Institut Universitaire de Technologie,
Aix-Marseille Université

RAPPORT DE STAGE
Diplôme Universitaire de Technologie
Spécialité Réseaux et Télécommunications

Restructuration graphique et logique d'un logiciel de supervision réseau au niveau physique

Benjamin SABY

Direction Opérationnelle des Systèmes d'Informations (Aix-Marseille Université)

Responsable entreprise : Pascal MOURET

Responsable académique : Arnaud FEVRIER

2019

Table des matières

1 Introduction

Mon stage s'est déroulé à la Direction Opérationnelle des Systèmes d'Informations (DOSI) de Aix-Marseille Université, antenne de campus de Luminy, au sein de l'équipe réseau. Mon tuteur en entreprise, Pascal MOURET, en est le responsable. Le projet de ce stage était de réaliser une restructuration graphique et logique d'un logiciel appelé GRIP (Gestion des Réseaux Informatiques au niveau Physique), développé depuis 2004 par la DOSI, et mis à disposition de tous (sous licence GPLv3) depuis 2013 (cf https://sourcesup.renater.fr/projects/grip/).
GRIP est une interface web de gestion de l'infrastructure physique d'un réseau. Elle permet de modéliser les différents éléments des locaux techniques ainsi que les liens entre eux, et d'analyser, leurs interactions.
L'interface graphique de GRIP ne permettait pas une utilisation simple et rassemblait beaucoup d'informations à l'écran, rendant sa lecture complexe.
Le but du projet était donc de créer une interface plus simple à prendre en mains, et d'adapter le code exécuté en arrière-plan.
La DOSI m'a aussi permis de travailler sur divers projets réseaux, ainsi que sur l'installation de nouveaux équipements, que nous détaillerons plus tard.

2 Présentation de l'entreprise

2.1 La DOSI

La DOSI est un service de l'université d'Aix-Marseille ayant pour mission de mettre en œuvre la politique de l'établissement en matière de systèmes d'informations. C'est-à-dire fournir aux étudiants, comme aux personnels, un service informatique opérationnel et maintenu.
Le service est structuré en pôles transverses et en équipes de proximité, les antennes de campus.
L'antenne DOSI du campus Luminy est aussi appelée DOSICALU.
Au quotidien, le service peut avoir à travailler avec de nombreuses personnes, chaque composante de Luminy est susceptible d'avoir recours aux services de la DOSI. Au cours de mon stage, j'ai pu voir certains d'entre eux, dont l'IUT Réseaux et Télécommunications, l'Institut Méditerranéen d'Océanologie (Océanomed), la Faculté des Sciences du Sport (FSS), ainsi que le laboratoire Lasers, Plasmas et Procédés Photoniques (LP3).
A Luminy, la DOSI se scinde en trois équipes :

- Equipe Réseau : chargé de maintenir une connexion constante et une qualité de service maximale afin que les utilisateurs puissent travailler sans problèmes
- Equipe Système : chargé de définir les besoins en systèmes d'informations, de mettre en place les serveurs nécessaires, ainsi que protéger et sauvegarder les données.
- Equipe Gestion Parc et Assistance (GPA) : prise en charge des matériels administratifs et pédagogiques, des logiciels, ainsi qu'assistance aux utilisateurs du parc.

Le tout dans une ambiance agréable, une communication hors pair, et un partage de connaissance constant.
Mais la DOSI c'est aussi de petits plaisirs, comme le goude, qui est le goûter du jeudi en fin de journée. Chaque semaine, une personne différente (choisie en fonction de l'ordre alphabétique) apporte à manger pour le service. Ce sont des moments toujours très agréables de partage, et de rires.

2.2 Le service réseau

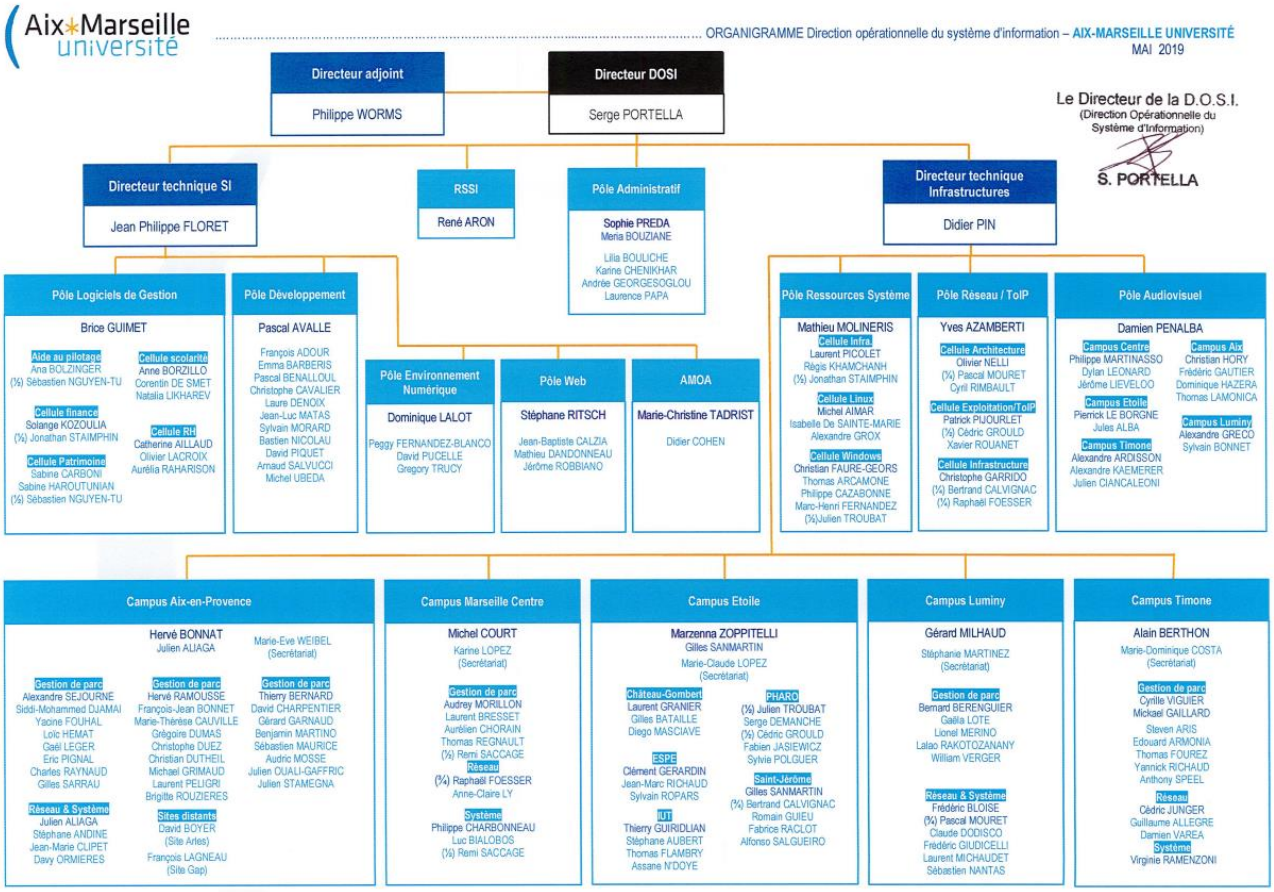

Figure 1 : Organigramme de la DOSI

Comme nous pouvons le voir sur cet organigramme, la DOSI du campus de Luminy est administrée par Monsieur Gérard MILHAUD, puis par différents responsables de pôles, comme mon tuteur en entreprise, Monsieur Pascal MOURET, qui m'a pris en charge tout au long de mon stage. J'ai aussi eu l'occasion de travailler avec Monsieur Laurent MICHAUDET, et Monsieur Frédéric GIUDICELLI, qui sont aussi deux membres de l'équipe réseau.

3 Cadre technique général du sujet

Mon stage s'est déroulé en 3 parties bien distinctes, avec des compétences requises différentes :

1. La restructuration graphique et logique de GRIP (programmation, base de données et administration système)
2. La configuration et la sécurisation de Micro-switch pour l'installation de bornes wifi sur la plaine sportive de Luminy (administration réseaux)
3. Mise en place d'une maquette de portail captif et firewall à l'aide de pfSense (administration système)

Nous allons donc voir en 3 parties les contextes relatifs à chacune de ces missions.

4 Restructuration graphique et logique de GRIP

4.1 Présentation de GRIP

GRIP est adossé à une base de données et permet de modéliser les locaux techniques, et les armoires réseau qui s'y trouvent ainsi que la disposition des éléments à l'intérieur. Elle permet également de représenter les liaisons entre les armoires ou les précâblages des locaux et de stocker les caractéristiques des liaisons fibre optique ou des liaisons cuivre, comme par exemple la connectique utilisée. On peut voir aisément le nombre de liaisons dont on dispose avec leurs caractéristiques ou le nombre et la localisation des armoires réseau. On voit les équipements réseau administrables ou non avec les ports libres, etc. En bref, tout ce qui permet d'avoir une vue d'ensemble de l'utilisation du réseau et de planifier plus aisément ses évolutions.

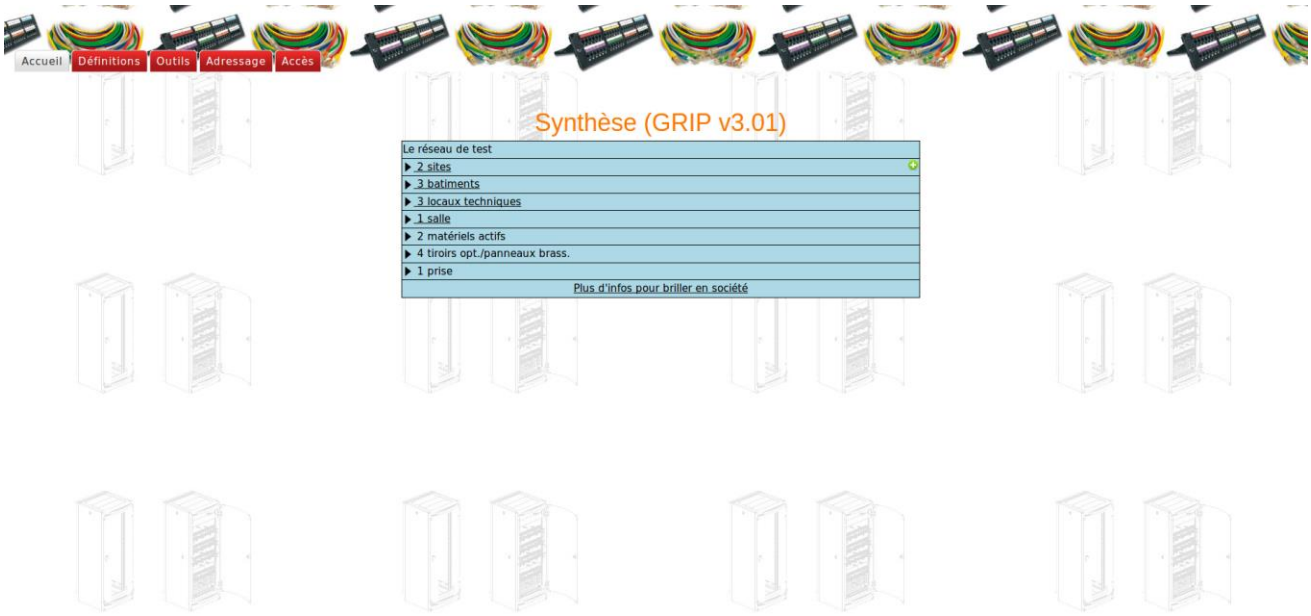

Figure 2 : Page d'accueil de GRIP

Cette page d'accueil est simple : il suffit de choisir le site sur lequel nous voulons travailler, puis le bâtiment associé au site, puis le local technique assigné au bâtiment. En cliquant simplement sur les liens affichés à l'écran, une liste déroulante apparaîtra (voir figure 4 ci-dessous).

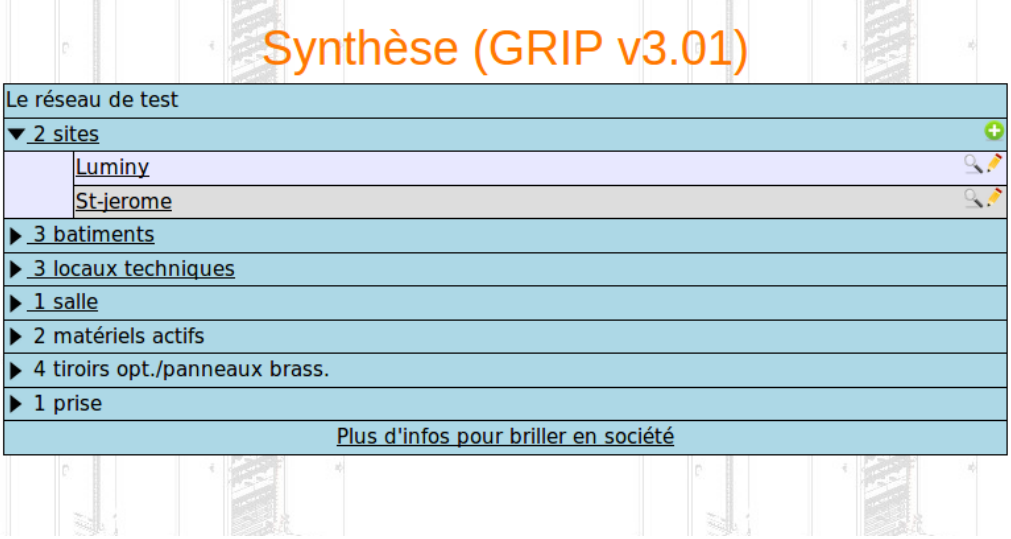

Figure 4 : Sélection du site

Une fois le site et le bâtiment sélectionné, il suffit de cliquer sur un local technique, pour être redirigé vers une page présentant une vue d'ensemble des baies présentes dans le local technique que nous avons sélectionné.

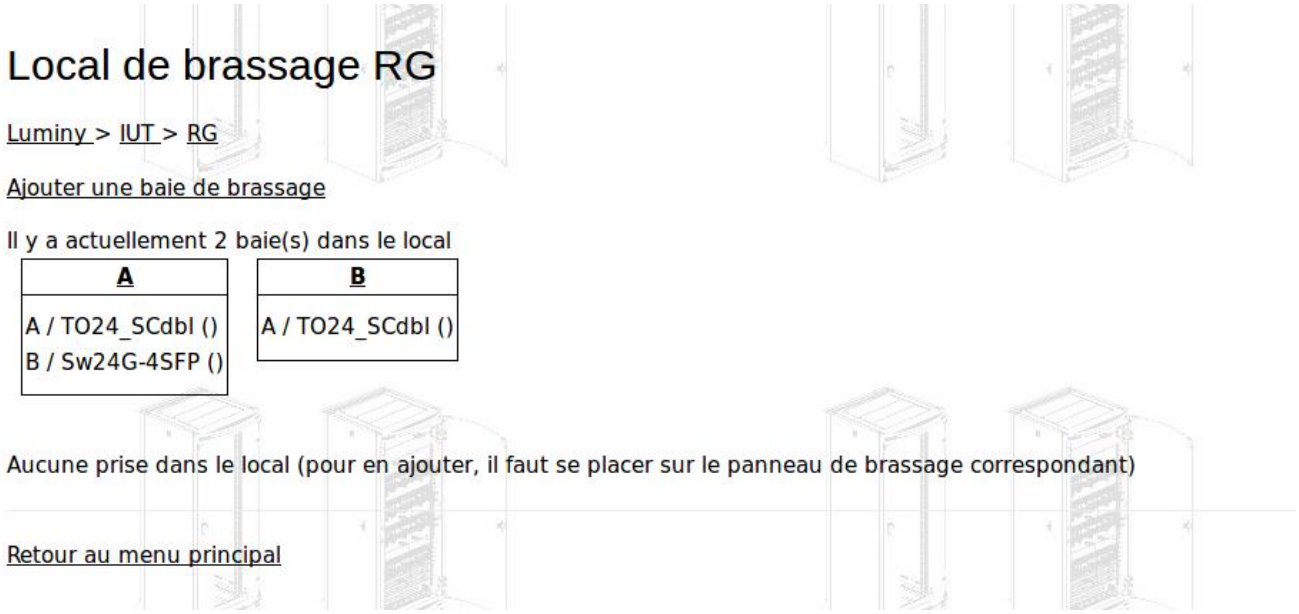

Figure 5 : Exemple de contenu d'un local technique

Sur la figure 5 ci-dessus vous pouvez voir plusieurs informations :

- Nous nous plaçons actuellement dans le local de brassage RG, dans le bâtiment IUT, appartenant au site de Luminy.
- Il existe deux baies de brassages dans ce local, nommées A et B.
- La baie A contient deux équipements réseaux, et la baie B n'en contient qu'un seul.

