BTS SERVICES INFORMATIQUES AUX ORGANISATIONS **SESSION 2022**

Épreuve E5 - Administration des systèmes et des réseaux (option SISR)

ANNEXE 7-1-A : Fiche descriptive de réalisation professionnelle (recto)

DESCRIPTION D'UNE RÉALISATION PROFESSIONNELLE

N° réalisation : 16

Nom, prénom : Leduc Quentin

N° candidat : 02145639104

Épreuve ponctuelle **Contrôle en cours de formation**

Date : 18 / 04 / 2022

Organisation support de la réalisation professionnelle
Dans Scani nous souhaitons partager un VPN à nos coopérateurs qui nous le demande pour faire passer tout leur réseau quand ils sont en déplacement ou quand il se connecte à des accès Wi-Fi public.

Intitulé de la réalisation professionnelle
Installation d'un VPN Wireguard avec monitoring de la carte reseau avec Zabbix

Période de réalisation : 18/04/2022 **Lieu :** Joigny
Modalité : Seul(e)

Compétences travaillées
- **Recenser et identifier les ressources numériques**
- **Vérifier les conditions de la continuité d'un service informatique**
- **Déployer un service**
- **Accompagner les utilisateurs dans la mise en place d'un service**
- **Mettre en place son environnement d'apprentissage personnel**
- **Développer son projet professionnel**

Conditions de réalisation (ressources fournies, résultats attendus)
Pour nos tests, car ce projet est encore en beta, nous avons utiliser le logiciel Wireguard car il est plus rapide et plus léger que OpenVPN mais surtout il est compatible avec PC, MAC, Linux, Android et IOS grâce a l'application.

Description des ressources documentaires, matérielles et logicielles utilisées

- Rocky Linux 8
- Zabbix 5.0 LTS
- Wireguard

Modalités d'accès aux productions et à leur documentation

Site internet : www.netwaze.fr
Aller dans réalisation professionnelle et mettre dans mot de passe : Mr.Robot

BTS SERVICES INFORMATIQUES AUX ORGANISATIONS **SESSION 2022**

Épreuve E5 - Administration des systèmes et des réseaux (option SISR)

ANNEXE 7-1-A : Fiche descriptive de réalisation professionnelle (verso, éventuellement pages suivantes)

SYNOPSIS

Pour répondre à notre contexte nous allons installer et configurer un serveur Wireguard avec la création de nos utilisateurs.

Wireguard fonctionne quand notre interface de gestion fonctionne, par conséquent nous allons la superviser avec notre serveur Zabbix pour remonter les alertes en cas de panne.

Descriptif de la réalisation professionnelle, y compris les productions réalisées et schémas explicatifs

CULTURE

Wireguard est un jeune VPN qui a commencer a éclore le 30 Juin 2016 et les premières release du vin commence en Juin 2018 avec une version pas stable et encore expérimentale.

L'objectif de Wireguard est de fournir un VPN simple, rapide et sécuriser. A la différence de OpenVPN, Wireguard tient sur 4000 lignes de code comparer a OpenVPN qui lui en a au minimum 70 000.

PRÉREQUIS

Nous allons installer notre serveur Wireguard sous Rocky Linux 8 et intégrer le zabbix-agent dans notre serveur Zabbix pour surveiller notre Wireguard.

On met a jour notre distribution et on installe les dépôts EPEL

sudo dnf install epel-release elrepo-release

dnf update --refresh -y

INSTALLATION ET CONFIGURATION DE WIREGUARD

Apres avoir installer le dépôt epel, nous installons le logiciel Wireguard

dnf install wireguard-tools kmod-wireguard

On doit changer l'umask du dossier wireguard pour pouvoir créer nos fichiers dedans.

umask 077 /etc/wireguard

On va dans /etc/wireguard et on créer un fichier qui portera le nom de notre interface virtuelle (wg1.conf) de notre Wireguard.

cd /etc/wireguard

touch wg1.conf

On créer ensuite notre clé privée pour l'interface de Wireguard

wg genkey > /etc/wireguard/private

Si on l'ouvre nous verrons notre clé privée

```
[[root@Wireguard wireguard]# cat private
 2Nxh/AXASNFHFX1IoiPX1ZTmFdbHsVPdtSPMYeo012E=
 [root@Wireguard wireguard]#
```

On génère maintenant la clé publique du serveur Wireguard

wg pubkey < /etc/wireguard/private > /etc/wireguard/public

```
[root@Wireguard wireguard]# cat public
usVz67WmTb5zuNgwaxr+q+8JOdw/KUPPTYi1BQoK1zM=
[root@Wireguard wireguard]#
```

