TP6 : Des graphs dans Nagios avec PNP4Nagios

Table des matières

Objectif : Ce document va nous permettre de grapher les plugins en s'appuyant sur leur performance.


INFO : C'est juste de l'information mais qui peut être nécessaire à la compréhension du TP


IMPORTANT : Information importante à prendre en compte


TP à Faire : Travaux à réaliser

TP6 : Des graphs dans Nagios avec PNP4Nagios

 PNP est l'acronyme de PNP is NOT Perfparse. Il permet de récupérer la partie performance de la sortie des plugins et d'injecter ces valeurs dans des bases rrdtool puis de les grapher via un front-end écrit en PHP.

6.1 – Installation de PNP4Nagios

 **- Prérequis :**

```
# sudo apt-get install rrdtool
# sudo apt-get install librrds-perl
# sudo apt-get install php5-gd
```

 **- Configurer PHP5 :**

```
# vi /etc/php5/apache2/php.ini

#Modifier la directive
#magic_quotes_gpc boolean

#Fixe le mode magic_quotes pour les opérations GPC (Get/Post/Cookie). Lorsque magic_quotes est activé, tous les caractères ' (guillemets simples), " (guillemets doubles), \ (antislash) et NUL sont échappés avec un antislash

magic_quotes_gpc = Off
```

 **- Activer le module rewrite de apache :**

```
# sudo a2enmod rewrite
```

 **- Redémarrer le service apache :**

```
# sudo /etc/init.d/apache2 reload
```

 **- Télécharger les sources de PNP :**

```
# cd /usr/src
# wget http://sourceforge.net/projects/pnp4nagios/files/PNP-0.6/pnp4nagios-0.6.24.tar.gz
```

 **- Décompresser les sources :**

```
# tar xvzf pnp4nagios-0.6.24.tar.gz
# cd pnp4nagios-0.6.24
```

 **- Lancer la compilation :**

```
# ./configure --prefix=/usr/local/pnp4nagios --with-nagios-user=nagios --with-
nagios-group=nagios

# make all
# make install
# make install-webconf
# make install-config
# make install-init
```

 **- Redémarrer le service apache :**

```
# sudo /etc/init.d/apache2 reload
```

 **- Tester :**

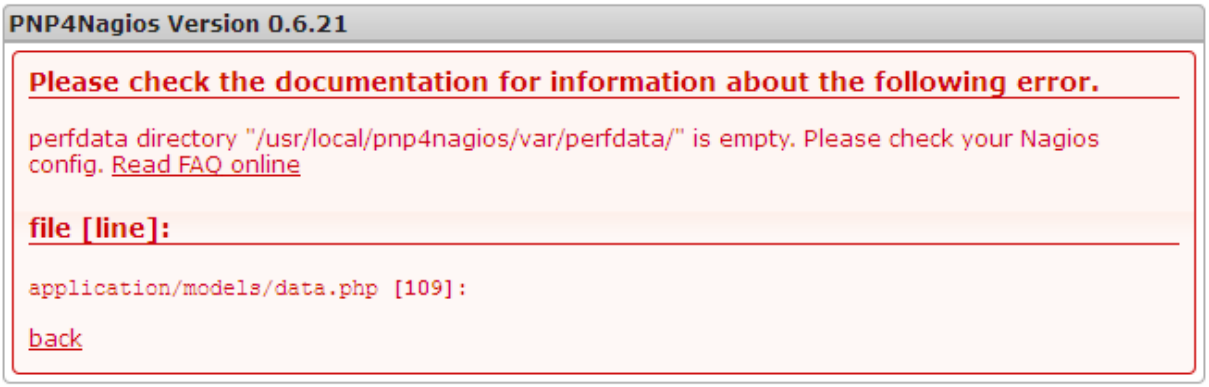
Utilisez votre navigateur préféré pour vérifier l'installation de PNP et du framework Kohana :

http://localhost/pnp4nagios/

- Nettoyer :

Nettoyage de l'installation : Supprimer le fichier /usr/local/pnp4nagios/share/install.php

Une fois le fichier d'installation supprimé, vous devez voir apparaitre ce message qui explique que le répertoire de données est vide, ce qui est normal.



6.2 – Configuration de PNP4Nagios

6.2.1 – Choix de la méthode

Il existe cinq méthodes pour intégrer PNP à Nagios :

- Synchronous Mode
- bulk mode
- bulk mode with NPCD.
- Bulk Mode with npcdmod
- Gearman Mode

Ici nous allons aborder la méthode « **Bulk mode with NPCD »**, ce mode est similaire au « **bulk mode with npcdmod** » (qui est plus simple au niveau de la configuration mais qui actuellement ne fonctionne plus avec Nagios 4.x) et il a exactement les mêmes fonctionnalités avec les mêmes performances.