Une fois que nous faisons notre choix parmi l'une des baies contenues dans ce local nous sommes redirigés sur une nouvelle page, représentant les équipements réseaux de la baie.

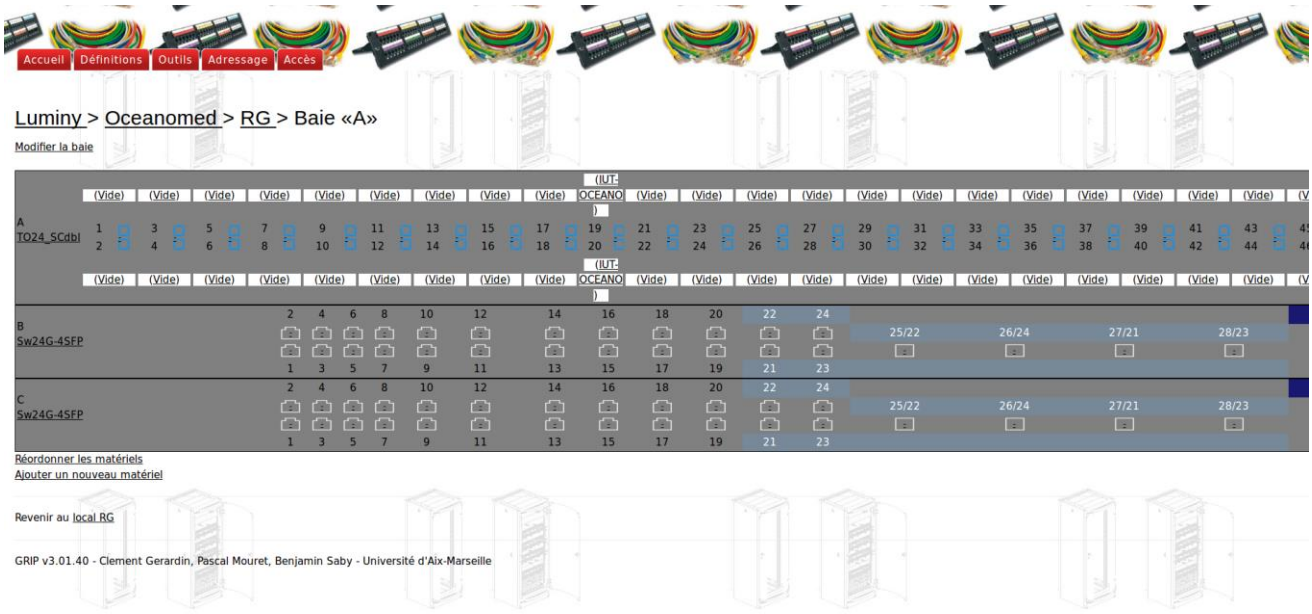

Figure 6 : Représentation des équipement réseaux de la baie A, local RG, bâtiment Oceanomed, Site de Luminy.

Sur la figure ci-dessus, vous pouvez voir trois matériels réseaux, dont un tiroir optique (panneau de brassage de fibres optiques) ainsi que deux commutateur en dessous. Pour créer un nouveau brassage, il faut cliquer sur le petit lien se trouvant à l'intérieur de la représentation du port, puis choisir à l'aide d'un menu déroulant l'équipement de destination du brassage, ainsi que son port, et enfin, valider le lien.

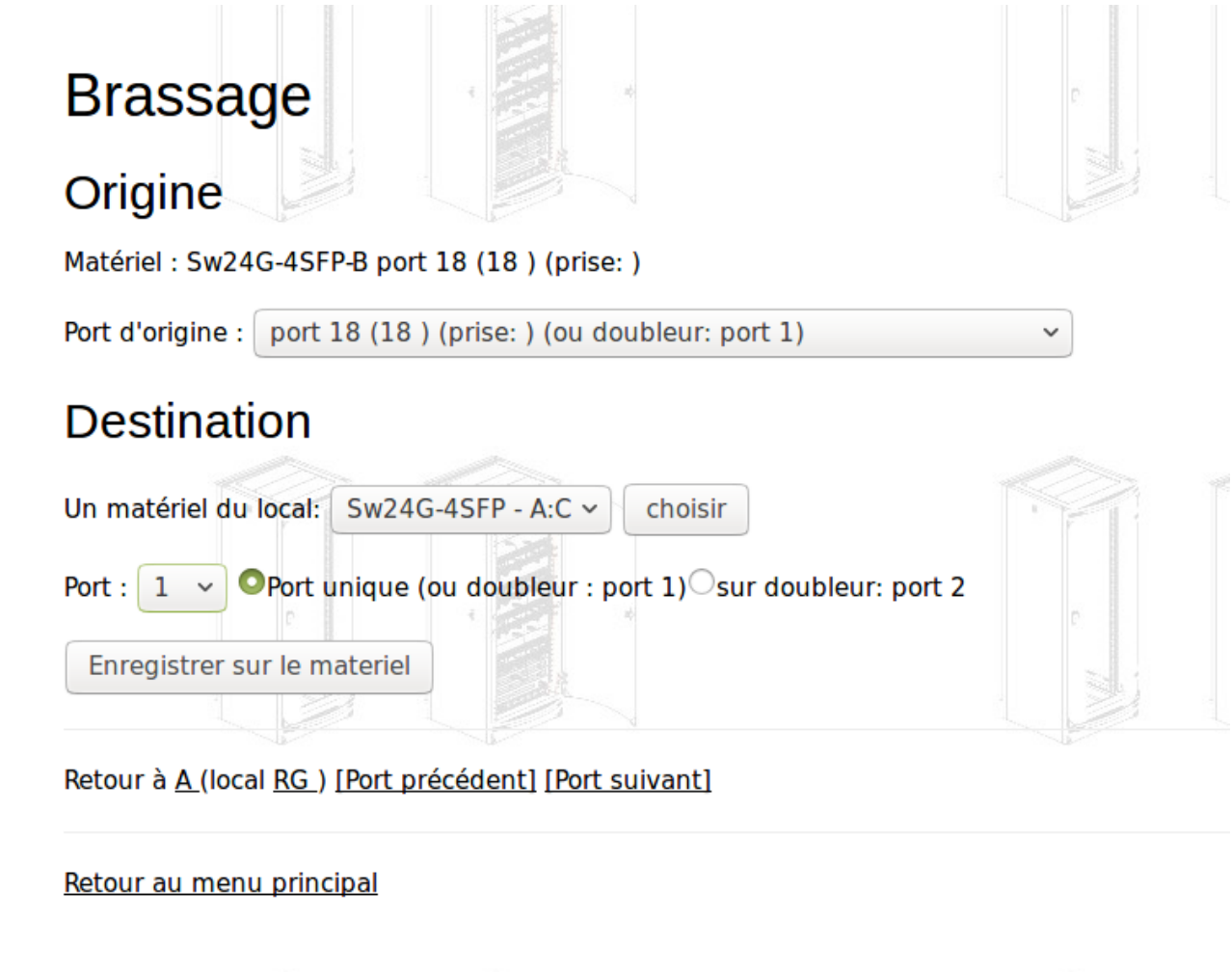

Figure 7 : Page d'ajout de brassage
En temps normal, les liaisons entre deux ports sont représentées par un petit lien se trouvant à l'intérieur du port, décrivant l'équipement de destination ainsi que son port et éventuellement son local technique (si différent du local technique actuel).

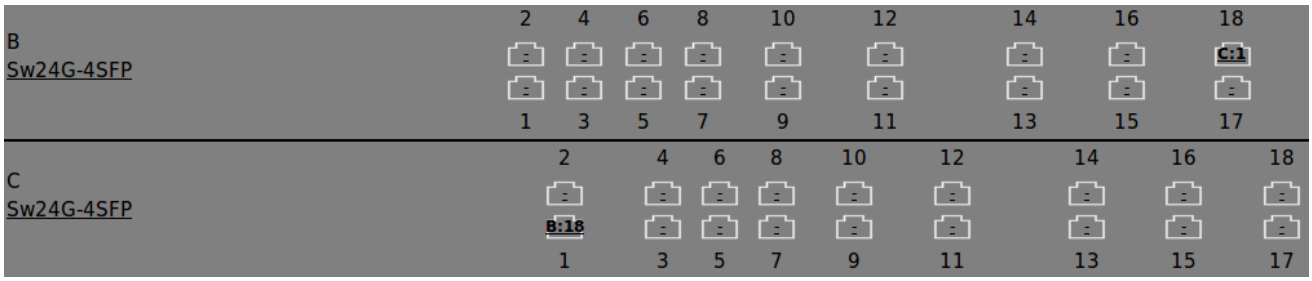

Figure 8 : Après ajout du brassage

Nous pouvons voir ci-dessus le brassage entre le port 18 du switch B, et le port 1 du switch C.

4.2 Technologies utilisées

4.2.1 Perl

Perl est un langage de programmation conçu pour traiter facilement de l'information de type textuelle. Il est inspiré des structures du langage C, mais aussi de certains langages de scripts, comme le shell. GRIP est basé sur Perl, à l'appel de la page, et il génerera le code HTML correspondant à ce qui est demandé. Au total, GRIP est composé de 106 scripts Perl.

4.2.2 Arborescence de GRIP

GRIP possède une arborescence de fichier s'organisant de cette manière :
Tout d'abord, la totalité des scripts se trouvent dans le répertoire /usr/lib/grip/cgi-bin/, alors que la partie « statique » du site (les feuilles de style CSS* et les images) se trouvent dans le répertoire /var/www/grip/html/.
Un alias est défini dans la configuration du site de grip, c'est-à-dire que lorsqu'on veut accéder au répertoire des scripts, au lieu de préciser le répertoire, on pourra appeler via un autre nom, ici, on remplace /usr/lib/grip/cgi-bin par /grip/cgi-bin.

GRIP a aussi un fichier de configuration tel que celui-ci :

```
#Nom de l'entité dont on traite le réseau

entite = "Nom qui apparait en haut"

# Definition des acces de base de donnees

# db_user : le nom de l'utilisateur qui a acces a la base de donnees
db_user='gripuser'
# db_name : le nom de la base
db_name='grip'
# db_host : l'adresse IP ou le nom de l'hote accueillant la base de donnees
db_host='127.0.0.1'
# db_password : le mot de passe de l'utilisateur <db_user>
db_passwd='mot de passe'

# Definition des url

# html_base : definit l'url de base -> http://<nom du serveur Apache><html_base>
html_base='/grip'
# html_static : definit l'url pour les contenus statiques -> http://<nom du serveur Apache><html_static>
html_static="$html_base/html"
# html_cgi : definit l'url pour les contenus dynamiques -> http://<nom du serveur Apache><html_cgi>
#            notamment l'url de depart est : http://<nom du serveur Apache><html_cgi>/index.cgi
html_cgi="$html_base/cgi-bin"

# pour debugging

debug = 0

# Interface graphique, 0=non, 1=oui

graph=1
```

Figure 9 : fichier de configuration de grip

Ce fichier de configuration nous permet notamment de définir le nom du réseau sur lequel nous travaillons, les informations de connexion à la base de données, les chemins d'accès aux différentes ressources de GRIP, et l'activation (ou désactivation) de la nouvelle interface graphique.

4.2.3 PostgreSQL

PostgreSQL est un système de gestion de base de données relationnelle et objet. Ce qui signifie que c'est un type de base de données où l'information est organisée en tableau à deux dimensions et où les informations sont stockées sous forme de collections d'objets.
La base de données PostgreSQL utilisée par GRIP compte 41 tables, notre utilisation pour la restructuration graphique et logique de GRIP va se limiter à l'utilisation d'un peu moins d'une dizaine de tables.

Actions		num	num_loc	taille	ordrenum	info	pos	nom	rem
Éditer	Effacer	0	0	*NULL*	*NULL*		1	Inconnu	*NULL*
Éditer	Effacer	1	1	42	1	800x800	0	A	
Éditer	Effacer	2	2	42	1	800x800	0	A	
Éditer	Effacer	3	1	42	1	800x800	1	B	
Éditer	Effacer	4	4	42	1	800x800	0	A	

Figure 10 : exemple de table postgreSQL

4.3 Cahier des charges

Jusqu'à aujourd'hui, peu de personnes utilisent GRIP pour gérer le brassage, dû à son interface peu pratique. Si peu de personnes se servent de GRIP, la base de données deviendra de plus en plus erronée, donc le logiciel perdra entièrement son sens. L'enjeu de cette restructuration est de rendre évident l'ajout d'un brassage, afin de faciliter l'utilisation du logiciel et ainsi que la cohérence de la base de données puisse être assurée.
GRIP permet à l'utilisateur d'exécuter plusieurs tâches :

- Modifier des baies
- Visualiser des baies, leurs équipements, et les ports de chaque équipement
- Visualiser des brassages entre des ports d'une même baie, ou des brassages avec d'autres baies
- Ajouter ou supprimer des brassages
- Naviguer entre les différentes baies
- Créer de nouvelles rocades, de nouvelles prises murales

La nouvelle interface de GRIP devra respecter tous ces critères, tout en étant plus parlant pour l'utilisateur.

5 Etude des solutions

GRIP est donc une interface web développée en Perl. Elle rassemble plusieurs scripts ayant tous une fonctionnalité propre. Mon travail s'est concentré particulièrement sur le redéveloppement du script viewRack.pl (qui permet de visualiser le contenu d'une baie). Ce script Perl exécute de multiples requêtes SQL sur la base de données PostgreSQL. Le but est de ne remplacer que la partie centrale de la page et laisser le reste en état.

5.1 Choix techniques utilisés

5.1.1 PHP

PHP* est un langage de programmation principalement utilisé pour produire des pages web dynamiques via un serveur HTTP*.
Actuellement, GRIP utilise la base de données pour produire ses pages.
Plusieurs raisons m'ont poussées à choisir le PHP plutôt que de continuer en Perl :

- Je suis bien plus à l'aise avec PHP qu'avec Perl, et PHP est fonctionnellement équivalent, voire même plus adapté pour la production des sites web
- Perl étant en perte de vitesse, nous avons pensé que c'était certainement l'occasion pour amorcer une migration de GRIP vers du PHP.

