ANGD Mathrice 2009 / CIRM
TP
Nagios

1 Introduction

Pour l'ensemble des TP de cette ANGD, vous avez à votre disposition deux machines virtuelles (VM). Ces machines tournent sous un FreeBSD 7.2. Il n'y a aucune différence majeure avec Linux. Ces machines utilisent la technologie des jails, il faudra utiliser `fping` au lieu du classique `ping`. Le second impact est la disparition du localhost, on passera obligatoirement par l'adresse publique de chaque machine. Nous vous indiquerons le moment venu les différences de localisation des fichiers de configuration entre la version FreeBSD et celle d'un Linux (distribution debian).

La première machine virtuelle sera à assimiler à votre serveur nagios (celui qui supervise vos serveurs/services) et la seconde sera à assimiler à un de vos serveurs à monitorer. Vous pouvez vous connectez en ssh en tant que root sur ces deux VM.

2 Installation de nagios et fichiers

Nous ne détaillerons pas l'installation de nagios, car cela est très «distrib»-dépendant. Pour la partie serveurs, les fichiers qui nous interesse seront dans

`/usr/local/etc/nagios`[1]

Tous les fichiers de configuration de nagios se trouve dans ce dossier.

Remarque 1. Les noms de fichiers ne sont pas figés, vous pouvez les renommer. Attention toutefois, si vous voulez modifier le nom du fichier `nagios.cfg`, de ne pas oublier de le modifier dans le script de lancement

Comme indiqué précédemment, pour le bon fonctionnement de Nagios, il est nécessaire d'avoir un ensemble de plugins. Ces plugins sont installés dans le dossier

`/usr/local/libexec/nagios`[2]

Il n'y a aucune modification à y apporter. Selon les configurations proposées par les distributions, il peut être utile de connaître la localisation des plugins.

3 Configuration

3.1 Configuration de apache

Lorsque vous installez nagios, il faudra le coupler avec un serveur apache, qui est indispensable pour beaucoup d'opérations au sein de nagios. Sur le serveur que vous utilisez pour le TP, apache est déjà installé et configuré. Nous vous invitons à aller jeter un coup d'oeil à cette configuration qui se trouve dans le fichier

[1] Sous debian `/etc/nagios`
[2] Sous debian `/usr/lib/nagios/plugins`

`/usr/local/etc/apache/Include/Nagios.cfg`

L'interface web de nagios intègre des fonctionnalités relativement complexes d'authentification, nous n'aborderons pas ces points dans ce TP.

Remarque 2. Pour ce TP, le serveur est configuré pour ne demander aucune authentification. En production, il est conseillé de mettre une authentification (htpasswd, LDAP, etc...).

Vous pouvez dès maintenant lancer un navigateur web sur votre poste et ouvrir l'URL : `http ://votre_serveur/nagios`.

3.2 Configuration de Nagios

Dans Nagios, la notion importante est la notion d'objet appelé `object` dans la documentation. Elle représente tout ce que vous pouvez définir/superviser/faire etc.. à savoir

- Les services
- Les machines
- Les personnes
- Les groupes
- Les commandes
- etc....

Un objet est toujours définit sous la forme

```
define nom_de_l_objet{
        variable            valeur, valeur
        etc...
        }
```

Le nombre d'objets disponible est de 10. Mais le nombre de variables est important (selon le type d'objet).

Remarque 3. Les valeurs, lorsqu'elles sont multiples, sont séparées par des «,».

Il y a une notion d'héritage dans Nagios. Il est possible de définir des `template` et ensuite de créer des `object` par héritage. Cela permet de factoriser les définitions. Nous n'aborderons pas dans les détails l'héritage. Sachez qu'il est possible de faire des choses un peu complexes, mais qu'elles ne sont pas indispensables au bon fonctionnement pour commencer.

3.3 Le fichier `nagios.cfg`

C'est **le** fichier de configuration de Nagios. Pour débuter il n'y a pas de modification importante à faire dans ce fichier. Actuellement vous pouvez voir

```
cfg_file=/usr/local/etc/nagios/objects/commands.cfg
cfg_file=/usr/local/etc/nagios/objects/contacts.cfg
cfg_file=/usr/local/etc/nagios/objects/timeperiods.cfg
cfg_file=/usr/local/etc/nagios/objects/templates.cfg
cfg_file=/usr/local/etc/nagios/objects/localhost.cfg
```

Il est cependant pratique d'avoir ses propres fichiers de configuration.

