Institut Universitaire de Technologie,
Aix-Marseille Université

RAPPORT DE STAGE de fin de deuxième année
Bachelor Universitaire de Technologie
Spécialité Réseaux et Télécommunications
parcours cybersécurité

Proxy, sauvegarde de configuration, Red Hat Single Sign-On

Victor BLAAS

SOFTWAY MEDICAL

Responsable entreprise : Guillaume Goupil

Responsable académique : Arnaud Février

2023

Table des matières

1 Introduction

J’ai eu l'opportunité d'effectuer ce stage de dix semaines au sein du service infrastructure du département Solutions de Softway Medical.

Softway Medical est une entreprise qui édite des logiciels à destination des organismes de santé.

Au cours de ce stage, j’ai travaillé sur trois sujets différents :

1. Mise en place et configuration d’un serveur proxy

Pour mener à bien cette première mission, le sujet a été découpé en différentes actions. Tout d’abord la configuration d’un serveur Squid, une solution de proxy open-source selon les besoins propres à Softway Medical. Ensuite le traitement des logs avec la conception de divers graphiques et enfin l’automatisation de la configuration du serveur.

2. Sauvegarde de configuration des équipements réseaux à l’aide des solutions Rancid et Oxidized.
3. En fin de stage, j’ai pu aborder la notion d’authentification unique avec la solution Red Hat Single Sign-On.

Ce rapport détaille les différentes tâches qui m’ont été confiées durant ces dix semaines.

2 Présentation de l'entreprise et du service

2.1 Présentation de Softway Medical

Le groupe Softway Medical propose différentes solutions logicielles à destination des professionnels du domaine médical depuis plus de vingt-cinq ans. Ayant une croissance forte depuis plusieurs années, le groupe s'est imposé dans le paysage de la e-santé, étant à la fois éditeur et hébergeur d'applications. En étant éditeur, intégrateur et hébergeur, Softway Médical possède et développe des expertises uniques dans le monde du médical.
En 2019, l'effectif était de quatre-cents employés pour un chiffre d'affaires de quarante-huit millions d'euros. En 2022, l'entreprise compte environ huit-cent-soixante personnes pour un chiffre d'affaires de cent millions d'euros.
Le pôle Recherche et Développement est appelé Solutions chez Softway et est dirigé par Sherley Brothier. Le service Infrastructure fait partie des "Solutions".

Figure 1 : organisation technique

2.2 Le Service Infrastructure

Le service infrastructure de Softway Medical est sous la direction de Fabien Muller. Au sein de ce service, il existe différents métiers:

- La supervision (24/7/365) et l'exploitation assurent le bon fonctionnement des applications 7 jours sur 7 et 24 heures sur 24 et fournissent le premier niveau de support technique aux différents clients.
- Le déploiement est assuré par les administrateurs, divisés en plusieurs groupes, systèmes et réseaux.

- La conception, assurée par les architectes aidés d'administrateurs, qui développent et conçoivent les solutions d'infrastructures pour héberger, administrer, déployer, exploiter et superviser les différentes applications proposées par Softway Médical à ses clients. La conception touche différents domaines comme les systèmes, la sécurité, les bases de données ou le réseau.

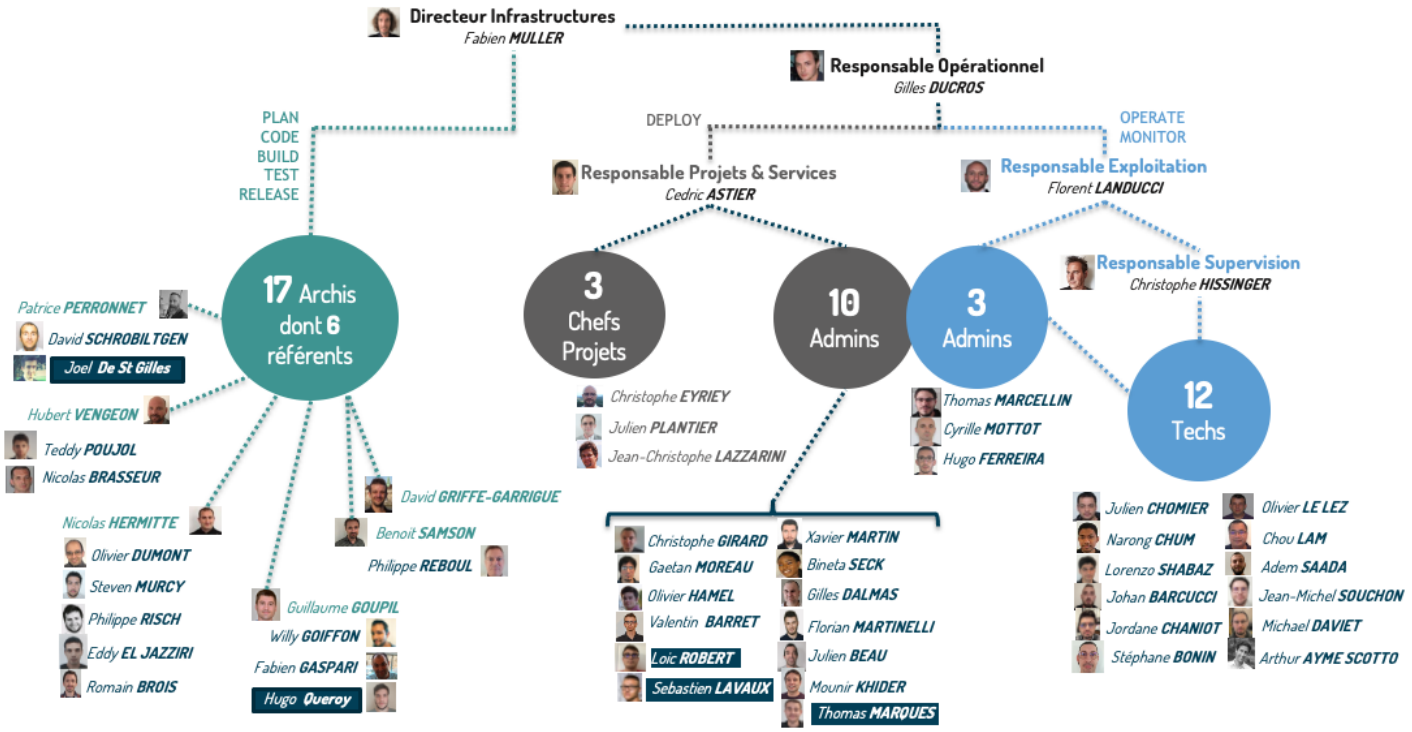

Figure 2 : organigramme Infrastructure

3 Présentation du premier sujet du stage

3.1 Contexte du projet

Pour ce projet de mise en place d'un serveur proxy, différents besoins ont été adressés.
Un serveur proxy peut en effet adresser différentes fonctions, comme du cache pour obtenir un gain de bande passante, du filtrage d'URL, de l'authentification ou encore une meilleure supervision du trafic réseau.
Dans notre cas, l'objectif principal a été de se conformer aux recommandations de l'Agence nationale de la sécurité des systèmes d'information, l'ANSSI. Cette agence conseille d'avoir un serveur mandataire proxy pour l'ensemble des flux issus du réseau de l'entreprise à destination d'internet pour empêcher les accès directs à internet. Le but est d'accroître la sécurité et également d'obtenir des informations sur le trafic passant sur le réseau. Il est en effet important de savoir ce qu'il se passe sur le réseau pour augmenter le niveau de sécurité et corriger rapidement les défaillances.
Voici un schéma qui illustre le résultat attendu pour ce projet.

