Table des matières

[Introduction 2](#_Toc488354523)

[Le Centre spatial guyanais (CSG) et Telespazio Guyane 3](#_Toc488354524)

[Google/capirca une solution pour une gestion organisée des demandes de flux 4](#_Toc488354525)

[La gestion des demandes de flux utilisateurs 4](#_Toc488354526)

[Google/capirca 5](#_Toc488354527)

[Google/capirca appliqué au PRINC 7](#_Toc488354528)

[Gogs (Go git service) 9](#_Toc488354529)

[La maquette 10](#_Toc488354530)

[REDEX (REtour D’EXperience) 11](#_Toc488354531)

[Conclusion 12](#_Toc488354532)

# Introduction

Ce stage de 4éme année d’école d’ingénieur, d’une durée de deux mois, a consisté à mettre en place CAPIRCA pour un firewall géré par Telespazio pour le Centre Spatial Guyanais (CSG).

Il s’agit d’un logiciel développé par GOOGLE pour automatiser la création des ACLs (access-list).

Ce rapport présente le travail réalisé durant mon stage au sein de Telespazio Guyane, dans le centre technique du CSG. Il s’est déroulé du 6 Juin au 6 août 2017.

Ce projet, qui s’inscrit dans le domaine de la sécurité et se situe au sein d’un lieu ultrasensible demandant de multiples procédures sécuritaires, s’intègre pleinement dans ma formation dont le cœur est la sécurité des systèmes d’information.

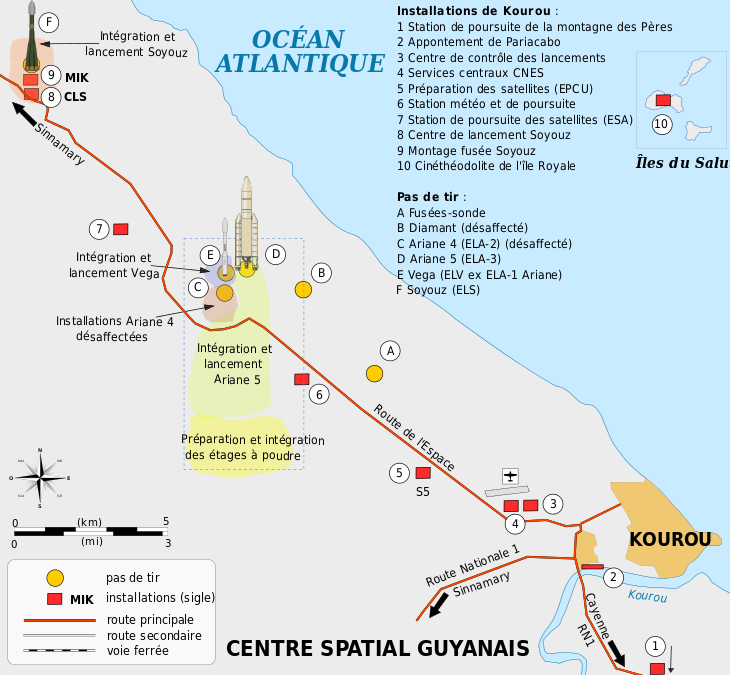
Ce stage m’a permis d’expérimenter de façon pratique la réalité du métier d’ingénieur dans ce secteur d’activité.

Le but de ce résumé d’activité n’est pas de faire une description exhaustive de tous les aspects techniques mis en œuvre, mais de présenter de manière claire et synthétique les différentes parties techniques.

Dans un premier temps est présentée l’entreprise d’accueil, Telespazio France. Ensuite seront exposées les différentes tâches réalisées au cours de ces 2 mois. Pour finir, je reviendrai sur les apports de ce stage dans ma formation, et conclurai.

# Le Centre Spatial Guyanais (CSG) et Telespazio Guyane

Le Centre National d’Etudes Spatiales (CNES) utilise à ses débuts des installations militaires implantées à Hammaguir dans le désert algérien. A l’indépendance de l’Algérie, un nouveau site doit être trouvé. Sur les 14 emplacements pressentis à travers le monde, la Guyane française arrive largement en tête car elle offre des conditions de lancement optimales. Grâce à sa large ouverture sur l’océan les lancements se font avec un maximum de sécurité tant vers l’est que vers le nord. En lançant vers l’est les lanceurs bénéficient à plein de la vitesse de la rotation de la Terre, plus importante au niveau de l’équateur. De par la proximité de l’équateur, les satellites géostationnaires minimisent les manœuvres de correction de trajectoire, économisant ainsi du carburant et augmentant notablement leur durée de vie.



