse trop long ou tout simplement un état éphémère... Par conséquent notifier les administrateurs après seulement un contrôle ne serait pas pertinent. On imposera la règle suivante : si au bout de 3 contrôles successifs, un état NON-OK est persistant alors une notification sera envoyée. Ce choix est totalement arbitraire.
- *Les notifications sont elles nécessaires pour tous les hôtes ?*
  Excepté pour les hôtes n'appartenant pas à Weborama (à savoir : www.yahoo.fr, www.google.fr, ns1.fr.telecity.net et ns3.fr.telecity.net, ns6.gandi.net), sur lesquels les administrateurs ne pourront pas agir directement, toutes les notifications seront nécessaires.
- *A qui envoyer les notifications ?*
  Pour notifier intelligemment il a été nécessaire de définir des groupes de notifications comme suit :
  - ➔ Un groupe « Administrateurs » qui doit être absolument notifié de tout puisque ce groupe est composé des administrateurs réseaux et systèmes de l'entreprise qui sont susceptibles de pouvoir résoudre ces problèmes.
  - ➔ Un groupe « Techniciens / Développeurs » qui doit être notifié des problèmes concernant les services web, web sécurisé, mysql et les ressources systèmes de toutes les machines supervisées. De plus si une machine ne répond pas à un test Ping, ce groupe sera notifié. Ce groupe est composé de l'équipe de Recherche et Développement de la société.
  - ➔ Un groupe « Boss » qui doit être notifié des problèmes sur les machines critiques pour le traitement des données des clients. Donc tout ce qui concerne la gateway ainsi que le groupe Cluster Vip doit

être remonté à ce groupe de notification. Ce groupe est composé de toute l'équipe dirigeante de la société ayant un contact direct avec les clients. Ce groupe est important du point de vue de la crédibilité de la société : si lorsqu'un client appelle pour un problème technique et que l'un des membres de l'équipe technique est en mesure d'expliquer d'où vient le problème et de le conforter en lui indiquant le temps de rétablissement alors on améliore la réactivité des interventions, aspect primordial pour assurer la confiance du client.

- *Au bout de combien de temps une nouvelle notification doit elle être envoyée si l'état persiste ?*
  Si au bout d'une heure le problème recensé persiste (sans avoir été acquitté) alors une nouvelle notification sera émise aux personnes concernées afin de leur rappeler le problème.
- *Est il intéressant d'obtenir des graphes des résultats du contrôle ?*
  Il est intéressant d'obtenir des graphes pour avoir une évolution du temps de réponses avec les sites extérieurs, comme yahoo.fr ou google.fr. Mais le service le plus intéressant à grapher est le taux d'utilisation de la bande passante. Cela permettra de faire des statistiques plus fines du débit accordé et de pouvoir en plus les confondre avec les statistiques fournies par le provider pour éventuellement servir de preuves pour une contestation par exemple.
- *Quel plug-ins parmi ceux disponibles dans la suite nagios-plugins-1.3.1 sont nécessaires pour réaliser tous ces contrôles ?*
  Les explications suivantes donnent des informations supplémentaires sur la politique particulière à appliquer à chaques services.

b/ Ping (« is Alive »)

La politique à adopter pour ce service est simple : si un hôte ne répond pas au Ping, un état CRITICAL sera déclaré. De plus nous avons rajouté une regle suplémentaire pour ce test : si le temps de réponse est supérieur à 1000 ms alors l'état sera WARNING et si il est supérieur à 3000 ms alors l'état sera CRITICAL. Cela permettra de nous apercevoir des moments ou il y aura de la congestion sur le réseau.

c/ Http

Ce service est plus subtil que les autres services réseaux, en effet les configurations Http diffèrent suivant les serveurs, il est donc nécessaire de créer plusieurs services de contrôle adaptés pour chaque configuration : les serveurs Web classiques, les serveurs sécurisés, les serveurs qui répondent sur des ports particuliers ou à des urls spécifiques...
Dans cette optique nous avons pensé qu'il serait bon de créer les services suivants :
le service HTTP : requête classique, retourne un état CRITICAL pour les réponses HTTP dfférentes de 200.
le service HTTP_fcgi : pareil que précédemment sauf qu'une url particulière est demandée : /fcgi-bin/env.fcgi
le service HTTP443 fcgi : identique au service HTTP fcgi mais la requête est envoyée sur le port 443
le service HTTPs_fcgi : réalise une requête HTTPs sur le port 443 en demandant une url précise : /fcgi-bin/env.fcgi
le service HTTPs : réalise une requête HTTPs sur le port 443

d/ Dns

Comme il l'a été décrit dans la politique générale sur les services réseaux, une alarme avec l'état CRITICAL sera générée uniquement si la procédure de résolution de nom n'aboutit pas et reste sans réponse.

e/ Imap

Une tentative de connexion TCP est réalisée sur le port passé en paramètre, si aucune réponse n'est renvoyée une alarme CRITICAL est générée. C'est le même principe pour les service NTP, POP, SMTP.

f/ Trafic

Le contrôle du trafic est intéressant pour repérer si il y a des congestions dans le réseau au niveau du goulot d'étranglement qui est le switch de sortie : switch-ext1. Nous avons imposé donc les quotas suivant : si la bande passante est utilisée à plus de 40% alors un état WARNING sera déclaré. Mais si plus de 50% de la bande passante est utilisée alors une alarme CRITICAL sera générée.
Pour ce contrôle, il sera intéressant d'obtenir des graphes afin de les mettre directement en opposition avec ce que mesurent les FAIs à l'aide d'un MRTG (outil mesurant le trafic réseau) en terme de dispnibilité et de facturation..

g/ MySQL

A la manière d'un contrôle réseau, si aucune réponse n'est retournée apres l'envoi d'une requête de type *« Select 1; »* alors un état CRITICAL sera généré.

4-3/ Politique pour les ressources systèmes

a/ Questions communes

Il n'est pas facile de normaliser une politique pour toutes les ressources d'un système à cause de leurs différences fondamentales : sur leurs types, sur leur niveau critique, etc... C'est pour cela que pour chaque ressource une politique particulière sera développée. Cependant pour certaines questions les réponses seront les mêmes :

- *A qui envoyer les notifications ?*
  La politique de notification est exactement la même que pour les services réseaux.
- *Les notifications sont elles nécessaires pour tous les hôtes ?*
  Ce type de contrôle ne peut se faire que sur des machines sur lesquelles les administrateurs de Weborama ont la main à savoir tous les hôtes appartenant à la société (vu que ces contrôles nécessitent l'installation préalable de l'agent NRPE)
- *Au bout de combien de temps une nouvelle notification doit elle être envoyée si l'état persiste ?*
  Si au bout d'une heure le problème recensé persiste alors une nouvelle notification sera émise aux personnes concernées.
- *Les notifications sont elles nécessaires pour tous les hôtes ?*
  Il est nécessaire d'envoyer une notification pour tous les hôtes que l'on supervise si un problème est détecté. Ayant la main sur les machines qui nécessitent ce genre de contrôle, les administrateurs pourront agir pour le rétablissement à un état OK.
- *Est il intéressant d'obtenir des graphes des résultats du contrôle ?*
  Oui il serait intéressant de pouvoir voir l'évolution de la charge CPU des hôtes supervisés afin de mieux comprendre et gérer l'utilisation des ressources systèmes. Cependant c'est la seule ressource qui ait un intérêt certain, la description des contrôles qui suit expliquera pourquoi.

b/ Contrôle de la charge CPU

- *Quelle sera la fréquence de contrôle ?*
  Le contrôle de ce type de ressource se fera toutes les minutes.
- *A partir de quel seuil le service ou la ressource sera considéré en état WARNING ou en état CRITICAL ?*
  Pour connaître la charge du processeur il faut pouvoir se référer au fichier */proc/loadavg* qui détient l'information sous la forme suivante a,b,c . a,b,c représente le nombre moyen de processus en queue (ou en attente d'exécution) sur une période de 1 minute (pour a), de 5 minutes (pour b), de 15minutes (pour c).
  Pour déterminer le seuil de l'état CRITICAL, nous avons considéré que pour un hôte mono-processeur le seuil critique était de 8,6,4 (8 processus en moyenne en queue sur 1 minute, 6 processus sur 5minutes et 4 sur 15 minutes). Cette moyenne sera multipliée par le nombre de processeurs que possède l'hôte, pour connaître cette information il suffira de consulter les informations du fichier */proc/cpuinfo* dont voici un exemple :

```
[root@testbox root]# more /proc/cpuinfo | grep processor
processor       : 0
processor       : 1
```

  Cet hôte possède donc 2 processeurs, on considèrera que le seuil critique est de 16,12,8. Si lors du contrôle, l'une des 3 valeurs retournées excède ce qui est précisé dans le seuil alors une alarme avec l'état CRITICAL sera générée.

- *Quand une notification sera-t-elle envoyée ?*
  En supposant qu'un état ne peut être que passager (il serait aberrant de notifier les administrateurs si le processeur est surchargé que sur une minute) c'est au bout du troisième contrôle négatif qu'une notification sera envoyée. Donc si un état NON-OK persiste pendant 3 minutes une notification sera envoyée.

c/ Contrôle de l'espace disque disponible

- *Quelle sera la fréquence de contrôle ?*
  Le contrôle de ce type de ressource se fera toutes les minutes.
- *A partir de quel seuil le service (ou la ressource) sera considéré en état WARNING ou en état CRITICAL ?*
  De façon générale un disque dépassant 85% de sa capacité commence à avoir une baisse notable de ses performances (problèmes dans la recherche des inodes et allocation des blocs). On considérera alors que c'est le seuil de WARNING. Le seuil CRITICAL sera différent suivant la capacité totale du disque à superviser, si ce dernier a une capacité importante (72 Go) alors le seuil CRITICAL sera atteint lorsque les 95% du disque seront occupés ; pour les autres disques le seuil sera baissé à 90%. Le seuil CRITICAL représente le temps de réaction laissé aux administrateurs pour régulariser la situation (effacer les fichiers temporaires par exemple...).
- *Quand une notification sera-t-elle envoyée ?*
  Dès que le contrôle retourne un état WARNING ou CRITICAL, il sera nécessaire d'envoyer une notification aux groupes concernés.
- *Est il intéressant d'obtenir des graphes des résultats du contrôle ?*
  Vu que l'espace libre d'un disque varie relativement lentement avec le temps, il ne sera pas vraiment pertinant de créer des bases rrd pour visualiser des graphes.

d/ Contrôle du processus crond

Le processus crond est chargé de faire exécuter par le système toutes tâches (commandes et scripts) définies et planifiées à l`avance.

- *Quelle sera la fréquence de contrôle ?*

Le contrôle de ce type de ressource se fera toutes les minutes.

- *A partir de quel seuil le service ou la ressource sera considéré en état WARNING ou en état CRITICAL ?*
  Il doit toujours avoir une unique instance de crond, qui est le démon qui lance les tâches planifiées. On considère donc un état normal si le nombre de processus lancés par crond est supérieur à 1et inférieur à 4. Entre 4 et 5 processus lancés par crond, une alarme WARNING sera générée. Au delà de 5 c'est une alarme CRITICAL qui sera déclarée. Si le nombre de processus lancés par crond est trop important cela signifie que soit un des processus ne s'est pas fini normalement ou que tout simplement trop de tâches étaient prévues au mêmes instant (baissant entre les performances globales de la machines). Il est donc important de contrôler le nombre maximum de processus lancés via crond pour soit « killer » manuellement un processus ou mieux répartir les tâches planifiées.
- *Quand une notification sera-t-elle envoyée ?*
  Dès que le contrôle retourne un état WARNING ou CRITICAL, il sera nécessaire d'envoyer une notification aux groupes concernés.
- *Est il intéressant d'obtenir des graphes des résultat du contrôle ?*
  Non, la représentation du nombre de processus crond et des processus lancés par son biais n'a pas un intérêt important, il ne sera pas utile de créer les base rrd associées.

Service/Ressource à superviser	Seuils / Politique	Machines supervisées
Ping (Host Alive)	l'hôte doit répondre dans un délai - inférieur à 1sec sinon état WARNING - inférieur à 3sec sinon état CRITICAL	Toutes
Http	l'hôte testé doit retourner un code retour 200 signifiant le bon fonctionnement du serveur (plus d'informations dans la description du service)	Cluster-Vip, Cluster Nodes, Web Servers, ns2
Dns	l'hôte testé doit retourner la résolution de nom dans le cas contraire une alarme CRITICAL sera générée.	Dns Secondaries, ns1, ns2, ns-cache
Pop	l'hôte testé doit répondre à la requête émise sinon un état CRITICAL sera généré	ns2
Imap	l'hôte testé doit répondre à la requête émise sinon un état CRITICAL sera généré	ns2
Smtp	l'hôte testé doit répondre à la requête émise sinon un état CRITICAL sera généré	ns2, ns1
Ntp	l'hôte testé doit répondre à la requête émise sinon un état CRITICAL sera généré	ns-cache
Trafic	si le pourcentage de bande passante utilisée excède les 40% alors une alarme WARNING est déclaré, au-delà de 50% un état CRITICAL est généré	switch-ext1
MySQL	l'hôte testé doit pouvoir répondre à la requête de type "select 1" si aucune réponse n'est retournée un état CRITICAL est déclaré	Cluster nodes, Database Servers sauf bdd-slave
Contrôle de la charge CPU	l'hôte testé doit avoir une charge moyenne inférieure à : - 8*le nombre de processeurs de la machine sur 1 minute - 6*le nombre de processeurs de la machine sur 5 minutes - 4*le nombre de processeurs de la machine sur 15 minutes Si une des trois conditions n'est pas vérifiée un état CRITICAL est déclaré	Cluster Directors, Cluster Nodes, Database Servers, Web Servers, Infrastructure Servers
Contrôle de l'espace disque	l'hôte testé doit avoir utiliser moins de 85% sinon un état WARNING est déclaré, au-delà de 90 % (pour un disque de capacité inférieure à 18Go) ou 95% (pour les autres disques) une alarme CRITICAL est générée.	Cluster Director, Cluster Nodes, Database Servers, Web Servers
Contrôle des processus crond	Il faut absolument qu'il y'ait au moins une instance crond sinon une alarme CRITICAL est déclarée. De plus si il y a 5 ou 4 instances alors un état WARNING est déclaré. Au-delà de 5 instances une alarmes CRITICAL est générée.	Cluster Director, Cluster Nodes, Database Servers, Web Servers, Infrastructure Servers

Fig. 10 : Bilan de la politique

Cette matrice reprend les principaux points de la politique qui va être appliquée au déploiement de Nagios, on y retrouve bien toutes les machines et serveurs cités et décrits précédemment.

Bien sûr cette matrice et ces choix ne sont pas figés et pourront être revus lors des phases de tests de déploiement.

V/ Concepts avancés de Nagios

5-1/ Filtres de notification

Le simple fait qu'une notification d'hôte ou de service puisse être émise ne signifie pas que des contacts vont la recevoir. Il y a plusieurs filtres qu'une notification doit traverser avant d'être émise. Cela signifie que des contacts peuvent ne pas la recevoir si leurs filtres de notification ne le permettent pas. Les options de notifications sont définies lors de la création d'un service, un hôte et un contact.

Il existe 3 catégories de filtres qui sont :

- Le filtre du mode de programme : Le premier filtre à traverser est un test pour savoir si les notifications sont activées au niveau global pour Nagios ou non. Ceci est spécifié par la directive *enable-notifications* dans le fichier de configuration principal, mais ce peut être modifié en cours d'exécution via l'interface web. Si les notifications sont désactivées de manière globale, aucune notification ne sera envoyée. Si elles sont activées, les filtres qui suivent seront appliqués.

Les filtres d'hôte et de service : Il existe 5 filtres différents qui sont les suivants :

- Le premier filtre des notifications d'hôte et de service consiste à vérifier que l'hôte ou le service n'est pas dans une période d'arrêt planifié. Si c'est le cas, personne n'est notifié.
- Le deuxième filtre des notifications d'hôte et de service consiste à vérifier si l'hôte ou le service oscille (à condition que vous ayez activé la détection d'oscillation). Si le service ou l'hôte oscille, personne n'est notifié.
- Le troisième filtre des notifications d'hôte et de service à traverser est formé par les options de notification. Chaque définition de service contient des paramètres qui déterminent si les notifications doivent être envoyées pour les états WARNING, CRITICAL et RECOVERY. De la même manière, chaque définition d'hôte contient des paramètres qui déterminent si les notifications doivent être envoyées quand l'hôte s'arrête, devient inaccessible, ou se rétablit. Si la notification d'hôte ou de service est bloquée par ces paramètres, personne n'est notifié.
- Le quatrième filtre des notifications d'hôte et de service à traverser concerne la période. Chaque définition d'hôte et de service a un paramètre <*notification_period*> qui spécifie quelle période contient les heures de notification valides pour cet hôte ou service. Si le moment où la notification apparaît n'est pas dans une plage valide de la période spécifiée, personne n'est contacté.
- Le dernier jeu de filtres d'hôte et de service à traverser est conditionnée par deux éléments : (1) une notification a déjà été émise par le passé concernant un problème avec l'hôte ou le service, et (2) l'hôte ou le service est resté dans le même état non-OK depuis la dernière notification. Si ces deux conditions sont réunies, Nagios vérifie que le temps écoulé depuis l'émission de la dernière notification est supérieur ou égal à la valeur spécifiée par l'option <*intervalle_de_notification*> dans la définition de l'hôte ou du service. Si le temps écoulé depuis la dernière notification est insuffisant, personne n'est contacté.

Les filtres de contacts : A ce point la notification a traversé le filtre du mode de programme et tous les filtres d'hôte et de service, et Nagios commence à envoyer des notifications à tous ceux qui doivent en recevoir. Cela signifie-t-il que tous les contacts vont recevoir la notification ? Non ! Chaque contact possède son propre jeu de filtres que la notification doit passer avant qu'il ne la reçoive. Le jeu de filtres est composé des deux filtres suivants :

- Le premier filtre à passer pour chaque contact concerne les paramètres de notification. Chaque définition de contact comprend des paramètres qui déterminent si les notifications de service peuvent être émises pour les états alerte, critique, et rétablissement. Chaque définition de contact contient également des options qui déterminent si les notifications d'hôte peuvent être émises lorsqu'un hôte passe hors fonction, devient inaccessible, ou se rétablit. Si la notification d'hôte ou de service ne remplit pas ces conditions, les contacts ne recevront pas de notification.
- Le dernier filtre à passer pour chaque contact concerne la période. Chaque définition de contact comprend un paramètre <*notification_period*> qui spécifie la période durant laquelle on peut envoyer

des notifications à ce contact. Si l'heure à laquelle la notification est faite n'est pas comprise dans la plage de temps de la période spécifiée, le contact ne recevra pas la notification.

C'est uniquement après être passé au travers de ces 8 filtres que le contact pourra recevoir une notification ! (a moins que malgré tous ces efforts le mail se perde :-p).

