Table des matières

INTRODUCTION

Le tunneling, plus communément appelé transfert de port, est le processus de transmission de données qui est destiné à un usage privé uniquement, ou d'un serveur vers une application web.

Un tunnel localhost établit une connexion entre votre machine locale et une connexion distante. La connexion est destinée au trafic à partir d'une adresse IP et d'une URL adressables publiquement vers votre ordinateur local.

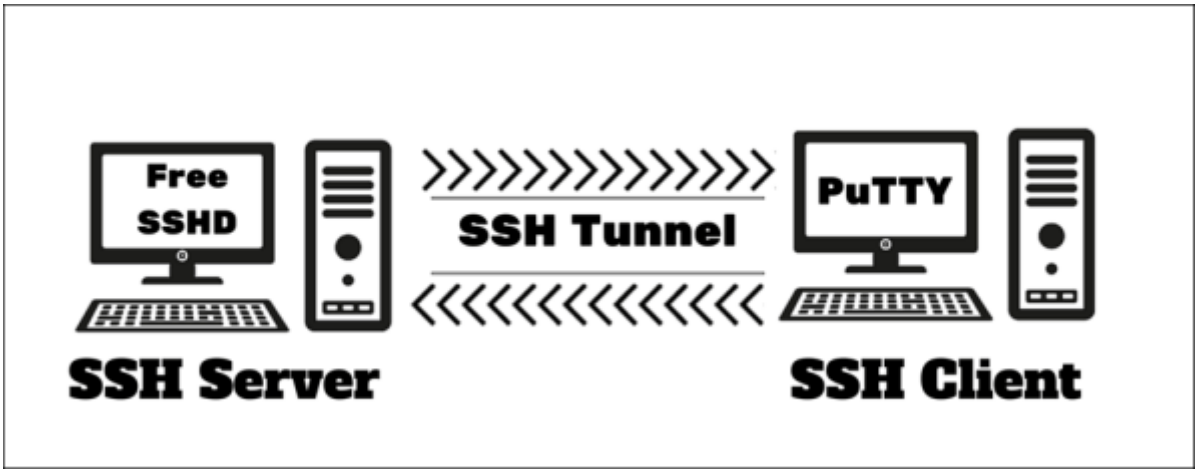
Des lors que se pose le besoin de visiter notre serveur web domestique de l'extérieur, nous devons en principe configurer un DNS dynamique car le FAI ne nous donnera pas d'adresse IP publique statique et certains fournisseurs dans le monde n'attribuent même pas d'adresse IP publique dynamique à chaque abonné. De nombreux abonnés partagent une seule adresse IP publique. Dans ce scenario, le DNS dynamique ne fonctionnera pas et c'est à ce niveau qu'intervient les diverses technologies de tunneling mises en place.
En effet, dans le besoin d'exposer son serveur local sur le web, il existe une solution très repondue qui est l'utilisation d'applications/outils permettant de créer un tunnel, non seulement pour cette exposition, mais aussi pour l'échange permanant de données entre son serveur local et l'application chargée en ligne. Comme outils/applications nous pouvons citer serveo, ngrok, pagekit, localXpose, beame-insta ssl, sish et bien d'autres.
Notre travail ici ne s'intéressera qu'à la solution sish. Cet outil sous MIT License Copyright (c) 2019 Antonio Mika est une alternative open source serveo/ngrok (les plus célèbres de sa catégorie) fournissant des tunnels http(s)/WS(s)/TCP à l'hôte local en utilisant uniquement SSH.

I- Le tunneling : présentation

1- Définition

Le tunneling, plus communément appelé transfert de port, est le processus de transmission de données qui est destiné à un usage privé uniquement. Généralement, il s'agit d'informations confidentielles dans un réseau d'entreprise via un réseau public de telle sorte que les nœuds qui sont routé dans le réseau public deviennent inconscients que le processus de transmission fait partie du réseau privé. En termes simples, le tunneling est un protocole de communication qui permet le transfert de données d'un réseau à un autre. Il implique des étapes spécifiques qui permettent d'envoyer des communications sur un réseau privé à travers un réseau public, ce processus s'appelle l'encapsulation. Dans ce processus d'encapsulation, les paquets de données apparaissent comme s'ils étaient de nature publique sur un réseau public alors qu'en réalité ils sont considérés comme des paquets de données privés. Cela leur permet de passer inaperçus.

Dans notre cas il s'agit d'un tunnel ssh, permettant d'exposer sur le web un serveur local ssh et d'échanger les do avec ce dernier.



