Institut Universitaire de Technologie,
Aix-Marseille Université

RAPPORT DE STAGE
Diplôme Universitaire de Technologie
Spécialité Réseaux et Télécommunications

Stage en en développement d'outil de monitoring et de test pour des système de communications sol bord d'un métro

Thomas LUCOT

ATOS

Responsable entreprise : Clément GUEZOU

Responsable académique : Delphine ROUSSEAU

2019

Introduction :

Dans le cadre de mon cursus en IUT Réseaux et Télécommunications à Marseille Luminy, la réalisation d'un stage de dix semaines en fin d'études était indispensable à la validation de mon diplôme. Ce stage fut une expérience enrichissante en Réseaux mais aussi au point de vue relationnel. J'ai effectué ce stage au sein de l'entreprise Atos située à Aix-en-Provence. Il a été réalisé du 8 Avril 2019 au 14 Juin 2019.

J'ai donc intégré le projet MAV* et son équipe afin de travailler sur l'élaboration de la réalisation d'un banc de test. Mon travail constituait à ajouter des fonctions au banc de test afin de pouvoir passer des tests de performances. Ce projet permettra à l'équipe de pouvoir valider en situation réelle le système SVE car jusque-là les deux baies n'étaient câblées pas comme dans le train.
Dans ce rapport, nous verrons donc, après avoir fixé les objectifs et le contexte du stage, les détails de ce banc de test, son architecture et ses configurations puis nous finirons par les tests de supervision réseaux à l'aide d'un outil de supervision.

Table des matières

1 Contextes et Objectifs

1.1 Présentation de l'entreprise

Atos est l'un des principaux acteurs en service numérique et digital dans le monde, classé 8ème mondial parmi les entreprises numériques.
L'entreprise a été créé en 2000 et compte aujourd'hui environ 100 000 collaborateurs dans 72 pays avec un chiffre d'affaire annuel marqué autour de 12.3 milliards d'€ en fin 2018.
Les mots clés pour l'entreprise sont la transformation digitale, l'innovation et la création de valeur. Elle accompagne de grandes entreprises dans le monde en leur fournissant des outils permettant de les aider à réaliser leurs visions de l'entreprise du futur. ATOS s'installe dans de nombreux domaines tels que :

- Le conseil & le service technologique et l'intégration de systèmes
- L'infogérance
- Les services transactionnels par l'intermédiaire de la filière Worldline, leader européen des services en ligne
- Le cloud, le « big data » et la cyber sécurité

L'ensemble de ces domaines demandent diverses connaissances dans plusieurs grands secteurs tels que la défense, la santé, les médias, etc…
De plus l'entreprise fut partenaire technologique mondiale des jeux olympiques et paralympique de Pyeongchang 2018 en étant chargée d'organiser l'infrastructure informatique et de se préparer aux éventuelles cyberattaques.
ATOS SE a (2014) racheté l'entreprise Bull, qui avait racheté deux ans auparavant l'entreprise Amesys. L'entité transport dans laquelle j'ai fait mon stage est historiquement un service de AMESYS

1.2 Objectifs et Problématique du stage

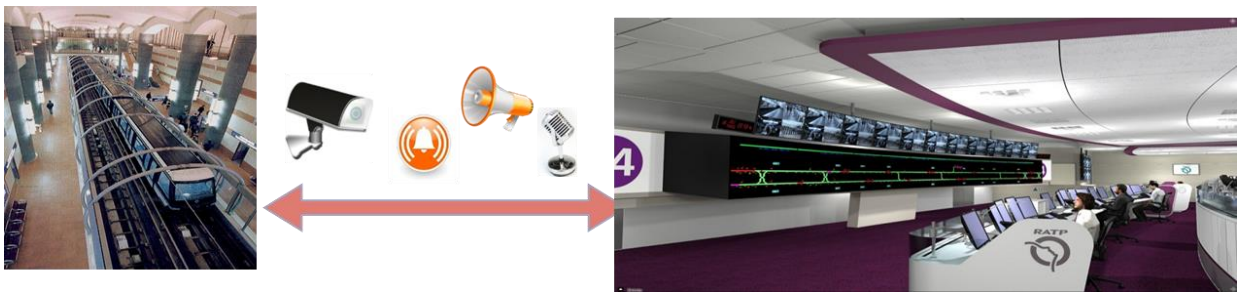
Le projet sur lequel j'ai travaillé durant mon stage est un système d'enregistrement vidéo embarquée dans des trains, qui permet, outre la visualisation en temps réel des images, d'enregistrer la vidéo lorsqu'une alarme est déclenchée (système SVE*). Le SVE dialogue avec des serveurs SOL dont le PCT. Il y a donc nécessité d'effectuer des tests sur le réseau train, mais également sur la partie réseau qui consiste à relier le train (Bord) aux équipements au sol (Sol) (système de transmission Sol/Bord TDSE*). Actuellement, il existe deux bancs de tests, un pour le test des caméras et un pour le test de la liaison extérieur. Les phases d'intégrations nécessitent de travailler de manière séparée sur les deux systèmes.
L'objectif de mon stage était de simuler le banc de test TDSE sur le banc de test SVE afin de pouvoir effectuer des tests de performances réseau. Il a donc fallu passer par différentes étapes que ce soit de l'architecture, aux configurations et au choix du matériel. Enfin, il a fallu définir comment effectuer les tests réseau, comment analyser les résultats et quel outil de supervision utiliser.
Pour réaliser mon projet je devais respecter différentes contraintes comme l'architecture déjà existante, qu'il a d'ailleurs fallu que je schématise, et l'utilisation de matériel imposé par le Client.

2 Etude d'une Solution MAV

2.1 Présentation du projet MAV

La supervision des moyens audio-visuels (MAV) pour les métros depuis le PCC (Poste de Contrôle Centralisé) permet :

- Le traitement des alarmes d'exploitation : bouton d'appel, blocage de portes palières, …
- L'information voyageurs (visualisation, sonorisation)
- La surveillance des équipements impactant l'exploitation



L'automatisation d'une ligne de métro impose la création de différents systèmes dont les systèmes Moyens Audio Visuels (MAV) qui participent à l'aide à l'exploitation du système de transport et à l'information des voyageurs dans les navettes et sur les quais.
Les éléments indispensables pour l'exploitant (traitement d'incident et info voyageur) sont :

- La levée de doute par vidéo
- L'information en cas de situation perturbée
- L'anticipation pour ne pas avoir de « sur incident »

Le sous-système MAV est lui-même con constitué de différents sous-systèmes :

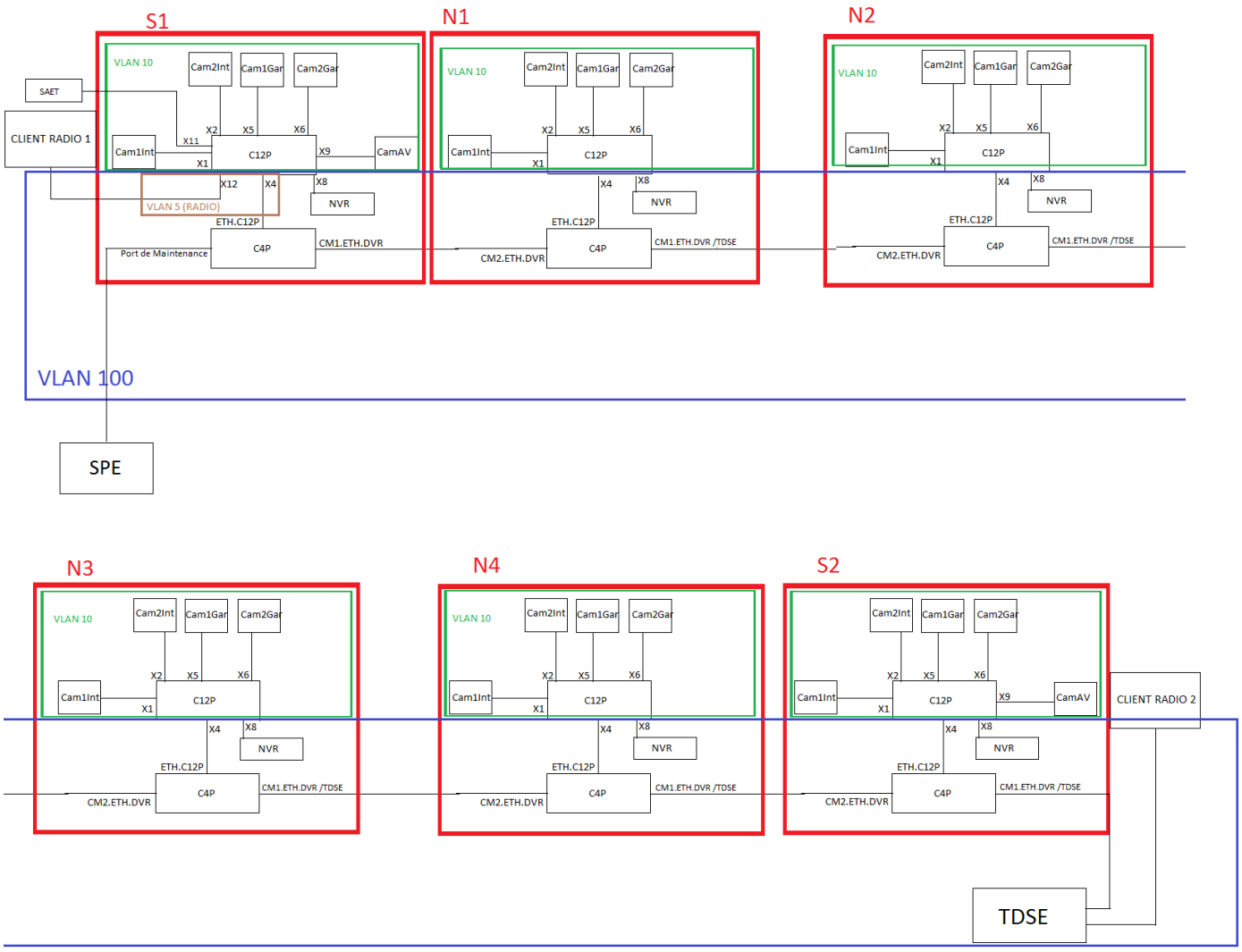
- Enregistreur vidéo embarqué (ENRVEMB),
- Caméras surveillance embarquées,
- Caméras garage embarquées (sur certaines lignes),
- Caméras voies embarquées,
- Micros embarqués (sur certaines lignes),
- Serveur phonie embarqué (SPE C6V),
- Le TDSE EMB (pour la liaison Sol/Bord).

