Centreon Plugins Documentation

Version

Merethis

11 January 2019

centreon 46/52, rue Albert – 75013 Paris – FRANCE
tél +33 (0)1 49 69 97 12 fax +33 (0)1 78 12 00 28 mail contact@centreon.com
SAS au capital de 500 000 € – 483 494 589 RCS Créteil – TVA FR 02 483 494 589
www.centreon.com

Table des matières

Centreon Plugins est un ensemble de bibliothèques et plugins de supervision écrits en Perl. Cet ensemble est licencié sous les termes de *Apache License Version 2 <https ://www.apache.org/licenses/LICENSE-2.0>* tel que publié par la “Free Software Fondation”.

Sommaire :

CHAPITRE 1

Guide utilisateur

1.1 Description

"centreon-plugins" est un projet gratuit et open source de supervision des systèmes. Ce projet peut être utilisé avec Centreon, Icinga et tout autre logiciel de supervision compatible avec les plugins Nagios.

La dernière version est disponible sur le dépôt git suivant : https://github.com/centreon/centreon-plugins.git

1.2 Installation

1.2.1 Debian Wheezy

Télécharger la dernière version de "centreon-plugins" depuis le dépôt :

```
# aptitude install git
# git clone https://github.com/centreon/centreon-plugins.git
```

Pour superviser les systèmes SNMP, vous devez installer les paquets suivants :

```
# aptitude install perl libsnmp-perl
```

Vous pouvez installer d'autres paquets pour utiliser plus de plugins :

```
# aptitude install libxml-libxml-perl libjson-perl libwww-perl libxml-xpath-perl libnet-telnet-perl
```

Pour utiliser la fonctionnalité 'memcached', vous devez installer le module CPAN suivant (pas de paquet debian) : http://search.cpan.org/~wolfsage/Memcached-libmemcached-1.001702/libmemcached.pm

1.2.2 Centos/Rhel 6

Télécharger la dernière version de "centreon-plugins" depuis le dépôt :

```
# yum install git
# git clone https://github.com/centreon/centreon-plugins.git
```

Pour superviser les systèmes SNMP, vous devez installer les paquets suivants :

```
# yum install perl net-snmp-perl
```

Vous pouvez installer d'autres paquets pour utiliser plus de plugins :

```
# yum install perl-XML-LibXML perl-JSON perl-libwww-perl perl-XML-XPath perl-Net-Telnet perl-Net-DNS
```

Pour utiliser la fonctionnalité 'memcached', vous devez installer le module CPAN suivant (paquet disponible dans 'rpmforge') : http://search.cpan.org/~wolfsage/Memcached-libmemcached-1.001702/libmemcached.pm

1.3 Utilisation basique

Nous allons utiliser un exemple basique pour montrer comment superviser un système. J'ai terminé partie installation et je veux superviser un système Linux par SNMP. Tout d'abord, j'ai besoin de trouver le plugin à utiliser dans la liste :

```
$ perl centreon_plugins.pl --list-plugin | grep -i linux | grep 'PLUGIN'
PLUGIN: os::linux::local::plugin
PLUGIN: os::linux::snmp::plugin
```

Il semblerait que 'os : :linux : :snmp : :plugin' est le bon donc je vérifie avec l'option `--help` pour être sûr :

```
$ perl centreon_plugins.pl --plugin=os::linux::snmp::plugin --help
...
Plugin Description:
  Check Linux operating systems in SNMP.
```

C'est exactement ce dont j'ai besoin. Maintenant je vais utiliser l'option `--list-mode` pour connaître ce que je peux faire avec celui-ci :

```
$ perl centreon_plugins.pl --plugin=os::linux::snmp::plugin --list-mode
...
Modes Available:
 cpu
 cpu-detailed
 disk-usage
 diskio
 inodes
 interfaces
 list-diskspath
 list-interfaces
 list-storages
 load
 memory
 processcount
 storage
 swap
 tcpcon
 time
 uptime
```

J'aimerais tester le mode 'load' :

```
$ perl centreon_plugins.pl --plugin=os::linux::snmp::plugin --mode=load
UNKNOWN: Missing parameter --hostname.
```

Il ne fonctionne pas car certaines options sont manquantes. Je peux avoir une description du mode et ses options avec l'option `--help` :

```
$ perl centreon_plugins.pl --plugin=os::linux::snmp::plugin --mode=load --help
```

Je dois éventuellement configurer certaines options SNMP :

```
$ perl centreon_plugins.pl --plugin=os::linux::snmp::plugin --mode=load --hostname=127.0.0.1 --snmp-v
OK: Load average: 0.00, 0.00, 0.00 | 'load1'=0.00;;;0; 'load5'=0.00;;;0; 'load15'=0.00;;;0;
```

Je peux spécifier des seuils avec les options `--warning` et `--critical` :

```
$ perl centreon_plugins.pl --plugin=os::linux::snmp::plugin --mode=load --hostname=127.0.0.1 --snmp-v
OK: Load average: 0.00, 0.00, 0.00 | 'load1'=0.00;0:1;0:2;0; 'load5'=0.00;0:2;0:3;0; 'load15'=0.00;0
```

1.4 FAQ

1.4.1 Qu'est ce que je peux superviser ?

L'option `--list-plugin` peut être utilisée pour obtenir la liste des plugins, ainsi qu'une courte description.

Les en-têtes du tableau signifient :

- Transport : Le point de contrôle a des options internes pour le transport.
- Protocole : qu'est-ce qui est utilisé pour obtenir les informations de supervision ?
- Expérimental : Le point de contrôle est en cours de développement.

Categorie	Contrôle	Transport			Protocole					Experimental	Comm
		SSH	TELNET	WSMAN	SNMP		HTTP		WMI		JMX
Application	Active Directory								—		Utilis la co mand 'dc-diag'. Doit être instal sur Win-dows.
	Apache					—					Requi le m dule Apacl 'mod
	Apc					—					
	Apcupsd	—							—		UUtil les co mand 'ap-cupsd

Suite sur la p

TABLE 1.1 – Suite de la page précédente

Categorie	Contrôle	Transport			Protocole					Experimental	Comm
	SSH		TELNET	WSMAN	SNMP		HTTP		WMI		JMX
	Bluemind					—					Utilis l'API 'in- fluxdl
	Checkmyws					—					
	Elasticsearch					—					
	Exchange								—		Utilis un scr powe shell. Doit être i tallé Win- dows.
	Github					—					Utilis l'API 'gi- thub'.
	Hddtemp								—		Ouvre une conne TCP perso nalisé
	IIS						—				Doit être i tallé Win- dows.
			—			—			—		openv MIM

Suite sur la p

TABLE 1.1 – Suite de la page précédente

Categorie	Contrôle	Transport			Protocole					Experimental	Calm
	SSH		TELNET	WSMAN	SNMP		HTTP		WMI		JMX
Jenkins					—						JSON LWP rAgen URI, HTTI okies
Kayako					—					Utilise l'API 'kaya-ko'.	XML Di-gest : LWP rAgen URI, HTTI okies
Lmsensors				—							
Msmq								—	—	Doit être ins-tallé sur Win-dows. Pas encore déve-loppé.	
Nginx					—					Requiert le mo-dule 'HttpS-tubSta-tusMo-dule'.	LWP rAgen URI, HTTI okies
Pacemaker	—							—		Utilise la com-mande 'crm_mon'.	
Pfsense				—							
Selenium								—		Se connecte à un ser-veur Sele-nium pour jouer un scena-rio.	XML WWV leniu

Suite sur la p

TABLE 1.1 – Suite de la page précédente

Categorie	Contrôle	Transport			Protocole					Experimental	Comm
	SSH		TELNET	WSMAN	SNMP		HTTP		WMI		JMX
Tomcat					—					Requiert tomcat webmanager.	XML LWP rAgen URI, HTTP okies
Varnish	—							—		Utilise les commandes varnish.	
VMWare								—		Requiert le connecteur ‘centreon-vmware’ de Centreon.	
Pfsense				—							
	Bgp				—						
	Dhcp								—		
Protocoles	Dns								—		
	Ftp								—		
	Http					—					
	Ftp								—		
	Imap								—		

Suite sur la p

TABLE 1.1 – Suite de la page précédente

Categorie	Contrôle	Transport			Protocole					Experimental	Comm
	SSH		TELNET	WSMAN	SNMP		HTTP		WMI		JMX
	Jmx							—			
	Ldap								—		
	Ntp								—		
	Radius								—		
	Smtp								—		
	Tcp								—		
	Udp								—		
	x509								—		
	Informix								—		
Bases de données	Firebird								—		
	MS SQL								—		
	MySQL								—		
	Oracle								—		
	Postgres								—		
	ATS Apc				—					—	
	PDU Apc				—						

Suite sur la p

TABLE 1.1 – Suite de la page précédente

Categorie Contrôle	Transport			Protocole					Experimental	
	SSH	TELNET	WSMAN	SNMP		HTTP		WMI		JMX
	PDU Eaton			—					—	
	PDU Raritan			—						
	Standard Printers			—						
	Hwgste			—						
	Sensorip			—						
	Sensormetrix Em01				—					
	Serverscheck			—						
	Cisco UCS			—						
	Dell CMC			—						
	Dell iDrac			—						
	Dell Open-manage			—						Requi “l’age open-ma-nage” sur systèr d’ex-ploita tion.

