

# **Rapport d'activité**

# **ROUBAIX**



## **REMERCIEMENT**

Je remercie Monsieur Hubert VINCENT responsable du service technique informatique  
ainsi que Monsieur Rachid SEGHIR-OUALI mon tuteur  
pour l'accueil et la pédagogie dont il a su faire preuve.

Je remercie l'ensemble de l'équipe informatique de la mairie  
pour la confiance et l'implication dans les projets confiés durant ce stage.





## Sommaire

---

1	Introduction	2
2	Mission de stage	3
2.1	Première mission	3
2.2	Création de cartes	4
2.3	Tableau de bord	5
2.4	Notification	6
2.5	Déclencheur	6
3	Seconde mission	7
4	Annexe	8

# 1 *Introduction*

Dans le cadre de mes études de Brevet de Technicien Supérieur Service Informatique aux Organisations, je dois effectuer un stage d'une durée de six semaines en milieu de seconde année dans une entreprise dans le but de compléter mes connaissances théoriques par une mise en situation. J'ai donc réalisé ce premier stage du 5 février au 15 mars à la mairie de la ville de Roubaix.

## 2 Mission de stage

### 2.1 Première mission

Afin de monitorer les équipements du parc informatique, un serveur Zabbix a été mis en place.

Ma mission consiste à améliorer la supervision du serveur Zabbix sur quatre grands axes :

- ⇒ Création de cartes
- ⇒ Création d'un nouveau tableau de bord
- ⇒ Ajout de notification
- ⇒ Supervision d'un port selon son alias

Ainsi que la création d'une procédure à des fins de compréhension et de répliquabilité.

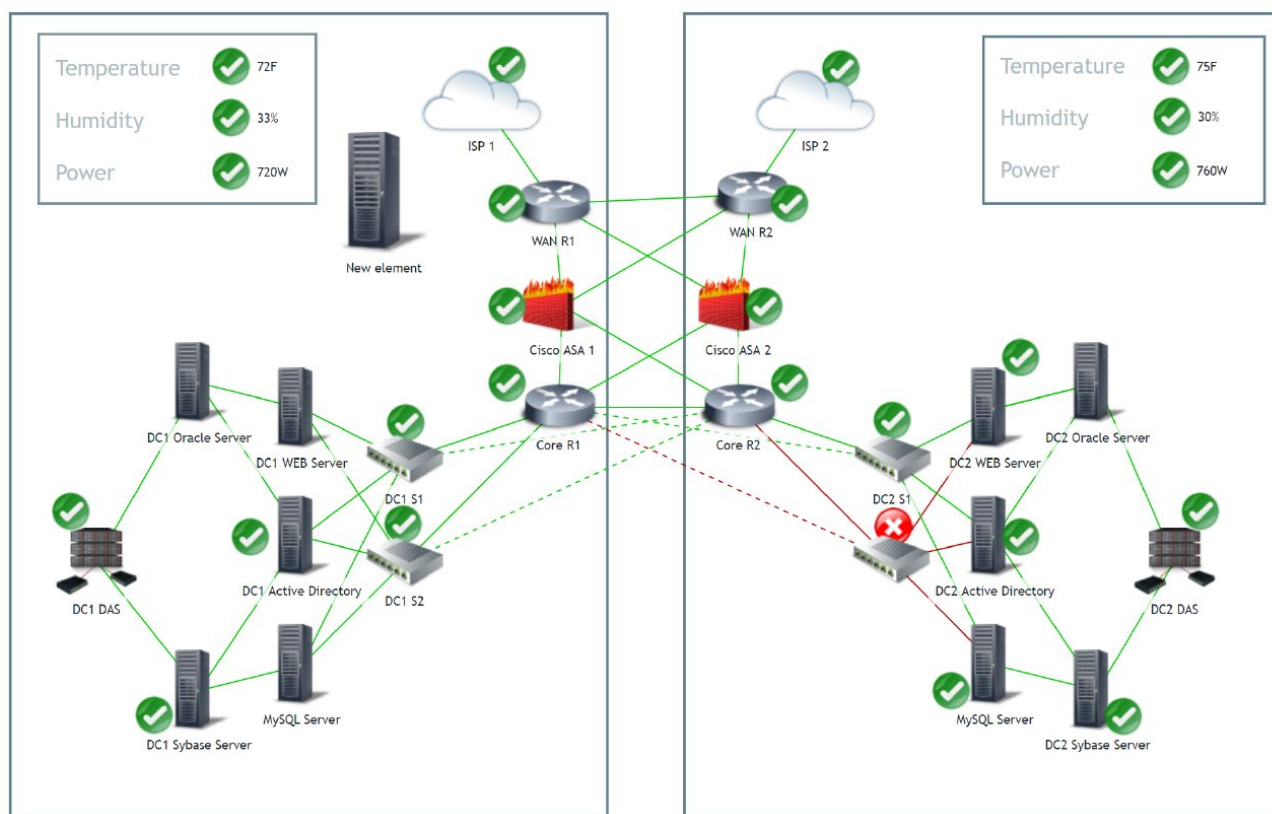


## 2.2 Création de cartes

Le centre technique dispose d'un écran diffusant les différents tableaux de bord de multiples services.

Dans le but d'améliorer la supervision des équipements réseaux, une cartographie des équipements selon leur site sera ajoutée au tableau de bord de Zabbix.

Le détail de la création de cartes est disponible dans l'[Annexe I-I](#).



## 2.3 Tableau de bord

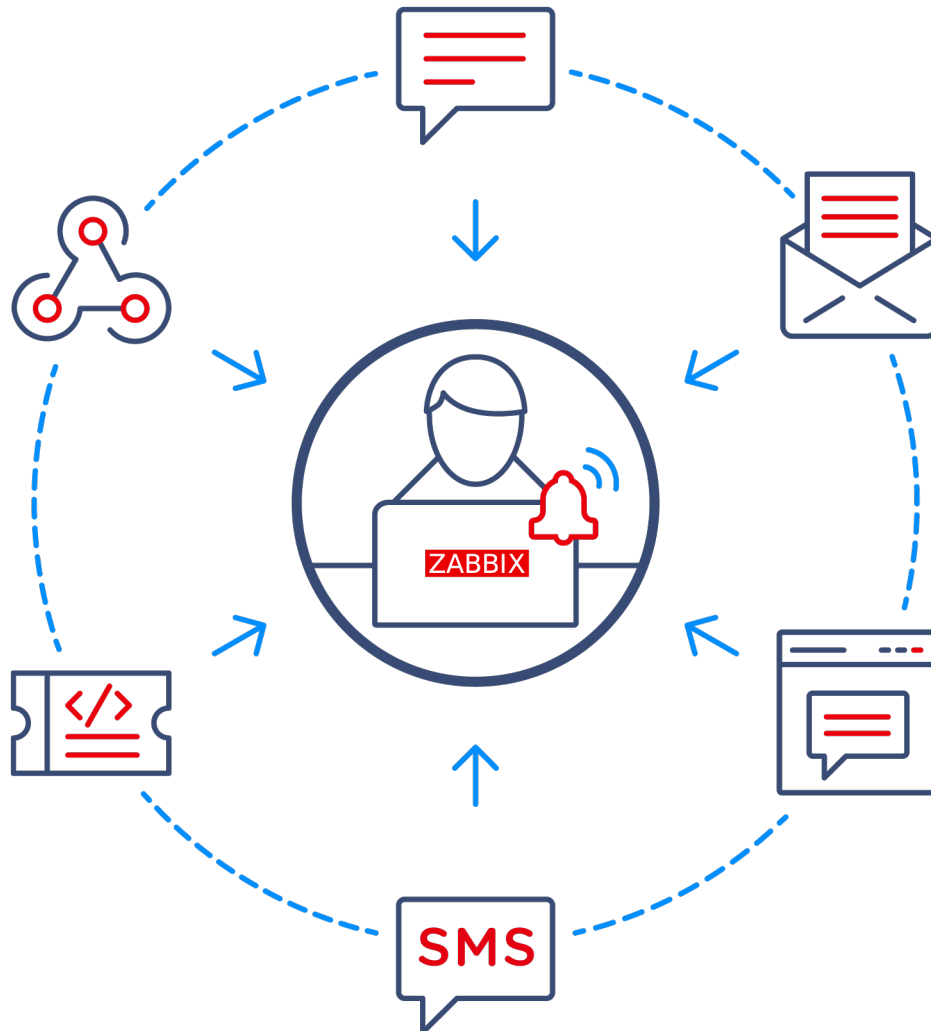
Après leurs créations, les cartes doivent être implémenter dans un tableau de bord consistant en un diaporama entre les cartes et un affichage des problèmes selon un filtre prédéfini.



Détail en [Annexe I-II](#).

## 2.4 Notification

Afin d'alerter les équipes si un incident de niveau grave ou désastre survient, l'ajout de notification sonore et l'envoi d'email est ajouter à la supervision Zabbix.



Détails en [Annexe I-III](#).

## 2.5 Déclencheur

Les commutateurs étant relié aux autres commutateurs ainsi qu'aux postes utilisateurs, seuls les ports d'interconnexion entre commutateur ont une nécessitées à être supervisées.

Dans le but d'optimiser la recherche et la supervision de ces ports l'option retenue est de supervisé les ports procédant une chaîne de caractères précise dans son alias.

Détails en [Annexe I-IV](#).

### 3 Seconde mission

Dans l'objectif de restaurer les configurations des équipement réseaux en cas de panne, la mairie doit se munir d'un système de sauvegarde se connectant aux différents équipements de divers âge et modèles et si possible permettant un versionnage des configurations sauvegardées.

Le choix c'est porté pour la création d'un script en python.

- ⇒ Les modules NetMiko et Telnet permettent la connexion SSH et telnet aux différents équipements.
- ⇒ Le module pyZabbix permet la connexion à l'API de Zabbix permettant de récupérer les informations sur les équipements tel que l'adresse IP, le site, le modèle
- ⇒ Le module GitPython permet de gérer le versionnage des configurations sauvegarder
- ⇒ Le module pyInstaller permet de compiler le code afin de le déployer sur le serveur de sauvegarde

Détail du code dans l'[Annexe II](#).

Fonctionnement du programme dans l'[Annexe III](#).

# Annexe



## Sommaire

---

Annexe I.: Procédure Zabbix	1
I.I Cartes	1
I.I.I Nouvelle carte	1
I.I.II Modification	4
I.I.III Affichage	14
I.I.IV Tableau de bord	16
I.II Tableau de bord	19
I.II.I Nouveau tableau de bord	19
I.II.II Ajouter des widgets	20
I.III Notification	27
I.III.I Son d'alerte	27
I.III.II E-mail	29
I.IV Déclencheurs	34
I.IV.I Déclencheur selon l'alias d'un port (SNMP)	34
Annexe II.: Code Python	41
II.I Main.py	41
II.II switch.py	47
II.III Zabbix.py	54
II.IV gitManager.py	65
II.V logs.py	68
II.VI .env	70
Annexe III.: Fonctionnement du programme	72
III.I Exécutable	72
III.I.I Fonctionnement	72
III.I.II Zabbix	73
III.I.III Fichier environnement .env	75
III.I.IV Logs	76
III.II Développement	77
III.II.I Require	77
III.II.II Compilation	77

# Procédure

**ZABBIX**





## Table des matières

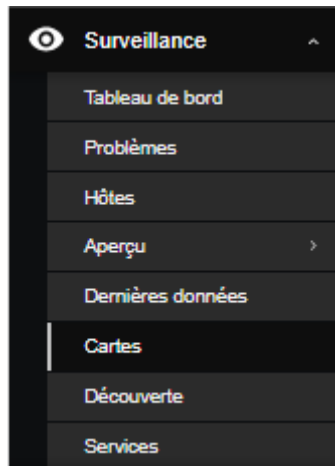
Annexe I: Procédure Zabbix .....	1
I.I Cartes .....	1
I.I.I Nouvelle carte .....	1
I.I.II Modification .....	4
I.I.III Affichage .....	14
I.I.IV Tableau de bord .....	16
I.II Tableau de bord .....	19
I.II.I Nouveau tableau de bord .....	19
I.II.II Ajouter des widgets .....	20
I.III Notification .....	27
I.III.I Son d'alerte .....	27
I.III.II E-mail .....	29
I.IV Déclencheurs .....	34
I.IV.I Déclencheur selon l'alias d'un port (SNMP) .....	34
Annexe II: Code Python .....	41
II.I Main.py .....	41
II.II switch.py .....	47
II.III Zabbix.py .....	54
II.IV gitManager.py .....	65
II.V logs.py .....	68
II.VI .env .....	70
Annexe III: Fonctionnement du programme .....	72
III.I Exécutable .....	72
III.I.I Fonctionnement .....	72
III.I.II Zabbix .....	73
III.I.III Fichier environnement .env .....	75
III.I.IV Logs .....	76
III.II Développement .....	77
III.II.I Require .....	77
III.II.II Compilation .....	77



# Annexe I.: Procédure Zabbix

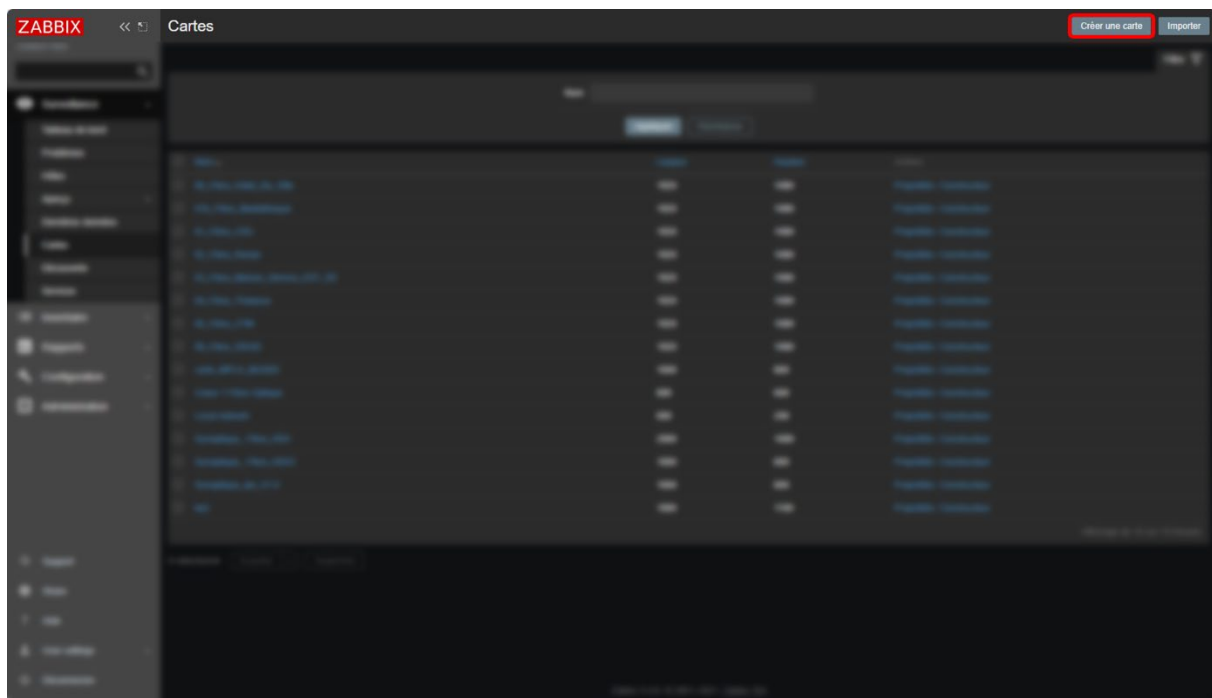
## I.I Cartes

Les cartes sont affichées dans l'onglet *cartes* dans la rubrique *Surveillance*.



### I.I.I Nouvelle carte

Pour créer une nouvelle carte cliquer sur [Créer une carte](#) en haut à droite de la fenêtre.



### I.I.I.I Création

Lors de la création d'une nouvelle carte des propriétés sont à définir, ceux précédés d'un astérisque « \* » sont obligatoires.

**Propriétaire** (requis) : Propriétaire de la nouvelle carte.

**Nom** (requis) : Nom de la nouvelle carte.

**Largeur/Hauteur** (requis) : Dimension de la nouvelle carte en pixels.

**Image de fond** : Ajouter une image de fond, par défaut un fond blanc est appliqué.

**Icône surlignée** : Si vous cochez cette case, les éléments de la carte seront mis en surbrillance.

Les éléments avec un déclencheur actif recevront un arrière-plan rond, de la même couleur que le déclencheur de gravité la plus élevée. De plus, une ligne verte épaisse sera affichée autour du cercle, si tous les problèmes sont reconnus.

Les éléments avec le statut "désactivé" ou "en maintenance" auront un fond carré, respectivement gris et orange.

**Marquer les éléments lors du changement d'état du déclencheur** : Un changement récent de l'état du déclencheur (problème récent ou résolution) sera mis en évidence avec des marqueurs (triangles rouges pointant vers l'intérieur) sur les trois côtés de l'icône de l'élément qui sont libres de l'étiquette. Les marqueurs sont affichés pendant 30 minutes.


**Afficher les problèmes** : Comment les problèmes liés à un élément seront affichés.

**Détailler problème unique** -> s'il n'y a qu'un seul problème, le nom du problème s'affiche. Sinon, le nombre total de problèmes est affiché.

**Nombre de problèmes** -> le nombre total de problèmes est affiché

**Nombre de problèmes et détailler le plus critique** -> le nom du plus critique problème et le nombre total de problèmes s'affiche.

**Sévérité minimale** : Les problèmes en dessous du niveau de sévérité minimum sélectionné ne seront pas affichés sur la carte.

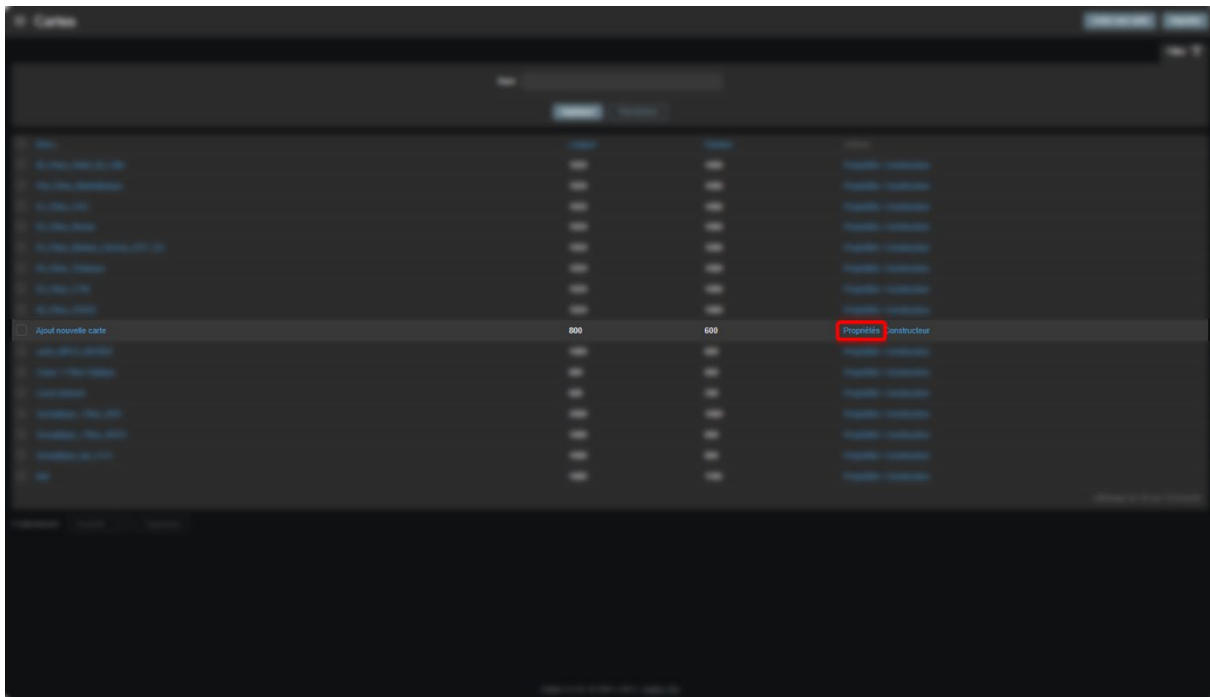
Valider les propriétés en cliquant sur 

Un bandeau validera la création de la nouvelle carte.



## I.I.II Modification

Pour modifier les propriétés d'une carte cliquer sur **Propriétés** aligné au nom de la carte.



### I.I.I.II Modification des propriétés

Voir [Création](#) puis validé avec **Actualiser**

**Actualiser**: Valider les changements apportés.

**Cloner**: Cloner conservera les attributs de mise en page généraux de la carte d'origine, mais aucun élément.

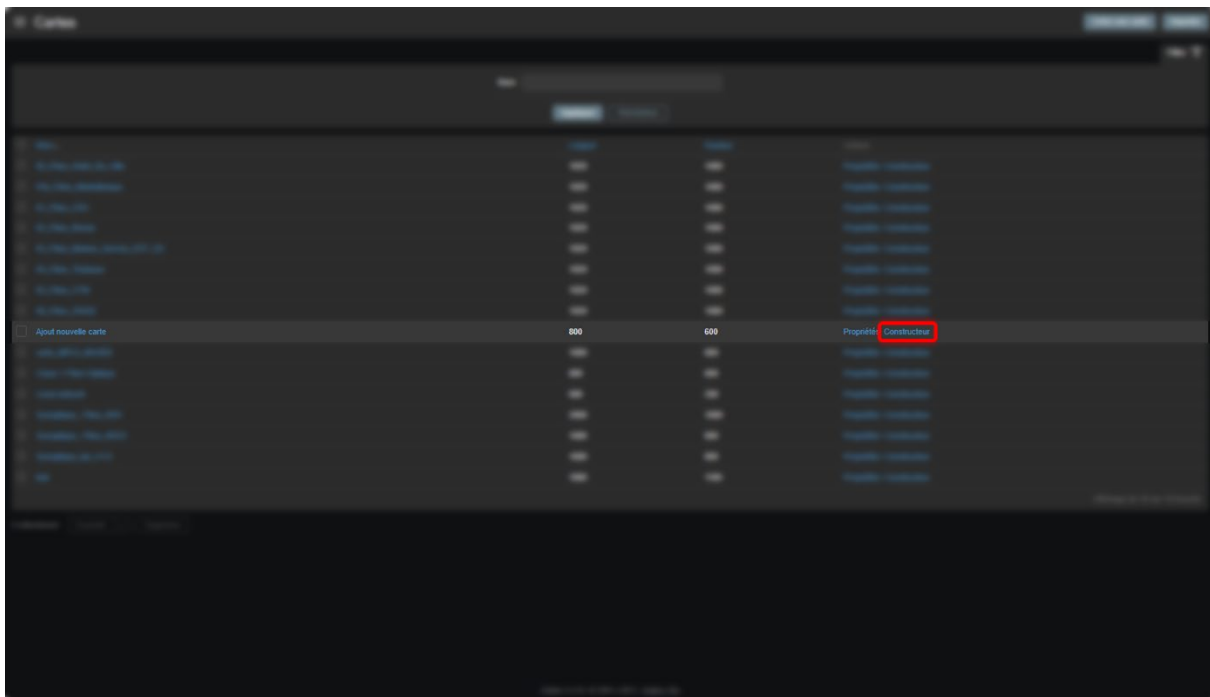
**Clone complet**: conserve à la fois les attributs de mise en page généraux et tous les éléments de la carte d'origine.

**Supprimer**: supprime la carte.

**Annuler**: annule les changements apportés

### I.I.II.II Modification des éléments de la carte

Pour modifier les éléments d'une carte cliquer sur **Constructeur** aligné au nom de la carte.



