
*Supervision avec
NAGIOS4 Mise en
place*

I. Prérequis

On utilisera une Debian 10 hébergeant un serveur LAMP

On suppose qu'elle est connectée au réseau, à jour et adressée sur une IP fixe

II. Installation du service

Pour installer NAGIOS4 il faut installer plusieurs paquets dont :

- nagios4
- nagios-plugins-contrib
- nagios-nrpe-plugin

Activez les modules apache2 : `a2enmod auth_digest` et `a2enmod authz_groupfile`.

III. Accès au service

Pour se connecter au service de NAGIOS on utilise une interface WEB accessible en entrant `IPserv/nagios4`

IV. Supervision de base

a. Configuration des hôtes

Pour que Nagios supervise nos machines il nous faut les lui définir, pour cela on crée un fichier `host.cfg` dans le répertoire `/etc/nagios4/conf.d/` :

```
define host{
    use linux-server
    host_name DHCP-DNS-Linux
    alias DHCP-DNS-Linux
    address 172.17.1.5
}
```

```
define host{
    use linux-server
    host_name OpenVPN
    alias OpenVPN
    address 172.17.1.15
}
```

```
define host{
    use linux-server
    host_name Windows-MM
    alias Windows-MM
}
```

```

        address 172.17.1.7
    }

    define host{
        use linux-server
        host_name AD-DNS
        alias AD-DNS
        address 172.17.1.9
    }

    define host{
        use linux-server
        host_name Windows-AB
        alias Windows-AB
        address 172.17.1.6
    }

    define host{
        use linux-server
        host_name All-Machine
        alias All-Machine
        address 172.17.1.14
    }
}

define host{
    use linux-server
    host_name Switch-Aruba
    alias Switch-Aruba
    address 172.17.1.2
}

define host{
    use linux-server
    host_name Switch-Cisco
    alias Switch-Cisco
    address 172.17.1.3
}

define host{
    use linux-server
    host_name Router-Cisco
    alias Router-Cisco
    address 172.17.1.1
}

```

On recommence cette opération pour chaque machine à superviser

b. Configuration des groupes

Maintenant que nos hôtes sont définis on peut les regrouper par groupe pour effectuer des actions sur les groupes au lieu de passer sur chaque hôte. On peut définir les groupes dans le même fichier que les hôtes, mais par soucis de lisibilité nous allons les définir dans un autre fichier

BookticGroupes.cfg :

```

define hostgroup{
    hostgroup_name Machine-Physique
    alias Machine-Physique
    members All-Machine,Windows-AB,Windows-MM
}

define hostgroup{
    hostgroup_name Machine-Interconnexion
    alias Machine-Interconnexion
    members Router-Cisco,Switch-Cisco,Switch-Aruba,Switch-ArubaWifi
}

define hostgroup{
    hostgroup_name Serveur
    alias Serveur
members DHCP-DNS-Linux,AD-DNS,SRVnagios,OpenVPN,ReverseProxy,OwnCloud,GLPI
}

define hostgroup{
    hostgroup_name Machine-Windows
    alias Machine-Windows
    members All-Machine,Windows-AB,Windows-MM,AD-DNS
}



define hostgroup{
    hostgroup_name Machine-Linux
    alias Machine-Linux
members DHCP-DNS Linux,SRVnagios,OpenVPN,ReverseProxy,OwnCloud,GLPI
}