Suite sur la p

TABLE 1.1 – Suite de la page précédente

Categorie Contrôle	Transport			Protocole					Experimental	Comm
	SSH	TELNET	WSMAN	SNMP		HTTP		WMI		JMX
	HP Proliant			—						Requi “l’age HP Insigh sur systèr d’ex- ploita tion.
	HP Blade Chassis			—						
	IBM BladeCenter			—						
	IBM HMC	—						—	—	
	IBM IMM			—						
	Sun hardware	—	—	—				—		Peut pervis plu- sieurs types matér Sun.
	UPS APC			—						
	UPS Mge			—						
	UPS Standard			—						
	UPS Powerware			—						
	3com			—						
	Alcatel Omniswitch			—						
	Arkoon			—						

Suite sur la p

TABLE 1.1 – Suite de la page précédente

Categorie Contrôle	Transport			Protocole					Experimental	
	SSH	TELNET	WSMAN	SNMP		HTTP		WMI		JMX
	Aruba			—						
	Bluecoat			—						
	Brocade			—						
	Checkpoint			—						
	Cisco			—						Many cisco (2800 Nexu Wlc, Iron- port,..
	Citrix Netsca- ler			—						
	Dell Power- connect			—						
	Dlink			—						
	Extreme			—						
	F5 Big- Ip			—						
	Fortinet Forti- gate			—						
	Fritzbox			—						
	H3C			—						
	Hirschmann			—						
	HP Pro- curve			—						

Suite sur la p

TABLE 1.1 – Suite de la page précédente

Categorie	Contrôle	Transport			Protocole					Experimental	Com
	SSH		TELNET	WSMAN	SNMP		HTTP		WMI		JMX
Systèmes	HP Virtual Connect				—						
	Juniper				—						Peut super viser ‘SSG ‘SA’, ‘SRX ‘MAG ‘EX’, ‘Ggsn
	Palo Alto				—						
	Netasq				—						
	Oneaccess				—						
	Radware Alteon				—					—	
	Redback				—						
	Riverbed				—						
	Ruggedcom				—						
	Securactive				—						
	Stonesoft				—						
	AIX	—							—		Utilis les co mand AIX.
	Freebsd				—		—				Utilis l’ager ‘bsnm

Suite sur la p

TABLE 1.1 – Suite de la page précédente

Categorie Contrôle	Transport			Protocole					Experimental	Comm
	SSH	TELNET	WSMAN	SNMP		HTTP		WMI		JMX
	Linux	—						—		Utilis les co mand Linux
			—							
Solaris	—						—		Utilise les commandes Solaris.	
					—					
Windows			—							
		—			—			—		
	Dell Equal-Logic			—						
	Dell MD3000							—		Requi la co mand ‘SMc
	Dell TL2000			—						
	Dell ML6000			—						
Stockage	EMC Celerra	—						—		Utilis les co mand de l’ pliand
	EMC Clariion							—		Requi la co mand ‘navis phere
	EMC Data-Domain			—						

Suite sur la p

TABLE 1.1 – Suite de la page précédente

Categorie	Contrôle	Transport			Protocole					Experimental	
	SSH		TELNET	WSMAN	SNMP		HTTP		WMI		JMX
	EMC Recove-rypoint	—							—		Utilis les co mand de l' pliand
	EMC Vplex						—				Utilis l'API JSON
	EMC Xtremio						—				Utilis l'API JSON
	Fujitsu Eternus DX	—							—		Utilis les co mand de l' pliand
	HP 3par	—							—		Utilis les co mand de l' pliand
	HP Lef-thand				—						
	HP MSA2000				—						
	HP p2000						—				Utilis l'API XML
	IBM DS3000								—		Utilis la co mand 'SMc
	IBM DS4000								—		Utilis la co mand 'SMc

Suite sur la p

TABLE 1.1 – Suite de la page précédente

Categorie	Contrôle	Transport			Protocole					Experimental	
	SSH		TELNET	WSMAN	SNMP		HTTP		WMI		JMX
	IBM DS5000								—		Utilis la co mand 'SMc
	IBM TS3100				—						
	IBM TS3200				—						
	Netapp				—						
	Nimble				—						
	Panzura				—						
	Qnap				—						
	Synology				—						
	Violin 3000				—						

1.4.2 Comment puis-je supprimer les données de performance ?

Par exemple, je vérifie les connexions TCP d'un serveur Linux par SNMP avec la commande suivante :

```
$ perl centreon_plugins.pl --plugin=os::linux::snmp::plugin --mode=tcpcon --hostname=127.0.0.1 --snmp
OK: Total connections: 1 | 'total'=1;;;0; 'con_closed'=0;;;0; 'con_closeWait'=0;;;0; '  con_synSent'=
```

Il y a trop de données de performances et je veux seulement garder la donnée de performance 'total'. J'utilise l'option `--filter-perfdata='total'` :

```
$ perl centreon_plugins.pl --plugin=os::linux::snmp::plugin --mode=tcpcon --hostname=127.0.0.1 --snmp
OK: Total connections: 1 | 'total'=1;;;0;
```

Je peux utiliser une expression régulière dans l'option `--filter-perfdata`. Donc je peux exclure les données de performance commençant par 'total' :

```
$ perl centreon_plugins.pl --plugin=os::linux::snmp::plugin --mode=tcpcon --hostname=127.0.0.1 --snmp
OK: Total connections: 1 | 'con_closed'=0;;;0; 'con_closeWait'=0;;;0; 'con_synSent'=0;;;0; 'con_estab
```

1.4.3 Comment puis-je ajuster un seuil : critique si valeur < X ?

“centreon-plugins” gère les seuils Nagios : https://nagios-plugins.org/doc/guidelines.html#THRESHOLDFORMAT
Par exemple, je veux vérifier que ‘crond’ fonctionne (s’il y a moins de 1 processus, critique). J’ai deux solutions :

```
$ perl centreon_plugins.pl --plugin=os::linux::snmp::plugin --mode=processcount --hostname=127.0.0.1
CRITICAL: Number of current processes running: 0 | 'nbproc'=0;;1:;0;

$ perl centreon_plugins.pl --plugin=os::linux::snmp::plugin --mode=processcount --hostname=127.0.0.1
CRITICAL: Number of current processes running: 0 | 'nbproc'=0;;@0:0;0;
```

1.4.4 Comment puis-je vérifier des équipements ipv6 en SNMP ?

Vous pouvez vérifier des équipements ipv6 en SNMP via la syntaxe suivante (`udp6:[xxxx]`) :

```
$ perl centreon_plugins.pl --plugin=os::linux::snmp::plugin --hostname='udp6:[fe80::250:56ff:feb5:6ae
```

1.4.5 Comment puis-je vérifier la valeur d’un OID SNMP générique ?

Il y a un plugin SNMP générique pour vérifier cela. Voici un exemple pour obtenir l’OID SNMP ‘SysUptime’ :

```
$ perl centreon_plugins.pl --plugin=apps::protocols::snmp::plugin --mode=numeric-value --oid='.1.3.6.
```

1.4.6 Comment utiliser un serveur memcached pour la rétention des données ?

Quelques plugins ont besoin de stocker des données. Il y a deux solutions pour cela :

- Un fichier sur le disque (par defaut).
- Un serveur memcached.

Pour utiliser ‘memcached’, vous devez avoir un serveur memcached et le module CPAN ‘Memcached : :libmemcached’ installé. Vous pouvez renseigner le serveur memcached avec l’option `--memcached` :

```
$ perl centreon_plugins.pl --plugin=os::linux::snmp::plugin --mode=interfaces --hostname=127.0.0.1 --
OK: All traffic are ok | 'traffic_in_lo'=197.40b/s;;;0;10000000 'traffic_out_lo'=197.40b/s;;;0;100000
Interface 'lo' Traffic In : 197.40b/s (0.00 %), Out : 197.40b/s (0.00 %)
Interface 'eth0' Traffic In : 14.54Kb/s (0.00 %), Out : 399.59b/s (0.00 %)
Interface 'eth1' Traffic In : 13.88Kb/s (0.00 %), Out : 1.69Kb/s (0.00 %)
```

Astuce : Un fichier local est utilisé si le serveur memcached ne répond pas.

1.4.7 Qu’est-ce que l’option `--dyn-mode` fait ?

Avec cette option, vous pouvez utiliser un mode avec un plugin. Cela est couramment utilisé pour les bases de données. Par exemple, j’ai une application qui stocke des informations de supervision dans une base de données. Le développeur peut utiliser un autre plugin pour créer le point de contrôle (pas besoin de faire les connexions SQL,... cela fait gagner du temps) :

```
$ perl centreon_plugins.pl --plugin=database::mysql::plugin --dyn-mode=apps::centreon::mysql::mode::p
OK: All poller delay for last update are ok | 'delay_Central'=2s;0:300;0:600;0; 'delay_Poller-Engine'
Delay for last update of Central is 2 seconds
Delay for last update of Poller-Engine is 2 seconds
```

Warning : Un mode utilisant le système suivant doit le notifier (dans l'aide associée). Donc vous devriez ouvrir un fichier dans éditeur et lire à la fin de la description.

1.4.8 Comment puis-je vérifier la version du plugin ?

Vous pouvez vérifier la version des plugins et des modes avec l'option `--version` :

```
$ perl centreon_plugins.pl --version
Global Version: 20160524
$ perl centreon_plugins.pl --plugin=os::linux::snmp::plugin --version
Plugin Version: 0.1
$ perl centreon_plugins.pl --plugin=os::linux::snmp::plugin --mode=storage --version
Mode Version: 1.0
```

Vous pouvez également utiliser l'option `--mode-version` pour exécuter le mode seulement s'il est dans la bonne version. Par exemple, nous voulons exécuter le mode seulement si sa version >= 2.x :

```
$  perl  centreon_plugins.pl --plugin=os::linux::snmp::plugin --mode=storage --hostname=127.0.0.1 --s
UNKNOWN: Not good version for plugin mode. Excepted at least: 2.x. Get: 1.0
```

1.4.9 Comment puis-je avoir un seul fichier Perl ?

Nous avons réalisé des tests et le temps d'éxecution est augmenté d'environ 4%. Nous allons créer un fichier unique pour la sonde Linux SNMP.

Télécharger le module Perl `App::FatPacker` sur metacpan :

```
# tar zxvf App-FatPacker-0.010005.tar.gz
# cd App-FatPacker-0.010005
# perl Makefile.PL && make && make install
```

Créer un répertoire de construction :

```
# mkdir -p build/plugin
# cd build
```

Cloner `centreon-plugins` :

```
# git clone https://github.com/centreon/centreon-plugins.git
```

`fatpack` inclut les fichiers `pm` présent dans le répertoire `lib` :

```
# mkdir plugin/lib && cd centreon-plugins
```

Copier les fichiers communs à l'ensemble des sondes :

```
# find . -name "*.pm" -exec sed -i ' /__END__/d' \{\} \;
# cp -R --parent centreon/plugins/{misc,mode,options,output,perfdata,script,statefile,values}.pm cent
# cp centreon_plugins.pl ../plugin
# sed -i 's/alternative_fatpacker = 0/alternative_fatpacker = 1/' ../plugin/lib/centreon/plugins/scri
```

Copier les fichiers pour la sonde Linux SNMP :

```
# cp -R --parent centreon/plugins/{script_snmp,snmp}.pm os/linux/snmp/ snmp_standard/mode/{cpu,cpudet
```

Construire le fichier Perl unique :

```
# cd ../plugin
# fatpack file centreon_plugins.pl > centreon_linux_snmp.pl
```

La sonde fonctionne de la même façon :

```
# perl centreon_linux_snmp.pl --plugin os::linux::snmp::plugin --mode=processcount --snmp-community p
```

1.4.10 Comment puis-je créer un binaire Windows ?

Cette procédure permet d'utiliser les sondes sans installer installer Perl sur le système Windows.