L'utilisation de ce langage nous permettra de faire appel à un script qui va générer la totalité du code nécessaire à la récupération d'informations sur la base de données, et à l'affichage de la nouvelle interface. Nous ferons donc un appel au script PHP à l'intérieur du script Perl déjà existant.

5.1.2 JavaScript

JavaScript est un langage de programmation de scripts principalement employé dans les pages web interactives. Ce langage va nous être particulièrement utiles pour plusieurs choses :

- Nous allons avoir besoin de générer un code javascript pour rendre l'affichage des baies plus simple
- La page web sera interactive, JavaScript permettant de capturer des événements sur la page tels que des clics, double clics, clics droit.

JavaScript va être une partie très importante de cette restructuration.

5.1.3 Bibliothèque JavaScript : vis js

Vis js est une bibliothèque JavaScript, c'est-à-dire un fichier JavaScript contenant une collection de fonctions pré-écrites, prêtes à être utilisées par des programmes.
Vis js a pour but de faire des représentations dynamiques de données. Elle a de multiples applications : affichage de frises chronologiques, de graphiques 2D et 3D, mais aussi représentation de réseaux de points. Elle est basée sur une représentation graphique à l'aide de la balise HTML canvas, qui est une balise permettant de dessiner des éléments graphiques à l'aide de scripts.

Vis js a une spécificité : le placement absolu (placer un nœud à un endroit précis) se fait en fonction du point (0,0) qui se trouve en réalité tout en haut à gauche du canvas. Pour l'axe des abscisses, plus on se déplace vers la droite, plus le chiffre augmente. Pour l'axe des ordonnées, c'est un peu différent, plus on va vers le bas, plus le chiffre augmente.

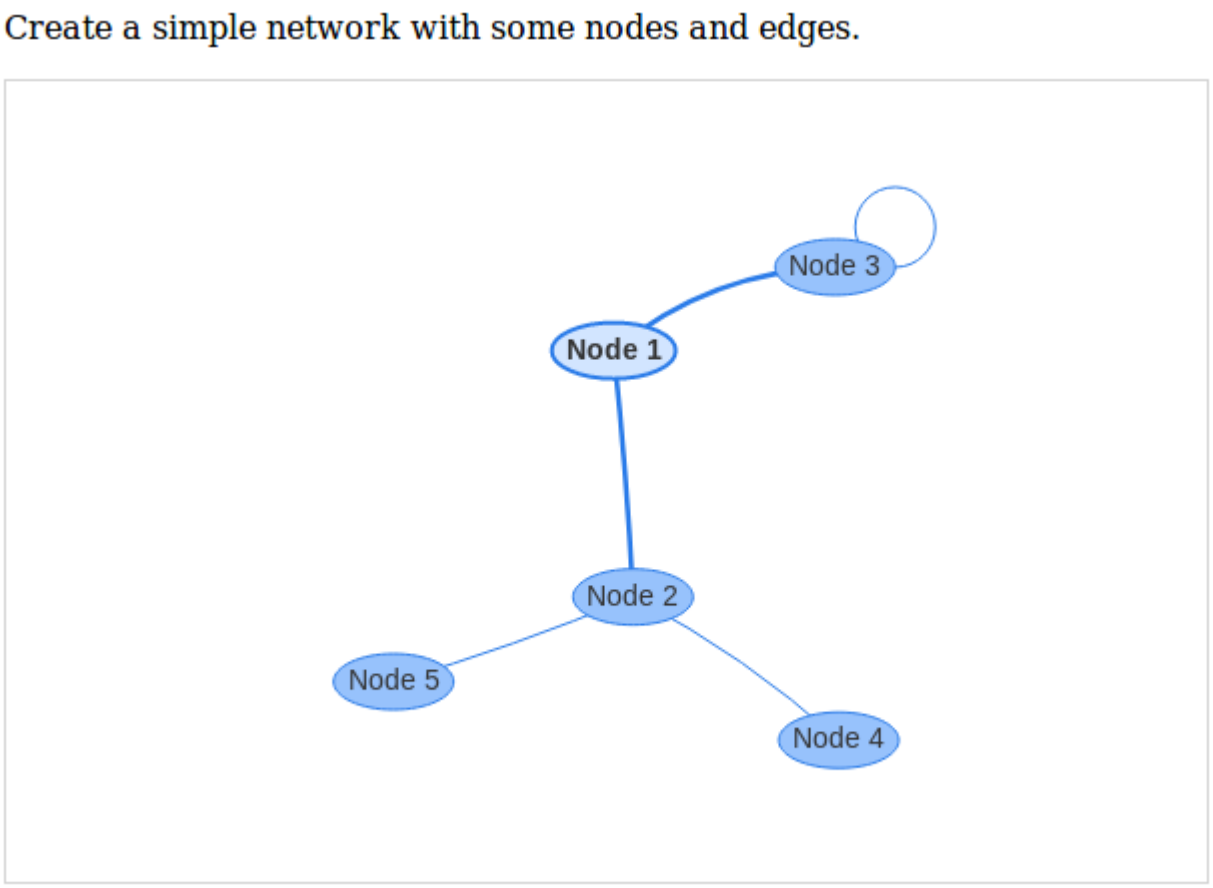

Figure 11 : réseau de points simple par la bibliothèque vis js

Notre choix s'est porté sur vis js pour plusieurs raisons :

- On a la possibilité de rajouter des images dans les nœuds (les nœuds serviront pour la représentation des ports).
- On peut empêcher les nœuds de bouger et les fixer à un endroit précis.
- On peut relier les ports facilement à l'aide de traits (edges)
- Elle possède une documentation complète, avec de multiples fonction permettant de récupérer des informations en temps réel sur le réseau de points.
- La déclaration des nœuds peut être facilement automatisée du fait de sa simplicité.

```
<script type="text/javascript">
  // create an array with nodes
  var nodes = new vis.DataSet([
    {id: 1, label: 'Node 1'},
    {id: 2, label: 'Node 2'},
    {id: 3, label: 'Node 3'},
    {id: 4, label: 'Node 4'},
    {id: 5, label: 'Node 5'}
  ]);

  // create an array with edges
  var edges = new vis.DataSet([
    {from: 1, to: 3},
    {from: 1, to: 2},
    {from: 2, to: 4},
    {from: 2, to: 5},
    {from: 3, to: 3}
  ]);

  // create a network
  var container = document.getElementById('mynetwork');
  var data = {
    nodes: nodes,
    edges: edges
  };
  var options = {};
  var network = new vis.Network(container, data, options);
</script>
```

Figure 12 : exemple de déclaration JavaScript du réseau vu précédemment

Pour fonctionner vis js a besoin d'une balise HTML afin d'y insérer le canvas. Dans le JavaScript il a besoin de trois variables :
- la référence de la balise
- les données à insérer dans le canvas (la déclaration des nœuds, et des liens)
- les options à appliquer sur le réseau (exemple : la taille, la largeur du canvas, la couleur des nœuds, l'activation de l'interaction avec les nœuds, et bien d'autres encore).

5.1.4 Architecture Ajax

Ajax* est l'architecture informatique permettant de construire des applications Web et des sites web dynamiques interactifs sur le poste client.
Dans notre cas, ajax nous permet d'exécuter un script après le chargement de la page, lorsque l'utilisateur interagit avec la page.
Cela nous permet de ne pas avoir à exécuter plusieurs requêtes SQL au chargement de la page, ce qui, à force d'en rajouter, finirait par ralentir considérablement le chargement de la page.

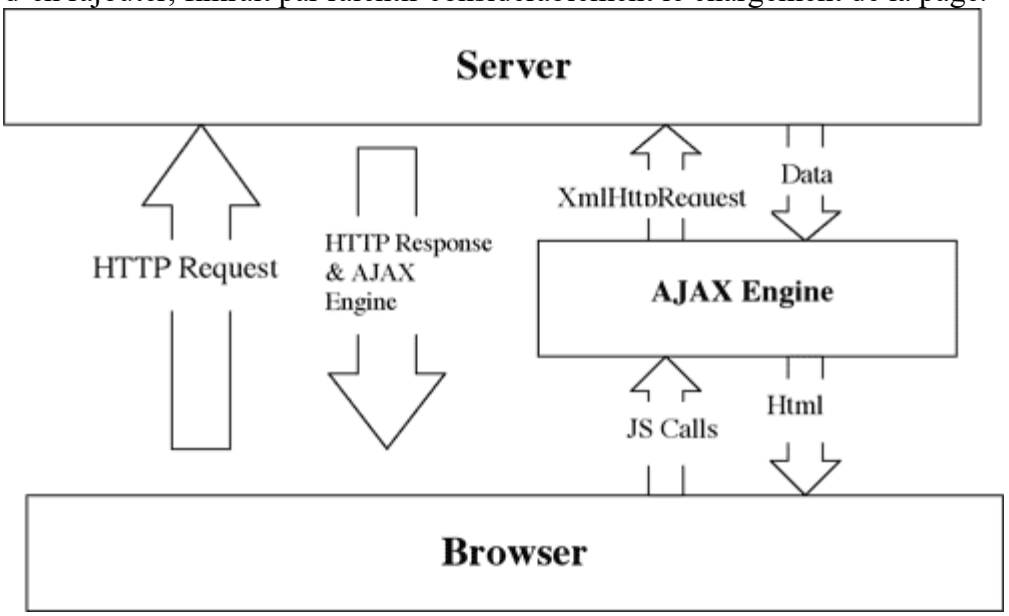

Figure 13 : exemple d'architecture AJAX

Ci-dessus nous pouvons voir un exemple représentatif de notre situation. Initialement, la page se chargera après une requête HTTP. Une fois la page chargée, nous pourrons obtenir de nouvelles informations en interagissant avec la page, qui appellera une nouvelle requête sur un autre script, et le serveur nous retournera l'html qui a été créé via ce dernier script.

6 Développement de GRIP

6.1 Elaboration du script PHP

La première étape a été d'automatiser la mise en place des nœuds (c'est-à-dire les ports), en prenant en compte plusieurs critères :
- La connexion à la base de données doit se faire à partir des identifiants présents dans le fichier de configuration de GRIP.
- Chaque nœud doit avoir un id, ils doivent se suivre et ne pas se chevaucher. Il faudra donc créer un tableau contenant la correspondance entre l'id de nœud, et le numéro de switch et numéro de port.
- Les ports dans la base de données, dépendent d'un numéro de ligne, car les ports d'un commutateur peuvent être sur plusieurs lignes, il faudra donc que l'algorithme prenne en compte cet aspect.
- A chaque fois qu'un nouvel équipement est traité par l'algorithme, on reviendra à la ligne.
- Pour un meilleur repérage pour l'utilisateur, il faudra qu'à chaque fois qu'un nouvel équipement est traité par l'algorithme, un nœud soit inséré pour y mettre le nom de l'équipement, et ses informations.
- Il y a différents types de ports, il faudra donc stocker des images correspondant à chaque type de port afin de pouvoir afficher l'image correspondante. On nommera les fichiers avec le numéro d'id de type de port stocké dans la base de donnée (exemple : le numéro 8 correspond au RJ45 donc, quand un port aura le type de port 8, grip ira chercher l'image 8.png)
- La position des ports sera absolue, il faudra donc incrémenter des variables afin de les placer convenablement.

Récupérations des fichiers de configuration GRIP :

```
    $fichiertab = file("../etc/grip.conf"); //on va chercher dans le fichier
    for($i=0; $i<sizeof($fichiertab); $i++){
        $mots = explode("=",$fichiertab[$i]);
        if($mots[0] == "db_user"){
            $DBUSER = preg_replace("/'+/","",$mots[1]); // récupération du user pour la base de données
        }
        else if($mots[0] == "db_name"){
            $DBNAME = preg_replace("/'+/","",$mots[1]); // récupération du nom de la base de données
        }
        else if($mots[0] == "db_host"){
            $DBHOST = preg_replace("/'+/","",$mots[1]); // récupération l'IP de la base de données
        }
        else if($mots[0] == "db_passwd"){
            $DBPASS = preg_replace("/'+/","",$mots[1]); // récupération du mot de passe user pour la base de données
        }
else if($mots[0] == "html_static"){ // on récupère l'endroit ou les fichiers statiques se trouvent
    $DIRECTORY = preg_replace("/'+/","",$mots[1]);
    $DIRECTORY =preg_replace("/\s+$/","",$DIRECTORY);
    $DIRECTORY = trim($DIRECTORY, '"');
    $DIRECTORY = trim($DIRECTORY, "'");
    $DIRECTORY = preg_replace('/\$html_base/',$BASE,$DIRECTORY);
}
else if($mots[0] == "html_base"){ // on récupère la racine de l'arborescence
            $BASE = preg_replace("/'+/","",$mots[1]);
            $BASE =preg_replace("/\s+$/","",$BASE);
            $BASE = trim($BASE, '"');
            $BASE = trim($BASE, "'");
        }

    }
```

Figure 14 : Code de récupération des valeurs du fichier de configuration

Pour l'algorithme de placement des ports, le script prend en paramètre une valeur, qui se trouve être le numéro de baie, afin de pouvoir faire une reqûete SQL uniquement sur les équipements de la baie. Pour résumer cet algorithme :
On traite à la chaine chacun des ports retournés par la requête SQL, on les affiche en ligne en fonction de leur équipement, ou de leur numéro de ligne. On indique les ports déjà brassé en les écrivant en rouge, ou en vert s'ils ne sont pas brassés. Chacun des nœuds a une image qui lui est attribuée en fonction de son type de port. On créé aussi un tableau de correspondance entre le numéro d'id du nœud, et le numéro d'équipement et de port. (exemple : correspondance[numéronoeud] = numéroéquipement,numéroport).Les problèmes que j'ai pu rencontrer sont surtout d'ordre algorithmique. Particulièrement l'attribution de l'id de nœud qui m'a paru complexe. J'utilisais initialement la variable $i (la variable d'incrémentation de la boucle for) pour attribuer la valeur d'id au nœud. Cependant, avec les nœuds qui contiennent les informations d'équipements, la tâche est devenue plus complexe puisque c'est un nœud en plus de chacune des lignes retournées par la requête SQL qui se rajoute. Donc le nombre de nœuds étant supérieur au nombre de ligne de la requête SQL, j'ai différencié la variable d'incrémentation de la boucle for, et la variable d'incrémentation d'id des nœuds. Cette variable s'appelle $z.