```

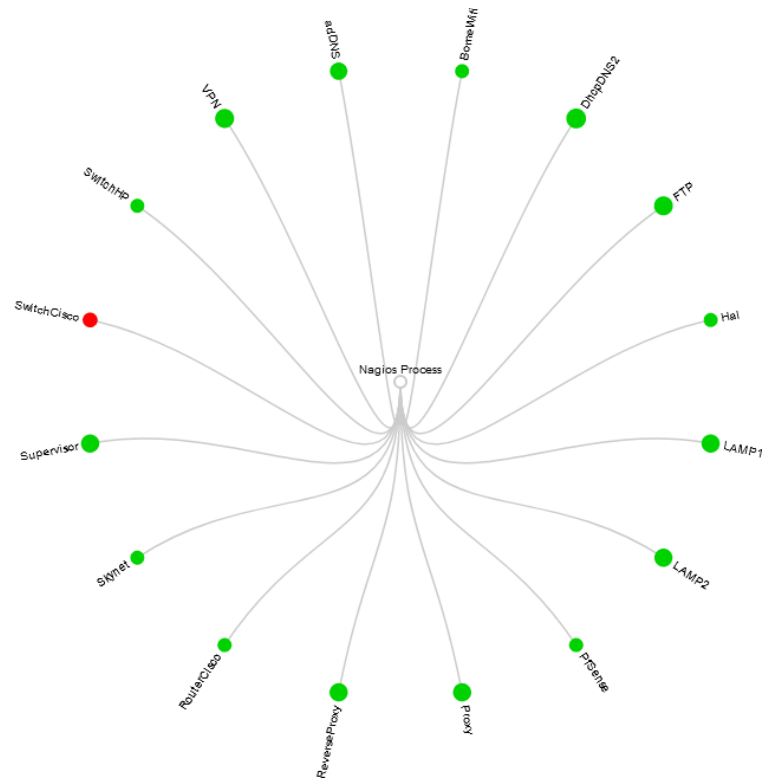
Dans notre cas nous allons créer un groupe pour les serveurs Linux, un pour les Windows et un pour le matériel d'interconnexion.

c. Résultat sur l'interface web

Si on consulte l'onglet Hosts on peut voir que nos machines sont répertoriées, Nagios va alors tenter un ping vers chacune d'elles et affichera leur statut.

Host ♦♦	Status ♦♦	Last Check ♦♦	Duration ♦♦	Status Information
AD-DNS	 UP	03-10-2022 10:34:55	2d 0h 34m 25s	PING OK - Paquets perdus = 0%, RTA = 2.02 ms
All-Machine	 UP	03-10-2022 10:35:25	2d 0h 34m 25s	PING OK - Paquets perdus = 0%, RTA = 2.13 ms
DHCP-DNS-Linux	 UP	03-10-2022 10:35:55	0d 22h 8m 58s	PING OK - Paquets perdus = 0%, RTA = 1.79 ms
OpenVPN	 UP	03-10-2022 10:36:25	0d 22h 11m 34s	PING OK - Paquets perdus = 0%, RTA = 2.19 ms
Router-Cisco	 UP	03-10-2022 10:36:55	91d 22h 34m 42s	PING OK - Paquets perdus = 0%, RTA = 1.56 ms
SRVnagios	 UP	03-10-2022 10:37:25	98d 22h 32m 9s	PING OK - Paquets perdus = 0%, RTA = 0.04 ms
Switch-Aruba	 UP	03-10-2022 10:37:55	91d 22h 34m 42s	PING OK - Paquets perdus = 0%, RTA = 2.37 ms
Switch-Cisco	 UP	03-10-2022 10:38:25	91d 22h 24m 19s	PING OK - Paquets perdus = 0%, RTA = 1.43 ms
Windows-AB	 UP	03-10-2022 10:38:55	0d 22h 10m 5s	PING OK - Paquets perdus = 0%, RTA = 1.90 ms
Windows-MM	 UP	03-10-2022 10:34:25	0d 22h 9m 43s	PING OK - Paquets perdus = 0%, RTA = 0.54 ms

En plus de cet affichage Nagios propose une carte réseau qu'il génère lui-même et où sont présents toutes les machines, néanmoins elle ressemble plus à un fouillis qu'à un réseau organisé et lisible :



V. Amélioration

a. Rendra la carte plus digeste

Comme visible sur l'image précédente toutes nos machines sont liées au *Nagios Process*, cependant il serait plus pratique si les machines connectées au SwitchCisco apparaissaient également derrière ce Switch sur notre carte.