- Installer Strawberry Perl 5.24.x sur un Windows (Télécharger sur http://strawberryperl.com/)
- Récupérer la dernière version des centreon-plugins (Télécharger https://github.com/centreon/centreon-plugins/archive/master.zip)

Après les installations, installer le module `PAR::Packer` (remplacer `<PERL_INSTALL_DIR>`) :

```
cmd> <PERL_INSTALL_DIR>\perl\bin\cpan.bat
cpan> install PAR::Packer
```

L'installation peut prendre plusieurs minutes.

Dans un dossier contenant le répertoire `centreon-plugins`, créer un fichier `build.bat` (remplacer `<PERL_INSTALL_DIR>`). Nous excluons le module `IO::Socket::INET6` (Perl 5.14 intègre la fonctionnalité IPv6 en natif).

```
set PERL_INSTALL_DIR=<PERL_INSTALL_DIR>

chdir /d %~dp0
set PAR_VERBATIM=1

cmd /C %PERL_INSTALL_DIR%\perl\site\bin\pp --lib=centreon-plugins\ -o centreon_plugins.exe centreon-p
--unicode ^
-X IO::Socket::INET6 ^
--link=%PERL_INSTALL_DIR%\c\bin\libxml2-2__.dll ^
--link=%PERL_INSTALL_DIR%\c\bin\libiconv-2__.dll ^
--link=%PERL_INSTALL_DIR%\c\bin\liblzma-5__.dll ^
--link=%PERL_INSTALL_DIR%\c\bin\zlib1__.dll ^
-M Win32::Job ^
-M centreon::plugins::script ^
-M centreon::plugins::alternative::Getopt ^
-M apps::backup::netbackup::local::plugin ^
-M apps::backup::netbackup::local::mode::dedupstatus ^
-M apps::backup::netbackup::local::mode::drivecleaning ^
-M apps::backup::netbackup::local::mode::drivestatus ^
-M apps::backup::netbackup::local::mode::jobstatus ^
-M apps::backup::netbackup::local::mode::listpolicies ^
-M apps::backup::netbackup::local::mode::tapeusage ^
-M apps::activedirectory::local::plugin ^
-M apps::activedirectory::local::mode::dcdiag ^
-M apps::activedirectory::local::mode::netdom ^
-M apps::citrix::local::plugin ^
-M apps::citrix::local::mode::license ^
-M apps::citrix::local::mode::session ^
-M apps::citrix::local::mode::zone ^
-M apps::citrix::local::mode::folder ^
-M apps::iis::local::plugin ^
```

```
-M apps::iis::local::mode::listapplicationpools ^
-M apps::iis::local::mode::applicationpoolstate ^
-M apps::iis::local::mode::listsites ^
-M apps::iis::local::mode::webservicestatistics ^
-M apps::exchange::2010::local::plugin ^
-M apps::exchange::2010::local::mode::activesyncmailbox ^
-M apps::exchange::2010::local::mode::databases ^
-M apps::exchange::2010::local::mode::listdatabases ^
-M apps::exchange::2010::local::mode::imapmailbox ^
-M apps::exchange::2010::local::mode::mapimailbox ^
-M apps::exchange::2010::local::mode::outlookwebservices ^
-M apps::exchange::2010::local::mode::owamailbox ^
-M apps::exchange::2010::local::mode::queues ^
-M apps::exchange::2010::local::mode::replicationhealth ^
-M apps::exchange::2010::local::mode::services ^
-M centreon::common::powershell::exchange::2010::powershell ^
-M apps::cluster::mscs::local::plugin ^
-M apps::cluster::mscs::local::mode::listnodes ^
-M apps::cluster::mscs::local::mode::listresources ^
-M apps::cluster::mscs::local::mode::networkstatus ^
-M apps::cluster::mscs::local::mode::nodestatus ^
-M apps::cluster::mscs::local::mode::resourcestatus ^
-M apps::cluster::mscs::local::mode::resourcegroupstatus ^
-M os::windows::local::plugin ^
-M os::windows::local::mode::ntp ^
-M os::windows::local::mode::rdpsessions ^
-M storage::dell::compellent::local::plugin ^
-M storage::dell::compellent::local::mode::hbausage ^
-M storage::dell::compellent::local::mode::volumeusage ^
--verbose

pause
```

Lancer le fichier “build.bat” pour créer le binaire “centreon_plugins.exe”.

Pour changer la version et l’icône du binaire, ajouter le code suivant après `PERL_INSTALL_DIR` (première ligne) :

```
set ICO_FILE=centreon.ico
set RC_FILE=centreon.rc

chdir /d %~dp0

for /f "tokens=4 delims= " %%i in ('type centreon-plugins\centreon\plugins\script.pm ^| findstr globa
set VERSION_PLUGIN=%VERSION_PLUGIN:~0,8%

(
echo #define PP_MANIFEST_FILEFLAGS 0
echo #include ^<windows.h^>
echo.
echo CREATEPROCESS_MANIFEST_RESOURCE_ID RT_MANIFEST "winres\\pp.manifest"
echo.
echo VS_VERSION_INFO VERSIONINFO
echo    FILEVERSION        0,0,0,0
echo    PRODUCTVERSION     0,0,0,0
echo    FILEFLAGSMASK      VS_FFI_FILEFLAGSMASK
echo    FILEFLAGS          PP_MANIFEST_FILEFLAGS
echo    FILEOS             VOS_NT_WINDOWS32
echo    FILETYPE           VFT_APP
echo    FILESUBTYPE        VFT2_UNKNOWN
```

```
echo BEGIN
echo    BLOCK "StringFileInfo"
echo    BEGIN
echo        BLOCK "000004B0"
echo        BEGIN
echo            VALUE "CompanyName", "Centreon\0"
echo            VALUE "FileDescription", " \0"
echo            VALUE "FileVersion", "1.0.0.0\0"
echo            VALUE "InternalName", " \0"
echo            VALUE "LegalCopyright", " \0"
echo            VALUE "LegalTrademarks", " \0"
echo            VALUE "OriginalFilename", " \0"
echo            VALUE "ProductName", "centreon-plugins\0"
echo            VALUE "ProductVersion", "%VERSION_PLUGIN%.0\0"
echo        END
echo    END
echo    BLOCK "VarFileInfo"
echo    BEGIN
echo        VALUE "Translation", 0x00, 0x04B0
echo    END
echo END
echo.
echo WINEXE ICON winres\\pp.ico
)> %RC_FILE%

for /f "delims=" %%i in ('dir /ad /B %PERL_INSTALL_DIR%\cpan\build\PAR-Packer-*') do set "PAR_PACKER_
SET PAR_PACKER_SRC=%PERL_INSTALL_DIR%\cpan\build\%PAR_PACKER_DIRNAME%

copy /Y %ICO_FILE% %PAR_PACKER_SRC%\myldr\winres\pp.ico
copy /Y centreon.rc %PAR_PACKER_SRC%\myldr\winres\pp.rc
del %PAR_PACKER_SRC%\myldr\ppresource.coff
cd /D %PAR_PACKER_SRC%\myldr\ && perl Makefile.PL
cd /D %PAR_PACKER_SRC%\myldr\ && dmake boot.exe
cd /D %PAR_PACKER_SRC%\myldr\ && dmake Static.pm
attrib -R %PERL_INSTALL_DIR%\perl\site\lib\PAR\StrippedPARL\Static.pm
copy /Y %PAR_PACKER_SRC%\myldr\Static.pm %PERL_INSTALL_DIR%\perl\site\lib\PAR\StrippedPARL\Static.pm
```

Il est possible de compiler un binaire 32 bits depuis une version 64 bits Windows :

- Installer Strawberry Perl 5.24.x version 32 bits sur un Windows (Télécharger sur http://strawberryperl.com/)
- Installer le module cpan “PAR : :Packer”
- Ajouter la ligne suivante au début du script : PATH=%PERL_INSTALL_DIR%cbin ;%PERL_INSTALL_DIR%perlbin ;C :WindowsSystem32

1.5 Dépannage

1.5.1 SNMP

J’ai l’erreur SNMP : ‘UNKNOWN :.* (tooBig).*’

L’erreur suivante peut se produire avec certains équipements. Vous pouvez la résoudre si vous paramétrez les options suivantes :

- `--subsetleef=20 --maxrepetitions=20`

J'ai l'erreur SNMP : 'UNKNOWN :.*Timeout'

L'erreur suivante signifie :

- Pas d'accès réseau vers le serveur SNMP cible (un pare-feu peut bloquer le port UDP 161).
- La communauté ou la version SNMP paramétrées n'est pas correcte.

J'ai l'erreur SNMP : 'UNKNOWN :.*Cant get a single value'

L'erreur suivante signifie : l'accès SNMP fonctionne mais vous ne pouvez pas récupérer les valeurs SNMP. Plusieurs raisons possibles :

- La valeur SNMP n'est pas encore renseignée (peut se produire lorsqu'un serveur SNMP vient juste de démarrer).
- La valeur SNMP n'est pas implémentée par le constructeur.
- La valeur SNMP est renseignée sur un firmware ou OS spécifique.

Il semblerait que le contrôle de processus ne fonctionne pas avec certains filtres sur les arguments

Avec le SNMP, il y a une limite pour la longueur des arguments qui est fixée à 128 caractères. Donc, si vous essayez de filtrer avec un argument après 128 caractères, cela ne fonctionnera pas. Cela peut arriver avec les arguments Java. Pour résoudre le problème, vous devez privilégier le contrôle via NRPE.

Pas d'accès en SNMP v3

Tout d'abord, vous devez valider la connexion SNMP v3 avec snmpwalk. Lorsque cela fonctionne, vous renseignez les options SNMP v3 en ligne de commande. L'association entre les options 'snmpwalk' et les options "centreon-plugins" :

- -a => `--authprotocol`
- -A => `--authpassphrase`
- -u => `--snmp-username`
- -x => `--privprotocol`
- -X => `--privpassphrase`
- -l => non nécessaire (automatique)
- -e => `--securityengineid`
- -E => `--contextengineid`

1.5.2 Divers

J'ai l'erreur : "UNKNOWN : Need to specify '–custommode'."