Figure 15 : Exemple d'affichage de ports avec 3 équipements

Dans certains locaux, il y a plusieurs baies et il peut être nécessaire de réaliser un brassage depuis un équipement d'une baie, vers un équipement de l'autre. Pour traiter ce cas, j'ai donc rajouté un script qui permet de choisir une deuxième baie et d'afficher les deux baies dans le même canvas avec deux couleurs différentes :

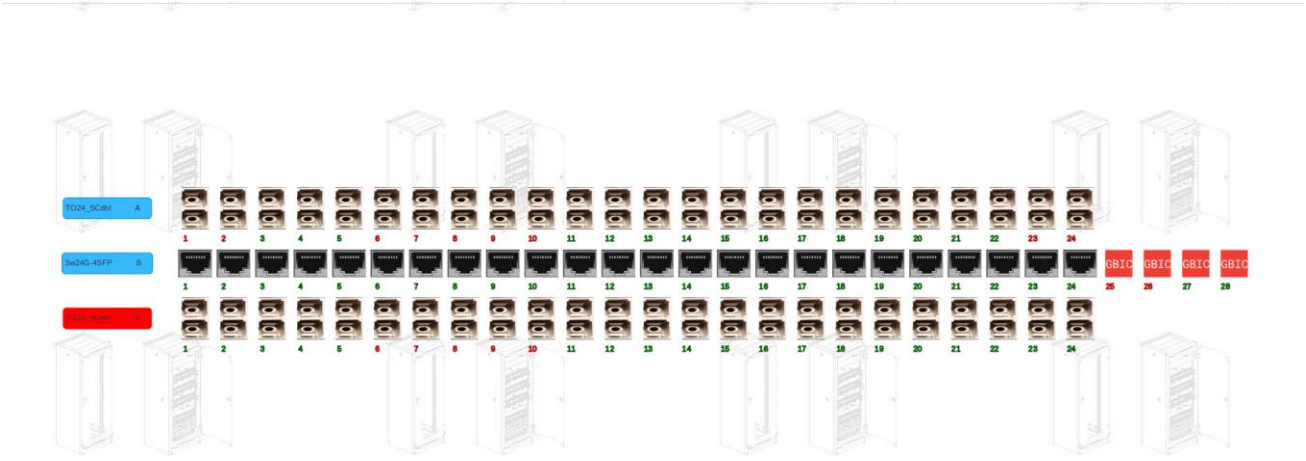

Figure 16 : brassage inter-baie avec en bleu la première baie, et en rouge la deuxième.

L'exécution du script se déroule exactement de la même façon que pour celui avec une seule baie. A la seule différence que lorsqu'on change de baie, le nœud d'informations des équipements prend la couleur rouge.

Afin de répondre aux besoin exprimés, nous avons travaillé sur l'affichage d'une représentation complète du câblage présent dans la baie, en utilisant le numéro des équipements dans la baie, qui ont été préalablement stockés dans un tableau « switcheslocaux ». J'ai donc, à la suite du script précèdent, fait une deuxième requête SQL dans une boucle, afin de récupérer les câbles présents dans

la baie, puis j'en ai fait un tableau de liens (edges pour vis js) appelé schemacomplet. La table câble est constitué de 4 éléments principaux :

- Le numéro de matériel source et le numéro de matériel destination
- Le numéro de port source et le numéro de port destination

Il faut aussi savoir que dans certains cas il existe des doubles brassages (c'est-à-dire deux câbles logiques sur un même port physique) cela arrive lorsque l'on doit raccorder un port de fibre optique duplex avec deux ports fibre optiques simplex. Lorsque c'est le cas, la base de données considère que le premier port est positif et le deuxième port négatif. Exemple : il y a un double câble sur le port 45. Dans la base de données il existera deux entrées avec comme port source 45 pour la première entrée, et pour la deuxième entrée -45 en port source. Il faut donc prendre en compte ce système dans la recherche.

Pour afficher le câblage complet de la baie, il suffit d'appuyer sur le bouton « Afficher câblage » en haut à gauche de la représentation de la baie.

Figure 17 : Exemple d'affichage du schéma de câblage complet d'une baie

```
echo "var sccompe$nbrswitch = sizeof($i<$nbwitch; $i++){
            2 ec);
                for($y=0 $y<$nbrrow; $y++){


                    if(sult["num_pr2"{ /numoeif  nt2 = sbsult["nm); //on enle le
- {
                    $numresult["nu_o
                     = aslt]."," .$num_port1, $corrdance); // on cherche le numéro
de nœud correspondant à cet équipement, et ce numéro de port
                    $indexdestinto lt.$in $indexdestiati //si les exorrectes, on
éct
                } "]
```

6.2 Elaboration du JavaScript

Une partie du PHP génère un code JavaScript statique, en particulier certaines fonctions importantes :

- La fonction network.ondoubleclick : elle permet de capturer le moment où l'utilisateur fait un double clic sur un endroit du canvas. Elle permet de récupérer les informations du neoud afin de les envoyer à la fonction postNode(). Si le nœud cliqué est une information de switch, cela redirige l'utilisateur sur la page d'information d'équipement.
- La fonction network.onSelectEdge : elle permet de capturer un clic sur un lien entre deux ports. Elle récupère les informations des nœuds connectés pour les envoyer à la fonction ajaxDelete().
- La fonction network.onContext : elle permet de capturer un clic droit sur le canvas. Elle récupère les informations pour l'endroit où l'utilisateur a cliqué, pour le transmettre à la fonction getNearestNode() qui va retourner le numéro du nœud le plus proche. On envoi ensuite le numéro de port et de matériel à la fonction ajaxNodesRocades() afin de récupérer les informations sur les éventuelles prises, et rocades.
- La fonction calcul() : elle permet de récupérer les brassages qui ont été entrés par l'utilisateur et les envoyer de manière asynchrone à un script qui va ajouter le brassage dans la base de données. Elle parcourt la totalité des nœuds et grâce à une fonction de la bibliothèque, regarde

à quoi ils sont connectés. Cette fonction est appelée lors de l'appui sur le bouton « Valider le brassage ».

- La fonction reset() : remet l'affichage par défaut des nœuds, comme au chargement de la page.
- La fonction postAjax() : Envoi de manière asynchrone les numéros de matériels, et numéros de ports, des nœuds qui doivent être brassés. Le script sollicité ajoutera le brassage à la base de données.
- La fonction postNode() : prend pour paramètre un nœuds, et envoi en asynchrone le numéro de matériel et de port afin d'obtenir les câblages qui correspondent à ce port. Le script sollicité renverra les informations sur les brassages correspondant à ce port.
- La fonction ajaxDelete() : prend en paramètres, les numéros de matériels et les numéros de ports des deux extrémités d'un lien. Envoi en asynchrone des informations, et suppression du lien dans la base de données.
- La fonction ajaxNodesRocades() : Récupère les informations du nœud qui a été cliqué avec le clic droit, puis envoie de manière asynchrone une requête à un script qui va parcourir la table des rocades, ainsi que la table des prises. Le script sollicité renverra soit les informations de la rocade, soit les informations de la prise, soit renverra une information pour indiquer qu'il n'y a ni prise ni rocade, et que par conséquent, on peut en créer. Cette fonction renvoi aussi la baie de destination de la rocade si elle existe. Si on décide de se rendre sur la baie de destination, le port d'arrivée de la rocade sera surligné en bleu.
  Exemple :

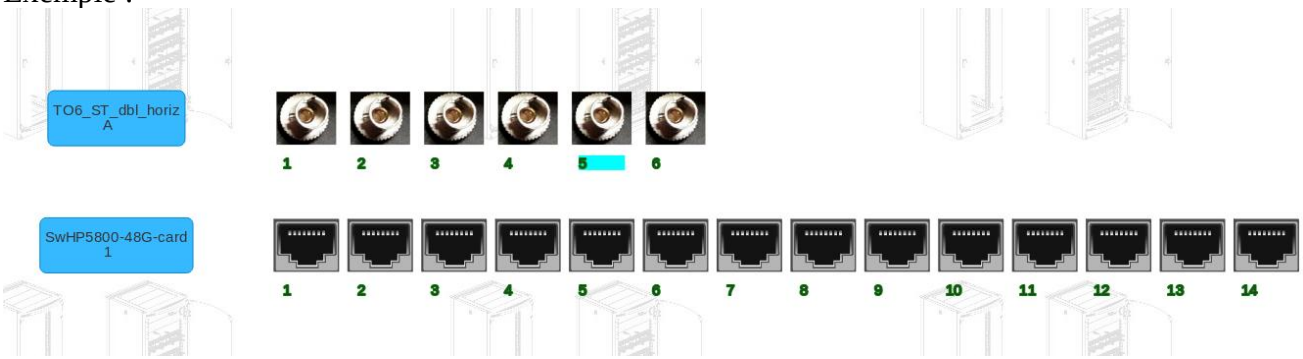

Figure 18 : exemple de baie de destination d'une rocade, avec port surligné en bleu.

6.3 Résultats finaux

Figure 19 : nouvelle interface de GRIP

La nouvelle interface graphique de GRIP respecte le cahier des charges qu'on lui avait imposé :

- Modifier des baies
- Visualiser des baies, leurs équipements, et les ports de chaque équipement
- Visualiser des brassages entre des ports d'une même baie, ou des brassages avec d'autres baies
- Ajouter ou supprimer des brassages
- Naviguer entre les différentes baies
- Créer de nouvelles rocades, de nouvelles prises murales

Cette version a été mise sur le serveur de production utilisé par les employés de la DOSI de Luminy, le logiciel est donc fonctionnel, et a été mis à disposition de tous.

6.4 Fonctionnalités de la page

Nous allons maintenant détailler les différentes fonctionnalités associées aux boutons de la barre du haut.

Bouton « Ajouter un câble » :

- Ce bouton permet de créer un nouveau lien entre deux ports. Il suffit de placer la souris sur un port, puis de drag and drop le nouveau lien sur un autre port. Remarque : l'utilisateur doit ré appuyer sur ce bouton à chaque fois qu'il veut créer un nouveau lien.

Bouton « Réinitialiser » :

- Ce bouton permet de réinitialiser les liaisons sur la page, mais pas sur la base de données. Cela permet de corriger les erreurs de sélection de port lors de l'ajout d'un nouveau câble. Mais aussi d'arrêter l'affichage de l'ensemble des câbles.

Bouton « Ajouter le brassage » :

- Ce bouton permet d'ajouter les brassages que vous avez ajoutés (à l'aide du bouton "ajouter un câble"), à la base de données. Remarque : L'utilisateur ne peut pas ajouter plus de 2 brassages sur un port (à cause des fibres doubles). Ce bouton n'est pas disponible lorsque la totalité des brassages de la baie sont affichés (Bouton "afficher câblage" activé).

Bouton « Afficher câblage » :

- Ce bouton permet d'afficher la totalité des brassages présents dans la baie. Sa principale utilité est de pouvoir visionner les ports libres, ainsi que de visionner, voire même supprimer des liens. Remarque : le câblage est récupéré au chargement de la page. Si l'utilisateur a ajouté un brassage, il faudra qu'il recharge la page pour le voir s'afficher avec le bouton "afficher câblage".

Bouton « Revenir à la sélection » :

- Permet de revenir à la sélection de la baie que l'utilisateur souhaite visualiser.

Bouton « Brassage inter-baie » :

- Permet d'être redirigé sur la sélection d'une deuxième baie, afin de procéder à un brassage entre deux baies d'un même local.

Cependant, l'utilisation de grip ne se limite pas à ces boutons :

- Le double clic sur un port affichera les connexions correspondantes à ce port. Pour enlever l'affichage de ces liens plusieurs options s'offrent à l'utilisateur. Soit double cliquer sur un autre port, soit double cliquer dans le vide, ou bien appuyer sur le bouton "réinitialiser".
- Le double clic sur un nom d'équipement (carré bleu de début de ligne), vous redirigera vers la page d'information du matériel concerné.
- La sélection d'un lien fait apparaître un pop-up avec un choix pour détruire le lien sélectionné. Ce changement sera fait sur la page mais aussi dans la base de données. Si le lien n'existe pas dans la base, la page web renverra un message pour informer l'utilisateur.
- Un clic droit sur un port affichera un menu avec un ou deux choix :

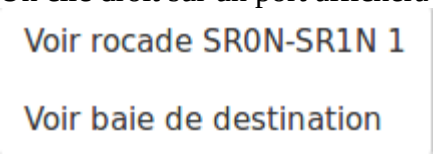

Figure 20 : menu clic droit pour rocades

Si le port a une rocade associée, il proposera ces deux solutions. Pour aller sur la modification de la rocade, il suffit de cliquer sur "Voir rocade XXXX". On peut aussi aller sur la page de visualisation de la baie de destination de la rocade en cliquant sur "voir baie de destination".

Figure 21 : menu clic droit pour les prises

Si le port a une prise associée, cliquer sur "Voir prise XXX (local XXX)" vous redirigera vers la page du local contenant la prise.

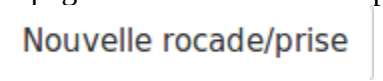

Figure 22 : menu click droit pour paramétrer une nouvelle prise ou une nouvelle rocade

Enfin, si le port n'a ni prise, ni rocade associée, en cliquant sur "Nouvelle rocade/prise", vous pourrez créer une prise, ou une rocade sur ce port.

6.5 Problèmes rencontrés

Les problèmes que j'ai pu rencontrer au cours du développement de ce logiciel étaient essentiellement algorithmiques :

- La mise en place des ports a été complexe à programmer puisqu'il fallait prendre en compte une certaine quantité de paramètres et avancer pas à pas jusqu'à obtenir une version fonctionnelle.
- Certaines requêtes SQL ont pris un certain temps de réflexion pour arriver à obtenir une requête optimisée pour avoir le plus d'informations en évitant de rallonger le temps de chargement de la page.

7 Configuration d'un mini commutateur MICROSENS

7.1 Contexte de la configuration

Depuis fin 2017, la faculté des sciences du sport a engagé un projet de connecter la plaine sportive de Luminy.
Le but est de la munir de bornes wifi pour que les arbitres de parties puissent directement, à l'aide d'une application, arbitrer les matchs. Ou même munir les joueurs de plastrons wifi afin de pouvoir mesurer certaines valeurs, et l'envoyer directement via le wifi. Il faudrait donc placer 1 borne wifi sur chacun des poteaux de la plaine sportive, donc 6 bornes wifi.

Initialement, une société avait installé plusieurs RJ45 (câbles réseau) jusqu'aux poteaux. Cependant, la longueur des RJ45 était trop importante, nous perdions donc le signal. La solution est donc d'installer des mini commutateurs installés dans le poteau, et reliés en fibre optique afin de mieux répartir l'utilisation des câbles.
Le but de la configuration de ce mini commutateur était de lui appliquer une configuration à partir de 0, et appliquer des ACL* afin de le sécuriser.

Figure 23 : Mini commutateur microsens dans un boitier de protection

7.2 Présentation et configuration

Le Ruggedized Micro Switch G6 est un commutateur disposant de 4 ports GigaEthernet, 1 port série, et 2 ports SFP*, les ports Ethernet supportent le PoE* et le PoE+ (Alimentation par câble Ethernet).
Le but de la configuration est de :

- Configurer les VLAN*, un pour l'administration du commutateur, et l'autre pour le wifi.
- Configurer les ports pour faire remonter les VLAN depuis le réseau jusqu'au commutateur, et configurer les ports terminaux des bornes wifi
- Configurer les ACL afin d'empêcher l'accès à une personne tierce
- Configurer le PoE pour alimenter la borne wifi

La borne wifi récupère sa configuration auprès d'un serveur lorsqu'elle se connecte au réseau.

Le mini commutateur peut se configurer de deux manières :

- A l'aide d'une interface web
- A l'aide de l'interface série.

La configuration de cet équipement par l'interface série marche d'une façon qui m'était étrangère jusqu'ici.
Chaque commande dépend d'objets, dépendants d'autres objets. Je m'explique :
Pour désactiver le DHCP nous devons entrer la commande :
>>Device.IP.v4_config.dhcp_mode = DISABLED

Donc on voit ici que le premier objet est l'objet Device. Un des objets dépendant de Device est IP. Chacun de ces objets permettront la configuration de l'appareil.
Pour notre configuration, disons que le VLAN 100 est le VLAN de management, et le vlan 200 le VLAN wifi. Un lien en trunk (lien sur lequel plusieurs VLAN ont le droit de passer) arrive sur le port numéro 6 du micro-commutateur à l'aide d'une fibre optique. La borne wifi est reliée grâce à un câble RJ45 sur le port numéro 1.