Le TDSE permet les échanges de data
Le SPE permet les échanges Audio entre les passagers et le centre de supervision de la ligne.

2.2 Elaboration de l'Architecture du Réseaux MAV (du train)

Dans un premier temps j'ai dû m'approprier l'infrastructure réseaux sur laquelle j'allais intervenir. Ma première mission a donc été de réaliser le schéma de l'architecture de la baie. Pour ce faire, je me suis aidé des fichiers de configuration des routeurs C4P* et C12P*. J'ai donc, à l'aide du logiciel Paint, réaliser le schéma de ce réseau au niveau physique et VLAN (voir Annexe 1).

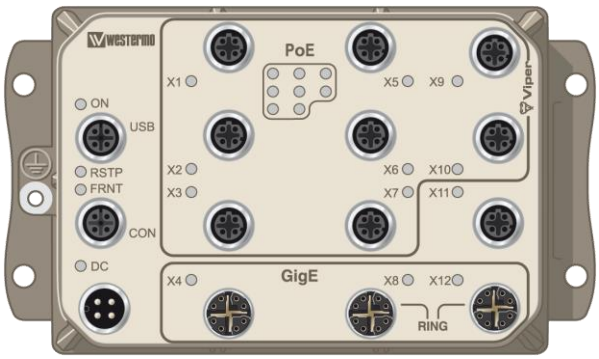
Le schéma représente l'architecture globale, or je n'ai travaillé que sur le VLAN* 100, qui est en fait le VLAN Train sur lequel les C4P, C12P, NVR et TDSE discutent et sur le VLAN 5, qui est le VLAN Radio sur lequel, le TDSE et les clients radio communiquent. Je vais donc devoir simuler les clients radio et le TDSE sur le Banc de test SVE.



Architecture Réseau du banc de Test

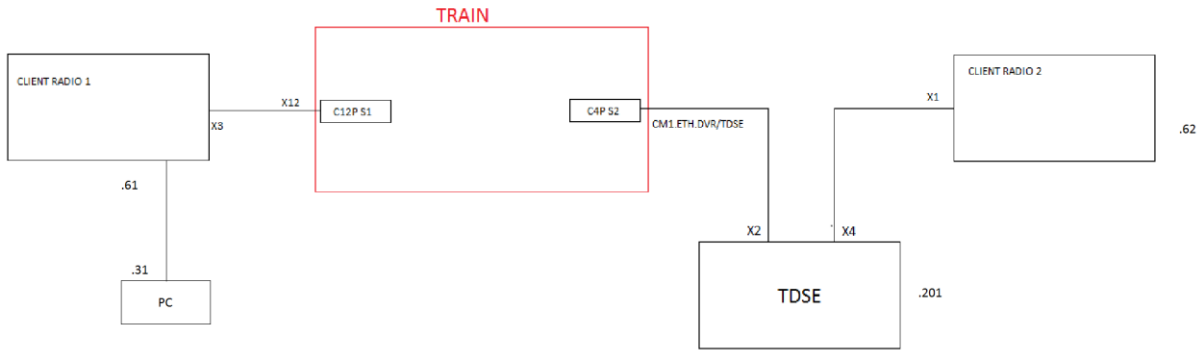
Les deux Clients Radio ne fonctionnent pas en même temps, il y en a un qui est le Master (maître) et l'autre le Backup (secours), leur statut varie selon le sens du train (celui qui est à l'arrière est le Master). C'est un script intégré dans les Clients Radio qui gère le basculement d'état.
Les Clients Radio sont chacun composés de deux cartes radio et d'une CPU routeur qui sont ensuite reliés au TDSE. Une des deux cartes radio sert pour la recherche du prochain point d'accès réseau et pour l'authentification tandis que la deuxième sert pour la connexion, c'est à travers elle que les informations sont échangées sur le réseau interne.

Pour créer cette simulation du TDSE et des clients radio nous allons utiliser des routeurs Westermo (modèle Viper-212-A). Le choix du matériel m'a été imposé par mon tuteur. Comme le train ne peut qu'aller dans deux sens il y a donc deux cas possibles celui où le client radio 1 est le master et celui où le client radio 2 est le master.



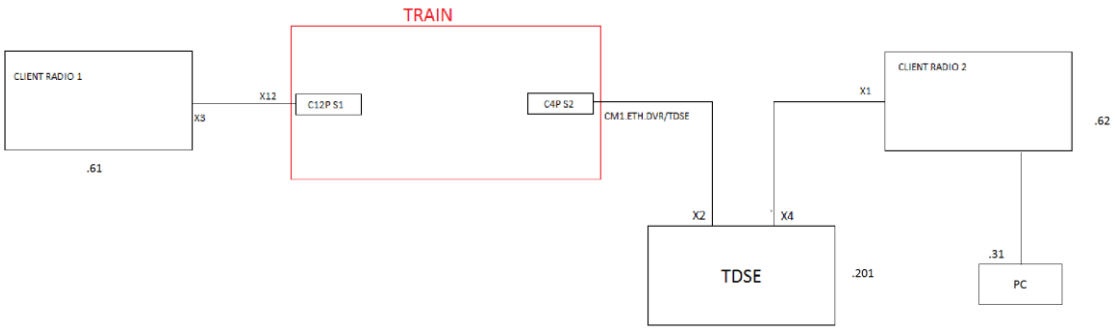
Westermo Viper-212-A

Dans le cas n°1, nous allons brancher notre PC superviseur sur le Client 1.



Architecture du Réseau Cas 1

Dans le cas n°2, nous allons déplacer le Pc superviseur sur le Client 2.



Architecture du Réseau Cas 2

Le Client Radio 1, le Client Radio 2 et le TDSE sont simulés par trois routeurs Westermo Viper-212A.
Il a fallu ensuite, dans notre architecture, définir les trois Vlans qui sont présents dans le système :

- Le Vlan 5 qui est le Vlan ou les trois Westermo discutent entre eux.
- Le Vlan 100 qui est le Vlan train (voir annexe 1)
- Le Vlan 15 qui représente le Vlan extérieur là où le Pc est connecté.

Dans cette architecture de simulation nous avons fait le choix de ne pas recréer de faire de transmission sans fil mais plutôt de nous connecter en filaire pour simplifier le banc de test.
Le Client Radio 1 sera en .61, le Client Radio 2 sera en .62 le TDSE sera en .201 et le Pc sera en .31 et ça peu importe dans quel Vlans ils se situent.

2.3 Choix des Configurations

Les configurations vont être séparées en trois ensembles, dans un premier temps les Vlans, ensuite le routage et pour finir nous verrons la traduction d'adresse.

Comme vu précédemment dans notre domaine d'application nous travaillons sur trois Vlans, le Vlan 5 pour la Radio, le Vlan 100 pour le réseau Train et le Vlan 15 pour la communication avec l'extérieur.

Les plages d'adresses définies sont les suivantes :

- Le Vlan 15 sera en 10.200.200.0/24
- Le Vlan 100 sera en 192.168.0.0/24
- Le Vlan 5 sera en 10.100.100.0/24

Ensuite, pour que les Vlans se voient mutuellement en utilisant le minimum de ports possible, nous les avons « tagué » sur les interfaces qui correspondent.

- Le port X2 du TDSE aura le Vlan 5 et 100 tagué
- Le port X4 du TDSE aura le Vlan 5 tagué
- Le port X3 de Client Radio 1 aura le Vlan 5 tagué
- Le port X1 de Client Radio 2 aura le Vlan 5 tagué
- Les port X5 des deux Clients Radio auront le Vlan 15 tagué

Du point de vue des configurations du routage, nous avons mis en place quelques routes statiques. Une route statique est un chemin qui permet d'atteindre un élément réseau en en fixant le chemin réseau pour l'atteindre.
Par exemple sur le Client Radio 1 dans le cas 1 (voir ci-dessous) le réseau de destination est 155.155.155.0/24 et on nous indique qu'il faut atteindre 10.100.100.201 pour discuter avec le réseau de destination.

Destination	Netmask	Distance	Gateway	Interface	Description	
155.155.155.0	255.255.255.0	1	10.100.100.201	*		

Route Statique Présente sur Client Radio dans le cas 1

Destination	Netmask	Distance	Gateway	Interface	Description	
10.200.200.0	255.255.255.0	1	10.100.100.61	*		
0.0.0.0	0.0.0.0	1	10.100.100.61	*	Default Route	

Route Statique Présente sur TDSE dans le cas 1

Les routes statiques dans ce cas permettent aux paquets de se diriger vers la bonne interface afin de rejoindre leur réseau de destination. Sur 10.100.100.61 nous pointons le réseau 155.155.155.0/24 qui est le réseau train après traduction (voir la suite). Quant à 192.168.0.201 pointe vers le réseau extérieur en passant par l'interface de client radio1 cela permet de forcer le passage sur client radio 1 et donc de tester le pire des cas.

Destination	Netmask	Distance	Gateway	Interface	Description	
155.155.155.0	255.255.255.0	1	10.100.100.201	*		

Route Statique Présente sur Client Radio 2 dans le cas 2

Destination	Netmask	Distance	Gateway	Interface	Description	
10.200.200.0	255.255.255.0	1	10.100.100.62	*		
0.0.0.0	0.0.0.0	1	10.100.100.62	*	Default Route	

Route Statique Présente sur TDSE dans le cas 2

Les routes statiques dans ce cas permettent aux paquets de se diriger vers la bonne interface afin de rejoindre leur réseau de destination. Sur 10.100.100.62 nous pointons le réseau 155.155.155.0/24 qui est le réseau train après traduction (voir ci-dessus). Quant à l'adresse 192.168.0.201 elle pointe vers le réseau extérieur en passant par l'interface de client radio 2, ce qui permet de forcer le passage sur client radio 2 et donc de tester le cas n°2.

Il n'est pas nécessaire de définir plus de Routages statiques dans notre cas, mais en situation réelle un script permet de faire de la redondance entre les deux clients radio.

Et pour finir avec les configurations, nous allons voir la traduction d'adresse présente sur le routeur TDSE.