5-2/ Les options de notifications

Plusieurs options sont très importantes dans l'établissement de la politique de notification. On retrouve ces options dans les définitions de services et d'hôtes :

- <*max_check_attempt*> : cette option permet de préciser le nombre de fois que le contrôle est relancé si un état différent de OK est retourné. Si le nombre de tentative CONSECUTIVE est égale à <*max_check_attempt*> alors une notification sera envoyée (cf. explication sur l'état courant).
- <*normal_check_interval*> : cette option indique le nombre de minutes qui s'écoulera entre deux contrôles lorsque le contrôle précédent a retourné l'état OK. Exemple : si le contrôle du disque d'un hôte distant renvoie l'état OK alors le prochain contrôle de ce disque pour cet hôte sera prévu dans <normal_check_interval> minutes. Cette option n'est pas présente pour les hôtes, elle est imposée à 1 minute.
- <*retry_check_interval*> : est identique à <normal_check_interval> sauf que le contrôle précédent a retourné un état différent de OK. Cette option n'est pas présente pour les hôtes, elle est imposée à 1 minute.
- <*notification_interval*> : représente le nombre de minutes à patienter avant de notifier une nouvelle fois les contacts.
- <*notification_options*> : permet de spécifier pour quels états une notification sera envoyée. Cette option est aussi présente dans la définition de contacts pour permettre de créer un filtre au niveau du contact et lui assurer de ne recevoir uniquement certaines notifications (comme ne recevoir que les notifications pour l'état CRITICAL).

5-3/ L'état courant

L'état courant des services et des hôtes est déterminé par deux composants : l'état du service et le type d'état.
L'état du service ou de l'hôte peut prendre les valeurs suivantes : OK, WARNING, UP, DOWN, CRITICAL. Ils sont déterminés à la suite d'un test (réalisé par un plug-in) suivant des paramètres (un pourcentage d'utilisation, un code retour etc...) . L'état du service correspond au retour du plug-in à un instant t.
Il existe deux types d'état dans Nagios : les états "soft" et les états "hard". Les types d'état sont une partie cruciale de la logique de supervision de Nagios en particulier pour les politiques de notifications.

- Le type d'état soft survient quand un contrôle de service ou d'hôte retourne un état non-OK et que le contrôle n'a pas été effectué autant de fois que le précise la variable <*max_check_attempt*>. Suivant la configuration de Nagios, un gestionnaire d'événement peut être exécuté. De plus une sauvegarde dans le fichier de log peut être programmée.
- le type d'état hard survient quand un contrôle de service ou d'hôte retourne un état non-OK et que le contrôle a été réalisé <*max_check_attempt*> de fois. Suivant la configuration de Nagios, un gestionnaire d'événement peut être exécuté et une sauvegarde dans le fichier de log peut être programmée. De plus les notifications seront envoyées aux contacts concernés uniquement dans ce cas la (si les opions de notifications le permettent).

Voici un schéma imageant le mode de fonctionnement :

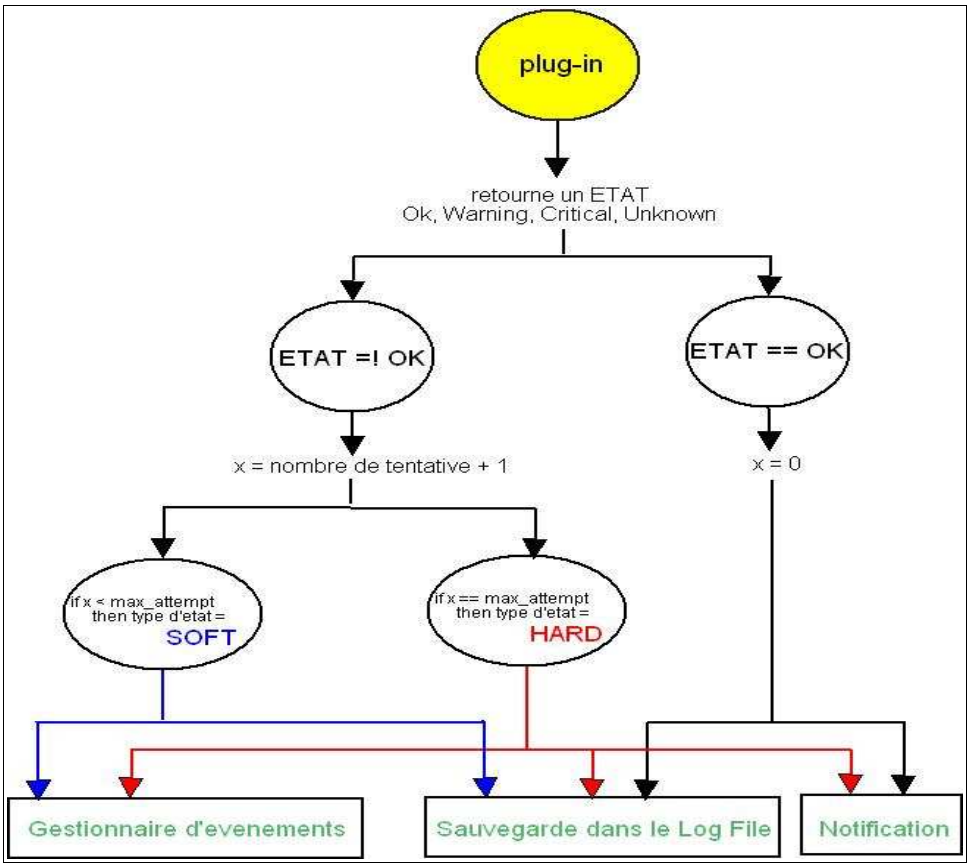

Fig. 11 : Mécanisme des etats Nagios

C'est donc en comprenant le mécanisme de changement des types d'états et en jouant avec les options de notifications que l'on peut mettre en place une politique de notification pertinente.
Par exemple pour les ressources critiques il peut être intéressant d'avoir un <*retry_check_interval*> relativement bas pour rapidement contrôler si l'état critique persiste. De la même manière les ressources stables n'ont pas toujours besoin d'être contrôlées fréquemment auquel cas <*normal_check_interval*> sera plus important. Il est possible aussi que les administrateurs ne souhaitent pas recevoir de notification pour certains services car ils n'ont pas la main dessus par exemple. La configuration de Nagios sera donc optimisée suivant les besoins.

5-4/ Les dépendances

Les dépendances permettent de créer des liens entre les services afin d'éviter entre autre de contrôler certains services ou d'envoyer des notifications qui seraient superflues. Par exemple si un le service PING ne répond pas il sera inutile de contrôler les autres services et ressources de cette machine. En effet si le PING ne répond théoriquement les autres services ne peuvent répondre.
Pour définir une dépendance de service il faut suivre le schéma suivant :

```
define servicedependency {
        dependent_host_name
        dependent_service_description
        host_name
        service_description
        execution_failure_criteria
        notification_failure_criteria
}
```

Le service de l'hôte décrit dans les paramètres rouges, sera dépendant du service de la machine définie en bleu.
Le champ execution_failure_criteria permet de spécifier pour quel état de « bleu » le contrôle de « rouge » sera abandonné.
Par exemple : si dans la définition "*execution_failure_criteria : w,c*" cela signifie que « rouge » ne sera plus contrôlé si le service « bleu » est dans un état WARNING ou CRITICAL.<*notification_failure_criteria*> fonctionne de la même manière que <*execution_failure_criteria*>, pour certains états décrits du service « bleu » aucune notification liée au service « rouge » ne sera envoyée.
Il est important de noter aussi que les dépendances ne sont pas héritées : si A dépend de B et B dépend de C si rien n'est précisé alors A ne dépend pas deC. Il est donc obligatoire de définir un lien pour toutes les dépendances.
De plus toutes définitions de dépendances doivent aboutir à un ou plusieurs arbres.
Les dépendances d'hôte sont légèrement différentes de part leur définition puisqu'elles ne permettent uniquement de supprimer les notifications d'hôtes. Les dépendances d'hôtes suivent les mêmes règles que les dépendances de services (arbre et héritage). De plus un hôte ne sera pas consideré comme étant DOWN si il dépend d'un hôte en panne, il sera consideré comme étant UNREACHABLE (injoignable).

VI/ Manipulation et déploiement

6-1/ Pratique : aprentissage et tests

Cette phase de pratique et de tests a pour but de me familiariser avec l'outil et ses composants avant le déploiement de la solution. L'objectif étant qu'à la fin de cette phase je sois en mesure de réaliser tout ce qui a été abordé lors de l'élaboration du cahier des charges et d'écrire une méthode d'installation du logiciel et de ses éléments. Pour ce faire, une machine de test a été reconstituée à partir d'anciennes stations sur laquelle une WhiteBox a été installée de façon à se mettre dans les mêmes conditions que pour le futur déploiement. C'est cette machine qui me servira de station de supervision Nagios.

Pour organiser les tests la Todo List suivante a été élaborée :

a/ Tester les plug-ins

La première étape consiste à manipuler les divers scripts et programmes qui sont utilisés par Nagios pour la supervision : les plug-ins. Voici un descriptif mettant en avant comment ils fonctionnent et comment ils sont utilisés :

check_ping : Ce plug-in envoie une requête Ping sur un hôte dont l'adresse est passée en paramètre. Ce plug-in sera donc utile pour tester la connexion avec un hôte ou un équipement distant et vérifier qu'il est bien toujours "en vie". Ce plug-in est basé sur la commande réseau "ping @host".
Syntaxe pour l'utiliser : `check_ping -H @host -w RTT,% -c RTT,% -p nbr_de_paquets`
Exemple d'utilisation:

```
[root@textbox local]#./check_ping -H 192.168.0.62 -w 1000.0,80% -c 3000.0,100% -p 5
PING OK - Packet loss = 0%, RTA = 0.20 ms
```

RTT représente le temps d'attente maximum pour recevoir une réponse avant de générer une alarme avec l'état Warning (-w) ou Critical (-c).
% représente le pourcentage maximum de paquets à perdre pour passer à l'état WARNING ou CRITICAL.
nbr_de_paquets représente le nombre de paquets à envoyer pour le contrôle.

---

check_http : Ce plug-in est util pour tester et vérifier qu'une connexion HTTP (ou HTTPS) est possible avec un hôte dont l'adresse est spécifiée en paramètre.
Syntaxe : `check_http -H @host [-p #port] [-S] [-u path]`
Exemple d'utilisation:

```
[root@textbox local]#./check_http -H 192.168.0.62 -p 80
HTTP ok: HTTP/1.1 200 OK - 0.006 second response time
```

@host représente l'adresse de l'hôte que l'on souhaite interroger.
#port est le numéro de port sur lequel le serveur HTTP de l'hôte écoute, par défaut cette valeur est à 80 (qui est le port dédié et réservé à HTTP).
L'option -S permet de spécifier que la requête va se faire via SSL de façon à envoyer une requête de type HTTPs.
path permet de préciser le chemin vers une page à interroger. Si l'option -u n'est pas présente dans la ligne de commande alors la page demandée sera @host/ !

---

check_dns : Ce plug-in permet de tester si le service DNS d'un hôte est bien actif. Pour cela ce plug-in va utiliser la commande "dig" (équivalent à la commande "nslookup") pour tenter une résolution de nom ou d'adresse à partir des arguments passés en paramètre.
Syntaxe : `check_dns -H @aresoudre -s @serveurDNS`

Exemple d'utilisation :

```
[root@testbox root]# ./check_dns -H rue-du.net -s ns2.weborama.fr
DNS ok - 1 seconds response time, Address(es) is/are 217.117.154.4
```

@aresoudre est le nom ou l'adresse IP que le serveur devra résoudre.
@serveurDNS est l'adresse IP ou le nom du serveur dont on contrôle le service.

---

check_pop : Ce plug-in permet de contrôler le bon fonctionnement du service POP d'un hôte. Pour cela une tentative de connexion TCP, sur le port passé en paramètre, est nécessaire.
Syntaxe : `check_pop -H @host [-p #port]`
Exemple d'utilisation :

```
[root@testbox root]# ./check_pop -H 217.117.154.50 -p 110
POP OK - 0.111 second response time on port 110 [+OK <5131.1115797046@ns2.weborama.fr>]
```

@host permet de spécifier l'adresse IP de l'hote à contrôler.
#port permet de définir un numéro de port si le service POP écoute sur un port différent de 110 (-p est une option).
Ce plug-in possède un fonctionnement et une syntaxe similaires aux plug-ins **check_imap** et **check_smtp** qui permettent de contrôler respectivement le service IMAP et SMTP d'un hôte passé en paramètre.

---

check_traffic : Ce plug-in, disponible en script Perl, permet de mesurer le trafic sur les interfaces d'un hôte ou d'un équipement réseaux par interrogation de la MIB (Management Information Base), sorte de base de données sur les informations réseaux de l'équipement. L'intérêt de ce plug-in est de pouvoir indiquer quel est le pourcentage de bande passante utilisée, cependant cette information n'est pas directement disponible dans la MIB en revanche il y a un compteur du nombre d'octets sortant et entrant de la carte réseau de l'équipement. Le plugin crée donc un fichier (spécifier dans le code du plug-in) dans lequel il écrit le nombre d'octets envoyés et reçus à l'instant "t" et lors du prochain contrôle le plug-in sera en mesure de calculer le pourcentage de bande passante utilisé entre les deux contrôles.
Syntaxe : `check_traffic -H @host  -i #int -b débit -w % -c %`
Exemple d'utilisation :

```
[root@testbox root]# ./check_traffic -H  192.168.0.1 -i 2 -b 1250000 -w 90 -c 98
Total RX Bytes: 235.61 MB, Total TX Bytes: 1668.59 MB
Average Traffic: 2.90 kB/s (0.2%) in, 34.82 kB/s (2.9%) out
```

@host permet de spécifier l'adresse IP de l'équipement que l'on souhaite contrôler.
#int est le numéro de l'interface surveillée, on peut obtenir ces numéros particuliers grâce à la commande "snmpwalk -v1 -c public @host" qui va parcourir et afficher toute la MIB de l'équipement, on y retrouve entre autre les numéros des interfaces supervisables.
débit représente le débit de l'interface surveillée en octets par secondes.
% représente le pourcentage maximum de bande passante pour passer à l'état WARNING ou CRITICAL.

---

check_mysql : Ce plug-in, disponible en script Perl, permet de contrôler la réactivité du service MySQL d'un hôte donné. Pour ce faire le plug-in envoie une requête "select 1" sous le compte d'un utilisateur spécifié en paramètre dans la commande. A noter que ce plug-in ne fait pas partie de la suite de plug-in officiel Nagios ou NetSaint.
Syntaxte : `check_mysql -s @host -u name -p password`

Exemple d'utilisation :

```
  [root@testbox root]# /usr/local/nagios/libexec/check_mysql -s 192.168.0.62 -u test -p
test
  OK - test@192.168.0.62 111 ms
```

@host permet de spécifier l'adresse IP de l'hôte à contrôler.
name permet de définir le nom de l'utilisateur pouvant faire des requêtes et password son mot de passe associé.

---

check_load : Ce plug-in permet de vérifier et de contrôler la charge CPU de l'hôte sur lequel le plug-in est lancé. Concrètement ce plug-in collecte les informations systèmes situées dans le fichier : */proc/loadavg*. Pour que ce plug-in soit opérationnel sur un hôte distant il faudra utiliser le plug-in via *check_nrpe.*
Syntaxe pour l'utiliser localement : `check_load -w  a,b,c -c e,f,g`
Exemple d'utilisation en local:

```
  [root@textbox local]#check_load -w 4,8,16 -c 4,8,16
  OK - load average: 0.01, 0.00, 0.00
```

a,b,c représente le nombre moyen de processus en queue (ou en attente d'exécution) sur une période de 1 minute (pour a), de 5 minutes (pour b), de 15minutes (pour c) qui une fois dépasser générera une alarme avec l'état Warning (-w).
Il en est de même pour *e,f,g* et si ces valeurs sont dépassées le plug-in retournera l'état Critical (-c).

---

check_disk : Ce plug-in ne contrôle que l'espace restant d'un disque de l'hôte sur lequel il est executé. Il utilise la commande système "df" afin de récupérer les informations sur l'espace disponible de chaque partition. De la même manière que pour le plug-in *check_load*, pour être opérationnel sur un hôte distant il faudra l'utiliser via le plug-in *check_nrpe.*
Syntaxe pour l'utiliser localement : `check_disk -w % -c % [-p path]`
Exemple d'utilisation en local:

```
  [root@textbox local]#check_disk -w 10% -c 5% -p/dev/sda1
  DISK OK [2742748 kB (70%) free on /dev/sda1]
```

% représente le pourcentage d'espace libre de la partition avant de passer à l'état WARNING (-w) ou CRITICAL (-c).
path (en option) permet de spécifier le chemin vers une partition particulière. Cette option peut être pratique dans le cas où l'on souhaite avoir une supervision précise et dédiée à une seule partition et non global (toutes les partitions de l'hôte).
A noter que si aucun chemin de partition n'est précisé tous les disques (que l'on peut lister grâce à la commande "df") seront vérifiés et si l'un d'entre eux n'est pas dans les normes définies alors une alarme sera générée.
Exemple d'utilisation en local:

```
  [root@textbox local]#check_disk -w 10% -c 5%
  DISK OK [9287088 kB (71%) free on /dev/hda3] [88799 kB (92%) free on /dev/hda1] [192984
kB (100%) free on /dev/shm] [26903424 kB (35%) free on /dev/hdc1]
```

---

check_proc : Ce plug-in fonctionne un peu à la manière de la commande suivante "ps aux | grep nom_du_process" qui permet d'obtenir la liste des processus nom_du_process lancés actuellement sur l'hôte sur lequel on exécute la commande. Pour que ce contrôle soit opérationnel sur un hôte distant il faudra l'utiliser via le plug-in *check_nrpe.*
Syntaxe pour l'utiliser localement le plug-in : `check_proc -w a:b -c c:d [-C nom_process]`

Exemple d'utilisation en local:

```
[root@textbox local]#check_proc -w 1:2 -c 1:5 -C nagios
OK - 1 processes running with command name nagios
```

a et *b* représentent les seuils pour lesquels l'état WARNING peut être déclaré si le résultat excède les seuils. *a* représente la borne minimale et *b* représente la borne maximale. De la même façon que *c* et *d* sont les bornes minimales et maximales pour lesquelles l'état CRITICAL peut être déclaré. L'option -C est utile pour spécifier *nom_process* qui est le nom du processus que l'on souhaite surveiller.