Pour mener à bien chacune de ses activités le Centre Spatial Guyanais est composé d’un consortium d’entreprises provenant de milieux divers. On peut citer par exemple Videlio qui s’occupe des prises de vues, ou encore Europropulsion qui se charge de la production du propergol solide (carburant). Toutes ces entreprises, parmi lesquelles Telespazio, sont réunies sur la base spatiale, dénommée CSG (Centre Spatial Guyanais), et placées sous la tutelle du CNES (Centre National d’Etudes Spatiales).

Telespazio France est organisé autour de trois métiers d'excellence :

* Les Systèmes Satellitaires et Opérations, à savoir l'exploitation des moyens et systèmes spatiaux des clients (tels que les systèmes européen EGNOS, la base spatiale de Kourou ou les moyens de télécommunications spatiales de la Défense Française...) et du développement des applications et solutions associées.
* Les Télécommunications Spatiales proposant un large portefeuille de solutions et services de connectivité.
* La Géo-Information fournissant une offre unique en imagerie spatiale radar et optique, produits et services d’observation et de surveillance de la terre et des océans.

En Guyane Française, Telespazio Guyane concentre la plupart de ses activités sur la base spatiale. Sur les 3 métiers d’excellence, on retrouve surtout les deux premiers. Une des parties de la base spatiale est appelé centre technique (CT), chaque bâtiment portant le nom d’un astre, et regroupant un service (généralement une société). Personnellement j’ai effectué mon stage dans le bâtiment Mercure, dans le service « *Groupe Réseaux, Commutation, Configuration et Supports »*. Telespazio Guyane est constitué de 2 services, le deuxième étant le service qui se charge de tout ce qui est télémesure et antenne de poursuite lanceur.

Chaque accès au centre spatial est réglementé, pour entrer dans les grandes zones (CT, CDL3 …) il faut un badge, chaque couleur correspondant à une zone. Mais même avec une couleur on ne peut pas aller partout, il faut faire des demandes justifiées (passant par le responsable du service) pour que certains accès à des portes ou encore des bâtiments soient activés. Pour information, le bleu correspond au CT et le rouge au CDL3 (Centre de Lancement Ariane 5).



Figure

# Google/capirca : une solution pour une gestion organisée des demandes de flux

## La gestion des demandes de flux utilisateurs

# description du réseaux REMUS demmandé demain comment cela ce passe.

Le firewall PRINC centralise les règles d’accès des différetns pôles de la Base Spatiale ‘centre technique, CDL3… ). Il est configuré en « deny all », c’est-à-dire que, à la base, tout accès est rejeté. Toute personne, ou tout service qui désire l’ouverture d’un accès, en fait la demande à la SSI. La SSI transmet à Telespazio cette demande au travers d’une « demande d’ouverture de flux utilisateur ». Cette demande se présente sous la forme d’un fichier constitué d’une référence, des nom et prénom du demandeur etc… Mais surtout elle fait mention des différents accès à ouvrir. Sont indiqués les adresses IP, les ports, les services et le sujet de cette ouverture.

Une fois cette demande reçue par Telespazio elle est transmise à une personne que l’on nommera, dans le cadre de ce résumé d’activité, gestionnaire. Ce gestionnaire se chargera alors d’ouvrir les différents accès en ajoutant les ACL correspondantes dans le firewall PRINC.

Pour réaliser cela le gestionnaire doit ce placer au niveau du réseau privé du PRINC et se connecter au manager. Le manager n’est autre que le ciscoa asdm, en effet le PRINC et un firewall de type ciscoasa.

Cette manipulation prend du temps, notamment car elle ne peut être réalisé de n’importe où, elle demande en effet d’être physiquement présent derrière une machine précise qu’est le manager. Mais aussi parce que l’ajout des règles au niveau du manager est long et fastidieux, surtout quand on sait qu’une demande d’ouverture de flux demande en moyenne l’ajout d’une dizaine d’ACLs.