Le NAT est obligatoire pour pouvoir communiquer avec le réseau du Vlan Train depuis un équipement extérieur d'un autre réseau car pour des raisons de maintenance les équipements ont l même adresse dans tous les trains.
Nous avons donc traduit les adresses du réseau privé en 192.168.0.0/24, qui sera visible depuis le sol (depuis un réseau extérieur au réseau train) en 155.155.155.0/24.
Pour ce faire, il a fallu se connecter en SSH* sur le routeur TDSE puis, pour pouvoir appliquer le NAT*, utiliser la commande shell suivante :

```
iptables -t nat -A PREROUTING -d 155.155.155.0/24 -j NETMAP –to 192.168.0.0/24
```

Cette commande aura pour effet de traduire les requêtes à destination de 155.155.155.0/24 en 192.168.0.0/24. Là nous avons traduit notre réseau dans le sens entrant mais nous ne pouvons pas sortir, pour cela il faut donc compléter la configuration avec la commande suivante :

```
iptables -t nat -A POSTROUTING -s 10.200.200.0/24 -j MASQUERADE
```

Cette commande permettra à notre requête traduite en provenance de 10.200.200.0/24 de pouvoir retrouver son chemin vers l'extérieur.

De l'extérieur nous pourrons donc discuter avec le Réseaux train qui ne sera pas vu en 192.168.0.0/24 mais en 155.155.155.0/24

Maintenant que nous avons nos configurations il nous suffit plus que de les mettre en place.

2.4 Mise en place sur le Banc de Test

Après avoir découvert l'architecture de notre réseau et sa configuration, nous allons maintenant voir l'implémentation de notre travail sur le banc de test.
Dans un premier temps, j'ai donc travaillé dans un réseau contenant uniquement mes trois routeurs pour vérifier que mes configurations étaient bonnes en pratique.
J'ai donc défini deux configurations car il existe deux versions différentes du TDSEe en fonction du train. SVE est présent uniquement sur les trains MP89et MP05. TDSE est aussi installé sur le MP14
Pour réaliser ces configurations, soit j'utilisais l'IHM Web des routeurs (pages http permettant d'accéder aux statuts et à la configuration des routeurs), soit je me connectais en ssh puis utilisais les commandes shell des routeurs.
La différence entre les deux configurations était la présence d'un protocole de redondance qui s'appelle VRRP*, il consiste à créer de la redondance dans le réseau en créant une adresse Ip virtuel et de la définir comme passerelle par défaut pour les hôtes qui est référencé dans un groupe de router. Dans notre cas les deux routeurs simulant les clients radio auraient été dans le groupe de routeur associé à l'adresse virtuel. J'ai configuré le NAT dans le shell car l'IHM* ne me permettait pas de faire la traduction que je souhaitais.

Après les avoir testées, j'ai pu commencer mon installation sur le banc de test.
Lors de cette installation, j'ai rencontré différents problèmes. En effet, lors de ma phase de test avant de passer sur le banc de test, je ne travaillais pas sur le même modele de routeur, il s'agissait d'un Moxa TN 5508 et d'un Westermo Viper-212-A. Il a donc fallu s'adapter à ce changement.
L'architecture du réseau train étant complexe, je n'avais pas repéré certaines spécificités, j'ai donc dû revoir certaines configurations.
Par exemple lors de mes premières configurations de test, j'utilisais un protocole de routage VRRP, mais ce protocole a été abandonnée sur le TDSE car étant un peu trop lent et surchargeant le réseau. Le protocole n'est donc pas présent car il est remplaçable par un script qui fera mieux le travail.
Il a donc été nécessaire de modifier les configurations et de refaire une phase de test.
Enfin, j'ai pu passer à l'installation finale de ma simulation sur le banc de test.

3 Supervision de MAV

Après avoir vu notre architecture et nos configurations pour notre nouvelle fonctionnalité du banc de test. Maintenant il nous faut le superviser, c'est-à-dire surveiller les équipements présents à l'aide de SNMP qui est le protocole le plus présent sur les équipements réseaux. L'objectif sera de réaliser des indicateurs sur la production afin de valider une phase projet (Vérification Service Régulier, elle a pour but de constater que le matériel ou installation fournis sont capables d'assurer un service régulier dans les conditions normales d'exploitation pour remplir les fonctions visées).
Dans notre cas nous allons utiliser Zabbix qui dispose d'une interface graphique qui nous permettra d'exploiter SNMP de manière plus lisible.

3.1 Introduction à SNMP

SNMP* est le protocole de gestion de réseaux proposé par l'IETF*. Il est actuellement le protocole le plus utilisé pour la supervision réseaux.
L'environnement de gestion SNMP est constitué de plusieurs composantes : la station de supervision, les éléments actifs du réseau, les variables MIB* et le protocole SNMP :

- La station de supervision (qui va être le manager) exécute les applications de gestion qui contrôlent les éléments réseaux. Physiquement, la station est un PC.
- Le MIB est une collection d'objets résidant dans une base d'information virtuelle. Ces collections d'objets sont définies dans des modules MIB spécifiques. Chaque information est reliée à un OID qui représente chacun des objets. Les OID sont rangés dans un ordre hiérarchique et se trouveront toujours au même endroit.
- Le protocole SNMP, qui permet à la station de supervision d'aller chercher les informations sur les éléments de réseaux et de recevoir des alertes provenant de ces mêmes éléments.

SNMP est basé sur un fonctionnement de questions/réponses avec des alertes. Il fonctionne donc avec un manager représentant notre PC de supervision dans notre cas, et un agent qui sera le client ou dans notre cas les éléments de notre réseau comme les C4P ou NVR… Le manager envoie des requêtes à l'agent(client), lequel retourne des réponses.
L'agent peut également renvoyer des alertes (« trap ») si besoin.
Le protocole SNMP utilise le port 161 pour l'agent afin qu'il puisse recevoir les requêtes de la station de gestion. Le port 162 est réservé pour la machine de gestion (manager) pour réceptionner les alertes des agents.

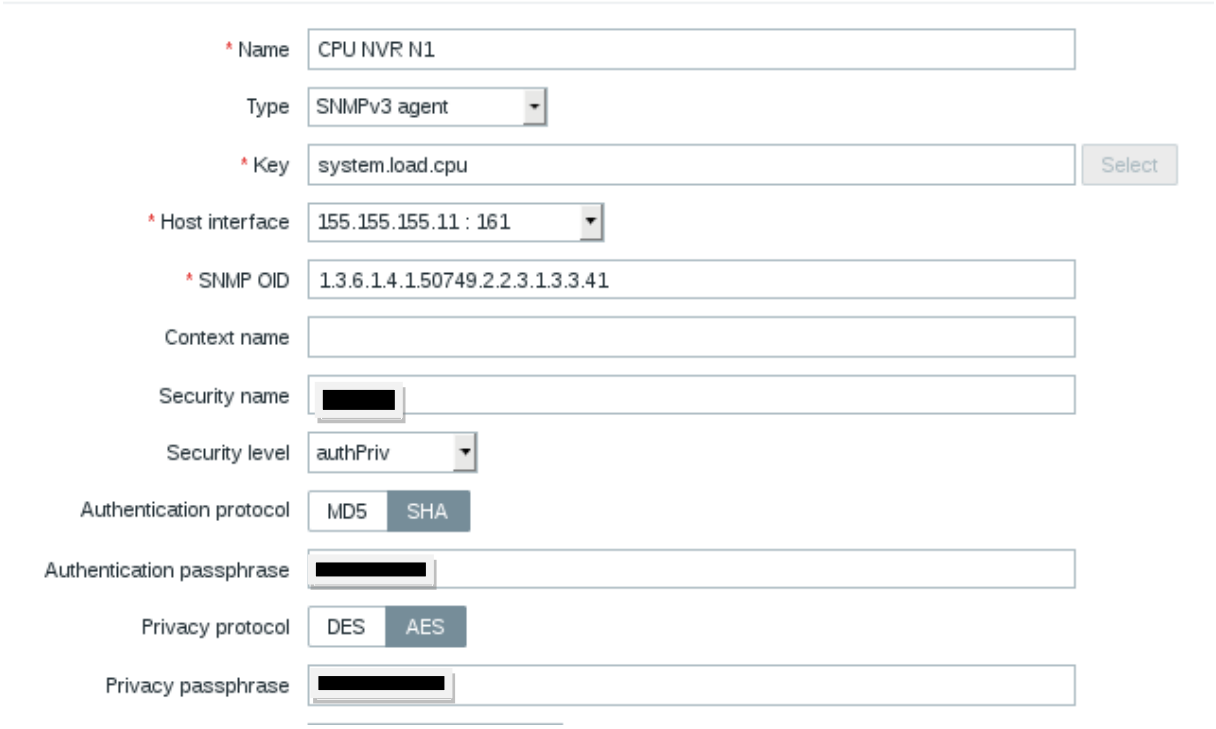
Il existe quatre types de requêtes : GetRequest, GetNextRequest, GetBulk, SetRequest.

- La requête GetRequest permet la recherche d'une variable sur un agent.
- La requête GetNextRequest permet la recherche de la variable suivante.
- La requête GetBulk permet la recherche d'un ensemble de variables regroupées.
- La requête SetRequest permet de changer la valeur d'une variable sur un agent.

A la suite de requêtes, l'agent répond toujours par un GetResponse. Toutefois si la variable demandée n'est pas disponible sur l'agent, la GetResponse sera accompagné d'une erreur noSubjectObject.
Les alertes sont envoyées lorsqu'un évènement non attendu se produit sur l'agent. Celui-ci en informe la station de supervision via un trap. Les alertes possibles sont : ColdStart, WarmStart, LinkDown, Linkup et AuthentificationFailure.