Cette étape est intéressante car elle m'a permis d'évaluer si les plug-ins étaient tous réellement adaptés au cahier des charges. Malheureusement certains d'entre eux ne nous permettent pas une intégration facile dans Nagios comme : *check_load*, *check_disk* ou le *check_mysql* par exemple. En effet les configurations étant différentes, il faudrait définir un service Nagios par configuration ce qui rendrait plus lourds les fichiers de Nagios ! Nous verrons dans la partie déploiement comment pallier à ce problème en réécrivant ces plug-ins.

b/ Installation de Nagios

Durant cette étape j'ai installé Nagios sur la machine de test allouée pour. L'objectif étant de voir les différentes installations possibles afin d'optimiser un maximum le ratio entre ce qui est installé et ce qui sera utilisé et de connaître les options importantes à utiliser lors de la compilation du programme.

c/ Mise en place d'une configuration de test

L'objectif de cette étape consiste à mettre en place une pseudo configuration afin de travailler avec Nagios (intégration des plug-ins, jouer avec les options de Nagios) ainsi que de le manipuler via son interface Web :

ajout/retrait de commentaires
arrêt planifié d'un contrôle
reprogrammation d'un contrôle
mise en veille de Nagios
génération et consultation des rapports
cartographie du réseau
...

d/ Tests des politiques de notification

Le but de ce test est de mettre en place des filtres de notifications et une hiérarchie de contacts, afin de bien maîtriser les remontées d'alertes par le biais d'envois de mails. J'ai donc fait des tests en créant plusieurs contacts Nagios avec des adresses différentes puis j'ai joué et testé les différentes options de notifications afin de satisfaire plusieurs scénarios comme par exemple : l'un des contacts étant prévenu de toutes les alertes et un autre uniquement des plus critiques...

e/ Tests des possibilités d'utilisation du gestionnaire d'événements

Nagios permet l'utilisation d'un gestionnaire d'évènements. Dans mon cas il m'est apparu intéressant de pouvoir relancer automatiquement un service si ce dernier est détecté comme DOWN. Pour cela j'ai rédigé le script shell suivant, capable de relancer (par ssh) un serveur Web à partir de la machine Nagios. J'ai rajouté dans la configuration Nagios que pour le service Http, le script doit être exécuté à chaque contrôle.
A noter qu'il a fallu au préalable configurer le client *ssh* pour ne pas préciser le mot de passe dans la commande.

Voici le script en question :

```
#!/bin/sh

# Note: Ce script redémarrera le serveur web si on essaie de le joindre
#       3 fois (dans un état "soft")ou si le serveur web finit
#       par tomber dans un état d'erreur "hard"
#
#Paramètres :
#$1 == SERVICESTATE
#$2 == STATE
#$3 == SERVICEATTEMPT



# On regarde par rapport au SERVICESTATE
case "$1" in
OK)
       # Si OK on ne fait rien
       ;;
WARNING)
       # Si WARNING on ne fait rien
       ;;
UNKNOWN)
       # Si UNKNOWN on ne fait rien
       ;;
CRITICAL)
       # Il doit y avoir un probleme

       # On va tester l'état SOFT ou HARD
       case "$2" in

       SOFT)
              # Si SOFT ca va dependre du nombre de tentatives
              case "$3" in
              3)
                     # Si c'est la troisième tentative :
                     # Connexion en ssh sur la machine distante pour redémarrer le service
                     ssh nagios@$4 sudo /usr/local/httpd/bin/httpd start
                     ;;
                     esac
              ;;


       HARD)
              #  Si  HARD  on  relance  directement  le  service  quelque  soit  le  nombre  de
tentatives
              # Connexion en ssh sur la machine distante pour redémarrer le service
              ssh nagios@$4 sudo /usr/local/httpd/bin/httpd start
              ;;
       esac
       ;;
esac
exit 0
```

Ce script concluant est facilement adaptable pour relancer n'importe quels serveurs. Cependant après concertation avec mon tuteur de stage il est apparu que de redémarrer directement un serveur n'est pas la meilleure chose à faire. En effet si un serveur tombe sans raison apparente, le fait de relancer directement le serveur, empêche de réellement identifier la source du problème. Le script que j'ai donc développé n'est pas adapté au besoin de l'entreprise. Il est, cependant, important de pouvoir maîtriser les gestionnaires d'évènements de Nagios pour d'autres tâches préventives (comme par exemple kill des process...) .
A noter que les scripts des gestionnaires d'évènements s'exécutent avec les permissions associées à l'utilisateur Nagios ! Il faudra donc prendre garde à donner les droits nécessaires sur les fichiers et les commandes pour qu'ils puissent exploiter pleinement le gestionnaire d'évènements.

f/ Tests des politiques de dépendances de services et d'hôtes

Pour ces tests, mon objectifs était de configurer l'arrêt du contrôle d'un service si un autre service tombait ; ainsi au lieu d'envoyer une notification pour chaque service DOWN, une seule sera envoyée pour tous. L'idée d'exploitation serait la suivante : si une machine ne répond pas au PING alors il est inutile de tester les autres services. Les dépendances sont donc intéressantes, seulement pour l'exploitation voulue il n'est pas forcement nécessaire de passer par la configuration de dépendances. En effet grâce au paramètre <*check_command*> dans la définition d'hôte on peut palier à ce problème : le <*check_command*> définit le nom de la commande à utiliser pour déterminer si l'hôte est hors service ou non. Typiquement, cette commande lance un "ping" vers l'hôte pour vérifier si il est "vivant". Lorsque un service renvoie un état non-OK, la commande associée au <*check_command*> est exécuté et si cette dernière renvoie aussi un état non-OK alors aucun autre contrôle de service ne sera effectué. Ce qui permet de créer une sorte de dépendance simplifiée entre le Ping et les autres services d'un même hôte.

g/ Tests des contrôles de ressources à distance

C'est durant cette phase que j'ai commencé à me familiariser avec NRPE, sa configuration et ses composants. Pour ce faire j'ai donc installé sur la machine de développement le démon NRPE ainsi que des plug-ins pour contrôler l'espace disque, la charge cpu... J'ai , ensuite, rajouté cette machine à la configuration de Nagios déjà mis en place et fait des tests jusqu'à obtenir une intégration parfaite des résultats retournés par le démon NRPE, dans Nagios.

h/ Installation des outils permettant de générer des graphes

Durant cette étape j'ai installé et configuré RRDTool ainsi que les scripts utiles à Nagiosgraph. J'ai en plus défini les expressions régulières pour tous les services et ressources correspondant à la matrice de politique, de telle sorte à pouvoir obtenir les graphes résultant de n'importe quel contrôle. Lors du déploiement, il n'y aura plus qu'à choisir les graphes souhaités (fichier pour le déploiement en annexe : **Annexe 9 :** map, page 73). J'ai ensuite configuré le <*service-perfdata*> de Nagios pour exécuter les scripts de Nagiosgraph de façon à ce que pour chaque contrôle de services réseaux ou de ressource systèmes les scripts soient exécutés et les informations collectées.

i/ Installation des outils permettant le polling SNMP

Dans cette partie je me suis familiarisé avec les outils Net-SNMP, en particulier avec les commandes *snmpget* et *snmpwalk* qui permettent de faire le polling d'un équipement en parcourant sa MIB. J'ai par la suite installé le script *check_traffic* qui grâce au polling d'un équipement permet de récupérer la quantité de bande passante utilisée. A noter que normalement dans la configuration du script *check_traffic*, il est possible de le lier directement à une base RRD de façon à stocker directement les résultats. Cependant cette option n'était pas intéressante, vu que lors de l'étape précédente j'ai automatiquement lié Nagios avec Nagiosgraph qui sauvegarde les résultats dans une base RRD. Donc pour éviter de faire deux fois le même travail je n'ai pas activé cette option. Il suffit juste d'écrire l'expression régulière voulue pour Nagiosgraph.

j/ Optimisation de la configuration

Cette phase n'est pas nécessaire au bon fonctionnement de Nagios, toutefois elle reste intéressante pour assurer une certaine lisibilité et compréhension des fichiers de configurations, et améliorer les performances générales et le confort d'utilisation.Les améliorations apportées sont les suivantes :

- réduction de la taille des fichiers de configuration en utilisant des templates (sorte de parties communes à tout les services ou hôtes) qui allègent considérablement les fichiers de configurations.
- modifications de certaines options du fichier de configuration principal pour optimiser les temps entre deux contrôles, les remontées d'alertes etc...
- modifications de l'interface Web en particulier avec l'ajout de nouvelles catégories et d'une pages Web regroupant les politiques établies.

Après cette phase de pratique j'étais en mesure d'écrire un howto complet (disponible en annexe : **Annexe 1 :** nagios-install.howto, page 56) récapitulant les points importants de l'installation et de la configuration de Nagios et de ses composants. En complétant mon étude théorique de début de stage avec cette phase de manipulation j'ai pu obtenir une vision globale des possibilités d'exploitation du produit Nagios.
Une fois les résultats validés par mon tuteur de stage, nous avons convenu de planifier le déploiement de la solution sur le réseau entier de Weborama.

6-2/ Pratique : le déploiement

Le déploiement de la solution a été réalisé par Francois Chassaing (mon tuteur de stage) et moi même en se basant sur le howto rédigé à la fin de la phases de tests. Au cours du déploiement il s'est avéré que certaines configurations prévues ne pourraient se faire ou que certaines prévisions ne seraient pas vraiment adaptées aux conditions réelles. Malgré la phase de test il a donc fallu modifier plusieurs paramètres :

La première adaptation nécessaire a été la réécriture des plug-ins suivants : *check-load*, *check-mysql* et *check-disk* (disponibles en annexe : **Annexe 5,6,7** a parti de la page 68). Le problème des versions fournies dans le pack de plug-ins Nagios, se situe au niveau de leur configuration. En effet pour chaque plug-ins plusieurs paramètres sont nécessaires et ces derniers différents suivant les machines. La solution de facilité aurait été de définir un service particulier pour chaque configuration différente du plug-in. Ceci entraînerait donc des fichiers de configuration encore plus lourds. Par exemple il faudrait un service DISK1 avec les seuils Warning à 80% et CRITICAL à 90 et un autre service DISK2 avec les seuils Warning à 80% et CRITICAL à 95%. Il peut donc y avoir très vite une charge supplémentaire de lignes dans les fichiers de configuration ce qui rendrait moins lisible le fichier. De plus le fait que les services ne portent pas le même nom alors qu'on teste la même ressource peut prêter à confusion pour l'utilisateur.

Nous n'avons donc pas opté pour cette solution. Notre idée était de rendre un peu plus malin les plug-ins en les rendant autonomes : pouvoir déterminer les paramètres ou les seuils à partir de la configuration de la machine en ne spécifiant acun argument. Ainsi on aura juste à créer un service DISK dont les seuils WARNING et CRITICAL seront établis directement par le script, ce qui permettra d'avoir le même service DISK pour toutes les machines supervisées malgré leurs configurations matérielles différentes.

- Ainsi le nouveau plug-in *check_disk* (disponible en annexe : **Annexe 6 :** check_diskauto.pl, page 69) déterminera les seuils d'utilisation suivant la capacité des disques que la station possède.
- Le nouveau plug-in *check_load* (disponible en annexe : **Annexe 5 :** check_cpuauto.pl, page 68) sera en mesure de définir les seuils critiques en fonction du nombre de processeurs que la machine possède.
- Le nouveau plug-in *check_mysql* (disponible en annexe : **Annexe 7 :** check_mysqltest.pl, page 70) pourra récupérer dynamiquement les différentes sockets (avec login et mot de passe associés) des instances MySQL qui tournent sur une machine et de tester chacune de ces instances.

On voit donc tout de suite l'intérêt d'automatiser la spécification d'arguments divers. Les scripts ont tous été développés en Perl.

Nous avons profité du développement de ces plug-ins pour montrer le projet Nagios aux autres techniciens de l'entreprise qui nous ont fait remarquer qu'il serait bon de pouvoir contrôler la taille des tables et des bases sauvegardées par MySQL. C'est dans cette optique que j'ai développé un nouveau plug-in en Perl qui réalise automatiquement ce contrôle. Un peu à la manière du plug-in MySQL, après une consultation dynamique des sockets utilisées par les serveurs MySQL, le plug-in réalise une requête de type *show databases* afin de récupérer toutes les bases utilisées. Puis pour chaque base une requête *show table status* est exécutée afin de récupérer les informations sur toutes les tables de la base. Les informations sont ensuites traitées pour contrôler si elles sont aux normes.
Après avoir rédigé le script (disponible en annexe : **Annexe 8 :** check_mysqldatasize.pl, page 71), nous l'avons testé sur chaque machine concernée avant de le définir en tant que service au niveau de Nagios. Malheureusement ce test ne fut pas concluant. En effet la quantité d'informations à vérifier étant vraiment importante, le contrôle s'éternise : on a pu compter une exécution longue de 179 minutes sur l'un des hôtes !
Même si en relançant de suite après le programme, le temps d'exécution est grossièrement divisé par 5 (le même process relancé de suite après n'a duré que 34 minutes), le temps reste trop important pour une exécution via NRPE. Nous avons donc du abandonner ce script car ce dernier n'était donc pas réellement adapté ! Le temps d'exécution est plus court tout simplement car une partie des résultats est resté stocké en mémoire.

La deuxième adaptation a été d'utiliser NRPE même pour contrôler les ressources de la station Nagios. Cette adaptation est directement issue de la première : nous ne souhaitions pas avoir un fichier de configuration encore plus important et d'avoir des services Nagios avec des noms différents qui supervisent la même chose.

Dans cette optique nous avons décidé de superviser les ressources de la machine Nagios de la même façon que les machines distantes sont contrôlées.

La dernière adaptation a été la modification du script de démarrage de NRPE. En effet il a fallu prendre en compte les règles d'Iptables de chaque machine. Par défaut tout paquet était rejeté car le port pour les communications NRPE était fermé, j'ai donc du rajouter de démarrage les règles Iptables suivantes :

- ➔ Ouvrir le port Nrpe en entrée uniquement si l'adresse source est l'adresse de la station Nagios.
- ➔ Ouvrir le port Nrpe en sortie uniquement si l'adresse destination est l'adresse de la station Nagios.

Il est évident que le script de fermeture doit être modifié en conséquence en fermant les ports qui auront été ouverts par le script de démarrage. Le script a été écrit en shell et se trouve en annexe de ce rapport (**Annexe 4** : init-script.in, page 66).

Nous avons aussi changé de versions de plug-ins car ceux proposés dans la version 1.3.1 ne fonctionnaient pas tous correctement. En effet le *check_ping* ne renvoyait pas le bon code retour ce qui créait une alerte au niveau de Nagios. Lors du stage une nouvelle version des plug-ins (la 1.4.1) a été mise à la disposition de la communauté et après l'avoir déployée plus aucun problème dû aux plug-ins n'a été recensé.

6-3/ Résultat du déploiement

Voici quelques captures des principales sections de Nagios :
La page *service detail* est la page principale de Nagios, elle permet d'avoir très vite une vision globale des contrôles réalisés. L'intérêt de cette vue est de pouvoir apprécier les outputs retournés par les plug-ins, qui peut aider à la compréhension et à la justification d'un état. C'est donc pour cela que nous l'avons défini en tant que page d'accueil à l'interface Nagios.

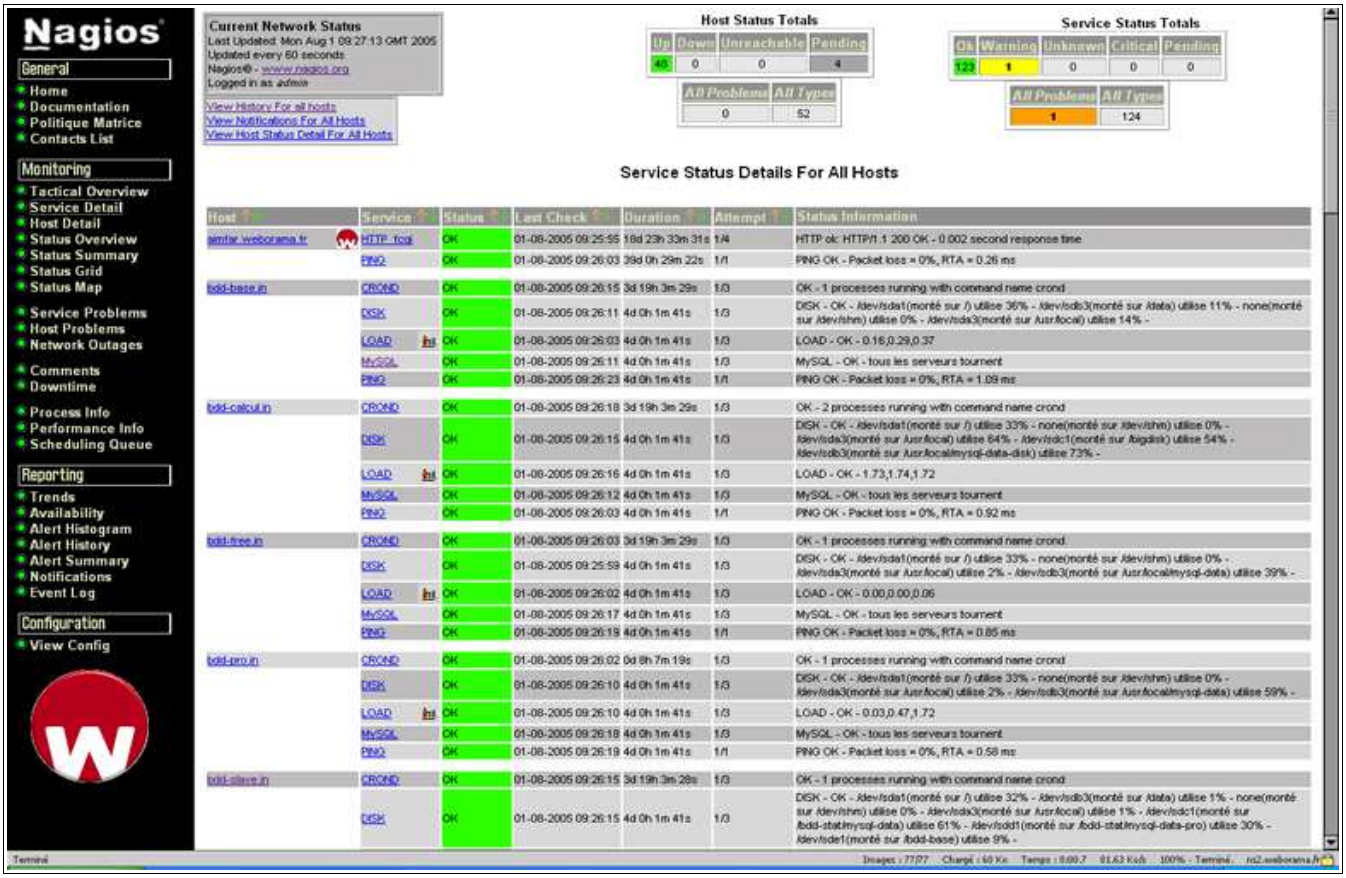

Fig. 12 : Nagios – Service Details

Sur la page de *service details* on peut voir plusieurs fois un logo représentant un graph :

Ces derniers sont en fait que des liens imagés vers le graphe du service associé. Par exemple si je clique sur l'icône du service trafic de l'hôte sxitch-ext1 voila ce que j'obtiens :

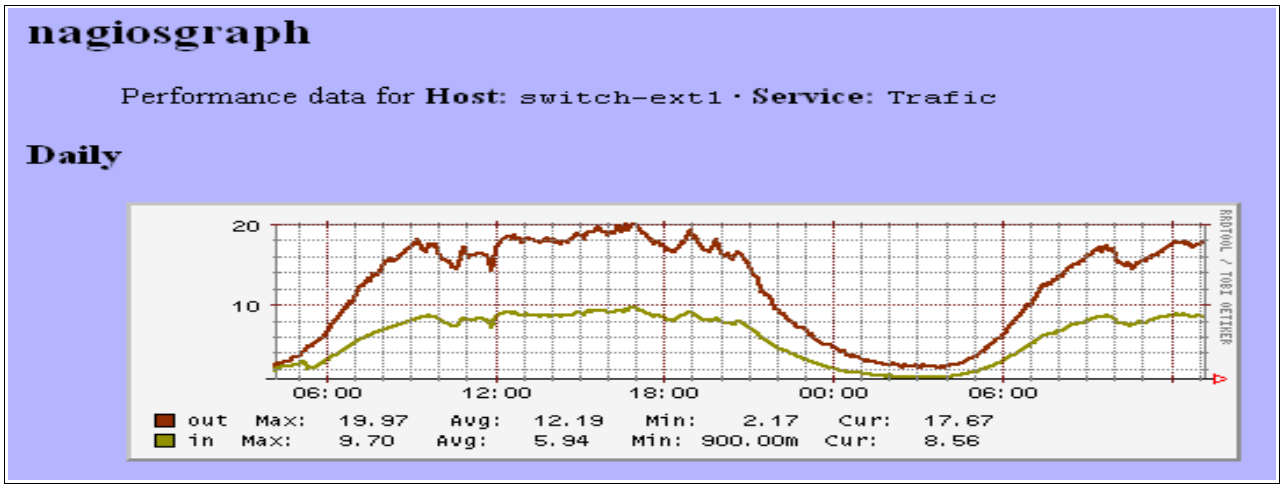

Fig. 13 : Nagios – Graph

Ce graph représente l'évolution du pourcentage d'exploitation de la bande passante en entrée et en sortie sur une journée. Mais sur la même page plusieurs graphes sont disponibles : l'évolution sur une journée, une semaine, un mois et une année. Les graphes sont générés par un script cgi qui récupère les informations dans les bases rrd.
La *statut grid* reprend les informations affichées par la page service details. L'avantage est que l'affichage est beaucoup plus léger que pour le *service details* et que les hôtes supervisés sont rangés par groupe.