Le manager permet l’ajout d’ACLs mais aussi d’objets. Un objet peut être l’ip d’un hôte, d’un réseau, un service (par exemple : https), une liste de protocole … ces objets correspondent bien entendus à des objets de type ciscoasa. Ils apparaissent donc dans la running config.

Or il n’est pa toujours créé d’objet pour chaque ip, ainsi souvent on ce retrouve avec des doublons au sein du PRINC, ce qu’il faut le dire ne pose pas de problème en termes de fonctionnement. Prenons un exemple simple : le gestionnaire A traite une demmande de flux à l’aide d’objets créé ou près existants. Le gestionnaire B fait de même avec une autre demmande d’ouverture de flux. Il s’avère qu’entre ces deux demmande 4 règles était similaire. Etant donné que leur méthode d’ajout et différente et qu’il n’y a aucun système de détection, il y a doublon.

Il faut ajouter concernant la technique de gestion actuelle au travers du manager, que cette dernière ne permet pas de garder un visuel sur les demandes de flux traitées. En d’autres, termes on ne peut pratiquement pas retrouver une demande de flux dans la configuration du PRINC avec ou sans le manager, après traitement de cette dernière.

Sinon outre tous ses aspects positifs tels qu’une visualisation amélioré des ACLs, sans le manager ( en oubliant la cli) il n’y plus de moyens « facilité » de configuration du PRINC.

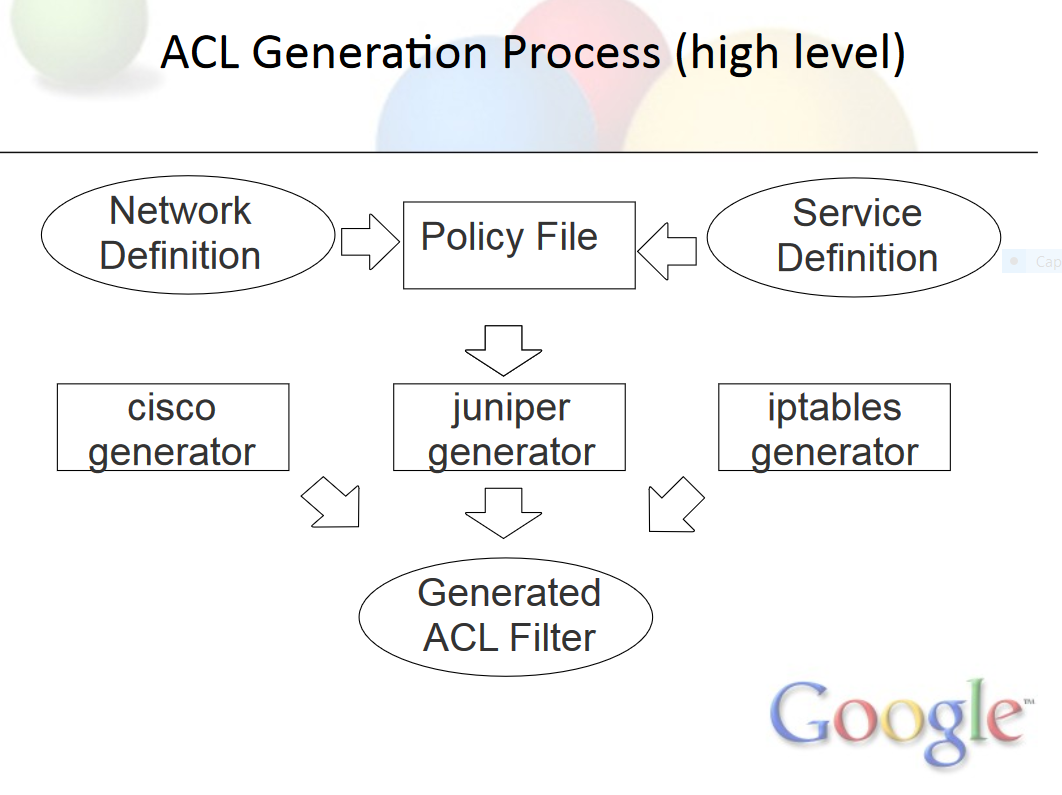
C’est dans ce cadre que l’on introduira google/capirca.