2- Fonctionnalités

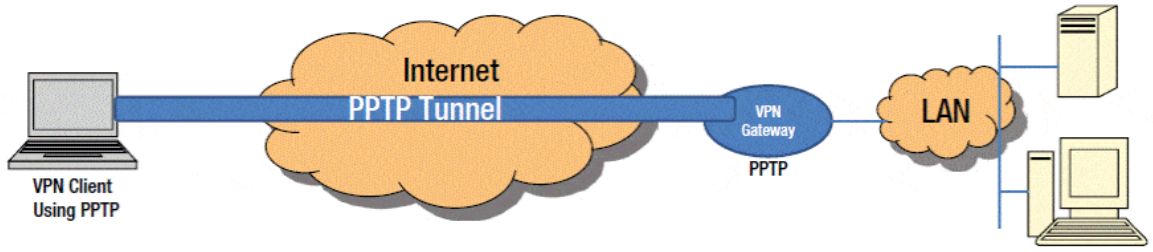
Les Dans le processus de construction du tunnel, les données seront décomposées en plus petits morceaux, connus sous le nom de paquets, qui se déplaceront le long du "tunnel" pour être transportés jusqu'à leur destination finale. Lorsque ces paquets traversent le tunnel, ils sont cryptés et encapsulés. données du réseau privé et le protocole d'information qui l'accompagne sont également encapsulés dans des unités de transmission du réseau public pour l'envoi. Le processus de décapsulation et de décryptage aura lieu à la réception. De plus, le tunnel est considéré comme le chemin logique ou la connexion qui encapsulera les paquets qui traversent le réseau de transit. Ce protocole de tunneling cryptera la trame d'origine afin que le contenu ne soit pas interprété en dehors de sa route. Pour que le processus fonctionne vraiment, les données seront envoyées une fois que le tunnel est déjà en place et que les clients ou le serveur utiliseront le même tunnel pour envoyer et recevoir les données sur le réseau Internet. Le transfert des données dépendra des protocoles de *tunnelage* utilisés pour le transfert.

3- Protocoles de tunnelage

Vous trouverez ci-dessous les différents protocoles qui permettent de réaliser un tunnel :

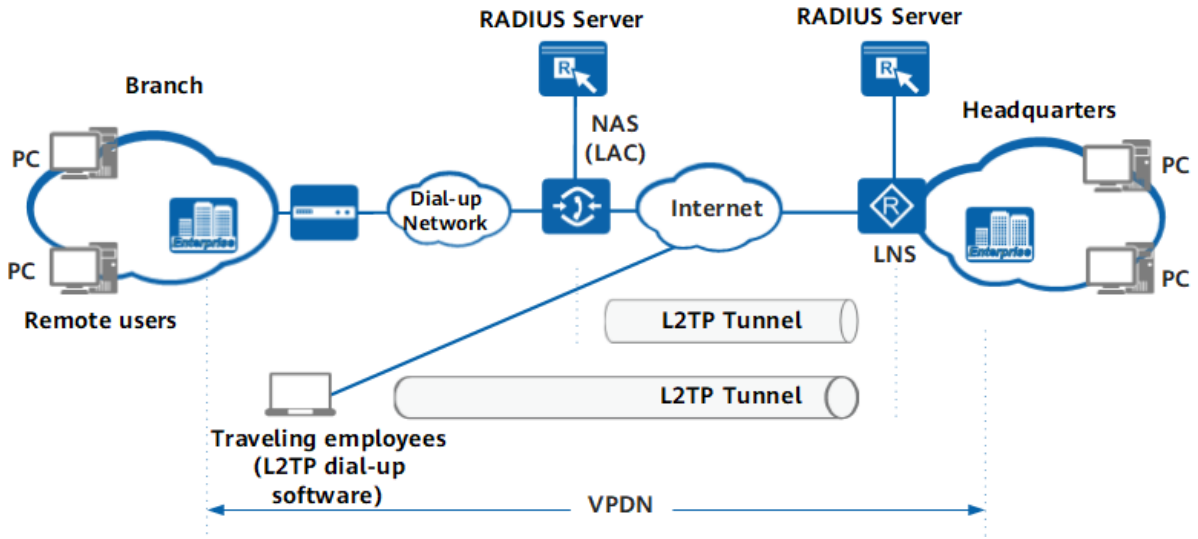
Protocole de tunnelage point à point (PPTP)

Ainsi, les données sont sécurisées même si elles sont communiquées sur des réseaux publics. Les utilisateurs autorisés peuvent accéder à un réseau privé appelé réseau privé virtuel ou VPN qui est fourni par un fournisseur de services Internet ou un FAI. Il s'agit d'un réseau privé au sens virtuel du terme car il est créé dans un environnement tunnelisé. Ce protocole permet aux entreprises d'étendre leur propre réseau d'entreprise via un canal privé sur Internet public.