Une autre vue peut être intéressante pour avoir un plan de l'architecture supervisée : la *statut map.*

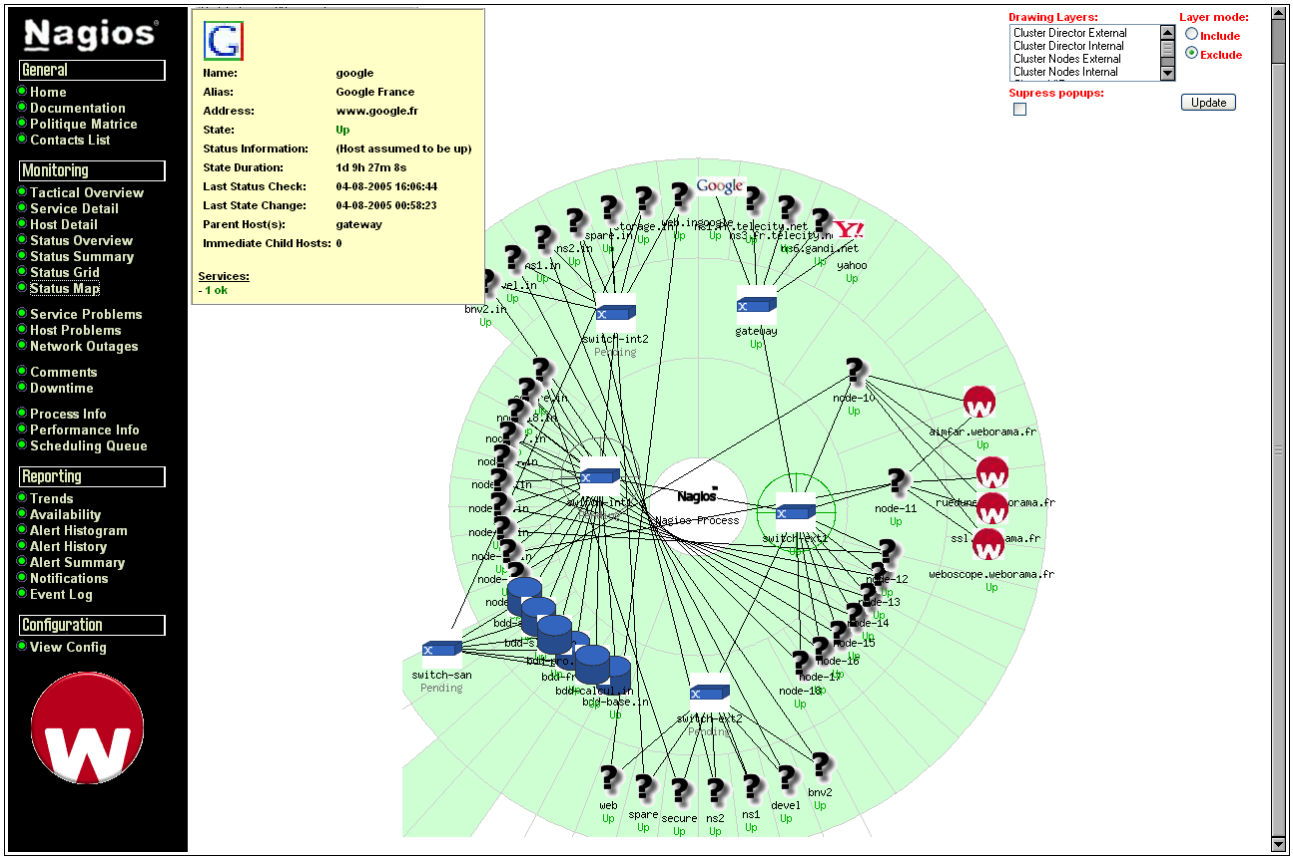

Fig. 14 : Nagios – Status Map

Cette représentation graphique est utile pour comprendre l'origine d'un problème : par exemple si un switch tombe, il est normal que certains hôtes ne répondent plus et c'est donc dans ce genre de cas que cette vue sera utile afin de cibler rapidement le problème.

La page *eventlog* permet d'avoir une vision globale des évènements antérieurs que Nagios a pu recensé. Cette page est une interprétation dynamique et illustrée des logs que Nagios crée.

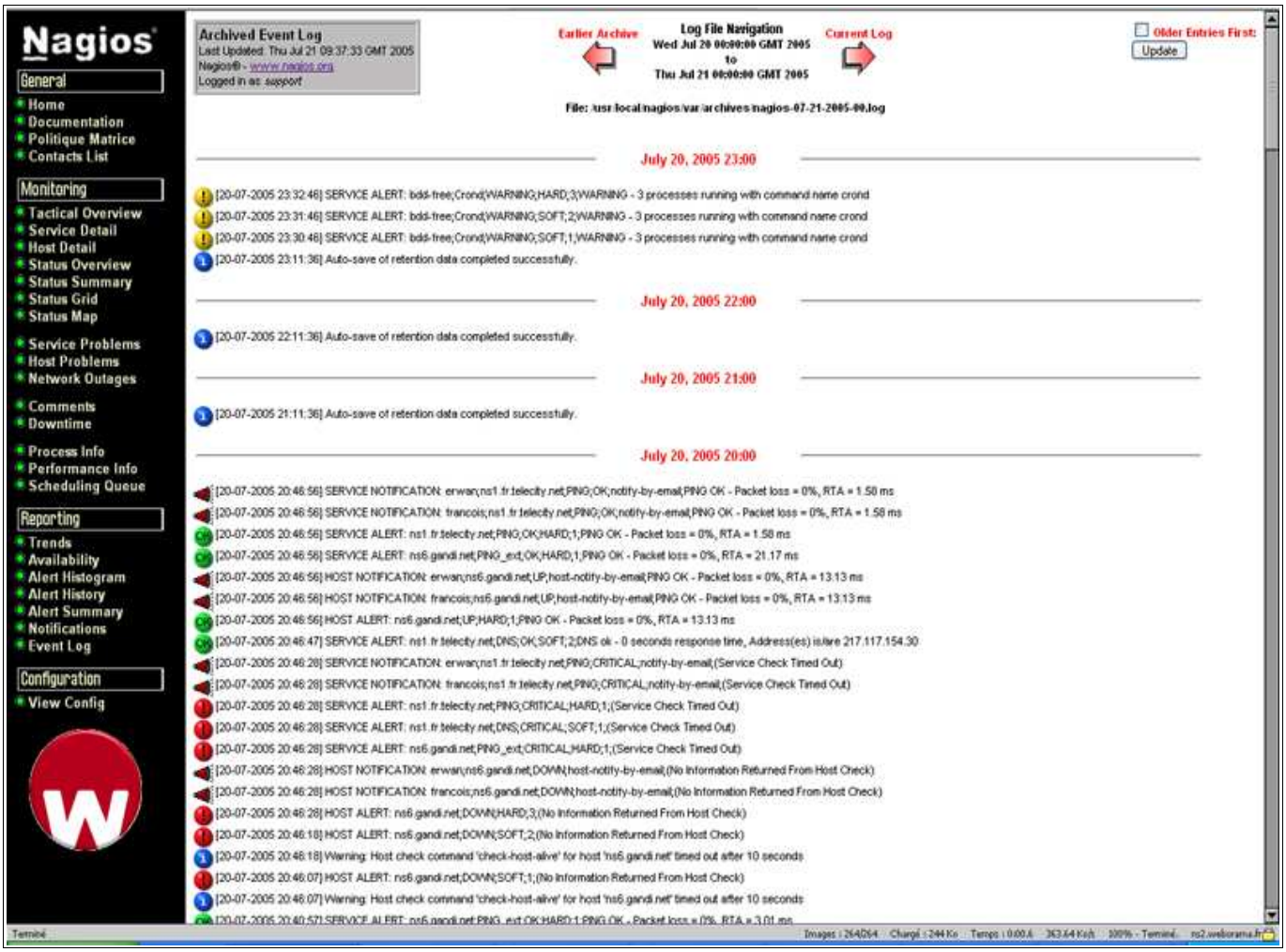

Fig. 15 : Nagios - Eventlog

Par le biais des autres menus, il est possible de générer des rapports ou des graphs toujours à partir des logs :
Trends : permet d'avoir une représentation graphique, selon la date, du statut (« up », « down », « unreachable », « indeterminate ») de chaque hôte ou service spécifié.
Availability : permet d'avoir un rapport sous la forme d'un tableau présentant des statistiques des états d'un hôte sur une période donnée.

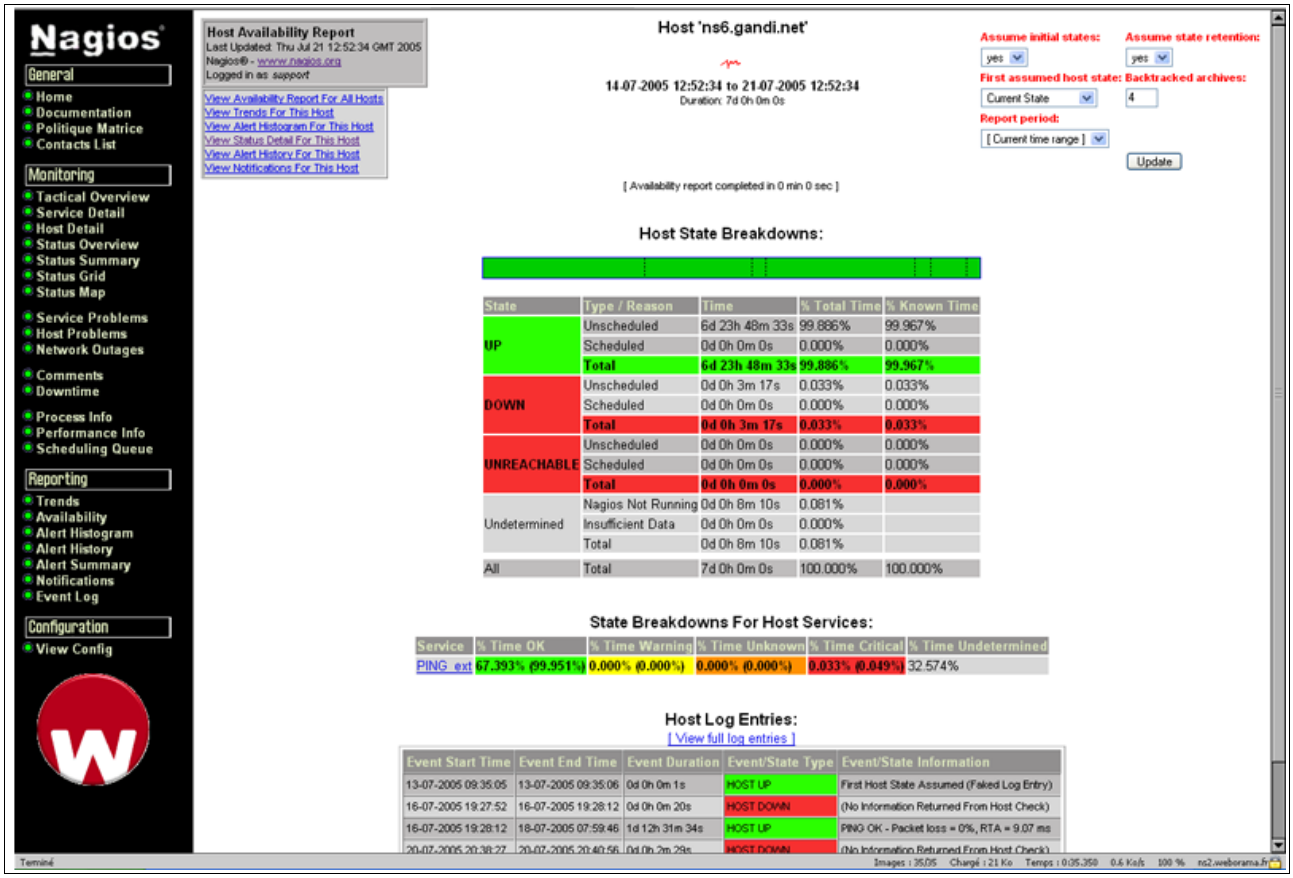

Fig. 16 : Nagios - Availability

Alert Histogram : cette vue permet de générer un graphe présentant le nombre d'alertes générées sur une période donnée.

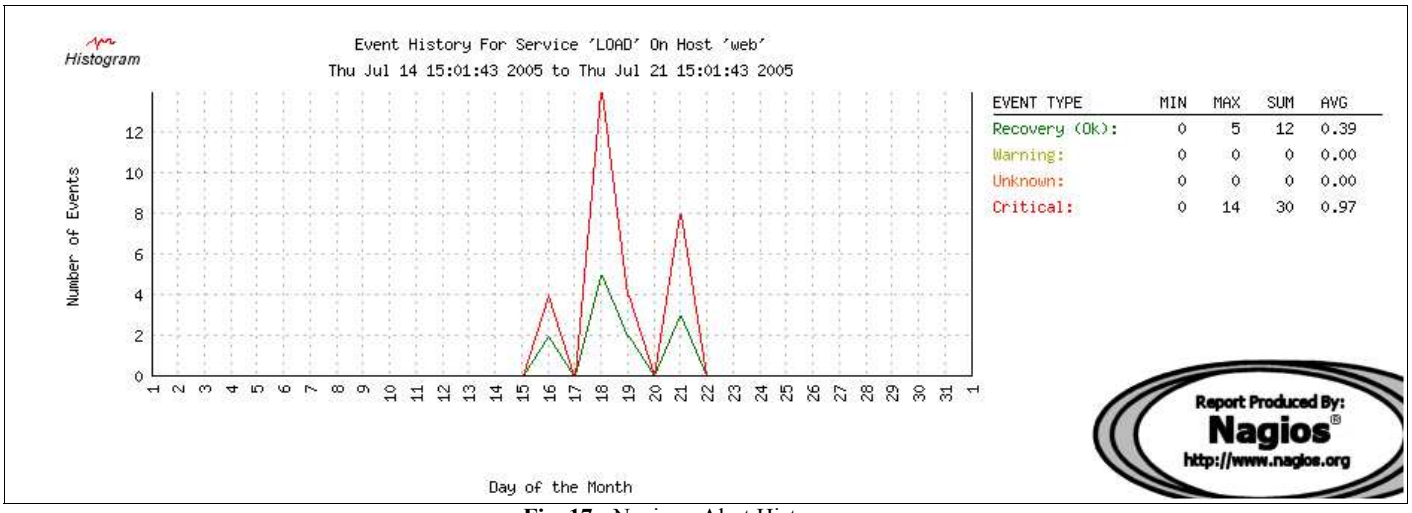

Fig. 17 : Nagios - Alert Histogram

Alert History : visualisation des événements importants sous la forme d'un listing

Displaying most recent 25 of 352 total matching alerts

Time	Alert Type	Host	Service	State	State Type	Information
21-07-2005 08:45:26	Service Alert	bdd-slave	DISK	UNKNOWN	SOFT	NRPE: Unable to read output
21-07-2005 07:41:06	Service Alert	web	LOAD	OK	SOFT	LOAD - OK - 2.83,4.26,3.94
21-07-2005 07:40:06	Service Alert	web	LOAD	CRITICAL	SOFT	LOAD - CRITICAL - 5.49,4.93,4.12
21-07-2005 07:39:07	Service Alert	web	LOAD	CRITICAL	SOFT	LOAD - CRITICAL - 7.48,4.93,4.06
21-07-2005 07:32:07	Service Alert	web	LOAD	OK	HARD	LOAD - OK - 2.92,5.05,3.97
21-07-2005 07:30:06	Service Alert	web	LOAD	CRITICAL	HARD	LOAD - CRITICAL - 3.09,6.23,4.16
21-07-2005 07:29:06	Service Alert	web	LOAD	CRITICAL	SOFT	LOAD - CRITICAL - 6.51,7.38,4.37
21-07-2005 07:28:06	Service Alert	web	LOAD	CRITICAL	SOFT	LOAD - CRITICAL - 10.80,7.96,4.35
21-07-2005 07:14:26	Service Alert	bdd-calcul	Crond	OK	HARD	OK - 2 processes running with command name crond
21-07-2005 07:03:26	Service Alert	bdd-calcul	Crond	WARNING	HARD	WARNING - 3 processes running with command name crond
21-07-2005 07:02:27	Service Alert	bdd-calcul	Crond	WARNING	SOFT	WARNING - 3 processes running with command name crond
21-07-2005 07:01:26	Service Alert	bdd-calcul	Crond	WARNING	SOFT	WARNING - 3 processes running with command name crond
21-07-2005 05:36:16	Service Alert	storage	LOAD	OK	HARD	LOAD - OK - 0.00,0.67,1.99
21-07-2005 04:10:06	Service Alert	bdd-slave	Crond	WARNING	HARD	WARNING - 5 processes running with command name crond
21-07-2005 04:05:06	Service Alert	bdd-slave	Crond	CRITICAL	HARD	CRITICAL - 6 processes running with command name crond
21-07-2005 04:03:56	Service Alert	bdd-pro	Crond	OK	SOFT	OK - 2 processes running with command name crond
21-07-2005 04:03:56	Service Alert	node-17	HTTP443	OK	SOFT	HTTP ok: HTTP/1.1 200 OK - 0.002 second response time
21-07-2005 04:03:56	Service Alert	node-15	HTTP443	OK	SOFT	HTTP ok: HTTP/1.1 200 OK - 0.027 second response time
21-07-2005 04:03:06	Service Alert	node-14	HTTP443	OK	SOFT	HTTP ok: HTTP/1.1 200 OK - 0.002 second response time
21-07-2005 04:03:06	Service Alert	node-18	HTTP443	OK	SOFT	HTTP ok: HTTP/1.1 200 OK - 0.003 second response time
21-07-2005 04:02:56	Service Alert	node-17	HTTP443	CRITICAL	SOFT	Connection refused by host
21-07-2005 04:02:56	Service Alert	bdd-pro	Crond	WARNING	SOFT	WARNING - 3 processes running with command name crond
21-07-2005 04:02:56	Service Alert	node-15	HTTP443	CRITICAL	SOFT	Connection refused by host
21-07-2005 04:02:06	Service Alert	node-14	HTTP443	CRITICAL	SOFT	Connection refused by host
21-07-2005 04:02:06	Service Alert	node-18	HTTP443	CRITICAL	SOFT	Connection refused by host

Fig. 18 : Nagios - Alert History

Alert Summary : permet de n'afficher que les dernières alertes générées.