## Google/capirca

Pour pouvoir accélérer l’intégration des demandes de flux et de faciliter leur regroupement, on m’a demandé d’introduire google/capirca dans leur gestion.

Capirca est un outil conçu pour utiliser des définitions communes des réseaux, des services et des fichiers de politiques de haut niveau pour faciliter le développement et la manipulation des listes de contrôle d'accès au réseau (ACL) pour diverses plates-formes. Il a été développé par Google pour un usage interne et est désormais open source, récupérable sur Github.

Le gestionnaire retranscrit en language de haut niveau capirca les différentes règles exprimès dans le fichier de demmande d’ouverture de flux. Il créé ce que l’on appelle le policy file. Le policy ne prend que des objets (ip, service …) ces derniers doivent être inscrit préaleblemet dans des fichiers .svc pour les services et .net pour les ips, réseaux. Suffit ensuite de lancer le script python aclgen.py présent dans le répertoire principale de Capirca. On obtient en sortie les ACLs désiré après que Capirca et fait appelle au différent générateur sélectionné. Le seul qui nous intéresse ici étant le ciscoasa, on obtient donc qu’un seul fichier de type .asa.



Figure

Un fichier de stratégie (policy file) consiste en un ou plusieurs filtres, chaque filtre contenant un ou plusieurs termes. Chaque terme spécifie les informations de base du filtre réseau, telles que les adresses, les ports, les protocoles et les actions.

Un fichier de stratégie comprend une ou plusieurs sections d'en-tête, chaque section d'en-tête étant suivie d'un ou plusieurs terms. Une section d'en-tête est généralement utilisée pour spécifier un filtre pour une direction donnée

De plus, le langage de la stratégie prend en charge les "fichiers inclus" qui injectent le texte du fichier inclus dans la politique à l'emplacement spécifié.

Chaque filtre est identifié par une section d'en-tête. La section d'en-tête sert à définir le type de filtre, un descripteur ou un nom, une direction (le cas échéant) et un format (ipv4 / ipv6).

Par exemple, l'en-tête simple suivant définit un filtre qui peut générer des résultats pour les formats cisco, juniper, iptables ,ciscoasa ….

header {

comment:: "Example header for juniper and iptables filter."

target:: juniper edge-filter

target:: speedway INPUT

target:: iptables INPUT

target:: cisco edge-filter

target:: ciscoasa edge-filter

}

Chaque fichier de sortie prend le nom de la targette spécifié avec l’extension qui est fonction du type de targette (ex : edge-filter.asa). Sauf, configuration différente des paramètres dans le scrypt aclgen.py, les filtres de sorties généré se trouve retrouve dans capirca/filters. Les fichiers de haut-niveau doivent quant à eux se trouver dans capirca/policies/pol.

Les termes définissent les règles de contrôle d'accès dans un filtre. Une fois que le filtre est défini dans les sections d'en-tête, il est suivi d'un ou plusieurs termes. Les termes sont entre parenthèses et utilisent des mots-clés pour spécifier la fonctionnalité d'un contrôle d'accès spécifique.

Une section de terme commence par le mot-clé, suivi d'un nom de terme. Des crochets d'ouverture et de fermeture suivent, y compris les mots-clés et les jetons pour définir l'appariement et l'action du terme de contrôle d'accès.

Les mots-clés se divisent en deux catégories, ceux-ci doivent être pris en charge par tous les générateurs de sortie et ceux qui sont éventuellement pris en charge par chaque générateur. Les mots clés facultatifs sont destinés à offrir une flexibilité supplémentaire lors de l'élaboration de stratégies sur une plate-forme cible unique.