Certains plugins ont besoin de renseigner l'option `--custommode`. Vous pouvez connaître la valeur à renseigner avec l'option `--list-custommode`. Un exemple :

```
$ perl centreon_plugins.pl --plugin=storage::ibm::DS3000::cli::plugin --list-custommode
...
Custom Modes Available:
 smcli

$ perl centreon_plugins.pl --plugin=storage::ibm::DS3000::cli::plugin --custommode=smcli --list-mode
```

J'ai l'erreur : "UNKNOWN : Cannot write statefile .*"

Vous devez créer le dossier (avec les droits d'écriture) pour autoriser le plugin à stocker certaines données sur le disque.

J'ai l'erreur : "UNKNOWN : Cannot load module 'xxx'."

Le problème peut être :
- Un module CPAN prérequis est manquant. Vous devez l'installer.
- Le module CPAN ne peut pas être chargé en raison de son chemin d'accès. Les modules Perl doivent être installés dans des chemins spécifiques.

Je ne peux pas vois les messages d'aide

Les fichiers "centreon-plugins" doivent être sous format Unix (pas de retour à la ligne Windows). Vous pouvez les modifier avec la commande suivante :

```
$ find . -name "*.p[ml]" -type f -exec dos2unix \{\} \;
```

Warning : Exécuter cette commande dans le dossier "centreon-plugins".

1.6 Examples de commandes

1.6.1 Windows

Contrôler tous les disques en SNMP

Dégradé si l'espace utilisé > 80% et critique sur l'espace utilisé > 90% :

```
$ perl centreon_plugins.pl --plugin=os::windows::snmp::plugin --mode=storage --hostname=xxx.xxx.xxx.x
OK: All storages are ok. | used_C:'=38623698944B;0:108796887040;0:122396497920;0;135996108800 used_D
Storage 'C:' Total: 126.66 GB Used: 35.97 GB (28.40%) Free: 90.69 GB (71.60%)
Storage 'D:' Total: 126.66 GB Used: 35.97 GB (28.40%) Free: 90.69 GB (71.60%)
```

Dégradé si l'espace disponible < 5G et critique si l'espace disponible < 2G :

```
$ perl centreon_plugins.pl --plugin=os::windows::snmp::plugin --mode=storage --hostname=xxx.xxx.xxx.x
OK: All storages are ok. | 'free_C:'=97372344320B;0:5497558138880;0:2199023255552;0;135996108800 'fre
Storage 'C:' Total: 126.66 GB Used: 35.97 GB (28.40%) Free: 90.69 GB (71.60%)
Storage 'D:' Total: 126.66 GB Used: 35.97 GB (28.40%) Free: 90.69 GB (71.60%)
```

1.6.2 Linux

Contrôler le trafic de toutes les interfaces en SNMP

Dégradé si le trafic entrant/sortant utilisé > 80% et critique si le trafic entrant/sortant utilisé > 90% :

```
$ perl centreon_plugins.pl --plugin=os::linux::snmp::plugin --mode=interfaces --hostname=127.0.0.1 --
OK: All traffic are ok | 'traffic_in_lo'=126.58b/s;0.00:8000000.00;0.00:9000000.00;0;10000000 'traffi
Interface 'lo' Traffic In : 126.58b/s (0.00 %), Out : 126.58b/s (0.00 %)
Interface 'eth0' Traffic In : 1.87Kb/s (0.00 %), Out : 266.32b/s (0.00 %)
Interface 'eth1' Traffic In : 976.65b/s (0.00 %), Out : 1.02Kb/s (0.00 %)
```

1.6.3 Protocole HTTP

Contrôler l'authentification à une application (requête POST)

Voici un exemple pour le formulaire d'authentification de `demo.centreon.com` :

```
$ perl centreon_plugins.pl --plugin=apps::protocols::http::plugin --mode=expected-content --hostname=
OK: 'color_UNREACHABLE' is present in content. | 'time'=0.575s;;;0; 'size'=20708B;;;0;
```

1.6.4 Protocole Modbus

Contrôler 3 registres holding

Le contenu du fichier `modbus.json` peut être spécifié directement dans l'option `--config` (exemple : `--config='{ "selection": { "metric1":{...'`). L'attribut `type` peut avoir les valeurs suivantes :

- holding (défaut)
- coils
- discrete
- input

```
{
    "selection":{
        "metric1":{
            "address": 1,
            "quantity": 1,
            "type": "holding",
            "display": true
        },
        "metric2":{
            "address": 2,
            "quantity": 1,
            "type": "holding",
            "display": true
        },
        "metric3":{
            "address": 3,
            "quantity": 1,
            "type": "holding",
            "display": true
        }
    }
}
```

Le résultat de la commande :

```
$ perl centreon_plugins.pl --plugin=apps/protocols/modbus/plugin.pm --mode=numeric-value --tcp-host=1
OK: All metrics are OK | 'metric1'=0;;;; 'metric2'=41291;;;; 'metric3'=42655;;;;
Metric 'metric1' value is '0'
```

```
Metric 'metric2' value is '41291'
Metric 'metric3' value is '42655'
```

Comment modifier la sortie ?

Il existe une section pour modifier la sortie globallement. Il est aussi possible de surcharger une métrique spécifiquement :

```
{
    "selection":{
        "metric1":{
            "address": 1,
            "quantity": 1,
            "type": "holding",
            "display": true
        },
        "metric2":{
            "address": 2,
            "quantity": 1,
            "type": "holding",
            "display": true
        },
        "metric3":{
            "address": 3,
            "quantity": 1,
            "type": "holding",
            "display": true,
            "formatting": {
                "printf_msg": "Override '%s' value is %.2f",
                "printf_var": "$self->{result_values}->{instance}, $self->{result_values}->{value}"
            }
        }
    },
    "formatting": {
        "printf_msg": "My metric '%s' value is %.2f",
        "printf_var": "$self->{result_values}->{instance}, $self->{result_values}->{value}"
    }
}
```

Le résultat de la commande :

```
$ perl centreon_plugins.pl --plugin=apps/protocols/modbus/plugin.pm --mode=numeric-value --tcp-host=1
OK: All metrics are OK | 'metric1'=0;;;; 'metric2'=41291;;;; 'metric3'=42655;;;;
My Metric 'metric1' value is 0.00
My Metric 'metric2' value is 41291.00
Override 'metric3' value is 42655.00
```

Comment moyenner 4 registres ?

Nous créons les valeurs moyennées suivantes : `[x1 + x2 / 2 = y] [x3 + x4 / 2 = z]`. Avec l'attribut `pattern`, il est possible de sélectionner les valeurs. Dans notre cas, nous récupérons 4 valeurs en 1 seule sélection. Les valeurs sélectionnées se nomment : `metrics.0`, `metrics.1`, `metrics.2`, `metrics.3` (order is preserved).

L'attribut `aggregation` peut avoir les valeurs suivantes :

- avg : retourne la moyenne des valeurs.
- sum : retourne la somme des valeurs.

— min : retourne la plus petite valeur numérique des valeurs.
— max : retourne la plus grande valeur numérique des valeurs.

```
{
    "selection":{
        "metrics":{
            "address": 1,
            "quantity": 4,
            "type": "holding",
            "display": false
        }
    },
    "virtualcurve":{
        "avg1":{
            "pattern": "metrics\\.[01]$",
            "aggregation": "avg",
            "unit": "con"
        },
        "avg2":{
            "pattern": "metrics\\.[23]$",
            "aggregation": "avg",
            "unit": "con"
        }
    }
}
```

Le résultat de la commande :

```
$ perl centreon_plugins.pl --plugin=apps/protocols/modbus/plugin.pm --mode=numeric-value --tcp-host=1
OK: Global metrics are OK | 'avg1'=42192con;;;; 'avg2'=40574con;;;;
Metric 'avg1' value is '42192'
Metric 'avg2' value is '40574'
```

Appliquer un calcul spécifique

L'attribut `custom` permet d'appliquer des modifications à la valeur :

```
{
    "selection":{
        "metrics":{
            "address": 1,
            "quantity": 4,
            "type": "holding",
            "display": false
        }
    },
    "virtualcurve":{
        "avg":{
            "aggregation": "avg",
            "custom": " / 10",
            "unit": "con"
        }
    }
}
```

Le résultat de la commande :

```
$ perl centreon_plugins.pl --plugin=apps/protocols/modbus/plugin.pm --mode=numeric-value --tcp-host=1
OK: Metric 'avg' value is '3072.3' | 'avg'=3072.3con;;;;
Metric 'avg' value is '3072.3'
```

1.6.5 Multi-service plugin

Ce mode permet de compiler/aggréger le résultat de plusieurs checks dans un seul. Il peut aussi être utilisé pour réaliser des aggregation dans des groupes logiques, il a été pensé pour récupérer au travers de réseaux bas débit des résultats de contrôle sur des Centreon distants, mais il peut aussi permettre d'éviter de checker deux fois les ressources dans deux Centreon différents.

Format du fichier de configuration

```
{
    "mode":"sqlmatching",
    "selection":{
        "ESX":{
            "host_name_filter":"%clus-esx-n%",
            "service_name_filter":"Esx-Status"
        },
        "XIVO":{
            "host_name_filter":"%xivo%",
            "service_name_filter":"Ping"
        }
    },
    "counters":{
        "totalservices":true,
        "totalhosts":true,
        "groups":true
    },
    "formatting":{
        "groups_global_msg":"Nothing special on groups",
        "host_service_separator":"/",
        "display_details":true
    }
}
```

- mode (obligatoire) : valeurs possibles : 'sqlmatching' or 'exactmatch'. Liés au format du bloc "selection" ;
- selection (obligatoire) : Lorsque le mode 'sqlmatching' est choisis, on va alors définir les filtres comme ci-dessus (host/service_name_filter). Au contraire, si l'on utilise le mode "exactmatch", alors on passe des clés valeurs correspondant à la correspondant host/service. (Exemple avec deux aggregation logqies "groups" esx-status/load ci-dessous)

```
"selection":{
    "esx-status":{
        "esx-n1":"Esx-Status",
        "esx-n2":"Esx-Status",
        "esx-n3":"Esx-Status"
    },
    "esx-load":{
        "esx-n1":"Esx-Memory",
        "esx-n2":"Esx-Memory",
        "esx-n3":"Esx-Memory",
        "esx-n1":"Esx-Cpu",
        "esx-n2":"Esx-Cpu",
```

```
        "esx-n3":"Esx-Cpu"
    }
},
```

— counters (optionnel) : Contiens trois booléens, à configurer en 'true' ou 'false' selon les compteurs que l'on veut utiliser et considérer (totalservices, totalhosts, groups).
— formatting (optionnel) : Contiens trois clés/valeurs, 'groups_global_msg' pour définir un statut global lorsque tout va bien, 'host_service_separator' pour choisir le séparateur entre le nom de l'hôte et celui du service dans les éléments de l'output, 'display_details' afin de définir si le plugin doit détailler les hôtes et/ou services en erreur dans l'output étendu (mode verbose)