Maintenant nous pouvons modifier notre fichier wg1.conf et rentrer notre configuration

```
[Interface]
Address = 10.8.0.1/24
SaveConfig = true
ListenPort = 49530
DNS = 1.1.1.1,8.8.8.8
PrivateKey = 2Nxh/AXASNFHFX1IoiPX1ZTmFdbHsVPdtSPMYeo012E=
PostUp = firewall-cmd --add-port=49530/udp; firewall-cmd --zone=public
--add-masquerade; firewall-cmd --direct --add-rule ipv4 filter FORWARD 0
-i wg1 -o ens192 -j ACCEPT; firewall-cmd --direct --add-rule ipv4 nat
POSTROUTING 0 -o ens192 -j MASQUERADE
PostDown = firewall-cmd --remove-port=49530/udp; firewall-cmd --
zone=public --remove-masquerade; firewall-cmd --direct --remove-rule
ipv4 filter FORWARD 0 -i wg1 -o ens192 -j ACCEPT; firewall-cmd --direct
--remove-rule ipv4 nat POSTROUTING 0 -o ens192 -j MASQUERADE
```

On ouvre avec firewall-cmd le port 49530 en udp en ipv4 sur l'interface wg1 et qui permet de communiquer avec notre réseau local avec l'interface ens192.

On créer une interface avec comme adresse de vpn 10.8.0.1, on met notre clé privée créer précédemment et on choisi un port de connexion pour pouvoir se connecter a distance. Attention toutefois d'ouvrir le port sur notre box ou pare-feu.

Maintenant on monte notre interface en la démarrant avec notre Wireguard. On l'active aussi au démarrage.

sudo systemctl enable wg-quick@wg1

sudo systemctl start wg-quick@wg1

```
[root@Wireguard wireguard]# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group defaul
t qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: ens192: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP gr
oup default qlen 1000
    link/ether 00:0c:29:94:a5:33 brd ff:ff:ff:ff:ff:ff
    inet 192.168.250.57/24 brd 192.168.250.255 scope global dynamic noprefixrout
e ens192
       valid_lft 30647sec preferred_lft 30647sec
    inet6 2a01:e0a:128:4080:20c:29ff:fe94:a533/64 scope global dynamic noprefixr
oute
       valid_lft 86382sec preferred_lft 86382sec
    inet6 fe80::20c:29ff:fe94:a533/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
3: wg1: <POINTOPOINT,NOARP,UP,LOWER_UP> mtu 1420 qdisc noqueue state UNKNOWN gro
up default qlen 1000
    link/none
    inet 10.8.0.1/24 scope global wg1
       valid_lft forever preferred_lft forever
[root@Wireguard wireguard]#
```

On peut voir si notre Wireguard fonctionne bien avec systemctl

sudo systemctl status wg-quick@wg1

```
[root@Wireguard wireguard]# sudo systemctl status wg-quick@wg1
● wg-quick@wg1.service - WireGuard via wg-quick(8) for wg1
   Loaded: loaded (/usr/lib/systemd/system/wg-quick@.service; enabled; vendor p>
   Active: active (exited) since Mon 2022-04-18 15:48:14 CEST; 5min ago
     Docs: man:wg-quick(8)
           man:wg(8)
           https://www.wireguard.com/
           https://www.wireguard.com/quickstart/
           https://git.zx2c4.com/wireguard-tools/about/src/man/wg-quick.8
           https://git.zx2c4.com/wireguard-tools/about/src/man/wg.8
  Process: 18216 ExecStart=/usr/bin/wg-quick up wg1 (code=exited, status=0/SUCC>
 Main PID: 18216 (code=exited, status=0/SUCCESS)

avril 18 15:48:13 Wireguard systemd[1]: Starting WireGuard via wg-quick(8) for >
avril 18 15:48:13 Wireguard wg-quick[18216]: [#] ip link add wg1 type wireguard
avril 18 15:48:13 Wireguard wg-quick[18216]: [#] wg setconf wg1 /dev/fd/63
avril 18 15:48:13 Wireguard wg-quick[18216]: [#] ip -4 address add 10.8.0.1/24 >
avril 18 15:48:14 Wireguard wg-quick[18216]: [#] ip link set mtu 1420 up dev wg1
avril 18 15:48:14 Wireguard systemd[1]: Started WireGuard via wg-quick(8) for w>
lines 1-18/18 (END)
```

On active le routage pour faire passer les paquets de l'extérieur dans notre VPN.
On modifie le fichier sysctl.conf dans /etc

nano /etc/sysctl.conf

et on ajoute dans le fichier net.ipv4.ip_forward=1

```
# sysctl settings are defined through files in
# /usr/lib/sysctl.d/, /run/sysctl.d/, and /etc/sysctl.d/.
#
# Vendors settings live in /usr/lib/sysctl.d/.
# To override a whole file, create a new file with the same in
# /etc/sysctl.d/ and put new settings there. To override
# only specific settings, add a file with a lexically later
# name in /etc/sysctl.d/ and put new settings there.
#
# For more information, see sysctl.conf(5) and sysctl.d(5).
net.ipv4.ip_forward=1
```