Exercice 1. *Rajouter la définition de cinq fichiers* `hosts-angd.cfg`*,* `commands-angd.cfg`*,* `services-angd.cfg`*,* `templates-angd.cfg` *et* `contacts-angd.cfg` *dans* `nagios.cfg`*. Redémarrer le serveur nagios avec la commande*

`/usr/local/etc/rc.d/nagios restart`.

Que constatez vous ? Corrigez les problèmes.

3.4 Les templates

Comme indiquez précédemment la définition d'un objet nécessite celle de beaucoup de variables[3]. Par exemple voici la liste des variables pour un objet `contact` :

```
define contact{
        contact_name                    contact_name
        alias                           alias
        contactgroups                   contactgroup_names
        host_notifications_enabled      [0/1]
        service_notifications_enabled   [0/1]
        host_notification_period        timeperiod_name
        service_notification_period     timeperiod_name
        host_notification_options       [d,u,r,f,s,n]
        service_notification_options    [w,u,c,r,f,s,n]
        host_notification_commands      command_name
        service_notification_commands   command_name
        email                           email_address
        pager                           pager_number
        addressx                        additional_contact_address
        can_submit_commands             [0/1]
        retain_status_information       [0/1]
        retain_nonstatus_information    [0/1]
        }
```

dont une partie importante est obligatoire. Pour simplifier les fichiers/configurations, nous utiliserons les *templates*. Ils permettent de définir un ensemble de valeurs communes. Par exemple dans le fichier

```
/usr/local/etc/nagios/objects/templates.cfg
```

vous pouvez voir la définition d'un template pour un contact «générique»

```
define contact{
        name                            generic-contact         ;
        service_notification_period     24x7                    ;
        host_notification_period        24x7                    ;
        service_notification_options    w,u,c,r,f,s             ;
        host_notification_options       d,u,r,f,s               ;
        service_notification_commands   notify-service-by-email ;
        host_notification_commands      notify-host-by-email    ;
        register                        0                       ;
        }
```

En utilisant ce template de contact générique, il est possible de faire des contacts en héritant des propriétés définies ci-dessus.

Exercice 2. *En vous inspirant du fichier déjà en place, définissez un/deux contacts dans le fichier* `contacts-angd.cfg` *(vous et votre binôme) qui héritent des propriétés du* `generic-contact` *ainsi qu'un groupe qui ne contient que vous et votre binôme.*

[3] voir les annexes pour une description exhaustive de la configuration des objects

3.5 Timeperiod

Pour la plupart des actions que vous allez effectuer il est possible de définir un interval de temps. Dans nagios, il s'agit de l'objet `timeperiod`. Vous pouvez par exemple définir une periode qui exclut les samedi/dimanche pour certaines alarmes. Ou bien que certains tests soit supprimés durant la nuit, etc..... Il faut pour cela définir un objet `timeperiod`. Nous vous invitons à lire en détail la définition de quelques `timeperiod` dans le fchier `/usr/local/etc/nagios/objects/timeperiod.cfg`.

4 Les hosts

4.1 Introduction

Un «host» correspond aux serveurs que nous allons surveiller.

Nous allons configurer nagios pour qu'il supervise le serveur sur lequel il tourne, soit le «localhost». Il ne nous sera cependant pas possible de faire tous les tests de manière classique (principalement les ressources réseau) car les VM que nous utilisons (les jails) impliquent la disparition du 127.0.0.1. On passera par l'interface externe.

4.2 Définition d'un host

La définition d'un host passe d'abord par celle d'un template. Cela n'est pas obligatoire mais très fortement conseillé. Il y a de nombreuses choses à renseigner pour pouvoir définir un host. Créer un template permet de ne pas répéter les informations à chaque nouveau host.

Exercice 3. *Relisez en détail la partie sur le* `generic-host` *du fichier*

`/usr/local/etc/nagios/objects/template.cfg`,

et créez dans

`/usr/local/etc/nagios/objects/templates-angd.cfg`

la définition d'un template `angd-host` *avec une directive* `contact_groups` *pour que le groupe que vous avez défini dans l'exercice 2 soit le contact pour le template* `angd-host`

Exercice 4. *En utilisant le template que vous venez de créer, rajoutez dans le fichier*

`/usr/local/etc/nagios/objects/hosts-angd.cfg`

la définition du localhost. Vous aurez à définir `host_name`, `alias`, `address`.

