BTS SIO Gap (option SISR)

Lycée Dominique Villars 05000 Gap

Ecriture de commandes check pour nagios

Pré-requis:

Vous avez déjà ajouté des hôtes et des services et vous avez déjà exploité nagios.

Vous avez exploré les commandes disponibles et vous en avez testé un certain nombre.

Vous éprouvez le besoin d'exploiter un peu mieux nagios en lui en demandant plus.

Vous avez aussi déjà écrit un script en bash et vous savez le lancer.

Alors vous pouvez continuer. Sinon à plus tard … peut-être.

Introduction

On peut enrichir les plugins nagios en y ajoutant ses propres commandes qui peuvent être écrites en n'importe quel langage:

- c/c++
- perl
- php
- shell
- ...

Pourquoi

On code ses propres commandes quand on a des besoins spécifiques pour lesquels il n'existe pas encore de commande.

Vous pourriez avoir besoin de vérifier par exemple que le nombre de requêtes http envoyées à votre serveur web ne s'écarte pas de plus de 10% de la moyenne journalière.

Vous pouvez aussi vérifier que le nombre d'enregistrement dans une table d'une de vos base de données ne diminue pas etc…

Ou que le nombre de ligne dans vos journaux concernant les connexions ssh réussies est à zéro.

nagios et les commandes:

Les programmes correspondant aux commandes sont en général à stocker dans /usr/lib/nagios/plugins/

Une commande doit être définie avant de pouvoir être utilisée par nagios.

Il y a un fichier qui contient toutes les définitions des commandes que nagios peut utiliser.

Pour le trouver, il suffit de rechercher en récursif les define command dans /etc/nagios3: grep -r “define command” /etc/nagios3/*

Il faut y ajouter votre propre commande.

define command{

command_name maPropreCommande

command_line /usr/lib/nagios/plugins/checkMonBesoin.sh $HOSTADDRESS$ $arg1 $arg2 ...

}

Puis il faudra pour un **hôte** créer un **service** qui utilisera la commande en question (ça vous savez faire).

BTS SIO Gap (option SISR)

Lycée Dominique Villars 05000 Gap

Le code de retour de la commande:

Une commande renvoie un code :

- 0 si elle s'est bien passée,
- 1 s'il y a un warning ou
- 2 si l'erreur est critique

en shell exit 0 ou exit 1 ou exit 2

Les messages

En plus elle renvoie un message:

Commençant par

- OK -
- WARNING -
- ou CRITICAL -

l'affichage du message en shell se fait par un echo ex: echo "CRITICAL Le nombre de produit diminue".

;) L'echo est à effectuer avant le exit.

BTS SIO Gap (option SISR)

Lycée Dominique Villars 05000 Gap

Les paramètres (vus de la commande)

$0 contient la commande elle-même ex check_http

$1 contient le premier paramètre

$2 le deuxième

Dans l'exemple ci-dessus checkMonBesoin.sh $HOSTADDRESS$ $arg1 $arg2

$0 contiendra checkMonBesoin.sh

$1 contiendra l'adresse ip de l'hôte sur lequel le service appelant la commande est défini.

$2 contient le premier paramètre du service

$3 contient le deuxième paramètre du service

Exemple de commande

Voici un exemple de commande check de nagios écrite en shell:

#!/bin/bash

URL=$1

```
KEY=$2
CRIT=$3
http_proxy=""

DEBUG=

if [ -z $CRIT ]; then
echo "Usage $0 "
exit 3
fi

TC=`echo ${URL} | awk -F. '{print $1}' |awk -F/ '{print $NF}'`
TMP="/tmp/check_http_sh_${TC}.tmp"

CMD_TIME="curl -k --location --no-buffer --silent --output ${TMP} -w
%{time_connect}:%{time_starttransfer}:%{time_total} '${URL}'"
TIME=`eval $CMD_TIME`

if [ -f $TMP ]; then
RESULT=`grep -c $KEY $TMP`
else
echo "UNKOWN - Could not create tmp file $TMP"
# exit 3
fi

TIMETOT=`echo $TIME | gawk -F: '{ print $3 }'`
```

```
if [ ! -z $DEBUG ]; then
echo "CMD_TIME: $CMD_TIME"
echo "NUMBER OF $KEY FOUNDS: $RESULT"
echo "TIMES: $TIME"
echo "TIME TOTAL: $TIMETOT"
echo "TMP: $TMP"
ls $TMP
fi

rm -f $TMP

SURL=`echo $URL | cut -d "/" -f3-4`

MSGOK="Site $SURL key $KEY time $TIMETOT |'time'=${TIMETOT}s;${CRIT}"
MSGKO="Site $SURL has problems, time $TIMETOT |'time'=${TIMETOT}s;${CRIT}"

#PERFDATA HOWTO 'label'=value[UOM];[warn];[crit];[min];[max]

if [ "$RESULT" -ge "1" ] && [ $(echo "$TIMETOT < $CRIT"|bc) -eq 1 ]; then
echo "OK - $MSGOK"
exit 0
else
echo "CRITICAL - $MSGKO"
exit 2
```

BTS SIO Gap (option SISR)

Lycée Dominique Villars 05000 Gap

fi

TAF:

+Exercice 0: Testez et intégrez à nagios la commande fournie en exemple.

+Exercice 1: Écrire une commande nommée check_btssiogap

qui renvoie OK si www.btsinfogap.org est accessible

et CRITICAL suivi du code de l'erreur http s'il n'est pas accessible

Testez la commande

Intégrez la à Nagios.

+Exercice 2: Écrire une commande nommée check_defacage

prenant deux paramètres

- l'url de la page à tester
- et le checksum(somme de contrôle md5) attendu

permettant de vérifier que le contenu de la page statique passée en paramètre n'a pas été modifié par un pirate.

Testez la commande.

Intégrez-là à nagios: vous prendrez garde à ne l'utiliser que dans le service d'un seul host. Ce n'est pas la peine de la faire exécuter plus d'une fois.

Boite à outil (commandes à expérimenter):

Cut

echo"Jean-Paul" | cut -d"-" -f1 affiche Jean

echo"Jean-Paul" | cut -d"-" -f2 affiche Paul

Définition d'une variable Affectation Utilisation

DEBUG="ploum"

echo $DEBUG affiche ploum

Une commande dans une variable

CMD="ls"

eval $CMD //fait exécuter la commande

BTS SIO Gap (option SISR)

Lycée Dominique Villars 05000 Gap

Tester si un fichier existe:

if [-f “/tmp/ploum”]; then

  echo “il existe”

else

  echo “il n’existe pas”

fi

Tester une variable

toujours dans le if un -z $variable permet de tester si la variable est vide

-eq permet de tester l’égalité

-ge >= etc…

Faire une requête http et en récupérer le résultat

curl permet de faire une requête http en ligne de commande (man est votre ami et vous dira tout sur curl).

Charger le contenu d’un fichier d’une seule ligne dans une variable:

MAVARIABLE=`cat leFichier` attention (alt Gr 7 pour `)

Pensez à rendre votre commande exécutable pour l'utilisateur sous lequel le service nagios tourne.

Sous quel utilisateur tourne nagios?

La commande suivante vous apprend que le service est exécuté par un utilisateur nommé nagios (on pouvait le deviner mais il faut s'en assurer).

$ps -aef | grep nagios

nagios 19262 19258 0 07:45 ? 00:00:00 /bin/ping -n -U -w 30 -c 5 172.16.50.12

nagios 19263 19260 0 07:45 ? 00:00:00 /bin/ping -n -U -w 30 -c 5 172.16.50.13

nagios 19264 19261 0 07:45 ? 00:00:00 /bin/ping -n -U -w 30 -c 5 172.16.50.14

root 19266 4291 0 07:45 pts/0 00:00:00 grep nagios

nagios 31090 3118 0 2016 ? 00:47:51 /usr/sbin/ndo2db -c /etc/nagios3/ndo2db.cfg

Vous pouvez alors donner à l'utilisateur nagios votre commande avec la commande "change owner":

#chown nagios /usr/lib/nagios/plugins/checkMonBesoin.sh

Donner au propriétaire le droit de l'exécuter:

#chmod u+x /usr/lib/nagios/plugins/checkMonBesoin.sh