Nagios est aujourd'hui entièrement opérationnel et exploitable par toute l'équipe technique de Weborama. De plus il est facile de le mettre à jour avec de nouveaux plug-ins à utiliser ou hôtes à superviser. La mission principale de ce stage a donc été entièrement accomplie.

6-4/ Add-on Nagios : Ntray

Dans le cadre du stage j'ai pu mettre en place un système de notifications par e-mails lorsqu'un certain comportement est détecté. Cependant pour suivre réellement l'évolution des états, l'utilisateur est obligé d'avoir une page de son browser dédié à Nagios. L'idée était donc de trouver une solution pour être averti de façon encore plus significative que les mails sans pour autant rester connecté à l'interface Web.
La solution permettant de faire cela se nomme Ntray, qui est développée par Rob Wagner, et que l'on peut télécharger sur le site dédié aux projets parallèles de Nagios : NagiosExchange (http://www.nagiosexchange.org/Frontends.37.0.html?&tx_netnagext_pi1[p_view]=168) . Cette petite application, qui tourne en background et qui est très légère en consommation de ressource, permet de suivre l'évolution de Nagios sans pour autant être connecté à l'interface Web. De façon périodique, elle va consulter les outputs gérés par les CGI de status pour afficher une icône de la couleur de l'état de Nagios : rouge pour CRITICAL, jaune pour WARNING, vert pour OK et orange pour UNKNOWN. En plus de l'icône, un pop-up, recensant les problèmes, est généré. Il est de surcroît accompagné d'un son configurable suivant l'état de Nagios.
L'intérêt est le suivant : le mail reste une méthode de notifications totalement asynchrone et rien n'oblige le destinataire à lire le mail à la réception et rien ne justifie que le mail a bien été délivré ! Donc dans cette optique si pour une raison quelconque l'utilisateur n'a pu consulter ses mails cette application pourra tout de même le prévenir d'un problème. De plus grâce au message du pop-up il est possible d'avoir directement accès a la page concernée de l'interface Web de Nagios, afin d'agir au plus vite : comme acquitter, désactiver le contrôle du service...
Lorsque l'application Ntray tourne, on peut suivre l'évolution de Nagios dans la taskbar comme suit :

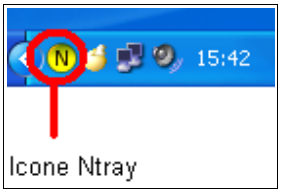

Fig. 19 : Ntray dans taskbar

On peut constater que l'icône est jaune ce qui correspond à un état WARNING. Si l'on souhaite savoir qu'est ce qui ne fonctionne pas correctement il suffit de double-cliquer sur cette icône ce qui a pour conséquence d'ouvrir la fenêtre suivante :

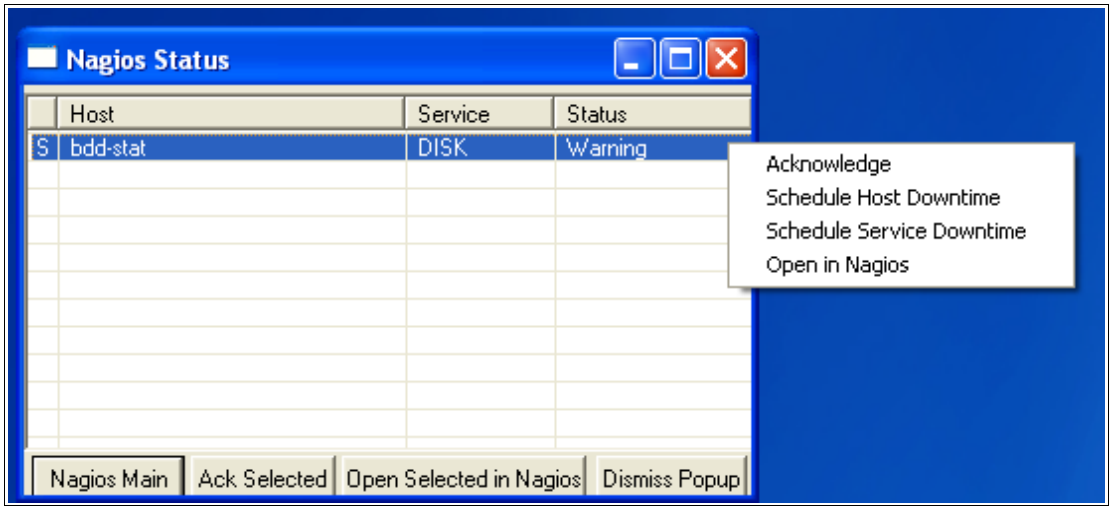

Fig. 20 : Ntray fenetre

Ainsi on peut voir directement d'ou vient le problème (service et hôte concernés) et directement agir dessus sans passer par l'interface Web. Bien sur en plus les contacts concernés recevront un email de notification.

Pour éviter à tous les membres de l'équipe technique d'installer ce petit Add-on, nous avons utilisé une des machines inactives de l'entreprise sur laquelle nous avons installé Ntray. En rajoutant de quoi émettre un son relativement fort, cette machine sera donc une sorte de tour de contrôle. En effet j'ai par la suite change les sons par défaut pour les remplacer par des sons marquants : comme une sirène d'alarme lorsqu'un sérieux (CRITICAL) problème a été détecté.

6-5/ Add-on Nagios : Notification par SMS

Même avec l'add-on précédemment décrit, si personne n'est a portée de la machine Ntray ou si le réseau entier tombe (ce qui empêche Ntray de fonctionner) , il n'y a aucun moyen d'être notifié. L'idée etait de trouver un moyen ultime permettant d'être prévenu indépendamment de l'état du réseau. La solution à ce problème est d'envoyer un SMS par le biais d'un modem dédié : un modem GSM.
Une fois la solution identifiée, j'ai donc cherché avec mon tuteur un outil et un modem correspondant à nos besoins. L'outil a très vite été trouvé : Smstools. Ce petit programme développé par Stefan Frings, permet d'envoyer un sms grâce à un modem connecté sur un port série d'une station sous Unix. Pour cela le programme utilise des commandes AT qui sont un jeu de commandes permettant d'envoyer des commandes sur des ports séries.
Pour le modem nous avons regardé les modems compatibles avec ce programme : la seule réelle contrainte est que le modem doit être compatible avec les normes GSM 07.05 ou GSM 07.07 qui sont les normes spécifiant que le modem est capable d'utiliser les commandes pour envoyer des SMS. Une contrainte supplémentaire a été imposée par l'entreprise : la solution doit avoir un coût inférieur à 200 euros hors taxes.
La recherche a abouti sur deux modems : le Falcom Twist et le Wavecom M1306. Malgré la similitude des caractéristiques techniques le choix fut très vite fait : en effet le Falcom Twist n'est plus produit en port série depuis la fin Juin, nous avons donc choisi de prendre le Wavecom M1306.
Une fois le matériel nous avons pu tester le système d'envoi de SMS. La configuration étant relativement simple et aisée, j'ai donc pu directement essayé d'intégrer l'envoi de SMS à Nagios. Pour cette opération je me suis servi du champs, dans la définition de contact, *page*r. En effet ce champ permet de spécifier un numéro de téléphone ou de bippeur, et ce champs est utilisable via la macro Nagios *$CONTACTPAGER$*. Ensuite il a suffi de rajouter dans les champs où on spécifie les commandes de notifications (*<service_notification_commands>* et *<host_notification_commands>*) le nom de la commande pour envoyer des sms qui est définie dans le fichier *miscommand.cfg* .

Voilà ce que donne les modifications que j'ai apporté :

1/ Définir dans le *miscommand.cfg* la commande pour envoyer des sms :

```
define command{
        command_name    notify_by_sms
        command_line    /usr/local/smstools/bin/smsd $CONTACTPAGER$ « Probleme sur
                          $HOSTNAME$ , $SERVICESTATE$/$STATETYPE$ »
        }
```

2/ Rajouter les éléments nécessaires dans la définition d'hôte

```
define contact{
        contact_name                    erwan
        alias                           Erwan
        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,notify_by_sms
        host_notification_commands      host-notify-by-email,host_notify_by_sms

        email                           erwan@weborama.fr
        pager                           33664099787
        }
```

Il n'y a absolument rien de plus a faire, il faut donc modifier de la même manière tous les contacts qui sont concernés par ce genre de notifications.
Et si un problème survient voilà comment se présentera la notification par SMS :

Fig. 21 : Photo de notification par SMS

VII/ Projets parallèles

Même si la mise en place d'un outil de supervision reste le sujet principal de mon stage, il n'en est pas pour autant le seul projet. En effet, entre la partie théorique de recherche de solution et le déploiement de Nagios quelques semaines se sont écoulées me laissant disponible pour d'autres projets décrits dans cette section. Ces projets sont assez caractéristiques et typiques aux systèmes et réseaux.

7-1 / Réplication de serveur MySQL

En attendant de pouvoir commencer le déploiement de Nagios, mon tuteur m'a proposé de travailler sur des serveurs MySQL. Mes objectifs étant :

- comprendre le fonctionnement,
- être capable de mettre en place une architecture pour la réplication des bases de données,
- et proposer une procédure d'installation d'un tel système

La première partie de ce projet parallèle est essentiellement théorique. L'objectif de cette partie n'est pas de connaître toutes les subtilités de la configuration d'un serveur MySQL mais de comprendre les principes et les enjeux.
Pour cela deux livres m'auront été très utiles : *Managing & Using MySQL* (par George Reese, Randy Jay Yarger & Tim King chez O'Reilly) et *High Performance MySQL* (par Jeremy D. Zawodny & Derek J. Balling chez O'Reilly).

Voici donc comment fonctionne la réplication des serveurs MySQL :
La réplication met en scène deux entités : le master (serveur maître) et les slaves (les serveurs esclaves). Le master reçoit les requêtes des clients, il les "parse" et les exécute sur ses propres bases. Toutes les requêtes modifiant les bases sont sauvegardées dans les logs binaires. Les slaves lisent le fichier de logs binaires afin de récupérer les requêtes qui ont modifié les bases, puis ils exécutent ces requêtes sur leurs propres bases locales.

Voici un schéma expliquant plus précisément le fonctionnement de la réplication :

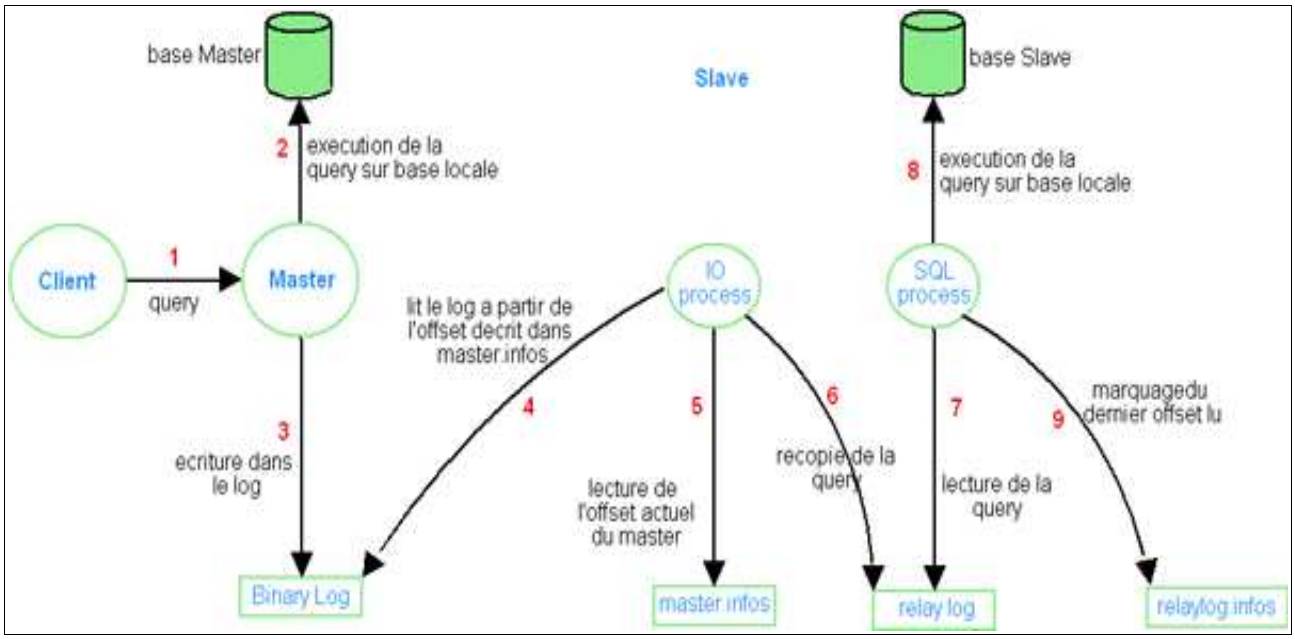

Fig. 22 : Fonctionnement de la replication MySQL

Comme le montre ce schéma, le mécanisme de réplication au niveau des SLAVES repose sur 2 threads indépendants :

- ➔ le thread IO permet de lire le contenu du fichier de logs binaires du MASTER et le recopie dans le fichier de *"relay log"*. Le thread lit uniquement à partir du dernier offset lu. Cette information est présente dans le fichier *master.info* . Ce fichier est mis a jour, avec la dernière valeur de l'offset lu, chaque fois que le thread IO lit les logs binaires du MASTER.
- ➔ le thread SQL va exécuter les requêtes qui auront été recopiées dans le fichier de *"relay log"*. Il exécute les requêtes à partir de la dernière requête executée. Pour connaître cette information le thread SQL lit le fichier *relay-log.info*. Pour chaque requête exécutée le thread met à jour la valeur de l'offset du fichier *relay-log.info* pour garder la cohérence.

La seconde partie de ce projet était de réaliser un document décrivant les procédures nécessaires pour mettre en place la réplication. Pour cela je disposais de 2 machines, sur lesquelles les serveurs MySQL v4.0.24 étaient installés, pour les tests. Pour ce remettre dans les conditions dans lesquelles la procédure allait être utilisée, il fallait considérer que le serveur MASTER possédait déjà une base que des utilisateurs avaient modifiés. En effet partir de deux serveurs vierges ne pose aucun problème pour mettre en place la réplication. Dans le cas contraire certaines manipulations particulières sont nécessaires.

Après plusieurs tests de configuration des serveurs MySQL, la réplication fonctionnait sur une petite configuration. J'ai donc pu écrire la procédure à suivre pour mettre en place la réplication de serveur MySQL.
Cette procédure se décompose de la façon suivante :

- verrou en écriture des bases sur le Master, de façon à ce qu'aucune modification sur la base n'intervienne
- copie (on parle de snapshot) des bases verrouillées sur les machines Slaves
- configuration des Slaves
- déverrouillage des bases du Master

Les informations complètes de la manipulation sont disponibles en annexe dans le howto en annexe (**Annexe 2 :** mysql-replication.howto, page 63) pour la mise en place de la réplication d'un serveur MySQL.

La troisieme étape de ce projet est d'appliquer le howto pour chaque machine concernée par cette intervention. Ce déploiement est pévu pour la fin du stage.

7-2/ Système de sauvegarde automatique des documents Windows des utilisateurs

Le but de ce projet parallèle est de proposer une solution permettant de sauvegarder de façon automatique les données des utilisateurs Windows. Jusqu'à présent aucun réel système n'a vraiment été implémenté.
L'objectif de ce projet est d'atteindre les points suivants :

- Le système devra être en mesure de centraliser les données sur un serveur dédié aux sauvegardes.
- Il faudra garder plusieurs versions d'archives (sauvegarde du jour ainsi que les sauvegarde à Jour-1 et Jour-2).
- Il faudra de façon récurrente externaliser les backups sur un DVD par exemple. Ainsi cela permettra d'avoir toujours des sauvegardes complètes des environnements sur un support fiable et externe permettant sa mise à l'abri (à la banque, coffre...).

La première étape de ce projet consistait à trouver et à tester plusieurs solutions de sauvegarde et restauration d'environnements. Plusieurs logiciels réalisent cette fonction :

- ➔ Ntbackup de Microsoft
- ➔ Facilosave de EveryDev
- ➔ BackupGenie de Speedman

Cependant l'outil de sauvegarde doit répondre aux besoins suivants :

- Permettre de paramétrer totalement les sauvegardes (pouvoir choisir dans une arborescence les dossiers et fichiers à sauver)
- Permettre de paramétrer totalement les restaurations (pouvoir définir l'emplacement de restauration et possibilité de faire des restaurations partielles : uniquement de certains dossiers ou fichiers par exemple)
- Pouvoir créer des planifications de sauvegardes ou de restaurations
- Doit pouvoir travailler de façon transparente pour les utilisateurs
- Permettre de créer des filtres sur les fichiers à sauvegarder (filtre par extension ou par nom de fichiers)
- Eventuellement avoir une interface "friendly-to-use"
- Etre gratuit

Facilosave ne correspondait pas vraiment aux besoins. Même s'il existe une version gratuite de ce logiciel, elle reste beaucoup trop limitée : nombre de filtres limités, emplacement de restauration figé, très gourmand en ressources systèmes... BackupGenie est un très bon outil qui correspond en tout point à nos besoins (il était même utilisé par certains salariés) cependant l'utiliser de façon automatique par le biais d'un script semble plus compliqué (en ce qui concerne le passage en paramètre et les options) que le dernier produit disponible : Ntbackup.

Le choix s'est donc posé sur ce dernier qui correspond en tout point aux besoins sauf qu'il n'est pas réellement gratuit. En fait Ntbackup est intégré à Windows, il est donc compris dans le prix de la licence Windows. Tous les postes utilisateurs de l'entreprise étant sous Windows Xp Pro ils possèdent donc tous par défaut la même version de l'outil de sauvegarde. Il est intéressant aussi de noter que Ntbackup peut être lancé en ligne de commande, à partir d'une console DOS, en passant en paramètres toutes les options de sauvegardes. Cet aspect est intéressant car il permettra de rédiger un petit script pour rendre les sauvegardes automatiques.