Vous pouvez maintenant effacer le contenu du fichier `localhost.cfg` par un `cat /dev/null > localhost.cfg`.

Remarque 4. Comme indiqué précédemment, vous devez utiliser pour l'`address` le numéro IP public de votre VM. En règle générale, il faut utiliser l'adresse du localhost, à savoir 127.0.0.1. Dans le cas d'une VM cette adresse n'existe pas.

4.3 Les plugins

Pour effectuer les tests, nous utilisons les plugins. Une certaine quantité de plugins existe déjà dans l'installation de nagios. Ils sont dans

`/usr/local/libexec/nagios/`

Remarque 5. Il vous est possible de faire vos plugins personnels. Ils peuvent être fait dans n'importe quelle langage.

Pour chaque test il y a quatre statuts principaux :
- `OK` : état normal
- `Warning` : état *dangereux*
- `Critical` : état critique.
- `Unknown` : état inconnu.

Chaque plugin est donc capable de retourner un des quatre statuts. Étudions le premier cas de `ping`

Exercice 5. *Lancer la commande*

`/usr/local/libexec/nagios/check_fping --help`

lisez avec attention le résultat. En utilisant comme cible votre adresse IP, créer une commande nécessaire pour vérifier que votre VM répond au ping avec les seuils que vous considérez comme raisonnables.

Il n'est pas possible d'utiliser directement la ligne de commande ci-dessus pour effectuer un test par nagios. Il faut créer un `command` (au sens nagios du terme) pour pouvoir être utilisé. La définition d'un `command` se fait par

```
define command{
        command_name     ...
        command_line     ...
        }
```

Exercice 6. *En utilisant le résultat de l'exercice 5 définissez un `command` qui permet de vérifier que votre VM répond au ping.*

Remarque 6. Utilisez le nom que vous voulez pour le `command`, mais n'oubliez pas qu'une configuration complète de nagios peut être constituée de centaines de `command`. Essayez donc de donner des noms cohérents.

Pour chaque host il est possible d'ajouter une règle qui vérifie sa disponibilité. Il suffit de rajouter la variable `check_command`

```
define host{
        ...              ...
        check_command    ...
        }
```

Exercice 7. *Rajoutez ce qu'il faut dans la définition du localhost que vous avez faites dans l'exercice 4. Utilisez le `command` définit dans l'exercice 6 pour que nagios puisse vérifier que le host est en ligne. Vérifiez via l'interface de nagios que cela fonctionne.*

Ce que nous venons de faire est difficilement exploitable dans la réalité, il n'est pas possible d'avoir un `command` de type ping pour chaque machine. Pour répondre à cela Nagios propose la possibilité d'utiliser une variable `$HOSTADDRESS$` qui est passé en argument à tout `command`.

Exercice 8. *En utilisant la variable `$HOSTADDRESS$` à la place du numéro IP reprenez l'exercice 5 pour rendre le `command` utilisable sur n'importe qu'elle host.*

4.4 Les services

Nous allons définir un `service` qui correspond aux tests effectués par nagios.

Exercice 9. *Lisez attentivement la définition du* `generic-service` *dans le fichier*

`/usr/local/etc/nagios/objects/templates.cfg`

et créer un template pour `angd-service` *pour qu'un test soit effectué toutes les 5 minutes.*

Voici la méthode pour définir un service qui hérite des propriétés de `generic-service`

```
define service{
        use                     generic-service
        host_name               ...
        service_description     ...
        check_command           ...
        }
```

Exercice 10. *Lisez attentivement le help de la commande*

`/usr/local/libexec/nagios/check_disk`.

Créez une `command` *puis un service qui surveille le remplissage de votre partition* `/`

Il est très souvent beaucoup plus confortable de pouvoir passer des valeurs au `command` qu'on veut effectuer. C'est typiquement le cas avec la variable `$HOSTADDRESS$`. Mais il est possible aussi lors de la création d'un service de passer d'autres valeurs au `command`. Pour cela on créer un `command` avec la syntaxe :

```
define command{
        ...
        command_line PATH_COMMAND  -w $ARG1$ -c $ARG2$
...
}
```

et de définir un service avec

```
define service{
        ...
        command NOM_COMMAND!VALEUR_DE_ARG1!VALUER_DE_ARG2
...
}
```

Exercice 11. *Reprenez l'exercice 10 en créant un* `command` *qui prend des seuils en arguments*

Exercice 12. *Lisez attentivement le help de la commande*

`/usr/local/libexec/nagios/check_load`.