On définit l'adresse IP du switch et sa gateway :
>>Device.IP.v4_config.static_device_ip = 10.1.1.1
>>Device.IP.v4_config.static_gateway = 10.1.1.250

On crée les VLANS :
>> Protocol.VLAN.enable_vlan_filtering = Disabled
>> Protocol.VLAN.filter_config[*].vlan_id = 100
>> Protocol.VLAN.filter_config[*].vlan_id = 200

On les paramètre :
>> Protocol.VLAN.filter_config[100].alias = admin_dosi
>>Protocol.VLAN.filter_config[100].management_members = ALL
>>Protocol.VLAN.filter_config[100].entry_mode = Enabled
>> Protocol.VLAN.filter_config[200].alias = wifi_luminy
>>Protocol.VLAN.filter_config[200].management_members = ALL
>>Protocol.VLAN.filter_config[200].entry_mode = Enabled

On affecte les VLAN sur les ports qui correspondent à notre topologie :
>> Protocol.VLAN.filter_config[100].port_members= 1/6
>> Protocol.VLAN.filter_config[200].port_members= 1/1, 1/6

Ci-dessus nous avons défini que le VLAN 100 (VLAN de management) allait être présent sur le lien du port 1/6. Le VLAN 200, lui, sera présent sur le port 1/1 et le port 1/6.

Et on enlève les port du VLAN 1 (si non utilisé)
>> Protocol.VLAN.filter_config[1].port_members=
>> Protocol.VLAN.enable_vlan_filtering = ENABLED

On définit le vlan de management :
>> Protocol.VLAN.vlan_id_config.management_vlan_id = 100

On définit les link-type de chaque port et leur VLAN par défaut :
>>Protocol.VLAN.port_config[1/1].vlan_mode = ACCESS
>>Protocol.VLAN.port_config[1/1].default_vlan_id = 200
>>Protocol.VLAN.port_config[1/6].vlan_mode = TRUNK
>>Protocol.VLAN.port_config[1/6].default_vlan_id = 100

Ainsi, si le commutateur voisin est correctement configuré, le micro switch aura la possibilité de se connecter au réseau.

Pour la configuration des ACL, c'est un peu différent de la façon que j'ai pu voir en cours :
- Il faut créer l'ACL avec son nom : Management.ACL.list[*].name = acl2000
- Créer une règle à lui appliquer (voire plusieurs) :
  >>Management.ACL.rules[*].name = une
  >>Management.ACL.rules[une].mode = PERMIT

>>Management.ACL.rules[une].vlan_id = 100
>>Management.ACL.rules[une].source_ip = 192.168.1.0
>>Management.ACL.rules[une].source_mask = 255.255.255.0
>>Management.ACL.rules[une].destination_mask = 0.0.0.0

- Puis on applique les règles sur l'ACL :
  >>Management.ACL.list[acl2000].rules = une
  >>Management.ACL.enable_acl_filtering = enabled
- Puis on l'applique sur l'interface 1/6 :
  >>Management.ACL.active_filter_port_config[1/6].enable_acl_filtering = enabled

L'ACL est donc appliquée sur tout le trafic entrant du port 1/6.

7.3 Résultats

J'ai pu configurer 3 mini commutateurs qui doivent être installés dans les poteaux de la plaine sportive. Actuellement, il est prévu d'en placer un avec une borne wifi afin de tester l'efficacité de la couverture wifi. Nous avons pu essayer de voir si la maquette que j'ai mise en place était fonctionnelle sur place, et nous n'avons eu aucun problème. La configuration de ce commutateur était intéressante car elle m'a permis de voir un autre type d'interface de configuration.

8 Portail captif pfSense

8.1 Présentation

pfSense est un routeur et/ou pare-feu open source basé sur le système d'exploitation FreeBSD (un système d'exploitation Linux). Il convient en général pour la sécurisation d'un réseau domestique ou de petite entreprise.

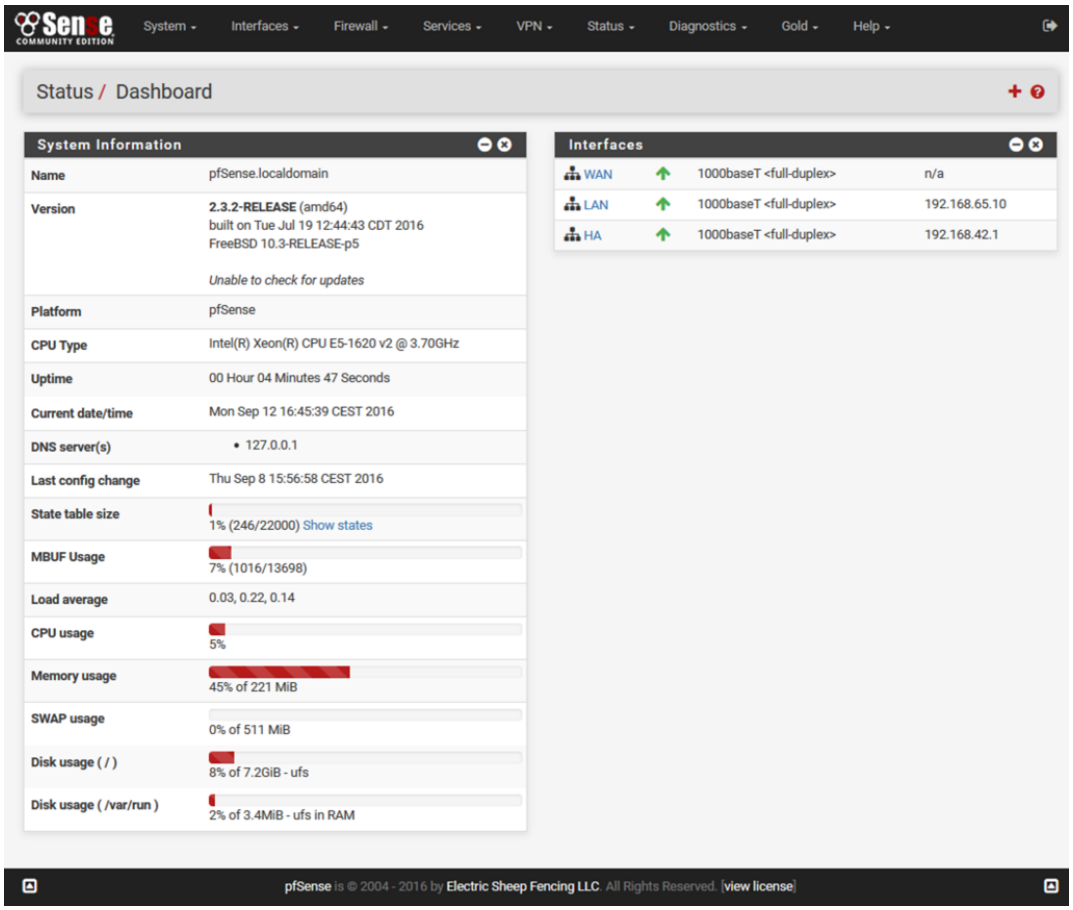

Figure 24 : Interface Web de pfSense

Nous aimerions utiliser pfSense pour voir les capacités de son portail captif. Mais commençons par définir ce qu'est un portail captif. C'est une technique consistant à forcer les clients voulant se connecter au réseau à passer par une page spécifique d'authentification avant d'accéder à internet normalement.
Le but est de faire une maquette de ce qui pourrait éventuellement remplacer le portail captif actuel de l'université d'Aix-Marseille.

8.2 Réalisation

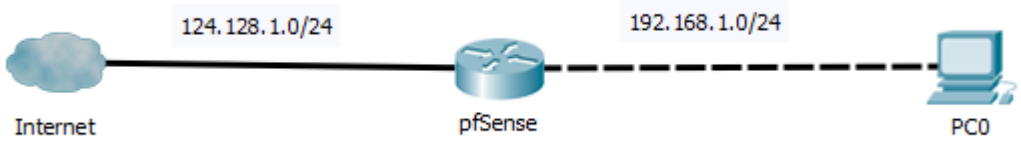

Figure 25 : Schéma de la maquette mise en place

Le but est de faire en sorte que les utilisateurs du réseau de droite (192.168.1.0/24) ne puissent pas se connecter à Internet tant qu'ils n'ont pas été authentifiés sur un serveur auparavant.
Pour cela, nous devons d'abord paramétrer certaines choses :

- Les adresses IP et les zones
  Considérons le réseau 192.168.1.0/24 comme la zone LAN*, et le réseau 124.128.1.0/24 comme la zone WAN*. Les adresses IP du pfSense seront : 192.168.1.1 et 124.128.1.1
- Définir les caractéristiques du serveur d'authentification
- Et enfin, configurer le portail captif pour que la connexion des utilisateurs soit le plus facilitée possible.

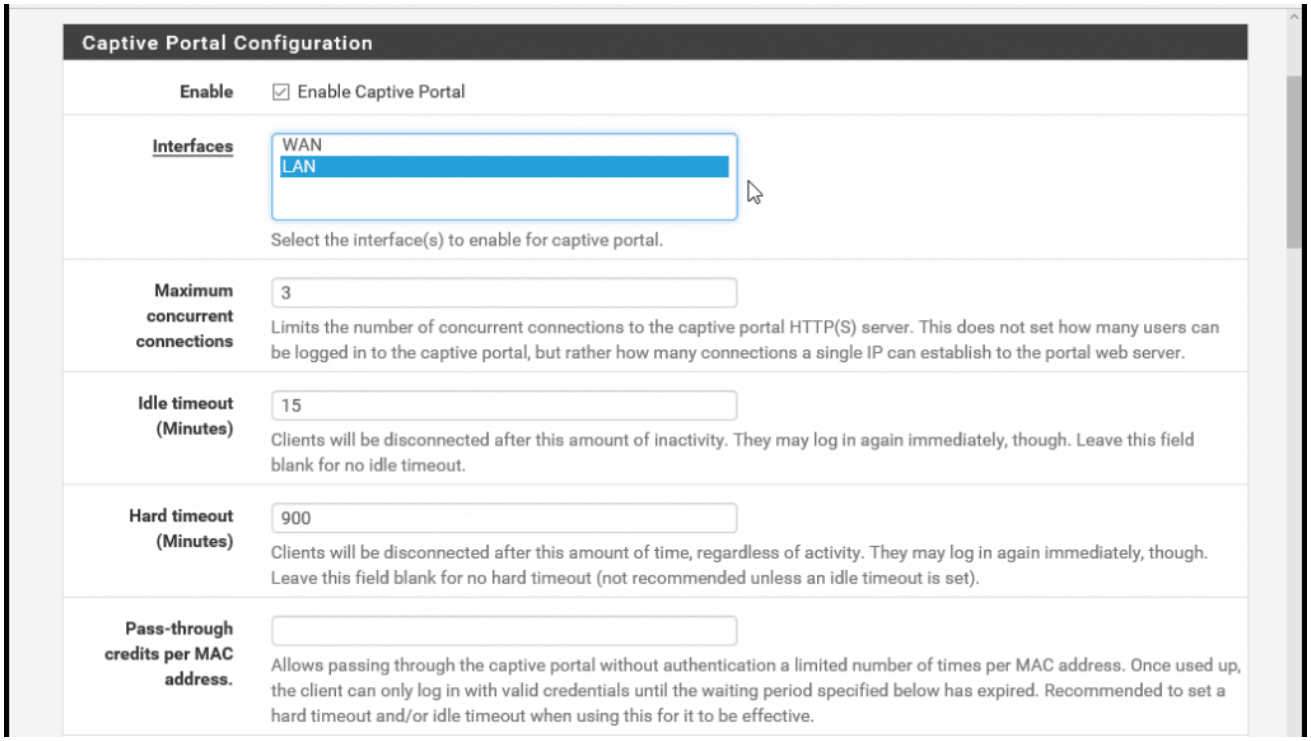

Figure 26 : configuration du portail captif

Nous devons choisir l'interface sur laquelle le portail captif sera actif. Ici, nous prendrons LAN.
Nous pouvons aussi choisir le temps au bout duquel l'utilisateur sera déconnecté, ou le temps d'inactivité au bout duquel il sera deconnecté.

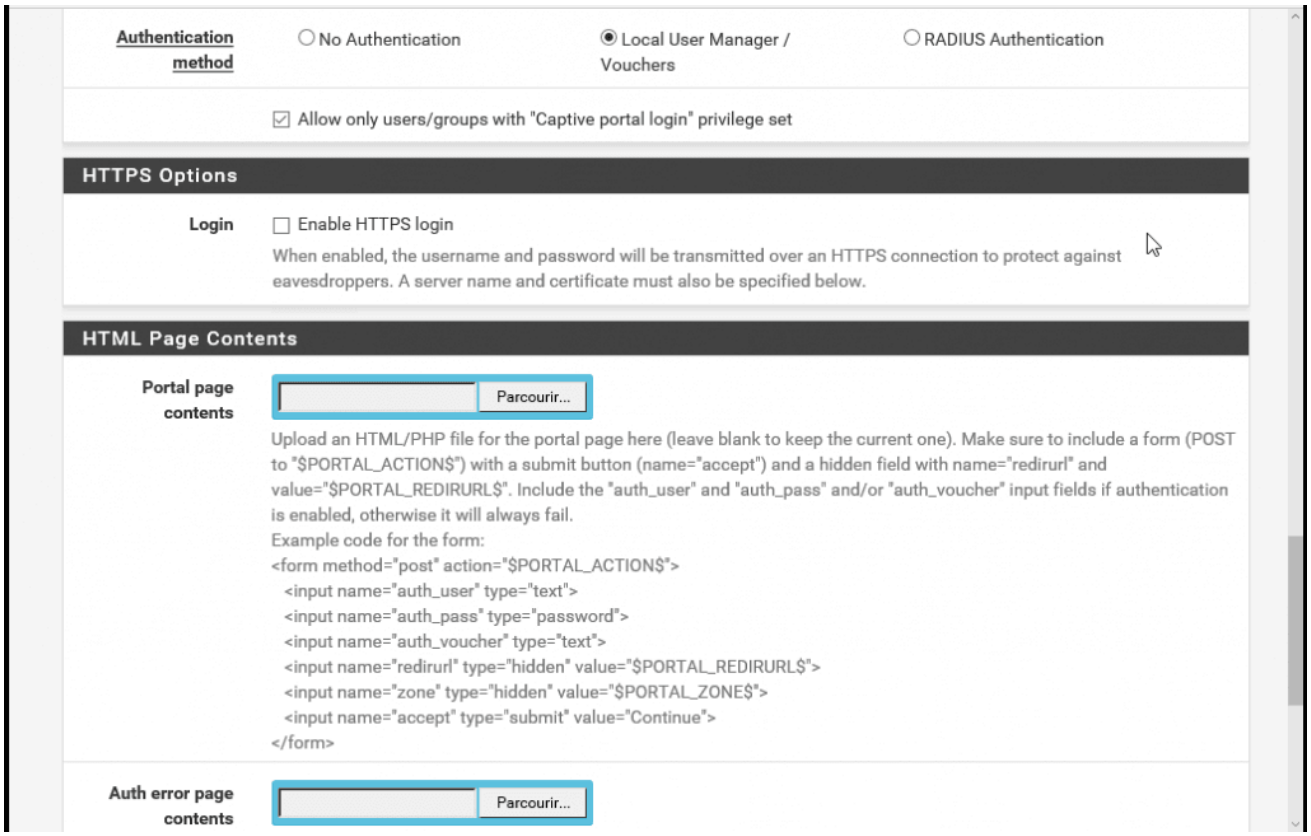

Figure 27 : configuration du portail captif : selection de la methode d'authentification et HTTPS*

Nous utiliserons la méthode d'uahtentification via un serveur radius. Une fois les informations du radius entrées, l'authentification doit normalement pouvoir se tester à partir de pfSense, il ne devrait pas avoir de problèmes.