Ligne de commande, output, seuils

Exemple de commande :

```
/usr/lib/nagios/plugins/centreon_plugins.pl --plugin database::mysql::plugin --dyn-mode apps::centreo
```

Example de sortie :

```
OK: Hosts state summary [up:4][down:2][unreachable:0] - Services state summary [ok:4][warning:0][crit
```

Données de performance :

```
'total_host_up'=4;;;0; 'total_host_down'=2;;;0; 'total_host_unreachable'=0;;;0; 'total_host_ok'=4;;;0
```

Mode verbeux (avec l'affichage détaillé d'activé) :

```
Group 'ESX': HOSTS: [up: 4 (clus-esx-n1.com - clus-esx-n2.com - clus-esx-n3.com - clus-esx-n4.com)][d
Group 'XIVO': HOSTS: [up: 0][down: 2 (srvi-xivo-n1 - srvi-xivo-n2)][unreachable: 0] - SERVICES: [ok:
```

Voici la manière de définir les seuils (total_statut pour les warning/critical-total et définition de l'instance et du décompte du nombre de statut pour les seuils par "groupes logiques") :

```
--critical-total '%{total_down} > 4' --critical-groups '%{instance} eq 'ESX' && %{unknown} > 5'
```

1.6.6 NSClient

Vous pouvez superviser des systèmes Windows/Linux via l'API Rest de NSClient. Les commandes et arguments sont les mêmes que via NRPE (veuillez lire la documentation NSClient pour plus d'informations) :

```
$ perl centreon_plugins.pl --plugin=apps::nsclient::restapi::plugin --mode=query --hostname="10.30.2
OK All 2 drive(s) are ok | '\\?\Volume{7cd2d555-9868-11e7-8199-806e6f6e6963}\ used'=289468416.000B;29
```

CHAPITRE 2

Guide de développeur

2.1 Description

Ce document présente les bonnes pratiques pour le développement de “centreon-plugins”. Comme tous les plugins sont écrits en Perl, “There’s more than one way to do it”. Afib de ne pas réinventer la roue, vous devriez d’abord regarder le dossier “example”. Vous aurez alors un aperçu de la méthodologie pour construire votre propre plugin ainsi que ses modes associés.

La dernière version est disponible sur le dépôt git suivant : https://github.com/centreon/centreon-plugins.git

2.2 Démarrage rapide

2.2.1 Création du dossier

Premièrement, vous avez besoin de créer un dossier sur le git afin de stocker le nouveau plugin.

Les répertoires racines sont oganisés par catégorie :
- Application : apps
- Base de données : database
- Matériel : hardware
- Equipement réseau : network
- Système d’exploitation : os
- Equipement de stockage : storage

Selon l’objet supervisé, il existe une organisation qui peut être utilisé :
- Type
- Constructeur
- Modèle
- Protocole de supervision

Par exemple, si vous voulez ajouter un plugin pour superviser Linux par SNMP, vous devez créer ce dossier :

```
$ mkdir -p os/linux/snmp
```

Vous avez également besoin de créer une répertoire “mode” pour les futurs modes créés :

```
$ mkdir os/linux/snmp/mode
```

2.2.2 Création du plugin

Une fois le dossier créé, ajouter le fichier du plugin à l'intérieur de celui-ci :

```
$ touch plugin.pm
```

Ensuite, éditer le fichier plugin.pm pour ajouter les **conditions de licence** en les copiant à partir d'un autre plugin. N'oubliez pas d'ajouter votre nom à la fin de celles-ci :

```
# ...
# Authors : <your name> <<your email>>
```

Renseigner votre nom de **package** : il correspond au dossier de votre plugin.

```
package path::to::plugin;
```

Déclarer les bibliothèques utilisées (**strict** et **warnings** sont obligatoires). Les bibliothèques Centreon sont décrites par la suite :

```
use strict;
use warnings;
use base qw(**centreon_library**);
```

Le plugin a besoin d'un constructeur **new** pour instancier l'objet :

```
sub new {
      my ($class, %options) = @_;
      my $self = $class->SUPER::new(package => __PACKAGE__, %options);
      bless $self, $class;

      ...

      return $self;
}
```

La version du plugin doit être déclarée dans le constructeur **new** :

```
$self->{version} = '0.1';
```

Plusieurs modes peuvent être déclarés dans le constructeur **new** :

```
%{$self->{modes}} = (
                      'mode1'     => '<plugin_path>::mode::mode1',
                      'mode2'     => '<plugin_path>::mode::mode2',
                      ...
                      );
```

Ensuite, déclarer le module :

```
1;
```

Une description du plugin est nécessaire pour générer la documentation :

```
__END__

=head1 PLUGIN DESCRIPTION

<Add a plugin description here>.

=cut
```

Astuce : Vous pouvez copier/coller les éléments d'un autre plugin et adapter les lignes (paquets, arguments, ...).

Astuce : Le plugin possède une extension ".pm" car c'est un module PERL. Par conséquent, ne pas oublier d'ajouter un **1 ;**.

2.2.3 Création du mode

Une fois que le fichier **plugin.pm** existe et que ses modes sont déclarés, créer les modes dans le dossier **mode** :

```
cd mode
touch mode1.pm
```

Ensuite, éditer mode1.pm pour ajouter les **conditions de licence** en les copiant à partir d'un autre mode. Ne pas oublier pas d'ajouter votre nom à la fin de celles-ci :

```
# ...
# Authors : <your name> <<your email>>
```

Décrire votre nom de **package** : il correspond au dossier de votre mode.

```
package path::to::plugin::mode::mode1;
```

Déclarer les bibliothèques utilisées (toujours les mêmes) :

```
use strict;
use warnings;
use base qw(centreon::plugins::mode);
```

Le mode nécessite un constructeur **new** pour instancier l'objet :

```
sub new {
      my ($class, %options) = @_;
      my $self = $class->SUPER::new(package => __PACKAGE__, %options);
      bless $self, $class;

      ...

      return $self;
}
```

La version du mode doit être déclarée dans le constructeur **new** :

```
$self->{version} = '1.0';
```

Plusieurs options peuvent être déclarées dans le constructeur **new** :

```
$options{options}->add_options(arguments =>
                               {
                                 "option1:s" => { name => 'option1' },
                                 "option2:s" => { name => 'option2', default => 'value1' },
                                 "option3"   => { name => 'option3' },
                               });
```

Voici la description des arguments de cet exemple :

— option1 : Chaîne de caractères

— option2 : Chaîne de caractères avec “value1” comme valeur par défaut
— option3 : Booléen

Astuce : Vous pouvez obtenir plus d’informations sur les formats des options ici : http://perldoc.perl.org/Getopt/Long.html

---

Le mode nécessite une méthode **check_options** pour valider les options :

```
sub check_options {
  my ($self, %options) = @_;
  $self->SUPER::init(%options);
  ...
}
```

Par exemple, les seuils Dégradé (Warning) et Critique (Critical) doivent être validés dans la méthode **check_options** :

```
if (($self->{perfdata}->threshold_validate(label => 'warning', value => $self->{option_results}->{wa
     $self->{output}->add_option_msg(short_msg => "Wrong warning threshold '" . $self->{option_result
     $self->{output}->option_exit();
  }
if (($self->{perfdata}->threshold_validate(label => 'critical', value => $self->{option_results}->{cr
     $self->{output}->add_option_msg(short_msg => "Wrong critical threshold '" . $self->{option_resul
     $self->{output}->option_exit();
}
```

Dans cet exemple, l’aide est affichée si les seuils spécifiés ne sont pas au bon format.

Ensuite vient la méthode **run** où sera effectué le traitement, vérifié les seuils et affichés le message de sortie et les données de performance. Voici un exemple pour vérifier une valeur SNMP :

```
sub run {
  my ($self, %options) = @_;
  $self->{snmp} = $options{snmp};
  $self->{hostname} = $self->{snmp}->get_hostname();

  my $result = $self->{snmp}->get_leef(oids => [$self->{option_results}->{oid}], nothing_quit => 1);
  my $value = $result->{$self->{option_results}->{oid}};

  my $exit = $self->{perfdata}->threshold_check(value => $value,
                               threshold => [ { label => 'critical', 'exit_litteral' => 'critical' },
  $self->{output}->output_add(severity => $exit,
                              short_msg => sprintf("SNMP Value is %s.", $value));

  $self->{output}->perfdata_add(label => 'value', unit => undef,
                                value => $value,
                                warning => $self->{perfdata}->get_perfdata_for_output(label => 'warni
                                critical => $self->{perfdata}->get_perfdata_for_output(label => 'crit
                                min => undef, max => undef);

  $self->{output}->display();
  $self->{output}->exit();
}
```

Dans cet exemple, un OID SNMP sera vérifié et comparé aux seuils Dégradé et Critique. Voici les méthodes utilisées :
— get_leef : pour obtenir une valeur snmp à partir d’un OID
— threshold_check : pour comparer une valeur snmp à des seuils dégradé et critique
— output_add : pour ajouter des informations au message de sortie
— perfdata_add : pour ajouter des données de performance au message de sortie
— display : pour afficher le message de sortie

— exit : pour sortir du programme

Ensuite, déclarer le module :

```
1;
```

Une description du mode et de ses arguments est nécessaire pour générer la documentation :

```
__END__

=head1 PLUGIN DESCRIPTION

<Add a plugin description here>.

=cut
```

2.2.4 Commit et push

Avant de commiter le plugin, vous devez créer un **ticket amélioration** (enhancement) dans la forge centreon-plugins : http://forge.centreon.com/projects/centreon-plugins

Une fois que le plugin et ses modes sont développés, vous pouvez commiter (messages de commit en anglais) et envoyer votre travail :

```
git add path/to/plugin
git commit -m "Add new plugin for XXXX refs #<ticked_id>"
git push
```

2.3 Référentiel des bibliothèques

Ce chapitre décrit les bibliothèques Centreon qui peuvent être utilisées dans votre développement.

2.3.1 Output

Cette bibliothèque vous permet de construire la sortie de votre plugin.

output_add

Description

Ajouter une chaîne de caractères à la sortie (affichée avec la méthode **display**). Si le statut est différent de 'OK', le message de sortie associé à 'OK' ne sera pas affiché.

Paramètres

Paramètre	Type	Défaut	Description
severity	String	OK	Statut du message de sortie.
separator	String	-	Séparateur entre le statut et le message de sortie.
short_msg	String		Message de sortie court (première ligne).
long_msg	String		Message de sortie long (utilisé avec l'option `--verbose`).