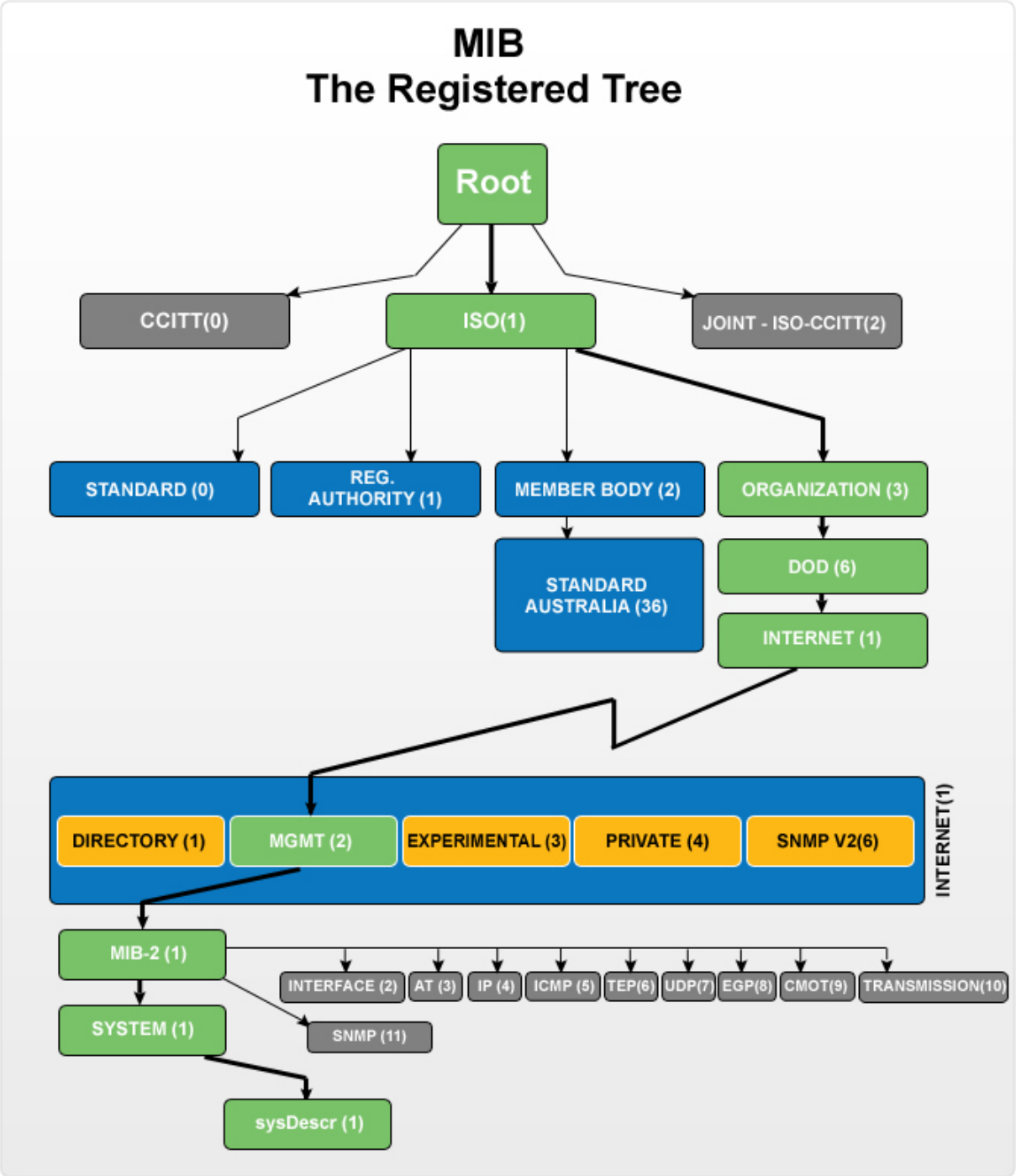
La MIB est la base de données des informations de gestion maintenue par l'agent, auprès de laquelle le manager va venir questionner l'agent.
La MIB est une structure arborescente dont chaque nœud est défini par un OID. Elle contient une partie commune à tous les agents SNMP en général, une partie commune à tous les agents SNMP d'un même type de matériel et une partie spécifique à chaque constructeur. Chaque équipement à sa propre MIB. Non seulement la structure est normalisée, mais également les appellations des diverses rubriques.
Durant mon installation j'ai dû superviser la charge CPU des C4P et du NVR, à l'aide d'une table comme ci-dessous, j'ai donc récupéré les OID nécessaire :

- OID charge CPU du NVR : .1.3.6.1.4.1.50749.2.2.3.1.3.3.41
- OID charge CPU du C4P : .1.3.6.1.4.1.50749.2.2.3.1.3.4.47



Exemple de la configuration d'un hôte

On peut donc voir interface hôte qui correspond à notre équipement à atteindre et l'OID du paramètre qui nous intéresse.



Exemple Arborescence MIB

Donc pour interroger les différentes variables d'activité sur un appareil, il faudra explorer son arborescence MIB. Celle-ci est généralement fournie par le constructeur mais il est aussi possible d'utiliser un explorateur de MIB tel que « GetIf MIB Browser ». Pour accéder aux variables souhaitées, on utilisera l'OID qui désigne l'emplacement de la variable à consulter dans le MIB.

3.2 Outil de Supervision Zabbix

Pour réaliser notre supervision réseau nous allons utiliser Zabbix, un outil de supervision qui va nous permettre, grâce à une interface graphique, de mieux analyser les résultats de nos requêtes SNMP. Dans un premier temps nous allons voir ce qu'est Zabbix puis comment il fonctionne.

Il existe divers outils de supervision réseau, plus ou moins complexes et incluant plus ou moins les mêmes fonctionnalités. La base étant d'avoir une vue sur l'ensemble des équipements surveillés. Certains outils, tels que Zabbix, assurent bien plus de fonctionnalités.
Zabbix est un outil de supervision réseau, et à ce titre, il permet de surveiller l'ensemble d'une infrastructure réseau. Son interface utilisateur est une interface web, s'exécutant donc sur un serveur HTTP. Il n'y a pas besoin de logiciel particulier côté client pour pouvoir bénéficier de Zabbix, un simple « navigateur internet » suffit. Etant relativement léger, il peut être installé sur un serveur peu performant, ou sur un serveur utilisé pour un autre service, mais par mesure de sécurité il est préférable de ne pas le laisser accessible depuis l'extérieur.