Créez un `command` *puis un service qui passe les valeurs de seuil, qui surveille la charge CPU de votre serveur.*

Exercice 13. *Lisez attentivement le help de la commande*

`/usr/local/libexec/nagios/check_proc`.

Créez un `command` *puis un service qui surveille la présence d'au moins un processus* `apache`. *Faites la même chose avec* `bash` *avec une alerte si on depasse un seuil (4 ou 5 shells), puis lancer suffisamment de shells pour créer une alerte.*

Exercice 14. *Lisez attentivement les commentaires dans*

`/usr/local/libexec/nagios/check_log`.

Créez ce qu'il faut pour surveiller la non présence de la chaine `ALERTETPNAGIOS` *dans* `/var/log/syslog`. *Utilisez la commande* `echo "ALERTETPNAGIOS" >> /var/log/all.log` *pour créer une alerte.*

5 Monitoring depuis le localhost

5.1 Services de base

Dans la section précédente nous avons vu comment utiliser nagios pour surveiller des services localement. Nous allons voir dans ce paragraphe comment surveiller une machine à distance depuis votre serveur nagios.

Exercice 15. *En vous inspirant de l'exercice 4, créez un nouvel host correspondant au routeur de sortie du CIRM (Numéro IP :). Faites en sorte que nagios sache que le routeur répond au ping ou non.*

La surveillance par ping d'un host peut être considéré comme indispensable en interne (souvent en externe le ping est filtré). Il est donc intéressant de le factoriser dans la définition d'un template.

Exercice 16. *Rajoutez la définition d'un* `check-command` *dans le template* `angd-host`*. Faites les modifications nécessaires pour rendre l'ensemble cohérent.*

Exercice 17. *Utilisez* `/usr/local/libexec/nagios/check_dns` *pour superviser le serveur DNS du CIRM.*

Exercice 18. *Utiliser* `/usr/local/libexec/nagios/check_ssh` *pour superviser le serveur ssh de votre labo.*

5.2 Monitorer le serveur web de votre laboratoire

Le plugin qu'il faut utiliser est `/usr/local/libexec/nagios/check_http`.

Exercice 19. *Utilisez ce* `check_http` *juste avec l'option* `-I` *pour superviser le bon fonctionnement de votre site web*

Par défaut le plugin va faire une requête HTTP sur le host, donc sur le numéro IP d'une machine. Il ne permet donc pas de déterminer si un virtual_host est en état de fonctionnement. Or de plus en plus de sites utilisent les virtuels hosts. Nous avons donc besoin de pouvoir superviser un virtual host.

Exercice 20. *Utilisez le* `check_http` *pour superviser un virtual host d'un serveur Apache. Appliquez-le à votre serveur web.*

Exercice 21. *Faites vérifier à Nagios que la homepage de votre virtual host est bien la bonne*[4]*. Pour cela, vérifiez qu'une chaîne ou une expression régulière est bien présente dans la page.*

Exercice 22. *Utilisez l'option* `-C` *du plugin* `check_http` *pour créer un service qui surveille la validité du certificat de votre site web https. Rajoutez ce qu'il faut pour que le test ne soit fait qu'une fois par jour. Vérifiez que tout fonctionne.*

Exercice 23. *Repérez l'adresse IP du routeur de votre site. Créez un host correspondant à ce dernier. Renseignez le champ* `parent` *pour ce host et pour votre serveur web. Regardez le plan du réseau généré par Nagios.*

[4] il arrive parfois que des erreurs de configurations aboutissent à une intervertion de virtual hosts

6 Créer un plugin personnel

Il est possible de créer vos propres plugins.

Exercice 24. *Créez un plugin qui vérifie qu'un fichier n'a pas changé de signature md5*[5]*. Intégrez-le dans votre Nagios.*

Exercice 25. *Créez un plugin qui vérifie l'espace libre du dossier* `/tmp` *en utilisant la commande unix* `du`*. Passez deux bornes en argument pour la définition du service. La première pour une alerte type* warning *et une seconde de type* critical*. Intégrez-le dans votre nagios.*

7 Event handler

Exercice 26. *Modifiez le service que vous avez créé dans l'exercice 25 pour créer un event handler qui détruit les fichiers de* `/tmp` *de plus de 30 jours au premier appel, et ceux de plus de 8 jours au second*[6]*.*