Exemple

Voici un exemple de gestion de la sortie du plugin :

```
$self->{output}->output_add(severity  => 'OK',
                            short_msg => 'All is ok');
$self->{output}->output_add(severity  => 'Critical',
                            short_msg => 'There is a critical problem');
$self->{output}->output_add(long_msg  => 'Port 1 is disconnected');

$self->{output}->display();
```

La sortie affichera :

```
CRITICAL - There is a critical problem
Port 1 is disconnected
```

perfdata_add

Description

Ajouter une donnée de performance à la sortie (affichée avec la méthode **display**). Les données de performance sont affichées après le symbole 'l'.

Paramètres

Paramètre	Type	Défaut	Description
label	String		Label de la donnée de performance.
value	Int		Valeur de la donnée de performance.
unit	String		Unité de la donnée de performance.
warning	String		Seuil Dégradé.
critical	String		Seuil Critique.
min	Int		Valeur minimum de la donnée de performance.
max	Int		Valeur maximum de la donnée de performance.

Exemple

Voici un exemple d'ajout d'une donnée de performance :

```
$self->{output}->output_add(severity  => 'OK',
                            short_msg => 'Memory is ok');
$self->{output}->perfdata_add(label    => 'memory_used',
                              value    => 30000000,
                              unit     => 'B',
                              warning  => '80000000',
                              critical => '90000000',
                              min      => 0,
                              max      => 100000000);

$self->{output}->display();
```

La sortie affichera :

```
OK - Memory is ok | 'memory_used'=30000000B;80000000;90000000;0;100000000
```

2.3.2 Perfdata

Cette bibliothèque vous permet de gérer les données de performance.

get_perfdata_for_output

Description

Gérer les seuils des données de performance pour la sortie.

Paramètres

Paramètre	Type	Défaut	Description
label	String		Label du seuil.
total	Int		Seuil en pourcentage à transformer en valeur globale.
cast_int	Int (0 or 1)		Cast une valeur absolue en entier.
op	String		Opérateur à appliquer à la valeur de début/fin (utilisé avec `value`).
value	Int		Valeur à appliquer avec l'option `op`.

Exemple

Voici un exemple de gestion des données de performance pour la sortie :

```
my $format_warning_perfdata  = $self->{perfdata}->get_perfdata_for_output(label => 'warning', total =
my $format_critical_perfdata = $self->{perfdata}->get_perfdata_for_output(label => 'critical', total

$self->{output}->perfdata_add(label    => 'memory_used',
                              value    => 30000000,
                              unit     => 'B',
                              warning  => $format_warning_perfdata,
                              critical => $format_critical_perfdata,
                              min      => 0,
                              max      => 1000000000);
```

Astuce : Dans cet exemple, au lieu d'afficher les seuils Dégradé et Critique en 'pourcentage', la fonction calculera et affichera ceux-ci en 'bytes'.

---

threshold_validate

Description

Valider et associer un seuil à un label.

Paramètres

Paramètre	Type	Défaut	Description
label	String		Label du seuil.
value	String		Valeur du seuil.

Exemple

Voici un exemple vérifiant si le seuil dégradé est correct :

```
if ((($self->{perfdata}->threshold_validate(label => 'warning', value => $self->{option_results}->{wa
  $self->{output}->add_option_msg(short_msg => "Wrong warning threshold '" . $self->{option_results}-
  $self->{output}->option_exit();
}
```

Astuce : Les bon formats de seuils sont consultables ici : https://nagios-plugins.org/doc/guidelines.html#THRESHOLDFORMAT

---

threshold_check

Description

Vérifier la valeur d'une donnée de performance avec un seuil pour déterminer son statut.

Paramètres

Paramètre	Type	Défaut	Description
value	Int		Valeur de la donnée de performance à comparer.
threshold	String array		Label du seuil à comparer et statut de sortie si celui-ci est atteint.

Exemple

Voici un exemple vérifiant si une donnée de performance a atteint certains seuils :

```
$self->{perfdata}->threshold_validate(label => 'warning', value => 80);
$self->{perfdata}->threshold_validate(label => 'critical', value => 90);
my $prct_used = 85;

my $exit = $self->{perfdata}->threshold_check(value => $prct_used, threshold => [ { label => 'critica

$self->{output}->output_add(severity  => $exit,
                            short_msg => sprint("Used memory is %i%%", $prct_used));
$self->{output}->display();
```

La sortie affichera :

```
WARNING - Used memory is 85% |
```

change_bytes

Description

Convertir des bytes en unité de mesure lisible. Retourner une valeur et une unité.

Paramètres

Paramètre	Type	Défaut	Description
value	Int		Valeur de données de performance à convertir.
network		1024	Unité de division (1000 si définie).

Exemple

Voici un exemple de conversion des bytes en unité de mesure lisible :

```
my ($value, $unit) = $self->{perfdata}->change_bytes(value => 100000);

print $value.' '.$unit."\n";
```

La sortie affichera :

```
100 KB
```

2.3.3 SNMP

Cette bibliothèque vous permet d'utiliser le protocole SNMP dans votre plugin. Pour l'utiliser, vous devez ajouter la ligne suivant au début de votre fichier **plugin.pm** :

```
use base qw(centreon::plugins::script_snmp);
```

get_leef

Description

Retourne une table de hashage de valeurs SNMP pour plusieurs OIDs (ne fonctionne pas avec les tables SNMP).

Paramètres

Paramètre	Type	Défaut	Description
oids	String array		Tableau d'OIDs à contrôler (Peut être spécifier avec la méthode `load`).
dont_quit	Int (0 or 1)	0	Ne pas quitter pas même si une erreur SNMP se produit.
nothing_quit	Int (0 or 1)	0	Quitter si aucune valeur n'est retournée.

Exemple

Voici un exemple pour récupérer 2 valeurs SNMP :

```
my $oid_hrSystemUptime = '.1.3.6.1.2.1.25.1.1.0';
my $oid_sysUpTime = '.1.3.6.1.2.1.1.3.0';

my $result = $self->{snmp}->get_leef(oids => [ $oid_hrSystemUptime, $oid_sysUpTime ], nothing_quit =>

print $result->{$oid_hrSystemUptime}."\n";
print $result->{$oid_sysUpTime}."\n";
```

load

Description

Charger une liste d'OIDs à utiliser avec la méthode **get_leef**.

Paramètres

Exemple

Voici un exemple pour obtenir les 4 premières instances d'une table SNMP en utilisant la méthode **load** :

```
my $oid_dskPath = '.1.3.6.1.4.1.2021.9.1.2';

$self->{snmp}->load(oids => [$oid_dskPercentNode], instances => [1,2,3,4]);

my $result = $self->{snmp}->get_leef(nothing_quit => 1);

use Data::Dumper;
print Dumper($result);
```

Voici un exemple pour obtenir plusieurs instances dynamiquement (modules mémoire de matériel Dell) en utilisant la méthode **load** :

```
my $oid_memoryDeviceStatus = '.1.3.6.1.4.1.674.10892.1.1100.50.1.5';
my $oid_memoryDeviceLocationName = '.1.3.6.1.4.1.674.10892.1.1100.50.1.8';
my $oid_memoryDeviceSize = '.1.3.6.1.4.1.674.10892.1.1100.50.1.14';
my $oid_memoryDeviceFailureModes = '.1.3.6.1.4.1.674.10892.1.1100.50.1.20';

my $result = $self->{snmp}->get_table(oid => $oid_memoryDeviceStatus);
$self->{snmp}->load(oids => [$oid_memoryDeviceLocationName, $oid_memoryDeviceSize, $oid_memoryDeviceF
                    instances => [keys %$result],
                    instance_regexp => '(\d+\.\d+)$');

my $result2 = $self->{snmp}->get_leef();

use Data::Dumper;
print Dumper($result2);
```

get_table

Description

Retourner une table de hashage de valeurs SNMP pour une table SNMP.

Paramètres

Paramètre	Type	Défaut	Description
oid	String		OID de la talbe SNMP à récupérer.
start	Int		Premier OID à récupérer.
end	Int		Dernier OID à récupérer.
dont_quit	Int (0 or 1)	0	Ne pas quitter même si une erreur SNMP se produit.
nothing_quit	Int (0 or 1)	0	Quitter si aucune valeur n'est retournée.
return_type	Int (0 or 1)	0	Retourner une table de hashage à un niveau au lieu de plusieurs.

Exemple

Voici un exemple pour récupérer une table SNMP :

```
my $oid_rcDeviceError            = '.1.3.6.1.4.1.15004.4.2.1';
my $oid_rcDeviceErrWatchdogReset = '.1.3.6.1.4.1.15004.4.2.1.2.0';

my $results = $self->{snmp}->get_table(oid => $oid_rcDeviceError, start => $oid_rcDeviceErrWatchdogRe

use Data::Dumper;
print Dumper($results);
```

get_multiple_table

Description

Retourner une table de hashage de valeurs SNMP pour plusieurs tables SNMP.

Paramètres

Paramètre	Type	Défaut	Description
oids	Hash table		Table de hashage des OIDs à récupérer (Peut être spécifié avec la méthode `load`). Les clés peuvent être : "oid", "start", "end".
dont_quit	Int (0 or 1)	0	Ne pas quitter même si une erreur snmp se produit.
nothing_quit	Int (0 or 1)	0	Quitter si aucune valeur n'est retournée.
return_type	Int (0 or 1)	0	Retourner une table de hashage à un niveau au lieu de plusieurs.

Exemple

Voici un exemple pour récupérer 2 tables SNMP :

```
my $oid_sysDescr        = ".1.3.6.1.2.1.1.1";
my $aix_swap_pool       = ".1.3.6.1.4.1.2.6.191.2.4.2.1";

my $results = $self->{snmp}->get_multiple_table(oids => [
                                                    { oid => $aix_swap_pool, start => 1 },
                                                    { oid => $oid_sysDescr },
                                                ]);

use Data::Dumper;
print Dumper($results);
```

get_hostname

Description

Récupérer le nom d'hôte en paramètre (utile pour obtenir le nom d'hôte dans un mode).

Paramètres

Aucun.

Exemple

Voici un exemple pour obtenir le nom d'hôte en paramètre :

```
my $hostname = $self->{snmp}->get_hostname();
```

get_port

Description

Récupérer le port en paramètre (utile pour obtenir le port dans un mode).

Parameters

Aucun.