On reload le service et on voit si les changements on bien était fait.

sysctl -p

```
[root@Wireguard wireguard]# sysctl -p
net.ipv4.ip_forward = 1
[root@Wireguard wireguard]#
```

On redémarre notre serveur Wireguard

sudo systemctl restart wg-quick@wg1

CREATION D'UN UTILISATEUR

Pour créer notre utilisateur on aura besoin de 2 clés, une public et une privée.
Pour générer ces clés on tape la commande :

wg genkey | sudo tee privatekey | wg pubkey | sudo tee /etc/wireguard/publickey

```
[root@Wireguard wireguard]# ls
privatekey  publickey  wg1.conf
[root@Wireguard wireguard]# cat privatekey
KC/4N+5D3oGPuvM8HpzPf6SffE6njAi1NVejS6LxNWM=
[root@Wireguard wireguard]# cat publickey
REqEynoLpvvZbPht/J1Sit60FwwPWt5MEnuXGd8B9HU=
[root@Wireguard wireguard]#
```

On réouvre notre fichier wg1.conf et on ajoute notre client.

#Peer Kiu
[Peer]
PublicKey = REqEynoLpvvZbPht/J1Sit60FwwPWt5MEnuXGd8B9HU=
AllowedIPs = 10.8.0.2/32

```
[Interface]
Address = 10.8.0.1/24
PrivateKey = 2Nxh/AXASNFHFX1IoiPX1ZTmFdbHsVPdtSPMYeo012E=
ListenPort = 49530

[Peer]
PublicKey = REqEynoLpvvZbPht/J1Sit60FwwPWt5MEnuXGd8B9HU=
AllowedIPs = 10.8.0.2/32
```

et on reload la configuration de notre serveur wireguard

wg syncconf wg1 <(wg-quick strip wg1)

CONNEXION DE L'UTILISATEUR

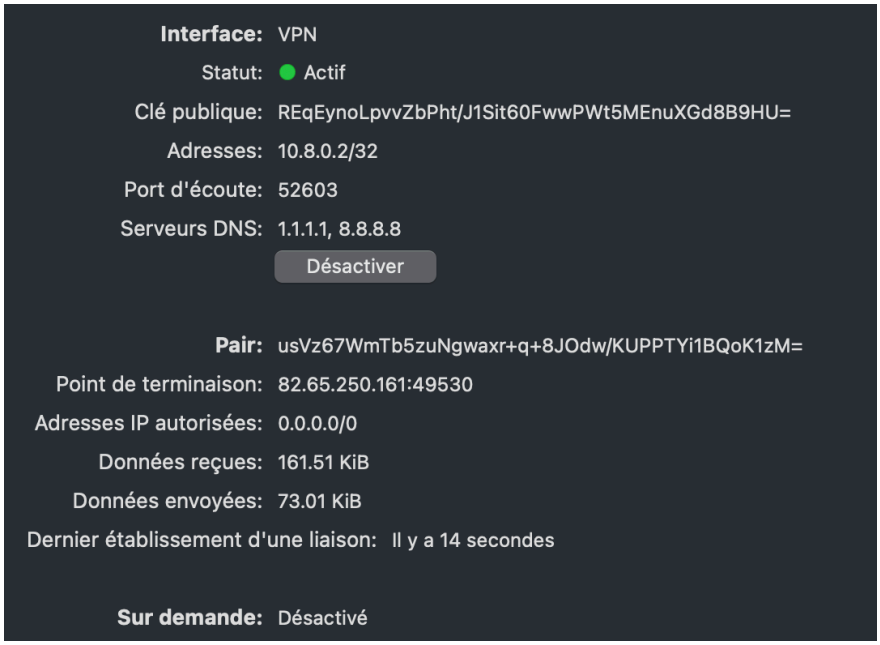
Sur Windows et sous MacOS il existe une application Wireguard qui sert à pouvoir se connecter avec interface graphique. Pour Linux, on peut le faire mais en ligne de code.

On créer une nouvelle interface dans le logiciel et on met dedans :

[Interface]
PrivateKey = KC/4N+5D3oGPuvM8HpzPf6SffE6njAi1NVejS6LxNWM= #Notre clé privé Kiu
Address = 10.8.0.2/32
DNS = 80.67.169.12, 80.67.169.40

[Peer]
PublicKey = usVz67WmTb5zuNgwaxr+q+8JOdw/KUPPTYi1BQoK1zM= = #Clé Public de notre wg1
AllowedIPs = 0.0.0.0/0 #fait passer tout le traffic internet dans le VPN
Endpoint = 0.0.0.0:51820 #0.0.0.0 = notre ip public ou un nom de domaine.