Remarque 7. Pour effacer les fichiers de plus de X jours vous pouvez utiliser la commande

```
find /tmp -mtime +X -delete
```

8 NSCA

Exercice 27. *Modifiez les fichiers*

`/usr/local/etc/nagios/nsca.cfg`

et

`/usr/local/etc/nagios/nsca-send.cfg`

pour mettre un secret commun qui sera utilisé pour chiffrer les échanges. Relancez `nsca` *en utilisant*

`/usr/local/etc/rc.d/nsca restart`

Exercice 28. *Créez un nouveau* `template` *de service pour les services passifs. Vérifiez la syntaxe.*

Exercice 29. *Vous avez dans*

`/usr/local/adm/sauvegarde/backup.sh`

un petit script qui fait une sauvegarde sur une machine distante. Modifiez la fin du script pour qu'il renvoit un message `nsca` *vers le serveur Nagios. Faites ce qu'il faut sur le serveur pour qu'on puisse avoir une alerte en cas de soucis de la sauvegarde.*

Exercice 30. *Ajoutez un contrôle de fraicheur pour ce service, de façon a être alerté si l'état de ce service n'est pas raffraîchi au bout de n minutes.*

[5] Commande : `md5` sous FreeBSD

[6] utilisez la macro $SERVICEATTEMPT$

9 Monitoring via NRPE

9.1 Introduction

Nagios propose (heureusement) la possibilité de surveiller des services locaux sur une machine à distance. Les cas les plus classiques sont l'espace disque disponible, la charge CPU, etc...Pour cela le serveur Nagios utilise une connexion NRPE (Nagios Remote Plugin Execution). Sur votre deuxième serveur vous avez déjà un nrpe qui fonctionne. Nous allons dans cette partie configurer votre serveur NRPE pour superviser ce deuxième serveur. Nous allons voir que la configuration d'un superviseur via NRPE est très similaire à celui d'un service classique sous Nagios.

9.2 Définition d'un host externe

Exercice 31. *Sur le serveur Nagios, rajoutez un nouveau host correspondant à votre seconde VM*

9.3 Configuration de NRPE

En général NRPE ne possède qu'un seul fichier de configuration. Sur un serveur FreeBSD le fichier de configuration de NRPE se trouve dans

`/usr/local/etc/nrpe.cfg`[7]

La seule modification dans ce fichier au niveau système est l'adresse IP du serveur Nagios. Vous devez autoriser le serveur Nagios à se connecter sur le serveur NRPE pour récupérer les informations. Cela se fait via la directive `allowed_hosts`. Il est possible d'autoriser plusieurs serveurs Nagios.

Exercice 32. *Modifiez le fichier de configuration de NRPE pour autoriser votre serveur nagios. Relancer NRPE via la commande* `/usr/local/etc/rc.d/nrpe restart`

Vous trouverez vers la fin du fichier une partie qui liste un ensemble de `command` (au sens NRPE, mais similaire à celui de Nagios). Ce sont ces `command` qui sont interrogés par le serveur Nagios. Pour que Nagios puisse superviser via NRPE un serveur il faut rajouter des `service` en utilisant le plugin

`/usr/local/libexec/check_nrpe2`

en passant via l'option `-c` le nom du `command` que vous avez défini dans NRPE.

Exercice 33. *Utilisez le plugin* `check_nrpe2` *et créez sur votre serveur Nagios, un* `command` *qui permet de faire une interrogation sur un serveur NRPE. Par rapport à votre seconde VM, mettez en place la supervision de l'espace disque de* `/`*, de la charge CPU, du nombre d'utilisateurs.*

Exercice 34. *Rajoutez ce qu'il faut pour surveiller via NRPE la présence du processus* `apache` *et* `mysqld` *sur votre deuxième VM.*

[7] Sous Debian : `/etc/nagios/nrpe.cfg`

10 Dépendances

Comme vu dans la présentation, Nagios permet de gérer les dépendances. Cela permet d'avoir moins d'alerte inutile (on sait que si le service X tombe, alors le service Y ne fonctionne plus).

Vous avez sur votre deuxième VM un site tournant Drupal.

Exercice 35. *Utilisez l'option* `-r` *de* `check_http` *pour vérifier que dans la réponse du site web on ait bien la chaine de caractère* `Page de DRUPAL pour le TP ANGD`.

Exercice 36. *Créez un* `servicedependency` *entre ce que vous venez de faire dans l'exercice 35 et celui que vous avez fait avec 34.*