Protocole de tunnelisation de couche 2 (L2TP)

Ce protocole implique une combinaison d'utilisation de PPTP et de transfert de couche 2. Il est utilisé pour soutenir les réseaux privés virtuels (VPN) dans le cadre de la fourniture de services par des protocoles de services Internet ou des FAI. Il n'assure pas à lui seul le chiffrement et la confidentialité. Mais, il s'appuie sur un protocole de cryptage qu'il fait passer dans le tunnel afin d'assurer la confidentialité. Il utilise des connexions réseau à commutation par paquets qui permettent aux terminaux d'être situés sur différentes machines. Cela signifie simplement que la connexion peut se terminer à un concentrateur de circuit local et élimine d'éventuels frais d'interurbain, entre autres avantages. Par conséquent, d'un autre point de vue, il n'y a pas vraiment de différence en termes d'opération.



Ainsi, le tunneling est vraiment utile et utile dans un cadre d'entreprise et il donne aussi des caractéristiques de sécurité telles que les options de cryptage. En résumé, les tunnels sont considérés comme un mécanisme utilisé pour envoyer des protocoles non pris en charge sur des réseaux différents et divers. Les données tunnées, VPN ou autres, ajouteront à la taille du paquet, ce qui réduira le nombre de données envoyées par paquet. Ces données de tunneling sur SSH sont normalement un VPN par application, mais la dernière version du SSH ouvert implémentera un VPN sans tracas.

II- Présentation de SISH

Une alternative open source serveo / ngrok. HTTP (S) / WS (S) / TCP Tunnels vers localhost en utilisant uniquement SSH.

Il s'agit d'un projet open Source développé et détaillé sur le compte Antonio Mika sur le GitHub et dont la licence et les droits reviennent au depuis 2019 au MIT.

Comme toutes les autres technologies citées ci-haut SISH permet d'exposer un localhost sur le web en créant un tunnel qui offre à un navigateur la possibilité d'échanger avec un localhost et de charger des données depuis ce dernier. Ainsi, non seulement cet outil permet d'exposer notre serveur mais il permet surtout une communication à distance avec celui-ci.

Pour cela, SISH utilise un certain nombre de technologies et de protocoles, tels que le http, ws, tcp, ssh, docker, customs domains, CLI …etc, qui seront détaillées dans la prochaine section.

III- Protocoles et technologies utilisés par SISH

Le SISH repose sur quelques technologies que nous allons essayer d'expliciter ici.

HTTP (S)

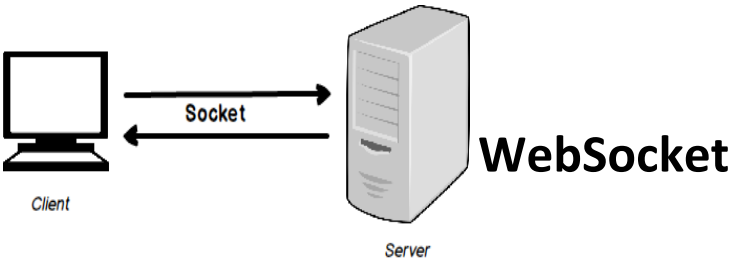
L'HyperText Transfer Protocol (HTTP, littéralement « protocole de transfert hypertexte ») est un protocole de communication client-serveur développé pour le World Wide Web. HTTPS (avec S pour secured, soit « sécurisé ») est la variante sécurisée par l'usage des protocoles Transport Layer Security (TLS).

HTTP est un protocole de la couche application. Il peut fonctionner sur n'importe quelle connexion fiable, dans les faits on utilise le protocole TCP comme couche de transport. Un serveur HTTP utilise alors par défaut le port 80 (443 pour HTTPS).