### I.I.II.III Éléments

**Élément de carte**: Ajouter ou Supprimer un élément de carte.

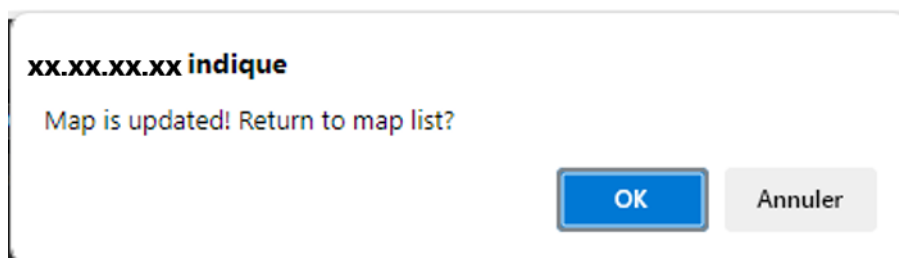
**Forme**: Ajouter ou Supprimer une forme.

**Lien**: Ajouter ou Supprimer un lien entre deux éléments de carte.

**Substituer les macros**: *inactif* -> affiche les macros par leur nom, *actif* -> affiche les macros par leurs valeurs.

**Grille**: Afficher ou Caché la grille, Actif -> active le magnétisme de la carte, Inactif -> désactive le magnétisme de la carte.

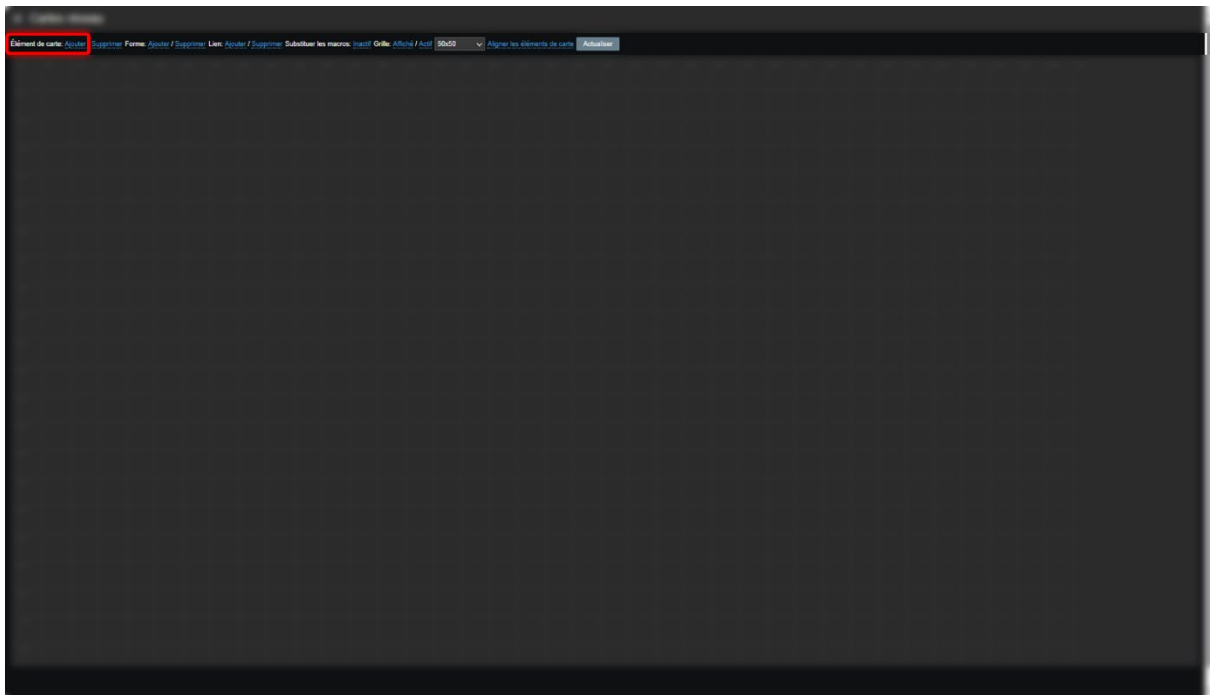
**Actualiser**: Sauvegarde les changements effectués, ouvre un pop-up, *OK* -> retourne à la liste des cartes, *Annuler* -> reste sur la page actuelle (la sauvegarde des modifications est effectuée)





### I.I.II.III.I Ajouter un élément

Pour ajouter une nouvelle forme cliquer sur **Ajouter** à côté de **Élément de carte:**



Cliquer sur la forme pour accéder à ses propriétés.

**Type:** *Hôte* -> icône représentant l'état de tous les déclencheurs de l'hôte sélectionné

*Carte* -> icône représentant l'état de tous les éléments d'une carte

Déclencheur -> icône représentant l'état d'un ou plusieurs déclencheurs

*Groupe d'hôtes* -> icône représentant l'état de tous les déclencheurs de tous les hôtes appartenant au groupe sélectionné

*Image* -> une icône, non liée à une ressource

#### I.I.II.III.I.I Image (Défaut)

**Étiquette:** étiquette d'objet, peut contenir du texte brut ou des [macros](#).

**Icônes:** Liste prédéfini d'icônes.

**Coordonnées**: Coordonnées de l'objet en pixels.

**URL**: Lien URL, peut rediriger vers un lien URL ou un lien zabbix via une requête ->  
zabbix.php?action=map.view&sysmapid=XX  
Le nom peut contenir une macro

### I.I.I.I.I.I.II Hôte

The screenshot shows the 'Host' configuration form in Zabbix. The form is titled 'Élément de carte' (Map Element). It includes a 'Type' dropdown set to 'Hôte'. There is a text field for 'Étiquette' (Label) and a dropdown for 'Positionnement de l'étiquette' (Label Positioning) set to 'Défaut'. Below this is a search field for 'Hôte' with a 'Sélectionner' button. The 'Tags' section has two tabs: 'Et/Ou' (selected) and 'Ou'. It contains a table with columns 'tag', 'Contient' (set to 'valeur'), and 'valeur', with an 'Ajouter' button and a 'Supprimer' link. There is a checkbox for 'Sélection automatique d'icône'. The 'Icônes' (Icons) section has four dropdowns: 'Défaut' (Server\_96), 'Problème' (Défaut), 'Maintenance' (Défaut), and 'Désactivé' (Défaut). The 'Coordonnées' (Coordinates) section has input fields for 'X' (889) and 'Y' (327). The 'URLs' section has a table with columns 'Nom', 'URL', and 'Action', with an 'Ajouter' button and a 'Supprimer' link. At the bottom are 'Appliquer', 'Supprimer', and 'Fermer' buttons.

Voir [défaut](#).

**Étiquette**: Peut contenir des macros tel que {HOST.NAME} -> affiche le nom, {HOST.CONN} ou {HOST.IP} -> affiche l'adresse IP.

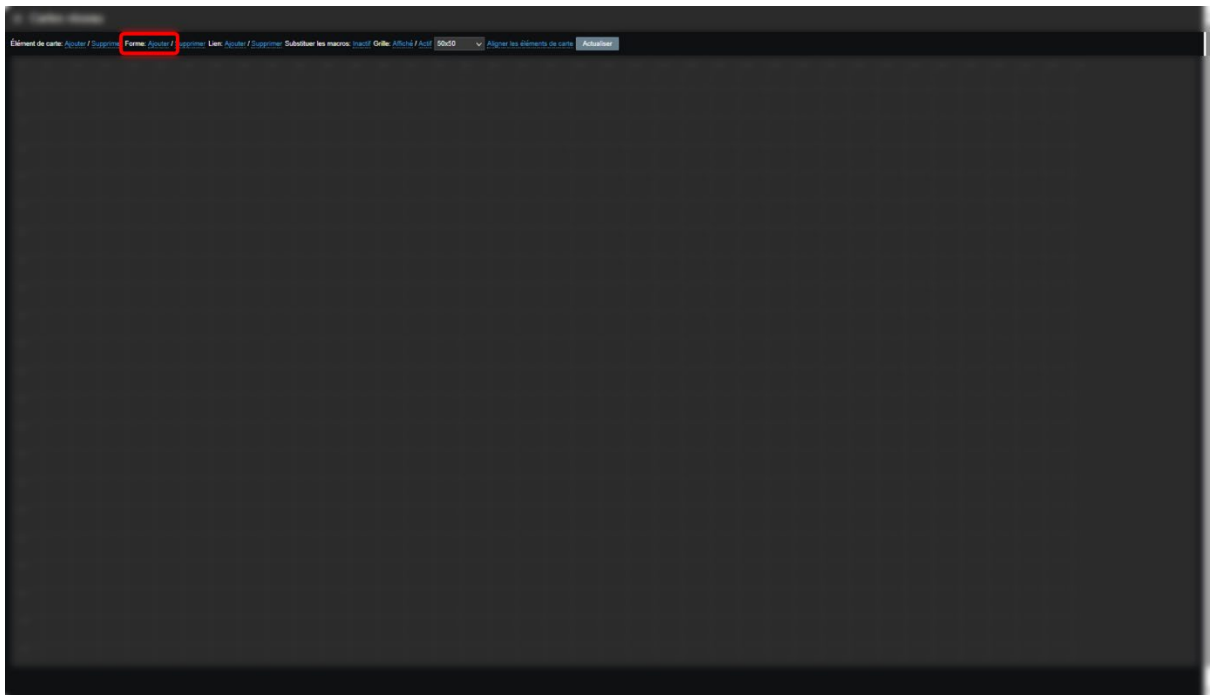
**Hôte**: Défini quel hôte est associé.

**Tags**: Spécifiez des tags pour limiter le nombre de problèmes affichés dans le widget. Il est possible d'inclure et d'exclure des tags et des valeurs de tag spécifiques. Plusieurs conditions peuvent être définies. La correspondance des noms de tag est toujours sensible à la casse.



### I.I.II.III.II Ajouter une forme

Pour ajouter une nouvelle forme cliquer sur **Ajouter** à côté de **Forme**:



Cliquer sur la forme pour accéder à ses propriétés.

A screenshot of the 'Forme de carte' (Map Shape) properties dialog box. The dialog has a title bar 'Forme de carte'. It contains several sections: 'Forme' with buttons for 'Rectangle', 'Ellipse', and 'Ligne'; 'Texte' with a text input field, 'Police' (Helvetica), 'Taille de police' (11), 'Couleur' (000000), 'Alignement horizontal' (Centre), and 'Alignement vertical' (Milieu); 'Fond' with a 'Couleur' button and a color swatch; 'Bordure' with 'Type' (solid line), 'Largeur' (2), and 'Couleur' (000000); 'Coordonnées' with 'X' (10) and 'Y' (10) input fields; and 'Taille' with 'Largeur' (50) and 'Hauteur' (50) input fields. At the bottom are three buttons: 'Appliquer', 'Supprimer', and 'Fermer'.

**Forme**: Forme de l'objet

**Texte**: étiquette d'objet, peut contenir du texte brut ou des [macros](#) ({MAP.NAME} pour afficher le nom de la carte)

**Fond**: Couleur de remplissage de l'objet.

**Bordure**: Bordure de l'objet.

**Coordonnées**: Coordonnées de l'objet en pixels.

**Taille**: Dimension de l'objet en pixels.

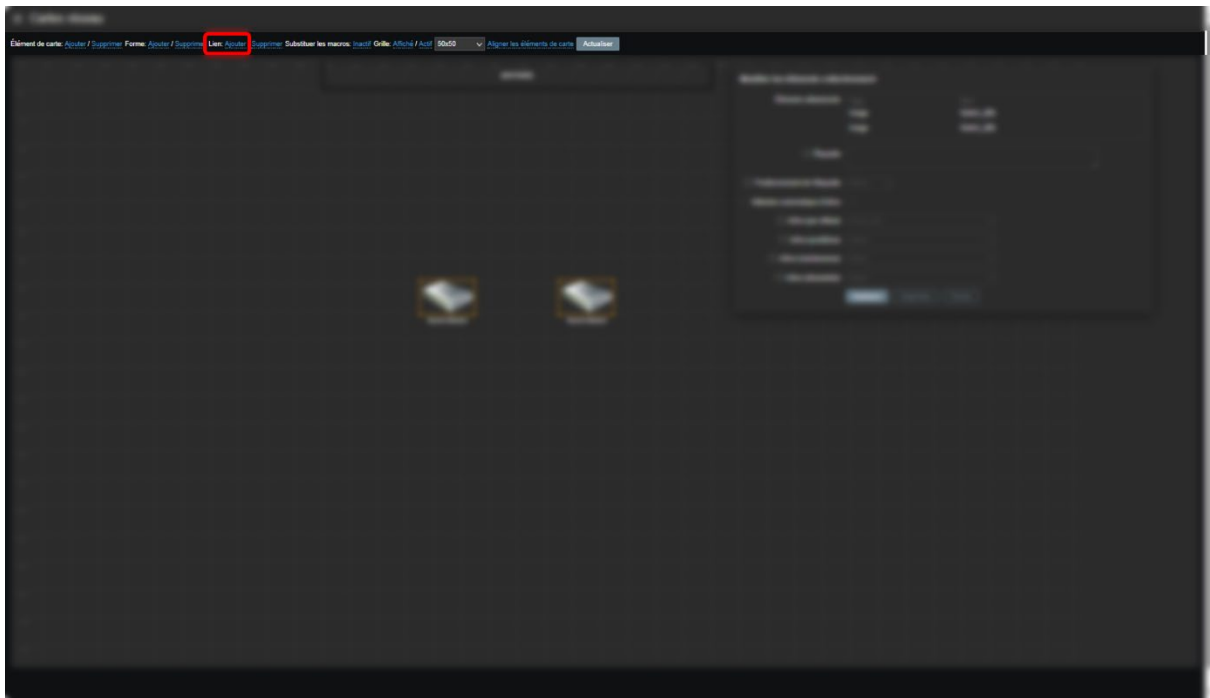
**Appliquer**: Valider les propriétés.

**Supprimer**: Supprimer l'objet.

**Fermer**: Fermer sans valider les changements.

### I.I.II.III.III Ajouter un lien

Pour lier deux éléments sélectionner les deux éléments à lier, puis cliquer sur **Ajouter** à côté de **Lien**.



Les deux éléments sont alors liés, il n'y a pas de limite de lien entre deux éléments.



En plus des [propriétés standards des éléments](#), une propriété liens est ajoutée.

Si deux éléments liés sont sélectionnés, alors la propriété liens affiche uniquement leurs liaisons.

De	À	Indicateurs de lien	Action
Switch_(96)	Switch_(96)		Édition

**De**: Nom du 1<sup>er</sup> élément sélectionné


**À**: Nom du 2<sup>ème</sup> élément sélectionné

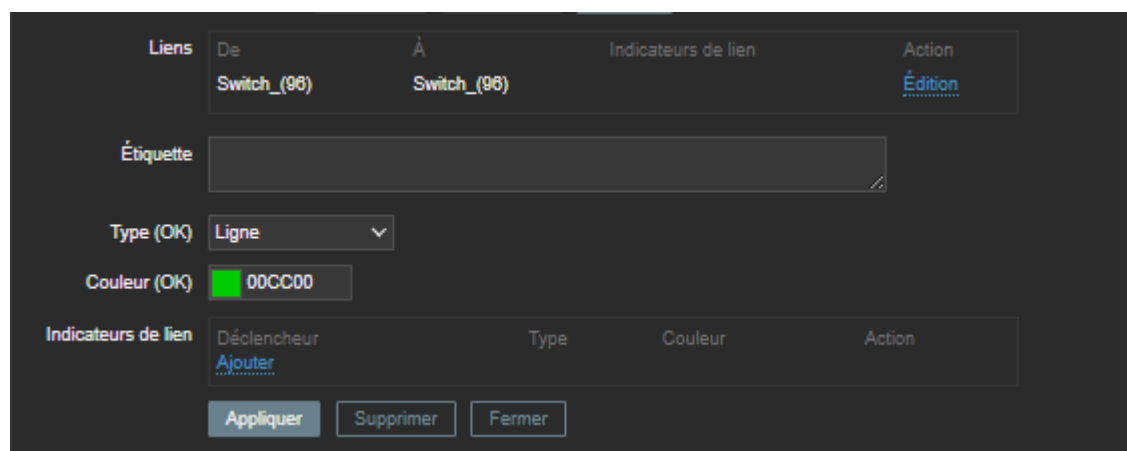
**Indicateurs de lien**: Déclencheur surveillés

*Action*: Édition des déclencheurs

### I.I.II.III.IV Propriété d'un lien

Sélectionner les deux éléments du lien auquel le déclencheur doit être ajouté.

Dans la propriété Liens cliquer sur 



**Étiquette** : Étiquette du lien peut contenir du texte brut ou des macros

**Type(OK)** : Style du lien lorsqu'il n'y a pas de problèmes à afficher

**Couleur(OK)** : Couleur du lien lorsqu'il n'y a pas de problèmes à afficher

**Indicateur de lien** : Déclencheur associé au lien

### I.I.II.III.IV.I Ajouter un déclencheur

Pour ajouter un déclencheur cliquer sur **Ajouter**

Puis sélectionner un hôte

**Déclencheurs**

Hôte taper ici pour rechercher **Sélectionner**

<input type="checkbox"/> Nom	Sévérité	État
Spécifier des conditions de filtre pour voir les valeurs.		

**Sélectionner** **Annuler**

Ensuite sélectionner un ou plusieurs déclencheurs à ajouter

<input type="checkbox"/> Nom	Sévérité	État
<input type="checkbox"/> Device: Temperature is above critical threshold: >85	Haut	Inconnu
<input type="checkbox"/> Device: Temperature is above warning threshold: >55 Dépend de Device: Temperature is above critical threshold: >85	Avertissement	Inconnu
<input type="checkbox"/> Device: Temperature is too low: <5	Moyen	Inconnu
<input type="checkbox"/> Device has been replaced (new serial number received)	Information	Activé
<input type="checkbox"/> Firmware has changed	Information	Activé

Valider avec **Sélectionner**

La couleur et le type de chaque déclencheur peut être défini individuellement.

	Type	Couleur	Action
Interface ifc1 (Slot: 1 Port: 1): Ethernet has changed to lower speed than it	Ligne	DD0000	Supprimer
Interface ifc1 (Slot: 1 Port: 1): High bandwidth usage (>90% )	Ligne	DD0000	Supprimer
Interface ifc1 (Slot: 1 Port: 1): High error rate (>2 for 5m)	Ligne	DD0000	Supprimer
Interface ifc1 (Slot: 1 Port: 1): In half-duplex mode	Ligne	DD0000	Supprimer
Interface ifc1 (Slot: 1 Port: 1): Link down	Ligne	DD0000	Supprimer

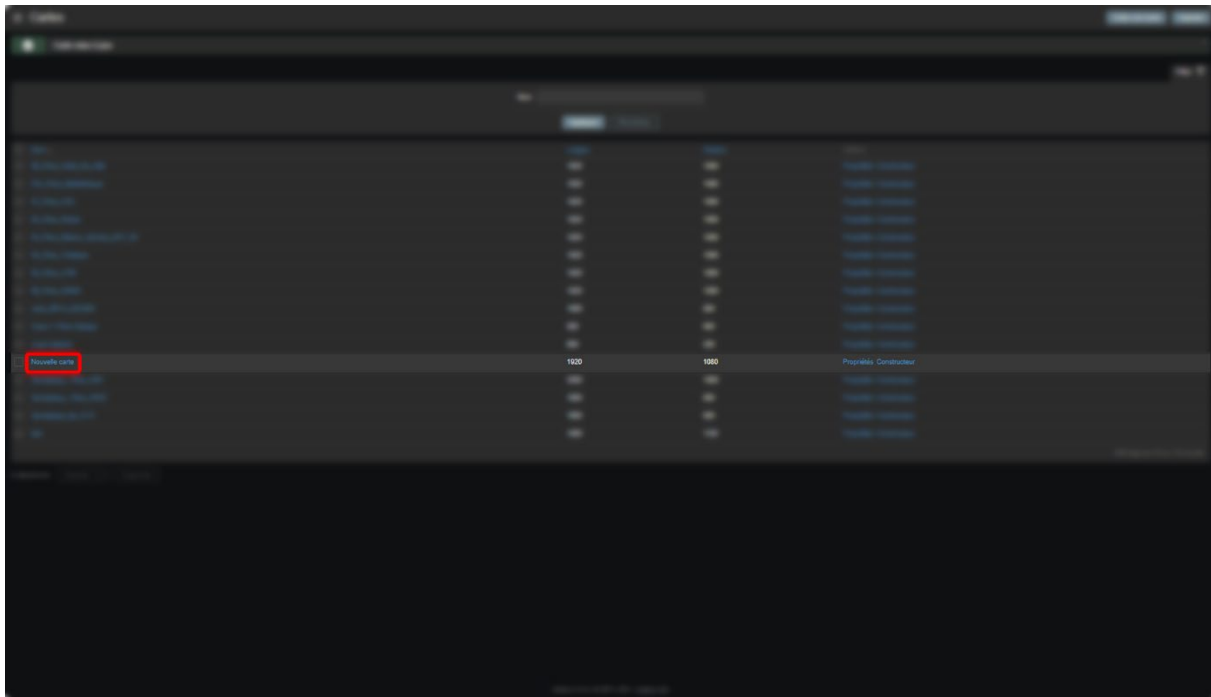
Cliquer de nouveau sur **Ajouter** et choisir le second hôte pour y ajouter ses déclencheur.



### I.I.III Affichage

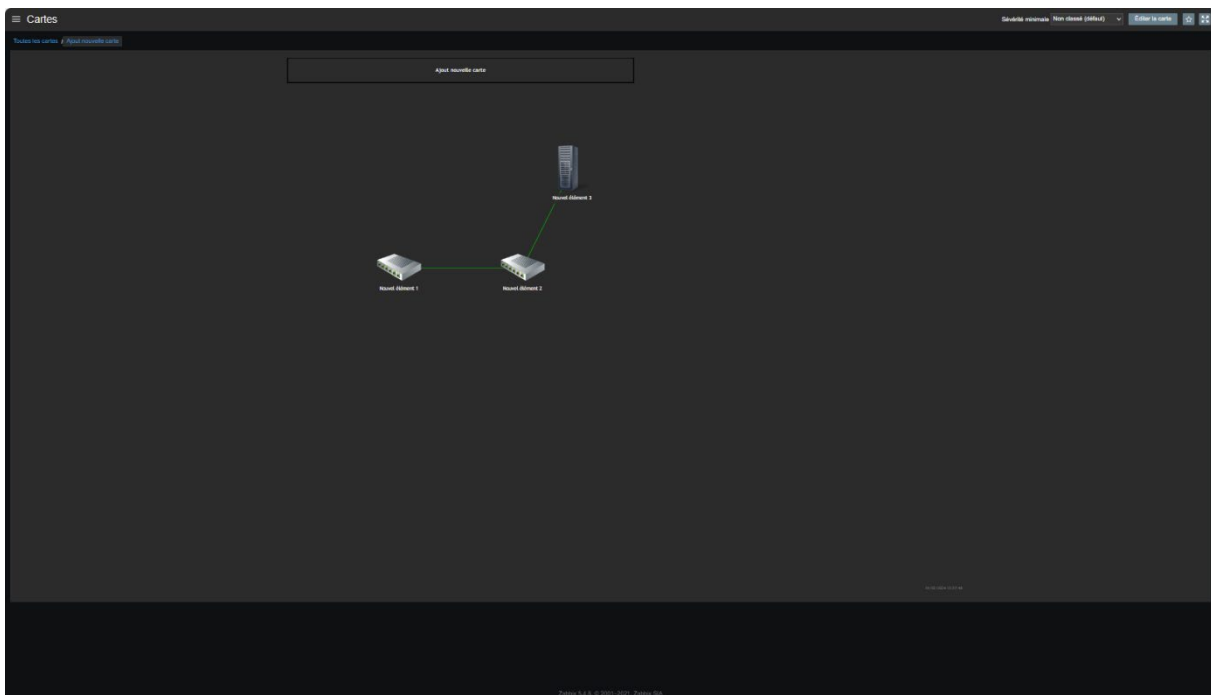
#### I.I.III.I Afficher la carte

Pour voir une carte cliquer sur le nom de la carte dans la liste.