6.2.2 – Fonctionnement

Nagios utilise un fichier temporaire pour stocker ses données et exécute régulièrement une commande. Au lieu de lancer directement le traitement avec process_perfdata.pl, le fichier est déplacé dans un répertoire spool. Comme le déplacement d'un fichier est pour ainsi dire instantané, Nagios peut donc continuer à travailler sur des tâches essentielles.

Le démon NPCD (Nagios Performance C Daemon) surveillera le contenu du répertoire et passera au script process_perfdata.pl les nouveaux fichiers. Dans ce mode, la gestion des données de

performance est complètement découplée du fonctionnement de Nagios. NPCD est capable de lancer plusieurs threads de traitement afin de traiter les données.

6.2.3 - Configuration

 **- Copier le fichier de configuration exemple et lui donner les bons droits :**

```
# cp /usr/src/pnp4nagios-0.6.24/sample-config/pnp/npcd.cfg-sample
/usr/local/pnp4nagios/etc/npcd.cfg
# chown nagios:nagios npcd.cfg
```

 **- Editer le fichier nagios.cfg et modifier la configuration :**

```
process_performance_data=1

#
# service performance data
#
service_perfdata_file=/usr/local/pnp4nagios/var/service-perfdata
service_perfdata_file_template=DATATYPE::SERVICEPERFDATA\tTIMET::$TIMET$\tHOSTNAME::$HOSTNA
ME$\tSERVICEDESC::$SERVICEDESC$\tSERVICEPERFDATA::$SERVICEPERFDATA$\tSERVICECHECKCOMMAND::$
SERVICECHECKCOMMAND$\tHOSTSTATE::$HOSTSTATE$\tHOSTSTATETYPE::$HOSTSTATETYPE$\tSERVICESTATE:
:$SERVICESTATE$\tSERVICESTATETYPE::$SERVICESTATETYPE$
service_perfdata_file_mode=a
service_perfdata_file_processing_interval=15
service_perfdata_file_processing_command=process-service-perfdata-file

#
# host performance data starting with Nagios 3.0
#
host_perfdata_file=/usr/local/pnp4nagios/var/host-perfdata
host_perfdata_file_template=DATATYPE::HOSTPERFDATA\tTIMET::$TIMET$\tHOSTNAME::$HOSTNAME$\tH
OSTPERFDATA::$HOSTPERFDATA$\tHOSTCHECKCOMMAND::$HOSTCHECKCOMMAND$\tHOSTSTATE::$HOSTSTATE$\t
HOSTSTATETYPE::$HOSTSTATETYPE$
host_perfdata_file_mode=a
host_perfdata_file_processing_interval=15
host_perfdata_file_processing_command=process-host-perfdata-file
```

 **Explication des différents paramètres du fichier nagios.cfg :**

- **service_perfdata_file** : chemin du fichier temporaire qui contiendra les données.
- **service_perfdata_file_template** : format du fichier temporaire. Les données qui seront définies utilisent les macros de Nagios.
- **service_perfdata_file_mode** : option “a” spécifie que les données doivent être ajoutées dans ce fichier.
- **service_perfdata_file_processing_interval** : l’intervalle d’exécution est de 15 secondes.

- **service_perfdata_file_processing_command** : nom de la commande qui sera déclarée dans le fichier commands.cfg.

Les commandes **process-service-perfdata-file** et **process-host-perfdata-file** :

```
define command{
       command_name    process-service-perfdata-file
       command_line    /bin/mv /usr/local/pnp4nagios/var/service-perfdata
/usr/local/pnp4nagios/var/spool/service-perfdata.$TIMET$
}

define command{
       command_name    process-host-perfdata-file
       command_line    /bin/mv /usr/local/pnp4nagios/var/host-perfdata
/usr/local/pnp4nagios/var/spool/host-perfdata.$TIMET$
}
```

 **- Démarrer le démon npcd :**

```
# /etc/init.d/npcd start
```

 **- Vérifier et relancer Nagios :**

```
# /usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg
# sudo /etc/init.d/nagios restart
```