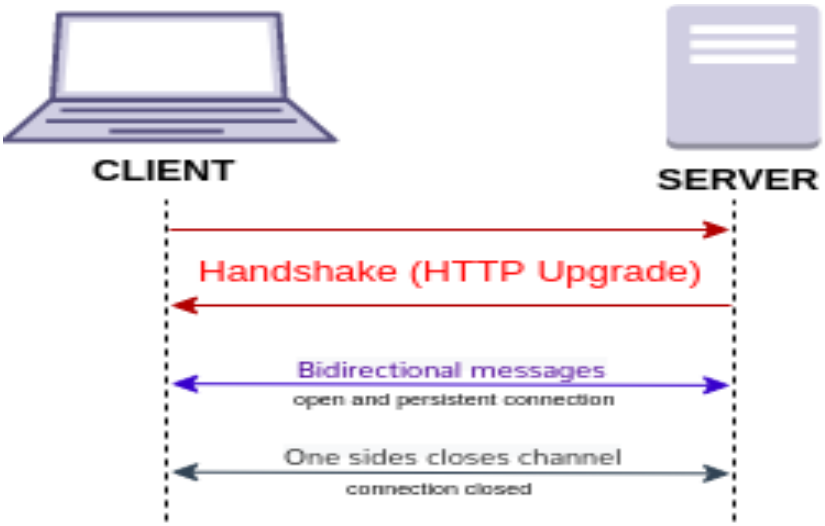
Les clients HTTP les plus connus sont les navigateurs Web permettant à un utilisateur d'accéder à un serveur contenant les données. Il existe aussi des systèmes pour récupérer automatiquement le contenu d'un site tel que les aspirateurs de site Web ou les robots d'indexation.

Dans le protocole HTTP, une méthode est une commande spécifiant un type de requête, c'est-à-dire qu'elle demande au serveur d'effectuer une action. En général l'action concerne une ressource identifiée par l'URL qui suit le nom de la méthode.

Dans ce cas il permet aux navigateurs web d'échanger avec le serveur et surtout de récupérer les données pour afficher les pages web.



WebSocket est un standard du Web désignant un protocole réseau1 de la couche application et une interface de programmation du World Wide Web visant à créer des canaux de communication full-duplex par-dessus une connexion TCP pour les navigateurs web. Le protocole a été normalisé par l'IETF dans la RFC 64552 en 2011 et l'interface de programmation3 par le W3C.



Le protocole WebSocket permet d'ouvrir un canal de communication bidirectionnel (ou "full-duplex") sur un socket TCP pour les navigateurs et les serveurs web. Plus spécifiquement, il permet donc :

- La notification au client d'un changement d'état du serveur,
- L'envoi de données en mode « pousser » (méthode Push) du serveur vers le client, sans que ce dernier ait à effectuer une requête.

TCP

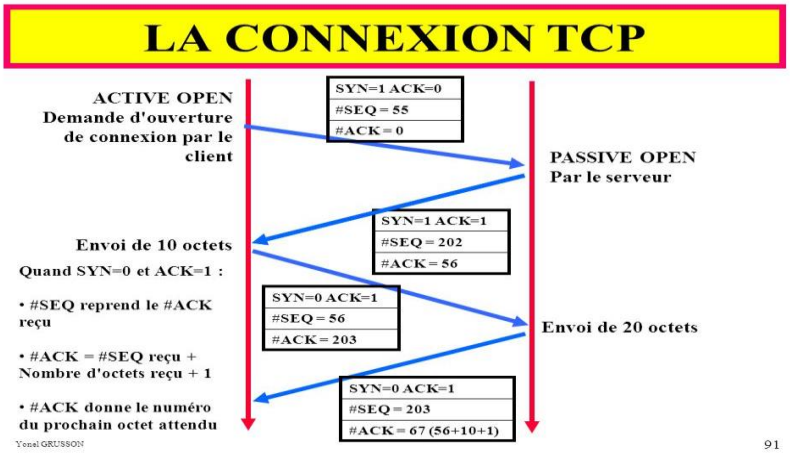
Transmission Control Protocol (littéralement, « protocole de contrôle de transmissions »), abrégé TCP, est un protocole de transport fiable, en mode connecté, documenté dans la RFC 793 de l'IETF.

Dans le modèle Internet, aussi appelé modèle TCP/IP, TCP est situé au-dessus de IP. Dans le modèle OSI, il correspond à la couche transport, intermédiaire de la couche réseau et de la couche session. Les applications transmettent des flux de données sur une connexion réseau. TCP découpe le flux d'octets en *segments* dont la taille dépend de la MTU du réseau sous-jacent (couche liaison de données).

TCP a été développé en 1973 puis adopté pour Arpanet en 1983, remplaçant NCP (RFC 801).

Une session TCP fonctionne en trois phases :

- L'établissement de la connexion ;
- Les transferts de données ;
- La fin de la connexion.