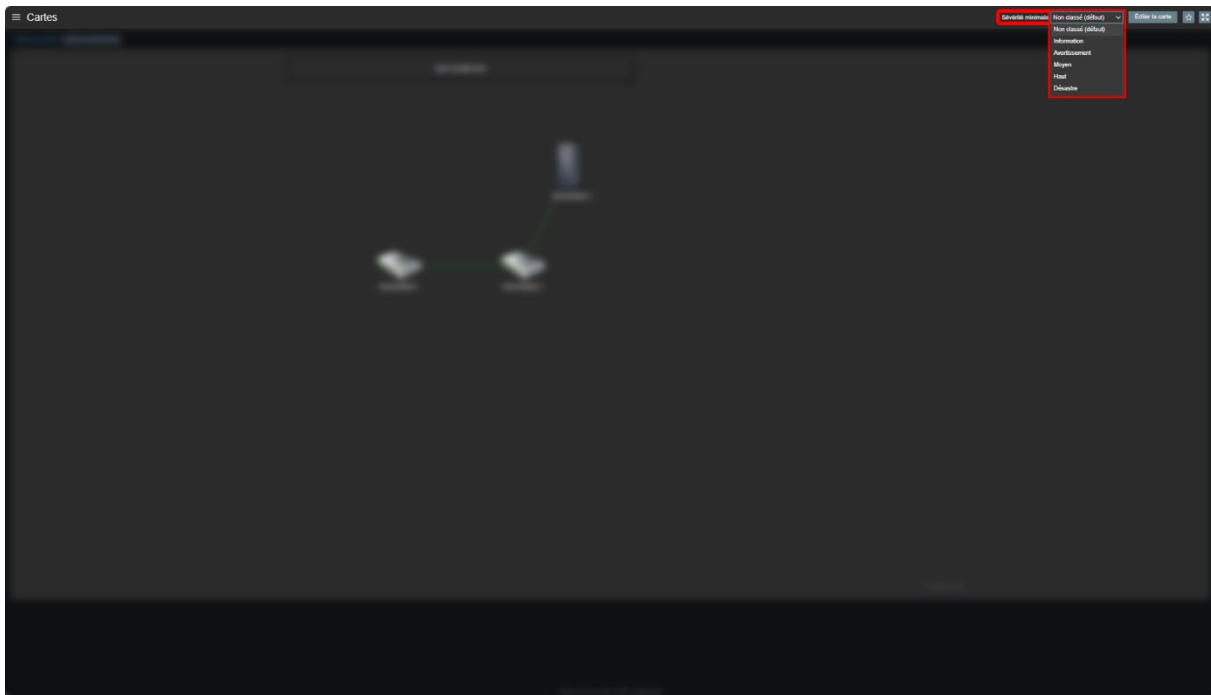
La carte affiche les éléments et leurs problèmes.

Si un élément est de type [Carte](#) et qu'un ou plusieurs éléments de cette sous-carte comportent des problèmes, alors l'élément de type carte affichera les problèmes liés.



### I.I.III.II Afficher une sévérité différente

Choisir la sévérité affichée dans la liste déroulante **Sévérité minimale**



### I.I.III.III Action

**Éditer la carte**: Cliquer sur **Éditer la carte**, voir [Éléments](#)

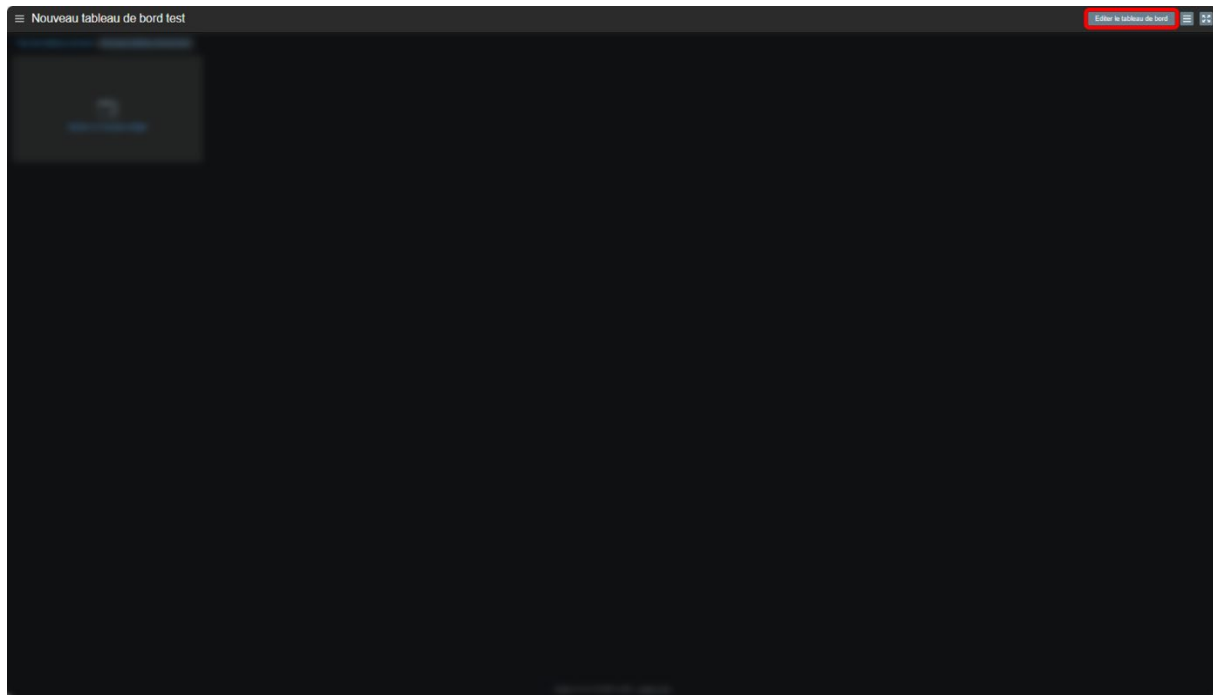
**Favori**: Cliquer sur  pour enregistrer la carte dans les favoris

**Plein écran**: Cliquer sur  pour passer la carte en plein écran.

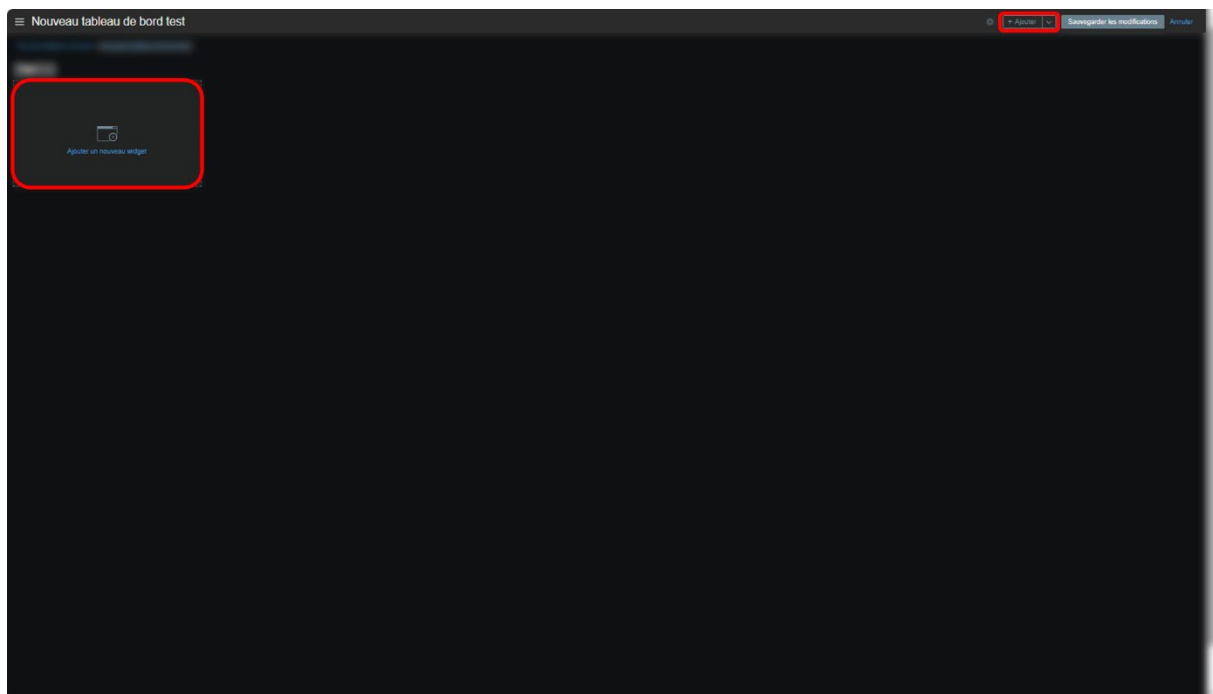
## I.I.IV Tableau de bord

### I.I.IV.I Ajouter une carte à un tableau de bord

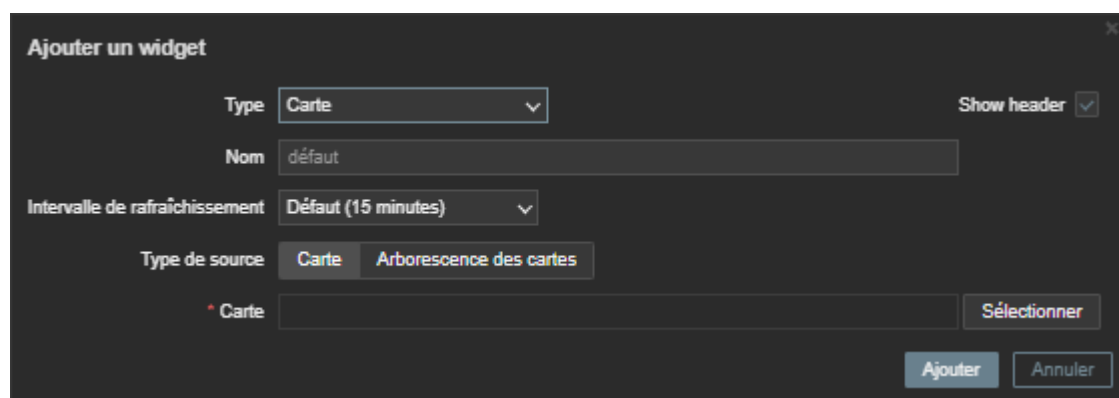
Cliquer sur **Editer le tableau de bord** pour éditer le tableau de bord.



Puis Ajouter un nouveau widget.



Choisir le type : carte.



Ajouter un widget

Type: Carte

Nom: défaut

Intervalle de rafraîchissement: Défaut (15 minutes)

Type de source: Carte, Arborescence des cartes

\* Carte

Sélectionner

Ajouter Annuler

**Nom** : Nommé le widget (Optionnel).

**Intervalle de rafraîchissement** : Rafraîchit la carte selon un intervalle prédéterminer {10s | 30s| 1 min | 2 min | 10 min | 15 min}

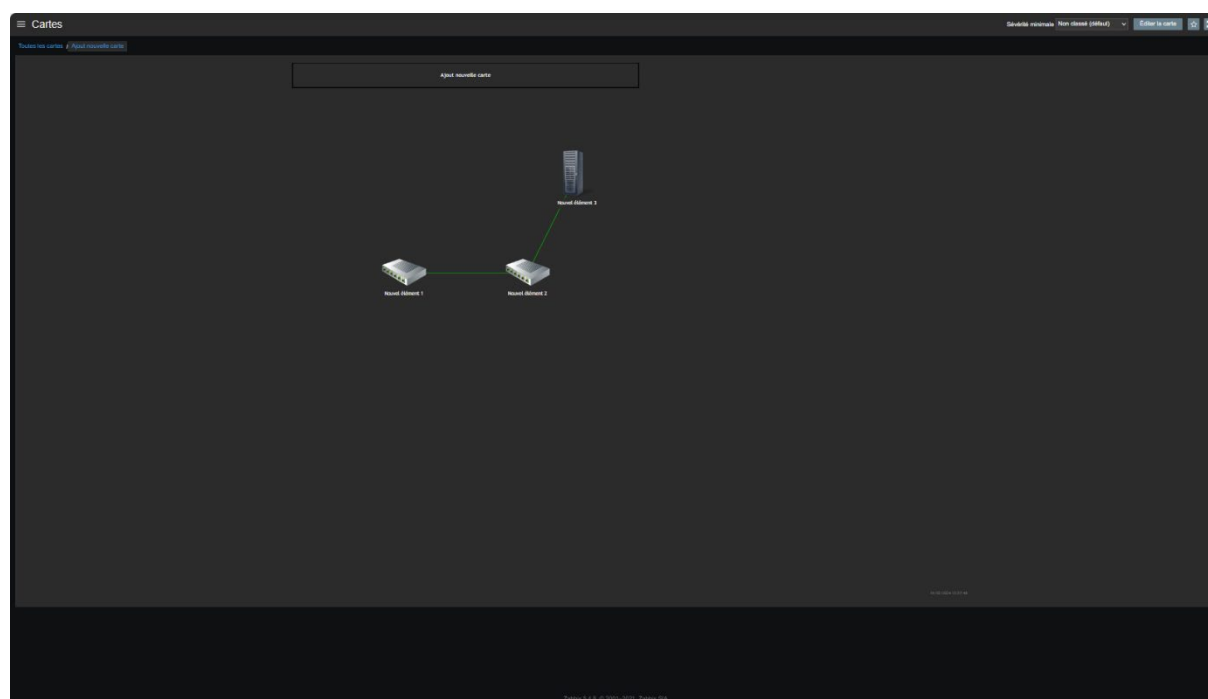
**Type de source** : Carte -> sélectionne et affiche une carte

Arborescence des cartes -> Affiche des cartes selon une arborescence prédéfini dans un widget « Arborescence des cartes ».

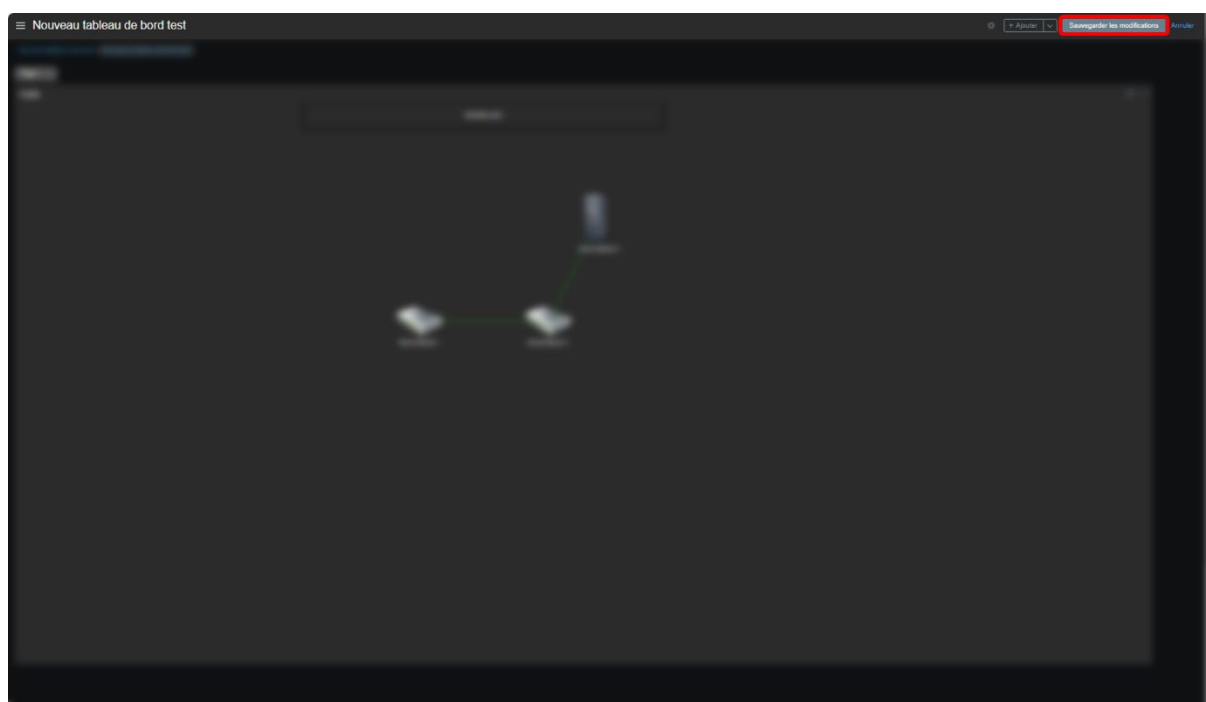
**Carte** (requis) : Sélection de la carte à afficher dans le widget.

Pour valider, cliquer sur .

Le widget de la carte peut être déplacé et redimensionner.

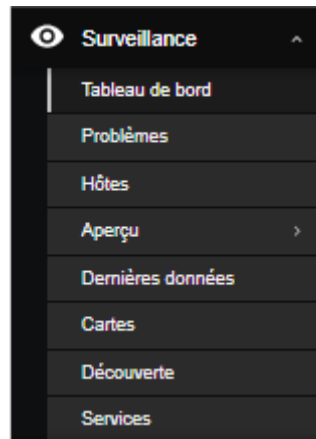


Enregistrer le tableau de bord en cliquant sur **Sauvegarder les modifications**



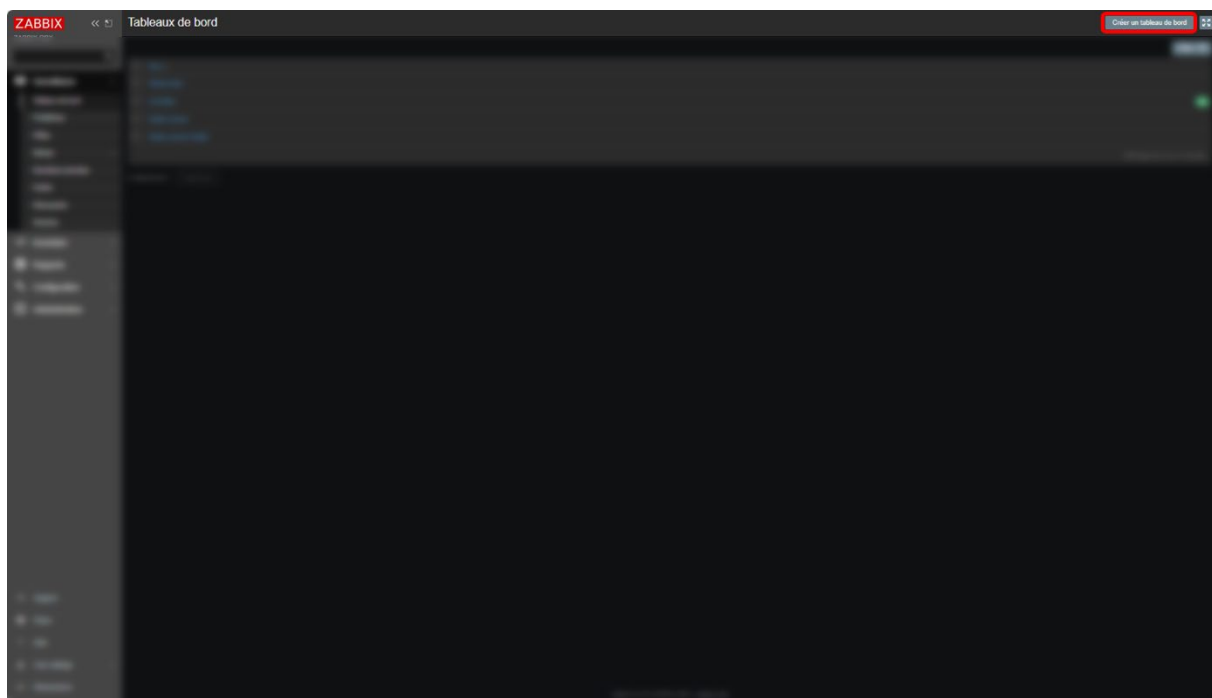
## I.II Tableau de bord

Les tableaux de bords sont affichés dans l'onglet *tableau de bord* dans la rubrique *Surveillance*.



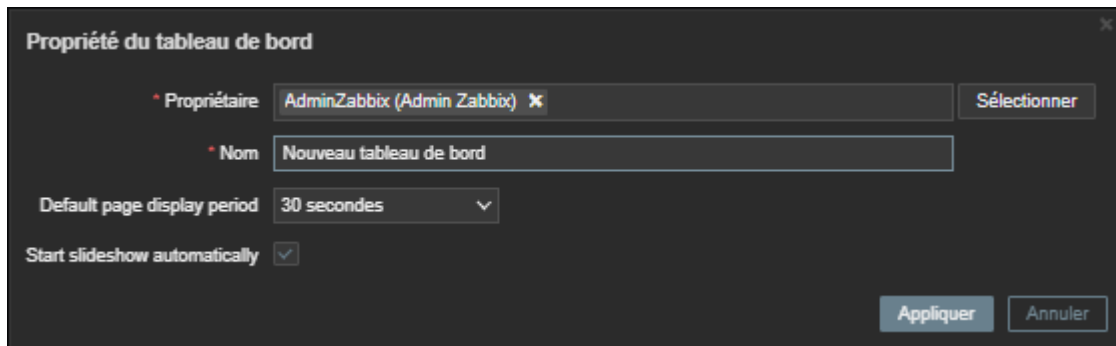
### I.II.I Nouveau tableau de bord

Pour créer un nouveau tableau de bord cliquer sur [Créer un tableau de bord](#)



#### I.II.I.I Création

Lors de la création d'un nouveau tableau de bord des propriétés sont à définir, ceux précédés d'un astérisque « \* » sont obligatoires.




**Propriétaire** (requis) : Propriétaire de la nouvelle carte.

**Nom** (requis) : Nom de la nouvelle carte.

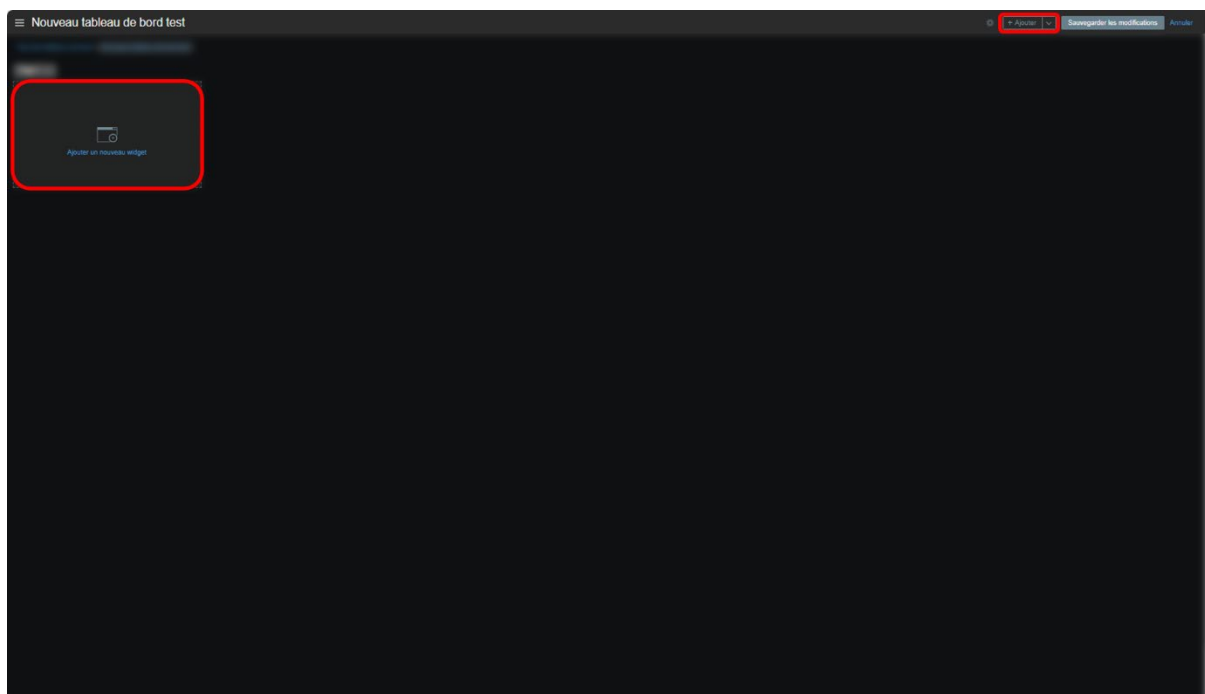
**Default page display period**: Intervalle du changement de page lorsque le mode diaporama est activé selon un intervalle prédéterminer {10s | 30s | 1 min | 2 min | 10 min | 30 min | 1h}

**Start slideshow automatically**: Démarrer le mode diaporama automatiquement

Valider les propriétés en cliquant sur 

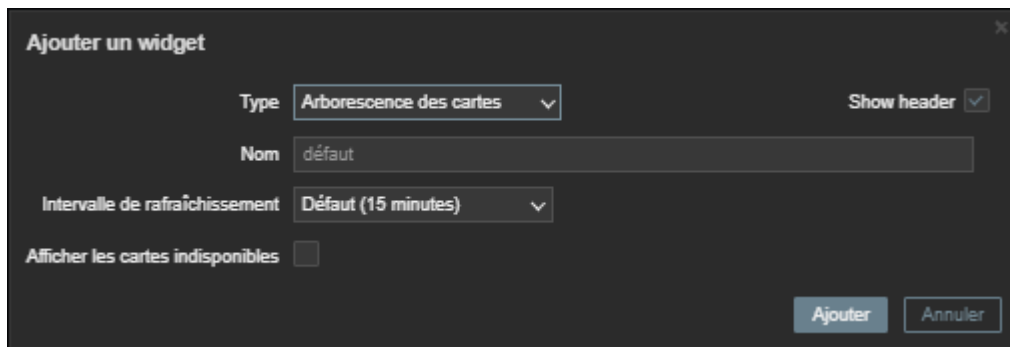
#### I.II.II Ajouter des widgets

Cliquer sur ajouter un nouveau widget.