La deuxième étape consiste à l'externalisation des données. L'externalisation des données sera assuré par la gravure d'un DVD contenant toutes les dernières archives des sauvegardes des utilisateurs. En suite il suffira de mettre le DVD à l'abri (un coffre par exemple). La fréquence d'externalisation des données sera fixée à une fois tous les mois.

La procédure de sauvegarde se doit d'être entièrement automatique. Elle doit en plus gérer les points suivants :

- Générer la sauvegarde de l'utilisateur à partir d'un fichier de listing *.bks* créé par Ntbackup (qui est facilement mise à jour si l'utilisateur souhaite modifier ses sauvegardes)
- Créer si nécessaire un répertoire particulier pour l'utilisateur
- Gérer la rotation des archives afin de garder uniquement les 3 archives les plus récentes
- Ne permettre qu'une sauvegarde journalière (pour éviter que l'utilisateur ne perde trop de temps si il est obligé de redémarrer par exemple)

Le script, développé pour réaliser ces tâches, a été écrit en batch DOS (code disponible en annexe : **Annexe 3 :** backupautoV23.cmd, page 63). Pour que la sauvegarde soit réalisée de façon automatique, il sera lancé au démarrage de Windows. Si l'utilisateur souhaite faire une sauvegarde instantanée il n'aura qu'a lancé le script lui même, auquel cas il est responsable de la fréquence d'archivage de ses données.

Voici une capture d'écran de l'exécution du script :

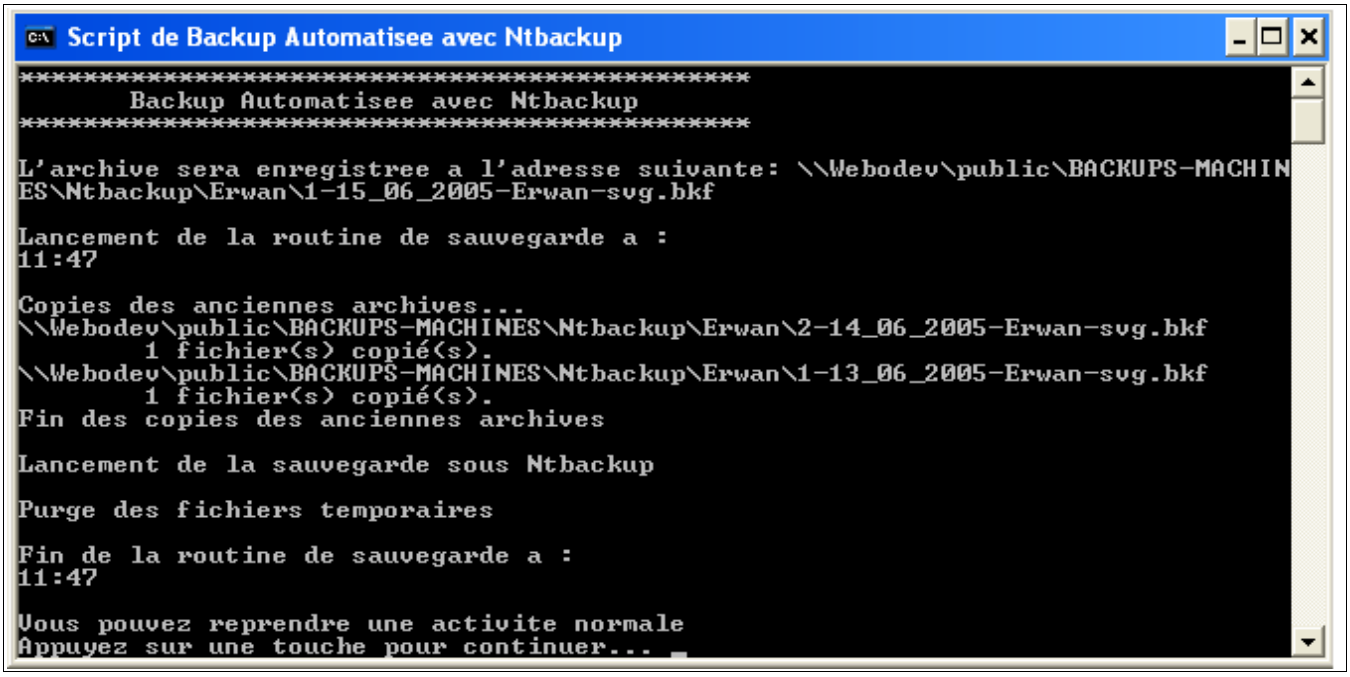

Fig. 23 : Capture d'écran de l'exécution du script

Le problème majeur qui s'est posé était l'activation des connexions réseaux pour les utilisateurs qui souhaitaient faire une sauvegarde automatique au démarrage. En effet il se peut, pour des raisons pas toujours compréhensibles, que lorsque le script se lance au démarrage, l'ordinateur ne découvre pas encore le réseau. Les conséquences sont assez radicales puisqu'aucune sauvegarde ne pourra être faite (vu que le poste ne trouve pas l'endroit où sauver les données). Pour parer ce problème, il a été nécessaire d'insérer une temporisation tel que : "tant que le répertoire de sauvegarde n'est pas trouvé alors on attend". Cependant aucune commande sleep n'existe, pour créer une temporistation j'ai donc utilisé un système de pings locaux (sur l'interface 127.0.0.1) en redirigeant le output pour ne pas créer d'affichage pouvant être troublant pour l'utilisateur.

L'algorithme final est donc le suivant :

```
début :
        Ping pendant 3 secondes
        Test de l'existance du répertoire de sauvegarde
        Si le répertoire n'a pas été retrouvé
                on retourne à "début"
               Si c'est le 5ème essai
                      fin de la procédure
        Sinon on poursuit la procédure de sauvegarde
        Compte du nombre d'archives
        Si une archive a déjà été crée aujourd'hui
                fin du script
        Rotation des archives
        Si le compte est supérieur à 3
                on efface l'archive la plus ancienne
        Génération de la nouvelle archive
```

7-3/ Tâches diverses d'administration

Le travail dans l'administration réseau ne se limite pas à des projets ponctuels, il y a une certaine maintenance à faire dans la gestion des mails par exemple : création de nouveaux comptes, mise à jour des listes de diffusions, mise en place des messages d'absence...
J'étais aussi en charge de la surveillance des pannes imprimantes. Lorsque l'imprimante principale est tombée, il a fallu mettre en place un nouveau serveur d'impressions de façon à ce que toute l'entreprise puisse poursuivre une activité "normale".
Ce ne sont donc pas les tâches les plus importantes mais elles font partie intégrante du travail d'un administrateur réseaux et systèmes.

VIII/ Conclusion

Avant ce stage, j'appréhendais beaucoup le monde professionnel de l'informatique, sûrement à cause de mon manque d'expérience et du manque de manipulation et de projets pratiques de cette année. Malgré cela un projet important m'a été confié : la supervision du réseau et des systèmes de Weborama.
Pour remplir cette tâche j'ai été intégré dans l'équipe technique de Weborama où j'ai été considéré comme un membre à part entière ce qui a grandement facilité mon intégration au sein de l'équipe et de l'entreprise. Mon projet n'était pas réalisé en marge du cours de l'entreprise, vu que régulièrement je tenais compte de mon avancée ou des problèmes rencontrés à mon tuteur ou à un autre membre de l'équipe. Au delà de ce projet je participais aux tâches quotidiennes d'un administrateur réseaux : gestion des mails, mise en place de serveurs d'impressions ou la participation à d'autres projets : réplication de serveurs MySQL ou mise en place d'une procédure de sauvegarde automatique des données des utilisateurs. Toujours de la même manière ces projets ont été réalisés en équipe ou avec l'aide de mon tuteur.
Grâce à cet encadrement, j'ai pu vraiment me sentir à l'aise, n'hésitant pas à proposer des solutions et donner mon point de vue sur divers choix techniques. Il y a donc eu durant tout ce stage un échange entre les professionnels et moi, ce que j'estime être vraiment positif et primordial pour ce genre d'expérience. Ces échanges permanents contribuent vraiment à acquérir des compétences et un savoir-faire par la transmission d'idées et de vécus. C'est pourquoi je me suis réellement senti progresser au côté de cette équipe technique toujours abordable et plus que compétente.

Cette expérience m'a, de plus, permis de constater que dans ce domaine il y a différents rôles avec lesquels il faut vite apprendre à jongler que ce soit pour l'administration courante ou le relais technique avec les utilisateurs par exemple... La polyvalence est donc une compétence très importante que je me suis efforcé de développer tout au long de ce stage.
J'ai pu approcher toutes sortes de technologies, de projets et d'environnements ne rendant jamais le travail monotone et récurrent. Evoluant aussi bien dans un environnement Windows que Linux, je devais connaître l'architecture et le fonctionnement global du réseau de l'entreprise afin d'optimiser au maximum la supervision qui restait ma mission principale.

Pour toutes ces raisons j'estime avoir eu un stage complet et positif aussi bien humainement, professionnellement que techniquement. J'ai pu au sein d'une équipe contribuer et participer à des projets qui aujourd'hui sont aboutis et font avancer l'entreprise. Cet aboutissement me remplit de satisfaction vis à vis de cette première réelle approche du monde informatique professionnel et me permet de me sentir utile dans le domaine dans lequel j'évolue désormais.

Sources

Nagios

http://nagios.org/ : site officiel du projet Nagios, on y retrouve les sources du projets ainsi qu'une FAQ complète.
http://nagiosexchange.org : site officiel des projets parallèles a Nagios : addons, plug-in, thèmes graphiques...
http://forums.opsyx.com/ : forum francais sur divers projet concernant la supervision Nagios, Cacti, Apan... très utile pour avoir un support et un retour sur les différents outils et configurations possibles.
http://docs.guill.net/article.php3?id_article=2 : site reprenant les étapes de l'installation de Nagios.
http://wp.wikidev.net/Nagios : wiki entièrement dédié a Nagios.
http://xavier.dusart.free.fr/netsaint/documentation-0.0.6/oldplugins.html : site expliquant ce que fait chacun des plug-ins officiels de Nagios et qui explique comment les utiliser.
http://sourceforge.net/project/showfiles.php?group_id=71182 : liens vers la documentation officielle de Nagios en français.
http://nagiosplug.sourceforge.net/developer-guidelines.html#PLUGOUTPUT : site expliquant la démarche et la logique à suivre pour créer des plug-ins compatibles avec Nagios.
http://www.nagiosexchange.org/Frontends.37.0.html?&tx_netnagext_pi1[p_view]=168 : liens vers le projet Ntray qui permet de suivre l'évolution de Nagios sans utiliser de browser.
http://smstools.meinemullemaus.de/ : liens vers le projet Smstools qui permet d'envoyer a partir d'une station Unix via un modem GSM.

RRDTool

http://people.ee.ethz.ch/~oetiker/webtools/rrdtool/ : site officiel du projet RRDTool.
http://ed.zehome.com/index.cgi?page=rrdtool : site donnant une description générale de RRDTool.
http://people.ee.ethz.ch/~oetiker/webtools/rrdtool/tutorial/rrdtutorial.html : tutorial sur RRDTool pour comprendre le fonctionnement des bases RRD et savoir optimiser leur création.

BIG BROTHER

http://www.bb4.com/ : site officiel sur le projet BigBrother.
http://demo.bb4.com/bb/help/help/administering_big_brother.htm : site anglais expliquant comment exploiter BigBrother et comment le configurer.

MySQL

Managing & Using MySQL par George Reese, Randy Jay Yarger & Tim King chez O'Reilly
High Performance MySQL par Jeremy D. Zawodny & Derek J. Balling chez O'Reilly

Ntbackup et Batch DOS

http://www.hotline-pc.org/sauvegarde.htm : page présentant les options disponibles avec Ntbackup.
http://www.hotline-pc.org/batch.htm : cours sur la programmation en batch DOS.

Divers

http://www.comp.leeds.ac.uk/Perl/start.html : cours sur l'écriture de scripts en Perl.
http://www.net-snmp.org/ : site officiel des outils Net-SNMP
http://lea-linux.org/ : site généraliste sur linux utile pour avoir des informations systèmes et des compléments de man.
http://www.linux-france.org/article/these/gpl.html : information sur les licences GPL GNU.
http://www.lexilogos.com/francais_langue_dictionnaires.htm : dictionnaire français utile quand on est nul comme moi en orthographe.

Annexes

Annexe 1 : nagios-install.howto

Howto : Install de Nagios v1.2
by erwan@weborama.fr

et des composants suivants :

- nagios-plugins v1.4.1 : plug-ins Nagios
- gd-2.0.15 : pour affichage de la statusmap
- nrpe v2.0 : pour la collecte d'information système a distance
- rrdtool v1.0.49 : pour le stockage des données et les générations des graphs.
- nagiosgraph v0.4 (ainsi que la lib CGI.pm v3.08)
- check_traffic v 0.90b

1/ Installation et configuration de Nagios et des plug-ins

les prérequis :

- un serveur Web
- les librairies graphiques suivantes (installées et testées durant la phase d'approche):
  freetype-2.1.4-4.0.src.rpm
  freetype-devel-2.1.4-4.0.i386.rpm
  libjpeg-6b-34.i386.rpm
  libjpeg-devel-6b-34.i386.rpm
  libpng-1.2.2-22.i386.rpm
  libpng-devel-1.2.2-22.i386.rpm

1.1/Installation

- Récuperer les archives nécessaires :
  nagios-1.2.tar.gz : http://www.nagios.org/download/
  nagios-plugins-1.4.1.tar.gz : http://sourceforge.net/project/showfiles.php?group_id=29880
  gd-2.0.15.tar.gz
- Créer un utilisateur dédié à Nagios (cf. la commande adduser) et un groupe dédié pour cet utilisateur (cf. la commande groupadd)
- Décompresser les 3 archives à l'aide de la commande : tar xvvfz nomarchive
- Dans le répertoire correspondant à gd-2.0.15 taper les commandes suivantes :
  A noter que pour l'installation de cette librairie graphique il faut vérifier que FreeType 2.x soit déjà installé.
  ./configure
  make
  make install
- Dans le répertoire correspondant à nagios-1.2 taper les commandes suivantes :
  ./configure (option pour le configure --with-nagios-user=SOMEUSER --with-nagios-grp=SOMEGROUP par defaut nagios)
  make all
  make install
  make install-init (installation d'un script de démarrage de Nagios lors du boot de la machine >> pas forcément nécessaire)
  make install-config (installation d'une configuration exemple si cette commande n'est pas lancée le répertoire /etc n'est pas créé)

Rem : si vous avez fait le make install-config, allez dans /etc du répertoire d'installation par default : /usr/local/nagios/etc, puis taper rename cfg-sample cfg *.cfg-sample

- Dans le répertoire correspondant à nagios-plugins v1.4.1 taper les commandes suivantes :
  ./configure --with-nagios-user=SOMEUSER --with-nagios-group=SOMEGROUP (par default ces deux options ont pour valeur "nagios")
  make
  make install

Donc en théorie on devrait retrouver dans /usr/local/nagios les répertoires suivants :
- /bin : contient l'executable Nagios
- /etc : contient les fichiers de configurations de Nagios, d'hôtes, de services...
- /libexec : contient les plugins installés par nagios-plugins v1.4.1
- /sbin : contient les scripts CGIs utilisés par Nagios
- /share : contient les fichiers HTML pour l'interface Web
- /var : contient les fichiers de log

1.2/Configuration

a/Accès Web

- Définir un alias pour accéder aux pages HTML de Nagios et un scriptalias vers les outils (soit les CGIs) de Nagios
  Rajouter dans le fichier de configuration du serveur Web (httpd.conf) les lignes suivantes :

```
ScriptAlias /nagios/cgi-bin/ /usr/local/nagios/sbin/      # définition du scriptalias

<Directory "/usr/local/nagios/sbin/">
AllowOverride AuthConfig
Options ExecCGI
Order allow,deny
```

```
Allow from all
</Directory>

Alias /nagios/ /usr/local/nagios/share/          # définition de l'alias

<Directory "/usr/local/nagios/share">
Options None
AllowOverride AuthConfig
Order allow,deny
    Allow from all
</Directory>
```

Après avoir relancé le serveur Web, théoriquement en tapant http://adressedelamachine/nagios/ on a accès à l'interface Web de Nagios.

- Rendre l'accès privé :

Dans le répertoire /usr/local/nagios/sbin créer le fichier .htaccess qui doit contenir les lignes suivantes :

```
AuthName "Nagios Access"
AuthType Basic
AuthUserFile /usr/local/nagios/etc/htpasswd.users
require valid-user
```

Créer la liste des personnes ayant accès : htpasswd -c /usr/local/nagios/etc/htpasswd.users nom_du_user

- Modifier /usr/local/nagios/etc/cgi.cfg de la manière suivante :

```
use_authentication=1                          # mettre à 1

authorized_for_system_information=nom_du_user     # ne mettre que l'utilisateur créé précedemment
authorized_for_configuration_information=nom_du_user
authorized_for_system_commands=nom_du_user
authorized_for_all_services=nom_du_user
authorized_for_all_hosts=nom_du_user
authorized_for_all_service_commands=nom_du_user
authorized_for_all_host_commands=nom_du_user
```

- De plus il faut suivre la procédure suivante pour permettre l'utilisation de certaines commandes via l'interface Web (mettre un commentaire, arrêter de contrôler un service ou d'envoyer des notifications pour un service spécifique...) :

```
usermod -G nagios nobody              # rajouter l'utilisateur du serveur Web au groupe nagios
mkdir /usr/local/nagios/var/rw        # créer le répertoire associé au fichier des commandes externes (cf. nagios.cfg)
chmod u+rwx /usr/local/nagios/var/rw  # modifications des droits
chmod g+rw /usr/local/nagios/var/rw
chmod g+s /usr/local/nagios/var/rw
```

Le répertoire doit avoir les permissions suivantes à la fin : drwxrws--- 2 nagios nagios.
Et ne pas oublier de redémarrer le serveur Web !

b/ Configuration

La configuration de Nagios passe par la modification des fichiers suivants :

- nagios.cfg : fichier de configuration principal de Nagios
- ressource.cfg : fichier permettant de définir les macros typiquement :
  $USER1$=/usr/local/nagios/libexec
- hosts.cfg : fichier permettant de définir les hôtes à superviser, par exemple :

```
# Generic host definition template
define host{
        name                            generic-host
        notifications_enabled           1       ; Notification activée
        event_handler_enabled           1       ; Gestionnaire d'événements activé
        flap_detection_enabled          1       ; Détection des osclillations activé
        process_perf_data               1       ; Process_perf_data activé
        retain_status_information       1       ; Informations liées à l'hôte retenues
        retain_nonstatus_information    1       ; Informations non-liées à l'hôte retenues

        register                        0       ; Template
        }
# Definition de ma machine
        define host{
        use                     generic-host            ; Template utilisé

        host_name               erwanode
        alias                   Erwan Host
        address                 192.168.0.62
        check_command           check-host-alive
        max_check_attempts      3
        notification_interval   3
        notification_period     24x7
        notification_options    d,u,r
        }
```