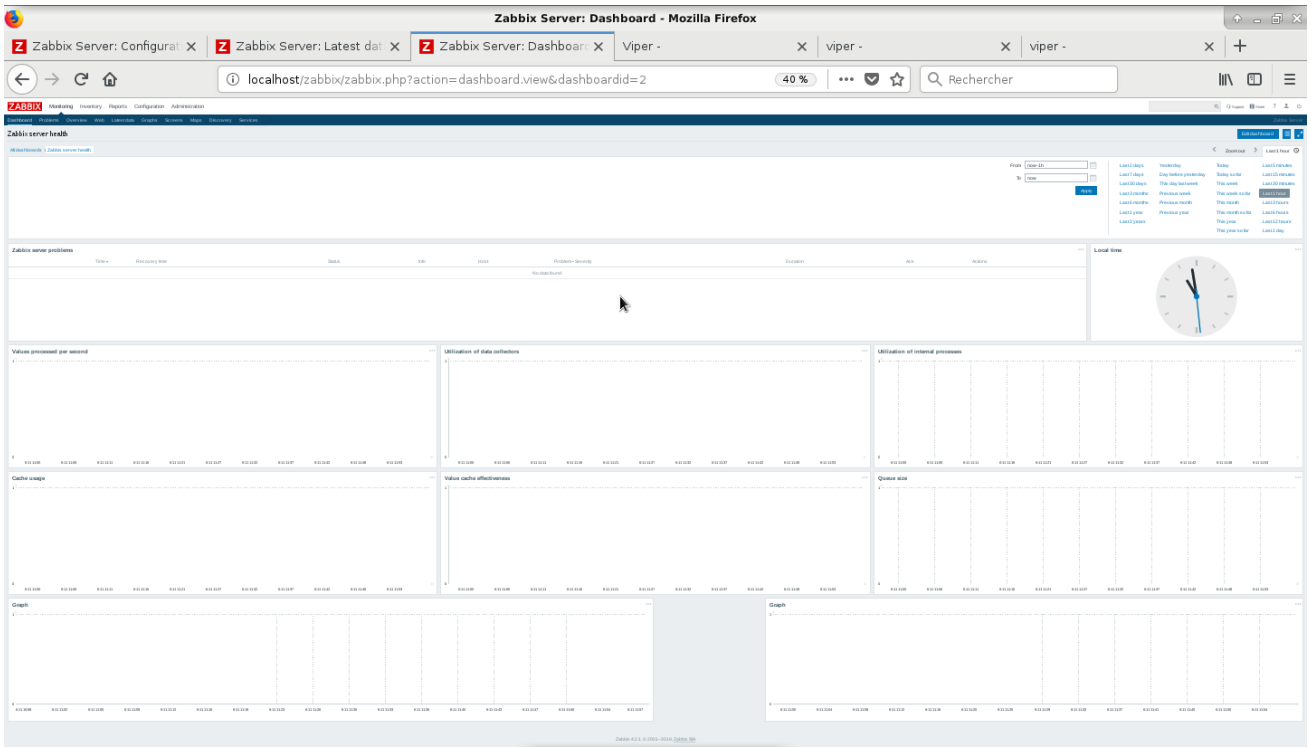
Maintenant nous allons voir comment fonctionne Zabbix.
Comme dit précédemment, Zabbix à une partie « serveur » qui va nous servir à questionner nos équipements.
Comme la plupart des outils de supervision Zabbix est capable d'utiliser le protocole SNMP, afin de récolter des données, mais il est aussi capable de récupérer des informations via d'autres protocoles comme JMX* ou IPMI*. Dans notre cas nous allons récupérer les données des équipements à l'aide de SNMP.
Concernant les alertes SNMP, Zabbix reçoit les paquets « trap » transmis par les équipements, dans lesquels se trouvent diverses informations, dont le message d'erreur ou d'information à remonter. La difficulté est que le protocole SNMP fonctionne avec des OID prédéfinis, qui ne sont pas forcément les mêmes pour chaque équipement, voir même ils peuvent être différent d'un numéro de série à l'autre d'un même fournisseur (mais c'est plus rare).

```
iso.3.6.1.2.1.1.1.0 = STRING: "Linux dlink-2973CB 2.6.31.8 #1 Wed Aug 22 16:55:05 CST 2012 armv5tel"
iso.3.6.1.2.1.1.2.0 = OID: iso.3.6.1.4.1.8072.3.2.10
iso.3.6.1.2.1.1.3.0 = Timeticks: (1204520) 3:20:45.20
iso.3.6.1.2.1.1.4.0 = STRING:
iso.3.6.1.2.1.1.5.0 = STRING: "dlink-2973CB"
iso.3.6.1.2.1.1.6.0 = STRING: "DNS-320"
iso.3.6.1.2.1.1.8.0 = Timeticks: (3) 0:00:00.03
iso.3.6.1.2.1.1.9.1.2.1 = OID: iso.3.6.1.6.3.11.3.1.1
iso.3.6.1.2.1.1.9.1.2.2 = OID: iso.3.6.1.6.3.15.2.1.1
iso.3.6.1.2.1.1.9.1.2.3 = OID: iso.3.6.1.6.3.10.3.1.1
iso.3.6.1.2.1.1.9.1.2.4 = OID: iso.3.6.1.6.3.1
iso.3.6.1.2.1.1.9.1.2.5 = OID: iso.3.6.1.2.1.49
iso.3.6.1.2.1.1.9.1.2.6 = OID: iso.3.6.1.2.1.4
iso.3.6.1.2.1.1.9.1.2.7 = OID: iso.3.6.1.2.1.50
iso.3.6.1.2.1.1.9.1.2.8 = OID: iso.3.6.1.6.3.16.2.2.1
iso.3.6.1.2.1.1.9.1.2.9 = OID: iso.3.6.1.6.3.13.3.1.3
iso.3.6.1.2.1.1.9.1.2.10 = OID: iso.3.6.1.2.1.92
iso.3.6.1.2.1.1.9.1.3.1 = STRING: "The MIB for Message Processing and Dispatching."
iso.3.6.1.2.1.1.9.1.3.2 = STRING: "The management information definitions for the SNMP User-based Secu
rity Model."
iso.3.6.1.2.1.1.9.1.3.3 = STRING: "The SNMP Management Architecture MIB."
iso.3.6.1.2.1.1.9.1.3.4 = STRING: "The MIB module for SNMPv2 entities"
iso.3.6.1.2.1.1.9.1.3.5 = STRING: "The MIB module for managing TCP implementations"
iso.3.6.1.2.1.1.9.1.3.6 = STRING: "The MIB module for managing IP and ICMP implementations"
iso.3.6.1.2.1.1.9.1.3.7 = STRING: "The MIB module for managing UDP implementations"
iso.3.6.1.2.1.1.9.1.3.8 = STRING: "View-based Access Control Model for SNMP."
iso.3.6.1.2.1.1.9.1.3.9 = STRING: "The MIB modules for managing SNMP Notification, plus filtering."
iso.3.6.1.2.1.1.9.1.3.10 = STRING: "The MIB module for logging SNMP Notifications."
iso.3.6.1.2.1.1.9.1.4.1 = Timeticks: (0) 0:00:00.00
iso.3.6.1.2.1.1.9.1.4.2 = Timeticks: (0) 0:00:00.00
iso.3.6.1.2.1.1.9.1.4.3 = Timeticks: (0) 0:00:00.00
iso.3.6.1.2.1.1.9.1.4.4 = Timeticks: (1) 0:00:00.01
iso.3.6.1.2.1.1.9.1.4.5 = Timeticks: (1) 0:00:00.01
iso.3.6.1.2.1.1.9.1.4.6 = Timeticks: (1) 0:00:00.01
iso.3.6.1.2.1.1.9.1.4.7 = Timeticks: (1) 0:00:00.01
iso.3.6.1.2.1.1.9.1.4.8 = Timeticks: (1) 0:00:00.01
iso.3.6.1.2.1.1.9.1.4.9 = Timeticks: (2) 0:00:00.02
iso.3.6.1.2.1.1.9.1.4.10 = Timeticks: (3) 0:00:00.03
iso.3.6.1.2.1.2.1.0 = INTEGER: 5
iso.3.6.1.2.1.2.2.1.1.1 = INTEGER: 1
iso.3.6.1.2.1.2.2.1.1.2 = INTEGER: 2
iso.3.6.1.2.1.2.2.1.1.3 = INTEGER: 3
iso.3.6.1.2.1.2.2.1.1.4 = INTEGER: 4
iso.3.6.1.2.1.2.2.1.1.5 = INTEGER: 5
iso.3.6.1.2.1.2.2.1.2.1 = STRING: "lo"
iso.3.6.1.2.1.2.2.1.2.2 = STRING: "egiga0"
iso.3.6.1.2.1.2.2.1.2.3 = STRING: "sit0"
iso.3.6.1.2.1.2.2.1.2.4 = STRING: "ip6tnl0"
iso.3.6.1.2.1.2.2.1.2.5 = STRING: "tunl0"
iso.3.6.1.2.1.2.2.1.3.1 = INTEGER: 24
iso.3.6.1.2.1.2.2.1.3.2 = INTEGER: 6
iso.3.6.1.2.1.2.2.1.3.3 = INTEGER: 131
iso.3.6.1.2.1.2.2.1.3.4 = INTEGER: 131
iso.3.6.1.2.1.2.2.1.3.5 = INTEGER: 131
iso.3.6.1.2.1.2.2.1.4.1 = INTEGER: 16436
iso.3.6.1.2.1.2.2.1.4.2 = INTEGER: 1500
iso.3.6.1.2.1.2.2.1.4.3 = INTEGER: 1480
```

Exemple d'OID

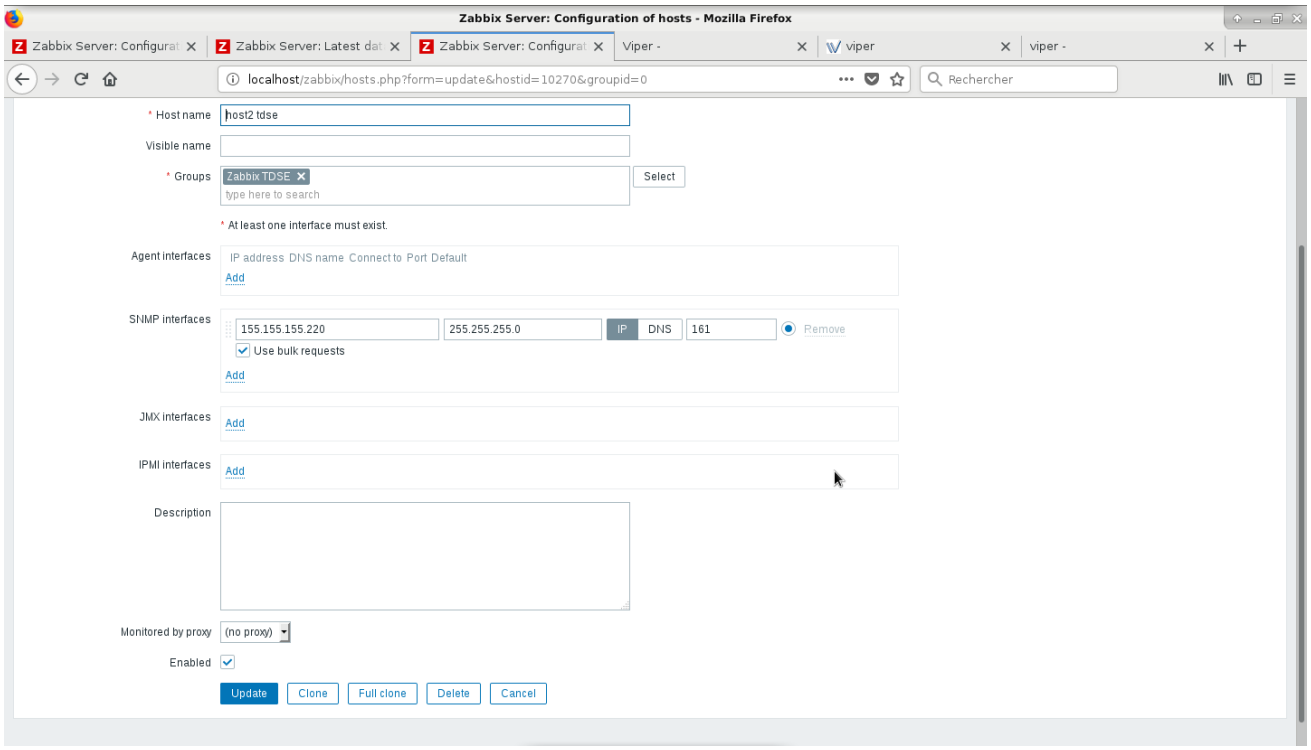
Cela dit, ce protocole reste très intéressant car la quasi-totalité des équipements réseau répondent aux requêtes SNMP. Il existe aussi un autre moyen de récupérer des données avec Zabbix : un outil s'appelant Zabbix Agent. Cet outil tourne en arrière-plan et communique régulièrement avec le serveur Zabbix. Contrairement au SNMP qui peut être plus complexe à mettre en place sur un serveur, les agents sont plus faciles à installer sur des serveurs. On peut donc avoir accès à énormément de données, telles que l'utilisation de la bande passante sur chaque carte réseau, etc….
La limite de ce système est qu'il n'est pas envisageable de l'installer sur un équipement de type routeur ou Switch, là où SNMP reste l'outil nous permettant de récolter des données sur ce genre d'équipement. Les Agents Zabbix sont disponibles sur la plupart des plateformes : Linux, Windows…

Après avoir détaillé le fonctionnement global de Zabbix, maintenant nous allons voir toutes les particularités de l'interface de Zabbix côté serveur.
Le tableau de bord de Zabbix est un outil très pratique qui permet de regrouper un ensemble d'informations. Il est possible d'en créer plusieurs et de modifier le nombre, les positions et les tailles des éléments à afficher (déplacer les éléments, en ajouter, mettre des favoris …). Chaque tableau de bord nécessite du temps pour sa mise en place mais une fois configuré, il est très facile de trouver les données qui peuvent nous intéresser. Il est également possible de créer des « maps » (cartes) qui représentent l'architecture réseau dans laquelle on se trouve.