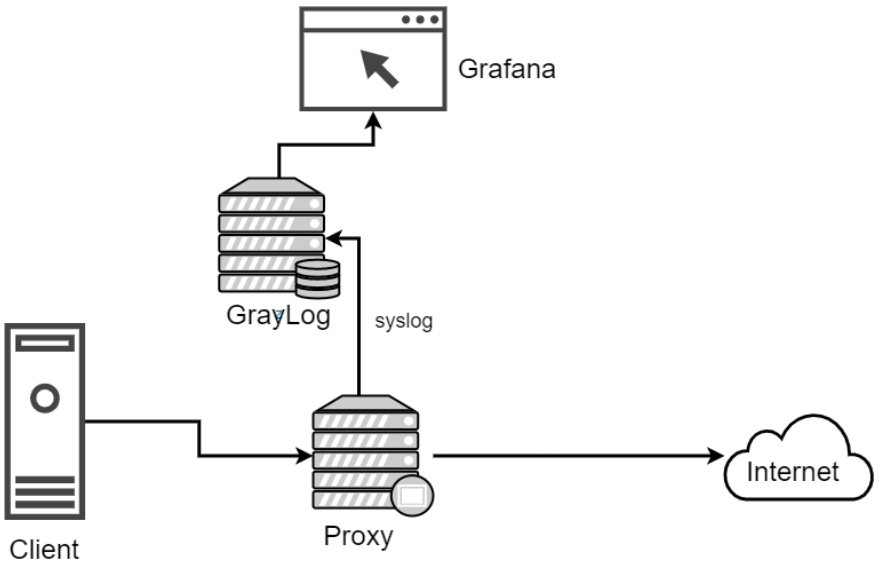

Figure 3 : Objectif du Proxy - empêcher les accès directs à internet

3.2 Qu'est-ce qu'un serveur proxy

Un serveur mandataire proxy est placé entre internet et le réseau protégé, c'est à dire que toute les requêtes à destination d'internet passe par un même serveur, ainsi si on regarde les adresses venant du réseau de l'entreprise sur le firewall et bien il n'y en aura plus qu'une, celle du serveur proxy. C'est donc une passerelle entre les serveurs du réseau et le reste de l'internet. Pour pouvoir faire cela, il faut opter pour un proxy explicite, donc les utilisateurs ont conscience de passer par un proxy et cela doit être configuré sur le poste client.

3.3 Quel proxy choisir ?

Il existe différents types de licences dans les mondes des serveurs proxy. L'offre open-source est intéressante et bien représentée. Les solutions open-souce sont souvent sous licence GNU/GPL ce qui signifie que c'est un logiciel libre, on est donc en droit d'exécuter le logiciel, pour n'importe quel usage, étudier le fonctionnement du programme et l'adapter à nos besoins. La licence oblige cependant à republier les modifications que vous avez apportées si c'est le cas.
La solution la plus répandue en matière de proxy est Squid. C'est open source, polyvalent et il répond parfaitement à un autre facteur, la présence de documentation pour faciliter l'apprentissage et la prise en main. Pour Squid il existe un wiki ainsi que de nombreux articles traitant du sujet. Cela témoigne donc de la popularité de Squid dans l'univers des proxys.

3.4 Présentation du travail réalisé.

3.4.1 Introduction à Squid

N'ayant pas de connaissances en proxy ou avec Squid, il m'a fallu passer par une phase d'étude de la documentation pour comprendre comment tout cela fonctionne et fournir une solution cohérente. C'est là que l'expérience de recherche et d'autoformation acquise durant certains cours et les divers projets me furent utiles. J'ai pu rapidement aboutir à un schéma simple du fonctionnement d'un proxy.

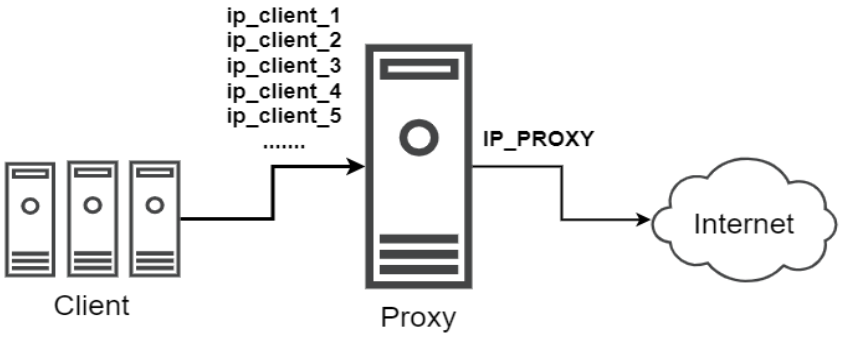

Figure 4 : Fonctionnement d'un proxy

Une fois la théorie acquise, la meilleure façon d'apprendre est de faire des tests. Pour cela, j'avais à ma disposition une machine virtuelle, Red Hat, basée sur le noyau linux, située sur le réseau de développement de l'entreprise.

3.4.2 Configuration de Squid

Basiquement, Squid fonctionne avec un système d'ACL, Access Control List, qui va indiquer quels réseaux ou utilisateurs ont le droit d'utiliser le proxy, mais également les ports et les protocoles autorisés. L'ordre d'écriture des ACL n'a pas d'importance par contre, l'ordre dans lequel on les applique lui en a. Squid prend en compte les ACL du haut vers le bas.

```
# acl fragment for Safe_ports
acl Safe_ports port 1025-65535
acl Safe_ports port 21
acl Safe_ports port 210
acl Safe_ports port 280
acl Safe_ports port 443
acl Safe_ports port 488
acl Safe_ports port 591
acl Safe_ports port 70
acl Safe_ports port 777
acl Safe_ports port 80

# acl fragment for permit_host
acl permit_host src 10.64.0.0/24
acl permit_host src localhost

# http_access fragment for Safe_ports
http_access allow Safe_ports

# http_access fragment for permit_host
http_access allow permit_host

# http_access fragment for all
http_access deny all
```

Figure 5 : Fonctionne des ACL Squid

On peut donc voir que d'abord on autorise l'ACL permit_host qui contient le réseau 10.64.0.0/24, puis on deny all, c'est-à-dire que l'on interdit tout ce qui n'est pas autorisé. De cette façon, seuls les réseaux contenus dans l'ACL permit_host pourront utiliser le proxy.

Avec ce type de proxy, il est aussi possible de configurer différentes solutions d'authentification, ainsi en plus de devoir être dans un réseau autorisé il faut également y être autorisé. Pour ce faire, Squid propose différent modules, allant de la lecture d'un simple fichier contenant les noms d'utilisateur et mot de passe à de l'authentification via un serveur LDAP que ce soit "basic", méthode simple offrant peu de sécurité car les échanges sont en clair, "Digest", qui permet un meilleur niveau de sécurité en hachant le mot de passe lors de l'envoi au serveur. Néanmoins, le principal problème

est que “Digest” ne fonctionne pas nativement avec Active Directory, utilisé dans de nombreuses entreprises. Il faut alors utiliser Kerberos qui est une méthode d'authentification de Microsoft basée sur l'échange de jetons. Le gros point fort de cette méthode est que le mot de passe ne transite pas sur le réseau.
Dans mon cas je ne me suis pas attardé trop longtemps sur les méthodes d’authentification car il n’y en avait pas le besoin.

3.4.3 Gestion de la redondance

Si des serveurs ou tout autre utilisateur ont besoin du proxy pour accéder à Internet, il est impératif qu'en cas de problème mettant le proxy dans l’incapacité de fonctionner qu’un autre serveur puisse prendre la relève.

Après consultation de mon tuteur, le système “actif-passif” a été choisi, c’est-à-dire qu’il y a un serveur qui reçoit l’ensemble des requêtes (actif) et un second serveur (passif) qui lui prendra le relais si jamais le premier serveur avait un dysfonctionnement. Il faut donc pouvoir détecter quand le proxy n’est plus en mesure d’effectuer sa tâche. Pour effectuer cela, j’ai choisi keepalived, sous une licence GNU GPU, également simple à configurer et permet de faire exactement ce dont j’ai besoin. Cela fonctionne avec un serveur “MASTER” et un serveur “BACKUP”, actif-passif, et une VIP, virtual IP address. En configurant les clients pour qu’ils utilisent cette adresse, si le MASTER a un problème, la VIP ira sur serveur BACKUP et ainsi, il n’y aura aucun changement à faire sur les clients et qu’une seule adresse à configurer.

Pour détecter si le MASTER a un problème, il y avait plusieurs solutions, la première, tant que le service keepalived est actif sur le MASTER, alors le serveur garde la VIP. Cette solution est un peu trop simpliste, car si keepalived est actif mais que squid ne fonctionne pas alors la VIP ne sera pas cédée. Deuxième solution, on vérifie que le service keepalived est actif ainsi que le service squid avec l'option “vrrp_track_process”. Ici le problème est le temps, car si on arrête le service squid par exemple avec systemctl stop squid, alors cela peut prendre de longues secondes avant de changer la VIP de serveur. Il existe une troisième solution avec “vrrp_script” de keepalived, on peut faire ainsi écrire un script bash qui vient faire à intervalle régulier des tentatives de requête en passant par le proxy grâce au module squidclient. Si la requête n'aboutit pas, il cédera directement la VIP. C’est donc cette troisième option que j’ai choisi, l'inconvénient de cette méthode est le volume de logs généré, car chaque tentative de requête est enregistrée.

Désormais, peut importe la raison pour laquelle le serveur Squid se retrouve dans l’incapacité de tenir son rôle de passerelle, le serveur passif prendra automatiquement le relais.