Tout service basé sur TCP peut être utilisé avec sish pour le transfert TCP et alias. Le transfert TCP établira un port distant sur le serveur sur lequel vous déployez sish et transférera toutes les connexions vers ce port via la connexion SSH et vers votre appareil local.

Dans le cas ou nous ne voulons pas que le service soit accessible par le reste du monde, vous pouvez alors utiliser un alias TCP. Un alias TCP est un type de connexion TCP transférée qui n'existe qu'à l'intérieur de sish. Vous pouvez accéder à l'alias en utilisant SSH avec l'indicateur

-W, qui transmettra stdin/stdout du processus SSH à la connexion TCP transmise. En combinaison avec l'authentification, cela garantira que votre service distant est à l'abri du reste du monde, car vous devez vous connecter pour sish avant de pouvoir y accéder.

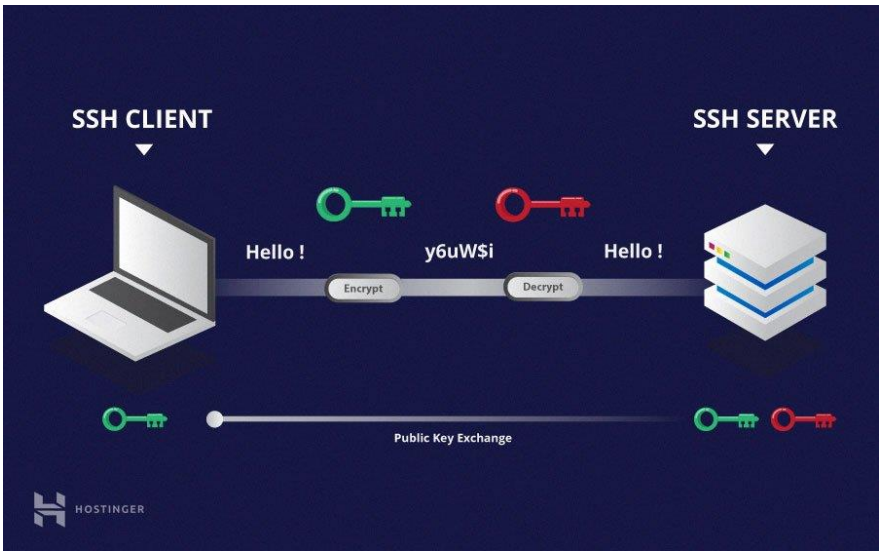
SSH

Secure Shell (SSH) est à la fois un programme informatique et un protocole de communication sécurisé. Le protocole de connexion impose un échange de clés de chiffrement en début de connexion. Par la suite, tous les segments TCP sont authentifiés et chiffrés. Il devient donc impossible d'utiliser un *sniffer* pour voir ce que fait l'utilisateur.

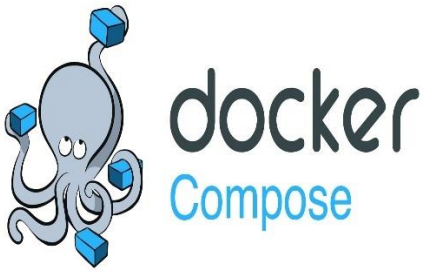
Le protocole SSH existe en deux versions majeures : la version 1.0 et la version 2.0.

- La première version permet de se connecter à distance à un ordinateur afin d'obtenir un shell ou ligne de commande. Cette version souffrait néanmoins de problèmes de sécurité dans la vérification de l'intégrité des données envoyées ou reçues, la rendant vulnérable à des attaques actives. En outre, cette version implémentait un système sommaire de transmission de fichiers, et du *port tunneling*.
- La version 2 qui était à l'état de brouillon jusqu'en janvier 2006 est largement utilisée à travers le monde.

Cette version est beaucoup plus sûre au niveau cryptographique, et possède en plus un protocole de transfert de fichiers complet, le SSH file Transfer Protocol.



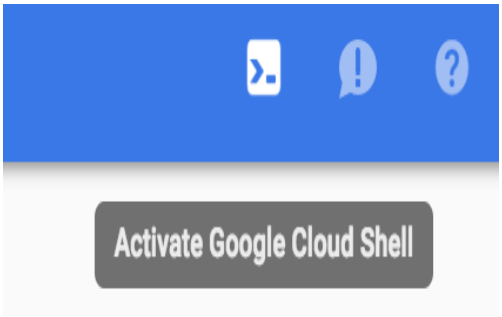
SSH peut normalement transférer les ports locaux et distants. Ce service implémente un serveur SSH qui ne gère que le transfert et rien d'autre. Le service prend en charge le multiplexage des connexions via HTTP/HTTPS avec la prise en charge de WebSocket.