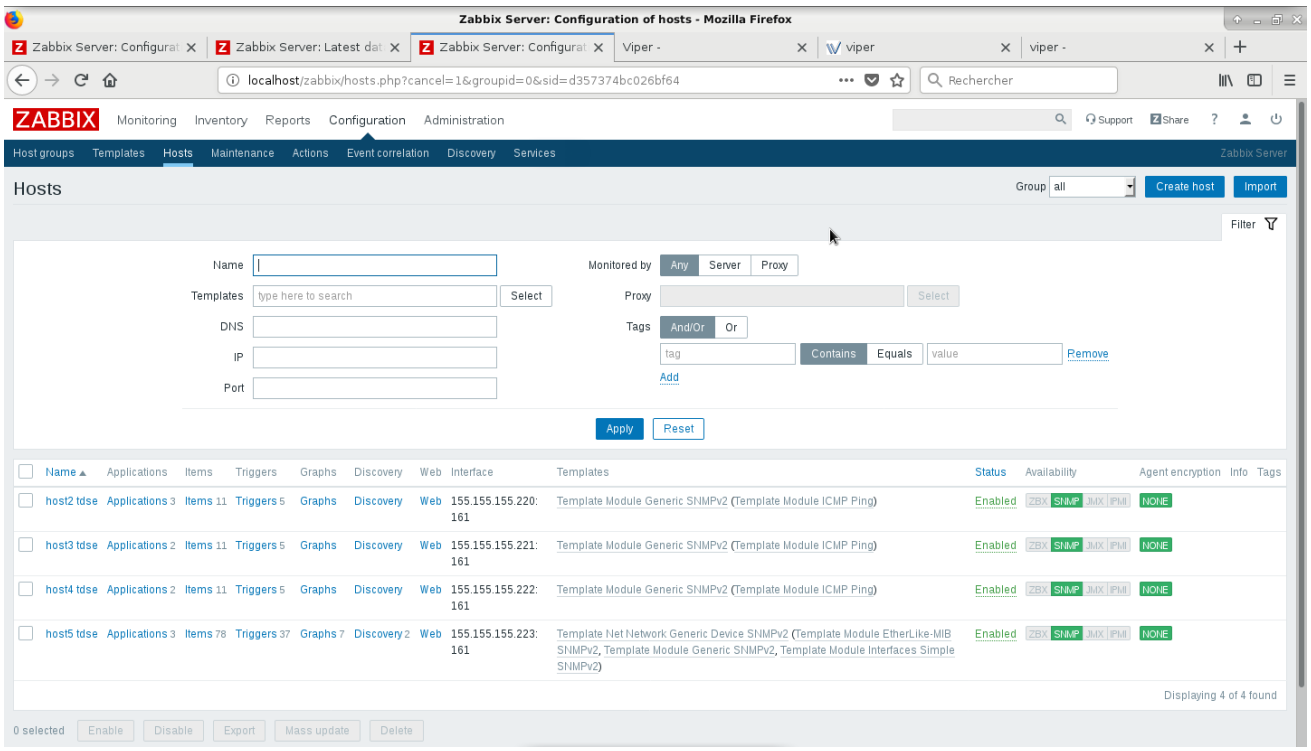


Exemple de tableau de bord Zabbix

Il y a ensuite la création d'hôte, qui est simplifié par Zabbix. Un hôte est un terme qui décrit un équipement réseau comme un PC, un périphérique est l'hôte d'un serveur quand il est raccordé en tant que client (équipement où le serveur récupère des données). Ici, pas besoin de passer par édition/création de fichiers pour créer un hôte à surveiller, comme nous aurions à le faire avec Nagios* par exemple. Il n'y a qu'à choisir le type de communication (Agent Zabbix, SNMP, …) et remplir les différents champs requis puis valider. Lorsque la connexion SNMP sera établie entre l'équipement et le serveur Zabbix l'icone SNMP passera au vert, s'il y a une erreur il passera au rouge. Il en va de même pour les autres protocoles (Agent Zabbix, IPMI et JMX).

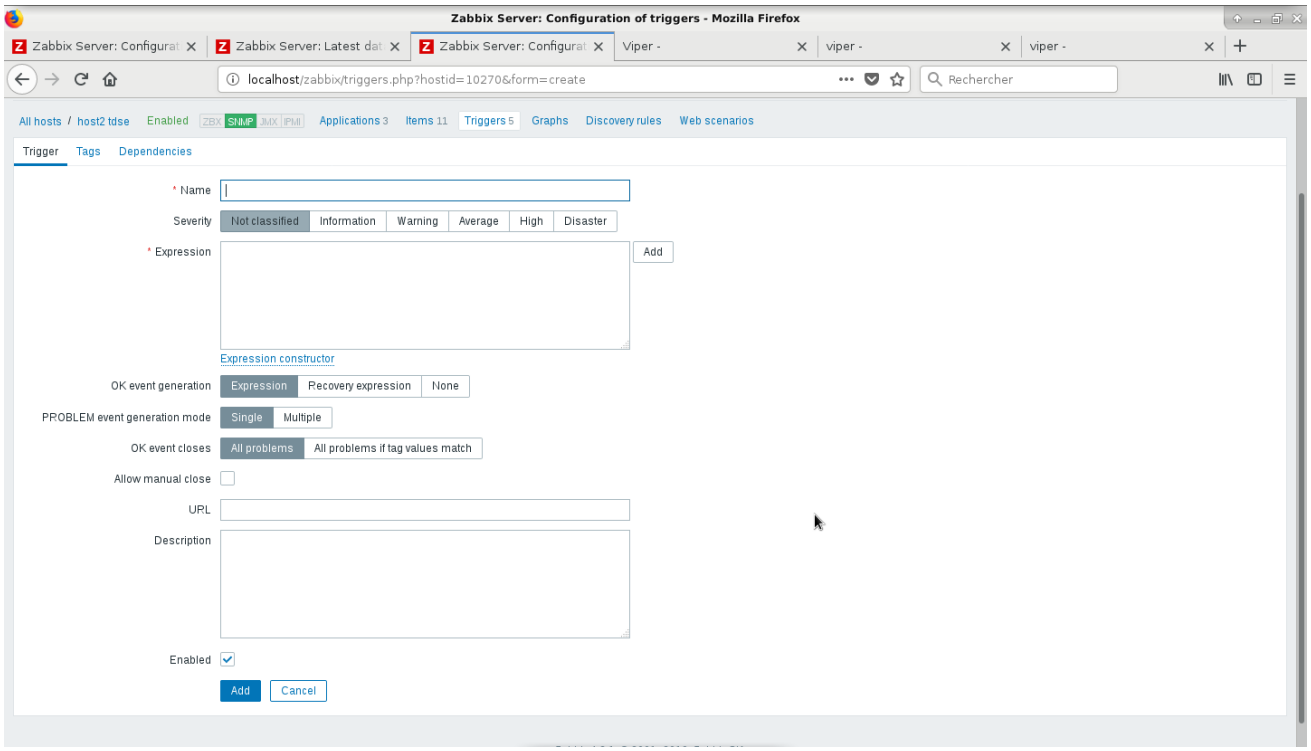


Exemple de Création d'un hôte



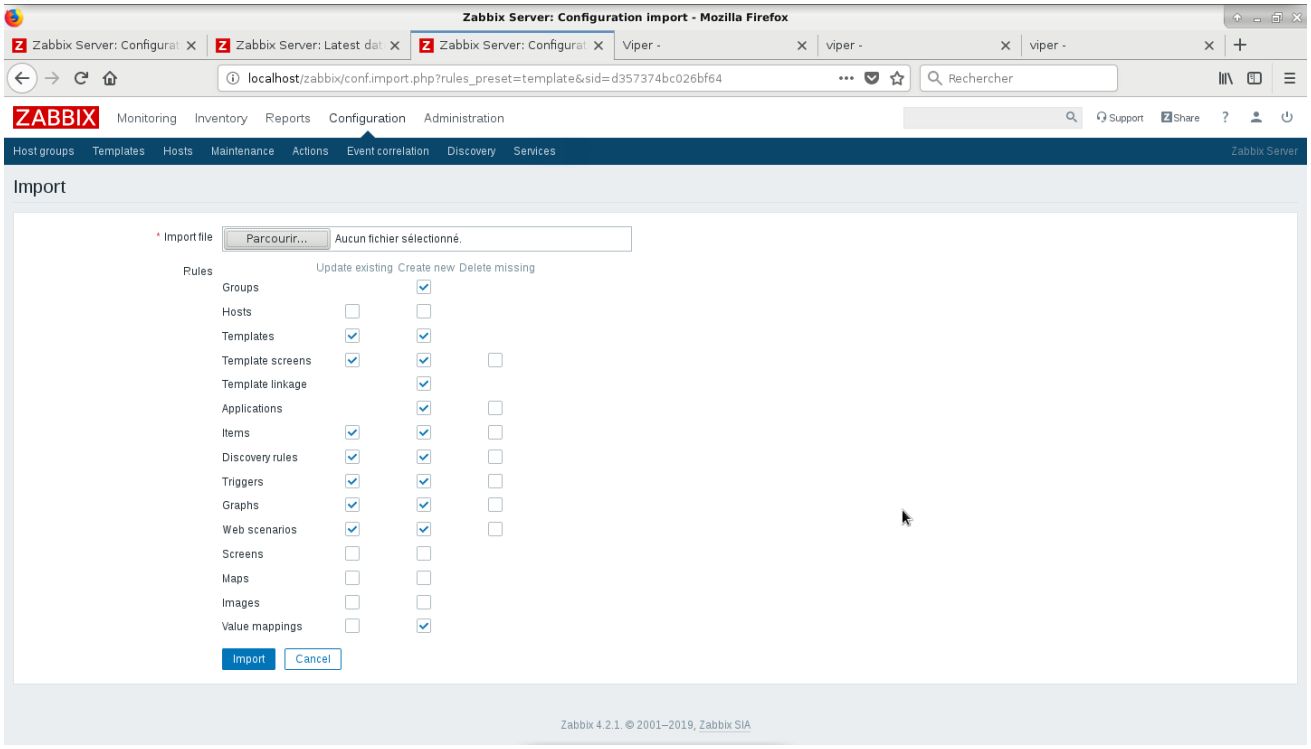
Hôtes Zabbix ou les ports sont verts

Il y a la création des « Triggers » (déclencheurs), sur le même principe de création d'hôte, on peut créer assez facilement des déclencheurs. Un déclencheur apparaît dans le tableau des erreurs si sa condition est vérifiée.