Fort heureusement Nagios intègre un paramètre parents dans la définition de nos hôtes, dans lequel on précise le nom de l'hôte parent. Reprenons notre serveur DHCPdns2, il est virtualisé sur Hal qui est connecté au SwitchCisco.

On modifie donc le fichier host.cfg comme dans l'exemple qui suit :

En appliquant cette configuration à l'intégralité de nos hôtes et en relance le processus de Nagios on obtient une carte un peu plus ordonnée :

```
define host {  
    use linux-server  
    host_name DHCP-DNS-Linux  
    alias DHCP-DNS-Linux  
    address 172.17.1.5  
    parents Windows-AB  
}
```

```
define host {  
    use linux-server  
    host_name OpenVPN  
    alias OpenVPN  
    address 172.17.1.15  
    parents All-Machine  
}
```

```
define host {  
    use linux-server  
    host_name Windows-MM  
    alias Windows-MM  
    address 172.17.1.7  
    parents Switch-Cisco  
}
```

```
define host {  
    use linux-server  
    host_name AD-DNS  
    alias AD-DNS  
    address 172.17.1.9  
    parents All-Machine  
}
```

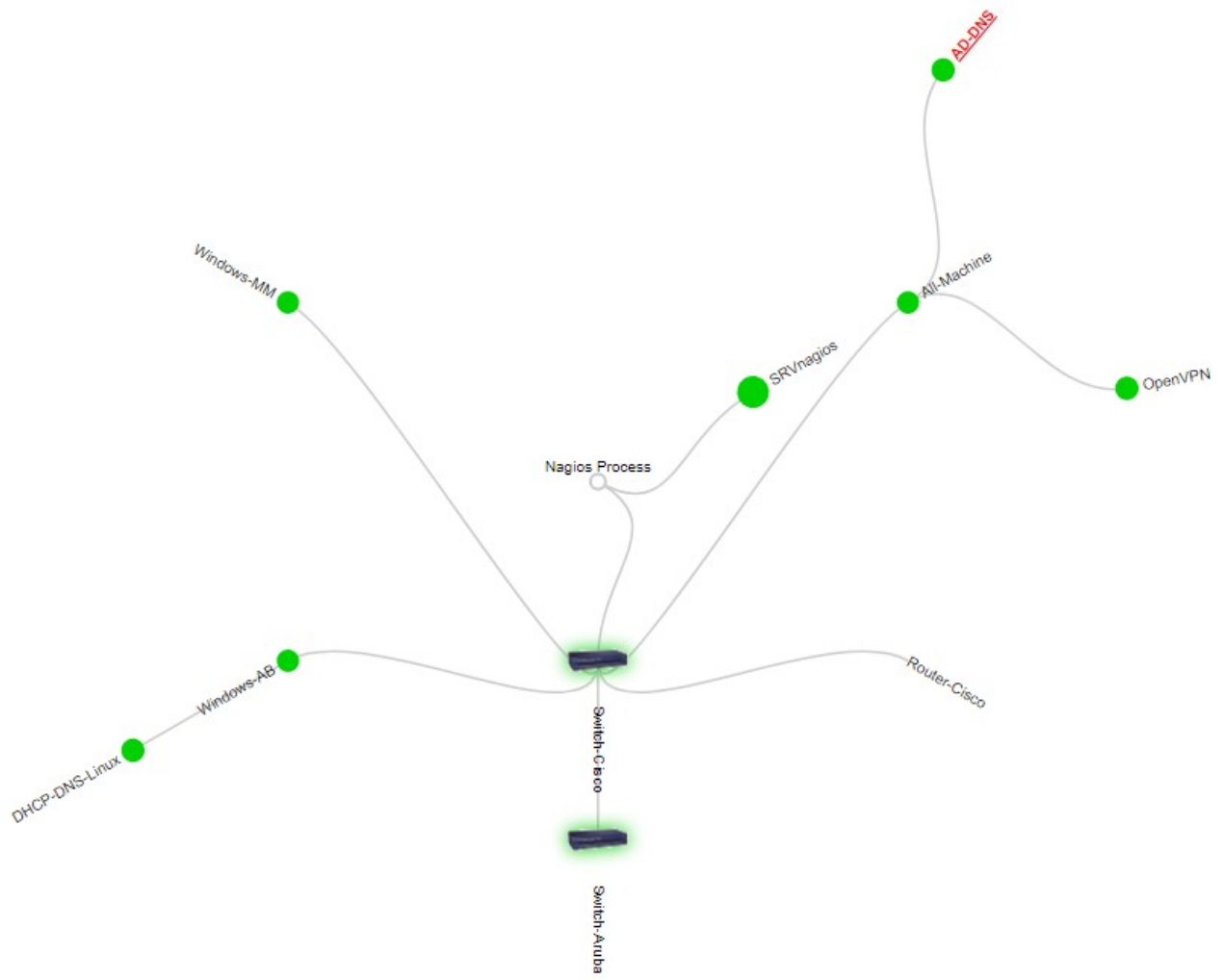
```
define host {  
    use linux-server  
    host_name Windows-AB  
    alias Windows-AB  
    address 172.17.1.6  
    parents Switch-Cisco  
}
```

```
define host {  
    use linux-server  
    host_name All-Machine  
    alias All-Machine  
    address 172.17.1.14  
    parents Switch-Cisco  
}
```

```
define host {  
    use linux-server  
    host_name Switch-Aruba  
    alias Switch-Aruba  
    address 172.17.1.2  
    parents Switch-Cisco  
}
```

```
define host {  
    use linux-server  
    host_name Switch-Cisco  
    alias Switch-Cisco  
    address 172.17.1.3  
}
```

```
define host {  
    use linux-server  
    host_name Router-Cisco  
    alias Router-Cisco  
    address 172.17.1.1  
    parents Switch-Cisco  
}
```