 **- Vérifier les logs :**

```
# tail –f /usr/local/nagios/var/nagios.log

[1320144464] npcdmod: Copyright (c) 2008-2009 Hendrik Baecker (andurin@process-zero.de) -
http://www.pnp4nagios.org
[1320144464] npcdmod: /usr/local/pnp4nagios/etc/npcd.cfg initialized
[1320144464] npcdmod: spool_dir = '/usr/local/pnp4nagios/var/spool/'.
[1320144464] npcdmod: perfdata file '/usr/local/pnp4nagios/var/perfdata.dump'.
[1320144464] npcdmod: Ready to run to have some fun!
[1320144464] Event broker module '/usr/local/pnp4nagios/lib/npcdmod.o' initialized
successfully.
```

6.3 – Comment accéder à PNP4Nagios

Après l'installation de Nagios on a deux interfaces distinctes : celle de **Nagios** et celle de **PNP**

 - **Accéder à PNP avec l'Url suivante : http://localhost/pnp4nagios/**

Si tout fonctionne, les premiers graphs devraient apparaitre :

Si vous avez un problème vous pouvez contrôler votre configuration avec la commande suivante :

```
#     /usr/src/pnp4nagios-0.6.24/scripts/verify_pnp_config_v2.pl     -m     bulk+npcd     -c
/usr/local/nagios/etc/nagios.cfg -p /usr/local/pnp4nagios/etc/

[OK  ]  Command process-service-perfdata-file is defined
[OK               ]               '/bin/mv       /usr/local/pnp4nagios/var/service-perfdata
/usr/local/pnp4nagios/var/spool/service-perfdata.$TIMET$'
[OK  ]  Command looks good
host_perfdata_file_processing_command                at                /usr/src/pnp4nagios-
0.6.24/scripts/verify_pnp_config_v2.pl line 462.
[OK  ]  Command process-host-perfdata-file is defined
[OK                ]                '/bin/mv        /usr/local/pnp4nagios/var/host-perfdata
/usr/local/pnp4nagios/var/spool/host-perfdata.$TIMET$'
[OK  ]  Command looks good
[OK  ]  Script /usr/local/pnp4nagios/libexec/process_perfdata.pl is executable
[INFO]  ========== Starting global checks ============
[OK  ]  status_file is defined
[OK  ]  status_file=/usr/local/nagios/var/status.dat
[INFO]  host_query =
[INFO]  service_query =
[INFO]  Reading /usr/local/nagios/var/status.dat
[INFO]  ==== Starting rrdtool checks ====
[OK  ]  RRDTOOL is defined
[OK  ]  RRDTOOL=/usr/bin/rrdtool
[OK  ]  /usr/bin/rrdtool is executable
[OK  ]  RRDtool 1.4.7  Copyright 1997-2012 by Tobias Oetiker <tobi@oetiker.ch>
[OK  ]  USE_RRDs is defined
[OK  ]  USE_RRDs=1
[OK  ]  Perl RRDs modules are loadable
…
```

 **Fonctionnement :**

Les "templates" sont des modèles que PNP utilise pour l'aspect des graphiques produits à partir des données fournies par RRD (Round-Robin-Data).

Le "template" correspond au service check_<commande> qui a récupéré les données. La suite décrit où sont stockés les "templates" et comment ils sont sélectionnés.

Exemple :

Commande : **check_load**
Template utilisé : **check_load**.php

Les “templates” sont stockés dans deux sous-répertoires de celui de pnp (ex. /usr/local/pnp4nagios/share/…)

- …/templates.dist - pour les “templates” fournis avec le packet PNP.
- …/templates - pour les “templates” personnalisés. Ceux-ci ne sont pas modifiés lors d'une mise à jour.

Si nous prenons l’exemple de la commande **check_load**, PNP recherche maintenant un “template” portant le nom **check_load**.php dans l'ordre suivant des dossiers :

1. templates/check_load.php
2. templates.dist/check_load.php
3. <autre défini dans config.php>/check_load.php
4. templates/default.php
5. templates.dist/default.php

Le “template” default.php est particulier car il sera sélectionné si aucun autre portant le nom recherché n'a été trouvé.