Le problème que nous avons rencontré sur cette partie a été l'affichage du portail captif.
En effet, lorsque les utilisateurs essayaient de se connecter à n'importe quel site, ils tombaient sur une erreur qui leur disait qu'on ne pouvait pas résoudre le nom de la page.
Cela venait en fait du fait que par défaut, les navigateurs envoient des requêtes HTTPS lorsque l'on entre quelque chose dans la barre de recherche. Lorsqu'on utilisait uniquement des requêtes HTTP, le portail s'affichait. Nous avons donc mis en place le portail captif en HTTPS, et à partir de là, nous n'avons plus eu de problèmes. N'importe quel utilisateur pouvait alors accéder au portail captif, se connecter avec ses identifiants Aix-Marseille Université et ensuite accéder à internet.

9 Conclusion

Durant mon stage à la DOSI, j'ai pu aborder beaucoup des aspects de l'informatique. Mon travail en développement sur GRIP était particulièrement enrichissant, et il est très gratifiant de voir que mon travail est aujourd'hui utilisé par les employés de la DOSI et qu'il permet une utilisation plus simple. Voir aussi les micro-commutateurs être utilisés et fonctionner correctement est aussi une réussite. Il est aussi très enrichissant de travailler sur des projets concrets qui nous permettent d'évaluer des contraintes physiques ou même des contraintes humaines.
Lors de mon stage, j'ai pu observer certaines interventions dans certains bâtiments de Luminy avec l'équipe réseau. On se rend très vite compte que le métier d'administrateur réseau n'est pas uniquement qu'informatique, il faut aussi gérer l'infrastructure physique, repérer des fibres, diagnostiquer des problèmes de câbles, prendre contact avec des entreprises pour l'installation de nouveaux passages de câbles ou réparer l'existant.
On peut toujours voir des améliorations sur GRIP, comme par exemple améliorer la visibilité du nom des rocades pour rendre la recherche de rocades plus simple. Il faudrait aussi essayer de basculer la totalité de GRIP en PHP pour permettre une véritable cohérence au sein du programme.

Ce stage m'a beaucoup apporté, les multiples missions qui m'ont été confiées étaient très intéressantes et toutes très différentes, ce qui m'a permis de me faire une idée de tous les aspects des métiers du réseau, et de l'informatique en général. La formation que j'ai reçue à l'IUT m'a permis d'avoir de solides bases pour répondre correctement au travail qui m'était demande durant la totalité de mon stage. Cette expérience professionnelle confirme la voie que j'emprunte, celle de l'ingénierie informatique et réseaux. C'est un milieu très stimulant qui me permet de toujours me surpasser en relevant de nouvelles épreuves.

10

10 Remerciements

Je tiens à remercie toutes les personnes qui ont contribué au succès de ce stage.

Je tiens à remercier mon tuteur en entreprise, Monsieur Pascal MOURET, qui a pris beaucoup de temps pour m'aider et me faire avancer dans mes projets. Son expertise et sa pédagogie m'ont permis d'avancer dans toutes mes missions, et de les mener à bien.

J'adresse aussi mes remerciements à Monsieur Laurent MICHAUDET, qui en l'absence de Pascal, a pris du temps pour m'aider dans mes missions, m'en proposer des nouvelles, ou même me faire des retours sur l'utilisation du logiciel GRIP.

Je remercie également tout le service de la DOSICALU pour leur accueil, leur bonne humeur, et leur gentillesse.

Enfin, je tiens à remercier encore une fois Monsieur Pascal MOURET pour la relecture de mon rapport et pour les multiples conseils qu'il m'a donné.

11 Glossaire

DUT, Diplôme Universitaire de Technologie

PHP, Personal Home Page tools

AJAX, Asynchronous Javascript And XML

HTTP, HyperText Transfer Protocol

CSS, Cascade Style Sheet

ACL, Access Control List

SFP, Small Form-Factor Pluggable

PoE, Power over Ethernet

VLAN, Virtual Local Area Network

LAN, Local Area Network

WAN, Wide Area Network

HTTPS, HyperText Transfer Protocol Secure

12 Bibliographie

Wikipedia (14 mai 2019). *Perl* (https://fr.wikipedia.org/wiki/Perl_(langage))

Wikipedia (5 juin 2019). *PostgreSQL* (https://fr.wikipedia.org/wiki/PostgreSQL)

Wikipedia (8 juin 2019). *PHP* (https://fr.wikipedia.org/wiki/PHP)

Wikipedia (16 juin 2019). *Javascript* (https://fr.wikipedia.org/wiki/JavaScript)

Wikipedia (26 mars 2019). *pfSense* (https://fr.wikipedia.org/wiki/PfSense)

HOMMET, Julien (2017). *pfSense 2.3.3 Mise en place d'un portail captif* (https://computerz.solutions/pfsense-portail-captif/)

Microsens, ruggedized switch (https://www.microsens.com/fr/produits/categories/micro-switchs/gigabit-ethernet/serie/Serie/show/gigabit-ethernet-micro-switch-6-port-poe-generation-6-776/)

Microsens. Basic configuration of G6 devices (https://www.microsens.com/fileadmin/user_upload/docs/app_notes/AN-16010_Basic_Config_of_G6_Devices_v1.0.pdf)

Institut Universitaire de Technologie,
Aix-Marseille Université

ANNEXES
Diplôme Universitaire de Technologie
Spécialité Réseaux et Télécommunications

Restructuration graphique et logique d'un logiciel de supervision réseau au niveau physique

Benjamin SABY

Direction Opérationnelle des Systèmes d'Informations (Aix-Marseille Université)

Responsable entreprise : Pascal MOURET

Responsable académique : Arnaud FEVRIER

2019

Script de mise en place des ports :

```
            $racknum = $argv[1]; // on récupère le numéro de baie en argument
            $dbconn = pg_connect("host=".$DBHOST." dbname=".$DBNAME."
user=".$DBUSER." password=".$DBPASS); //on définit la connexion à la base de
données
            $query = "SELECT DISTINCT num_loc FROM baies WHERE
num=".$racknum.";"; // on récupère le numéro de local
            $r = pg_exec($dbconn, $query);
            $numrow = pg_numrows($r); // nombre de lignes de la réponse
            $l=pg_fetch_array($r,0);
            echo "var numloc='".$l["num_loc"]."';";
            $query = "SELECT
c.num_port,c.nom,m.num,d.nom,c.num_pt,m.pos,COUNT(ca.num),m.rem,c.ligne FROM
designations d JOIN materiels m ON m.num_desi = d.num JOIN composantes c ON
c.num_desi = d.num LEFT OUTER JOIN cables ca ON (c.num_port = ABS(ca.num_port1)
AND m.num = ca.num_mat1) OR ((c.num_port = ABS(ca.num_port2) AND m.num =
ca.num_mat2)) WHERE m.num_baie=".$racknum." GROUP BY
c.nom,d.nom,c.num_pt,m.pos,m.rem,c.ligne,m.num,c.num_port,m.ord ORDER BY
m.ord,m.num,c.ligne,c.num_port;"; //query récupération des ports
            $r = pg_exec($dbconn, $query);
            $z=1; //variable d'incrémentation d'id
            $x=180; //variable position x
            $y=50; //variable position y
            $rattrapage = false; //variable de saut d'id
            $numrow = pg_numrows($r); // nombre de row  de la réponse
            $s=pg_fetch_array($r,1);
            $oldid = $s["num"]; // récupération de l'id du premier switch
            $switcheslocaux[] = $s["num"];
            $oldligne = $s["ligne"]; // récupération de la première valeur de
ligne
            $correspondance = array(); // on initialise la variable qui nous
permettra de faire correspondre l'id de node avec le numéro de switch et de port
            $font = "";
            echo "var rack ='".$racknum."';"; //on stocke le numéro de baie dans
une variable javascript
            echo 'var nodes = new vis.DataSet([; //on commence la définition des
nodes
            for($i=0; $i<$numrow; $i++){
                $l=pg_fetch_array($r,$i);
                if($l[6] == "0"){ // si le port n'a pas de brassage
                        $font = "font: {strokeWidth: 2, strokeColor: 'green'}, ";
//le texte sera en vert
                    }
                    else{ // si le port est déjà brassé
                        $font = "font: {strokeWidth: 2, strokeColor: 'red'}, ";
// le texte sera en rouge
                    }
                if($l["num"]==$oldid && $l["ligne"] == $oldligne){ // si on est
toujours sur le même commutateur et toujours sur la même ligne
                    if($i==0){ // si on est sur le traitement du premier node
                        $nom = trim($l[3]);
                        $chaine = addslashes(preg_replace("/\s+/"," ",$l[7]));
                        echo "{id: ".$i.", label:
'".$nom."\\n".$l["pos"]."\\n".$chaine."', shape: 'box', color: '#36B9FF', x:0,
y:".$y."},"; //on rajoute un node d'info sur l'équipement
                        $correspondance[$i] = "nomsw,".$l[2]; //on indique dans
la table correspondance que ce node là est une info d'équipement
                        if($i<$numrow-1){
                            echo "{id: ".$z.", label: '".$l[1]."',".$font."
image: '".$DIRECTORY."/imgports/".$l["num_pt"].".png', shape: 'image', x:".$x.",
y:".$y."},"; //on ajoute le node du premier port
```

```
                            $correspondance[$z] = $l[2].",".$l["num_port"]; //on
indique dans la table de correspondance que ce port là appartient au numéro
d'équipement $l[2] et a pour numero de port $l["num_port"]
                        }
                        else {
                            echo "{id: ".$z.", label: '".$l[1]."',".$font."
image: '".$DIRECTORY."/imgports/".$l["num_pt"].".png', shape: 'image', x:".$x.",
y:".$y."}";
                            $correspondance[$z] = $l[2].",".$l["num_port"]; //on
indique dans la table de correspondance que ce port là appartient au numéro
d'équipement $l[2] et a pour numero de port $l["num_port"]
                        }
                        $rattrapage = true; // on définit la variable pour
rattraper l'écart d'id qui cause des problèmes
                    }
                    else{ // si i est supérieur à 0
                        $x+=70; // on se décale à droite
                        if($rattrapage == true){
                            $z = $z +1; // on saute un id
                        }
                        if($i<$numrow-1){
                            echo "{id: ".$z.", label: '".$l[1]."',".$font."
image: '".$DIRECTORY."/imgports/".$l["num_pt"].".png', shape: 'image', x:".$x.",
y:".$y."},"; // on ajoute le port
                            $correspondance[$z] = $l[2].",".$l["num_port"]; //on
indique dans la table de correspondance que ce port là appartient au numéro
d'équipement $l[2] et a pour numero de port $l["num_port]
                        }
                        else {
                            echo "{id: ".$z.", label: '".$l[1]."',".$font."
image: '".$DIRECTORY."/imgports/".$l["num_pt"].".png', shape: 'image', x:".$x.",
y:".$y."}";
                            $correspondance[$z] = $l[2].",".$l["num_port"]; //on
indique dans la table de correspondance que ce port là appartient au numéro
d'équipement $l[2] et a pour numero de port $l["num_port"]
                        }
                    }
                }
                else if($l["num"]==$oldid && $l["ligne"] != $oldligne){ // si on
est toujours sur le même équipement, mais que la ligne a changée
                    $x=180; // on se remet à gauche
                    $y+=90; // on se déplace un peu plus vers le bas
                    if($rattrapage == true){
                            $z = $z +1;
                        }
                        if($i<$numrow-1){
                            echo "{id: ".$z.", label: '".$l[1]."',".$font."
image: '".$DIRECTORY."/imgports/".$l["num_pt"].".png', shape: 'image', x:".$x.",
y:".$y."},";
                            $correspondance[$z] = $l[2].",".$l["num_port"]; //on
indique dans la table de correspondance que ce port là appartient au numéro
d'équipement $l[2] et a pour numero de port $l["num_port"]
                        }
                        else {
                            echo "{id: ".$z.", label: '".$l[1]."',".$font."
image: '".$DIRECTORY."/imgports/".$l["num_pt"].".png', shape: 'image', x:".$x.",
y:".$y."}";
                            $correspondance[$z] = $l[2].",".$l["num_port"]; //on
indique dans la table de correspondance que ce port là appartient au numéro
d'équipement $l[2] et a pour numero de port $l["num_port"]
                        }
                }
                else { //si on change d'équipement
                    $x=180; // on revient à gauche
                    $y+=100; // on revient à la ligne
```

```
                    $z=$z+1;
                    $nom = trim($l[3]);
                    $chaine = addslashes(preg_replace("/\s+/"," ",$l[7]));
                    echo "{id: ".$z.", label:
'".$nom."\\n".$l["pos"]."\\n".$chaine."', shape: 'box', color: '#36B9FF', x:0,
y:".$y."},"; // on annonce les infos de l'équipement
                    $correspondance[$z] = "nomsw,".$l[2];  //on indique dans la
table correspondance que ce node là est une info d'équipement
                    $z=$z+1;
                    if($i<$numrow-1){
                        echo "{id: ".$z.", label: '".$l[1]."',".$font." image:
'".$DIRECTORY."/imgports/".$l["num_pt"].".png', shape: 'image', x:".$x.",
y:".$y."},"; // on ajoute le port
                        $correspondance[$z] = $l[2].",".$l["num_port"];//on
indique dans la table de correspondance que ce port là appartient au numéro
d'équipement $l[2] et a pour numero de port $l["num_port"]
                    }
                    else {
                        echo "{id: ".$z.", label: '".$l[1]."',".$font." image:
'".$DIRECTORY."/imgports/".$l["num_pt"].".png', shape: 'image', x:".$x.",
y:".$y."}";
                        $correspondance[$z] = $l[2].",".$l["num_port"];
                    }
                    $rattrapage = true; //on place la variable de saut d'id à
vraie
                }

                $oldid = $l["num"]; // on stocke la variable actuelle de numéro
d'équipement dans la variable oldid
                $oldligne = $l["ligne"]; //on stocke la variable actuelle de
numéro de ligne dans la variable oldligne
                if(!in_array($l[2], $switcheslocaux)){ // si le numéro
d'équipement n'existe pas dans la liste des équipements de la baie
                    $switcheslocaux[] = $l[2]; //on rajoute le numéro
d'équipement actuel dans la liste des équipement de la baie
                }

            }

            echo "]);";

            echo "var correspondance = [";  //ici on créée le tableau javascript
avec toutes les valeurs de correspondance
            $lengthcor = sizeof($correspondance);
            for($y=0; $y<$lengthcor; $y++){
                if ($y<$lengthcor-1) {
                    echo "'".$correspondance[$y]."',";
                }
                else {
                    echo "'".$correspondance[$y]."'";
                }
            }
            echo "];";
```

