Activité : Enrichir la supervision avec les packs

Présentation

Ouvrez les différents fichiers, regardez leur contenu pour comprendre leurs relations et effectuer les actions permettant de superviser **localhost** en tant que **hôte Linux**.

Pour l'instant la supervision se contente de tester la présence de notre hôte avec un **ping**. Mais il est possible d'obtenir beaucoup plus d'informations sur les hôtes supervisés.

La communauté des développeurs de Shinken met à disposition un certain nombre de solutions pour superviser une machine Linux.

```
root@ctShinken:~# su - shinken
shinken@cdShinken:~$ shinken search linux
glances (david-guenault) [pack,system,linux,glances] : Standard check
through checkglances.py and glances server
linux-snmp (naparuba) [pack,linux,snmp] : Linux checks based on SNMP
linux-ssh (naparuba) [pack,linux,ssh] : Linux checks based on SSH without
any script on distant server
pack-glances (david-guenault) [pack,system,linux,glances] : Standard check
through checkglances.py and glances server
raspberrypi (frescha) [pack,linux,raspberrypi,server,os] : Standard checks
varnish-ssh (kokosny) [pack,linux,varnish,ssh] : varnish checks based on ssh
```

Dans un premier temps, vous allez mettre en oeuvre **linux-snmp**.

Pour cela il faut télécharger le **pack** nécessaire :

- contenu du dossier **$ETC/packs**.

```
shinken@cdShinken:~$ ls /etc/shinken/packs/
readme.cfg
```

- contenu du fichier **readme.cfg** :

```
#In this place you will find all your packs downloaded from
shinken.iowebsite.
#
#you can freely adapt them to your own need
```

Le dossier est vide et prêt à recevoir les packs.

Installation du pack linux-snmp

```
shinken@cdShinken:~$ shinken install linux-snmp
Grabbing :linux-snmp
OK linux-snmp
```

Un dossier est créé dans le répertoire **/etc/shinken/packs/**:

```
shinken@cdShinken:~$ ls /etc/shinken/packs/
linux-snmp readme.cfg
shinken@cdShinken:~$ ls /etc/shinken/packs/linux-snmp/
commands.cfg discovery.cfg linux-snmp.pack  services    templates.cfg
```

Qu'est ce qu'un pack ?

Un **pack** est un ensemble de fichiers de configuration et/ou de commandes développés par la communauté des utilisateurs et permettant la **supervision** des hôtes correspondant à des modèles génériques (Linux par exemple).

La lecture du fichier **commands.cfg** montre les commandes qui seront appliquées sur les hôtes de type Linux.

```
# ------------------------------------------------------------------
#
#      Linux standard check
#
# ------------------------------------------------------------------
#
define command {
command_name check_linux_load
command_line  $PLUGINSDIR$/check_snmp_load.pl -H $HOSTADDRESS$ -C
$_HOSTSNMPCOMMUNITY$ -f -w $_HOSTLOAD_WARN$ -c $_HOSTLOAD_CRIT$ -T netsl -o
$_HOSTSNMP_MSG_MAX_SIZE$
}

define command {
command_name check_linux_disks
command_line  $PLUGINSDIR$/check_snmp_storage.pl -H $HOSTADDRESS$ -C
$_HOSTSNMPCOMMUNITY$ -m $_HOSTSTORAGE_PATH$ -f -w $_HOSTSTORAGE_WARN$ -c
$_HOSTSTORAGE_CRIT$ -S0,1 -o $_HOSTSNMP_MSG_MAX_SIZE$
}

define command {
command_name check_linux_cpu
command_line  $PLUGINSDIR$/check_snmp_load.pl -H $HOSTADDRESS$ -C
$_HOSTSNMPCOMMUNITY$ -f -w $_HOSTCPU_WARN$ -c $_HOSTCPU_CRIT$ -o
$_HOSTSNMP_MSG_MAX_SIZE$
```

```
}

# Added -g flag since all linux system used are 64bits.
define command {
command_name check_linux_network_usage
command_line  $PLUGINSDIR$/check_netint.pl -H $HOSTADDRESS$ -C
$_HOSTSNMPCOMMUNITY$ -n "$_HOSTNET_IFACES$" -g -2c -f -e -w $_HOSTNET_WARN$
-c $_HOSTNET_CRIT$ -q -k -y -M -B -m -P "$SERVICEPERFDATA$" -T
"$LASTSERVICECHECK$"  -o $_HOSTSNMP_MSG_MAX_SIZE$
}

define command {
command_name check_linux_memory
command_line  $PLUGINSDIR$/check_snmp_mem.pl -w $_HOSTMEMORY_WARN$ -c
$_HOSTMEMORY_CRIT$ -- -v 2c -c $_HOSTSNMPCOMMUNITY$ $HOSTADDRESS$
}

define command {
command_name check_linux_logfiles
command_line  $PLUGINSDIR$/check_logfiles -f $_HOSTCHKLOG_CONF$
}

define command {
command_name check_linux_time
command_line  $NAGIOSPLUGINSDIR$/check_ntp_time -H $HOSTADDRESS$ -w
$_HOSTNTP_WARN$ -c $_HOSTNTP_CRIT$
}
```