3.4.4 Mise en place de la récupération des logs

Maintenant que nous avons un proxy fonctionnel, il est important d’être capable de voir les requêtes effectuées au travers du proxy. Squid le fait parfaitement en émettant une ligne de log par défaut dans le fichier /var/log/squid/access.log, pour chaque requête. Squid offre également la possibilité de changer ce fichier pour par exemple l’envoyer vers un serveur syslog pour centraliser ses derniers, syslog étant un protocole permettant entre autres l’envoi de log.
Comme il existe déjà un serveur de log, j’ai pu envoyer dessus les logs de mon serveur, sans autre modification de la configuration de mon serveur que celle de Squid avec la ligne “access_log syslog:local2.info squid” car l’ensemble des VMs dans le réseau de développement sont configurées par défaut pour le faire, nous verrons plus tard comment cela a été possible.

Cependant, dans un premier temps, je n’ai pas utilisé cette méthode car avec le mécanisme mis en place pour la redondance expliqué plus tôt, j’ai des logs qui ne doivent pas être envoyés. J’ai donc fait un script en python qui envoie les logs en syslog avec un simple “if ” qui vérifie si c’est un log

correspondant à une requête faite par le script keepalived, pour ne pas l'envoyer si c'est le cas. De plus, je modifie les logs en identifiant "ip_client", "connect_typ", "port", et "url" de façon à rendre le traitement ultérieur plus simple.

Avec ces logs, il peut être intéressant de faire des graphiques pour faire ressortir certaines informations. Pour ce faire, Softway Medical utilise dans un premier temps Graylog qui permet la gestion des logs et dans un second temps, Grafana pour faire les graphiques.
Une fois les logs sur le serveur syslog, on peut les traiter avec Graylog, on retrouve les logs de l'ensemble des machines, dont le serveur proxy Squid qui envoie leur logs vers ce serveur spécifique. On a différents "labels" sur Graylog, grâce auxquels Grafana pourra par la suite identifier des informations. Le problème est que dans notre cas l'ensemble des informations est dans le label "message". Heureusement, il existe sur Graylog les extracteurs, pour pouvoir transférer des données d'un champ à un autre. C'est pour cela que j'ai avec le script python identifié chaque information, ainsi avec un extracteur d'expression régulière, je peux avoir les informations que je souhaite.

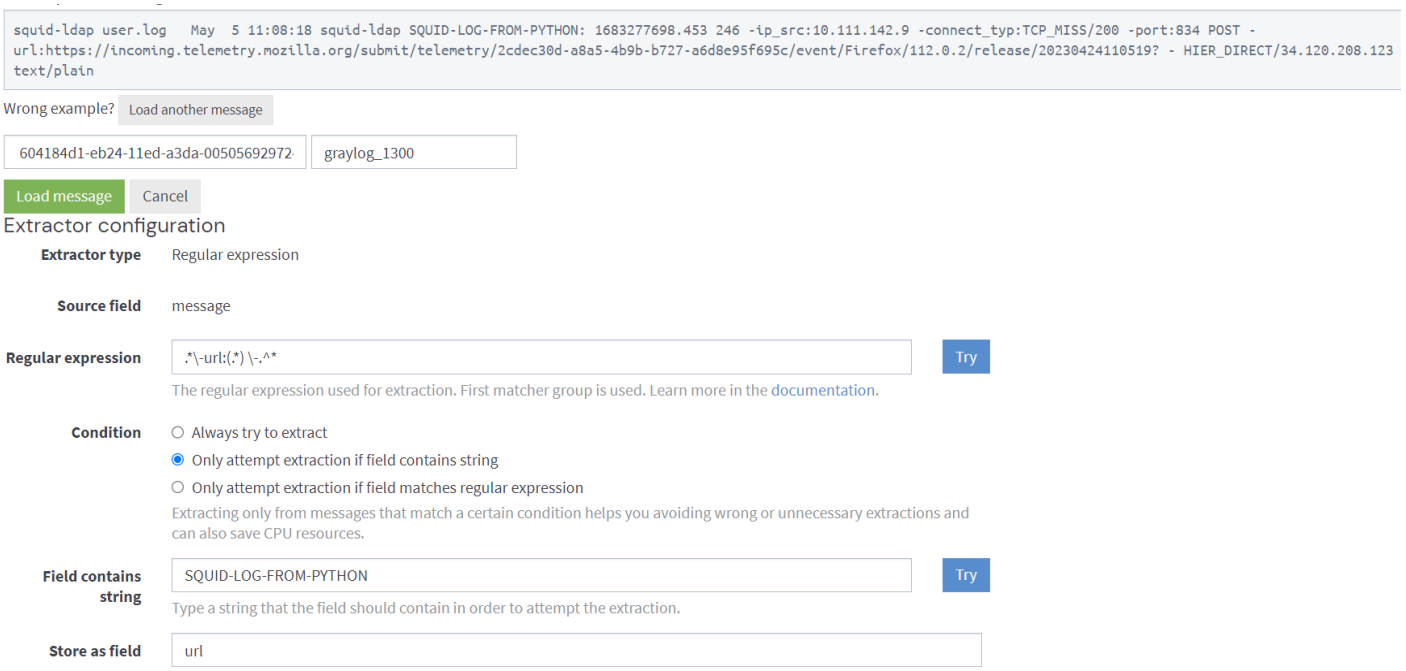

Figure 6: url extractor graylog

Avec des extracteurs comme la figure 4 pour chaque information cela nous donne un log exploitable avec Grafana.

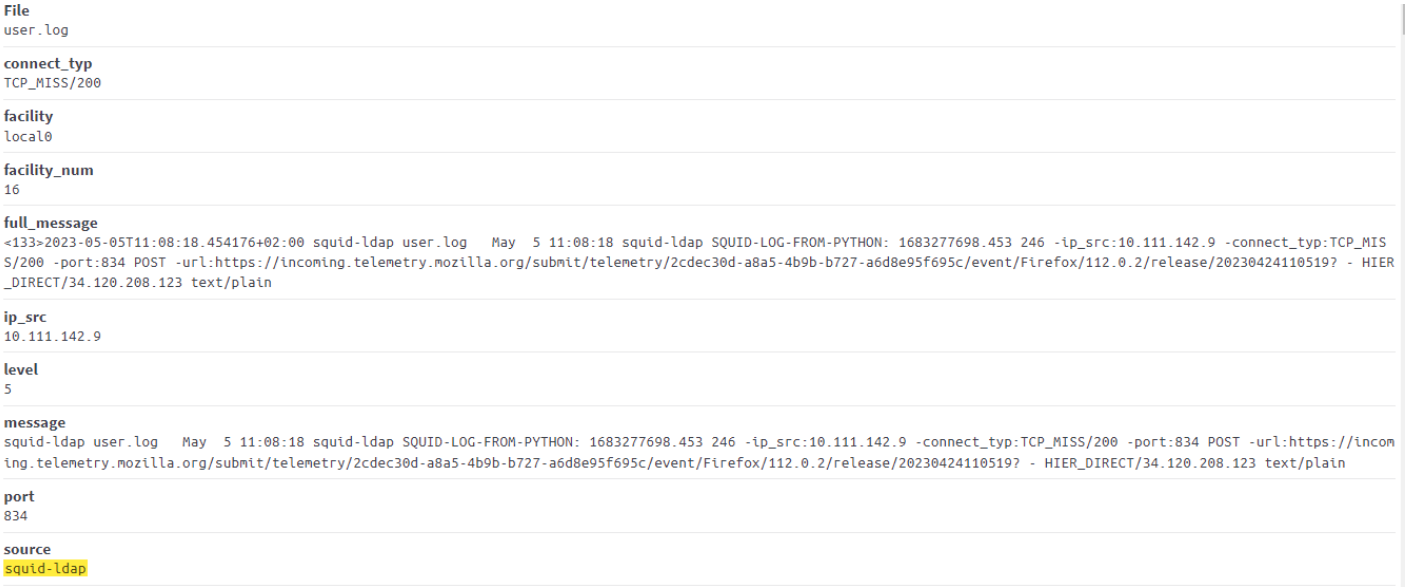

Figure 7: Log sur Graylog

Sur Grafana, il est possible de créer différents graphiques ou tableaux. Pour récupérer les informations que l'on souhaite afficher il y a un système de requête (query), fonctionnant sur les labels. Par exemple, si l'on veut afficher les ip sources, c'est-à-dire les ip des clients qui effectuent une requêtes passant par le proxy la requête sera :

source:server_name AND _exists_:ip_src

On définit la source des logs et on sélectionne les logs où l'information que l'on souhaite est présente. La prise en main de l'outil Grafana est simple et intuitif. J'ai également fait la liste des éléments qui pourrait être interessants à superviser.