term permit-to-web-servers {

destination-address:: WEB\_SERVERS

destination-port:: http

protocol:: tcp

log :: syslog

action:: accept

}

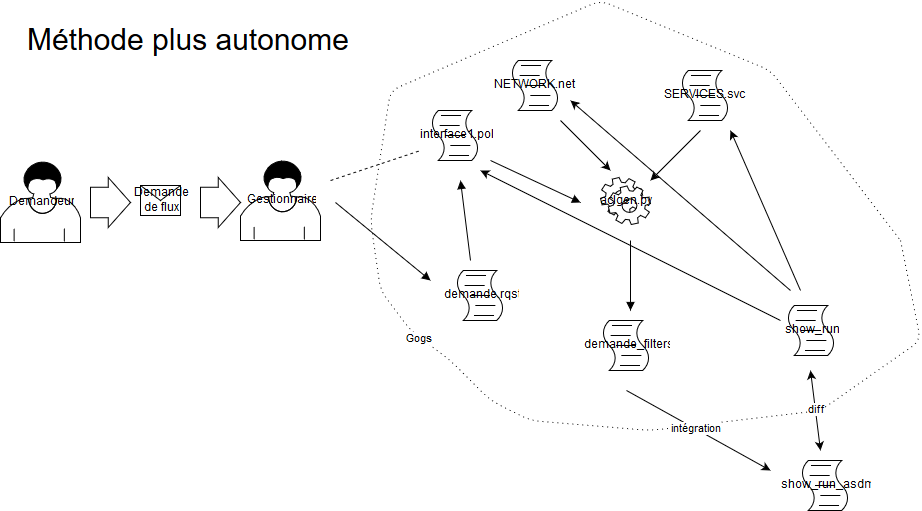
Comme soulevé plus haut le script de génération de Capirca aclgen.py est configurable, c’est-à-dire que l’on peut spécifier le répertoire des fichiers de haut-niveau et le répertoire de sortie ou seront stocké les filtres, d’autres paramètres tels que « shader » peuvent être activé. En activant « shader » et « verbose » permet d’avertir le gestionnaire de l’existence de doublon au sein d’un même fichier .pol.

Nous allons voire à présent la solution proposée pour l’intégration de Capirca à la gestion des ACLs pour le PRINC.

## Google/capirca appliqué au PRINC

Capirca ne peut être intégré directement tel quel dans le cadre du PRINC. Le but n’étant pas de repartir à zéro, mais d’intégré Capirca au système existant. Le système existant (évoqué en introduction) est un manager (cisco asdm). Toute une première partie du travail à donc était de savoir comment intégrer.

Plusieurs solutions se proposé, la première qui paraissait sur le moment la plus logique, était de récuperer par scripte toutes les access-liste existante de la running-config du PRINC et les récreer en langage de haut niveau capirca. Plus exactement un fichier par nom d’access-list, en effet les access-list sont nommer par interfaces. Interface qui correspond à des réseaux du centres spatiaul. Cette solution proposé c’est vite arrivé imparfaite. Tout d’abord malgrè le fait que le langage de haut-niveau de Capirca est bien plus lisible que du « bas niveau » ASA, une liste de plusieurs centaines de terms et loin d’être aussi lisible qu’une interface cisco asdm. Ensuite elle correspondrait à un « court-circuit » du système ASDM ce qui n’est absolument pas le but désiré.



Figure

La seconde solution que j’ai proposée et mise en place, correspond donc à une méthode dite collaborative. Ici, seuls les objets de type network et service sont récupéré à l’aide de script python depuis la running-config. En recevant une demmande d’ouverture de flux le gestionnaire créer simplement le fichier .pol (dans le dossier capirca/policies/pol) avec le header contenant une target avec pour nom l’interface de destination souhaitait et les différents terms, et ayant créer les objets nécessaire, ou réutiliser si importer.