On voit que notre VPN fonctionne car on envoi et on reçois des données qui passe dans Wireguard.



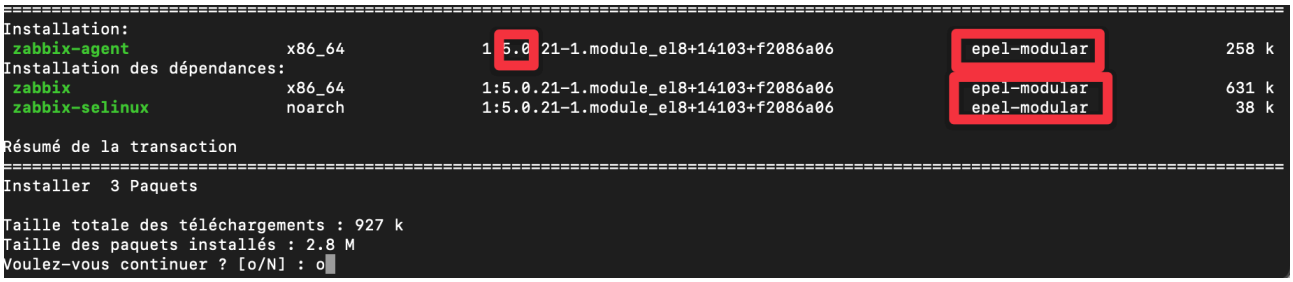
INSTALLATION DU ZABBIX-AGENT

Les Zabbix-agent ne sont pas intégrer dans les dépôts de Rocky Linux / Red Hat mais sont dans les dépôts EPEL. Par défaut le module de zabbix 4.0 est enable, seulement notre zabbix est en 5.0 LTS. Nous devons donc enable le nouveau module.

dnf module enable zabbix:5.0

et installer l'agent

dnf install zabbix-agent



On va modifier le fichier /etc/zabbix_agentd.conf et nous lui entrons la configuration de notre serveur Wireguard.

```
Server=192.168.0.13
ListenPort=10050
ServerActive=192.168.0.13
Hostname=Wireguard
```

On démarre notre client zabbix et on l'enable pour qu'il se lance automatiquement

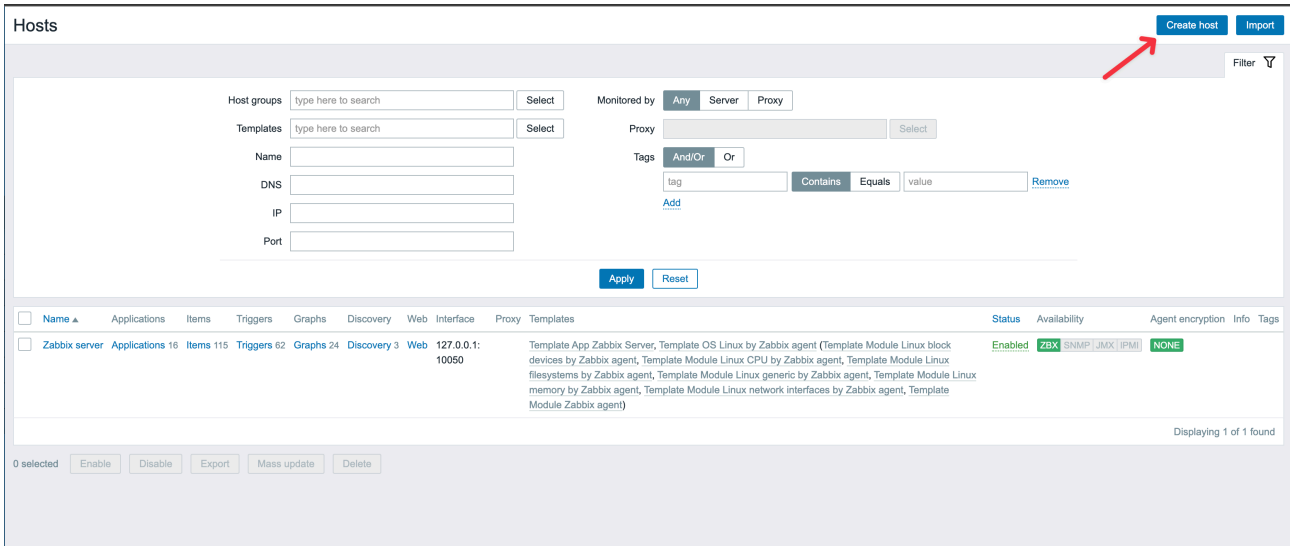
systemctl enable zabbix-agent.service
systemctl start zabbix-agent.service