Script de recherche de l'ensemble des câbles de la baie :

```
            echo "var schemacomplet = new vis.DataSet([";
            $nbrswitch = sizeof($switcheslocaux);
            for($i=0; $i<$nbrswitch; $i++){
                $query2 = "SELECT num_mat1, num_mat2, num_port1, num_port2 FROM
cables WHERE num_mat1=".$switcheslocaux[$i].";";
                $t = pg_exec($dbconn, $query2);
                $nbrrow = pg_numrows($t);
                for($y=0; $y<$nbrrow; $y++){
                    $result=pg_fetch_array($t,$y); // on va parcourir le resultat
                    if(substr($result["num_port1"],0,1) == '-'){ //si numéro de
port négatif
                        $num_port1 = substr($result["num_port1"], 1);//on enlève
le -
                    }
                    else {//sinon on prend le numéro de port
                        $num_port1 = $result["num_port1"];
                    }
                    if(substr($result["num_port2"],0,1) == '-'){ //si numéro de
port négatif
                        $num_port2 = substr($result["num_port2"], 1); //on enlève
le -
                    }
                    else {
                        $num_port2 = $result["num_port2"];
                    }
                    $indexsource =
array_search($result["num_mat1"].",".$num_port1, $correspondance); // on cherche
le numéro de nœud correspondant à cet équipement, et ce numéro de port
                    $indexdestination =
array_search($result["num_mat2"].",".$num_port2, $correspondance); // on cherche
le numéro de nœud correspondant à cet équipement, et ce numéro de port
                    if(!($indexsource == "" || $indexdestination == "")){ //si
les deux valeurs sont correctes, on écrit la liaison
                        echo "{from: ".$indexsource.", to:
".$indexdestination."},";
                    }
                }
            }
            echo "]);";
```

quelques exemples de fonctions JavaScript :

```
function calcul(){
	var idsource = [];
	var iddestination = [];
	var lengthcor = correspondance.length;
	var double = '';
	for (var i=0; i<lengthcor; i++){ // on parcours la totalité des noeuds pour savoir si ils sont connectés à quelque chose
		var valeur = network.getConnectedNodes(i.toString()); //on récupère le numero du noeud relié au noeud sur lequel nous sommes placés
		if(valeur.length == 1){ // si il est connecté à un seul noeud
			if(parseInt(i.toString(),10) < parseInt(valeur.toString(),10)){ //si l'id du noeud de destination est plus grand
				idsource.push(i.toString()); //on rajoute le noeud sur lequel nous sommes placés dans le tableau idsource
				iddestination.push(valeur.toString()); //on rajoute le noeud relié dans le tableau iddestination
			}
		}
		else if(valeur.length == 2){ // si il est connecté à deux noeud
			if(parseInt(i.toString(),10) < parseInt(valeur[0].toString(),10) && parseInt(i.toString(),10) < parseInt(valeur[1].toString(),10)){ //si les noeuds de destination ont un id plus grand
				idsource.push(i.toString()); //on met le noeud source dans le tableau idsource une première fois
				iddestination.push(valeur[0].toString()); //on met le premier noeud de destination dans le tableau iddestination
				idsource.push(i.toString()); //on met le noeud source dans le tableau idsource une deuxième fois
				iddestination.push(valeur[1].toString()); //on met le deuxième noeud de destination dans le tableau iddestination
			}
		}
		else if(valeur.length > 2) { // si il y a plus de deux noeuds
			alert('Vous ne pouvez pas brasser plus de 2 liens sur le même port !');
			return;
		}
	}
	var lengthsource = idsource.length;
	for(var i=0; i<lengthsource; i++){ //dans cette boucle on organise les informations, puis on envoie à la base de données les nouveaux brassages grace à la fonction postAjax
		var indexsource = idsource[i];
		var indexdestination = iddestination[i];
		var infosource = correspondance[indexsource].split(',');
		var infodestination = correspondance[indexdestination].split(',');
		var num_mat1 = infosource[0];
		var num_mat2 = infodestination[0];
		var num_port1 = infosource[1];
		var num_port2 = infodestination[1];
		postAjax(num_mat1,num_mat2,num_port1,num_port2);
	}
}
```

```
function postNode(idnode){
	var infonode = correspondance[idnode].split(',');
	var num_mat = infonode[0]; // on récupère les infos du port
	var num_port = infonode[1];
	var xmlhttp;var realedges = new vis.DataSet();
	if(window.XMLHttpRequest) {
		xmlhttp = new XMLHttpRequest();
	}
	xmlhttp.onreadystatechange = function() {
		if(xmlhttp.readyState == 4 && xmlhttp.status == 200) {
			var tableauconnec = xmlhttp.responseText.split(';');
			for(var i=0; i<tableauconnec.length; i++){
				var infoacorrespondre = tableauconnec[i].split('/');
				var from = correspondance.indexOf(infoacorrespondre[0]);
				var to = correspondance.indexOf(infoacorrespondre[1]);
				realedges.add({from: from, to: to}); //on rajoute les
liens entre les différents noeuds
			}
			network.setData({nodes: nodes, edges: realedges});
		}
	};
	xmlhttp.open('POST','edges.php',true);
	xmlhttp.setRequestHeader('Content-type','application/x-www-form-
urlencoded');
	xmlhttp.send('nummat='+num_mat+'&numport='+num_port); // on envoie en POST
les infos sur edges.php


function ajaxNodesRocades(idnode) {
	var infonode = correspondance[idnode].split(','); //on récupère les infos
du node
	if(infonode[0] == 'nomsw'){ //on vérifie si le node clické est un nom
d'équipement
		alert('Vous ne pouvez pas mettre de rocade sur un nom
d\'equipement');
		exit;
	}
	var num_mat = infonode[0];
	var num_port = infonode[1];
	var xmlhttp;
	if(window.XMLHttpRequest) {
		xmlhttp = new XMLHttpRequest();
	}
	xmlhttp.onreadystatechange = function() { //une fois que la requete nous
est revenu
		if(xmlhttp.readyState == 4 && xmlhttp.status == 200) {
			if(xmlhttp.responseText.split(',')[0] == 'rocade'){ //si la
réponse contient rocade
				document.getElementById('menuitem').innerHTML = 'Voir
rocade '+xmlhttp.responseText.split(',')[2];

	urldesti='modRocades.pl?num_roc='+xmlhttp.responseText.split(',')[1];
				document.getElementById('menuitem2').innerHTML = 'Voir
baie de destination'; //on met en forme le menu pour qu'il gère les rocades
				document.getElementById('menuitem2').style =
'display:block;';
				urldesti2 =
'voirBaie.pl?num='+xmlhttp.responseText.split(',')[3];
			}
			else if(xmlhttp.responseText.split(',')[0] == 'prise'){ //si la
réponse contient prises
				document.getElementById('menuitem2').style =
'display:none;'; //on met en forme le menu pour les prises
```

```
                        document.getElementById('menuitem').innerHTML = 'Voir
prise '+xmlhttp.responseText.split(',')[2]+' (Local
'+xmlhttp.responseText.split(',')[1]+')';

      urldesti='viewLoc.pl?num='+xmlhttp.responseText.split(',')[1]+'&chercheLoc=
suite';
                  }
                  else{ //si ce n'est ni une prise ni une rocade, ça veut dire
qu'on peut mettre une nouvelle prise/rocade
                        document.getElementById('menuitem2').style =
'display:none;';
                        document.getElementById('menuitem').innerHTML = 'Nouvelle
rocade/prise'; //donc on met le menu en forme

      urldesti='ajoutPrise.pl?num_mat='+num_mat+'&num_port='+num_port;
                  }
            }
      };
      xmlhttp.open('POST','chercherocade.php',true);
      xmlhttp.setRequestHeader('Content-type','application/x-www-form-
urlencoded');
      xmlhttp.send('nummat='+num_mat+'&numport='+num_port); //on envoie en POST à
chercherocade.php
}


function ajaxDelete(num_mat1, num_mat2, num_port1, num_port2) {
      var xmlhttp;
      if(window.XMLHttpRequest) {
            xmlhttp = new XMLHttpRequest();
      }
            xmlhttp.onreadystatechange = function() {
                  if(xmlhttp.readyState == 4 && xmlhttp.status == 200) {
                        alert(xmlhttp.responseText); //on affiche la réponse de
la page
                  }
            };
            xmlhttp.open('POST','supprBrassage.php',true);
            xmlhttp.setRequestHeader('Content-type','application/x-www-form-
urlencoded');

      xmlhttp.send('nummat1='+num_mat1+'&nummat2='+num_mat2+'&numport1='+num_port
1+'&numport2='+num_port2); //on envoie en POST sur la page supprBrassage.php les
infos des deux ports
      }
```