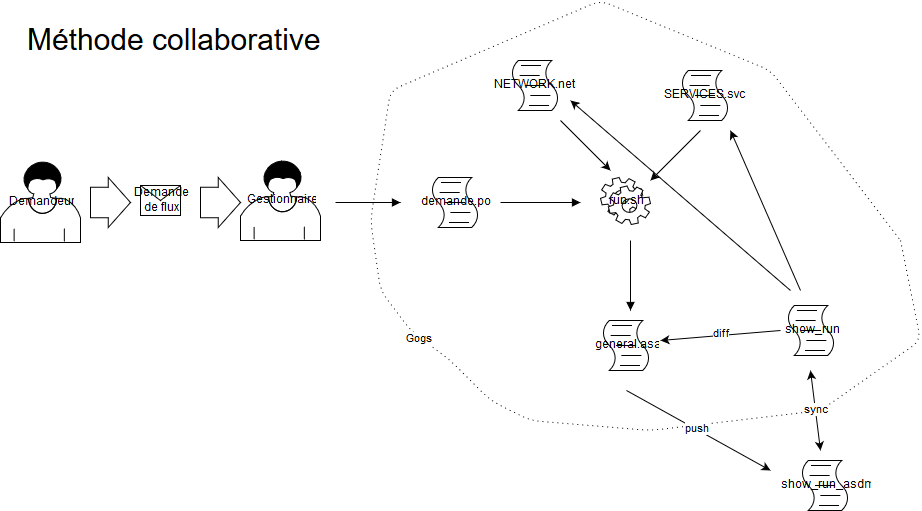
Le fichier .pol à comme nom la référence de la demande et dans le header plusieurs commentaire sont ajouter avec le nom et prénom du demandeur, le nom du gestionnaire … ect. Dans capirca un commentaire se note « comment :: ». Le nom des terms correpondent au nom du service sur le quel le terms agit (ex : WEB), si il y a plusieurs terms qui agissent sur le même service ils sont alors numérotés (ex : WEB1, WEB2, WEB3).

Ainsi chaque demmande d’ouverture de flux qui se présente sous la forme d’un fichier et qui détiens une référence ce retrouve dans Capirca dans un fichier, cela permet de retrouver, de modifier, de gérer une demande facilement. Le manager est utilisé quant à lui pour la gestion des anciennes règles, leur suppression, modification.

Or cela créer des problèmes de concurrences. Imaginons qu’un gestionnaire créé une règle sur la manager, et q’un deuxième créé une demmande.pol qui contient une règle similaire, il y a alors un doublon. Or l’un des buts de l’utilisation de Capirca et d’amélioré la propreté de la gestion du PRINC. Pour cela il à donc fallut créé des scripts python qui se basant sur la running-config permettent d’éviter la major partie des doublons. Cette gestion et intégré dans Capirca pour Telespazion dans le dossier diff. « Diff » permet d’évité les doublons entre Capirca et le manager mais aussi à l’intérieur d’une même demande.pol grâce aus « shader » de Capirca ainsi qu’entre chaque demmande.pol.

Le deuxième ajout majeur à Capirca pour Telespazio et le « sync ». « sync » et un ensemble de script, présent dans le dossier « sync » de Capirca pour Telespazio, permettant la syncronisation avec le PRINC, plus exactement sa running-config. En effet, pour le bon fonctionnement de deux « diff » il faut qu’à chaque compilation à l’aide de run.sh (script bash gérant diff et sync) la dernière running-config soit chargé.

Enfin chaque intégration dans la running config et faite à l’aide de « push ». Push est aussi dévellopé à l’aide la librairie Netmiko de Python.



Figure

Jusqu’à présent les fluxs peuvent être rapidement ouverts et bien rangé sous la forme de demmande.pol (un fichier par demmande). Jusqu’à présent plusieurs personnes ne peuvent pas travailler de manière collaborative, la partie suivante traitera de l’ajout de cette possibilité au travers de Gogs.

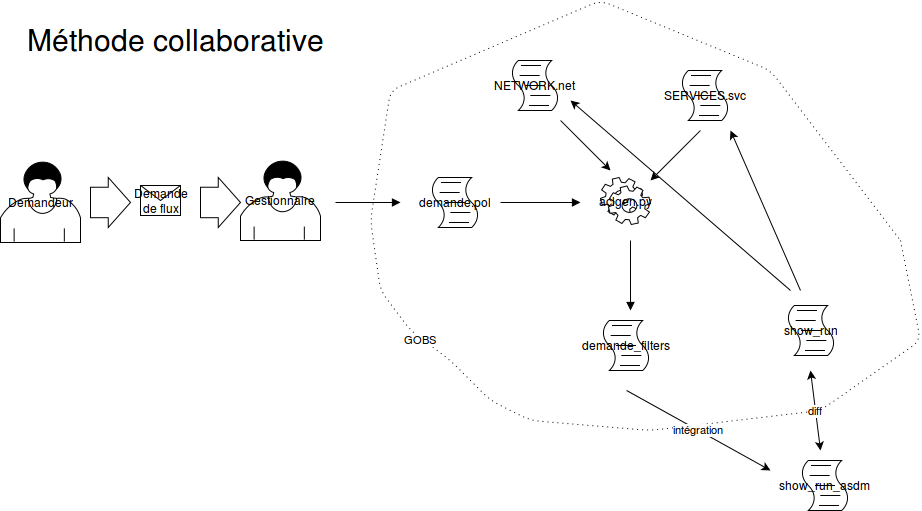
## Gogs (Go git service)