On configure le firewalld

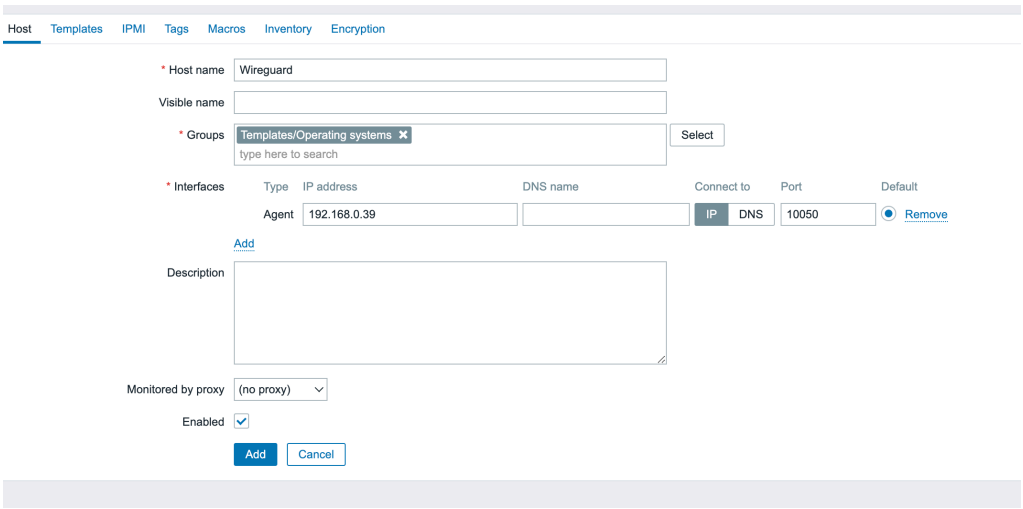
firewall-cmd --add-port=10050/tcp --permanent
firewall-cmd --reload

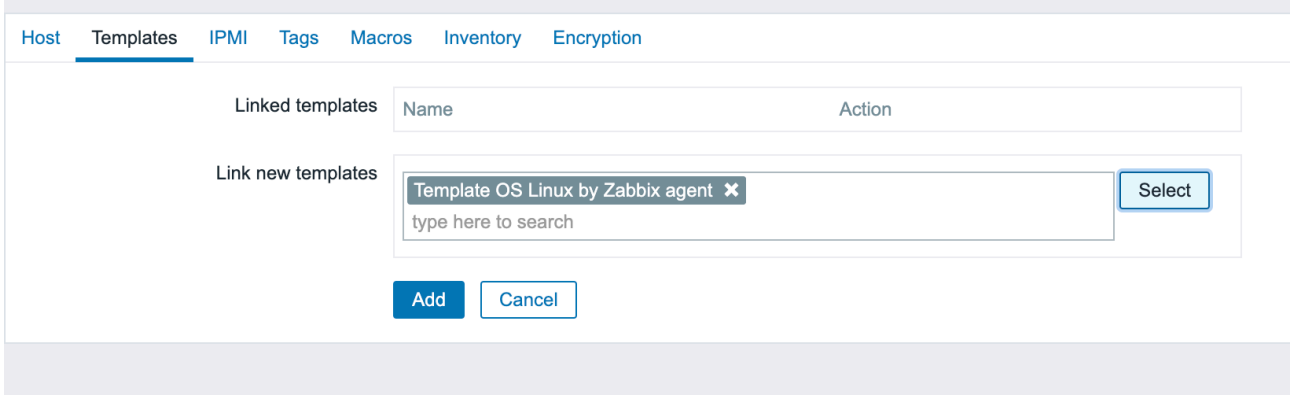
SURVEILLANCE DE NOTRE INTERFACE ZABBIX

On se connecte sur le Zabbix, et on créer notre nouveau hôte dans configuration - Hosts

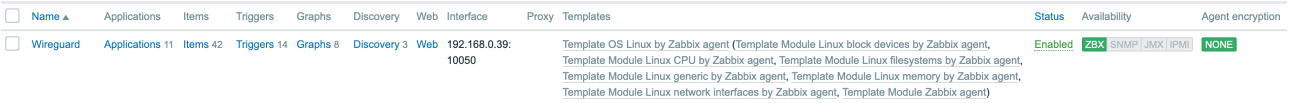


et on complète les informations de notre serveur Wireguard a savoir : son nom, son ip et son groupe et son template. Une fois terminé on clique sur add.