Docker compose

Docker Compose est un outil permettant d'exécuter des applications multi-conteneurs sur Docker définies à l'aide du format de fichier Compose. Un fichier Compose est utilisé pour définir comment le ou les conteneurs qui composent votre application sont configurés.

La commande docker-compose up agrège la sortie de chaque conteneur (essentiellement en exécutant docker-compose logs --follow). Lorsque la commande se termine, tous les conteneurs sont arrêtés. L'exécution de docker-compose up --detach démarre les conteneurs en arrière-plan et les laisse s'exécuter.



Google Cloud Shell

Google Cloud Shell est un Shell Bash en ligne basé sur Debian. Le niveau gratuit comprend 1,7 giga-octets de mémoire vive et un répertoire personnel persistant de 5 giga-octets. À l'exception des répertoires d'accueil et racine, l'environnement Cloud Shell est volatile. L'éditeur de Google Cloud Shell est basé sur Eclipse Théia.

C’est l’environnement utilisé pour développer ce projet.

IV- FONCTIONNALITES DE SISH

SISH est un utilitaire de ligne de commande qui implémente un serveur SSH capable de gérer le multiplexage, le transfert et l'équilibrage de charge HTTP(S)/WS(S)/TCP.

Il peut gérer plusieurs points de terminaison d'hébergement virtuel et de tunneling inversé pour un grand nombre de clients.

L'utilisation de SISH peut se faire à l'aide de plusieurs commandes ayant chacune une fonction précise Nous pouvons lister quelques commandes ainsi que leur utilité :

- --admin console : qui permet d'Activez la console d'administration accessible à l'adresse http(s)://domain/_sish/console ?x-authorization=admin-console-token
- --append-user-to-subdomain : qui permet d'ajouter un utilisateur SSH au sous-domaine
- --banned-aliases string : qui affiche Une liste séparée par des virgules d'alias interdits que les utilisateurs ne peuvent pas lier
- --banned-countries string : qui affiche Une liste de pays interdits séparés par des virgules. S'applique aux connexions HTTP, TCP et SSH
- --banned-ips strings : qui affiche Une liste séparée par des virgules d'adresses IP interdites qui ne peuvent pas accéder au service. S'applique aux connexions HTTP, TCP et SSH
- --log-to-client : qui permet d'activer la journalisation des requêtes HTTP et TCP au client
- --log-to-file : qui permet d'activer l'écriture de la sortie du journal dans un fichier spécifié par la commande log-to-file-path
- --ping client : qui permet d'envoyer des requêtes ping au client SSH sous-jacent. Ceci est utile pour s'assurer que les connexions SSH restent ouvertes ou fermées proprement
- ---port-bind -range string : Les ports ou plages de ports qui sish permettront d'être liés lorsqu'un utilisateur tente d'utiliser le transfert TCP
- --private-key-location string : qui donne l'L'emplacement de la clé privée du serveur SSH. Sish créera une clé privée ici s'il n'existe pas en utilisant --private-key-passphrase pour le chiffrer s'il est fourni (par défaut "deploy/keys/ssh_key")
- --http-load-balancer : qui permet d'activer l'équilibreur de charge HTTP (plusieurs clients peuvent lier le même domaine)
- --private-key-passphrase string : qui donne le mot de passe à utiliser pour chiffrer la clé privée du serveur (par défaut "S3Cr3tP4$$phrAsE")
- --proxy-Protocol : qui permet d'utiliser le protocole proxy lors de la transmission par proxy des connexions afin de transmettre l'adresse IP et les informations de port

- --service-console : qui permet d’activer la console de service pour chaque service et envoyer les informations aux clients connectés

- Ssh-adress string : qui permet d’afficher l'adresse pour écouter les connexions SSH (par défaut "localhost :2222")

- --verify-DNS : qui permet de vérifier les informations DNS pour les hôtes et assurez-vous qu'elles correspondent à une empreinte de clé sha256 des utilisateurs connectés (par défaut vrai)

- --tcp-aliases : qui permet d’activer l'utilisation de l'alias TCP

Sish peut transférer n'importe quel nombre de connexions HTTP via SSH. Il fournit également la journalisation des connexions au client connecté qui a transféré la connexion et une interface Web pour voir la demande complète et les réponses apportées à chaque connexion transférée. Chaque interface Web peut être unique à la connexion transférée ou utiliser un jeton d'accès unifié. Pour utiliser le transfert HTTP, les ports [80, 443] sont utilisés pour indiquer à sish qu'une connexion HTTP est transférée et que l'hébergement virtuel HTTP doit être défini pour le service.