- hostgroups.cfg : fichier permettant de définir les groupes d'hôtes à superviser :

```
# Definition du groupe de test
        define hostgroup{
                hostgroup_name      test
                alias               Groupe de Test
                contact_groups      nagios-admins
                members             erwanode,nagiosnode,webodev,switch
        }
```

- services.cfg : fichier permettant de définir les services de supervision :

```
# Generic service definition template
        define service{
                name                            generic-service
                active_checks_enabled           1       ; Active Check activé
                passive_checks_enabled          1       ; Passive Check activé
                parallelize_check               1       ; Contrôle parallèle activé
                obsess_over_service             1
                check_freshness                 0       ; Contrôle de validité désactivé
                notifications_enabled           1       ; Notification activée
                event_handler_enabled           1       ; Gestionnaire d'événements activé
                flap_detection_enabled          1       ; Flap detection is enabled
                process_perf_data               1       ; Process_perf_data activé
                retain_status_information       1       ; Informations liées à l'hôte retenues
                retain_nonstatus_information    1       ; Informations non-liées à l'hôte retenues

                register                        0       ; Template
        }

# Definition du Service Ping
        define service{
                use                             generic-service         ; Template utilisé

                host_name                       erwanode
                service_description             PING
                is_volatile                     0                       ; Service non volatil
                check_period                    24x7
                max_check_attempts              3                       ; 3 tentatives avant de passer en HARD
                normal_check_interval           1                       ; 1 minutes entre chaque contrôle
                retry_check_interval            1                       ; Si état non-OK nouveau contrôle dans 1 minutes
                contact_groups                  nagios-admins           ; Groupe à notifier
                notification_interval   60                              ; Nouvelle notification dans 60 minutes
                notification_period     24x7                            ; Période durant laquelle on peut être notifié
                notification_options    w,c,r                           ; Etat pour lesquels une notification es envoyee
                check_command                   check_ping!350.0,40%!600.0,60%
        }
```

- checkcommands.cfg : fichier qui permet de spécifier les commandes de contrôles pour la supervision :

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

- misccommands.cfg : fichier qui permet de spécifier les commandes de notification :

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

- timeperiods.cfg : fichier permettant de spécifier les calendriers et les tranches horaires :

```
# '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
    }
```

- contacts.cfg : fichier contenant les définitions des contacts à notifier :

```
# 'nagios' contact definition
    define contact{
        contact_name                    erwanadmin
        alias                           Erwan Nagios
        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,notify-by-epager
        host_notification_commands      host-notify-by-email,host-notify-by-epager
        email                           erwan@weborama.fr
        pager                           pagenagios-admin@localhost.localdomain
    }
```

- contactgroups.cfg : fichier contenant les groupes de contacts à notifier :

```
# 'nagios-admins' contact group definition
    define contactgroup{
        contactgroup_name       nagios-admins
        alias                   Nagios Administrators
        members                 erwanadmin
    }
```

- dependencies.cfg : fichier permettant de spécifier les dépendances entre services :

```
define servicedependency{
    dependent_host_name                 nagiosnode
    dependent_service_description       HTTP
    host_name                           nagiosnode
    service_description                 PING
    execution_failure_criteria          w,c
    notification_failure_criteria       w,c
}
```

- cgi.cfg : définit la configuration des CGI de Nagios utilisés dans l'interface web.

- Une fois cette configuration réalisée, le commande "/usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg" permet de vérifier la bonne déclaration et syntaxes des fichiers de configuration.
- La commande "/usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg" permet de lancer Nagios.

2/ Installation de Nrpe

- Récupérer l'archive nrpe-2.0.tar.gz : http://www.nagiosexchange.org/NRPE.77.0.html?&tx_netnagext_pi1[p_view]=126
- Sur l'hôte Nagios : et les hôtes que l'on souhaite superviser :
  - Décompresser l'archive : tar xvvzf nrpe-2.0.tar.gz
  - Les binaires seront dans le src/
  - Les binaires seront dans le src/ .Dans le répertoire correspondant taper les commandes suivantes :

```
./configure
make all
```

  - Copier le check_nrpe dans le répertoire des plug-ins par defaut : /usr/local/nagios/libexec/
- Sur les hôtes à superviser :
  - Décompresser l'archive : tar xvvzf nrpe-2.0.tar.gz
  - Créer un utilisateur nagios (cf début de la procédure)
  - Créer les répertoires (pour un soucis d'organisation) /usr/local/nagios/libexec , /usr/local/nagios/etc , /usr/local/nagios/bin
  - Installer la suite de plug-ins en suivant la procédure décrite plus haut
  - Dans le répertoire ou l'archive nrpe-2.0.tar.gz a été décompressée :

```
./configure
make all
```

  - Copier src/nrpe dans /usr/local/nagios/bin et le fichier nrpe.cfg dans /usr/local/nagios/etc
  - Editer le fichier nrpe.cfg :

```
server_port=5666                    # port d'écoute
allowed_hosts=192.168.0.60          # liste des hôtes autorisés à lancer des commandes
nrpe_user=nagios                    # sous quel utilisateur
nrpe_group=nagios                   #
dont_blame_nrpe=1                   # autorisation de mettre des arguments
debug=0
command_timeout=60                  # timeout si aucune réponse au bout d'une minute
# Définition des services à controler
```

```
command[check_diskLOCAL]=/usr/local/nagios/libexec/check_disk -w 10% -c 5%
# si l'hôte à superviser recoit une requête NRPE contenant le nom de commande "check_diskLOCAL" alors l'hôte executera
la commande "/usr/local/nagios/libexec/check_disk -w 10% -c 5%"
...
```

- Pour lancer le daemon NRPE : /usr/local/nagios/bin/nrpe -c /usr/local/nagios/etc/nrpe.cfg -d
  L'option -c permet de spécifier le fichier de configuration pour NRPE.
  L'option -d permet de préciser que NRPE se lance en tant que daemon.

3/ Installation de Nagiosgraph

Il faut dans un premier temps installer RRDTool qui est un prérequis pour Nagiosgraph

3.1/ Installation de RRDTool

- Récupérer l'archive rrdtool-1.0.49.tar.gz : http://www.fastmirrors.org/rrdtool/
- Décompresser l'archive : tar xvvzf rrdtool-1.0.49.tar.gz
- Dans le répertoire associé à l'archive :
  sh configure
  make
  make install

RRDTool sera installé par défaut dans le répertoire suivant : /usr/local/rrdtool-1.0.49/

3.2/ Installation de Nagiosgraph

a/Installation

- Récupérer l'archive nagiosgraph-0.3.tar.gz : http://sourceforge.net/project/showfiles.php?group_id=115472
  et CGI.pm-3.08.tar.gz (si cette bibliothèque n'est pas installé) : http://search.cpan.org/src/LDS/CGI.pm-3.08/cgi_docs.html#download
- Décompresser l'archive : tar xvvzf nomdelarchive
- Dans le répertoire associé à l'archive CGI.pm-3.08.tar.gz, executer les commandes suivantes :
  perl Makefile.PL
  make
  make test
  make install
- Créer un répertoire dans le répertoire d'installation de Nagios pour plus de lisibilité
  mkdir /usr/local/nagios/nagiosgraph
- Copier les fichiers se trouvant dans le répertoire associé à l'archive nagiosgraph-0.3.tar.gz dans le répertoire /usr/local/nagios/nagiosgraph

b/Configuration

- Editer le fichier nagiosgraph.conf en indiquant les divers chemins : exemple de configuration :

```
# Le fichier de log lié à nagiosgraph
logfile = /usr/local/nagios/var/nagiosgraph.log
# Le répertoire ou les base de données rrd seront sauvegrdées, créer ce repertoire si il n existe pas
rrddir =  /usr/local/nagios/nagiosgraph/rrd
# Chemin pour acceder au fichier map
mapfile = /usr/local/nagios/nagiosgraph/map
# Chemin ou se trouve l'executable rrdtool
rrdtool = /usr/local/rrdtool-1.0.48/bin/rrdtool
```

Eventuellement modifier la valeur de debug pour les tests.

- Vérifier les droits sur les fichiers :
  l'user nagios doit pouvoir écrire dans le répertoire des bases rrd (cf et l'user www doit pouvoir lire,
  les droits sur le répertoire /usr/local/nagios/nagiosgraph/rrd doivent être les suivants : drwxr-xr-x
  l'user nagios et les users www doivent pouvoir écrire dans le fichier log,
  les droits sur le fichier /usr/local/nagios/var/nagiosgraph.log doivent être les suivants : rw-rw-r--
- Editer le chemin "my $configfile" des fichiers show.cgi et insert.pl en précisant le chemin vers le fichier nagiosgraph.conf :
  my $configfile = '/usr/local/nagios/nagiosgraph/nagiosgraph.conf';

Il ne faut rien modifier d'autre dans ces fichiers.

- Dans le fichier nagios.cfg :
  * vérifier que l'option process_perf_data est bien activée, sinon activer la de la facon suivante :
    process_performance_data=1
  * éditer les service_perfdata_command ainsi :
    service_perfdata_command=process-service-perfdata

Cela signifie que pour chaque collecte de données (d'un contrôle de service) la commande process-service-perfdata sera executée

- Spécifier dans le checkcommands.cfg la commande process-service-perfdata de la manière suivante :

```
define command {
        command_name  process-service-perfdata
        command_line  /usr/local/nagios/nagiosgraph/insert.pl "$LASTCHECK$||$HOSTNAME$||$SERVICEDESC$||$OUTPUT$||$PERFDATA$"
        }
```

A ce stade là si on lance Nagios, aucun graph ne sera généré mais les bases rrd seront créées (suivant ce qui aura été défini dans le fichier map).

- Rajouter dans le fichier cgi.cfg la ligne suivante :
  xedtemplate_config_file=/usr/local/etc/nagios/extinfo.cfg

Ce fichier va contenir en gros des infos suplémentaires sur les services : icône associée, lien vers graph(!!!!)...

- Créer le fichier /usr/local/etc/nagios/extinfo.cfg et l'éditer de la manière suivante :

```
define serviceextinfo {
        service_description  DNS
        host_name       erwanode,dnssecondaire
        notes_url       /nagiosgraph/show.cgi?host=$HOSTNAME$&service=$SERVICEDESC$
        icon_image      /images/icon/nagios.gif
        icon_image_alt  View graphs
}
```

- Editer le fichier de configuration du serveur Web (httpd.conf) :

```
ScriptAlias /nagiosgraph/ /usr/local/nagios/nagiosgraph/
        <Directory "/usr/local/nagios/nagiosgraph">
                AllowOverride AuthConfig
                Options ExecCGI
                Order allow,deny
                Allow from all
        </Directory>
```

Rajouter le htaccess /usr/local/nagios/nagiosgraph/ si nécessaire ! (suivant la configuration Apache établie).

4/ Installation de check_traffic

Prérequis :

les outils SNMP "snmpget" et "snmpwalk" doivent être présents sur les équipements que l'on souhaite monitorer et la machine Nagios. Sinon il sera nécessaire d'installer NET-SNMP v5.2.1 qui contient les outils de polling SNMP.

Installation :

- Récupérer l'archive check_traffic-0.90b.tar.gz
- Décompresser l'archive : tar xvvzf check_traffic-0.90b.tar.gz
- Créer le répertoire /usr/local/nagios/traffic/
- Copier le fichier check_traffic dans /usr/local/nagios/libexec/
- Editer le fichier check_traffic, se trouvant dans le répertoire associé à l'archive, de la manière suivante :

```
$TRAFFIC_FILE = "/usr/local/nagios/traffic/";
$SNMPWALK     = "/usr/local/bin/snmpwalk";
$SNMPGET      = "/usr/local/bin/snmpget";
$COMMUNITY    = "public";
# RRD support:
$WITH_RRD     = 0;
$RRDTOOL      = "/usr/bin/rrdtool";
$DB_PATH      = "/usr/local/nagios/check_traffic-$VERSION/db";
```

Annexe 2 : mysql-replication.howto

Howto : Replication de serveur MySQL
by erwan@weborama.fr

Ce howto a été réalisé après une suite de tests faits sur des serveurs MySql v4.0.24

- Créer un compte utilisateur pour la réplication

Pour que la réplication soit possible il faut créer au niveau du master un utilisateur avec des droits spécifique :

mysql> GRANT REPLICATION SLAVE ON *.* TO <user_name>@'%' IDENTIFIED BY '<password>';

<user_name> : nom de l'utilisateur utilisé pour la réplication
<password> : mot de passe associé
A noter qu'il peut être bon de créer l'user sur la machine esclave au cas ou l'esclave passerait maître par exemple...

- Bloquer l'accès en écriture des tables du côté MASTER

Jusqu'à ce que le serveur esclave soit mis en place il ne faut pas que les bases copiées subissent des modifications, cela risquerait de créer des incohérences dans les mises à jour de l'esclave.
Pour bloquer les tables il suffit de taper la commande suivante

mysql> FLUSH TABLES WITH READ LOCK;

- Copier les bases

Il suffit de se placer dans le répertoire des bases :

shell> tar -cvf ./mysql-snapshot.tar ./cette_base

La commande suivante copie seulement "cette_base" pour toutes les copier il suffit de taper la ligne suivante :

shell> tar -cvf ./mysql-snapshot.tar

- Modification du my.cnf côté MASTER

Il est important de spécifier certains paramètres dans le fichier /etc/my.cnf, il suffit de rajouter dans la section [mysql] les lignes suivantes (si elles n'y étaient pas):

log-bin
server_id = <number>

<number> : est une valeur comprise entre 1 et 2^32 - 1
log-bin permet de spécifier la génération de fichiers binaires contenant une sorte d'historique des modifications des bases.
SHOW MASTER STATUS; permet de consulter quel est le dernier log-file modifié et quel est l'offset de la dernière commande.

- Reset le MASTER

Dans le cas où log-bin été déjà activé il faut vider les fichiers binaires pour ne pas créer d'inchohérences (plus loin il y aura une autre solution si vous ne souhaitez pas faire cette manipulation).

RESET MASTER;

- Débloquer l'accès en écriture des tables du côté MASTER

L'accès aux bases en écriture peut être rétabli vu qu'auparavantle snapshot des bases le RESET ont été faits. Le but pour le SLAVE sera donc de répliquer toutes les nouvelles commandes qui modifieront les bases.

mysql> UNLOCK TABLES;

- Copier les bases sur le SLAVE

En se placant dans le répertoire des bases de données du SLAVE taper la commande suivante :

shell> tar -xvf ./mysql-snapshot.tar

A ce stade le slave possède les mêmes bases que le MASTER moins les dernières modifications qui sont décrites dans les fichiers de logs binaires. C'est ces modifications que le SLAVE devra rattraper pour avoir les bases équivalentes au MASTER.

- Modification du my.cnf côté SLAVE

Plusieurs champs devront être spécifiés pour que même si le serveur SLAVE plante a son redémarrage il puisse reprendre un fonctionnement normal sans aucune commandes à taper.
Les modifications sont à faire dans la section [mysql] :

server_id = <number>
master-host = <master_name>
master-user = <user_name>
master-password = <password>
master-port = <port>

<number> : est une valeur comprise entre 1 et 2^32 - 1 différent de l'id d'un autre serveur.
<master_name> : adresse du serveur maître
<user_name> : utilisateur créé plus tot avec son mot de passe associé <password>
<port> : port utilisé par MySQL
A noter que si l'on veut mettre en place un second SLAVE à partir du meme snaphot des bases, il suffit de le configurer comme ci dessu, pas besoin de modifier autre chose : le nouvel esclave reprendra toutes les modifications qui auront été recensées dans les fichiers de logs binaires (ce qui peut parfois être long).

- Lancer les threads SLAVE

mysql> RESTART;

Annexe 3 : backupautoV23.cmd

```
@echo off

REM ------------------------------------------------------------------------
REM              Script de Backup Automatisee avec Ntbackup
REM ------------------------------------------------------------------------

Title Script de Backup Automatisee avec Ntbackup

REM ---------------------------------------
REM Boucle de Test des connections reseaux
REM ---------------------------------------

if not defined coupredecourant set coupuredecourant=0

:boucle
REM Attente de 3 ping envoye toutees secodnes soit une attente de 3sec
ping 127.0.0.1 -n 3 -w 1000 > nul
REM Test si le fichier existe
if exist \\Webodev\public\BACKUPS-MACHINES\Ntbackup\fichier.test goto main
REM si au bout de 5 test, le réseau n'est toujours pas disponible le programme se finit
if %coupuredecourant% EQU 5 goto exit
set /a coupuredecourant=coupuredecourant+1
echo Patientez le temps que les connections reseaux soient etablies
goto boucle

:main

REM -------------------------------
REM Declaration des variables
REM -------------------------------

REM Nom de l'utilisateur
set mon_nom=Erwan
REM Nom du répertoire de destination pour sauvegarder l'archive
set repertoire_svg=\\Webodev\public\BACKUPS-MACHINES\Ntbackup\%mon_nom%\
REM Creation du repertoire si ce dernier existe cette commande ne fera rien
mkdir \\Webodev\public\BACKUPS-MACHINES\Ntbackup\%mon_nom%\
REM Formatage de la date pour le nom du fichier jj_mm_aaaa
for /F "tokens=1,2,3,4 delims=/, " %%i in ('date /T';) do set ma_date=%%i_%%j_%%k
REM Nom de l'archive
set nom_archive=svg.bkf
REM Nom complet de l'archive c'est le nom que l'archive portera il est forme du nom-date de svg-archive
set nom_reel_archive=1-%ma_date%-%mon_nom%-%nom_archive%

REM Nom du fichier de listing de sauvegarde
set nom_listing=C:\liste_svg.bks

REM Procedure pour compter le nombre d'archives deja cree
dir /B /OD %repertoire_svg%*-%mon_nom%-%nom_archive% | find  /C ".bkf" > %repertoire_svg%temp
for /F "tokens=1 delims=" %%i in (%repertoire_svg%temp) do set count=%%i

REM Procedure pour verifier qu'une sauvegarde a deja ete faite dans la journee
dir /OD %repertoire_svg%*-%ma_date%-%mon_nom%-%nom_archive% | find /C ".bkf"> %repertoire_svg%temp2
for /F %%B in (%repertoire_svg%temp2) do set deja=%%B

REM Nettoyage de la fenetre d'execution
cls
REM Test si une sauvegarde a deja ete faite aujourd'hui
if "%deja%" == "0" GOTO Sauvegarde

REM Procedure executee si une sauvegarde a deja ete realisee aujourd'hui
:Abrege
echo ********************************************
echo       Backup Automatisee avec Ntbackup
echo ********************************************
echo.
echo Une sauvegarde a deja ete faite aujourd'hui
echo.
echo Purge des fichiers temporaires
del %repertoire_svg%temp
```

```
del %repertoire_svg%temp2
echo.
echo Vous pouvez reprendre une activite normale

goto exit

REM Procedure executee si aucune sauvegarde n a ete effectuee aujourd'hui
:Sauvegarde
echo ********************************************
echo       Backup Automatisee avec Ntbackup
echo ********************************************

echo.
echo L'archive sera enregistree a l'adresse suivante: %repertoire_svg%%nom_reel_archive%
echo.
echo Lancement de la routine de sauvegarde a :
TIME /T
echo.
REM Test conditionnel les procedures changent suivant le nombre d'archives deja creees
if "%count%" == "1" GOTO Procedure1
if "%count%" == "2" GOTO Procedure2
if "%count%" == "3" GOTO Procedure3

REM -------------------------------------
REM Procedure si aucune archive presente
REM -------------------------------------
echo Lancement de la sauvegarde sous Ntbackup
C:\WINDOWS\system32\ntbackup.exe backup "@%nom_listing%" /a /d "Sauvegarde de %mon_nom% " /v:no /r:no /rs:no /hc:off /m normal /j "%
nom_listing% %nom_archive%" /l:s /Snap:Off /f "%repertoire_svg%%nom_reel_archive%"
echo.
echo Purge des fichiers temporaires
del %repertoire_svg%temp
del %repertoire_svg%temp2
echo.
echo Fin de la routine de sauvegarde a :
TIME /T
echo.
echo Vous pouvez reprendre une activite normale
goto exit

REM -----------------------------------------
REM Procedure si 3 archives sont presentes
REM -----------------------------------------
:Procedure3
echo Rotation des anciennes archives...
del %repertoire_svg%3-*
rename %repertoire_svg%2-* 3-*
rename %repertoire_svg%1-* 2-*
echo Fin de la rotation des anciennes archives
echo.
echo Lancement de la sauvegarde sous Ntbackup
C:\WINDOWS\system32\ntbackup.exe backup "@%nom_listing%" /a /d "Sauvegarde de %mon_nom% " /v:no /r:no /rs:no /hc:off /m normal /j "%
nom_listing% %nom_archive%" /l:s /Snap:Off /f "%repertoire_svg%%nom_reel_archive%"
echo.
echo Purge des fichiers temporaires
del %repertoire_svg%temp
del %repertoire_svg%temp2
echo.
echo Fin de la routine de sauvegarde a :
TIME /T
echo.
echo Vous pouvez reprendre une activite normale
goto exit

REM -----------------------------------------
REM Procedure si 2 archives sont presentes
REM -----------------------------------------
:Procedure2
echo Rotation des anciennes archives...
rename %repertoire_svg%2-* 3-*
rename %repertoire_svg%1-* 2-*
echo Fin de la rotation des anciennes archives
```

```
echo.
echo Lancement de la sauvegarde sous Ntbackup
C:\WINDOWS\system32\ntbackup.exe backup "@%nom_listing%" /a /d "Sauvegarde de %mon_nom% " /v:no /r:no /rs:no /hc:off /m normal /j "%
nom_listing% %nom_archive%" /l:s /Snap:Off /f "%repertoire_svg%%nom_reel_archive%"
echo.
echo Purge des fichiers temporaires
del %repertoire_svg%temp
del %repertoire_svg%temp2
echo.
echo Fin de la routine de sauvegarde a :
TIME /T
echo.
echo Vous pouvez reprendre une activite normale
goto exit

REM -----------------------------------------
REM Procedure si 1 seule archive est presente
REM -----------------------------------------
:Procedure1
echo Rotation des anciennes archives...
rename %repertoire_svg%1-* 2-*
echo Fin de la rotation des anciennes archives
echo.
echo Lancement de la sauvegarde sous Ntbackup
C:\WINDOWS\system32\ntbackup.exe backup "@%nom_listing%" /a /d "Sauvegarde de %mon_nom% " /v:no /r:no /rs:no /hc:off /m normal /j "%
nom_listing% %nom_archive%" /l:s /Snap:Off /f "%repertoire_svg%%nom_reel_archive%"
echo.
echo Purge des fichiers temporaires
del %repertoire_svg%temp
del %repertoire_svg%temp2
echo.
echo Fin de la routine de sauvegarde a :
TIME /T
echo.
echo Vous pouvez reprendre une activite normale
goto exit

:exit
```