1) Tableau avec toutes les informations (ip_src, connect_typ, port, url, count)
2) Graphique des requêtes en fonction du temps.
3) Site les plus consultés
4) Site deny les recherchés
5) Les ips des clients (toutes ou juste les plus fréquents)
6) Top des sites visités par une même ip
7) Les codes Squid (TCP_DENIED, TCP_TUNNEL, …)

J'ai pu après un peu d'expérimentation arriver à un résultat satisfaisant.

Figure 8: Dashboard grafana version 1

3.4.5 Antivirus et SSL (Secure Sockets Layer) Bumping

3.4.5.1 Pourquoi faire de l'antivirus et du SSL Bumping

Comme les requêtes issues du web passent par le proxy, on pourrait mettre en place un antivirus sur le contenu des requêtes pour augmenter la sécurité. Mais cela n'a pas de réel intérêt car c'est uniquement applicable à du HTTP (Hypertext Transfer Protocol) et donc un firewall pourrait également le faire. Mais avec un proxy explicite, il est possible de faire ce que l'on appelle du SSL bumping. Cela consiste à recevoir le message chiffré par le client, le déchiffrer, le chiffrer et l'envoyer au serveur de destination initial, ainsi entre le déchiffrement et le chiffrement il sera possible d'analyser le contenu de la requête pour y appliquer l'antivirus. Il faut savoir que cela s'appelle du SSL bumping mais cela fonctionne avec TLS (Transport Layer Security).

3.4.5.2 Fonctionnement

Pour fonctionner, il faut générer un certificat sur le serveur et le placer dans le répertoire de certificat du client. Pour générer le certificat, on peut utiliser openssl qui nous permet de le faire en quelques commandes. Par la suite, on modifie la directive squid qui concerne le port pour lui indiquer de faire du ssl bumping avec les certificats que l'on a créé, une acl pour autoriser le trafic déchiffré, ainsi que quelques lignes permettant de spécifier la façon dont fonctionne le ssl bumping. Désormais, si on regarde le fichier access.log on aura plus de "TCP_TUNNEL" spécifiant que la requête est chiffrée mais des "TCP_HIT" ou "TCP_MISS" signalant que l'information est ou n'est pas dans le cache du proxy.

Pour l'antivirus, il se découpe en trois entités, un serveur ICAP (Internet Content Adaptation Protocol) ici c-icap qui permet d'intégrer au serveur proxy des fonctionnalités de traitement sur le contenu des requêtes, clamAV qui est un antivirus open source qui sera chargé de détecter les menaces et en troisième squidclamv qui permet d'intégrer dans squid l'antivirus.

Désormais, si un virus est détecté par clamAV, il sera automatiquement bloqué au niveau du proxy.

3.4.6 Puppet

3.4.6.1 Fonctionnement Puppet

Puppet est un outil développé en ruby permettant de gérer la configuration de différents serveurs. Puppet va être utilisé pour l'automatisation de l'installation d'un serveur ainsi que son maintien au fil du temps. Il centralise ainsi la partie gestion dans un seul langage. Pour cette partie d'automatisation pas vraiment de choix sur la solution à utiliser car c'est celle utilisée dans l'entreprise.

Il fonctionne avec un serveur Master (maître) qui distribue les tâches de configuration aux clients à intervalle régulier. Ainsi si une modification est effectuée dans un fichier de configuration, puppet écrasera cette modification avec la configuration souhaitée. C'est grâce à ça que je n'ai pas eu comme expliqué précédemment à configurer l'envoi de log via syslog car toute les machines sont déjà configurées pour le faire.

Il faut donc configurer via puppet l'ensemble des étapes précédentes, heureusement il existe différents modules développés par la communauté. Pour squid, un module existe, il permet de modifier chaque aspect de squid comme on le souhaite, que ce soit les acl, le port, le ssl bumping et bien d'autres. Pour keepalived, il existe également un module permettant l'installation et la configuration. J'ai également modifié le comportement de "rsyslog" pour qu'il n'envoie pas les requêtes liées à keepalived ainsi je n'ai plus besoin d'un script python en supplément pour gérer cela.

Malheureusement, après avoir discuté avec la personne s'occupant de Puppet dans l'entreprise, j'ai compris qu'il ne serait pas possible de faire de ssl bump et donc de l'antivirus, car l'automatisation de la création et du maintien des certificats ne serait pas possible. Comme la partie ssl bumping et antivirus sont des éléments ajoutés au projet ce n'était pas primordial qu'ils soient présents.

De cette façon, avec l'ensemble des étapes on obtient pas simplement un proxy Squid mais un proxy Squid qui utilise et communique avec les différents composants qu'utilisent l'entreprise, ainsi il peut être déployé en production et être répliqué sans grand effort.

3.5 Retour sur le Projet

Ce premier projet durant mon stage m'a permis de découvrir comment fonctionnait l'entreprise, découvrir les différents outils. Il m'a permis de toucher à de nombreux outils, Squid, syslog, Graylog et Grafana. J'ai pu arriver à une solution fonctionnelle en peu de temps ce qui m'a laissé du temps pour les autres projets détaillés dans ce rapport. Le seul inconvénient est le manque de comparaison, en effet je devais assister à une présentation avec Olfeo, une solution commerciale de proxy, mais pour des raisons d'emploi du temps cela fut impossible.

4 Projet Rancid

4.1 Contexte du projet

Une fois le serveur proxy fonctionnel, une autre mission me fut confiée, trouver un moyen de versionner les différentes versions des équipements réseaux, switch et routeur. Il y avait tous les jours une sauvegarde automatique des configurations mais la nouvelle écrasait l'ancienne. Donc impossible en cas de problème de revenir à une configuration datant de plus de vingt-quatre heures.

4.2 Solution

La solution que j'ai explorée en premier est open source et s'appelle Rancid , son fonctionnement est assez simple, il va se connecter via ssh ou telnet sur l'équipement et récupérer la configuration actuelle via par exemple si c'est un équipement cisco "show running-config" et l'enregistre sur le serveur.

Pour qu'il puisse se connecter, il lui faut un compte avec de préférence uniquement la permission d'effectuer les commandes dont il a besoin. Sur cisco cela est faisable simplement avec les différents niveaux de privilèges. Une authentification via un compte ldap ou radius est également possible.

On va pouvoir définir différents groupes contenant les adresse ip ou le FQDN (Fully qualified domain name). Il peut y en avoir un seul contenant toutes vos machines, mais également des groupes par régions, par clients, ou par catégories d'équipements, ou autres.

Le fait de sauvegarder automatiquement différentes versions des configurations récupérées est par défaut fait via CVS (Concurrent versions system) mais il est possible également d'utiliser Git. Après avoir mis en place et testé les deux je recommande l'utilisation de git, car il est plus simple de mettre en place ce dernier et il existe des outils connus pour visualiser tout cela avec Github ou Gitlab par exemple. Alors que pour CVS, les scripts proposés ne sont plus forcément très bien maintenus ou du moins bien présentés dans les différents tutos présents sur internet ou sur le site de rancid. De plus CVS est désormais beaucoup moins populaire et utilisé que Git.

Lorsque l'on installe Rancid, il va automatiquement créer un fichier crontab, permettant de définir la fréquence d'activation de Rancid, dans mon cas je l'ai modifié pour qu'il passe tous les jours à vingt-trois heures. Et il indiquera ce qu'il a fait, donc lister les changements dans la configuration ou bien dire qu'il n'y a eu aucun changement.

Tout comme squid, j'ai utilisé puppet pour que la configuration soit facile à déployer et à maintenir, mais cette fois si aucun module utilisé, car l'ensemble des modules puppet pour rancid sont obsolètes car non maintenus. Mais malgré cela, grâce aux connaissances que j'ai acquises avec l'automatisation de Squid j'ai pu réussir sans grande difficulté.

4.3 Deuxième Solution

Même si Rancid répondait aux besoins, j'ai testé une deuxième solution, oxidized, également open-source. Ce projet a pour ambition de remplacer Rancid comme outil de sauvegarde de configuration. Même si le fonctionnement reste globalement le même.

On aura également une connexion via SSH sur les différents équipements réseaux. Un système de groupe permettant de créer des catégories si besoin. On renseigne les différents équipements dans un fichier CSV (Comma-separated values) contenant entre autres le nom, l'ip et le groupe.

Il peut également utiliser GIT pour versionner les configurations, mais là où il se démarque de Rancid, c'est qu'il possède directement une interface web, où l'on peut voir les différentes configurations ainsi que les différences entre les versions.