b. Ajouter une image aux hôtes

Avoir une carte organisée c'est bien, mettre une image sur chaque machine c'est mieux. Pour facilement reconnaître le type de machine, l'OS qui la fait tourner ou bien le service principal on peut ajouter un icône pour la carte ainsi que pour la liste des hôtes.

Il nous faut donc une image qui respecte quelques critères :

- Format PNG
- Taille de 40 x 40 pixels

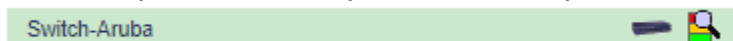
On ajoute nos images dans le répertoire
/usr/share/nagios4/htdocs/images/logos/

En réalité il faut 2 versions de cette image, l'une en .png pour la carte et la seconde en .gd2 pour le statut de la machine et des services. Le paquet libgd-tools permet de faire la conversion.

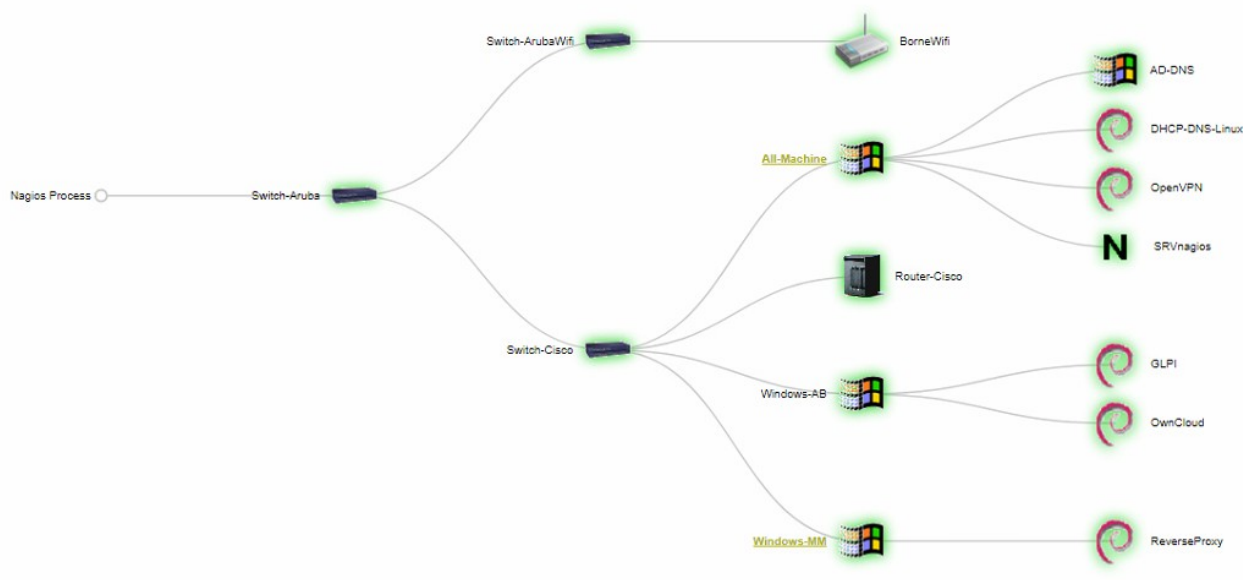
Enfin on modifie le fichier host.cfg en ajoutant deux nouvelles options :

```
define host {  
    use linux-server  
    host_name Switch-Aruba  
    alias Switch-Aruba  
    address 172.17.1.2  
    parents Switch-Cisco  
    icon_image switch40.png  
    statusmap_image switch40.gd2  
}
```

Après un redémarrage de Nagios on peut observer dans l'onglet Hosts que nos hôtes sont pourvus d'une petite icône en plus de leur nom :



Et sur la carte du réseau les images avec un halo de couleur remplacent les points pour représenter les hôtes :



VI. Supervision avancée

Pour le moment la seule supervision que nous apporte Nagios est de savoir si les hôtes sont accessibles, nous ne savons donc rien des services qu'ils proposent ou de leurs statuts matériels. On va donc y remédier en configurant une supervision avancée.

a. Préparation

1) Serveurs Linux

Pour que Nagios récupère des informations sur nos machines il faut installer un agent ncpa disponible sur le site de nagios :

wget <https://assets.nagios.com/downloads/ncpa/ncpa-listener.d10.amd64.deb>

Et on l'installe via `dpkg -i ncpa-listener.d10.amd64.deb`