 - **Créer de nouveaux templates pour grapher les services définis dans le fichier de configuration du serveur Nagios localhost.cfg :**

```
#               cp               /usr/local/pnp4nagios/share/templates.dist/check_swap.php
/usr/local/pnp4nagios/share/templates.dist/check_local_swap.php

#               cp               /usr/local/pnp4nagios/share/templates.dist/check_load.php
/usr/local/pnp4nagios/share/templates.dist/check_local_load.php

#               cp              /usr/local/pnp4nagios/share/templates.dist/check_users.php
/usr/local/pnp4nagios/share/templates.dist/check_local_users.php
```

Si *le template correspondant à la commande n’existe pas, PNP prend par défaut le template* ***default.php***

 **- Créer un template pour la commande check_nrpe :**

La commande check_nrpe permet de faire appel à une autre commande sur un serveur distant. Elle est définie de cette manière-là : check_command check_nrpe!check_users

PNP va vouloir utiliser un template qui se nomme check_nrpe.php alors que le template qui nous intéresse est check_users.php

Pour contourner ce problème il suffit de créer un fichier check_nrpe.cfg dans le répertoire /usr/local/pnp4nagios/etc/check_commands/

Fichier check_nrpe.cfg

```
#check_command      check_nrpe!check_users!-w 3 -c 6
#_______0_____________||
#_____________1_____________||
#_____________________2_____________________||
```

```
#
CUSTOM_TEMPLATE = 1
```

CUSTOM_TEMPLATE = 1 Permet d'utiliser le template contenu dans la variable $arg1$ de la commande.

Ici le template utilisé sera donc check_users.php pour la commande check_users.

6.4 – Intégration de PNP4Nagios à Nagios

Pour intégrer PNP à Nagios il faut utiliser les action_url.

 **- Définir les action_url dans deux nouveaux templates :**

Fichier templates.cfg

```
define host {
   name       pnp-host
   action_url /pnp4nagios/index.php/graph?host=$HOSTNAME$&srv=_HOST_' class='tips'
rel='/pnp4nagios/index.php/popup?host=$HOSTNAME$&srv=_HOST_
   register   0
}


define service {
   name       pnp-service
   action_url /pnp4nagios/index.php/graph?host=$HOSTNAME$&srv=$SERVICEDESC$' class='tips'
rel='/pnp4nagios/index.php/popup?host=$HOSTNAME$&srv=$SERVICEDESC$
   register   0
}
```

Optionnel : Pour intégrer PNP à Nagios sous forme de pop-up :

 **- Copier le fichier status-header.ssi dans /usr/local/nagios/share/ssi/ :**

```
# cp /usr/src/pnp4nagios-0.6.21/contrib/ssi/status-header.ssi /usr/local/nagios/share/ssi/
```

 **- Activer les données de performance sur un hôte :**

Editer la configuration d'un hôte puis ajouter le template « **pnp-host** » dans la définition de l'hôte.

Par exemple :

```
define host{
        host_name           serveur-linux
        use                 generic-server-linux,pnp-host
        alias               serveur-linux
        address             Adresse IP
}
```

Et le template « **pnp-service** » pour un service :

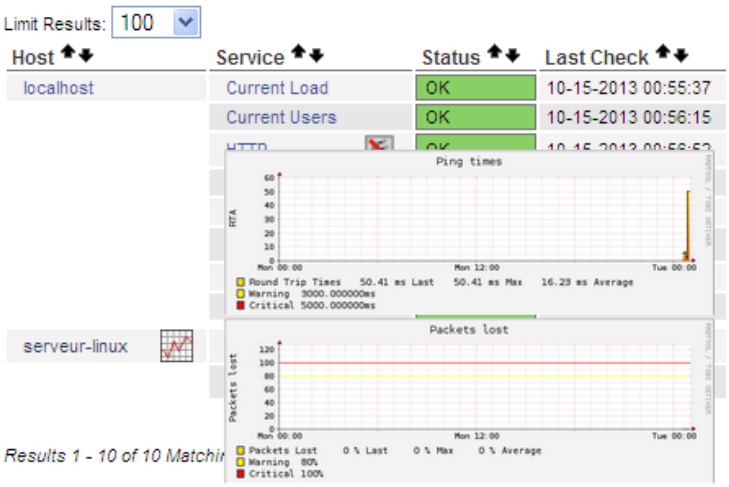
```
define service{
        use                         local-service,pnp-service
        host_name                   localhost
        service_description         Swap Usage
        check_command               check_local_swap!20!10
        }
```

- Redémarrer le service Nagios :

```
# sudo /etc/init.d/nagios restart
```

Une fois Nagios redémarré une nouvelle icône apparaît 

En vous positionnant (ou en cliquant) dessus vous avez accès aux graphs :



Compte Rendu-12 : Utiliser le template check_mk-cpu.loads développé par Romuald FRONTEAU (à rechercher sur internet) pour grapher le service Current Load du serveur Nagios.

- Faire une capture d'écran du graphique obtenu.