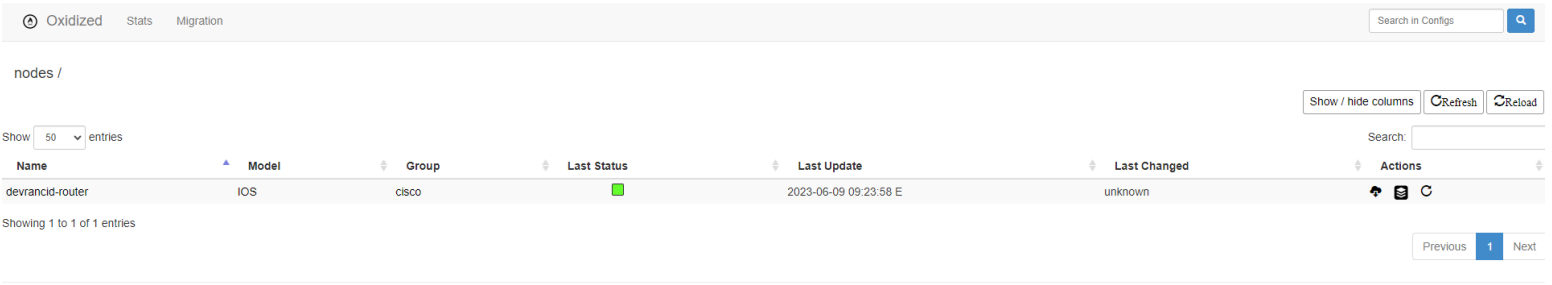

Figure 9 : oxidized web interface

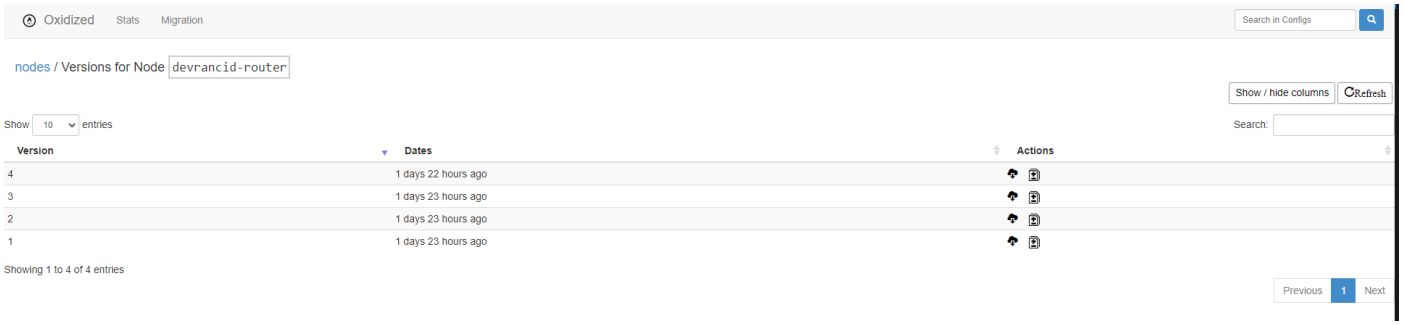

Figure 10: oxidized web interface des versions

Ensuite, il faut refaire la configuration avec Puppet, et tout comme avec Rancid il n'existe pas de module donc j'ai également dû faire sans. J'ai trouvé la configuration d'Oxidized un peu plus complexe que celle de Rancid, que ce soit avec ou sans Puppet. Notamment car il y a moins de documentation que pour Rancid.

4.4 Avis sur les deux solutions

Les deux produits répondent aux besoins, c'est-à-dire récupérer la configuration des équipements réseaux, la sauvegarder et également garder les différentes versions. De plus, les outils sont actuellement maintenus, il y a donc des mises à jour et des corrections de bugs. Ils sont également très similaires dans leur fonctionnement.

Pour ce qui est de la configuration j'ai trouvé que Rancid était plus simple et plus rapide à configurer, car comme expliqué précédemment il y a plus de documentation sur le sujet. Il faut aussi souligner que Rancid est compatible avec un plus grand nombre d'équipements et ce car il existe depuis plus

longtemps, même si dans mon cas, ayant fait des tests sur des équipements cisco je n'ai pas été confronté à des problèmes de compatibilité. Par contre, le fait d'avoir une interface web directement intégré rend oxidized beaucoup plus simple d'utilisation. Une fois installé, n'importe qui peut l'utiliser car l'interface est très intuitive.
C'est pour cela que je pense que Oxidized est un choix légèrement meilleur que Rancid. Même si encore une fois les deux solutions restent très similaires.

5 Red Hat Single Sign-On

5.1 Contexte du projet

La méthode d'authentification la plus courante est le mot de passe, mais avec le temps l'utilisation de ce dernier est de plus en plus contraignante pour les utilisateurs si l'on veut que ce soit sécurisé. La taille des mots de passe conseillée augmente, il faut avoir un mot de passe avec des caractères spéciaux, le changer régulièrement. C'est pour cela qu'il y a de plus en plus de solutions d'authentification sans mot de passe, comme l'OTP, One-Time Password, qui signifie un mot de passe à usage unique, cette méthode est très répandu comme lors d'authentification à multiples facteurs, donc en complément du mot de passe. Mais elle peut également être utilisée seule.

La solution que j'ai testée est Red Hat Single Sign-On, un produit de Red Hat qui fonctionne en utilisant KeyCloak un projet open source initialement développé par JBoss mais qui depuis 2018 est géré par Red Hat. En plus de permettre l'authentification sans mot de passe, elle devrait permettre que lorsque l'on se connecte à des applications utilisant le serveur Red Hat SSO, il suffira de se connecter une seule fois pour être connecté à tous les services.

Le but de tester cette solution est de voir si elle répondait à un cahier des charges que l'on m'a donné. Parmi les points importants on retrouve, la facilité de prise en main, s'authentifier sans mots de passe, la documentation, connexion à LDAP, compatibilité avec Radius et bien d'autres. Il faut donc prendre en main l'outil pour pouvoir fournir des informations pour les comparer à des solutions similaires.

5.2 Prise en main

Pour commencer mes tests, j'ai installé Red Hat SSO sur une machine virtuelle avec comme système d'exploitation Red Hat, une fois installée, je peux y accéder depuis l'interface web permettant de configurer la majorité des options. Le premier concept est celui des royaumes, il y a le master et ensuite on peut en créer autant que l'on souhaite et ils seront tous plus ou moins indépendants des autres.

Pour que des utilisateurs puissent s'authentifier il leur faut un compte, on peut donc soit créer chaque compte à la main via l'onglet "Users", ou alors utiliser l'onglet "User Federation" qui nous permettra de récupérer des utilisateurs via par exemple un annuaire LDAP. Dans mon cas, l'ensemble des utilisateurs sont présents sur ce type d'annuaire, je peux donc récupérer les noms, prénoms, email, groupe ou tout autre information présents dans le LDAP. Un autre avantage est que ainsi, les utilisateurs pourront se connecter avec leur nom d'utilisateur et mot de passe de LDAP.