Exemple

Voici un exemple pour obtenir le port en paramètre :

```
my $port = $self->{snmp}->get_port();
```

oid_lex_sort

Description

Retourner des OIDs triés.

Paramètres

Paramètre	Type	Défaut	Description
-	String array		Tableau d'OIDs à trier.

Exemple

Cet exemple affichera des OIDs triés :

```
foreach my $oid ($self->{snmp}->oid_lex_sort(keys %{$self->{results}->{$my_oid}})) {
  print $oid;
}
```

2.3.4 Misc

Cette bibliothèque fournit un ensemble de méthodes diverses. Pour l'utiliser, vous pouvez directement utiliser le chemin de la méthode :

```
centreon::plugins::misc::<my_method>;
```

trim

Description

Enlever les espaces de début et de fin d'une chaîne de caractères.

Paramètres

Exemple

Voici un exemple d'utilisation de la méthode **trim** :

```
my $word = '  Hello world !  ';
my $trim_word =  centreon::plugins::misc::trim($word);

print $word."\n";
print $trim_word."\n";
```

La sortie affichera :

```
Hello world !
```

change_seconds

Description

Convertir des secondes en unité de mesure lisible.

Paramètres

Paramètre	Type	Défaut	Description
-	Int		Nombre de secondes à convertir.

Exemple

Voici un exemple d'utilisation de la méthode **change_seconds** :

```
my $seconds = 3750;
my $human_readable_time =  centreon::plugins::misc::change_seconds($seconds);

print 'Human readable time : '.$human_readable_time."\n";
```

La sortie affichera :

```
Human readable time : 1h 2m 30s
```

backtick

Description

Exécuter une commande système.

Paramètres

Paramètre	Type	Défaut	Description
command	String		Commande à exécuter.
arguments	String array		Arguments de la commande.
timeout	Int	30	Timeout de la commande.
wait_exit	Int (0 or 1)	0	Le processus de la commande ignore les signaux SIGCHLD.
redirect_stderr	Int (0 or 1)	0	Afficher les erreurs dans la sortie.

Exemple

Voici un exemple d'utilisation de la méthode **backtick** :

```
my ($error, $stdout, $exit_code) = centreon::plugins::misc::backtick(
                                    command => 'ls /home',
                                    timeout => 5,
                                    wait_exit => 1
                                    );

print $stdout."\n";
```

La sortie affichera les fichiers du répertoire '/home'.

execute

Description

Exécuter une commande à distance.

Paramètres

Paramètre	Type	Défaut	Description
output	Object		Sortie du plugin ($self->{output}).
options	Object		Options du plugin ($self->{option_results}) pour obtenir les informations de connexion à distance.
sudo	String		Utiliser la commande sudo.
command	String		Commande à exécuter.
command_path	String		Chemin de la commande.
command_options	String		Arguments de la commande.

Exemple

Voici un exemple d'utilisation de la méthode **execute**. Nous supposons que l'option `--remote` soit activée :

```
my $stdout = centreon::plugins::misc::execute(output => $self->{output},
                                               options => $self->{option_results},
                                               sudo => 1,
                                               command => 'ls /home',
                                               command_path => '/bin/',
                                               command_options => '-l');
```

La sortie affichera les fichier du répertoire /home d'un hôte distant à travers une connexion SSH.

windows_execute

Description

Exécuter une commande sur Windows.

Paramètres

Paramètre	Type	Défaut	Description
output	Object		Sortie du plugin ($self->{output}).
command	String		Commande à exécuter.
command_path	String		Chemin de la commande.
command_options	String		Arguments de la commande.
timeout	Int		Timeout de la commande.
no_quit	Int		Ne pas quitter même si une erreur SNMP se produit.

Exemple

Voici un exemple d'utilisation de la méthode **windows_execute**.

```
my $stdout = centreon::plugins::misc::windows_execute(output => $self->{output},
                                                      timeout => 10,
                                                      command => 'ipconfig',
                                                      command_path => '',
                                                      command_options => '/all');
```

La sortie affichera la configuration IP d'un hôte Windows.

2.3.5 Statefile

Cette bibliothèque fournit un ensemble de méthodes pour utiliser un fichier de cache. Pour l'utiliser, ajouter la ligne suivante au début de votre **mode** :

```
use centreon::plugins::statefile;
```

read

Description

Lire un fichier de cache.

Paramètres

Paramètre	Type	Défaut	Description
statefile	String		Nom du fichier de cache.
statefile_dir	String		Répertoire du fichier de cache.
memcached	String		Serveur memcached à utiliser.

Exemple

Voici un exemple d'utilisation de la méthode **read** :

```
$self->{statefile_value} = centreon::plugins::statefile->new(%options);
$self->{statefile_value}->check_options(%options);
$self->{statefile_value}->read(statefile => 'my_cache_file',
                               statefile_dir => '/var/lib/centreon/centplugins'
                              );

use Data::Dumper;
print Dumper($self->{statefile_value});
```

La sortie affichera le fichier de cache et ses paramètres.

get

Description

Récupérer les données d'un fichier de cache.

Paramètres

Paramètre	Type	Défaut	Description
name	String		Récupérer une valeur du fichier de cache.

Exemple

Voici un exemple d'utilisation de la méthode **get** :

```
$self->{statefile_value} = centreon::plugins::statefile->new(%options);
$self->{statefile_value}->check_options(%options);
$self->{statefile_value}->read(statefile => 'my_cache_file',
                               statefile_dir => '/var/lib/centreon/centplugins'
                              );

my $value = $self->{statefile_value}->get(name => 'property1');
print $value."\n";
```

La sortie affichera la valeur associée à 'property1' du fichier de cache.

write

Description

Ecrire des données dans le fichier de cache.

Paramètres

Paramètre	Type	Défaut	Description
data	String		Données à écrire dans le fichier de cache.

Exemple

Voici un exemple d'utilisation de la méthode **write** :

```
$self->{statefile_value} = centreon::plugins::statefile->new(%options);
$self->{statefile_value}->check_options(%options);
$self->{statefile_value}->read(statefile => 'my_cache_file',
                               statefile_dir => '/var/lib/centreon/centplugins'
                              );

my $new_datas = {};
$new_datas->{last_timestamp} = time();
$self->{statefile_value}->write(data => $new_datas);
```

Ensuite, vous pouvez voir le résultat dans le fichier '/var/lib/centreon/centplugins/my_cache_file', le timestamp y est écrit.

2.3.6 HTTP

Cette bibliothèque fournit un ensemble de méthodes pour utiliser le protocole HTTP. Pour l'utiliser, ajouter la ligne suivante au début de votre **mode** :

```
use centreon::plugins::http;
```

Certaines options doivent être spécifiées dans **plugin.pm** :

Option	Type	Description
hostname	String	Adresse IP/FQDN du serveur web.
port	String	Port HTTP.
proto	String	Protocole utilisé ('HTTP' ou 'HTTPS').
credentials		Utiliser les informations d'authentification.
ntlm		Utiliser l'authentification NTLM (si `--credentials` est utilisée).
username	String	Nom d'utilisateur (si `--credentials` est utilisée).
password	String	Mot de passe (si `--credentials` est utilisée).
proxyurl	String	Proxy à utiliser.
url_path	String	URL à se connecter (commence par '/').

connect

Description

Tester la connexion vers une url HTTP. Retourner le contenu de la page web.

Paramètres

Cette méthode utilise les options du plugin précédemment définies.

Exemple

Voici un exemple d'utilisation de la méthode **connect**. Nous supposons que ces options sont définies : * –hostname = 'google.com' * –urlpath = '/' * –proto = 'http' * –port = 80

```
$self->{http} = centreon::plugins::http->new(output => $self->{output});
$self->{http}->set_options(%{$self->{option_results}});
my $webcontent = $self->{http}->request();
print $webcontent;
```

La sortie affichera le contenu de la page web 'http ://google.com/'.

2.3.7 DBI

Cette bibliothèque vous permet de vous connecter à une ou plusieurs bases de données. Pour l'utiliser, ajouter la ligne suivante au début de votre fichier **plugin.pm** :

```
use base qw(centreon::plugins::script_sql);
```

connect

Description

Se connecter à une ou plusieurs bases de données.

Paramètres

Paramètre	Type	Défaut	Description
dontquit	Int (0 or 1)	0	Ne pas quitter même si une erreur de connexion se produit.

Exemple

Voici un exemple d'utilisation de la méthode **connect**. Le format de la chaîne de connexion peut avoir les formes suivantes :

```
DriverName:database_name
DriverName:database_name@hostname:port
DriverName:database=database_name;host=hostname;port=port
```

Dans plugin.pm :

```
$self->{sqldefault}->{dbi} = ();
$self->{sqldefault}->{dbi} = { data_source => 'mysql:host=127.0.0.1;port=3306' };
```

Dans votre mode :

```
$self->{sql} = $options{sql};
my ($exit, $msg_error) = $self->{sql}->connect(dontquit => 1);
```

Vous êtes alors connecté à la base de données MySQL.

query

Description

Exécuter une requête SQL sur la base de données.

Paramètres

Paramètre	Type	Défaut	Description
query	String		Requête SQL à exécuter.

Exemple

Voici un exemple d'utilisation de la méthode **query** :

```
$self->{sql}->query(query => q{SHOW /*!50000 global */ STATUS LIKE 'Slow_queries'});
my ($name, $result) = $self->{sql}->fetchrow_array();

print 'Name : '.$name."\n";
print 'Value : '.$value."\n";
```

La sortie affichera le nombre de requêtes MySQL lentes.

fetchrow_array

Description

Retourner une tableau à partir d'une requête SQL.

Paramètres

Aucun.

Exemple

Voici un exemple d'utilisation de la méthode **fetchrow_array** :

```
$self->{sql}->query(query => q{SHOW /*!50000 global */ STATUS LIKE 'Uptime'});
my ($dummy, $result) = $self->{sql}->fetchrow_array();

print 'Uptime : '.$result."\n";
```

La sortie affichera l'uptime MySQL.

fetchall_arrayref

Description

Retourner un tableau à partir d'une requête SQL.

Paramètres

Aucun.

Exemple

Voici un exemple d'utilisation de la méthode **fetchrow_array** :

```
$self->{sql}->query(query => q{
      SELECT SUM(DECODE(name, 'physical reads', value, 0)),
          SUM(DECODE(name, 'physical reads direct', value, 0)),
          SUM(DECODE(name, 'physical reads direct (lob)', value, 0)),
          SUM(DECODE(name, 'session logical reads', value, 0))
      FROM sys.v_$sysstat
});
my $result = $self->{sql}->fetchall_arrayref();

my $physical_reads = @$result[0]->[0];
my $physical_reads_direct = @$result[0]->[1];
my $physical_reads_direct_lob = @$result[0]->[2];
my $session_logical_reads = @$result[0]->[3];

print $physical_reads."\n";
```

La sortie affichera les lectures physiques sur une base de données Oracle.

fetchrow_hashref

Description

Retourner une table de hashage à partir d'une requête SQL.