Il existe encore beaucoup d’autres commandes que nous pouvons citer pour illustrer le fonctionnement de SISH mais celles citées plus haut devrons faire l’affaire.

V- Les alternatives à SISH

Serveo

Utilisez SSH pour exposer les serveurs locaux sur Internet et inspecter et relire le trafic HTTP.

Lorsque vous établissez une connexion SSH sur serveo.net, un sous-domaine est affecté au transfert du trafic HTTP (et HTTPS) vers votre serveur local.

Tunnel de ssh - SSH Port Forwarding - Aucune inscription requise - Aucune installation requise.

Ngrok

Sécuriser les tunnels introspectables à localhost.

Ngrok expose les serveurs locaux derrière les NAT et les pares-feux à l'Internet public via des tunnels sécurisés.

Gratuit avec fonctionnalités limitées - FreeBSD - BSD - Linux - Windows - Mac

Pagekite

Pagekite est un logiciel qui attribue des noms à vos serveurs localhost et les rend visibles dans le monde entier. Cela fonctionne avec n'importe quel ordinateur et n'importe quelle connexion Internet.
C'est tellement simple que vous ne voudrez plus jamais penser aux routeurs, aux adresses IP ou à d'autres aspects techniques. Il est aussi open source.

LocalXpose

LocalXpose est un proxy inverse qui vous permet d'exposer votre hôte local à Internet.

Profitant des technologies innovantes issues de la crypto-monnaie et de la connexion d'égal à égal.
LocalXpose est une plate-forme de nœuds distribués à travers le monde (pays autorisant les connexions entrantes) prêts à louer leur bande passante et leurs ports pour les nœuds bloqués afin qu'ils puissent exposer les services hébergés locaux, ce qui permet de créer un système économique tout en utilisant le service de transfert de port.

LocalXpose fonctionne en construisant un tunnel entre Blocked-Node et Open-Node.
Ainsi, vous, amis ou clients, pourrez vous connecter à vos services hébergés localement directement en utilisant uniquement un nom de domaine.

Beame-insta-ssl

Exposez en toute sécurité votre propre ordinateur ou serveur sur le Web.

Beame-insta-ssl est open-source et gratuit ! Il permet à tout développeur Web d'utiliser facilement le cryptage et d'obtenir des communications sécurisées.

Beame-insta-ssl vous procurera votre propre nom de domaine entièrement qualifié sous le sous-domaine beame qui ressemble à ceci : https://ypxf72akb6onjvrq.ohkv8odznwh5jpwm.v1.p.beameio.net/insta-ssl et votre propre gratuit Certificat SSL (AC racine GlobalSign). Cela vous permettra de tunnelier en toute sécurité sur HTTP. Vous pouvez choisir de mettre fin ou de ne pas mettre fin à la sécurité de la couche de transport (TLS).

Et bien d’autres encore.

VI- Implémentation

Cette section servira à décrire le processus d'implémentation, celle-ci s'est faite sur google cloud Shell décrit dans la section Protocoles et technologies de SISH.

Les étapes seront données telles qu'elles nous ont été présentées sur le compte GitHub qui a servi de socle pour ce travail.

- Tirez l'image Docker

```
docker pull antoniomika/sish:latest
```

- Exécuter l'image

```
docker run -itd --name sish \
 -v ~/sish/ssl:/ssl \
 -v ~/sish/keys:/keys \
 -v ~/sish/pubkeys:/pubkeys \
 --net=host antoniomika/sish:latest \
 --ssh-address=:22 \
 --http-address=:80 \
 --https-address=:443 \
 --https=true \
 --https-certificate-directory=/ssl \
 --authentication-keys-directory=/pubkeys \
 --private-key-location=/keys/ssh_key \
 --bind-random-ports=false
```

- SSH à votre hôte pour communiquer avec sish

```
ssh -p 2222 -R 80:localhost:8080 ssi.sh
```

- Docker Compose

Vous pouvez également utiliser Docker Compose pour configurer votre instance sish

```
docker-compose -f deploy/docker-compose.yml up -d
```