Une fois le ZBX en vert cela veut dire que notre serveur zabbix et l'agent communiquent bien ensemble et que maintenant il peuvent transmettre des informations.



Maintenant nous voulons superviser notre interface virtuelle wg1, car si wg1 n'est pas monter alors wireguard ne fonctionne plus.

Dans notre serveur Wireguard on va modifier notre fichier zabbix_agentd

nano /etc/zabbix_agentd.conf

et on va a la fin de notre fichier pour ajouter la ligne :

```
UserParameter=Wireguard_Interface_WG1,ip a | grep -i -m 1 'wg1' | wc -l
```

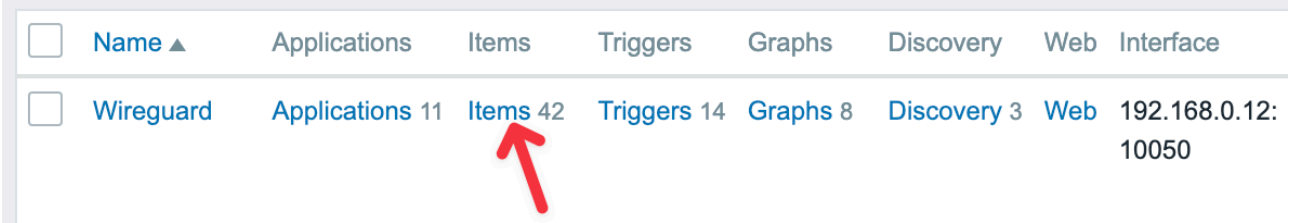
Si nous tapons la commande : # ip a | grep -i -m 1 'wg1' | wc -l

```
[root@Wireguard wireguard]# ip a | grep -i -m 1 'wg1' | wc -l
1
[root@Wireguard wireguard]#
```

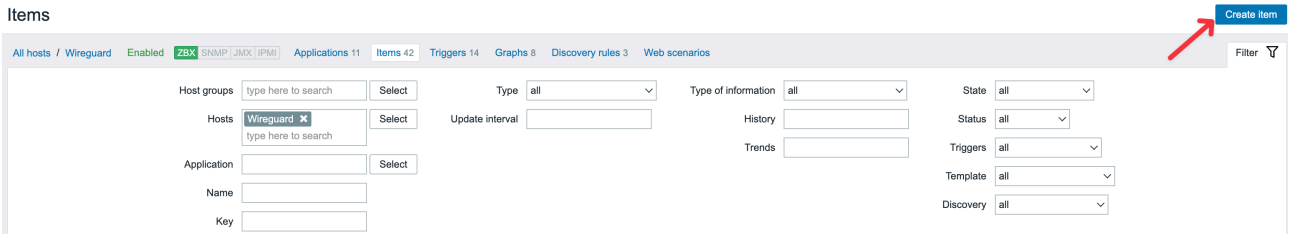
Nous devons avoir un 1 ce qui est notre but car Zabbix ne comprend que 1 et 0. 1 signifie que l'interface est la et 0 que l'interface a disparu.

Dans la ligne UserParameter il faut bien faire attention au nom de l'item, ici j'ai mis Wireguard_Interface_WG1 mais nous pouvons mettre au chose mais il faut avoir pareil dans le nom de notre Item zabbix.