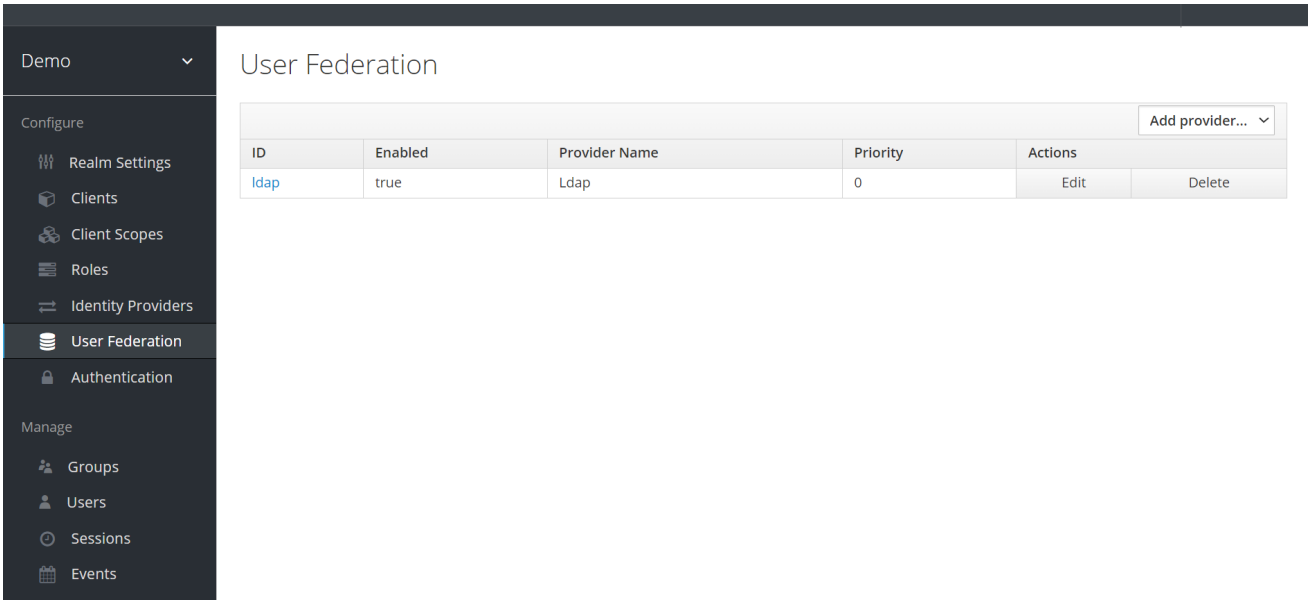

Figure 11 : User Federation

Un des objectifs, est d'utiliser OTP à la place des mots de passe, Red Hat SSO permet d'utiliser ce mécanisme comme unique méthode ou bien en parallèle avec d'autres. L'onglet Authentication permet de créer et de modifier des règles d'authentification. La règle que je devais créer devait répondre au besoin suivant, comme les utilisateurs proviennent de LDAP, ils n'ont pas d'OTP configuré, il faut donc que tant que l'OTP n'est pas configuré, alors ils peuvent utiliser leur mot de passe mais une fois que l'OTP est enregistré alors la seul méthode d'authentification proposé est OTP. Pour ce faire, il suffit d'utiliser ce qu'ils appellent une exécution, qui vérifie que le type d'authentification que l'on souhaite est bien configuré pour le client, si c'est le cas il aura un champ pour entrer le mot de passe à usage unique et sinon un champ pour son mot de passe classique sera affiché.

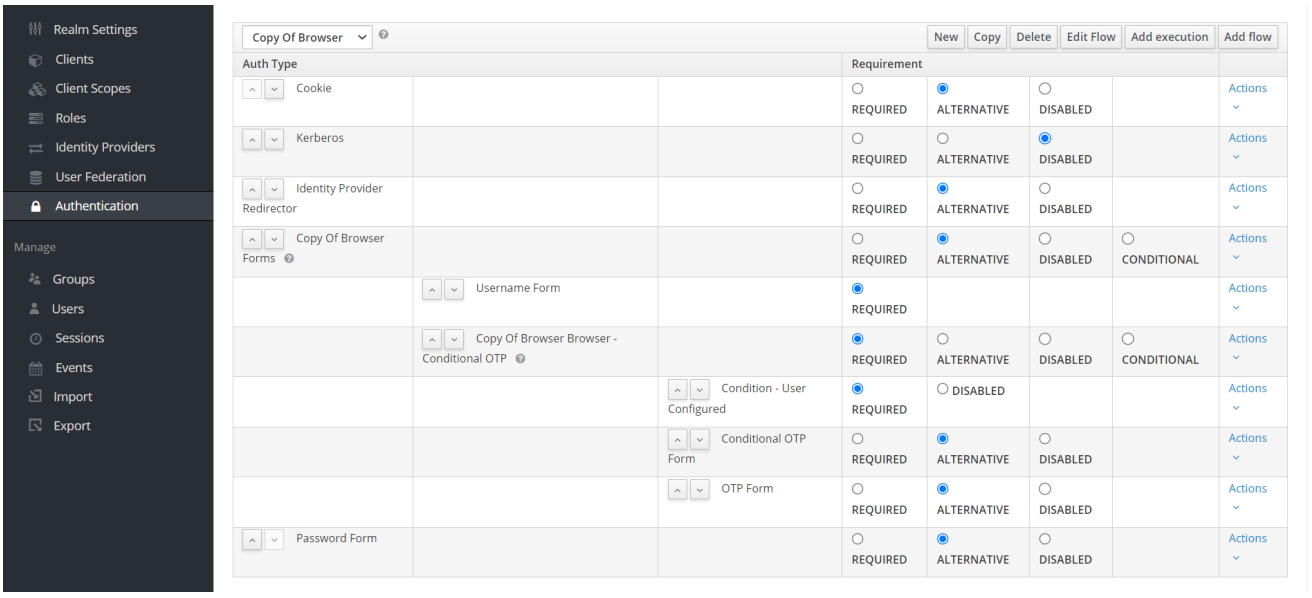

Figure 12 : Authentification avec OTP

Il a deux concepts importants, celui de IDP, IDentity Provider, correspondant à l'entité qui gérera l'authentification à la place des applications, ici le serveur Red Hat SSO, et celui de client, qui sera l'application utilisant le service de l'identity provider. Pour fonctionner, il existe deux standards principalement, SAML, Security assertion markup language, qui est basé sur le langage XML, permettant le transfert des données d'authentification. Le deuxième est Openid Connect, plus récent

que SAML, il utilise un système de jeton de type JSON Web Token (JWT) appelé jeton d'identification.

Pour qu'une application utilise Red Hat SSO, il faudra donc un client que l'on configure via l'interface web dans l'onglet "Client". Ensuite, on devra sélectionner la norme entre SAML et Openid Connect, ce choix se fera en fonction de la compatibilité avec les applications, car pas toutes les applications utilisent les deux. La configuration du client changera en fonction de la norme choisie, mais le premier champ est identique, il s'agit du Client ID, c'est à dire l'identifiant que devra utiliser une application voulant utiliser le serveur Red Hat SSO, pour SAML il n'y a pas beaucoup plus d'étapes pour avoir une configuration client qui fonctionne, pour Openid il faudra choisir le type d'accès, dans mon cas j'ai utilisé "confidential" c'est-à-dire qu'il faut un mot de passe ou un certificat pour connecter une application à l'IDP.

Comme dit précédemment, il est possible de transmettre des informations sur l'utilisateur, comme les rôles ou groupes pour une gestion simplifiée des accès. Cela se fait sur le Red Hat SSO via les "Mappers". Il est possible de définir quelles informations seront ou non transmises à l'application.

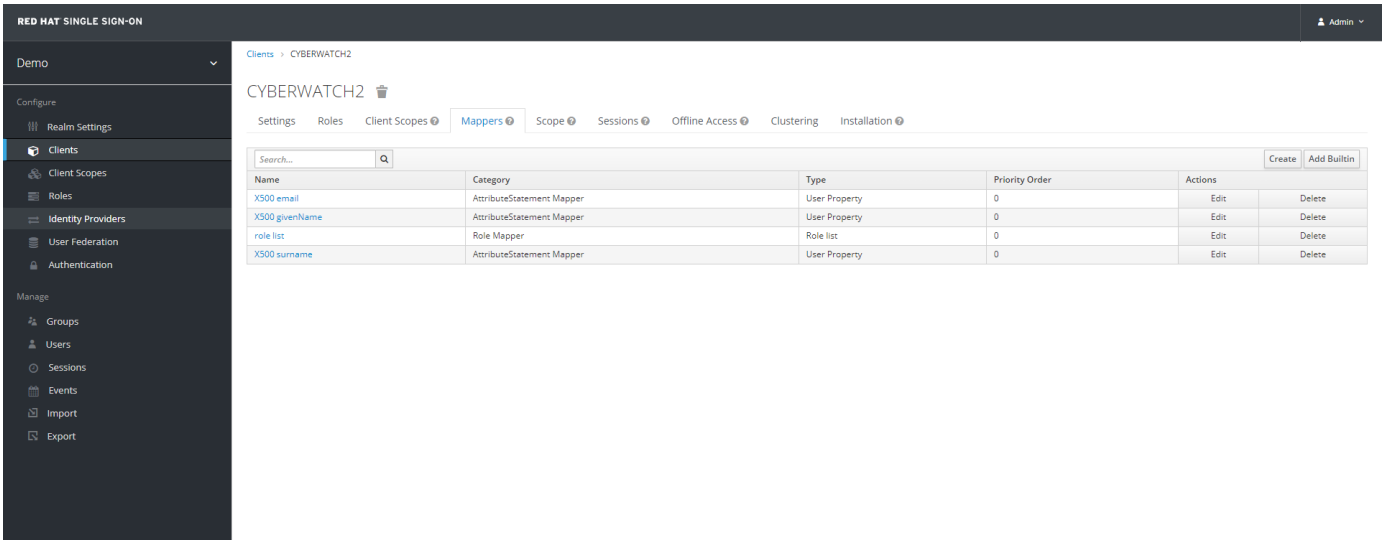

Figure 13 : Mappers SAML

Pour l'application, il faudra spécifier le nom de l'attribut que l'on souhaite utiliser, voici un exemple de configuration des attributs avec l'application Cyberwatch, qui m'a servi à faire des tests pour comprendre le fonctionnement.