Ces commandes sont situées dans le répertoire **$PLUGINSDIR$** ou **$NAGIOSPLUGINSDIR$**, soit comme défini dans le fichier **$ETC/resource.d/path.cfg**

```
# Nagios legacy macros
$USER1$=$NAGIOSPLUGINSDIR$
$NAGIOSPLUGINSDIR$=/usr/lib/nagios/plugins

#-- Location of the plugins for Shinken
$PLUGINSDIR$=/var/lib/shinken/libexec
```

Le **modèle linux hérite** lui-même du modèle **generic_host**, comme on le constate en ouvrant le fichier **templates.cfg**

```
# The LINUX template.
define host {
name linux-snmp
use    generic-host
check_command check_ping
register  0

    # We will show the linux custom view
custom_views        +linux
```

```
    _SNMPCOMMUNITY      $SNMPCOMMUNITYREAD$
    _SNMP_MSG_MAX_SIZE  65535

    _LOAD_WARN          2,2,2
    _LOAD_CRIT          3,3,3
    _STORAGE_WARN       90
    _STORAGE_CRIT       95
    _CPU_WARN           80
    _CPU_CRIT           90
    _MEMORY_WARN        80
    _MEMORY_CRIT        95
    _NTP_WARN           0.128
    _NTP_CRIT           1
    _NET_IFACES         eth\d+|em\d+
    _NET_WARN           90,90,0,0,0,0
_NET_CRIT           0,0,0,0,0,0

    _CHKLOG_CONF        $PLUGINSDIR$/logFiles_linux.conf
_STORAGE_PATH       /
}

define service {
name linux-service
use                 generic-service
register           0
aggregation    system
}
```

Il utilise donc des commandes définies au niveau le plus haut mais il définit aussi ses propres commandes spécifiques dans son fichier **commands.cfg**.

Voici de quelle manière les commandes du fichier **commands.cfg** sont appliquées au modèle Linux :

- Une commande n'**est pas appelée directement par l'hôte** (à part le ping),
- les commandes sont **appelées via les services** car ce sont des services qui tournent sur les OS.

Le **modèle Linux** définit un **service linux-service** qui fait appel lui-même à un service générique **generic-service**.

Exemple de définition d'un service spécifique **cpu.fcg** :

```
define service {
service_descriptionCpu
use                   20min_long,linux-service
register              0
host_name linux-snmp
check_command check_linux_cpu

    _DETAILLEDESC         Detect abnormal CPU usage
    _IMPACT               Slow down applications hosted by the system
```

```
    _FIXACTIONS           If recurrent situation then make performance audit
}
```