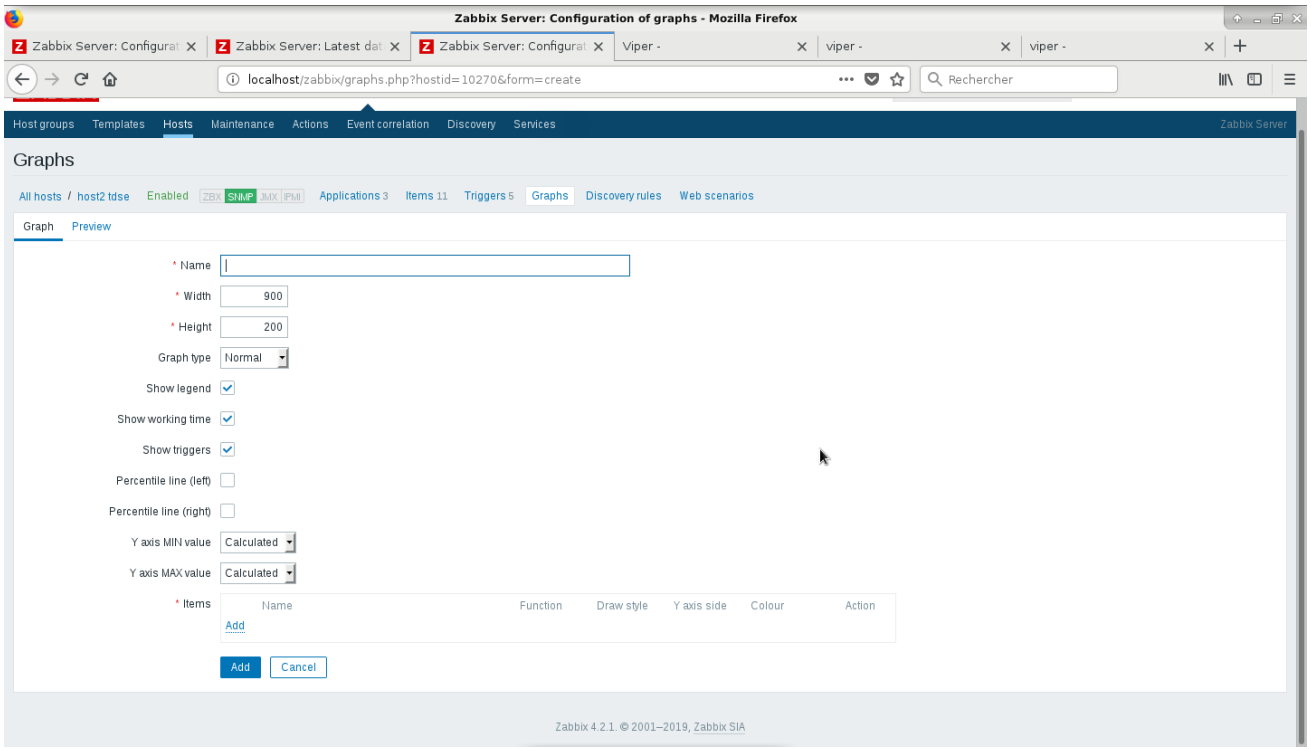
Exemple de Création de Triggers

Il est possible d'importer une liste d'hôtes/templates, cela peut être utile lors de la migration d'un serveur Zabbix par exemple. On peut sauvegarder sa liste d'hôtes, pour l'importer sur le nouveau serveur Zabbix. Il y a aussi la possibilité d'importer des templates d'équipements, fichiers rassemblant une liste de triggers. Cela peut être pratique lorsque notre équipement est sous SNMP et qu'on ne sait pas quoi correspond tous les OID. On peut aussi récupérer des Templates créer par la communauté.



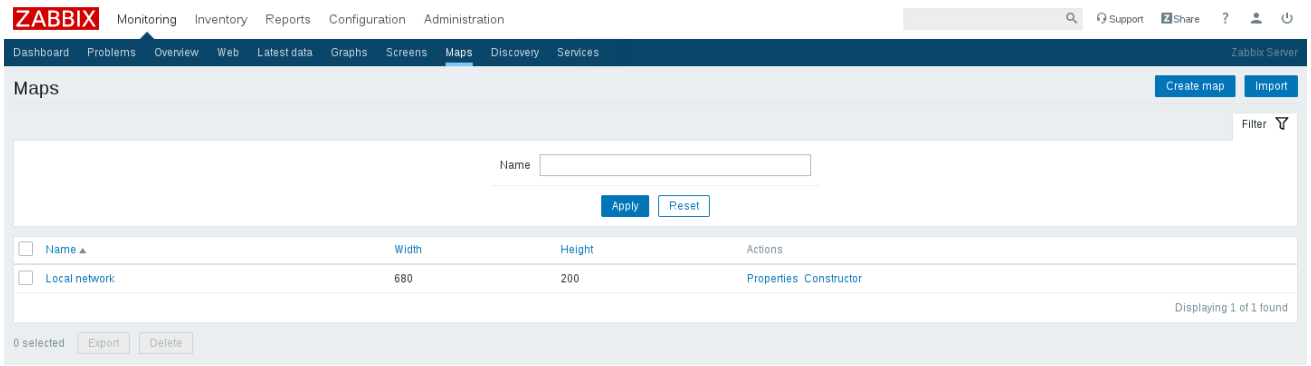
Exemple d'import de Template

Puis il y a les graphiques, cartes et diaporamas, Zabbix propose de base de nombreux graphiques remplis avec les informations des triggers de templates présents, mais offre aussi la possibilité de créer ses propres graphiques et cartes.



Exemple de Création d'un graph

Une carte permet de représenter son infrastructure réseau avec des éléments dynamiques, comme la couleur des liens selon leur état des valeurs de bande passante… Zabbix possède de base un certain nombre d'images pour nos cartes mais on peut en importer d'autres si on le souhaite.

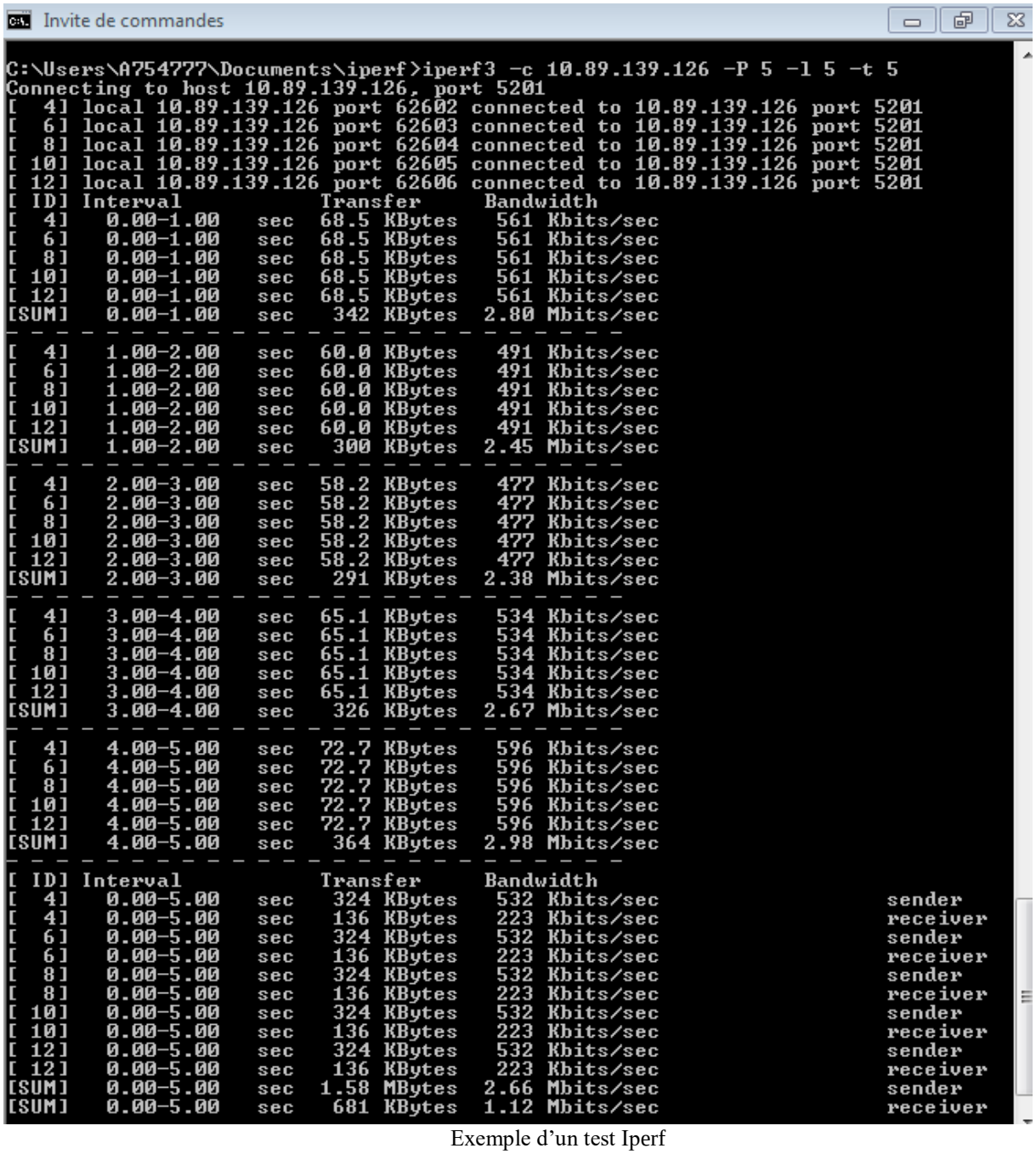


Exemple de Création d'une carte

Nous avons pu voir que Zabbix, au travers de ses différentes fonctionnalités, est relativement efficace, léger et simple d'utilisation et de configuration. Il a l'avantage d'être sous licence libre, et d'avoir une communauté active, qui partage facilement des astuces, plugins et autres templates.
Que ce soit appliqué à l'échelle d'un petit réseau ou d'une grosse architecture, Zabbix propose une gestion des équipements suffisamment simple et structurée. De plus il est gratuit.

3.3 Mise en place sur le Banc de Test

Pendant mon stage j'ai donc dû faire passer des tests au banc de test et à mes différents équipements. J'ai utilisé différents moyens, dans un premier temps j'ai utilisé Iperf qui est un outil qui génère des charges avec différentes options. Il prend en charge le réglage de divers paramètres liés à la synchronisation, aux tampons et aux protocoles de transmission (TCP*, UDP*…). Pour chaque test, il indique la bande passante, la perte et d'autres paramètres. Pour faire passer des tests il faut préalables rédiger une fiche de spécifications. J'ai rédigé et exécuté cette fiche de spécifications pour tester une petite architecture réseau (qui ne faisait pas partie du projet global mais qui était une infrastructure de test). La particularité d'Iperf c'est qu'il n'a pas « d'IHM élaborée », tout se fait par ligne de commandes.
Les commandes peuvent être tapées une par une mais on peut aussi écrire un script afin de lancer plusieurs commandes, mais il existe des outils comme Zabbix qui nous offre une facilité d'utilisation et de lisibilité de part l'interface graphique web.