Une fois l'installation terminée on doit indiquer le nom de communauté qui servira de token, dans `/usr/local/ncpa/etc/ncpa.cfg`, et on modifie la ligne `community_string` en `community_string = siojrr`

2) Serveurs Windows

Concernant les machines windows le fonctionnement est quasi-identique, on récupère l'installateur sur le site de Nagios et on l'installe. On laisse le paramétrage par défaut, à l'exception du community string qu'on change ici aussi en siojrr

3) Matériel d'interconnexion

Nos routeurs et switch peuvent communiquer avec Nagios par le biais du protocole SNMP (Simple Network Management Protocol), pour ce faire il faut l'activer et le configurer avec la communauté et le mode lecture-seule.

En se connectant via telnet pour Cisco / SSH pour Aruba :

```
#Pour Cisco
conf t
snmp-server community siojrr RO
snmp-server host 172.17.1.13 siojrr
snmp-server host 172.17.1.6 siojrr
snmp-server host 172.17.1.7 siojrr
#Pour Aruba (HP)
snmp-server community "public" unrestricted
snmp-server community "siojrr" operator
```

b. Création des services

1) Plugin NCPA

L'agent NCPA en place sur les serveurs va permettre un remonté d'informations, cependant il faut sur Nagios définir des services. On les créera dans le fichier */etc/nagios4/conf.d/cmdes_servicesNCPA.cfg* et dans un premier temps on ajoutera le modèle qu'on utilisera pour chaque service :

```
define command {
    #Nom de la commande lors de l'appel
    command_name checkncpa
    #Commande qui sera exécutée avec les arguments
    command_line $USER1$/check_ncpa.py -H $HOSTADDRESS$ $ARG1$
}
```

Sur une machine disposant du plugin NCPA une nouvelle interface web est disponible à l'adresse <https://IPmachine:5693/gui>

Cette interface nous permet de consulter les éléments récoltés par NCPA, notamment l'onglet API que nous utiliserons pour trouver "l'arborescence" des ressources.