Paramètres

Aucun.

Exemple

Voici un exemple d'utilisation de la méthode **fetchrow_hashref** :

```
$self->{sql}->query(query => q{
  SELECT datname FROM pg_database
});

while ((my $row = $self->{sql}->fetchrow_hashref())) {
  print $row->{datname}."\n";
}
```

La sortie affichera la liste des bases de données PostgreSQL.

2.4 Exemples complets

2.4.1 Requête SNMP simple

Description

Cet exemple explique comment vérifier une valeur SNMP unique sur un pare-feu PfSense (paquets supprimés pour cause de surcharge mémoire).
Un fichier de cache sera utilisé car c'est un compteur SNMP. Il est nécessaire d'obtenir la valeur différentielle entre 2 contrôles.
La valeur récupérée sera comparée aux seuils Dégradé et Critique.

Fichier du plugin

Tout d'abord, créer le dossier du plugin, ainsi que le fichier du plugin :

```
$ mkdir -p apps/pfsense/snmp
$ touch apps/pfsense/snmp/plugin.pm
```

Astuce : PfSense est un pare-feu applicatif et il sera contrôler en utilisant le protocole SNMP

Ensuite, éditer le fichier **plugin.pm** et ajouter les lignes suivantes :

```
#
# Copyright 2018 Centreon (http://www.centreon.com/)
#
# Centreon is a full-fledged industry-strength solution that meets
# the needs in IT infrastructure and application monitoring for
# service performance.
#
# Licensed under the Apache License, Version 2.0 (the "License");
# you may not use this file except in compliance with the License.
# You may obtain a copy of the License at
#
#     http://www.apache.org/licenses/LICENSE-2.0
#
# Unless required by applicable law or agreed to in writing, software
# distributed under the License is distributed on an "AS IS" BASIS,
# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
# See the License for the specific language governing permissions and
# limitations under the License.
#

# Chemin vers le plugin
package apps::pfsense::snmp::plugin;

# Bibliothèques nécessaires
use strict;
use warnings;
# Utiliser cette bibliothèque pour contrôle en utilisant le protocole SNMP
use base qw(centreon::plugins::script_snmp);
```

Astuce : N'oublier pas de modifier la ligne 'Authors'.

Ajouter la méthode **new** pour instancier le plugin :

```
sub new {
  my ($class, %options) = @_;
  my $self = $class->SUPER::new(package => __PACKAGE__, %options);
  bless $self, $class;
  # $options->{options} = options object

  # Version du plugin
  $self->{version} = '0.1';

  # Association des modes
  %{$self->{modes}} = (
                       # Nom du mode => Chemin vers le mode
                       'memory-dropped-packets'    => 'apps::pfsense::snmp::mode::memorydroppedpackets
                       );

  return $self;
}
```

Déclarer ce plugin en tant que module perl :

```
1;
```

Ajouter une description au plugin :

```
__END__

=head1 PLUGIN DESCRIPTION

Check pfSense in SNMP.

=cut
```

Astuce : Cette description est affichée avec l'option `--help`.

Fichier du mode

Ensuite, créer le répertoire du mode, ainsi que le fichier du mode :

```
$ mkdir apps/pfsense/snmp/mode
$ touch apps/pfsense/snmp/mode/memorydroppedpackets.pm
```

Editer le fichier **memorydroppedpackets.pm** et ajouter les lignes suivantes :

```
#
# Copyright 2018 Centreon (http://www.centreon.com/)
#
# Centreon is a full-fledged industry-strength solution that meets
# the needs in IT infrastructure and application monitoring for
# service performance.
#
# Licensed under the Apache License, Version 2.0 (the "License");
# you may not use this file except in compliance with the License.
# You may obtain a copy of the License at
#
#     http://www.apache.org/licenses/LICENSE-2.0
```

```
#
# Unless required by applicable law or agreed to in writing, software
# distributed under the License is distributed on an "AS IS" BASIS,
# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
# See the License for the specific language governing permissions and
# limitations under the License.
#

# Chemin vers le mode
package apps::pfsense::snmp::mode::memorydroppedpackets;

# Bibliothèque nécessaire pour le mode
use base qw(centreon::plugins::mode);

# Bibliothèques nécessaires
use strict;
use warnings;

# Bibliothèque nécessaire pour certaines fonctions
use POSIX;

# Bibliothèque nécessaire pour utiliser un fichier de cache
use centreon::plugins::statefile;
```

Ajouter la méthode **new** pour instancier le mode :

```
sub new {
  my ($class, %options) = @_;
  my $self = $class->SUPER::new(package => __PACKAGE__, %options);
  bless $self, $class;

  # Version du mode
  $self->{version} = '1.0';

  # Declaration des options
  $options{options}->add_options(arguments =>
                                {
                                  # nom de l'option    => nom de la variable
                                  "warning:s"          => { name => 'warning', },
                                  "critical:s"         => { name => 'critical', },
                                });

  # Instanciation du fichier de cache
  $self->{statefile_value} = centreon::plugins::statefile->new(%options);
  return $self;
}
```

Astuce : Une valeur par défaut peut être ajoutée aux options. Exemple : "warning :s" => { name => 'warning', default => '80' },

---

Ajouter la méthode **check_options** pour valider les options :

```
sub check_options {
  my ($self, %options) = @_;
  $self->SUPER::init(%options);

  # Validation des options de seuil avec la méthode threshold_validate
  if (($self->{perfdata}->threshold_validate(label => 'warning', value => $self->{option_results}->{w
```

```
        $self->{output}->add_option_msg(short_msg => "Wrong warning threshold '" . $self->{option_result
        $self->{output}->option_exit();
    }
    if (($self->{perfdata}->threshold_validate(label => 'critical', value => $self->{option_results}->
        $self->{output}->add_option_msg(short_msg => "Wrong critical threshold '" . $self->{option_resul
        $self->{output}->option_exit();
    }

    # Validation des options de fichier de cache en utilisant la méthode check_options de la bibliothè
    $self->{statefile_value}->check_options(%options);
}
```

Ajouter la méthode **run** pour exécuter le mode :

```
sub run {
    my ($self, %options) = @_;
    # $options{snmp} = snmp object

    # Récupération des options SNMP
    $self->{snmp} = $options{snmp};
    $self->{hostname} = $self->{snmp}->get_hostname();
    $self->{snmp_port} = $self->{snmp}->get_port();

    # oid SNMP à requêter
    my $oid_pfsenseMemDropPackets = '.1.3.6.1.4.1.12325.1.200.1.2.6.0';
    my ($result, $value);

    # Récupération de la valeur SNMP pour l'oid précédemment défini
    $result = $self->{snmp}->get_leef(oids => [ $oid_pfsenseMemDropPackets ], nothing_quit => 1);
    # $result est une table de hashage où les clés sont les oids
    $value = $result->{$oid_pfsenseMemDropPackets};

    # Lecture du fichier de cache
    $self->{statefile_value}->read(statefile => 'pfsense_' . $self->{hostname}  . '_' . $self->{snmp_po
    # Lecture des valeurs du fichier de cache
    my $old_timestamp = $self->{statefile_value}->get(name => 'last_timestamp');
    my $old_memDropPackets = $self->{statefile_value}->get(name => 'memDropPackets');

    # Création d'une table de hashage avec les nouvelles valeurs qui seront écrites dans le fichier de
    my $new_datas = {};
    $new_datas->{last_timestamp} = time();
    $new_datas->{memDropPackets} = $value;

    # Ecriture des nouvelles valeurs dans le fichier de cache
    $self->{statefile_value}->write(data => $new_datas);

    # Si le fichier de cache ne possède aucune valeur, nous les créons et attendons un nouveau contrôle
    if (!defined($old_timestamp) || !defined($old_memDropPackets)) {
        $self->{output}->output_add(severity => 'OK',
                                    short_msg => "Buffer creation...");
        $self->{output}->display();
        $self->{output}->exit();
    }

    # Correctif lorsque PfSense redémarre (les compteurs snmp sont réinitialisés à 0)
    $old_memDropPackets = 0 if ($old_memDropPackets > $new_datas->{memDropPackets});

    # Calcul de l'intervalle de temps entre 2 contrôles
```

```
  my $delta_time = $new_datas->{last_timestamp} - $old_timestamp;
  $delta_time = 1 if ($delta_time == 0);

  # Calcul de la valeur par seconde
  my $memDropPacketsPerSec = ($new_datas->{memDropPackets} - $old_memDropPackets) / $delta_time;

  # Calcul le code de retour en comparant la valeur aux seuils
  # Le code de retour peut être : 'OK', 'WARNING', 'CRITICAL', 'UNKNOWN'
  my $exit_code = $self->{perfdata}->threshold_check(value => $memDropPacketsPerSec,
                                                     threshold => [ { label => 'critical', 'exit_litt

  # Ajout d'une donnée de performance
  $self->{output}->perfdata_add(label => 'dropped_packets_Per_Sec',
                                value => sprintf("%.2f", $memDropPacketsPerSec),
                                warning => $self->{perfdata}->get_perfdata_for_output(label => 'warni
                                critical => $self->{perfdata}->get_perfdata_for_output(label => 'crit
                                min => 0);

  # Ajout du message de sortie
  $self->{output}->output_add(severity => $exit_code,
                              short_msg => sprintf("Dropped packets due to memory limitations : %.2f
                                  $memDropPacketsPerSec));

  # Affichage du message de sortie
  $self->{output}->display();
  $self->{output}->exit();
}
```

Déclarer ce mode comme un module perl :

```
1;
```

Ajouter une description aux options du mode :

```
__END__

=head1 MODE

Check number of packets per second dropped due to memory limitations.

=over 8

=item B<--warning>

Threshold warning for dropped packets in packets per second.

=item B<--critical>

Threshold critical for dropped packets in packets per second.

=back

=cut
```

Ligne de commande

Voici un exemple de ligne de commande :

```
$ perl centreon_plugins.pl --plugin apps::pfsense::snmp::plugin --mode memory-dropped-packets --hostn
```

La sortie pourrait afficher :

```
OK: Dropped packets due to memory limitations : 0.00 /s | dropped_packets_Per_Sec=0.00;0;;1;2
```