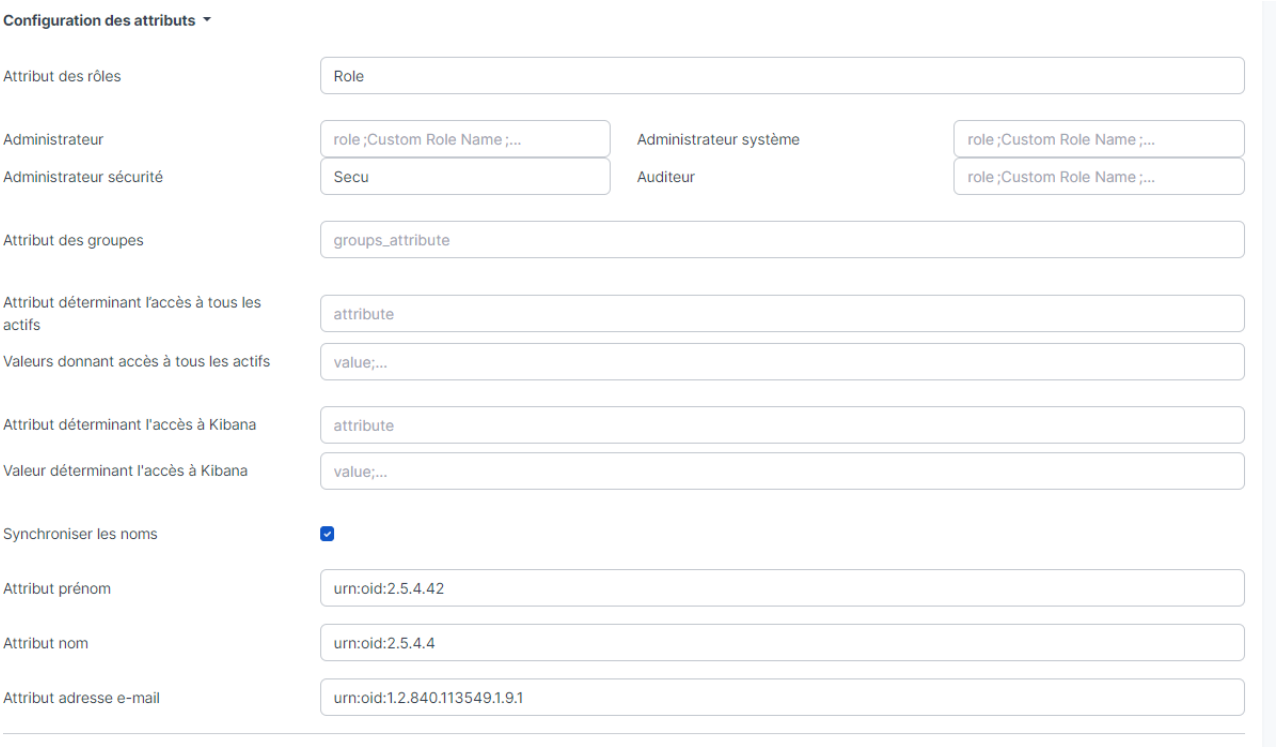

Figure 14 : Configuration des attributs sur Cyberwatch

Avec cette configuration, les utilisateurs ayant le rôle “Secu” sur Red Hat SSO, obtiendront automatiquement le rôle Administrateur sécurité, à savoir que par défaut le rôle donné est Auditeur. Pouvoir utiliser directement les rôles RedHat SSO sur chaque application permet de gagner du temps en n'ayant pas à le faire pour chaque utilisateur à la main. De plus, Cyberwatch remplira utilisera les attributs prénom, nom et adresse e-mail renseignés. Pour obtenir le nom de l’attribut pour ses derniers il suffit tout comme pour le rôle de regarder le SAML Attribute Name que l’on retrouve dans l’onglet Mappers.

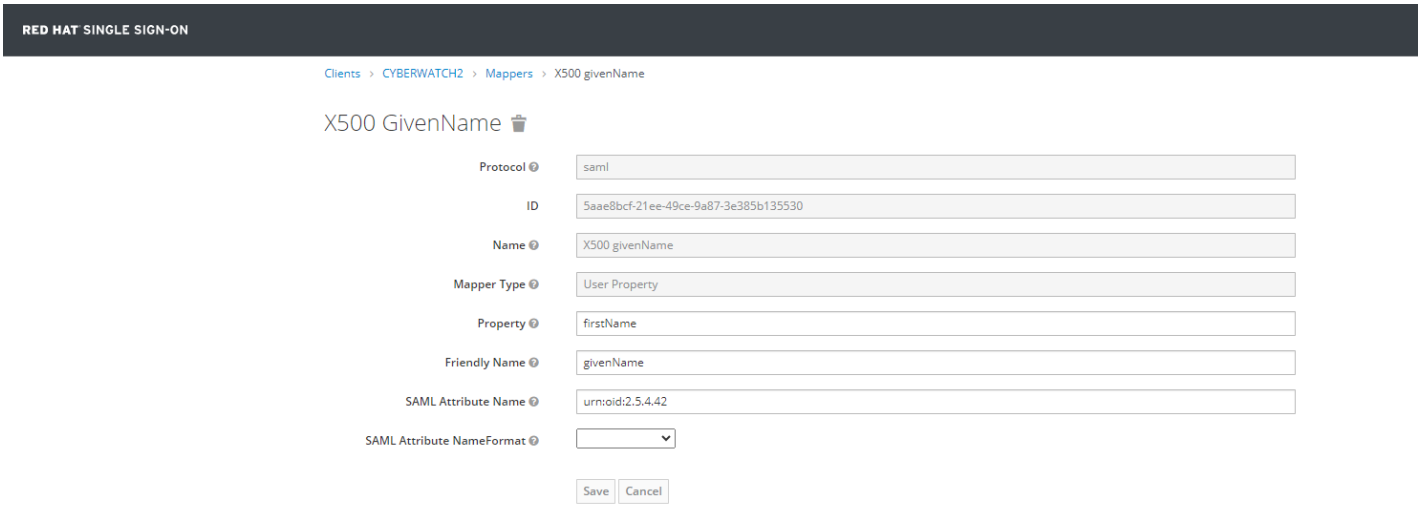

Figure 15 : Mappers Prénom

Pour Openid, il y a aussi des “Mappers” mais comme on utilise un token, le format sera différent, mais le fonctionnement reste le même, les noms des attributs sont les seuls à changer.

Il va être aussi possible de faire en sorte que seuls certains utilisateurs puissent accéder à Cyberwatch, car si rien n’est modifié l’ensemble des utilisateurs présents dans Red Hat SSO auront accès aux différentes applications connectées. Pour changer cela, il faut se rendre dans l’onglet “Authentication”, il faut modifier la règle précédente qui définissait l’utilisation de l’OTP, en y ajoutant des conditions qui vont vérifier après que l’utilisateur ait entré ses identifiants, s’il est ou non autorisé et ce en fonction de ses rôles, mais cela peut-être fait en fonction de n’importe quels attributs.

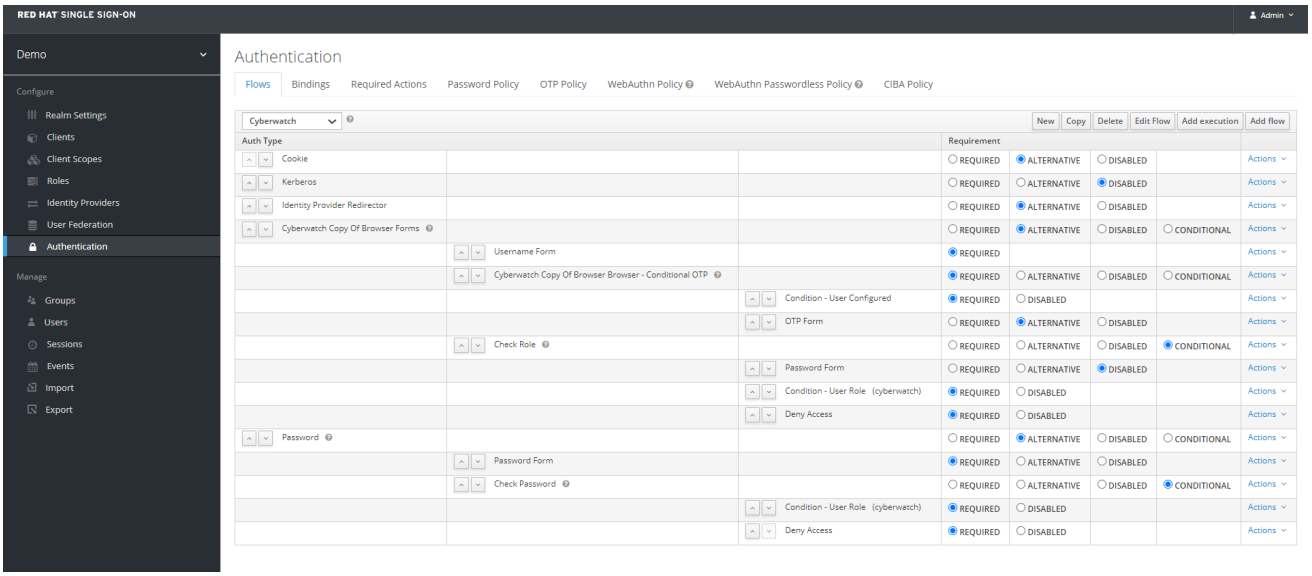

Figure 16 : Accès en fonction des rôles

5.3 Radius dans Keycloak

Avec Red Hat SSO, il est possible de configurer des clients SAML et Openid Connect, mais un besoin était de pouvoir l’utiliser pour des équipements réseaux, sauf que ses derniers n’utilisent ni SAML, ni Openid Connect, une solution est donc d’utiliser le protocole Radius, qui est un protocole de type client-serveur permettant de centraliser les données nécessaire à l’authentification. Il existe différents types de serveur, comme un des plus connus FreeRadius, mais il existe également un plugin permettant à Keycloak de pouvoir communiquer à l’aide de ce protocole.