Et voici ce que donne l'intégralité du code de voirBaie.php (avec génération du HTML et des fonctions javascript) :

```
<?php
        echo "<input type='button' class='w3-button' value='Ajouter un cable'
onclick='addCable();'/>";
        echo "<input type='button' class='w3-button' value='R&eacute;initialiser'
onclick='reset(); document.getElementById(\"add\").style=\"display:
blocks;\"'/>";
        echo "<input type='button' class='w3-button' id='add' value='Ajouter le
brassage' onclick='calcul();'/>";
        echo "<input type='button' class='w3-button' value='Afficher cablage'
onclick='affichage(); document.getElementById(\"add\").style=\"display:
none;\"'/>";
        echo '<input type="button" class="w3-button w3-right" value="Brassage
inter-baie" onclick="changeinterbaie();"/>';
        echo "<input type='button' class='w3-button w3-right' value='Revenir
&agrave; la s&eacute;lection' onclick='changerevenir();'/>";
            echo '<div id="mynetwork"></div>';
            echo '<script type="text/javascript"
src="https://ajax.googleapis.com/ajax/libs/jquery/1.10.1/jquery.js"></script>';
            echo "<ul class='custom-menu'><li data-action='first'
id='menuitem'>Voir rocade</li><li data-action='second' id='menuitem2'
class='nonvisible'>2</li></ul>";
            echo '<script type="text/javascript">';
            $fichiertab = file("../etc/grip.conf"); //on va chercher dans le
fichier
            for($i=0; $i<sizeof($fichiertab); $i++){
                $mots = explode("=",$fichiertab[$i]);
                if($mots[0] == "db_user"){
                    $DBUSER = preg_replace("/'+/","",$mots[1]); // récupération
du user pour la base de données
                }
                else if($mots[0] == "db_name"){
                    $DBNAME = preg_replace("/'+/","",$mots[1]); // récupération
du nom de la base de données
                }
                else if($mots[0] == "db_host"){
                    $DBHOST = preg_replace("/'+/","",$mots[1]); // récupération
l'IP de la base de données
                }
                else if($mots[0] == "db_passwd"){
                    $DBPASS = preg_replace("/'+/","",$mots[1]); // récupération
du mot de passe user pour la base de données
                }
            else if($mots[0] == "html_static"){ // on récupère l'endroit ou les
fichiers statiques se trouvent
                $DIRECTORY = preg_replace("/'+/","",$mots[1]);
                $DIRECTORY =preg_replace("/\s+$/","",$DIRECTORY);
                $DIRECTORY = trim($DIRECTORY, '"');
                $DIRECTORY = trim($DIRECTORY, "'");
                $DIRECTORY = preg_replace('/\$html_base/',$BASE,$DIRECTORY);
            }
            else if($mots[0] == "html_base"){ // on récupère la racine de
l'arborescence
                    $BASE = preg_replace("/'+/","",$mots[1]);
                    $BASE =preg_replace("/\s+$/","",$BASE);
                    $BASE = trim($BASE, '"');
                    $BASE = trim($BASE, "'");
                }

            }
            $racknum = $argv[1];
```

```
            $dbconn = pg_connect("host=".$DBHOST." dbname=".$DBNAME."
user=".$DBUSER." password=".$DBPASS);
            $query = "SELECT DISTINCT num_loc FROM baies WHERE
num=".$racknum.";";
            $r = pg_exec($dbconn, $query);
            $numrow = pg_numrows($r); // nombre de row  de la réponse
            $l=pg_fetch_array($r,0);
            echo "var numloc='".$l["num_loc"]."';";
            $query = "SELECT
c.num_port,c.nom,m.num,d.nom,c.num_pt,m.pos,COUNT(ca.num),m.rem,c.ligne FROM
designations d JOIN materiels m ON m.num_desi = d.num JOIN composantes c ON
c.num_desi = d.num LEFT OUTER JOIN cables ca ON (c.num_port = ABS(ca.num_port1)
AND m.num = ca.num_mat1) OR ((c.num_port = ABS(ca.num_port2) AND m.num =
ca.num_mat2)) WHERE m.num_baie=".$racknum." GROUP BY
c.nom,d.nom,c.num_pt,m.pos,m.rem,c.ligne,m.num,c.num_port,m.ord ORDER BY
m.ord,m.num,c.ligne,c.num_port;";
            $r = pg_exec($dbconn, $query);
            $z=1; //variable d'incrémentation d'id
            $x=180; //variable position x
            $y=50; //variable position y
            $rattrapage = false; //variable de saut d'id
            $numrow = pg_numrows($r); // nombre de row  de la réponse
            $s=pg_fetch_array($r,1);
            $oldid = $s["num"]; // récupération de l'id du premier switch
            $switcheslocaux[] = $s["num"];
            $oldligne = $s["ligne"]; // récupération de la première valeur de
ligne
            $correspondance = array(); // on initialise la variable qui nous
permettra de faire correspondre l'id de node avec le numéro de switch et de port
            $font = "";
            echo "var rack ='".$racknum."';"; //on stocke le numéro de baie dans
une variable javascript
            echo 'var nodes = new vis.DataSet([; //on commence la définition des
nodes
            for($i=0; $i<$numrow; $i++){
                $l=pg_fetch_array($r,$i);
                if($l[6] == "0"){ // si le port n'a pas de brassage
                        $font = "font: {strokeWidth: 2, strokeColor: 'green'}, ";
//le texte sera en vert
                    }
                    else{ // si le port est déjà brassé
                        $font = "font: {strokeWidth: 2, strokeColor: 'red'}, ";
// le texte sera en rouge
                    }
                if($l["num"]==$oldid && $l["ligne"] == $oldligne){ // si on est
toujours sur le même commutateur et toujours sur la même ligne
                    if($i==0){ // si on est sur le traitement du premier node
                        $nom = trim($l[3]);
                        $chaine = addslashes(preg_replace("/\s+/"," ",$l[7]));
                        echo "{id: ".$i.", label:
'".$nom."\\n".$l["pos"]."\\n".$chaine."', shape: 'box', color: '#36B9FF', x:0,
y:".$y."},"; //on rajoute un node d'info sur l'équipement
                        $correspondance[$i] = "nomsw,".$l[2]; //on indique dans
la table correspondance que ce node là est une info d'équipement
                        if($i<$numrow-1){
                            echo "{id: ".$z.", label: '".$l[1]."',".$font."
image: '".$DIRECTORY."/imgports/".$l["num_pt"].".png', shape: 'image', x:".$x.",
y:".$y."},"; //on ajoute le node du premier port
                            $correspondance[$z] = $l[2].",".$l["num_port"]; //on
indique dans la table de correspondance que ce port là appartient au numéro
d'équipement $l[2] et a pour numero de port $l["num_port"]
                        }
                        else {
```

```
                            echo "{id: ".$z.", label: '".$l[1]."',".$font."
image: '".$DIRECTORY."/imgports/".$l["num_pt"].".png', shape: 'image', x:".$x.",
y:".$y."}";
                            $correspondance[$z] = $l[2].",".$l["num_port"]; //on
indique dans la table de correspondance que ce port là appartient au numéro
d'équipement $l[2] et a pour numero de port $l["num_port"]
                        }
                        $rattrapage = true; // on définit la variable pour
rattraper l'écart d'id qui cause des problèmes
                    }
                    else{ // si i est supérieur à 0
                        $x+=70; // on se décale à droite
                        if($rattrapage == true){
                            $z = $z +1; // on saute un id
                        }
                        if($i<$numrow-1){
                            echo "{id: ".$z.", label: '".$l[1]."',".$font."
image: '".$DIRECTORY."/imgports/".$l["num_pt"].".png', shape: 'image', x:".$x.",
y:".$y."},"; // on ajoute le port
                            $correspondance[$z] = $l[2].",".$l["num_port"]; //on
indique dans la table de correspondance que ce port là appartient au numéro
d'équipement $l[2] et a pour numero de port $l["num_port]
                        }
                        else {
                            echo "{id: ".$z.", label: '".$l[1]."',".$font."
image: '".$DIRECTORY."/imgports/".$l["num_pt"].".png', shape: 'image', x:".$x.",
y:".$y."}";
                            $correspondance[$z] = $l[2].",".$l["num_port"]; //on
indique dans la table de correspondance que ce port là appartient au numéro
d'équipement $l[2] et a pour numero de port $l["num_port"]
                        }
                    }
                }
                else if($l["num"]==$oldid && $l["ligne"] != $oldligne){ // si on
est toujours sur le même équipement, mais que la ligne a changée
                    $x=180; // on se remet à gauche
                    $y+=90; // on se déplace un peu plus vers le bas
                    if($rattrapage == true){
                            $z = $z +1;
                        }
                        if($i<$numrow-1){
                            echo "{id: ".$z.", label: '".$l[1]."',".$font."
image: '".$DIRECTORY."/imgports/".$l["num_pt"].".png', shape: 'image', x:".$x.",
y:".$y."},";
                            $correspondance[$z] = $l[2].",".$l["num_port"]; //on
indique dans la table de correspondance que ce port là appartient au numéro
d'équipement $l[2] et a pour numero de port $l["num_port"]
                        }
                        else {
                            echo "{id: ".$z.", label: '".$l[1]."',".$font."
image: '".$DIRECTORY."/imgports/".$l["num_pt"].".png', shape: 'image', x:".$x.",
y:".$y."}";
                            $correspondance[$z] = $l[2].",".$l["num_port"]; //on
indique dans la table de correspondance que ce port là appartient au numéro
d'équipement $l[2] et a pour numero de port $l["num_port"]
                        }
                }
                else { //si on change d'équipement
                    $x=180; // on revient à gauche
                    $y+=100; // on revient à la ligne
                    $z=$z+1;
                    $nom = trim($l[3]);
                    $chaine = addslashes(preg_replace("/\s+/"," ",$l[7]));
```

```
                    echo "{id: ".$z.", label:
'".$nom."\\n".$l["pos"]."\\n".$chaine."', shape: 'box', color: '#36B9FF', x:0,
y:".$y."},"; // on annonce les infos de l'équipement
                    $correspondance[$z] = "nomsw,".$l[2];  //on indique dans la
table correspondance que ce node là est une info d'équipement
                    $z=$z+1;
                    if($i<$numrow-1){
                        echo "{id: ".$z.", label: '".$l[1]."',".$font." image:
'".$DIRECTORY."/imgports/".$l["num_pt"].".png', shape: 'image', x:".$x.",
y:".$y."},"; // on ajoute le port
                        $correspondance[$z] = $l[2].",".$l["num_port"];//on
indique dans la table de correspondance que ce port là appartient au numéro
d'équipement $l[2] et a pour numero de port $l["num_port"]
                    }
                    else {
                        echo "{id: ".$z.", label: '".$l[1]."',".$font." image:
'".$DIRECTORY."/imgports/".$l["num_pt"].".png', shape: 'image', x:".$x.",
y:".$y."}";
                        $correspondance[$z] = $l[2].",".$l["num_port"];
                    }
                    $rattrapage = true; //on place la variable de saut d'id à
vraie
                }

                $oldid = $l["num"]; // on stocke la variable actuelle de numéro
d'équipement dans la variable oldid
                $oldligne = $l["ligne"]; //on stocke la variable actuelle de
numéro de ligne dans la variable oldligne
                if(!in_array($l[2], $switcheslocaux)){ // si le numéro
d'équipement n'existe pas dans la liste des équipements de la baie
                    $switcheslocaux[] = $l[2]; //on rajoute le numéro
d'équipement actuel dans la liste des équipement de la baie
                }

            }

            echo "]);";



            echo "var correspondance = [";  //ici on créée le tableau javascript
avec toutes les valeurs de correspondance
            $lengthcor = sizeof($correspondance);
            for($y=0; $y<$lengthcor; $y++){
                if ($y<$lengthcor-1) {
                    echo "'".$correspondance[$y]."',";
                }
                else {
                    echo "'".$correspondance[$y]."'";
                }
            }
            echo "];";


            echo "var schemacomplet = new vis.DataSet([";
            $nbrswitch = sizeof($switcheslocaux);
            for($i=0; $i<$nbrswitch; $i++){
                $query2 = "SELECT num_mat1, num_mat2, num_port1, num_port2 FROM
cables WHERE num_mat1=".$switcheslocaux[$i].";";
                $t = pg_exec($dbconn, $query2);
                $nbrrow = pg_numrows($t);
                for($y=0; $y<$nbrrow; $y++){
                    $result=pg_fetch_array($t,$y);
                    if(substr($result["num_port1"],0,1) == '-'){
                        $num_port1 = substr($result["num_port1"], 1);
```

```
                    }
                    else {
                        $num_port1 = $result["num_port1"];
                    }
                    if(substr($result["num_port2"],0,1) == '-'){
                        $num_port2 = substr($result["num_port2"], 1);
                    }
                    else {
                        $num_port2 = $result["num_port2"];
                    }
                    $indexsource =
array_search($result["num_mat1"].",".$num_port1, $correspondance);
                    $indexdestination =
array_search($result["num_mat2"].",".$num_port2, $correspondance);
                    if(!($indexsource == "" || $indexdestination == "")){
                        echo "{from: ".$indexsource.", to:
".$indexdestination."},";
                    }
                }
            }
            echo "]);";

            echo "var edges = new vis.DataSet();var container =
document.getElementById('mynetwork');";
            echo "var data = {nodes: nodes,edges: edges};";
            echo "var parametre = '';";
            echo "var urldesti = '';";
            echo "var urldesti2 = '';";
            echo "var options = {autoResize: true,height: '100%',width:
'100%',physics: {enabled: false},nodes: {fixed:true},edges: {width:6,color:
{color:'red', hover: 'green'}},interaction: {hover: true}};";
            echo "var network = new vis.Network(container, data, options);";
            echo "network.on('doubleClick', function (params) {var valeurint =
parseInt(params.nodes.toString());var valcor =
correspondance[valeurint].split(',')[0];if(valcor == 'nomsw'){var valcor =
correspondance[valeurint].split(',')[1];window.location =
'infoMat.pl?num_mat='+valcor;}else{reset();params.event = '[original
event]';postNode(params.nodes.toString());}});";
            echo "network.on('deselectNode', function (params) {params.event =
'[original event]';reset();});";
            echo "network.on('selectEdge', function (params) {if(confirm('Voulez
vous supprimer ce brassage ?')){var idnodes =
network.getConnectedNodes(network.getSelectedEdges().toString());var mat1 =
(correspondance[idnodes[0]].split(','))[0];var port1 =
(correspondance[idnodes[0]].split(','))[1];var mat2 =
(correspondance[idnodes[1]].split(','))[0];var port2 =
(correspondance[idnodes[1]].split(','))[1];ajaxDelete(mat1,mat2,port1,port2);netw
ork.deleteSelected();}});";
            echo "function affichage() {network.setData({nodes: nodes, edges:
schemacomplet});}";
            echo "function addCable(){network.addEdgeMode();}";
            echo "function reset() {var edges = new
vis.DataSet();network.setData({nodes: nodes, edges: edges});}";
            echo "function calcul(){var idsource = [];var iddestination = [];var
lengthcor = correspondance.length;var double = '';for (var i=0; i<lengthcor;
i++){var valeur = network.getConnectedNodes(i.toString());if(valeur.length ==
1){if(parseInt(i.toString(),10) <
parseInt(valeur.toString(),10)){idsource.push(i.toString());iddestination.push(va
leur.toString());}}else if(valeur.length == 2){if(parseInt(i.toString(),10) <
parseInt(valeur[0].toString(),10) && parseInt(i.toString(),10) <
parseInt(valeur[1].toString(),10)){idsource.push(i.toString());iddestination.push
(valeur[0].toString());idsource.push(i.toString());iddestination.push(valeur[1].t
oString());}}else if(valeur.length > 2) {alert('Vous ne pouvez pas brasser plus
de 2 liens sur le même port !');return;}}var lengthsource =
idsource.length;for(var i=0; i<lengthsource; i++){var indexsource =
```

```
idsource[i];var indexdestination = iddestination[i];var infosource =
correspondance[indexsource].split(',');var infodestination =
correspondance[indexdestination].split(',');var num_mat1 = infosource[0];var
num_mat2 = infodestination[0];var num_port1 = infosource[1];var num_port2 =
infodestination[1];postAjax(num_mat1,num_mat2,num_port1,num_port2);}}";
            echo "function postAjax(num_mat1, num_mat2, num_port1, num_port2){var
xmlhttp;if(window.XMLHttpRequest) {xmlhttp = new
XMLHttpRequest();}xmlhttp.onreadystatechange = function() {if(xmlhttp.readyState
== 4 && xmlhttp.status == 200)
{alert(xmlhttp.responseText);if(xmlhttp.responseText ==
'OK'){schemacomplet.add({from: correspondance.indexOf(num_mat1+','+num_port1),
to:
correspondance.indexOf(num_mat2+','+num_port2)});}}};xmlhttp.open('POST','ajoutBr
assage.php',true);xmlhttp.setRequestHeader('Content-type','application/x-www-
form-
urlencoded');xmlhttp.send('nummat1='+num_mat1+'&nummat2='+num_mat2+'&numport1='+n
um_port1+'&numport2='+num_port2);sleep(50);}";
            echo "function postNode(idnode){var infonode =
correspondance[idnode].split(',');var num_mat = infonode[0];var num_port =
infonode[1];var xmlhttp;var realedges = new
vis.DataSet();if(window.XMLHttpRequest) {xmlhttp = new
XMLHttpRequest();}xmlhttp.onreadystatechange = function() {if(xmlhttp.readyState
== 4 && xmlhttp.status == 200) {var tableauconnec =
xmlhttp.responseText.split(';');for(var i=0; i<tableauconnec.length; i++){var
infoacorrespondre = tableauconnec[i].split('/');var from =
correspondance.indexOf(infoacorrespondre[0]);var to =
correspondance.indexOf(infoacorrespondre[1]);realedges.add({from: from, to:
to});}network.setData({nodes: nodes, edges:
realedges});}};xmlhttp.open('POST','edges.php',true);xmlhttp.setRequestHeader('Co
ntent-type','application/x-www-form-
urlencoded');xmlhttp.send('nummat='+num_mat+'&numport='+num_port);}";
            echo "function ajaxDelete(num_mat1, num_mat2, num_port1, num_port2)
{var xmlhttp;if(window.XMLHttpRequest) {xmlhttp = new
XMLHttpRequest();}xmlhttp.onreadystatechange = function() {if(xmlhttp.readyState
== 4 && xmlhttp.status == 200)
{alert(xmlhttp.responseText);}};xmlhttp.open('POST','supprBrassage.php',true);xml
http.setRequestHeader('Content-type','application/x-www-form-
urlencoded');xmlhttp.send('nummat1='+num_mat1+'&nummat2='+num_mat2+'&numport1='+n
um_port1+'&numport2='+num_port2);}";
            echo "function sleep(milliseconds) {var currentTime = new
Date().getTime();while (currentTime + milliseconds >= new Date().getTime()) {}}
";
            echo "function
changeinterbaie(){window.location='selectioninterbaie.pl?loc='+numloc+'&num='+rac
k;}";
            echo "function
changerevenir(){window.location='viewLoc.pl?num='+numloc+'&chercheLoc=suite';}";
            echo 'network.on("oncontext", function (params)
{params.event.preventDefault();parametre =
params;getNearestNode(parametre);$(".custom-
menu").finish().toggle(100);$(".custom-menu").css({top: params.event.pageY +
"px",left: params.event.pageX + "px"});});$(document).bind("mousedown", function
(e){if (!$(e.target).parents(".custom-menu").length > 0){$(".custom-
menu").hide(100);}});$(".custom-menu
li").click(function(){switch($(this).attr("data-action")){case "first":
window.location=urldesti; break;case "second": window.location=urldesti2;
break;case "third": alert("third"); break;}$(".custom-menu").hide(100);});';
            echo 'function getNearestNode(event) {if (event.nodes.length > 0 ||
event.edges.length > 0) {return;}var x = event.pointer.canvas.x;var y =
event.pointer.canvas.y;var nearestDist;var nearestId;var positions =
network.getPositions();for (let id in positions) {let curX = positions[id].x;let
curY = positions[id].y;let curDist = (curX - x)*(curX - x) + (curY - y)*(curY -
y);if (nearestDist === undefined || nearestDist > curDist){nearestDist =
curDist;nearestId = id;}};ajaxNodesRocades(nearestId);}';
```

```
            echo "function ajaxNodesRocades(idnode) {var infonode =
correspondance[idnode].split(',');if(infonode[0] == 'nomsw'){alert('Vous ne
pouvez pas mettre de rocade sur un nom d\'equipement');exit;}var num_mat =
infonode[0];var num_port = infonode[1];var xmlhttp;if(window.XMLHttpRequest)
{xmlhttp = new XMLHttpRequest();}xmlhttp.onreadystatechange = function()
{if(xmlhttp.readyState == 4 && xmlhttp.status == 200)
{if(xmlhttp.responseText.split(',')[0] ==
'rocade'){document.getElementById('menuitem').innerHTML = 'Voir rocade
'+xmlhttp.responseText.split(',')[2];urldesti='modRocades.pl?num_roc='+xmlhttp.re
sponseText.split(',')[1];document.getElementById('menuitem2').innerHTML = 'Voir
baie de destination';document.getElementById('menuitem2').style =
'display:block;';urldesti2 =
'voirBaie.pl?num='+xmlhttp.responseText.split(',')[3];}else
if(xmlhttp.responseText.split(',')[0] ==
'prise'){document.getElementById('menuitem2').style =
'display:none;';document.getElementById('menuitem').innerHTML = 'Voir prise
'+xmlhttp.responseText.split(',')[2]+' (Local
'+xmlhttp.responseText.split(',')[1]+')';urldesti='viewLoc.pl?num='+xmlhttp.respo
nseText.split(',')[1]+'&chercheLoc=suite';}else{document.getElementById('menuitem
2').style = 'display:none;';document.getElementById('menuitem').innerHTML =
'Nouvelle
rocade/prise';urldesti='ajoutPrise.pl?num_mat='+num_mat+'&num_port='+num_port;}}}
;xmlhttp.open('POST','chercherocade.php',true);xmlhttp.setRequestHeader('Content-
type','application/x-www-form-
urlencoded');xmlhttp.send('nummat='+num_mat+'&numport='+num_port);}";
        echo "</script>";

        ?>
```