Exemple d'un test Iperf

Après avoir expérimenter les différentes options d'Iperf, comme Iperf n'est pas le meilleur choix pour la supervision globale car ne traite pas toutes les options possibles et qu'ils auraient fallu compléter avec des requêtes SNMP. Je me suis donc penché sur Zabbix qui correspond plus à nos attentes. Il m'a donc fallu installer Zabbix sur un PC qui sera notre poste de supervision sur lequel on analysera les résultats des requêtes SNMP.

Maintenant nous allons voir l'installation de Zabbix, d'abord il faut qu'on installe le dépôt de Zabbix

```
# wget https://repo.zabbix.com/zabbix/4.2/debian/pool/main/z/zabbix-
release/zabbix-release_4.2-1+stretch_all.deb
# dpkg -i zabbix-release_4.2-1+stretch_all.deb
# apt update
```

Ensuite il faut qu'on installe Zabbix Server et le frontend Agent.

```
# apt -y install zabbix-server-mysql zabbix-frontend-php zabbix-agent
```

Après il faut créer et configurer une base de données,

```
# mysql -uroot -p
password
mysql> create database zabbix character set utf8 collate utf8_bin;
mysql> grant all privileges on zabbix.* to zabbix@localhost identified
by 'password';
mysql> quit;
```

Il nous faut importer le schéma et les données initiaux. Puis nous rentrerons un nouveau mot de passe.

```
# zcat /usr/share/doc/zabbix-server-mysql*/create.sql.gz | mysql -
uzabbix -p zabbix
```

Maintenant nous pouvons configurer la base de données, il faut aller dans /etc/zabbix/zabbix_server.conf et modifier comme dessous

```
DBPassword=password
```

Il faut configurer le PHP du frontend dans /etc/zabbix/apache.conf il faut décommenter et mettre le fuseau horaire que vous voulez.

```
# php_value date.timezone Europe/Riga
```

Un redémarrage du serveur et de l'agent sont nécessaire pour les mettre à jour

```
# systemctl restart zabbix-server zabbix-agent apache2
# systemctl enable zabbix-server zabbix-agent apache2
```

Ensuite pour aller sur Zabbix il suffit de taper dans un navigateur : http:// « ip_du_serveur »/zabbix
Dans notre cas l'adresse IP sera localhost ou 127.0.0.1. Maintenant nous pouvons utiliser Zabbix et configurer les hôtes et items qu'il nous faut.

Pour pouvoir utiliser SNMP, il faut installer les paquets SNMP sur le PC de supervision.
Pour cela, il faut premièrement éditer les sources.list puis décommenter les lignes multiverse qui se situent dans /etc/apt/sources.list afin de pouvoir installer les paquets qui nous intéressent.

 # nano /etc/apt/sources.list ensuite il faut décommenter les lignes.

Il faut ensuite, mettre à jour les paquets apt

 # apt-het update

Puis installer les paquets nécessaires.

 # apt-get install snmp snmp-mibs-downloader

Ensuite, il faut éditer snmp.conf et décommenter la ligne MIBs.

 # nano /etc/snmp/snmp.conf et décommenter la ligne.

Il faut ensuite installer les librairies MIVB pour SNMP.

 # download-mibsmibs

Enfin il faut redémarer le serveur Zabbix.

 # service zabbix-server restart

Maintenant il est possible de créer des hôtes SNMP et de faire tous les tests demandés.

Ce PC sert également de serveur Zabbix mais aussi de PC Superviseur dans l'architecture des cas 1 et 2 (de la partie 2.1 Elaboration d'une Architecture du réseaux MAV).
J'ai donc effectué, dans Zabbix, l'installation des hôtes comme vu précédemment. Ensuite j'ai mis en place un tableau de bord pour simplifier la lisibilité des tests à passer.

4 Résultats

4.1 Déroulement globale du stage

Le déroulement du stage n'a pas forcément été le même ordre que celui de mon rapport, mais dans sa globalité j'ai dû travailler sur mon projet ou sinon sur les autres missions réalisées avaient pour but de mieux me faire comprendre certains points de mon projet. Comme par exemple, la mise en place d'un réseau destiné à un véhicule de type bus à l'aide d'un router wifi et d'un switch, cela m'a permis de travailler sur la partie Wifi et de comprendre la complexité de la mise en place d'un réseau wifi dans le transport.
Durant tout ce stage, j'ai travaillé en autonomie : j'avais une mission que je devais remplir pour atteindre mes objectifs. J'ai navigué entre mon réseau de test et la baie SVE.

4.2 Travail réalisé

Durant ce stage, j'ai réalisé différentes parties de ce projet, de l'architecture à la supervision du réseau, en passant par la configuration des équipements. Mais certains choix n'étaient pas de mon ressort comme par exemple l'architecture globale du réseau ou encore le matériel imposé par le client. Le choix de l'outil de supervision Zabbix m'a également été imposé.

Dans un premier temps, j'ai dû comprendre le fonctionnement du Banc SVE et son architecture en utilisant les fichiers de configurations des Routeurs C12P et C4P, pour réaliser le schéma de l'infrastructure, première mission de ce stage. Sur ce schéma il fallait pouvoir voir le niveau dit « physique » c'est-à-dire les ports auxquels sont branchés les routeurs entre eux, ensuite le niveau Vlan et adressage IP.
Après avoir fourni mon schéma, j'ai donc travaillé sur la rédaction sur une fiche de spécificités (c'est-à-dire que c'est une fiche qui va détailler les différentes caractéristiques que notre réseau doit respecter) Pour ensuite passer à la phase de test d'un réseau test (composé de routeurs et switchs).
Pour réaliser ces tests, j'ai utilisé Iperf en faisant varier différents paramètres, que ce soit le nombre de connexions simultanées, la taille du paquet ou encore le protocole utilisé (TCP/UDP).
Pour les tests de la supervision je suis passé sur l'outil Zabbix qui offre plus d'options et une interface graphique ergonomique et qui facilite la lecture des résultats des différentes requêtes. J'ai donc installé Zabbix sur notre PC de supervision, puis je l'ai placé sur le banc de test SVE.

Pour pouvoir faire les tests, il fallait créer la simulation du banc TDSEe. Pour cela j'ai utilisé les trois routeurs Westermo Viper-212-A mis à ma disposition. J'ai commencé à étudier le banc de test TDSEe en analysant comme l'autre banc les fichiers de configurations du routeur. Il existe différentes versions de cette baie, j'ai donc testé ces différentes versions en simulant les différentes configurations sur mes routeurs. Puis je suis passé à l'installation de mon infrastructure sur la baie de test SVE.
Après avoir réglé quelques problèmes de configuration et avoir terminé l'installation du PC de supervision, j'ai pu tester mon réseau et mettre la supervision en place. Je devais superviser la charge CPU des C4P et des NVR. A l'aide d'OID qu'on m'a fourni.

Pour clôturer mon stage j'ai formé une partie des personnes travaillant dans l'open-space à mon infrastructure et à l'utilisation de Zabbix. En support de cette formation, j'ai rédigé et fourni un document dans lequel j'explique comment utiliser l'infrastructure et comment elle fonctionne.

5 Conclusion

Pendant mon stage j'ai eu l'occasion de travailler sur un projet du début à la fin, de la conception à la simulation en passant par les phases de tests et jusqu'à l'installation et la formation du personnel sur l'infrastructure.

En effet, j'ai pu suivre les différentes étapes nécessaires à la construction et validation d'un projet dans une entreprise. J'ai donc dû étudier et faire appel à mes connaissances pour schématiser le réseau du banc de test. Après cela, il a fallu que je simule un autre réseau en miniature afin de pouvoir l'intégrer sur le banc. Cela m'a permis de développer mes connaissances en architecture réseau et mon expertise dans ce domaine.

Les connaissances acquises à l'IUT m'ont été très utiles pour l'installation et la configuration d'équipements. Cela m'a permis de mieux comprendre et de pouvoir approfondir certaines des compétences qui m'ont été enseignées, comme la partie supervision réseau à l'aide de SNMP où j'ai appris à utiliser un logiciel de supervision.

Ce stage m'a fait découvrir le fonctionnement d'une entreprise et m'a permis de me faire une idée plus précise de ce qu'est le travail au sein d'une structure avec plusieurs collaborateurs.
Il a donc été une bonne expérience professionnelle et humaine, ce qui sera un atout dans le cadre de la poursuite de mes études.

Ce stage confirme mon choix projet professionnel, ayant fait de l'architecture réseaux, il m'a permis d'affirmer ma décision. Il m'a également offert la possibilité de poursuivre dans la licence pro ASSR en alternance au sein de la même entreprise.
Enfin, ce stage a permis de compléter ma formation en IUT en m'ouvrant au monde professionnel.

6 Remerciements

Je tiens tout particulièrement à remercier mon responsable de stage Clément Guezou pour son accueil, ainsi que Olivier Discours, de m'avoir accepté en tant que stagiaire au sein de son service.

Je remercie également toute l'équipe du service Transport qui m'ont aidé durant mon stage. Je souhaite les remercier pour l'intérêt qu'ils ont porté à mon travail tout au long de mon stage ainsi que pour leur aide et leur enseignement.

Je remercie de même mon tuteur de stage pour son encadrement pendant celui-ci.

7 Glossaire

MAV, Moyen Audio-Visuels

TDSEe, Transmission des Données Sol-Embarqué

SVE, Système Vidéo Embarqué

C12P, Commutateur douze ports

C4P, Commutateur quatre ports

NAT, Network Address Translation (traduction d'adresse réseau)

SSH, Secure Shell (connexion à distance sécurisé)

SNMP, Simple Network Management Protocol

OID, Object Identifier

MIB, Management Information Base (Base d'Information de Gestion)

IETF, Internet Engineering Task Force

Router, Routeur

Switch, Commutateur

JMX, Java Management Extension

IPMI, Intelligent Platform Management Protocol

8 Bibliographie

Iperf

https://iperf.fr/

Zabbix

https://www.zabbix.com/download

Westermo

https://www.westermo.fr/products/ethernet-switches/en50155/viper-212a

SNMP

https://fr.wikipedia.org/wiki/Simple_Network_Management_Protocol

MIB

https://fr.wikipedia.org/wiki/Management_information_base