Le nom du plugin est Keycloak-Radius-Plugin, et tout comme Keycloak sa licence nous permet de l’utiliser librement. Après plusieurs tentatives, je n’ai pas trouvé le moyen de l’intégrer à RedHat SSO, j’ai donc choisi d’avoir un serveur RedHat SSO et un serveur Keycloak-Radius, mais l’objectif étant de minimiser au maximum la configuration de la part des utilisateurs, il faut faire en sorte que les informations saisies dans le premier serveur soit partagées avec le deuxième. Pour l’utiliser avec des switchs ou routeurs, il n'était pas possible d’utiliser l’onglet Identity Provider, car normalement fait pour les applications Web, la solution était donc que les deux serveurs partage la même base de données. Par défaut, ils utilisent des bases de données relationnelles H2, mais il n’est pas conseillé d’utiliser cette dernière, c'est pour cela que j’ai utilisé une base PSQL, ainsi les deux serveurs ont accès aux mêmes utilisateurs, clients ou realms.

On a un client à configurer et ensuite il est possible de se connecter sur un équipement réseau relié à Keycloak via un compte présent importé depuis un annuaire LDAP. Et la fonctionnalité la plus

intéressante et qu'il est possible d'utiliser l'OTP, s'il est configuré, l'utilisateur aura simplement à renseigner son password avec à la suite l'OTP.

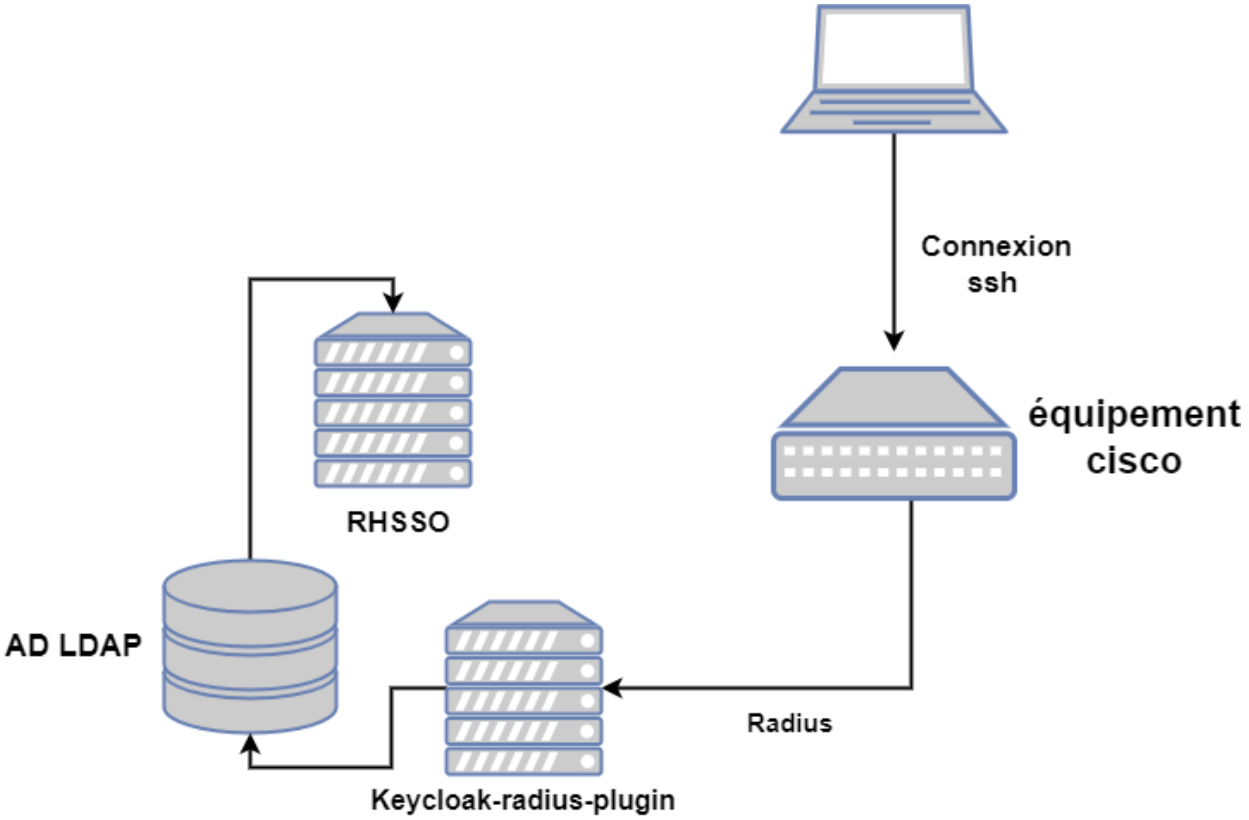

Figure 17 : Architecture du SSO

5.4 Retour sur Red Hat SSO

Ce projet est pour moi le plus compliqué des trois, en effet, il aborde des notions de sécurité et d'authentification avec SAML et OPENID Connect, d'une certaine complexité. Mais ce fut un excellent challenge, et l'objectif était simplement de tester la solution pour savoir si elle répondait à des besoins précis.

6 Conclusion

Ce stage m’a permis de découvrir le monde de l’entreprise, que ce soit la mise en application de mes connaissances acquises au cours de mes deux premières années de BUT mais également découvrir l’aspect social d’une entreprise. J’ai pu consolider mes acquis en réseau, administration système et cybersécurité grâce à ses trois projets abordant des domaines très divers.

J’ai pu découvrir de nouveaux logiciels, Squid, Rancid, Oxidized et RedHat SSO. Mais également des solutions utilisées au sein de l’entreprise comme Graylog, Grafana et Puppet. La gestion de projet dans une entreprise était également une découverte car les enjeux sont bien différents des projets que j’ai pu faire au cours de ma scolarité. Cette expérience aura enrichi ma compréhension des enjeux liés à la gestion de l'infrastructure réseau et renforcé mes compétences techniques dans ces domaines spécifiques.

Les sujets qui m’ont été confiés m’ont permis de me fixer de nouveaux objectifs et ainsi orienté mon projet professionnel. N’ayant jusqu'à présent pas encore trouvé précisément le domaine qui m’attire le plus, ce stage m’a permis de découvrir une attirance pour la conception de solutions informatiques.

7 Remerciements

Je tiens tout d'abord à remercier Fabien MULLER de m'avoir accueilli au sein du service infrastructure pendant ces 10 semaines de stages.

Je remercie également mon tuteur de stage, Guillaume GOUPIL, qui a su me trouver des sujets intéressants et challengeants tout au long de mon stage, ainsi que de m'avoir accordé du temps pour répondre à mes questions.

Enfin je remercie plus largement l'ensemble des personnes du pôle infrastructure pour leur accueil et m'avoir formé tout au long de mon stage.

8 Glossaire

BUT, Bachelor Universitaire de Technologie

GNU GPL, Licence publique générale GNU

VIP, virtual IP address

SSL,Secure Sockets Layer

HTTP, Hypertext Transfer Protocol

TLS, Transport Layer Security

ICAP, Internet Content Adaptation Protocol

CVS, Concurrent versions system

FQDN,Fully qualified domain name

TFTP, Trivial File Transfer Protocol

CSV, Comma-separated values

OTP, One-Time Password

SSO, Single Sign-On

IDP, Identity Provider

SAML,Security assertion markup language

XML, Extensible Markup Language

JWT, JSON Web Token

Mappers, Peut être traduit de l'anglais par faire correspondre, établir une correspondance avec.

PSQL, PostgreSQL