Par exemple nous souhaitons monitorer la quantité de RAM libre, pour se faire on ajoute dans le fichier *cmdes_servicesNCPA.cfg* un nouveau service :

```
define service {
```

```

#Le template utilisé
    use                generic-service
#Le groupe sur lesquels on applique le service
    hostgroup_name      Machine-Physique, Serveur
#La description affichée sur l'interface de Nagios
    service_description  Utilisation du CPU en pourcent
    check_command        checkncpa! -t siojrr -M cpu/percent -w 80 -c 90
}

define service {
#Le template utilisé
    use                generic-service
#Le groupe sur lesquels on applique le service
    hostgroup_name      Machine-Physique, Serveur
#La description affichée sur l'interface de Nagios
    service_description  Mémoire Libre
    check_command        checkncpa! -t siojrr -M memory/virtual/free -u G
}

define service {
    use                generic-service
    hostgroup_name      Serveur
    service_description  Service SSH
    check_command        check_ssh! -H $HOSTADDRESS$
}

define service {
    use                generic-service
    hostgroup_name      Machine-Windows
    service_description  Utilisation du Disque en Pourcent
    check_command        checkncpa! -t siojrr -M 'disk/logical/C:/used_percent' --warning
    90 --critical 95
}

define service {
    use                generic-service
    hostgroup_name      Machine-Linux
    service_description  Utilisation du Disque en Pourcent
    check_command        checkncpa! -t siojrr -M 'disk/logical/|/used_percent' --warning 90
--critical 95
}

```

Après un redémarrage on trouve dans l'onglet Services de Nagios notre hôte avec le service défini, avec le statut et les valeurs dans la dernière colonne :

All-Machine	Mémoire Libre	OK	03-10-2022 10:49:38	0d 17h 39m 22s	1/3	OK: Free was 7.75 GB
	Utilisation de la Mémoire en pourcent	OK	03-10-2022 10:51:53	0d 17h 36m 58s	1/3	OK: Percent was 54.50 %
	Utilisation du CPU en pourcent	OK	03-10-2022 10:54:08	0d 17h 34m 34s	1/3	OK: Percent was 3.10 %, 6.10 %, 3.10 %, 12.50 %
	Utilisation du Disque en Pourcent	OK	03-10-2022 10:47:37	0d 0h 47m 15s	1/3	OK: Used_percent was 40.50 %
DHCP-DNS-Linux	Mémoire Libre	OK	03-10-2022 10:49:52	0d 17h 32m 10s	1/3	OK: Free was 0.01 GB
	Service SSH	OK	03-10-2022 10:52:07	0d 17h 39m 6s	1/3	SSH OK - OpenSSH_7.9p1 Debian-10+deb10u2 (protocol 2.0)
	Utilisation de la Mémoire en pourcent	OK	03-10-2022 10:54:22	0d 17h 36m 42s	1/3	OK: Percent was 68.80 %
	Utilisation du CPU en pourcent	OK	03-10-2022 10:47:50	0d 17h 34m 18s	1/3	OK: Percent was 0.00 %
	Utilisation du Disque en Pourcent	OK	03-10-2022 10:50:05	0d 0h 44m 47s	1/3	OK: Used_percent was 10.30 %

*Statut en vert si normal, orange pour warning
et rouge pour critical*

S'il est possible de remonter beaucoup d'informations grâce à NCPA il peut être intéressant de passer par le plugin dédié à ce service (stockés dans /usr/lib/nagios4/plugins/), si on souhaite surveiller le service SSH on va donc utiliser check_ssh :

```
define service {
    use generic-service
    hostgroup_name srvLinux
    service_description Service
    SSH
    check_command check_ssh! -H $HOSTADDRESS$
```