Gogs permet de créer et configurer un service Git auto-hébergé. Avec Go, cela peut se faire avec une distribution binaire indépendante dans toutes les plates-formes que Go prend en charge, y compris Linux, Mac OS X, Windows et ARM.

Pour permettre le travail collaboratif de plusieurs gestionnaire sur différentes demandes au sein de Capirca, il a était décidé de mettre en place un serveur git. Le choix de Gogs repose sur différentes raisons. La première étant l’impossibilité d’utilisé de grande enseigne tel que GitHub out GitLab pour des raisons de sécurité. La seconde étant que Gogs propose une interface relativement proche de ces dernières, ce qui est un plus pour la gestion des utilisateurs et des droits associés.

Le serveur Gogs est destiné à être installé sur la même machine que le manager ou du moins une machine présente sur le même réseau qui a un accès au firewall PRINC. Sur le serveur est présent un repository distant contenant Capirca pour Telespazio ( Capirca avec « sync » et « diff »). Chaque gestionnaire peut alors claner le projet et ensuite ajouter les nouvelles demmandes d’ouvertures de flux ou en modifier.

Pour des raisons évidentes de sécurité ce projet n’a pu être testé directement au niveau du PRINC. Tous le projet a donc étaient réalisé et présenté au demandeur à l’aide d’une maquette. Cela est présenté dans la partie qui suit.



Figure

## La maquette

La maquette est relativement simple, elle consiste en un firewall de type Cisco ASA, d’un client (PC) et d’un serveur Gogs, chaque élément étant sur le même réseau. Le but de cette maquette étant de montrer simplement l’interaction entre les différentes parties.

Chaque liaison à était sécurisé le plus possible. Entre le serveur Gogs et le client à l’aide de l’utilisation du serveur ssh (avec certificats) intern de Gogs définis sur un port autre que le 22. Chaque gestionnaire utilise alors une addresse du type [git@gitlab.com:TPZ/Teles.git](mailto:git@gitlab.com:TPZ/Teles.git), pour cloner, pusher, puller …

Pour synchroniser la running-config il faut la récupérer, face à l’impossibilité d’utiliser scp, j’ai décidé d’utiliser la librairie python netmirko, le serveur gogs se charge donc de récupérer la running-config ( c’est « sync » qui se charge de cela). Les dernières version d’ASA ne permetttant pas d’utiliser de certificats il est utiliser ici le niveau maximum de sécurité au travers de ssh avec identifiant et mot de passe.

Pour pouvoir récupérer cette running config « sync » utilise les hooks de git, plus particulièrement le hook pre-receive. Sous git un hook pemet d’éxecuter du code (un script) avant ou après un commit, un push … Le script pre-receive est éxecuter avant le traitement d’un push. Ainsi à chaque executio de run.sh est éxecuter un push à blanc qui permet la synchronisation avec le firewal PRINC du serveur Gogs (le hook pre-receive étant côté serveur), à la suite à l’aide d’un scp le gestionnaire et synchroniser à son tour.

La maquette à donc servit de démonstration de la solution que j’ai réalisé. Dans un premier temps elle sera intégré seulement au manager, c’est-à-dire que seul depuis la machine du manager on auras accès à Capirca. Dans un futur proche après installation d’un VPN dans tous le bâtiment elle devrait être étendus à ouverte à un certain nombre de poste. Dans une base spatiale les nouveaux projets prennent du temps à être intégré, quand on connait la sensibilité de ce mieux on comprend mieux pourquoi. Ceci est approfondie dans la partie qui suit : REDEX.