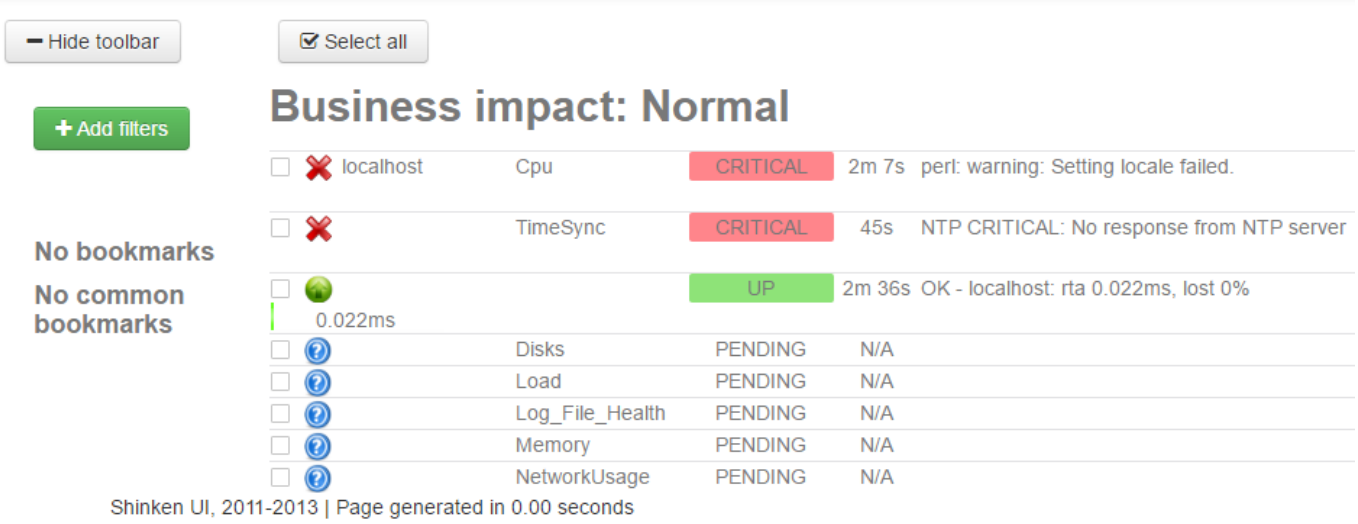
Deux informations sont à identifier :

- l'**appel au modèle linux-service** par la directive **use**,
- l'établissement de la **relation** entre le service et le modèle d'hôte linux-snmp par la directive **host_name**.

</code> Il reste maintenant à établir une relation entre notre hôte réel **localhost** et notre modèle **linux-snmp**. C'est dans le fichier **$ETC/hosts/localhost.cfg** que cette relation est établie :

```
define host {
       use                     linux-snmp,generic-host
       contact_groups          admins
       host_name               localhost
       address                 localhost
}
```

Relancez le service shinken puis **visualiser** le résultat dans l'interface webui.



De nouveaux indicateurs sont présents mais cela ne fonctionne pas car **snmp** n'est probablement **pas configuré** sur l'hôte.

Installation et configuration de SNMP

- installation du paquet snmpd :

```
root@ctShinken:~# apt-get install snmpd
```

Attention il faut que dans le fichier **/etc/snmp/snmpd.conf** on écoute bien sur **localhost** car la variable $HOSTADDRESS$ prendra la valeur **localhost** qui sera traduite

par 127.0.0.1 via /etc/hosts.

Il faut donc vérifier ou modifier le fichier de configuration pour que le serveur SNMP écoute sur le port 161 sur son adresse IP.

Pour vérifier si le port est à l'écoute, 2 solutions sont possibles :

- soit avec la commande **netstat –ulnp**,
- soit avec la commande **nmap –sU localhost** si **nmap** est installé.

Pour cela, éditez le fichier **/etc/snmp/snmpd.conf** et modifiez la valeur du paramètre **agentAddress** avec la valeur **localhost** de votre serveur :

```
#agentAddress udp:127.0.0.1:161
agentAddress udp:localhost:161
```

Il faut aussi modifier dans ce fichier les conditions d'accès à la MIB :

- l'accès en lecture est défini par la directive **rocommunity**.
- par défaut, cet accès est restreint à une seule vue **systemonly** via l'**option -V**.

La vue **systemonly** est définie ainsi :

```
viewsystemonly included .1.3.6.1.2.1.1
viewsystemonly included .1.3.6.1.2.1.25.1
```

L'accès par défaut en lecture, faisant référence à la vue **systemonly**, est défini ainsi :

```
rocommunity public default -V systemonly
```

Cela restreint l'accès aux deux branches .1.3.6.1.2.1.1 (**system**) et .1.3.6.1.2.1.25.1 (**hrSystem**) et uniquement celles-ci.