Service SSH	OK	04-15-2022 00:04:47	0d 9h 25m 18s	1/3	SSH OK - OpenSSH_7.9p1 Debian-10+deb10u2 (protocol 2.0)
-------------	----	---------------------	---------------	-----	---------------------------------------------------------

Mais alors que faire s'il n'y a pas de plugin pour un service en particulier, par exemple check_proxy n'existe pas. Et bien 3 solutions s'offrent à nous :

- Développer le plugin
- Utiliser un plugin communautaire (pratique mais pas forcément à jour)
- Contourner le problème avec un autre plugin déjà présent

C'est cette troisième option que l'on va suivre, dans le cas de notre proxy on peut utiliser le check_http pour tester la connexion à un site internet normalement bloqué par notre proxy. Il suffit de mettre en warning une communication établie pour détecter un problème de filtrage.

Une autre solution est tout simplement de vérifier, par le biais de check_ncpa que le service est actif sur la machine

SNMP

3) Matériel d'interconnexion

Nos routeurs et switch peuvent communiquer avec Nagios par le biais du protocole SNMP (Simple Network Management Protocol), pour ce faire il faut l'activer et le configurer avec la communauté et le mode lecture-seule.

En se connectant via telnet pour Cisco / en SSH pour Aruba (HP)

```
#Pour Cisco
conf t
snmp-server community siojrr RO
snmp-server host 172.17.1.13 siojrr
snmp-server host 172.17.1.6 siojrr
snmp-server host 172.17.1.7 siojrr
#Pour Aruba (HP)
snmp-server community "public" unrestricted
snmp-server community "siojrr" operator
```

-H = adresse du commutateur ou du router
-C = communauté utiliser dans ce cas c'est siojrr
-O = OID a mettre
-C = pour critical
-W = warning pas mis dans l'exemple

#exemple de define command pour des machine qui on le même OID (commutateur)

```
define command {
    command_name snmp_up
    command_line $USER1$/check_snmp -H
$HOSTADDRESS$ -C $ARG1$ -o .1.3.6.1.2.1.2.2.1.8.$ARG2$ -c 1
}
```

#exemple de define command pour une machine et un OID

```
#define command {
#    command_name snmp_up1
#    command_line $USER1$/check_snmp -H 172.17.1.2 -C siojrr
-o .1.3.6.1.2.1.2.2.1.8.1 -c 1
#
#}
```

#Switch Cisco

```
define service {  
    use          generic-service  
    host_name     Switch-Cisco  
    service_description  Lien Trunk 20  
    check_command snmp_up!siojlr!10020  
}
```

```
define service {  
    use          generic-service  
    host_name     Switch-Cisco  
    service_description  Lien Trunk 21  
    check_command snmp_up!siojlr!10021  
}
```

```
define service {  
    use          generic-service  
    host_name     Switch-Cisco  
    service_description  Lien Trunk 23  
    check_command snmp_up!siojlr!10023  
}
```

```
define service {  
    use          generic-service  
    host_name     Switch-Cisco  
    service_description  Lien Trunk 24  
    check_command snmp_up!siojlr!10024  
}
```

#Switch Aruba

```
define service {  
  
    use          generic-service  
    host_name     Switch-Aruba  
    service_description  Lien Trunk 5  
    check_command snmp_up!siojlr!5  
  
}
```

```
define service {  
  
    use          generic-service  
    host_name     Switch-Aruba  
    service_description  Lien Trunk 9  
    check_command snmp_up!siojlr!9  
  
}
```

```
}
```

```
define service {
```

```
    use                generic-service
```

```
    host_name          Switch-Aruba
```

```
    service_description Lien Trunk 10
```

```
    check_command       snmp_up!sioj!10
```

```
}
```