# REDEX (REtour D’EXperience)

Certaine difficulté on put apparaître au cours du projet comme par exemple la découverte du Cisco ASA. Plus qu’une difficulté j’appellerai cela une opportunité, celle d’apprendre à être ingénieur. En sortant d’un cursus on ne peut forcément tout connaitre, par contre on doit savoir s’adapter, c’est cela un ingénieur, et c’est cela que j’ai pu mettre en œuvre.

Il a était assez facile de s’approprier le projet, que cela soit par mon parcours (DUT R&T) mais aussi par la présence d’une bonne équipe dans le service. Tout particulièrement le responsable du groupe « Groupe Réseaux, Commutation, Configuration et Supports ». Quand des questions se présenter, comme par exemples ce qui pouvait être réalisable tout en respectant les procédures, il suffisait de demander au voisin de bureau. Tout le monde était répondant. Avant tout codé mon tuteur de stage Mr Philippe Charron demander une spécification technique du besoin. Cela lui permettait de savoir si j’avais bien compris les attentes. De m’on coter cela m’a permis de structurer ma pensée, de définir les objectifs. Dans l’ensemble ce projet m’a permis de voir les attentes porter envers un ingénieur, mais aussi et surtout à apprendre à endosser ce rôle. Ce projet fut aussi l’occasion de mettre en place les mesures de sécurité appris durant le cursus, particulièrement en sécurité réseau.

Comme évoquer plus haut il reste à intégrer le projet au système existant, c’est la façon courante ici de faire des maquettes avant intégration. L’entreprise test pendant une certaine durée avant son intégration, se rappeler que le PRINC et un élément critique. Pour améliorer le projet il y ‘aurait la possibilité de mettre en place une interface plus intuitive en ajoutant par exemples des graphismes.

# Conclusion

Le sujet proposé était l’étude de la possibilité de migrer les règles existantes du PRINC dans ce logiciel et mettre en place les mécanismes adéquat permettant la gestion de configuration des demandes flux utilisateurs

Le stage consistera donc en 2 points majeurs :

• Mettre en place un mécanisme de génération des règles et de leur gestion en configuration (git ou subversion ?) à partir des demandes de flux utilisateurs.

• Adapter les règles existante pour les injecter dans le système.

En ce qui concerne le premier point j’ai choisi et mis en place un serveur git au travers de Gogs.

Pour le deuxième point j’ai développé « sync » et «diff ». Au final comme réponse à l’étude j’ai une démonstration de ma solution au travers d’une maquette qui reflète de la manière la plus précise possible le système d’intégration cible (OS, réseau …). En tant que futur ingénieur j’ai eu la responsabilité d’une étude, c’est-à-dire de proposer une solution (démontré) à un besoin. Pour cela un cadre était défini, celui de la base spatial et ses mesures de sécurité à haut niveau, ici tout fonctionne aux procédures. Malgré ce cadre une grande autonomie m’étais accorder, c’est-à-dire que toute solution qui répondait au besoin et qui respecter les attentes du CSG, était bonne à prendre.

Pour des raisons de sécurité certaines informations n’on put être détaillé dans ce résumé d’activité. On peut citer par exemples le détail des systèmes utilisé, les ips, la définition des architectures réseaux … etc … Comme montrer ici c’était un projet en plein cœur de la filière sécurité informatique, la base spatial et un site critique à ce niveau, avoir travaillé à Telespazio Guyane, m’a permis d’entrevoir les possibilités qui y régner, et surtout m’a permis de me comprendre mon futur métier. Qui sait peut être que dans un futur stage je serai appelé à améliorer des systèmes que j’aurais personnellement mis en place.

J’ai eu la chance durant ce stage de travailler sur une base spatiale, avec tout ce que cela peut signifier. J’ai eu par exemple la posibilité de voire le transfert, du BIL (Batiment Intégration Lanceur) vers le BAF (Batiment Assemblage Finale, du lanceur lourd Ariane 5. Qui à dit que le métier d’ingénieur informaticien ce devait d’être ennuyeux.