Les widgets peuvent être redimensionnés et déplacés.

### I.II.II.I Widget : arborescence des cartes



Ajouter un widget

Type: Arborescence des cartes

Show header: ☒

Nom: défaut

Intervalle de rafraîchissement: Défaut (15 minutes)

Afficher les cartes indisponibles: ☐

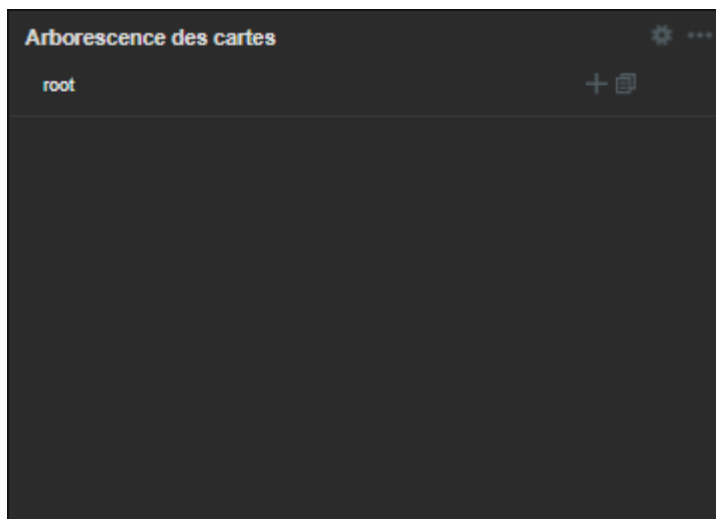
Ajouter Annuler

**Nom** : Nommé le widget (Optionnel).

**Intervalle de rafraîchissement** : Rafraîchit la carte selon un intervalle prédéterminer {10s | 30s | 1 min | 2 min | 10 min | 15 min}

**Afficher les cartes indisponibles**

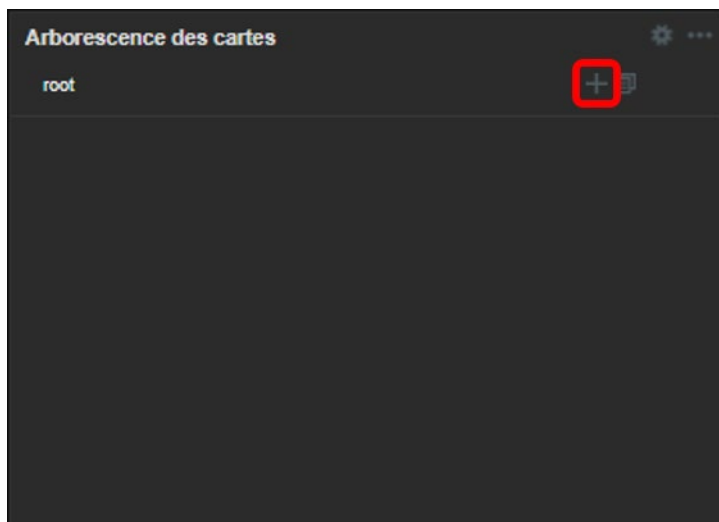
Pour valider, cliquer sur .



Ajouter une arborescence

Cliquer sur 





×

Éditer un élément d'arborescence

\*

Nom

Carte liée

Sélectionner

☐

Ajouter des sous-cartes

Ajouter

Annuler

**Nom** (requis) : Nom de l'arborescence.

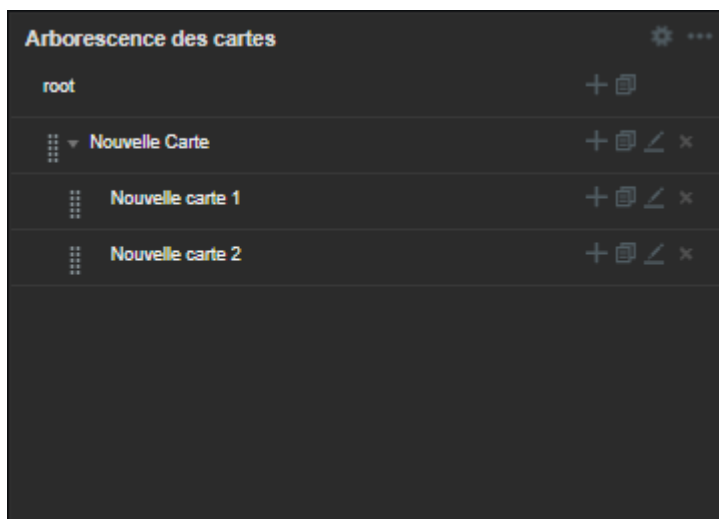
**Carte liée** : Ajouter une carte à l'arborescence.

**Ajouter des sous-cartes** : ajouter les sous-cartes liées à la carte sélectionnée.

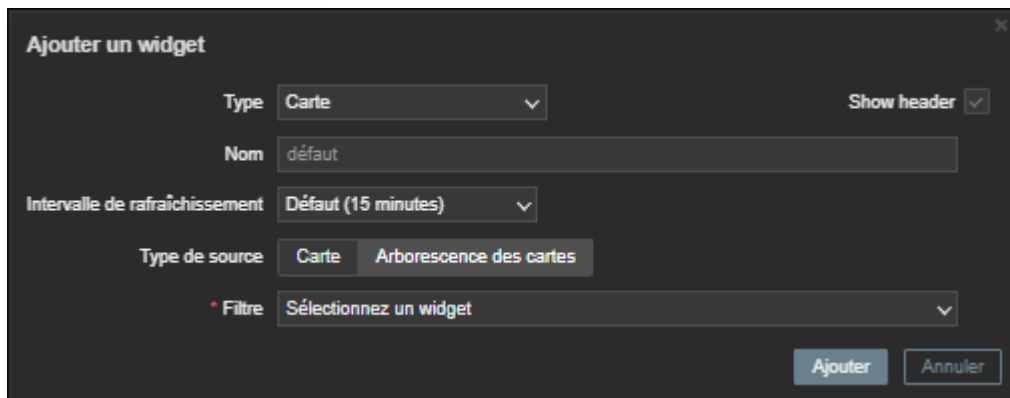
Pour valider, cliquer sur 

Ajouter

.



### I.II.II.II Widget : carte (arborescence)



The dialog box 'Ajouter un widget' (Add widget) is shown. It has a title bar with a close button. The fields are: 'Type' set to 'Carte' (Card) with a dropdown arrow; 'Show header' checked; 'Nom' (Name) set to 'défaut' (default) in a text input; 'Intervalle de rafraîchissement' (Refresh interval) set to 'Défaut (15 minutes)' (Default (15 minutes)) with a dropdown arrow; 'Type de source' (Source type) with two buttons: 'Carte' (Card) and 'Arborescence des cartes' (Card tree), where 'Arborescence des cartes' is selected; 'Filtre' (Filter) with a dropdown menu showing 'Sélectionnez un widget' (Select a widget); and two buttons at the bottom: 'Ajouter' (Add) and 'Annuler' (Cancel).

**Nom** : Nommé le widget (Optionnel).

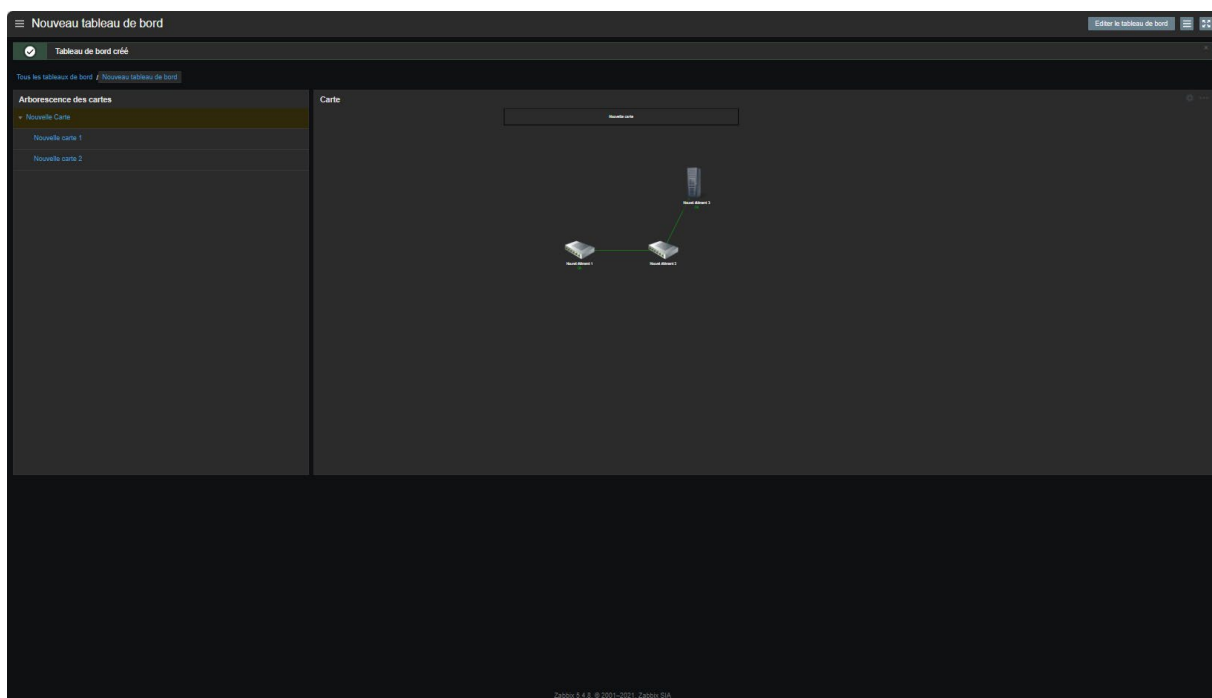
**Intervalle de rafraîchissement** : Rafraîchit la carte selon un intervalle prédéterminer {10s | 30s| 1 min | 2 min | 10 min | 15 min}

**Type de source** : *Carte* -> sélectionne et affiche une carte

*Arborescence des cartes* -> Affiche des cartes selon une arborescence prédéfini dans un widget  
« Arborescence des cartes ».

**Filtre** (requis) : Sélection du widget source.

Pour valider, cliquer sur .



### I.II.II.III Widget : carte (carte unique)

**Ajouter un widget**

Type:  Show header: ☒

Nom:

Intervalle de rafraîchissement:

Type de source:

\* Carte:

**Nom** : Nommé le widget (Optionnel).

**Intervalle de rafraîchissement** : Rafraîchit la carte selon un intervalle prédéterminer {10s | 30s| 1 min | 2 min | 10 min | 15 min}

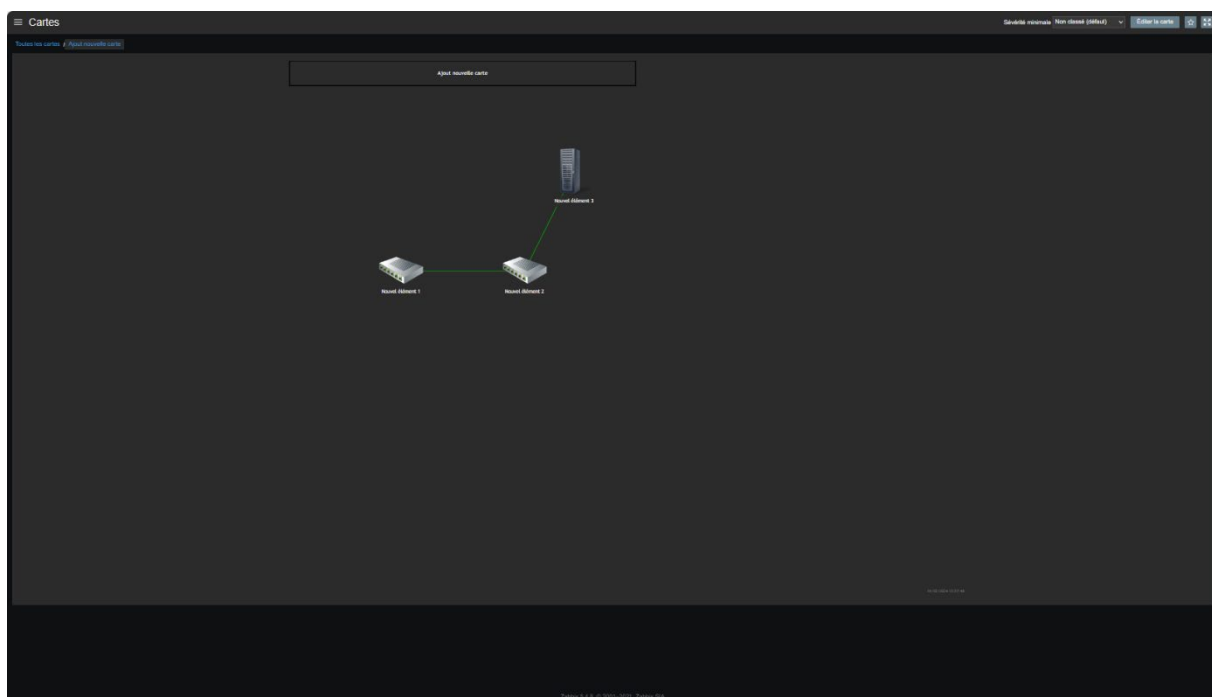
**Type de source** : *Carte* -> sélectionne et affiche une carte

*Arborescence des cartes* -> Affiche des cartes selon une arborescence prédéfini dans un widget  
« Arborescence des cartes ».

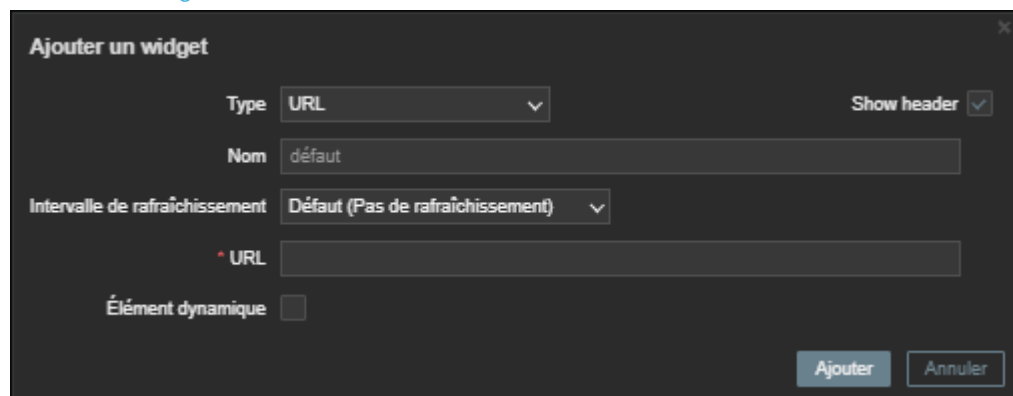
**Carte** (requis) : Sélection de la carte à afficher dans le widget.

Pour valider, cliquer sur .

Le widget de la carte peut être déplacé et redimensionner.



#### I.II.II.IV Widget : URL



**Ajouter un widget**

Type: URL

Nom: défaut

Intervalle de rafraîchissement: Défaut (Pas de rafraîchissement)

\* URL:

Élément dynamique: ☐

Ajouter Annuler

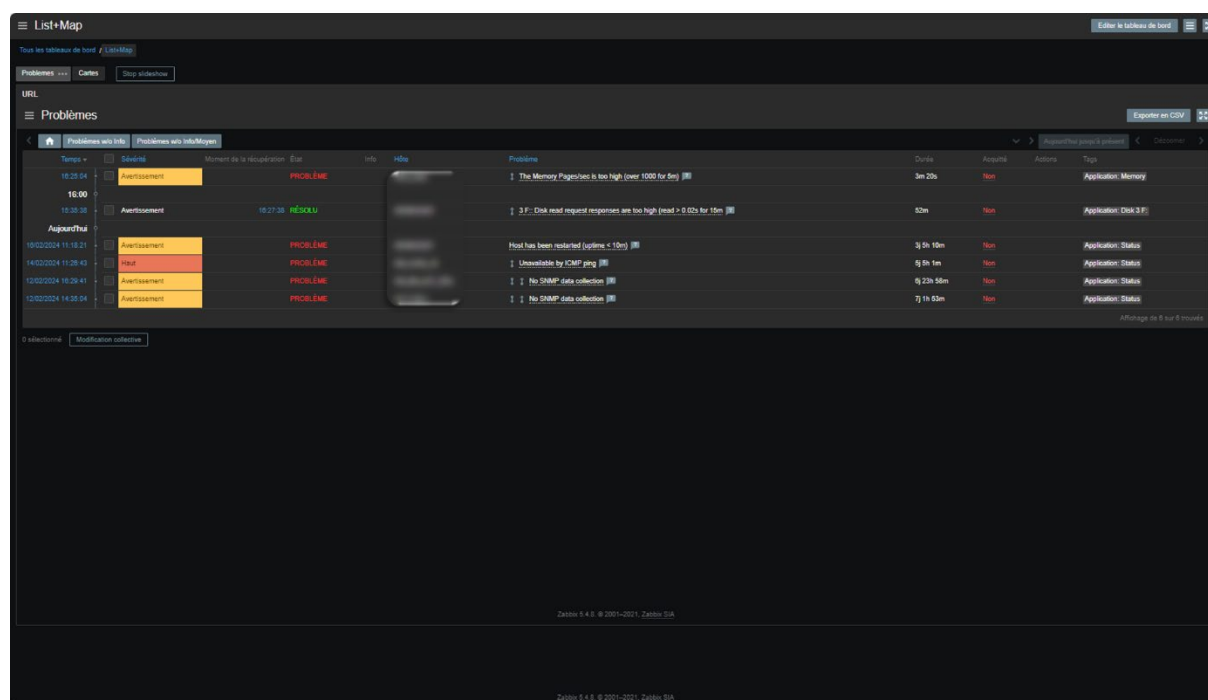
**Nom**: Nommé le widget (Optionnel).

**Intervalle de rafraîchissement**: Rafraîchit la carte selon un intervalle prédéterminer {pas de rafraîchissement | 10s | 30s | 1 min | 2 min | 10 min | 15 min}

**URL**: URL du lien à afficher, peut rediriger vers un lien URL ou un lien zabbix via une requête -> zabbix.php?action=map.view&sysmapid=XX

**Élément dynamique**: Configurez pour afficher un contenu d'URL différent en fonction de l'hôte sélectionné.

Cela peut fonctionner si les macros {HOST.\*} sont utilisées dans l'URL.