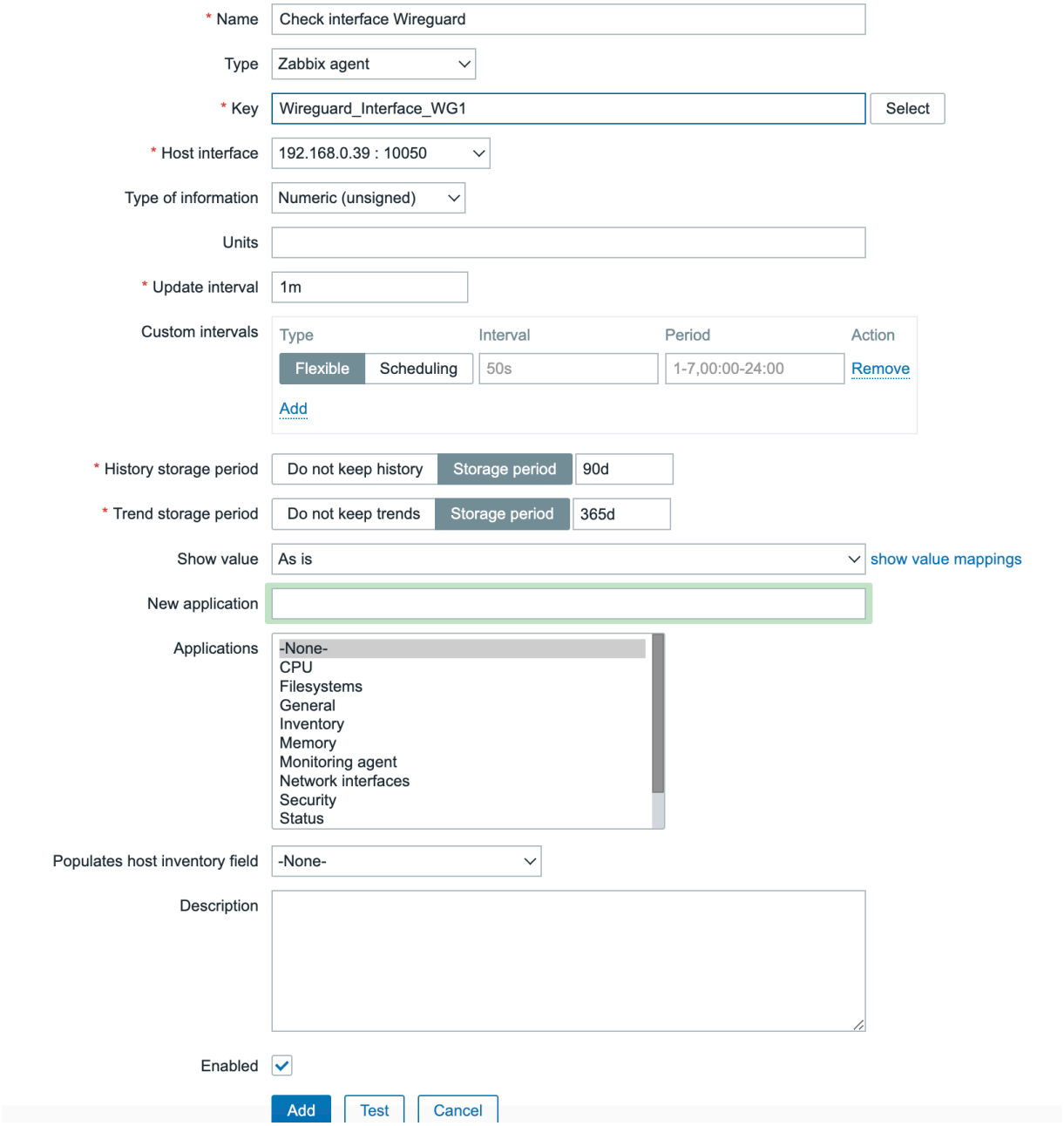
Par consequent nous allons depuis Hosts dans Items



et nous pouvons créer un nouvel Items



Pour le nom de l'item nous pouvons mettre ce que l'on veut, je vais mettre Check interface Wireguard, on laisse le type par défaut qui est le zabbix agent, et pour la key on met le nom que l'on a donner dans notre commande soit : Wireguard_Interface_WG1. On peut laisser le reste par défaut et cliquer sur add.

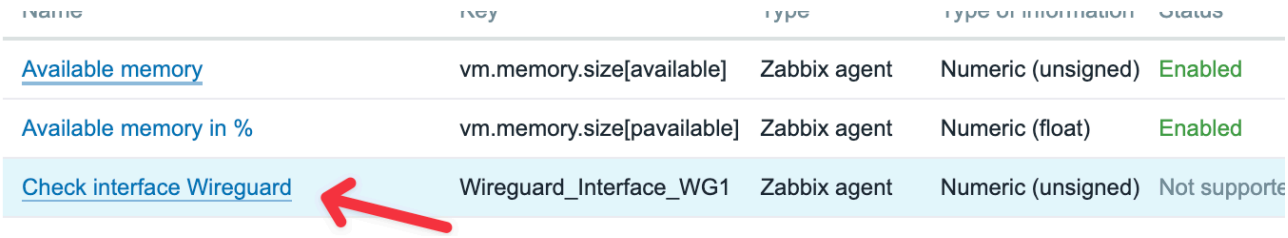


Nous devons aussi rajouter un triggers qui servira a remonter les alertes avec leur niveau de d'urgence.

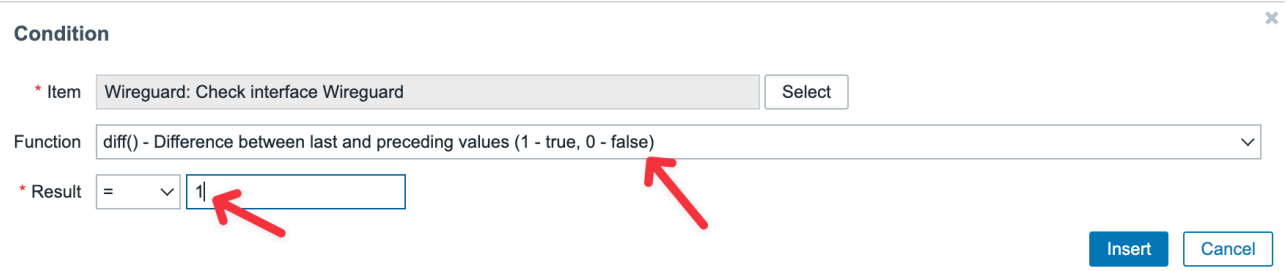
Dans Configuration - Hosts on clique sur triggers puis sur create triggers.