Annexe 4 : init-script.in : script de démarrage de NRPE

```
#!/bin/sh
# description: nrpe is a daemon for a remote nagios server, \
#              running nagios plugins on this host.
# processname: nrpe
# config: /usr/local/nagios/etc/nrpe.cfg

# Source function library
if [ -f /etc/rc.d/init.d/functions ]; then
. /etc/rc.d/init.d/functions
elif [ -f /etc/init.d/functions ]; then
. /etc/init.d/functions
elif [ -f /etc/rc.d/functions ]; then
. /etc/rc.d/functions
fi

# Source networking configuration.
. /etc/sysconfig/network

# Check that networking is up.
[ ${NETWORKING} = "no" ] && exit 0

NrpeBin=/usr/local/nrpe/bin/nrpe
NrpeCfg=/usr/local/nrpe/etc/nrpe.cfg
LockFile=/var/lock/subsys/nrpe

prog="nrpe"
server1=$(cat $NrpeCfg | grep allowed_hosts | cut -d= -f 2 | cut -d, -f1)
server2=$(cat $NrpeCfg | grep allowed_hosts | cut -d= -f 2 | cut -d, -f2)

# Is there a firewall running, and does it look like one we configured?
FWACTIVE=''
if [ -f /proc/net/ip_tables_names ]; then
        if iptables -L -n 2>/dev/null | grep -q RH-Firewall-1-INPUT ; then
                FWACTIVE=1
        fi
fi

# See how we were called.
case "$1" in
  start)
        # Open the firewall for nrpe
        if [ -n "$FWACTIVE" -a "$FIREWALL_MODS" != "no" ]; then
           echo -n $"$prog: Opening firewall for input from $server1 port 5666"
           iptables -I RH-Firewall-1-INPUT -m tcp -p tcp -s $server1/32 --sport 5666 -d 0/0 --dport 5666 -j ACCEPT \
                && success || failure
           echo
        fi

        if [ -n "$FWACTIVE" -a "$FIREWALL_MODS" != "no" ]; then
           echo -n $"$prog: Opening firewall for input from $server2 port 5666"
           iptables -I RH-Firewall-1-INPUT -m tcp -p tcp -s $server2/32 --sport 5666 -d 0/0 --dport 5666 -j ACCEPT \
                && success || failure
           echo
        fi

        # Start daemons.
        echo -n "Starting nrpe: "
        daemon $NrpeBin -c $NrpeCfg -d
        echo
        touch $LockFile
        ;;
  stop)
        # Stop daemons.
        echo -n "Shutting down nrpe: "
        killproc nrpe
        echo
        rm -f $LockFile

        # Remove the firewall opening for nrpe
        if [ -n "$FWACTIVE" -a "$FIREWALL_MODS" != "no" ]; then
```

```
                echo -n $"$prog: Removing firewall for input from $server1 port 5666"
                iptables -D RH-Firewall-1-INPUT -m tcp -p tcp -s $server1/32 --sport 5666 -d 0/0 --dport 5666 -j ACCEPT \
                        && success || failure
                echo
        fi
        if [ -n "$FWACTIVE" -a "$FIREWALL_MODS" != "no" ]; then
                echo -n $"$prog: Removing firewall for input from $server2 port 5666"
                iptables -D RH-Firewall-1-INPUT -m tcp -p tcp -s $server2/32 --sport 5666 -d 0/0 --dport 5666 -j ACCEPT \
                        && success || failure
                echo
        fi
        ;;
 restart)
        $0 stop
        $0 start
        ;;
 status)
        status nrpe
        ;;
 *)
        echo "Usage: nrpe {start|stop|restart|status}"
        exit 1
esac

exit 0
```

Annexe 5 : check_cpuauto.pl

```
#!/usr/bin/perl

use strict;

# Declaration des variables
my $nombre_proc;
my $ls2;

my $sur1min=0;
my $sur5min=0;
my $sur15min=0;

my %STATUS_CODE = (  'UNKNOWN'  => '-1',
                     'OK'       => '0',
                                   'WARNING'  => '1',
                                   'CRITICAL' => '2'   );

#Recuperation du nombre de processeur
$nombre_proc= `cat /proc/cpuinfo | grep processor | wc -l`;

#Recuperation du nombre de processus sur 1 5 et 15 minutes
$ls2=`cat /proc/loadavg`;
my ($sur1min,$sur5min,$sur15min,$junk) = split /\s/,$ls2;

#On suppose une charge maximale egale a 2*nombre_proc de processus sur 1 minute
#4*nombre_proc de processus sur 5 minutes
#8*nombre_proc de processus sur 15 minutes
if (($sur1min>(8*$nombre_proc)) or ($sur5min>(6*$nombre_proc)) or ($sur15min>(4*$nombre_proc))){
  print "LOAD - CRITICAL - $sur1min,$sur5min,$sur15min\n";
  exit($STATUS_CODE{"CRITICAL"});
}

print "LOAD - OK - $sur1min,$sur5min,$sur15min\n";
exit($STATUS_CODE{"OK"});
```

Annexe 6 : check_diskauto3.pl

```
#!/usr/bin/perl

use strict;
# Declaration des variables
my @ls;
my $ligne;
my @nomdisk;
my @taille;
my @utilise;
my @nommontage;
my $critic=0;
my $warn=0;
my $retour;
my $retourcritique;
my $retourwarning;

#Declaration des etats possible a retourner
my %STATUS_CODE = (  'UNKNOWN'  => '-1',
                     'OK'       => '0',
                                   'WARNING'  => '1',
                                   'CRITICAL' => '2'   );
@ls= `df`;
#On ecrit chaque ligne du resultat dans le fichier
foreach  $ligne (@ls) {
  if ($ligne=~/^(\S+)\s+(\d+)\s+\d+\s+\d+\s+(\d+)+%+\s+(\S+)/){
    #print "$1 $2 $3 montage : $4\n";
    push(@nomdisk,$1);
    push(@taille,$2);
    push(@utilise,$3);
    push(@nommontage,$4);
  }
}
#On distingue plusieurs cas :
#Si le disk a une capacite inferieure a 18Go : warning a partir de 85% d utilisation et critique au dela de 90%
# et les autres :  warning a partir de 85% d utilisation et critique au dela de 95
for (my $i=0; $i<=$#nomdisk; $i++){
  if ($taille[$i]<18000000){
    if ($utilise[$i]>=90){
      $critic++;
      $retourcritique.="$nomdisk[$i](monté sur $nommontage[$i]) utilise $utilise[$i]% -";
    }
    if ((85<$utilise[$i]) and ($utilise[$i]<90)){
      $warn++;
      $retourwarning.="$nomdisk[$i](monté sur $nommontage[$i]) utilise $utilise[$i]% -";
    }
  }
  else{
    if ($utilise[$i]>=95){
      $critic++;
      $retourcritique.="$nomdisk[$i](monté sur $nommontage[$i]) utilise $utilise[$i]% -";
    }
    if ((85<$utilise[$i]) and ($utilise[$i]<95)){
      $warn++;
      $retourwarning.="$nomdisk[$i](monté sur $nommontage[$i]) utilise $utilise[$i]% -";
    }
  }
  #Variable pour l output dans Nagios
  $retour.="$nomdisk[$i](monté sur $nommontage[$i]) utilise $utilise[$i]% - ";
}
if ($critic!=0){
  print "DISK - CRITICAL - il y a $critic warning - $retourcritique\n";
  exit($STATUS_CODE{"CRITICAL"});
}
if ($warn!=0){
  print "DISK - WARNING - il y a $warn warning - $retourwarning\n";
  exit($STATUS_CODE{"WARNING"});
}
print "DISK - OK - $retour\n";
exit($STATUS_CODE{"OK"});
```

Annexe 7 : check_mysqltest.pl

```
#!/usr/bin/perl

use DBI;
use strict;
my $configfile = "/usr/local/nrpe/etc/mysql.cfg";
my %servers;
my $retour;
#Declaration des etats possible a retourner
my %STATUS_CODE = (  'UNKNOWN'  => '-1',
                     'OK'       => '0',
                     'WARNING'  => '1',
                     'CRITICAL' => '2'   );
if (-f $configfile) {
        open FIC, $configfile;
        while (my $line=<FIC>) {
                chomp $line;
                #print "$line\n";
                my ($socket,$uid,$pwd) = split /:/,$line;
                #print "$socket, $uid, $pwd \n";
                $servers{$socket}->{'uid'}=$uid;
                $servers{$socket}->{'pwd'}=$pwd;
        }
        close FIC;
}
my $exitcode=0;
foreach my $socket (keys %servers) {
        my $rc=0;
        eval {
                $rc=test($socket,$servers{$socket}->{'uid'},$servers{$socket}->{'pwd'});
        };
        if (!$rc || $@) {
                $retour.="$socket $rc, ";
                $exitcode=2;
        } elsif ($rc) {
                # rien
        } else {
                $retour.="$socket $rc, ";
                $exitcode=-1 if ( $exitcode!=0 );
        }
}
if ($exitcode == 0){
        print "MySQL - OK - tous les serveurs tournent \n";
        exit($STATUS_CODE{"OK"});
}elsif ($exitcode == 2) {
        print "MySQL - CRITICAL - les sockets suivantes ont des soucis : $retour \n";
        exit($STATUS_CODE{"CRITICAL"});
}else {
        print "MySQL - UNKNOWN - les sockets suivantes ont des soucis : $retour \n";
        exit($STATUS_CODE{"UNKNOWN"});
}


sub test {
        my ($socket,$uid,$pwd) = @_;
        my $dbh;
        $dbh=DBI->connect("DBI:mysql:mysql;mysql_socket=$socket;".
                "loginTimeout=15;scriptName=nagios",
                $uid, $pwd, {PrintError=>0}) || die "Can't connect $DBI::errstr";
        my @row_ary = $dbh->selectrow_array("select 1")
                || die "Can't select";
        if ($row_ary[0] == 1) {
                return(1);
        } else {
                die "Select test did not return 1";
        }
}
```

Annexe 8 : check_mysqldatasize.pl

```
#!/usr/bin/perl

use DBI;
use strict;
my $configfile = "/usr/local/nrpe/etc/mysql.cfg";
my %servers;
my $retour;
my $ligne_retour;
my $compteur_critic = 0;

#Declaration des etats possible a retourner
my %STATUS_CODE = (  'UNKNOWN'  => '-1',
                     'OK'       => '0',
                     'WARNING'  => '1',
                     'CRITICAL' => '2'   );

if (-f $configfile) {
  open FIC, $configfile;
  while (my $line=<FIC>) {
    chomp $line;
    my ($socket,$uid,$pwd) = split /:/,$line;
    #print "$socket, $uid, $pwd \n";
    $servers{$socket}->{'uid'}=$uid;
    $servers{$socket}->{'pwd'}=$pwd;
  }
  close FIC;
}

#Pour chaque socket recupere dans /usr/local/nrpe/etc/mysql.cfg
foreach my $socket (keys %servers) {
  my $rc=0;
  #Appel de la fonction test
  test($socket,$servers{$socket}->{'uid'},$servers{$socket}->{'pwd'});
}

if ($compteur_critic != 0){
  print "Taille des BDs MySQL - CRITICAL - problèmes avec <br>\n $ligne_retour \n";
  exit($STATUS_CODE{"CRITICAL"});
}
else {
  print "Taille des BDs MySQL - OK - tout est dans les normes \n";
  exit($STATUS_CODE{"OK"});
}

sub test {
       my ($socket,$uid,$pwd) = @_;
       my $dbh;
              my $ligne;

              #Connexion sur la socket en parametres
              $dbh=DBI->connect("DBI:mysql:mysql;mysql_socket=$socket;".
             "loginTimeout=15;scriptName=nagios",
             $uid, $pwd, {PrintError=>0}) || die "Can't connect $DBI::errstr";

              #Execution de show databases pour recuperer toutes les bases de cette socket
              my $sth = $dbh->prepare(qq{show databases;})
             || die "Show database a échoué";
              $sth->execute();
              my (@matrix) = ();

              #Pour chaque base
              while (my @ary = $sth->fetchrow_array())
                {
                 #print "@ary \n";
                 push(@matrix, @ary);
                }
              $sth->finish();

              my $base_acheck;
```

```
        foreach $base_acheck (@matrix){

          #Execution de show table pour recuperer toutes les tables de cette base
          my $sth2 = $dbh->prepare(qq{show table status from $base_acheck;})
            || die "Show table status a échoué";
          $sth2->execute();

          my $taille_actu;
          my $index_actu;
          my $nomtable;
          my $i=0;

          #Pour chaque table
          while (my @ary2 = $sth2->fetchrow_array())
            {
              #Recupere des elements utils pour le test : nom, taille actuelle et taille de l'index
              $nomtable = @ary2[0];
              $taille_actu = @ary2[5];
              $index_actu = @ary2[7];

              #print "base : $base_acheck , table : $nomtable , taille actuelle : $taille_actu ,index actuel : $index_actu \n";

              #Test des tailles
              if ($taille_actu >= 2000000000)      {
                      #Hors norme il faut donc le signaler
                      $compteur_critic = $compteur_critic + 1;
                      $ligne_retour.=" - base : $base_acheck , table : $nomtable (données)<br>\n";
              }
              if ($index_actu >= 2000000000)      {
                      #Hors norme il faut donc le signaler
                      $compteur_critic = $compteur_critic + 1;
                      $ligne_retour.=" - base : $base_acheck , table : $nomtable (index)<br>\n";
              }

            }

          $sth2->finish();
        }
        return(1);
}
```

Annexe 9 : map

```
#  Weborama

#########################################################################
#
# /output|perfdata:<servicetype> <key>=<value> <key2=value2> .../
# and push @s, [ <databasename>,
#               [ <key>,  GAUGE|DERIVE, <value>  ],
#               [ <key2>, GAUGE|DERIVE, <value2> ],
#               [ .       .             .        ],
#               [ .       .             .        ] ];
#
# But more advanced code is possible, as long as the resulting
# datastructure is correct.
#
#########################################################################

# Service type: ping
#   output:PING OK - Packet loss = 0%, RTA = 0.00 ms
/output:PING.*?\d+%.+?([.\d]+)\sms/
and push @s, [ ping,
               [ rta,     GAUGE, $1/1000 ] ];


# Service type: check-traffic
#   output:Total RX Bytes: 3233.41 MB, Total TX Bytes: 928.73
# MB Average Traffic: 3.04 B/s (0.0%) in, 0.87 B/s (0.0%) out
/output:Total .*?\(([.0-9]+)%\) in.*?\(([.0-9]+)%\) out/
and push @s, [ traffic,
               [ in, GAUGE, $1 ],
               [ out, GAUGE, $2 ] ];

# Service type: unix-load
#   output: LOAD - OK - 2.41,2.47,2.40
/output:LOAD.*?([.0-9]+),([.0-9]+),([.0-9]+)/
and push @s, [ load,
               [ avg1min,  GAUGE, $1 ]];


# Service type: check-diskauto
#   output:DISK - OK - /dev/sda1(monté sur /) utilise 35% - /dev/sdb3(monté sur /data) utilise 11% - none(monté sur /dev/shm) utilise 0% - /
dev/sda3(monté sur /usr/local) utilise 14% - /dev/sda1(monté sur /) utilise 35% -
/output:DISK - OK -.*?/ and do {
       my @s1;
       my @_test=/- \/.*?\/([\S+]+)\(mont.*?\).*?utilise (\d+)%/g;
       while ( my($_d,$_b) = splice @_test,0,2) {
              push @s1, [ $_d,     GAUGE,  $_b];
       }
       push @s, [ disk,@s1];
};
```