List-Map

Tous les tableaux de bord / List-Map

Problèmes ... Cartes Stop watchdog

URL

Problèmes

Exporter en CSV

Temps	Sévérité	Moment de la récupération	État	Info	Host	Problème	Durée	Reconnu	Actions	Tags
16:25:54	Avertissement		PROBLÈME			1 The Memory Pages/sec is too high (over 1000 for 5s) [M]	3m 20s	Non		Application: Memory
16:26:38	Avertissement	16:27:38	RÉSOLU			1 3 F.: Disk read request responses are too high (read > 0.02s for 15m) [M]	52m	Non		Application: Disk 3 F.
19/02/2024 11:18:21	Avertissement		PROBLÈME			Host has been restricted (uptime < 10m) [M]	3h 15m	Non		Application: Status
14/02/2024 11:28:42	Erreur		PROBLÈME			1 Unavailable by ICMP ping [M]	1h 5m	Non		Application: Status
12/02/2024 16:29:41	Avertissement		PROBLÈME			1 No SNMP data collection [M]	1h 23m 56m	Non		Application: Status
12/02/2024 14:35:54	Avertissement		PROBLÈME			1 No SNMP data collection [M]	7h 1h 53m	Non		Application: Status

0 sélectionné Modification collective

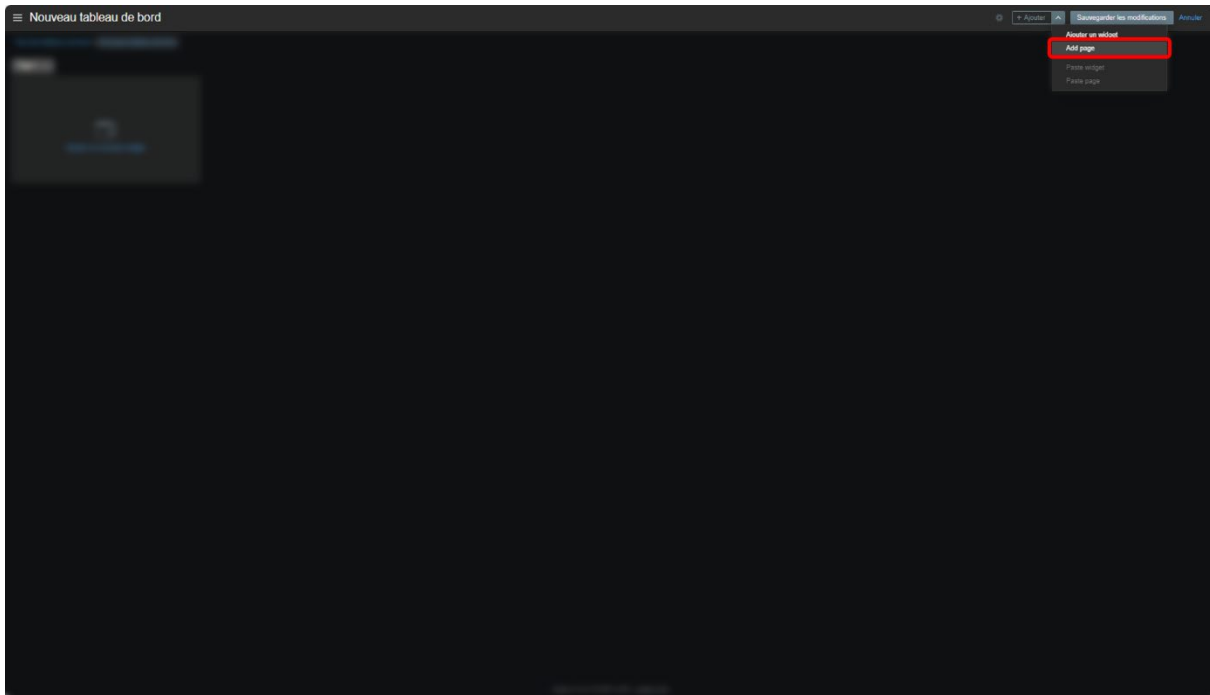
Zabbix 5.4.8. © 2001-2021, Zabbix SIA

### I.II.II.V Diaporama

Pour créer un diaporama entre plusieurs pages de widgets les propriétés *Default page display period* et *Start slideshow automatically* doivent être défini, voir [création](#)

### I.II.II.VI Nouvelle page

Pour créer une nouvelle page cliquer sur **Add page** dans la liste déroulante **+ Ajouter**

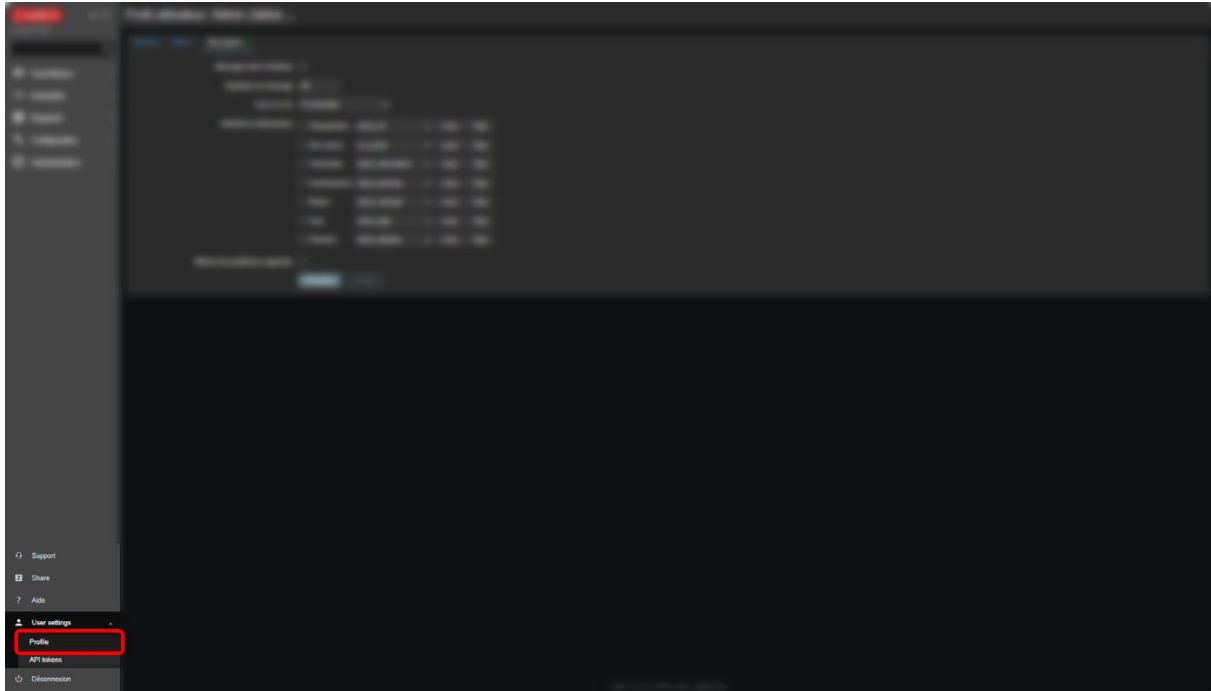


Puis suivre les étapes d'[Ajout de widgets](#)

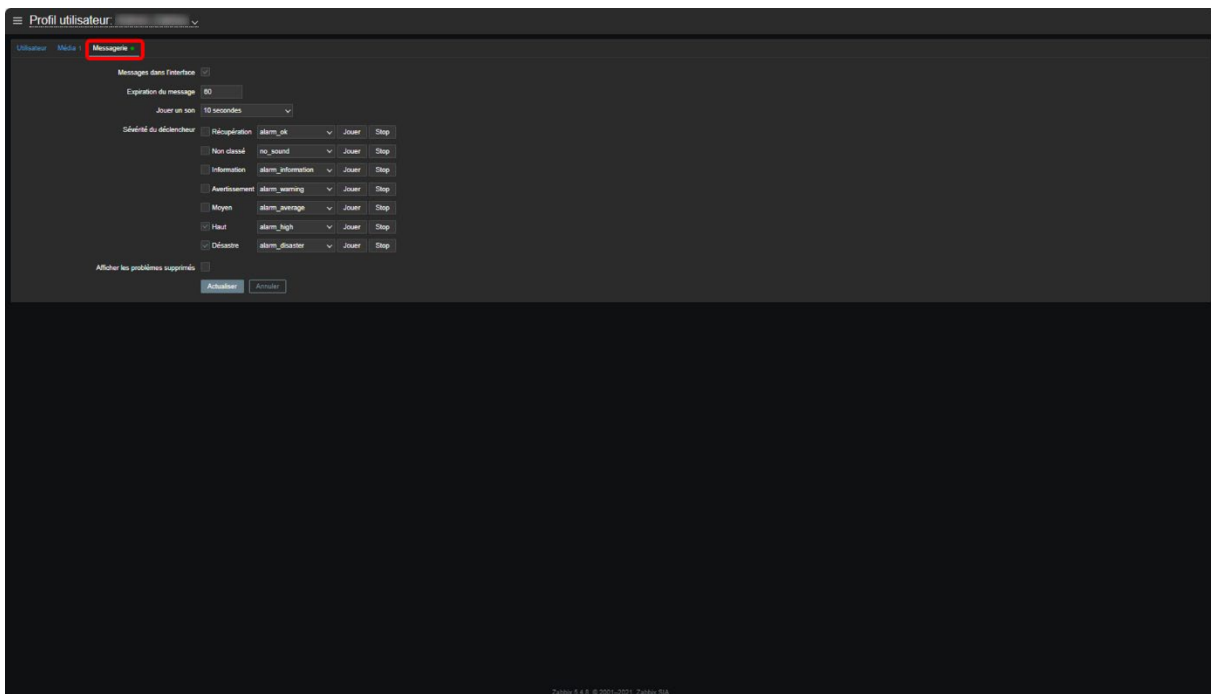
## I.III Notification

### I.III.I Son d'alerte

Les options d'utilisateurs sont dans l'onglet *profile* dans la rubrique *User settings*



Accéder à l'onglet **Messagerie**



**Messages dans l'interface**: Cochez la case pour activer les notifications globales.

**Expiration du message**: Durée d'affichage du message par défaut 60 sec.

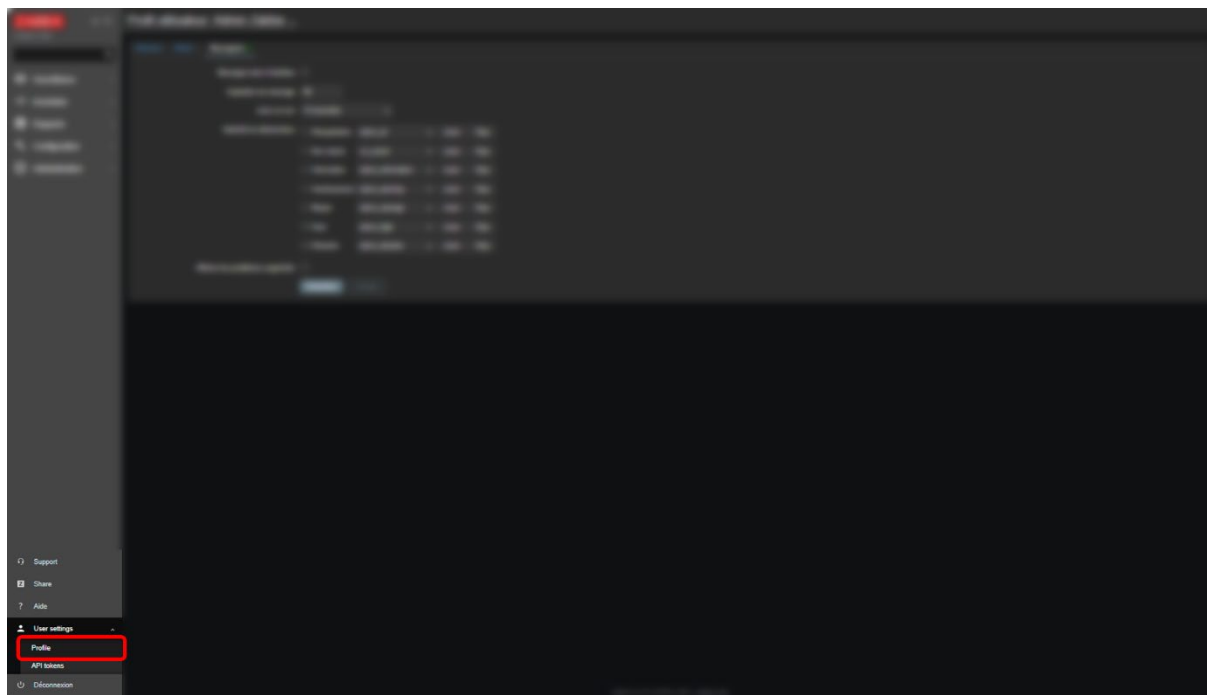
**Jouer un son**: Délai du son à jouer {Une fois | 10 s | Expiration du message}.

***Sévérité du déclencheur***: Sélectionner les sévérités pour lesquels jouer un son.

***Afficher les problèmes supprimés***: Cochez la case pour afficher les notifications relatives aux problèmes qui seraient autrement supprimés (non affichés) en raison de la maintenance de l'hôte.

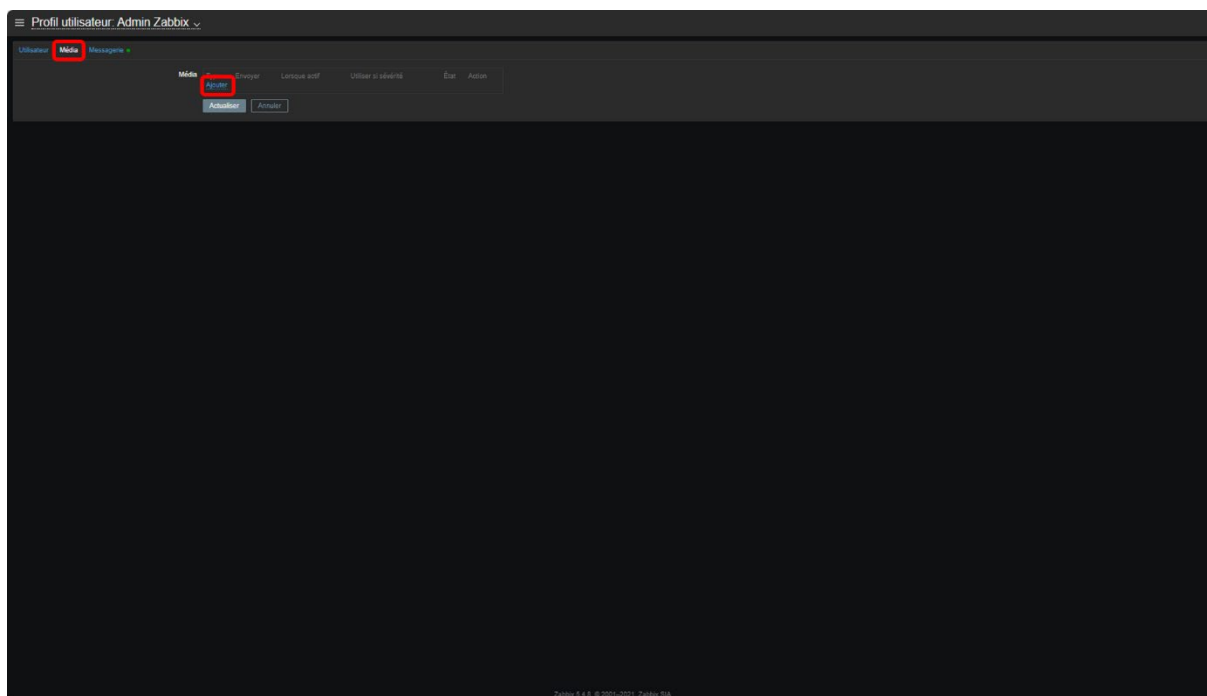
### I.III.II E-mail

Les options d'utilisateurs sont dans l'onglet *profile* dans la rubrique *User settings*



Accéder à [Média](#)

Puis [Ajouter](#) un nouveau média



Configurer les propriétés



**Média**

Type: Email

\* Envoyer: Supprimer Ajouter

\* Lorsque actif: 1-7,00:00-24:00

Utiliser si sévérité:

- ☒ Non classé
- ☒ Information
- ☒ Avertissement
- ☒ Moyen
- ☒ Haut
- ☒ Désastre

Activé: ☒

Actualiser Annuler

**Type**: Email

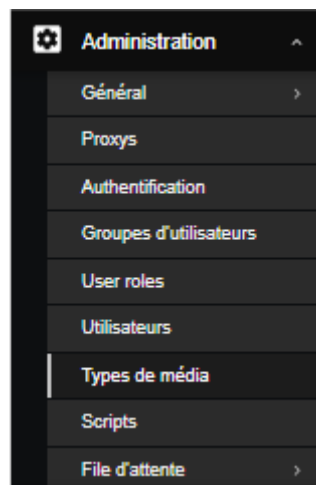
**Envoyer**: Email du destinataire, pour ajouter plusieurs destinataires cliquer sur Ajouter

**Lorsque actif**: Horaire d'activation d'envoi de mail {étendue des jour de la semaine (1 -> lundi, 7 -> dimanche), heure format 24h hh:mm}.

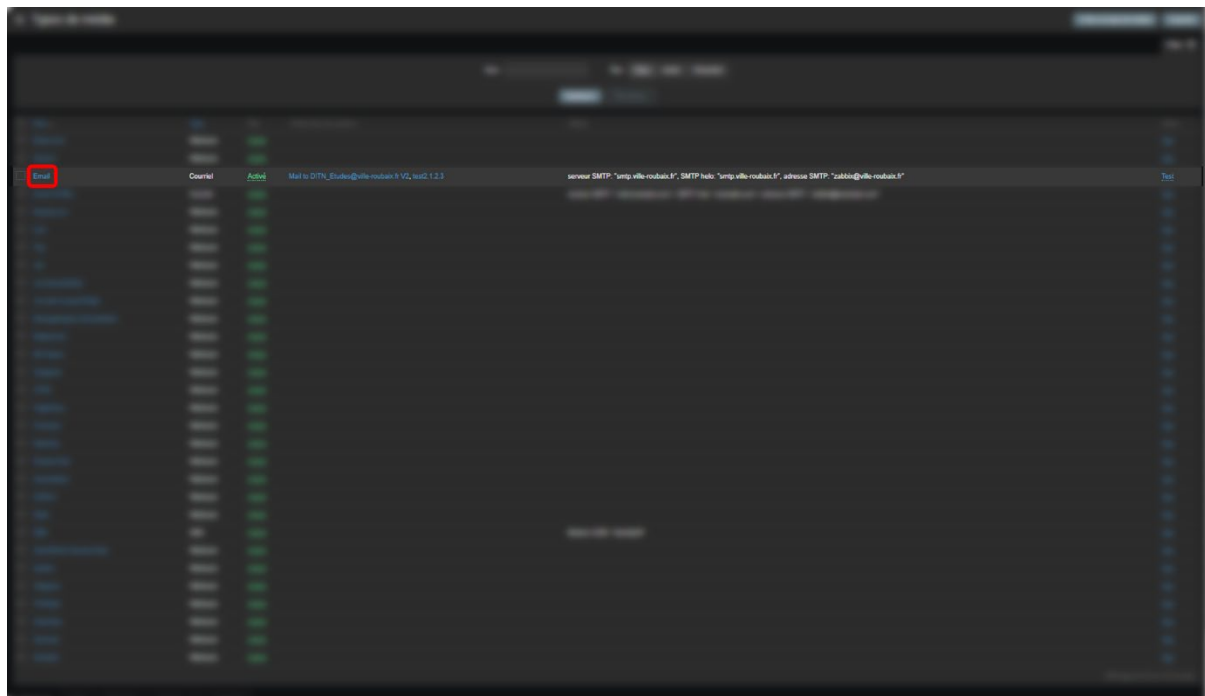
**Utilisé si sévérité**: Les alertes mails seront envoyer selon les niveaux de sévérité sélectionner

**Activé**: Activé l'envoi de mail à cette adresse

Accéder à l'onglet *Type de média* dans la rubrique *Administration*.



Cliquer sur Email



Configuration des propriétés

Types de média

Type de média Message templates Options

Nom Email

Type Courriel

serveur SMTP smtp.ville-roubaix.fr

Port du serveur SMTP 25

SMTP helo smtp.ville-roubaix.fr

adresse SMTP zabbix@ville-roubaix.fr

Sécurité de la connexion Aucun STARTTLS SSLTLS

Authentification Aucun Nom d'utilisateur et mot de passe

Format du message HTML Texte brut

Description test

Activé

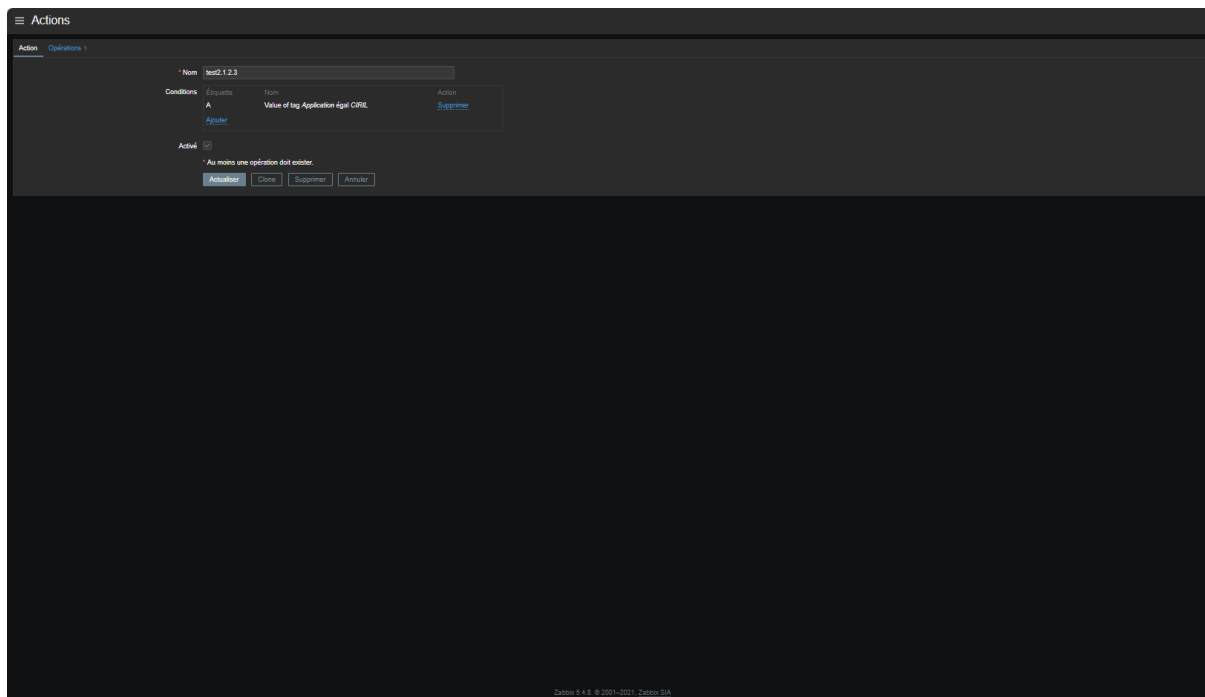
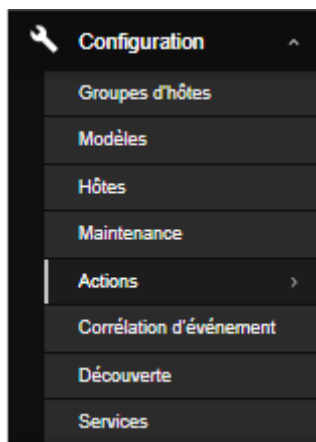
Actualiser Clone Supprimer Annuler

Zabbix 5.4.8 © 2001-2021, Zabbix SIA

Indiquer :

**Serveur SMTP**  
**Port du server SMTP**  
**SMTP Helo**  
**Adresse SMTP**

Configurer les actions dans le sous-onglet *Internal actions* de l'onglet *Actions* dans la rubrique *configuration*



**Nom** (requis)

**Condition** : Défini les conditions d'envoi de mail

Nouvelle condition

Type

Valeur du tag

Étiquette

Opérateur

égal

n'est pas égal

contient

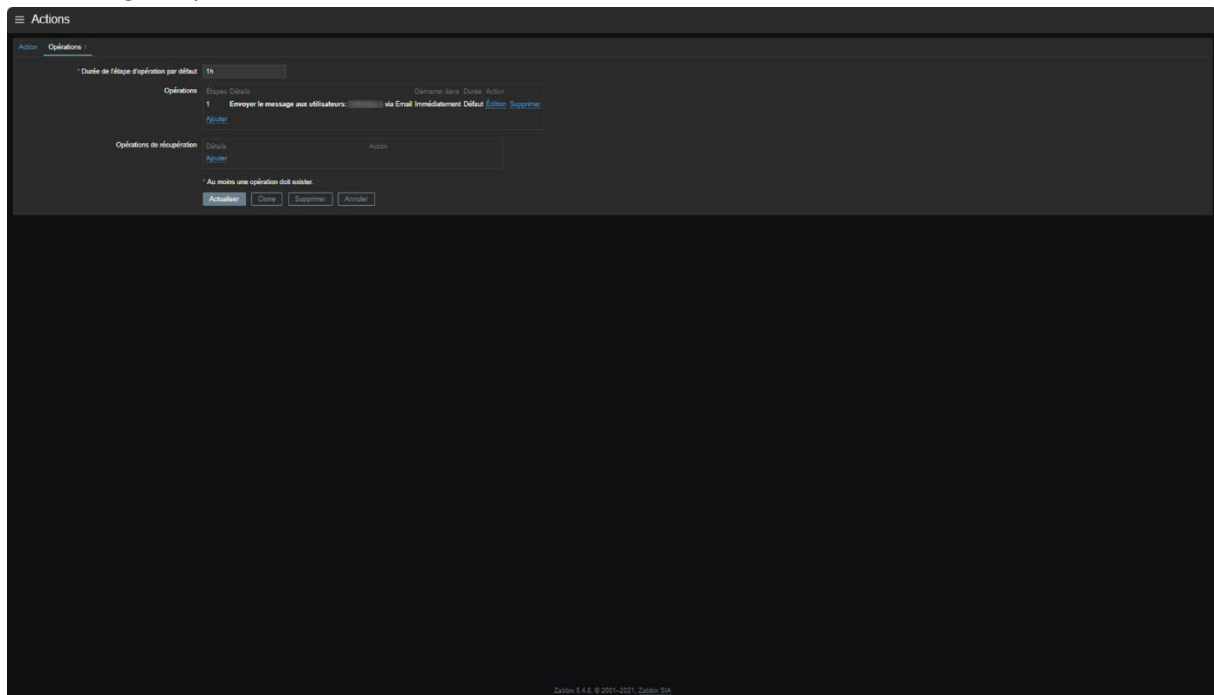
ne contient pas

Valeur

Ajouter

Annuler

**Activé:** Activer l'action  
Dans l'onglet *opération*



**Durée de l'étape d'opération par défaut:**  
**Opérations:** Opération à effectuer lors d'une alerte

**Détails de l'opération**

Operation: Envoi message

Étapes: 1 - 1 (0 - indéfiniment)

Durée de l'étape: 0 (0 - utiliser les paramètres par défaut de l'action)

\* Au moins un utilisateur ou un groupe d'utilisateurs doit être sélectionné.