On remplis les information demander comme le nom, ici je vais mettre Wireguard VPN, on ajoute un niveau de severity pour moi cela sera disaster, et le plus important on rajoute une expression en cliquant sur Add.

On clique sur notre Check interface Wireguard



et dans function on prends diff qui prendra la différence de numero (0 et 1) par rapport a d'anciennes données et dans le result on cherche en résultat le 1.



On insert la condition et on peut ajouter le triggers avec Add.

Maintenant on arrête notre VPN

systemctl stop wg-quick@wg1.service

et quand on fait un listing de nos interface réseau, on voit que notre wg1 a disparu.

```
[root@Wireguard ~]# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group defaul
t qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: ens192: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP gr
oup default qlen 1000
    link/ether 00:0c:29:94:a5:33 brd ff:ff:ff:ff:ff:ff
    inet 192.168.0.39/24 brd 192.168.0.255 scope global dynamic noprefixroute en
s192
       valid_lft 41592sec preferred_lft 41592sec
    inet6 2a01:e0a:3ec:ab90:20c:29ff:fe94:a533/64 scope global dynamic noprefixr
oute
       valid_lft 86278sec preferred_lft 86278sec
    inet6 fe80::20c:29ff:fe94:a533/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
[root@Wireguard ~]#
```

Sachant que SELINUX est activer, zabbix ne pourra pas lancer la commande car SELINUX rendra la commande inutilisable car elle sera lancer par sudo. par conséquent, on va autoriser zabbix de lancer des commandes en sudo.

On peut voir tous les modules que l'on peut accorder a zabbix avec SELINUX

getsebool -a | grep zabbix

```
[root@Wireguard ~]# getsebool -a | grep zabbix
httpd_can_connect_zabbix --> off
zabbix_can_network --> off
zabbix_run_sudo --> off
[root@Wireguard ~]#
```

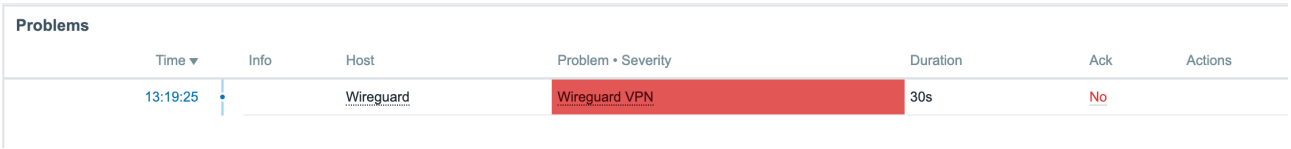
On active le le module zabbix_run_sudo

setsebool -P zabbix_run_sudo on

On retourne sur notre Zabbix et nous devrions avoir une alerte.

VALIDATION

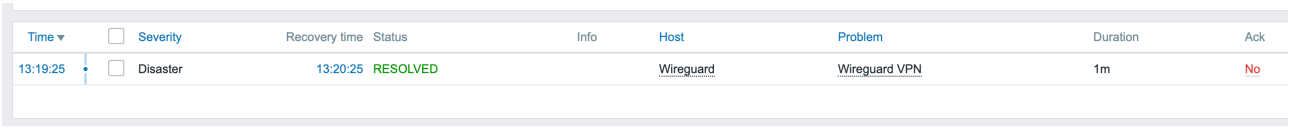
Une fois sur le zabbix on remarque que nous avons une alerte car notre interface virtuelle n'existe plus.



Si on redémarre notre Wireguard

sudo systemctl start wg-quick@wg1

et nous allons dans problems et nous voyons que notre problème est passer en résolu.



VEILLE TECHNOLOGIQUE

Pour la partie VPN nous aurions pu utiliser Openvpn ou des VPN IPSEC. Seulement Wireguard étant plus rapide, plus simple et aussi sécuriser qu'Openvpn j'ai décidé de partir dessus. Pour le coter Supervision, j'aurais pu prendre un Centreon, un prometheus ou EON avec Nagios. Seulement je n'ai jamais eu l'opportunité de travailler avec ces logiciels car dans Scani nous utilisons Zabbix.