Il est possible de modifier ces restrictions en créant par exemple une vue **all** qui sera utilisée ensuite à la place de **systemonly**. Cela donne les modifications suivantes dans le fichier **/etc/snmp/snmpd.conf** :

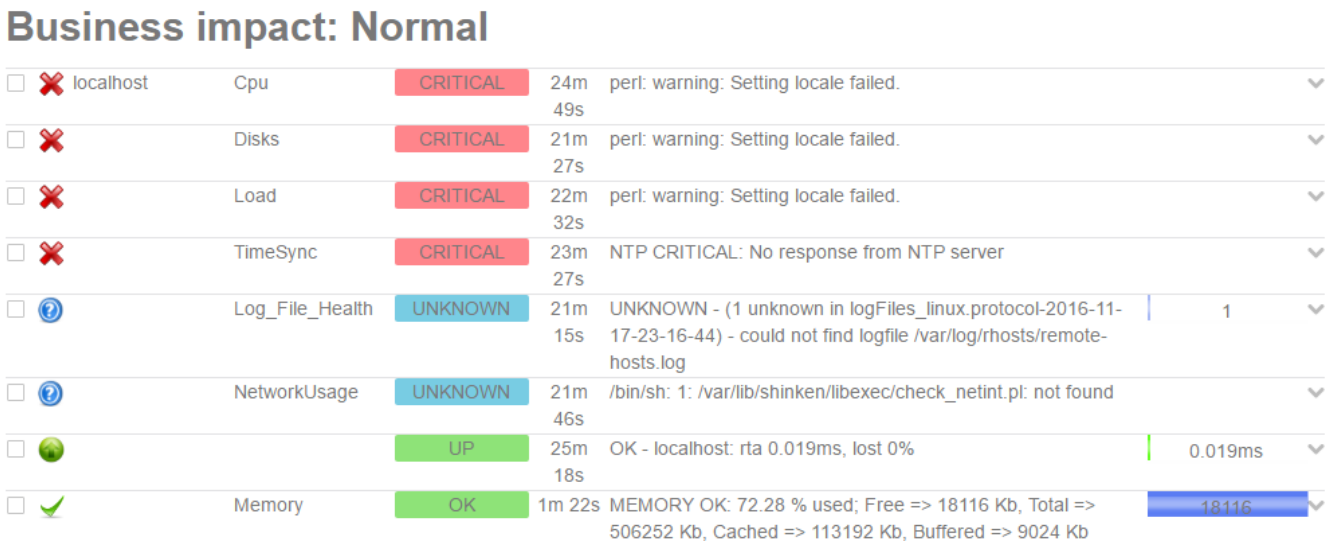
```
view all included .1
rocommunity public default -V all
```

Toute la branche **iso** sera ainsi accessible en lecture.

Il est nécessaire de **redémarrer** le service **snmpd** pour que le changement soit pris en compte.

```
root@ctShinken:~# service snmpd restart
```

Visualisez le résultat dans webui :



Il y a davantage de retour d'informations. La configuration n'est pas terminée.

Test des commandes du modèle linux-snmp

Le test manuel des commandes présentes dans le modèle (**template**) linux va permettre de voir celles qui ne fonctionnent pas correctement.

Rappel sur le test d'une commande :

Il faut :

- regarder quel est l'exécutable lancé par la commande,
- se positionner sur le dossier
- et lancer la commande.

Pour la commande la commande check_linux_cpu :

Visualiser cette commande dans le fichier **/etc/shinken/packs/linux-snmp/commands.cfg** :

```
define command {
command_namecheck_linux_cpu
command_line     $PLUGINSDIR$/check_snmp_load.pl -H $HOSTADDRESS$ -C
$_HOSTSNMPCOMMUNITY$ -f -w $_HOSTCPU_WARN$ -c $_HOSTCPU_CRIT$
}
```

Il faut remplacer toutes les variables par leurs valeurs.

Ces valeurs se Où trouver les valeurs ?

On va trouver les valeurs critiques $_HOSTCPU_CRIT$ et warning $_HOSTCPU_WARN$ du cpu dans le fichier **/etc/shinken/packs/linux-snmptemplates.cfg**.

```
# The LINUX template.
…
```

```
   # We will show the linux custom view
custom_views               +linux

   _SNMPCOMMUNITY       $SNMPCOMMUNITYREAD$

   _LOAD_WARN                  2,2,2
   _LOAD_CRIT                  3,3,3
   _STORAGE_WARN               90
   _STORAGE_CRIT               95
   _CPU_WARN                   80
   _CPU_CRIT                   90
   _MEMORY_WARN                90,20
   _MEMORY_CRIT                95,50
   _NET_WARN                   90,90,0,0,0,0
_NET_CRIT                   0,0,0,0,0,0

}
```