Send to user groups: Groupe d'utilisateurs Action  
Ajouter

Send to users: Utilisateur Action  
[redacted] Supprimer  
Ajouter

Envoyer uniquement à: Email

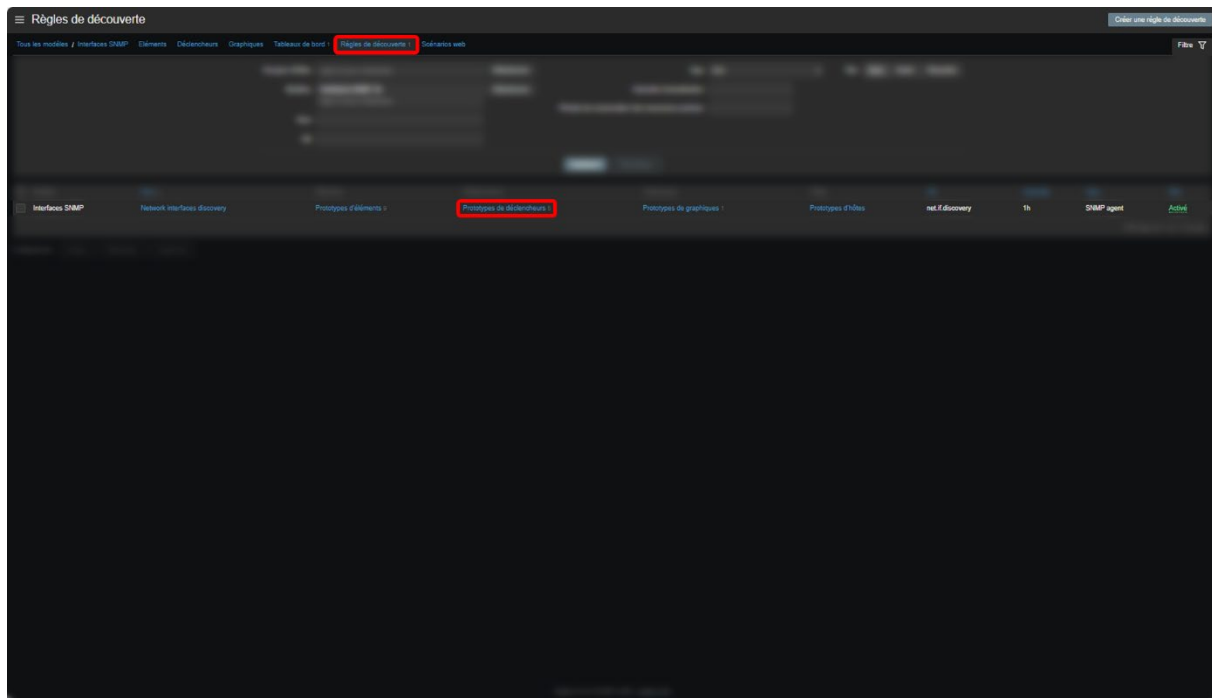
Custom message: ☐

Update Annuler

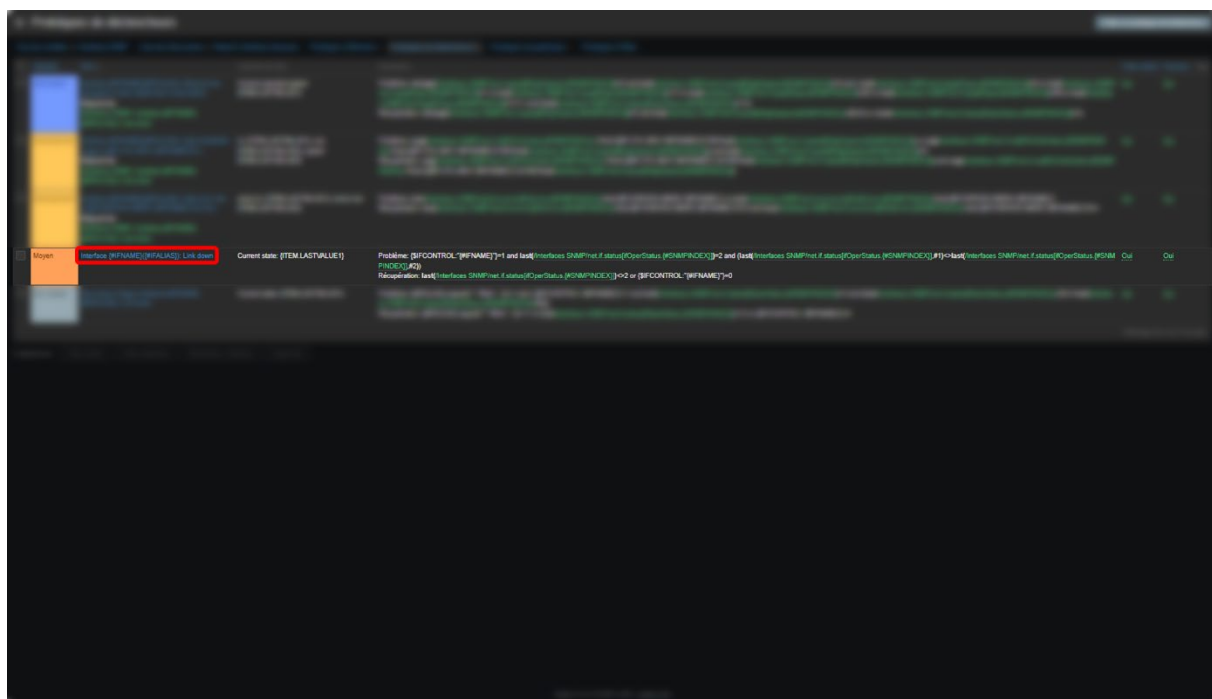
**Opération de récupération:** Opération à effectuer lors de la récupération



Cliquer sur *Règle de découvertes* puis *Prototypes de déclencheurs*.



Cliquer sur un déclencheur à cloner.



Et le cloner en cliquant sur Cloner

Modifier le nom

Ajouter dans *Expression de problème* la fonction de macro

`{{#IFALIAS}.regsub("^.*ALIASNAME", 1)}=1 and`



```
1 {{#IFALIAS}.regsub("^.*ALIASNAME", 1)}=1 and
```

Remplacer « ALIASNAME » par le nom d’alias recherché

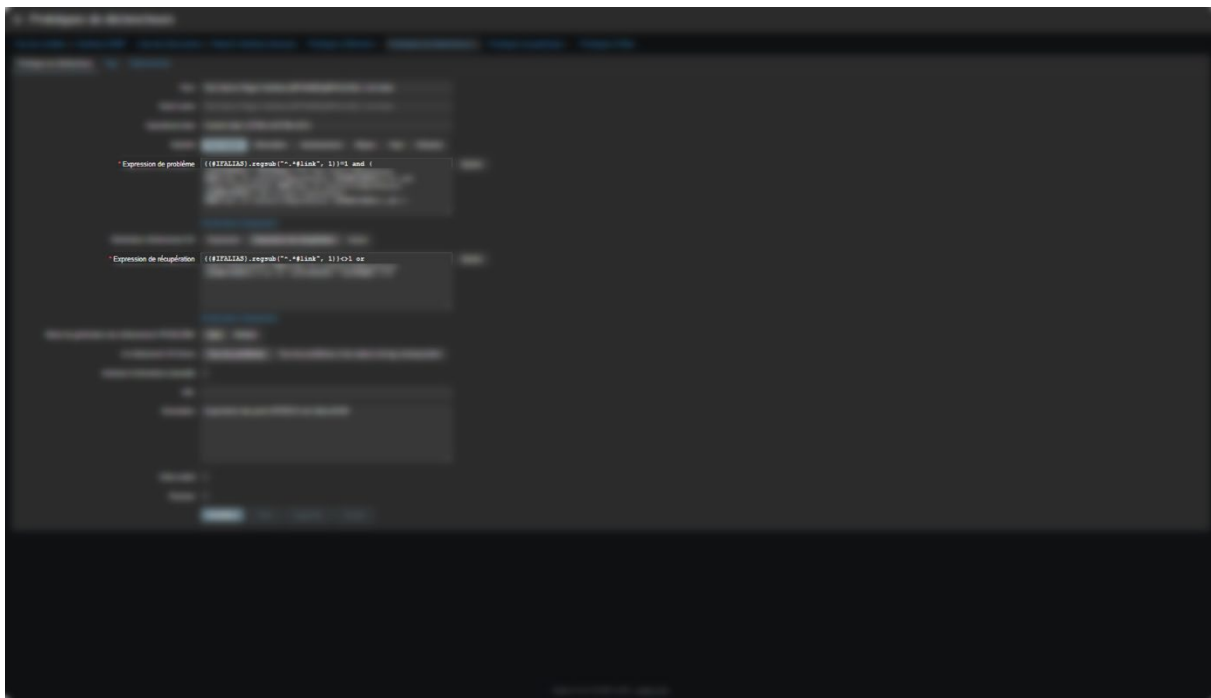
Ajouter dans *Expression de récupération* la fonction de macro

`{{#IFALIAS}.regsub("^.*ALIASNAME", 1)}<>1 or`



```
1 {{#IFALIAS}.regsub("^.*ALIASNAME", 1)}<>1 or
```

Remplacer « ALIASNAME » par le nom d’alias recherché

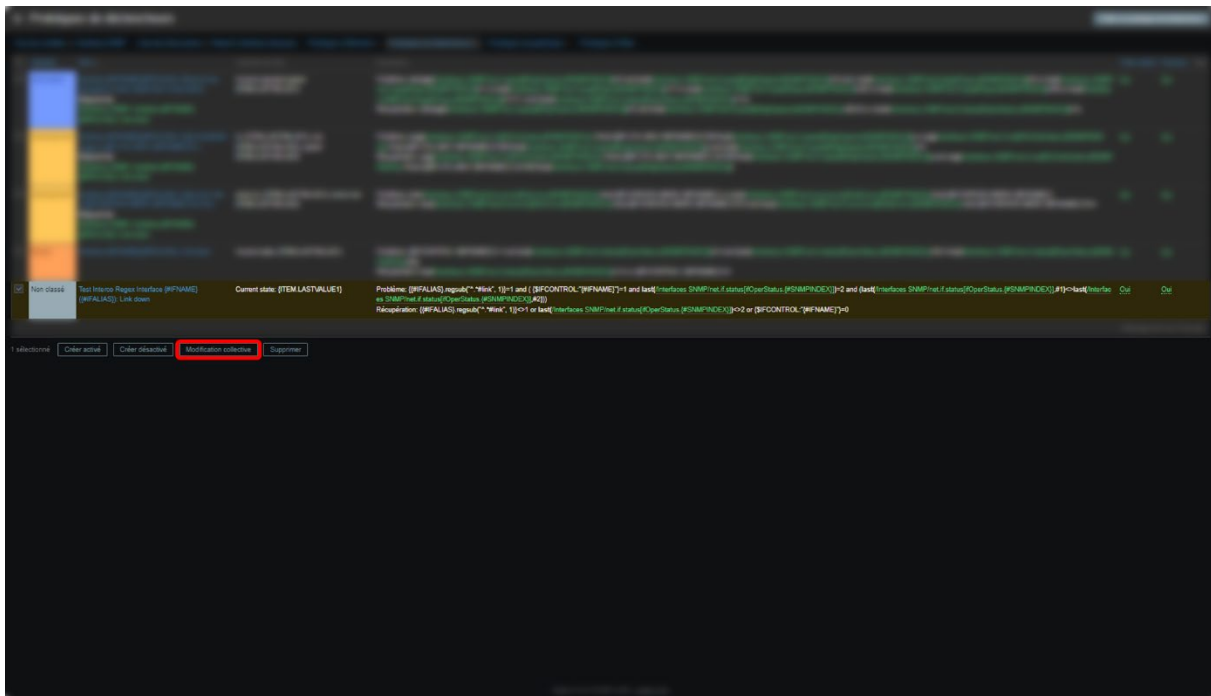


Valider en cliquant sur [Ajouter](#)

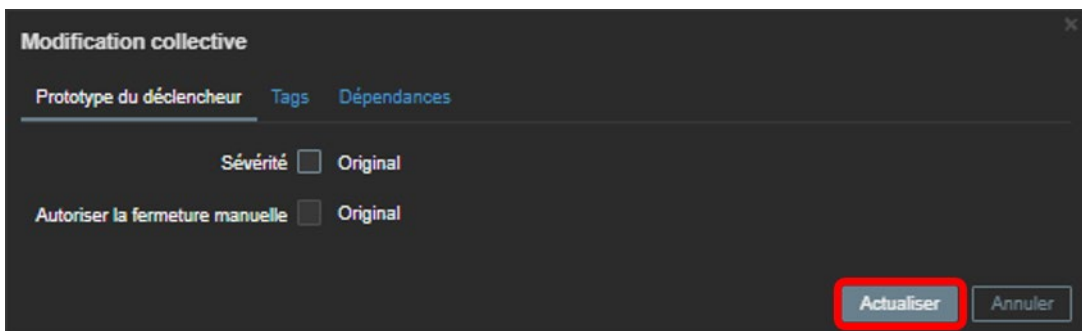


### I.IV.I.II Activation de la découverte

Cocher les déclencheurs nouvellement créés et cliquer sur **Modification collective**



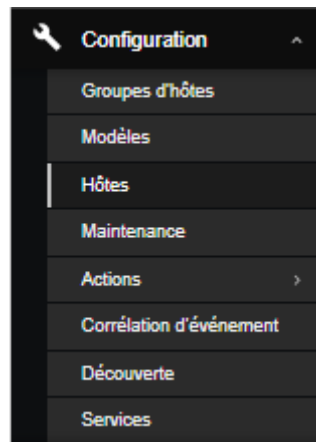
Lancer la nouvelle découverte en cliquant sur **Actualiser**



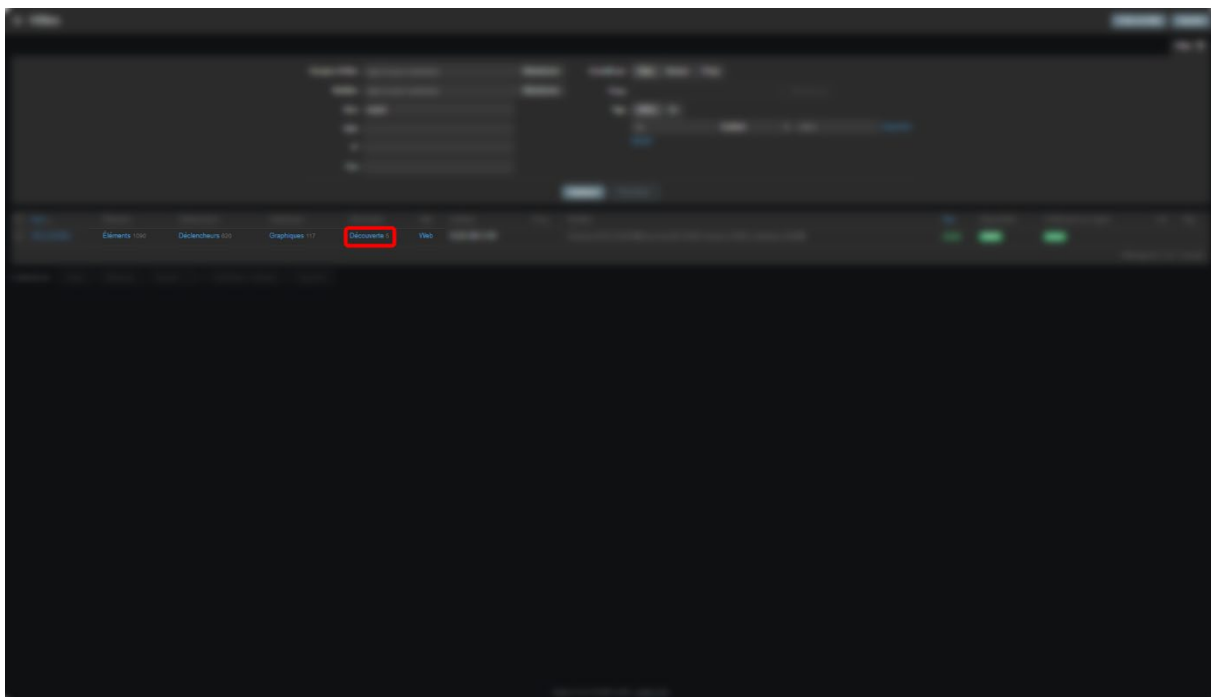
L'affichage des nouveaux déclencheurs dans Zabbix peut prendre plusieurs minutes

#### I.IV.I.III Découverte manuelle

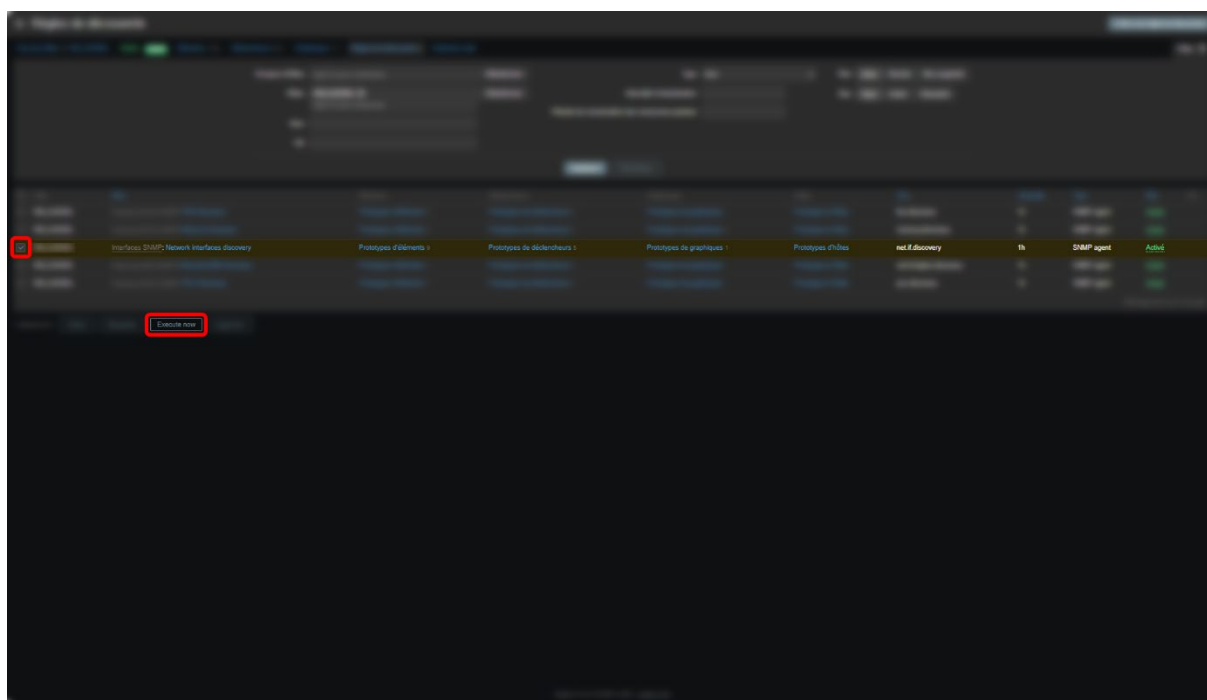
Pour relancer une découverte réseaux manuellement, il faut se rendre dans l'onglet *Hôtes* de la rubrique *Configuration*.



Sélectionner l'hôte rechercher et cliquer sur **Découverte**



Sélectionner la ligne **Interfaces SNMP: Network interfaces discovery** via la CheckBox et cliquer sur **Execute now**



Un message de confirmation sera affiché à l'écran



## Annexe II.: Code Python

### II.I Main.py

```
import csv,logs,argparse,logging
from logging.handlers import TimedRotatingFileHandler
from switch import SSH,Telnet
from zabbix import Zabbix
from dotenv import load_dotenv
from time import sleep
from os import getenv, environ, path

# class logging.handlers.SysLogHandler(address=('localhost', SYSLOG_UDP_PORT),
# facility=LOG_USER, socktype=socket.SOCK_DGRAM)

load_dotenv("C:\\Scripts\\ENV_GIT_Main\\.env")
sleep(1)
DIR: str | None = getenv("DIR")
BASEFILE: str | None = getenv("BASEFILE")
ZABBIXIP: str | None = environ.get("ZABBIXIP")
ZABBIXTOKEN: str | None = environ.get("ZABBIXTOKEN")
LOG_DIR: str | None = getenv("LOG_DIR")
LOG_LEVEL: str | None = getenv("LOG_LEVEL")
STREAM_HANDLER: str | None = getenv("STREAM_HANDLER")
DISCONNECT: str | None = getenv("DISCONNECT")
PASS_BIS: str | None = getenv("PASS_BIS")
LOG_FILE: str | None = LOG_DIR+"\\.log" if LOG_LEVEL else None
ERR_LOG_CSV: str | None = LOG_DIR+"\\err.csv" if LOG_LEVEL else None

AVAYA_USER: str | None = getenv("AVAYA_USER") if not environ.get("AVAYA_USER") else
environ.get("AVAYA_USER")
AVAYA_PASS: str | None = getenv("AVAYA_PASS") if not environ.get("AVAYA_PASS") else
environ.get("AVAYA_PASS")
ARUBA_ERS_USER: str | None = getenv("ARUBA_ERS_USER") if not
environ.get("ARUBA_ERS_USER") else environ.get("ARUBA_ERS_USER")
ARUBA_ERS_PASS: str | None = getenv("ARUBA_ERS_PASS") if not
environ.get("ARUBA_ERS_PASS") else environ.get("ARUBA_ERS_PASS")
ARUBA_ERS_USER_BIS: str | None = getenv("ARUBA_ERS_USER_BIS") if not
environ.get("ARUBA_ERS_USER_BIS") else environ.get("ARUBA_ERS_USER_BIS")

logLevel: dict = {
    "DEBUG":logging.DEBUG,
    "INFO":logging.INFO,
    "WARNING":logging.WARNING,
    "ERROR":logging.ERROR,
    "CRITICAL":logging.CRITICAL
}
```

```

# ----- log standard -----
logger_main = logging.getLogger(__name__)
logger_main.setLevel(logLevel[LOG_LEVEL])
formater_main = logging.Formatter('%(asctime)s:%(levelname)s:%(name)s:ligne_%(lineno)d
%(message)s')
file_handler_main = TimedRotatingFileHandler(
    filename=LOG_FILE, # type: ignore
    when='H',
    interval=24,
    backupCount=5,
    encoding='utf-8'
)
file_handler_main.setFormatter(formater_main)
file_handler_main.setLevel(logLevel[LOG_LEVEL])

stream_handler_main = logging.StreamHandler()
stream_handler_main.setFormatter(formater_main)

logger_main.addHandler(file_handler_main)
logger_main.addHandler(stream_handler_main) if STREAM_HANDLER.lower() == 'true' else
...

rotatingLogger: bool = True

# -----

def GetParser() -> argparse.Namespace:
    parser = argparse.ArgumentParser(description="Sauvegarde des configuration de
Switch via SSH et Telnet")

    parser.add_argument('-f', '--filelog', dest="ZabbixFileLog", help="Sauvegarder
la liste des switchs trouvé dans Zabbix", required=False, action='store_true')
    parser.add_argument('-g', '--group', dest="group", help="renvoie la liste des
groupes trouvé", required=False, action='store_true')
    parser.add_argument('--src', type=str, nargs="*", dest="source", help='source du
fichier csv', required=False)

    return parser.parse_args()

@logs.Timer
def Reroll(fileList: list):
    for fLst in fileList:
        with open(fLst, 'w') as newLog:
            newLog.close()

@logs.Timer
def OpenCSV(file:str) -> list:

```

```

switchLst: list=[]

with open(file,"r") as c:
    read = csv.reader(c, delimiter=";")
    logger_main.debug(f"Lecture .csv: {read}")
    for row in read:
        logger_main.debug(f"{read}")
        if 0<len(row):
            if row[0] == "type":
                continue
            match(row[4]):
                case "aruba_os":
                    device: dict = {
                        "ip":row[3],
                        "username":ARUBA_ERS_USER,
                        "password":ARUBA_ERS_PASS,
                        "device_type":row[4]
                        # 'use_keys': True, # Activer cle ssh
                        # 'key_file': '/data/05_PYTHON_DEMO/SSH_KEY/admin1' #
chemin cle ssh
                    }
                case "avaya_ers":
                    device: dict = {
                        "ip":row[3],
                        "username":AVAYA_USER,
                        "password":AVAYA_PASS,
                        "device_type":row[4]
                        # 'use_keys': True, # Activer cle ssh
                        # 'key_file': '/data/05_PYTHON_DEMO/SSH_KEY/admin1' #
chemin cle ssh
                    }
                case _:
                    raise ValueError
            s=[row[0], row[1], row[2], device]
            switchLst.append(s)

    return switchLst

def DisconnectSwitch(file:str) -> list:
    switchLst: list=[]

    with open(file,"r") as c:
        read = csv.reader(c, delimiter=";")
        for row in read:
            switchLst.append(row)

    return switchLst

```

```

def PassBisSwitch(file:str) -> list:
    switchLst: list=[]

    with open(file,"r") as c:
        read = csv.reader(c, delimiter=";")
        for row in read:
            switchLst.append(row)

    return switchLst

def WriteCSV(file: str, msg: list) -> None:
    logger_main.debug(f"Call by {msg}")
    with open(file, 'a') as f:
        writer = csv.writer(f, delimiter=";")
        writer.writerow(msg)
        logger_main.debug(f"Write in {f}; msg: {msg}")
        f.close()

@logs.Timer
def saveSwitch(switch: list, Zapi: Zabbix) -> None:
    match(switch[0]):
        case "ssh":
            ssh = SSH(DIR, switch[1], switch[2], switch[3]) # type: ignore
            ssh.main()
            Zapi.Declancheur(ssh.Name, ssh.File) # type: ignore
        case "telnet":
            tn = Telnet(DIR, switch[1], switch[2], switch[3]) # type: ignore
            tn.main()
            Zapi.Declancheur(tn.Name, tn.File) # type: ignore
        case _:
            logger_main.error(f"{switch[1]}:{switch[1]}: Erreur lors de la creation de la class (type de connexion errone).")

@logs.Timer
def main(args: argparse.Namespace) -> int:
    device: dict
    sw: tuple
    succ: int = 0; err: int = 0;
    if args.group:
        ZabbixGrp = Zabbix(ZABBIXIP, ZABBIXTOKEN) # type: ignore
        ZabbixGrp.connexion()
        grp = ZabbixGrp.GetGroup()
        from pprint import pprint
        pprint(grp, indent=4)
        exit()
    if (args.source != None):
        logger_main.info(f"run from '--src' {args.source}")
        Zapi = Zabbix(ZABBIXIP, ZABBIXTOKEN) # type: ignore

```

```

    Zapi.connexion()
    Zapi.Disconnect = DisconnectSwitch(DISCONNECT) if path.exists(DISCONNECT) else
[]

    Zapi.PassBis = PassBisSwitch(PASS_BIS) if path.exists(PASS_BIS) else []
    for file in args.source:
        switchLst = OpenCSV(file)
        logger_main.debug(f"Switchs recuperer: {len(switchLst)} -> {switchLst}")
        for switch in switchLst:
            total: int = len(switchLst)
            logger_main.info(f"Total de switch trouvé: {total:>4}")
            try:
                saveSwitch(switch, Zapi)
                succ += 1
            except Exception as e:
                err += 1
                row = [switch[0],switch[1],switch[2],switch[3],switch[6]]
                WriteCSV(ERR_LOG_CSV, row)
                logger_main.exception(e)
            finally:
                progress: str = f"success: {succ:>3}/{total:>3}; error
{err:>3}/{total:>3};"
                logger_main.info(f"\n\n{"-"*25} {progress:^40} {"-"*25}\n")
                progress: str = f"progress: {((succ+err)*100)/total:>6.2f}%"
                logger_main.info(f"\n\n{"-"*25} {progress:^40} {"-"*25}\n")
                continue
        else:
            logger_main.info(f"run from Default Zabbix")
            Reroll([ERR_LOG_CSV])
            Zapi = Zabbix(ZABBIXIP, ZABBIXTOKEN) # type: ignore
            Zapi.connexion()
            Zapi.Disconnect = DisconnectSwitch(DISCONNECT) if path.exists(DISCONNECT) else
[]

            Zapi.PassBis = PassBisSwitch(PASS_BIS) if path.exists(PASS_BIS) else []
            swLst = Zapi.main(args.ZabbixFileLog)
            total: int = len(swLst)
            logger_main.info(f"Total de switch trouvé: {total:>4}")
            for s in swLst:
                device = {
                    "ip":s[3],
                    "username":s[4],
                    "password":s[5],
                    "device_type":s[6],
                    "secret":s[7],
                    "conn_timeout":30
                }
                sw = (s[0], s[1], s[2], device)

            try:

```



```

        saveSwitch(sw, Zapi)
        succ += 1
    except Exception as e:
        err += 1
        row = [sw[0],sw[1],sw[2],device["ip"],device["device_type"]]
        WriteCSV(ERR_LOG_CSV, row)
        logger_main.exception(e)
    finally:
        progress: str = f"success: {succ:>3}/{total:>3}; error:
{err:>3}/{total:>3};"
        logger_main.info(f"\n\n{"-"*25} {progress:^40} {"-"*25}\n")
        progress: str = f"progress: {((succ+err)*100)/total:>6.2f}%"
        logger_main.info(f"\n\n{"-"*25} {progress:^40} {"-"*25}\n")
        continue

    return 0

if __name__=="__main__":
    try:
        Reroll([LOG_FILE])
        logger_main.info(f"\n\n{'='*100}\n\n{"***20}{"Debut du
programme":^30}{"***20}\n\n")
        args = GetParser()
        main(args)
    except Exception as e:
        logger_main.exception(e)
        logger_main.info(f'\n\n\n{"***20}{"Arret du
programme":^30}{"***20}\n\n{'='*100}\n')
    try:
        file_handler_main.doRollover()
    except PermissionError as p:
        logger_main.debug(f"Impossible deffectuer une rotation de log via logging:
{p}")
    rotatingLogger = False
    finally:
        logs.Rotation([LOG_FILE]) if not rotatingLogger else ...
    exit()

```

## II.II switch.py

```
import tempfile, datetime, logs, logging
from logging.handlers import TimedRotatingFileHandler
from gitManager import GitManager
from dotenv import load_dotenv
from os import path, getenv

class ErrorSize(Exception):
    """Le fichier de configuration est vide"""
    f"Le fichier de configuration est vide"

    def __init__(self, size: int, maxSize: int):
        self.size = size
        self.maxSize = maxSize
        self.message = f"Taille du fichier ({self.size}) inferieur a: {self.maxSize} Bytes"
        super().__init__(self.message)

    """
    
    """

    class Switch:
        load_dotenv("C:\\Scripts\\ENV_GIT_Main\\.env")
        ban: tuple=("\x1b", "", "-", "\n", "# show running-config", "#show running-config", "Running configuration:")
        TODAY: str = datetime.date.today().strftime("%Y-%m-%d")
        LOG_DIR: str | None = getenv("LOG_DIR")
        LOG_LEVEL: str | None = getenv("LOG_LEVEL")
        STREAM_HANDLER: str | None = getenv("STREAM_HANDLER")
        FILE_MIN_SIZE: int | None = int(getenv("FILE_MIN_SIZE"))
        LOG_FILE: str | None = LOG_DIR+"\\.log" if LOG_LEVEL else None

        logLevel: dict = {
            "DEBUG":logging.DEBUG,
            "INFO":logging.INFO,
            "WARNING":logging.WARNING,
            "ERROR":logging.ERROR,
            "CRITICAL":logging.CRITICAL
        }
    }
```

```

logger_switch = logging.getLogger(__name__)
logger_switch.setLevel(logLevel[LOG_LEVEL])

                                     formater_switch
logging.Formatter('%(asctime)s:%(levelname)s:%(name)s:ligne_%(lineno)d      =>
%(message)s')

file_handler_switch = TimedRotatingFileHandler(
    filename=LOG_FILE, # type: ignore
    when='H',
    interval=24,
    backupCount=5,
    encoding='utf-8'
)
file_handler_switch.setFormatter(formater_switch)
file_handler_switch.setLevel(logLevel[LOG_LEVEL])

stream_handler_switch = logging.StreamHandler()
stream_handler_switch.setFormatter(formater_switch)

logger_switch.addHandler(file_handler_switch)
logger_switch.addHandler(stream_handler_switch) if STREAM_HANDLER.lower() ==
'true' else ...

def __init__(self, pDir:str, pName:str, pLoc:str, pProp:dict, pConnType: str) ->
None:
    self.name: str = pName
    self.localisation: str = pLoc
    self.property: dict = pProp
    self.connType: str = pConnType
    self.dir: str = pDir+"\\ "+self.localisation+"\\ "+self.name
    self.file: str = self.dir+"\\ "+self.name+".cfg"
    self.git: GitManager = GitManager(self.dir, self.localisation+"\\ "+self.name)
    Switch.logger_switch.debug(f"Creation: {self}")

def __str__(self) -> str:
    return f"{self.name}, type de connexion: {self.connType}, local:
{self.localisation}, ip: {self.property["ip"]}, OS type:
{self.property["device_type"]}"

@property
def Name(self) -> str:
    return self.name

@property
def Dir(self) -> str:
    return self.dir

```

```

@property
def File(self) -> str:
    return self.file

@logs.Timer
def saveConf(self, conf:str) -> None:
    Switch.logger_switch.info("Sauvegarde de la configuration")
    tmp:      tempfile._TemporaryFileWrapper[str] =
tempfile.TemporaryFile("w+t",delete=False)
    with open(tmp.name, "w") as f:
        f.write(conf)
        Switch.logger_switch.debug(f"Creation fichier temporaire: {tmp.name}")
        Switch.logger_switch.debug(f"sizeof: {tmp.name}, {path.getsize(tmp.name)}
Bytes")
    if path.getsize(tmp.name) <= Switch.FILE_MIN_SIZE:
        Switch.logger_switch.error(f"Taille du fichier inferieur a:
{Switch.FILE_MIN_SIZE} Bytes")
        raise ErrorSize(path.getsize(tmp.name), Switch.FILE_MIN_SIZE)
    else:
        with open(self.file, "w") as c:
            with open(tmp.name,"r") as f:
                lines = f.readlines()
                for line in lines:
                    c.write(line) if not(line.startswith(Switch.ban) or
line.upper().startswith(self.name.upper()+"#") or line.__contains__("# show running-
config")) else ...
            f.close()
            c.close()
            msg = f'Sauvegarde le la config {self.name} -
{datetime.datetime.now().strftime("%Y-%m-%d.%Hh%M")}'
            self.git.main([self.file], msg)

def __del__(self) -> None:
    Switch.logger_switch.debug(f"Destruction: {self}")
    pass

"""

"""

```

```

from netmiko import ConnectHandler, BaseConnection # type: ignore
class SSH(Switch):

    def __init__(self, pDir:str, pName:str, pLoc:str, pProp:dict) -> None:
        self.connType = "ssh"
        self.conn: BaseConnection
        super().__init__(pDir, pName, pLoc, pProp, self.connType)
        Switch.logger_switch.debug(f"Creation: {self}")

    @logs.Timer
    def connection(self) -> bool:
        Switch.logger_switch.info(f"{self.name}:{self.property['ip']}: Connexion")
        self.conn = ConnectHandler(**self.property)
        self.conn.enable()
        # self.conn.send_command_timing("enable")
        # self.conn.send_command_timing(self.property['secret'])
        return True

    @logs.Timer
    def showRun(self) -> str:
        Switch.logger_switch.debug(f"{self.name}:{self.property['ip']}: Envoie commande show running-config")
        return self.conn.send_command_timing("display current-configuration",
last_read=45.0, read_timeout=120.0)
        # if "VSP" in self.name or "vsp" in self.name:
        #     return self.conn.send_command_timing("show running-config",
last_read=45.0, read_timeout=120.0)
        # elif self.property["device_type"]=="huawei":
        #     return self.conn.send_command_timing("display current-configuration",
last_read=45.0, read_timeout=120.0)
        # else:
        #     return self.conn.send_command_timing("show running-config")

    """
    
    """

    @logs.Timer
    def main(self) -> int:
        isConnect: bool

```

```

        isConnect = self.connection()
        if isConnect:
            output: str = self.showRun()
            self.saveConf(output)
            self.conn.disconnect()
            Switch.logger_switch.debug(f"{self.name}:{self.property['ip']}:
Deconnexion")
            return 0
        else:
            Switch.logger_switch.error(f"{self.name}:{self.property['ip']}: connexion
impossible")
            return 1

"""

"""

import telnetlib
from time import sleep
class Telnet(Switch):

    start: str = "\x19"
    space: str = "\x20"
    enter: str = "\x0D"

    def __init__(self, pDir:str, pName:str, pLoc:str, pProp:dict) -> None:
        self.connType = "telnet"
        self.conn: telnetlib.Telnet
        super().__init__(pDir, pName, pLoc, pProp, self.connType)
        Switch.logger_switch.debug(f"Creation: {self}")

    @logs.Timer
    def connection(self) -> tuple[bool, Exception | None]:
        Switch.logger_switch.info(f"{self.name}:{self.property['ip']}: Connexion")
        self.conn = telnetlib.Telnet(self.property["ip"], 23, timeout=2)
        sleep(1)
        self.conn.write(Telnet.start.encode("ascii"))
        sleep(1)

        #
        Switch.logger_switch.debug(f"{self.conn.read_very_eager().decode('ascii')}"*100)
    ) # Voir la console dans les logs direct

```

```

        # self.conn.read_until(b"Username:") # switch test: SW_IUTC_2eme ne permet
pas d'utiliser read_until UI non compatible
        # sleep(1)
        self.conn.write(self.property["username"].encode("ascii")+b"\n")
        self.conn.write(b"\r")
        sleep(3)

#
lSwitch.logger_switch.debug(f"{self.conn.read_very_eager().decode('ascii')}") # Voir
la console dans les logs direct
        # self.conn.read_until(b"Password:") # switch test: SW_IUTC_2eme ne permet
pas d'utiliser read_until UI non compatible
        # sleep(1)
        self.conn.write(b"\r")
        self.conn.write(self.property["password"].encode(encoding="ascii")+b"\n")

#
Switch.logger_switch.debug(f"{self.conn.read_very_eager().decode('ascii')}") # Voir
la console dans les logs direct
        sleep(3)
        self.conn.write(b"\r")
        # self.conn.read_until(b"#") # switch test: SW_IUTC_2eme ne permet pas
d'utiliser read_until UI non compatible
        # sleep(1)
        self.conn.write("enable".encode(encoding="ascii")+b"\n")
        sleep(3)
        self.conn.write(b"\r")
        # self.conn.read_until(b"#") # switch test: SW_IUTC_2eme ne permet pas
d'utiliser read_until UI non compatible
        # sleep(1)

#
Switch.logger_switch.debug(f"{self.conn.read_very_eager().decode('ascii')}") # Voir
la console dans les logs direct
        Switch.logger_switch.info(f"{self.name}:{self.property['ip']}: Connexion
semble effectue")
        return True, None

@logs.Timer
def showRun(self) -> str:
    sleep(3)
    self.conn.write(b"\r")

#
Switch.logger_switch.debug(f"{self.conn.read_very_eager().decode('ascii')}") # Voir
la console dans les logs direct
        Switch.logger_switch.debug(f"{self.name}:{self.property['ip']}: Envoie
commande show running-config")
        self.conn.write("show running-config".encode("ascii"))
        sleep(.5)
        self.conn.write(b"\r")

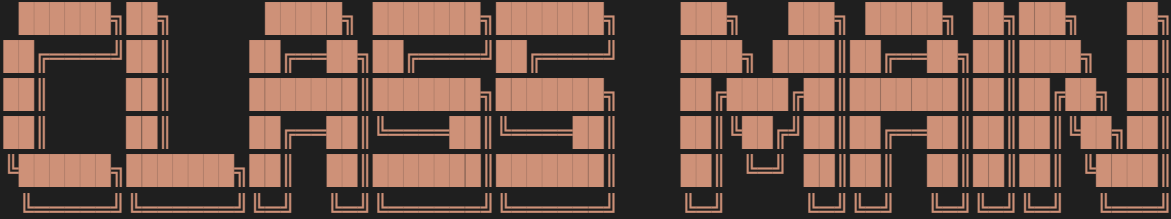
```

```

#
Switch.logger_switch.debug(f"{self.conn.read_very_eager().decode('ascii')}") # Voir
la console dans les logs direct
    sleep(2)
    for _ in range(0,11):
        self.conn.write(Telnet.space.encode("ascii"))
        sleep(2)

#
Switch.logger_switch.debug(f"{self.conn.read_very_eager().decode('ascii')}") # Voir
la console dans les logs direct
    Switch.logger_switch.debug(f"{self.name}:{self.property['ip']}: Commande show
running-config semble effectue")
    return self.conn.read_very_eager().decode("ascii")

"""



"""

@logs.Timer
def main(self) -> int:
    isConnect: bool; e: Exception | None
    isConnect,e = self.connection()
    if isConnect:
        output: str = self.showRun()
        self.saveConf(output)
        self.conn.write(b"exit")
        self.conn.write(b"\r")
        Switch.logger_switch.info(f"{self.name}:{self.property['ip']}:
Deconnexion")
        return 0
    else:
        Switch.logger_switch.error(f"{self.name}:{self.property['ip']}: {e}")
        return 1

```



## II.III Zabbix.py

```
import logs, datetime, logging
from logging.handlers import TimedRotatingFileHandler
from pyzabbix import ZabbixAPI
from dotenv import load_dotenv
from os import getenv, environ

class Zabbix:
    load_dotenv("C:\\Scripts\\ENV_GIT_Main\\.env")
    TODAY: str = datetime.date.today().strftime("%Y-%m-%d")
    CISCO_IOS_USER: str | None = getenv("CISCO_IOS_USER") if not
environ.get("CISCO_IOS_USER") else environ.get("CISCO_IOS_USER")
    CISCO_IOS_PASS: str | None = getenv("CISCO_IOS_PASS") if not
environ.get("CISCO_IOS_PASS") else environ.get("CISCO_IOS_PASS")
    CISCO_IOS_SECRET: str | None = getenv("CISCO_IOS_SECRET") if not
environ.get("CISCO_IOS_SECRET") else environ.get("CISCO_IOS_SECRET")
    TELNET_USER: str | None = getenv("TELNET_USER") if not environ.get("TELNET_USER")
else environ.get("TELNET_USER")
    TELNET_PASS: str | None = getenv("TELNET_PASS") if not environ.get("TELNET_PASS")
else environ.get("TELNET_PASS")
    DELAY: str | None = getenv("DELAY")
    INTERVAL: str | None = getenv("INTERVAL")
    ITEM_AGE: str | None = getenv("ITEM_AGE")
    ITEM_SIZE: str | None = getenv("ITEM_SIZE")
    TRIGGER_PRIORITY: int | None = int(getenv("TRIGGER_PRIORITY"))
    LOG_DIR: str | None = getenv("LOG_DIR")
    LOG_LEVEL: str | None = getenv("LOG_LEVEL")
    STREAM_HANDLER: str | None = getenv("STREAM_HANDLER")
    LOG_FILE: str | None = LOG_DIR+"\\.log" if LOG_LEVEL else None

    logLevel: dict = {
        "DEBUG": logging.DEBUG,
        "INFO": logging.INFO,
        "WARNING": logging.WARNING,
        "ERROR": logging.ERROR,
        "CRITICAL": logging.CRITICAL
    }

    triggerPriority: dict = {
        0: "Non-classe",
        1: "Information",
        2: "Avertissement",
        3: "Moyen",
        4: "Haut",
        5: "Desastre"
    }

    logger_zabbix = logging.getLogger(__name__)
```

```

logger_zabbix.setLevel(logLevel[LOG_LEVEL])

                                formater_zabbix                                =
logging.Formatter('%(asctime)s: %(levelname)s: %(name)s: ligne_%(lineno)d      ->
%(message)s')

file_handler_zabbix = TimedRotatingFileHandler(
    filename=LOG_FILE, # type: ignore
    when='H',
    interval=24,
    backupCount=5,
    encoding='utf-8'
)
file_handler_zabbix.setFormatter(formater_zabbix)
file_handler_zabbix.setLevel(logLevel[LOG_LEVEL])

stream_handler_zabbix = logging.StreamHandler()
stream_handler_zabbix.setFormatter(formater_zabbix)

logger_zabbix.addHandler(file_handler_zabbix)
logger_zabbix.addHandler(stream_handler_zabbix) if STREAM_HANDLER.lower() ==
'true' else ...

SWITCHID: dict={
    "Switchs":22,
    "Routers":23,
    "pyZabbixGitFileGroup": 24
}
os: dict = {
    29:"aruba_os",
    30:"avaya_ers"
}

tagList: dict = {
    'connexion_type': 'telnet',
    'device_type': 'avaya_ers'
}

def __init__(self, ip: str, token: str) -> None:
    self.ip: str = ip
    self.token: str = token
    self.api: ZabbixAPI
    self.diconnect: list = []
    self.passBis: list = []
    # Zabbix.logger_zabbix.debug(f"Creation: {self}")

def __str__(self) -> str:
    return f"Serveur Zabbix: {self.ip}>15}"

```

```

def __del__(self) -> None:
    # Zabbix.logger_zabbix.debug(f"Destruction: {self}")
    pass

@property
def Disconnect(self) -> list:
    return self.diconnect

@Disconnect.setter
def Disconnect(self, swLst: list) -> None:
    Zabbix.logger_zabbix.info(f"Disconnect List set")
    Zabbix.logger_zabbix.debug(f"{swLst}")
    self.diconnect = swLst

@property
def PassBis(self) -> list:
    return self.passBis

@PassBis.setter
def PassBis(self, bisLst: list) -> None:
    Zabbix.logger_zabbix.info(f"PassBis List set")
    Zabbix.logger_zabbix.debug(f"{bisLst}")
    self.passBis = bisLst

@logs.Timer
def connexion(self) -> None:
    self.api = ZabbixAPI(self.ip)
    Zabbix.logger_zabbix.info(f"Connexion: {self}")
    self.api.login(api_token=self.token)

def GetGroup(self) -> dict:
    groupe: dict = {}

    group = self.api.host.get(selectGroups='extend')
    for grp in group:
        for g in grp["groups"]:
            if not g['groupid'] in groupe.keys() and not int(g['groupid']) in
Zabbix.SWITCHID.values():
                groupe[g['name']] = int(g['groupid'])
    return groupe

def GetTag(self, seek: str = None, hostname: str = None) -> dict:
    tags: dict = {}
    hosts = self.api.host.get(selectTags='extend')
    if hostname:
        for host in hosts:
            if host['host'] == hostname:

```



```

        newpwd: bool = bool(self.GetTag("isNewPass", host['host'])) if
0<len(self.GetTag("lieux", host['host'])) else False
        pwd: str = Zabbix.CISCO_IOS_PASS_BIS if newpwd else Zabbix.CISCO_IOS_PASS
        cpwd: str = "CISCO_IOS_PASS_BIS" if newpwd else "CISCO_IOS_PASS"
        interfaces = self.api.hostinterface.get(hostids=[host["hostid"]])
        for intf in interfaces:
            h =["ssh",host['host'], grp,intf["ip"], Zabbix.CISCO_IOS_USER, pwd,
"cisco_ios", Zabbix.CISCO_IOS_SECRET]
            if filelog:
                h2 =["ssh",host['host'], grp,intf["ip"], "CISCO_IOS_USER", cpwd,
"cisco_ios", "CISCO_IOS_SECRET"]
                log.append(h2)
                hostlst.append(h)
            Zabbix.logger_zabbix.info(f"switchs, {len(hostlst):>5} cisco_ios trouves")
        return hostlst, log

@logs.Timer
def GetTelnet(self, filelog: bool) -> tuple:
    """ssh, name, group, ip, username, device_type"""
    Zabbix.logger_zabbix.debug(f"Recherche de switch: telnet")
    hostlst: list = []
    hostName: list = []
    log: list = []
    CRED: tuple = ()
    grp: str | None = None
    hostsgrp = self.api.host.get(selectTags='extend', tags=[{"tag":
"connexion_type", "value": "telnet"}])
    for host in hostsgrp:
        device = self.GetTag("device_type", host['host'])
        if 0<len(self.diconnect):
            if host['host'] in self.diconnect[0]:
                Zabbix.logger_zabbix.info(f"{host['host']} est dans la liste
deconnecte")
                continue
        if self.GetTag("device_type") == "avaya_ers":
            if 0<len(self.passBis):
                if host['host'] in self.passBis[0]:
                    CRED = Zabbix.AVAYA_USER, Zabbix.AVAYA_PASS_BIS, "avaya_ers",
"AVAYA_USER", "AVAYA_PASS_BIS"
                else:
                    CRED = Zabbix.AVAYA_USER, Zabbix.AVAYA_PASS, "avaya_ers",
"AVAYA_USER", "AVAYA_PASS"
            else:
                CRED = Zabbix.AVAYA_USER, Zabbix.AVAYA_PASS, "avaya_ers",
"AVAYA_USER", "AVAYA_PASS"
        elif self.GetTag("device_type") == "aruba_os":
            if 0<len(self.passBis):
                if host['host'] in self.passBis[0]:

```





```

        "value_type":3,
        "interfaceid":host["interfaces"][0]["interfaceid"],
        "delay":f"{Zabbix.DELAY};;{Zabbix.INTERVAL}",
        # "delay":'1d;h9',
    }
    self.api.item.create(
        hostid=itemExist['hostid'],
        name=itemExist['name'],
        key_=itemExist['key_'],
        type=itemExist['type'],
        value_type=itemExist['value_type'],
        interfaceid=itemExist['interfaceid'],
        delay=itemExist['delay'],
    )
    Zabbix.logger_zabbix.debug(f"Creation de l'item {itemExist['name']}")
    triggerExist: dict = {
        "event_name":"FileExits_"+name,
        "description":"FileExits_"+name,
        "comments":"FileExits_"+name,
        "expression":"last(/"+host['host']+ "/" +itemExist['key_']+)=0",
"recovery_mode":1,
        "recovery_expression":"last(/"+host['host']+ "/" +itemExist['key_']+)=1",
        "priority":Zabbix.TRIGGER_PRIORITY,
        "comments":f"Le fichier {name} n'a pas été trouvé."
    }
    self.api.trigger.create(triggerExist)
    Zabbix.logger_zabbix.debug(f"\n\nCreation du trigger {triggerExist['event_name']}\n")

def ItemSize(host, name: str, item:str, size: str, parentName):
    itemSize: dict = {
        "hostid":host['hostid'],
        "name":"FileSize_"+name,
        "key_":f"vfs.file.size["+item+"]",
        "type":0,
        "value_type":3,
        "interfaceid":host["interfaces"][0]["interfaceid"],
        "delay":f"{Zabbix.DELAY};;{Zabbix.INTERVAL}",
    }
    self.api.item.create(
        hostid=itemSize['hostid'],
        name=itemSize['name'],
        key_=itemSize['key_'],
        type=itemSize['type'],
        value_type=itemSize['value_type'],
        interfaceid=itemSize['interfaceid'],
        delay=itemSize['delay'],
    )

```



```

    )
    Zabbix.logger_zabbix.debug(f"Creation de l'item {itemSize['name']}")
    triggerSize: dict = {
        "event_name": "FileSize_"+name,
        "description": "FileSize_"+name,
        "expression": "last(/"+host['host']+ "/" +itemSize['key_']+" )<="+size,
        "recovery_mode": 1, "recovery_expression": "last(/"+host['host']+ "/" +it
emSize['key_']+" )>"+size,
        "priority": Zabbix.TRIGGER_PRIORITY,
        "comments": f"La taille du fichier {name} est inférieure à la limite
définie."
    }
    self.api.trigger.create(triggerSize)
    parentID = TriggerID(parentName)
    selfID = TriggerID("FileSize_"+name)
    self.api.trigger.addDependencies({"triggerid": selfID,
"dependsOnTriggerid": parentID})
    Zabbix.logger_zabbix.debug(f"\n\nCreation du trigger
{triggerSize['event_name']}\ndependant de {parentName}\n")

def ItemAge(host, name: str, item: str, age: str, parentName) -> None:
    itemAge: dict = {
        "hostid": host['hostid'],
        "name": "FileAge_"+name,
        "key_": "vfs.file.time["+item+"]",
        "type": 0,
        "value_type": 3,
        "interfaceid": host["interfaces"][0]["interfaceid"],
        "delay": f"{Zabbix.DELAY};{Zabbix.INTERVAL}",
    }
    self.api.item.create(
        hostid=itemAge['hostid'],
        name=itemAge['name'],
        key_=itemAge['key_'],
        type=itemAge['type'],
        value_type=itemAge['value_type'],
        interfaceid=itemAge['interfaceid'],
        delay=itemAge['delay'],
    )
    Zabbix.logger_zabbix.debug(f"Creation de l'item {itemAge['name']}")
    triggerAge: dict = {
        "event_name": "FileAge_"+name,
        "description": "FileAge_"+name,
        "expression": "abs(now()-
last(/"+host['host']+ "/" +itemAge['key_']+" ))>"+age,
        "recovery_mode": 1, "recovery_expression": "abs(now()-
last(/"+host['host']+ "/" +itemAge['key_']+" ))<="+age,
        "priority": Zabbix.TRIGGER_PRIORITY,

```



```
    main(name, file)

if __name__ == '__main__':
    Zapi = Zabbix("http://172.16.20.12/zabbix/",
"053b13d15a1d00c45ce4d2efc7ac3fcbfab6f9d0c136669ef6ef159602652f38") # type: ignore
    Zapi.connexion()
    grp = Zapi.GetGroup()
    from pprint import pprint
    pprint(grp)
```

## II.IV gitManager.py

```
import logs, logging
from git import Repo, Actor, Remote, RemoteProgress, Commit # type: ignore
from logging.handlers import TimedRotatingFileHandler
from os import path, getenv, environ
from dotenv import load_dotenv

class MyProgressPrinter(RemoteProgress):
    def update(self, op_code, cur_count, max_count=None, message=""):
        print(
            op_code,
            cur_count,
            max_count,
            cur_count / (max_count or 100.0),
            message or "NO MESSAGE",
        )

class GitManager:
    load_dotenv("C:\\Scripts\\ENV_GIT_Main\\.env")
    GIT_SERVER: str | None = getenv("GIT_SERVER") if not environ.get("GIT_SERVER")
else environ.get("GIT_SERVER")
    GIT_SSH_USER: str | None = getenv("GIT_SSH_USER") if not
environ.get("GIT_SSH_USER") else environ.get("GIT_SSH_USER")
    GIT_REMOTE: str | None = getenv("GIT_REMOTE") if not environ.get("GIT_REMOTE")
else environ.get("GIT_REMOTE")
    LOG_DIR: str | None = getenv("LOG_DIR")
    LOG_LEVEL: str | None = getenv("LOG_LEVEL")
    STREAM_HANDLER: str | None = getenv("STREAM_HANDLER")
    LOG_FILE: str | None = LOG_DIR+"\\.log" if LOG_LEVEL else None
    EMAIL = "null@null"
    NAME = "ConfigSaver"

    logLevel: dict = {
        "DEBUG":logging.DEBUG,
        "INFO":logging.INFO,
        "WARNING":logging.WARNING,
        "ERROR":logging.ERROR,
        "CRITICAL":logging.CRITICAL
    }

    logger_gittmanager = logging.getLogger(__name__)
    logger_gittmanager.setLevel(logLevel[LOG_LEVEL])

    formater_gitmanager =
logging.Formatter('%(asctime)s:%(levelname)s:%(name)s:ligne_%(lineno)d
-(message)s')

    file_handler_gitmanager = TimedRotatingFileHandler(
```

```

        filename=LOG_FILE, # type: ignore
        when='H',
        interval=24,
        backupCount=5,
        encoding='utf-8'
    )

    stream_handler_gitmanager = logging.StreamHandler()
    stream_handler_gitmanager.setFormatter(formatter_gitmanager)

    file_handler_gitmanager.setFormatter(formatter_gitmanager)
    file_handler_gitmanager.setLevel(logLevel[LOG_LEVEL])

    logger_gittmanager.addHandler(file_handler_gitmanager)
    logger_gittmanager.addHandler(stream_handler_gitmanager) if
STREAM_HANDLER.lower() == 'true' else ...

    def __init__(self, dir: str, originPath: str) -> None:
        self.author: Actor = Actor(GitManager.NAME, GitManager.EMAIL)
        self.committer: Actor = Actor(GitManager.NAME, GitManager.EMAIL)
        self.dir: str = dir
        self.originPath: str = GitManager.GIT_SERVER+"\\\\"+originPath+".git" if
GitManager.GIT_SERVER else originPath
        self.repo: Repo = Repo(self.dir) if path.exists(self.dir) else
Repo.init(self.dir, mkdir=True)
        self.origin: Remote | None = self.Remote() if GitManager.GIT_SERVER else None

        GitManager.logger_gittmanager.debug(f"creation Objet git {self.repo}")

    def __del__(self) -> None:
        # GitManager.logger_gittmanager.debug(f"Destruction Objet git {self.repo}")
        pass

    def __str__(self) -> str:
        return f"Repo: {self.repo}, origin: {self.origin}"

    @logs.Timer
    def Remote(self) -> Remote:
        GitManager.logger_gittmanager.info(f"Server Origin: {GitManager.GIT_SERVER}")
        try:
            remote = Repo(self.originPath) if path.exists(self.originPath) else
Repo.clone_from(self.dir, self.originPath, multi_options=["--bare"],
progress=MyProgressPrinter) if path.exists(self.dir) else Repo.init(self.originPath,
mkdir=True, bare=True) # type: ignore
        except Exception as e:
            GitManager.logger_gittmanager.exception(e)
        print(self.repo.remote("origin").exists())

```

```

        self.origin = self.repo.remote("origin") if
self.repo.remote("origin").exists() else remote.create_remote('origin',
f"{GitManager.GIT_SSH_USER}@{GitManager.GIT_REMOTE}:{self.originPath}")
        GitManager.logger_gittmanager.debug(f"git remote add origin
{GitManager.GIT_SSH_USER}@{GitManager.GIT_REMOTE}:{self.originPath}")
        self.origin.fetch()
        self.origin.pull()
        GitManager.logger_gittmanager.debug(f"git fetch origin main\ngit pull origin
main")
        return self.origin

    def Commit(self, addFile: list[str], msg: str) -> None:
        self.repo.index.add(addFile)
        self.repo.index.commit(msg, author=self.author, committer=self.committer)

    def last_commit_data(self) -> str:
        commit: Commit = self.repo.head.commit
        return f"\n\n{' '*50}\n{str(commit.hexsha)}\n\n\"{commit.summary}\" by
{commit.author.name}
({commit.author.email})\n{str(commit.authored_datetime)}\ncount: {commit.count()}
and size: {commit.size}\n{' '*50}\n"

    @property
    def Log(self):
        self.repo.git.log(p=True)

    """

    """

    @logs.Timer
    def main(self, dir: list[str], msg: str) -> None:
        self.Commit(dir, msg)
        GitManager.logger_gittmanager.info(self.last_commit_data())
        if GitManager.GIT_SERVER:
            self.origin.push() # type: ignore

```

## II.V logs.py

```
import datetime, logging, time
from logging.handlers import TimedRotatingFileHandler
from dotenv import load_dotenv
from os import getenv, path, listdir, remove

load_dotenv("C:\\Scripts\\ENV_GIT_Main\\.env")

TODAY: str = datetime.date.today().strftime("%Y-%m-%d")
LOG_DIR: str | None = getenv("LOG_DIR")
LOG_LEVEL: str | None = getenv("LOG_LEVEL")
MAX_AGE: float | None = float(getenv("MAX_AGE"))
STREAM_HANDLER: str | None = getenv("STREAM_HANDLER")
LOG_FILE: str | None = LOG_DIR+"\\.log" if LOG_LEVEL else None

logLevel: dict = {
    "DEBUG":logging.DEBUG,
    "INFO":logging.INFO,
    "WARNING":logging.WARNING,
    "ERROR":logging.ERROR,
    "CRITICAL":logging.CRITICAL
}

logger = logging.getLogger(__name__)
logger.setLevel(logLevel[LOG_LEVEL])

formatter = logging.Formatter('%(asctime)s:%(levelname)s:%(name)s:ligne_%(lineno)d -> %(message)s')

file_handler = TimedRotatingFileHandler(
    filename=LOG_FILE, # type: ignore
    when='H',
    interval=24,
    backupCount=5,
    encoding='utf-8'
)

file_handler.setFormatter(formatter)
file_handler.setLevel(logLevel[LOG_LEVEL])

stream_handler = logging.StreamHandler()
stream_handler.setFormatter(formatter)

logger.addHandler(file_handler)
logger.addHandler(stream_handler) if STREAM_HANDLER.lower() == 'true' else ...
```

```

def Timer(func):
    def wrapper(*args, **kwargs):
        msg: str = f"Debut de{func.__name__!r}"
        logger.debug(f'{"*"*10} {msg:^30} {"*"*10}')
        t1 = datetime.datetime.now()
        res = func(*args, **kwargs)
        t2 = datetime.datetime.now() - t1
        msg = f"Arret de{func.__name__!r}"
        logger.debug(f'{"*"*10} {msg:^30} {"*"*10}')
        logger.info(f'Fonction {func.__name__!r} executee en {(t2)}s')
        return res
    return wrapper

def Rotation(fileList: list):
    for fLst in fileList:
        with open(fLst+"."+TODAY, 'a') as newLog:
            logger.debug(f"Ouverture/Creation de {fLst+"."+TODAY}")
            with open(fLst, 'r') as log:
                logger.debug(f"Lecture de {fLst}")
                lines = log.readlines()
                for l in lines:
                    newLog.write(l)
            log.close()
        newLog.close()
    files = listdir(fLst)
    for file in files:
        age = time.time()-path.getctime(fLst+"\\ "+file)
        if MAX_AGE <= age and file!=".log":
            logger.debug(f"Suppression de {file}")
            remove(fLst+"\\ "+file)
    fLst.close()

```



## II.VI .env

```
# LOG
LOG_DIR = "C:\Backup\LOGS"
LOG_LEVEL = "DEBUG"
# "DEBUG"
# "INFO"
# "WARNING"
# "ERROR"
# "CRITICAL"
MAX_AGE = 604800.0
# historique max des logs
# 604800 epoch time 1 semaine
STREAM_HANDLER = "True"

# main -> Netmiko && Zabbix
DIR = "C:\Backup\GIT"
DISCONNECT = ""
PASS_BIS = ""

CISCO_IOS_USER = "Admin"
CISCO_IOS_PASS = "Soleil1"
CISCO_IOS_SECRET = "Soleil1"
ZABBIXIP = "http://172.16.20.12/zabbix/"
ZABBIXTOKEN = "053b13d15a1d00c45ce4d2efc7ac3fcbfab6f9d0c136669ef6ef159602652f38"

# Switch
FILE_MIN_SIZE = 256

# Zabbix -> pyZabbix
# DELAY = "${ITEM_DELAY};${FLEX_INTERVAL}"
DELAY = "86400"
INTERVAL = "h9"
# interval d obtention de la valeur de l item par default en second sinon preciser l
unite :: valeur mmax 86400s soit 1d (un jour)
# 30s
# 12h
# 1d
ITEM_AGE = "${CONFIG_AGE}"
ITEM_SIZE = "${ITEM_SIZE}"
# age maximum du fichier conf en sec avant de declancher un trigger
# 30 min    = 1800 utiliser pour les tests
# 1 heure   = 3600
# 1 jour    = 86400
# 1 semaine = 604800
# 1 moi     = 2629743
```

```
TRIGGER_PRIORITY = 2
# 0 = Non-classe
# 1 = Information
# 2 = Avertissement
# 3 = Moyen
# 4 = Haut
# 5 = Desastre

# GitManager -> GitPython
GIT_REMOTE = ""
GIT_SERVER = ""
GIT_SSH_USER = ""
connexion_type : ssh | telnet
device_type : aruba_os | avaya_ers | extreme_vsp | huawei | cisco_ios
lieux : Coeur_De_Reseaux | Distribution
isInactivate : True | False
isNewPass : True | False
```

## Annexe III.: Fonctionnement du programme

### III.I Exécutable

#### III.I.I Fonctionnement

Par défaut le programme récupère les informations depuis le serveur Zabbix, les informations telles que l'IP du serveur, le token de connections, les mots de passe sont enregistrés dans les variables d'environnements système.

Pour chaque switch découvert trois items/triggers (FileExist – FileAge – FileSize) sont créés s'ils n'existent pas

Les variables d'environnements à indiquer sont :

Variable	Valeur
ZABBIXIP	IP du serveur Zabbix
ZABBIXTOKEN	Token API
AVAYA_USER	Utilisateur AVAYA
AVAYA_PASS	Mot de passe AVAYA
AVAYA_PASS_BIS	Nouveau mot de passe AVAYA
HUAWEI_USER	Utilisateur HUAWEI
HUAWEI_PASS	Mot de passe HUAWEI
HUAWEI_PASS_BIS	Nouveau mot de passe HUAWEI
ARUBA_ERS_USER	Utilisateur ARUBA_ERS
ARUBA_ERS_PASS	Mot de passe ARUBA_ERS
ARUBA_ERS_PASS_BIS	Nouveau mot de passe ARUBA_ERS
EXTREME_VSP_USER	Utilisateur EXTREME_VSP
EXTREME_VSP_PASS	Mot de passe EXTREME_VSP
EXTREME_VSP_PASS_BIS	Nouveau mot de passe EXTREME_VSP
GIT_REMOTE	IP du serveur
GIT_SERVER	Chemin du répertoire du serveur
GIT_SSH_USER	Utilisateur de connexion ssh

### III.I.II Zabbix

Les hôtes zabbix doivent être membre d'un groupe {Site-Fibre\_Ville | Site-MPLS\_Adista | Site-MPLS\_Bouygues} pour être visible par le programme.

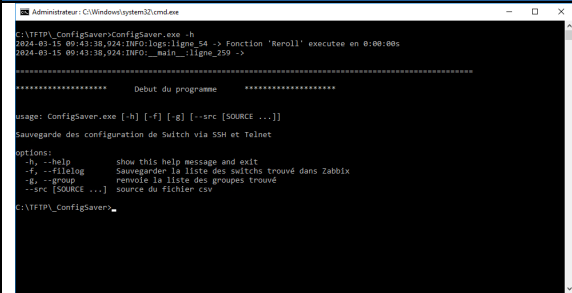

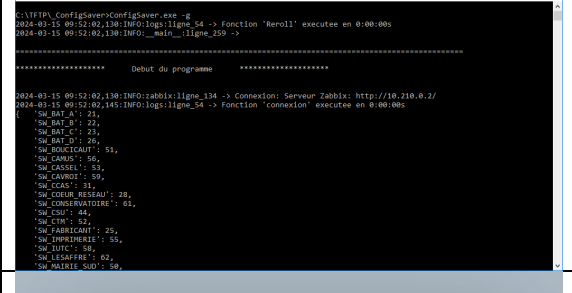

Ils doivent également contenir les tags suivants :

Tag	Valeur	Nécessité
connexion_type	ssh   telnet	Obligatoire
device_type	aruba_os   avaya_ers   extreme_vsp   huawei	Obligatoire
lieux		Obligatoire – None si inexistant
isInactivate	True   False	Facultatif – False par défaut
isNewPass	True   False	Facultatif – False par défaut

[Liste des OS supportés par Netmiko](#)

### III.I.I.I Arguments

Le lancement du programme depuis la ligne de commande permet d'activer des arguments :

Screenshots	Explications
	En ligne de commande se déplacer dans le répertoire contenant l'exécutable, entrer le nom de l'exécutable avec l'argument [-h] pour lister les arguments disponible
	L'argument [-f   --filelog] permet de sauvegarder la liste des switches trouvé dans Zabbix au format .json
	L'argument [-g   --group] permet d'afficher les groupes accessibles
	L'argument [-src] suivie d'un ou plusieurs fichier .csv, permet de lancer le programme avec uniquement les liste(s) définie(s) dans le(s) .csv  La différence ancien nouveau mot de passe n'est pas prise en charge.

### III.I.III Fichier environnement .env

Le fichier environnement doit contenir les informations suivantes :

Constante	Valeur
LOG_DIR	Chemin racine de la sauvegarde des logs
LOG_LEVEL	niveau de log à enregistrer "DEBUG" "INFO" "WARNING" "ERROR" "CRITICAL"
MAX_AGE	historique max des logs en secondes, 604800 = 1 semaine
STREAM_HANDLER	True   False affiche les logs dans la console
DIR	Chemin racine de la sauvegarde des configurations
FILE_MIN_SIZE	taille min d'un fichier de .cfg en octets
DELAY	interval entre deux requêtes de valeur d'éléments zabbix, 86400 sec = 1 jour
INTERVAL	heure à laquelle la requête sera envoyée, h10 = 10h
ITEM_AGE	âge maximum d'une sauvegarde de .cfg avant de retourner une alerte, défini dans la macro Zabbix "\${CONFIG_AGE}"
ITEM_SIZE	taille minimum d'une sauvegarde de .cfg avant de retourner une alerte, défini dans la macro Zabbix "\${ITEM_SIZE}"
TRIGGER_PRIORITY	niveau d'alerte du trigger créé par le programme 0 = Non-classe 1 = Information 2 = Avertissement 3 = Moyen 4 = Haut 5 = Désastre

Les changements effectués sur les items et triggers ne sont pris en compte que pour les éléments inexistantes.

### III.I.IV Logs

Le chemin des logs sont est indiqué dans le fichier `.env`

Différents niveaux de log peuvent être affichés

VERBOSE	NUMERIQUE	DESCRIPTION
DEBUG	10	Detailed information, typically only of interest to a developer trying to diagnose a problem.
INFO	20	Confirmation that things are working as expected.
WARNING	30	An indication that something unexpected happened, or that a problem might occur in the near future (e.g. 'disk space low'). The software is still working as expected.
ERROR	40	Due to a more serious problem, the software has not been able to perform some function.
CRITICAL	50	A serious error, indicating that the program itself may be unable to continue running.

Par défaut les logs sont enregistrés

```
>C:\\TFTP\\GIT\\LOGS
```

Si l'option `--filelog` est ajouté

```
>C:\\TFTP\\GIT\\LOGS\\SwitchList\\[hmm].ZabbixFileLogs.json
```

Pour observer les logs en direct lancer un terminal powershell et rentré la commande

```
`Get-Content .log -wait`
```

## III.II Développement

### III.II.I Require

Les dépendances nécessaires sont situés dans requirements.txt pour les installer lancer la commande dans le terminal:

```
``pip install -r requirements.txt``  
ou  
`py -m pip install -r requirements.txt`
```

### III.II.II Compilation

Pour compiler le script en exécutable le module pyinstaller doit être installé lancer un terminal et se déplacer dans le dossier du script et lancer la commande :

```
`pyinstaller --onefile <nomDuScript.py> -n <nomDeLexe.exe>`  
L'exécutable sera situé dans le dossier dist
```