Ainsi on peut taper la commande :

```
shinken@ctShinken:~$ /var/lib/shinken/libexec/check_snmp_load.pl -H
localhost -C public -f -w 80 -c 90
Can't locate utils.pm in @INC (@INC contains: /var/lib/shinken/libexec
/etc/perl /usr/local/lib/perl/5.14.2 /usr/local/share/perl/5.14.2
/usr/lib/perl5 /usr/share/perl5 /usr/lib/perl/5.14 /usr/share/perl/5.14
/usr/local/lib/site_perl .) at /var/lib/shinken/libexec/check_snmp_load.pl
line 22.
BEGIN failed--compilation aborted at
/var/lib/shinken/libexec/check_snmp_load.pl line 22.
```

On s'aperçoit que la commande fait appel à un fichier **utils.pm** qu'elle ne trouve pas. Ce fichier appartient en fait aux plugins nagios et la commande le cherche dans le répertoire courant.

Il est nécessaire de créer un lien symbolique du répertoire **libexec** vers le répertoire **nagios/plugins** (voir http://fr.wikipedia.org/wiki/Lien_symbolique).

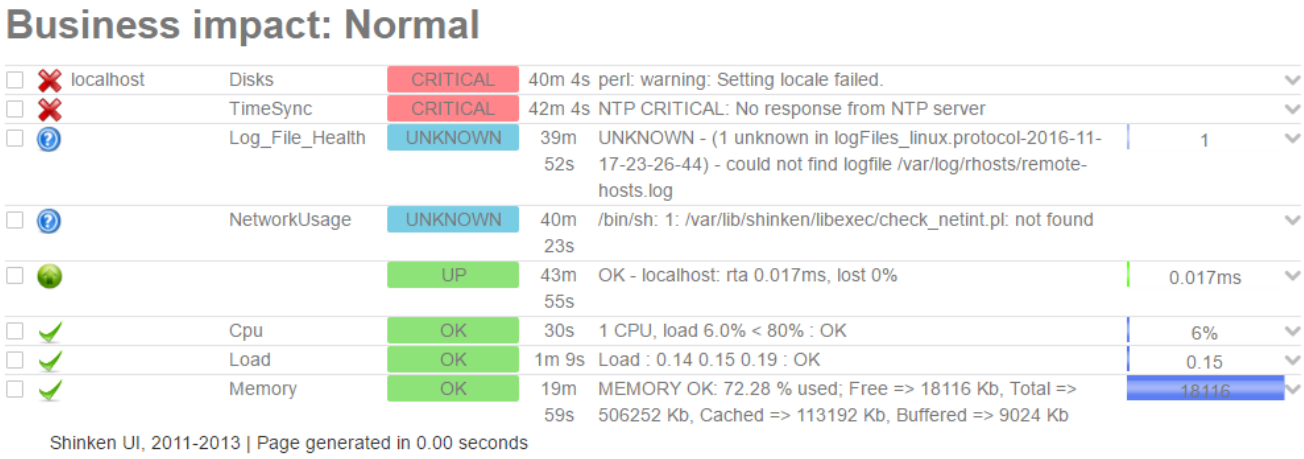
```
root@ctShinken# ln -s /usr/lib/nagios/plugins/utils.pm
/var/lib/shinken/libexec/utils.pm
```

Testez à nouveau la commande :

```
shinken@cShinken:~$ /var/lib/shinken/libexec/check_snmp_load.pl -H localhost
-C public -f -w 80 -c 90
Argument "v6.0.1" isn't numeric in numeric lt (<) at
/var/lib/shinken/libexec/check_snmp_load.pl line 685.
1 CPU, load 14.0% < 80% : OK | cpu_prct_used=14%;80;90
```

A priori cela fonctionne même si une erreur non bloquante est relevée dans le script.

Regardez dans webui :



- l'erreur sur **TimeSync** est due à l'absence de définition d'un serveur NTP,
- l'erreur sur **Log_File_Health** est due à l'absence du paramétrage de l'emplacement des logs.

A faire :

- **Mettre en place NTP** et vérifier sa prise en charge par Shinken :
  - pour vous aider : https://computerz.solutions/debian-8-shinken-corriger-lerreur-ntpq-not-found/
- **Enlever la référence** à la commande gérant les logs dans le modèle linux-snmp
- et vérifier le résultat sur webui.

Je reviens au menu Shinken

- Supervision des services avec Shinken