- Transfert http

```
ssh -R hereiam:80:localhost:8080 ssi.sh
```

- Transfert TCP

```
ssh -R 2222:localhost:22 ssi.sh
```

- Transfert d'alias TCP

```
ssh -R mylaptop:22:localhost:22 ssi.sh
ssh -o ProxyCommand="ssh -W %h:%p ssi.sh" mylaptop
ssh -J ssi.sh mylaptop
```

- Authentification

sish@sish0:~/sish/pubkeys# curl https://github.com/antoniomika.keys > antoniomika

- Domaines personnalisés

sish=SSHKEYFINGERPRINT

ssh-keygen -lf ~/.ssh/id_rsa | awk '{print $2}'

- Load balancing

`--http-load-balancer`

`--tcp-load-balancer`

`--alias-load-balancer`

- Ajouter des adresses IP à la liste blanche

La liste blanche des plages d'adresses IP ou des pays est également possible. Des plages CIDR entières peuvent être spécifiées avec l'option --whitelisted-ips qui accepte une chaîne séparée par des virgules.

Pour ajouter des pays à la liste blanche, utilisez --whitelisted-countries avec une chaîne de pays séparée par des virgules au format ISO (par exemple, "pt" pour le Portugal). Vous devrez également définir --geodb sur true.

- Configuration DNS

Pour utiliser sish, vous devez ajouter un enregistrement DNS générique qui est utilisé pour les sous-domaines multiplexés. L'ajout d'un enregistrement A avec * comme sous-domaine à l'adresse IP de votre serveur est le moyen le plus simple de réaliser cette configuration.

Il faudra ensuite se rendre sur le google cloud shell, sur google faudra créer un projet puis aller sur le shell et suivre les instructions.

VII-Difficultés rencontrées

Pour réaliser ce travail nous avons fait face à un certain nombre de difficultés, tout d'abord au niveau de la documentation ; en effet de toutes les solutions de ce genre exposé dans ce travail, SISH est celle avec la plus faible documentation, celle-ci est quasi inexistante, les seules références plus ou moins détaillées existantes sont celles de GitHub, et le peu de sites qui essaient d'en parler nous renvoient tous directement vers la référence qu'est GitHub. On pourrait comprendre une telle pauvreté dans sa littérature, notamment parce que cet outil est très récent, sa licence ne date que de 2019, et donc très certainement très peu de gens en ont connaissance, pire encore très peu ont eu à l'utiliser, du coup pour les difficultés rencontrées au niveau du déploiement il est quasi impossible de trouver des solutions déjà proposées car personne auparavant n'a encore eu à les soulever ou à les rencontrer vu le peu de personnes l'ayant déjà testé.

La seconde difficulté à été la méconnaissance de l'environnement de développement utilisé, en effet auparavant nous n'avions guère eu à utiliser cet environnement ni pour créer des projets ni pour en déployer.

La troisième difficulté a été la plus importante, pour le travail qui était le notre il nous a fallu un domaine, effet, le serveur étant local, mais une fois sur la toile il aurait fallu le nom d'un domaine afin de joindre le serveur du site à déployer, ce qui n'est pas donné, car en principe les domaines sont payant et nous n'avions pas les moyens pour nous en procurer un, du coup nous avons chercher une solution de replis, avoir un domaine gratuit, sauf qu'à ce niveau tous les domaine qui semblaient gratuits étaient par la suite obsolètes.

La troisième difficulté qui est celle que rencontrent quasiment tous ceux de notre domaine est celle de l'accès à une connexion internet fluide, en effet notre travail s'effectuant sur le google cloud Shell nous devrions être en permanence connectés lorsque nous travaillions ce qui très couteux et même lorsque nous arrivions à payer cela la qualité exécrable de la connexion ruinait sans cesse le travail.

Conclusion

Au terme de notre analyse sur le SISH, analyse dans laquelle nous avons commencé par présenter ce qu'est un tunnel dans le sens large de l'informatique, puis nous avons présenté ce qu'est le SISH en lui-même, un Tunnel HTTP (S) / WS (S) / TCP vers un localhost en utilisant uniquement SSH, ensuite il était primordial de ressortir ses protocoles puis ses fonctionnalités et par la suite de présenter des alternatives à SISH, SISH qui n'est pas le premier outil implémentant cette technologie encore moins l'unique. Tout cela fait, il était inévitable de faire une implémentation dans laquelle il fallait faire ressortir les fonctionnalités de SISH. Pour finir avec ce travail nous avons tenu à relever les difficultés auxquelles nous avons fait face dans la réalisation de ce travail.

Nous savons à présent combien cet outil est important, bien qu'il soit encore très récent ; relevons aussi le hic de celui-ci, contrairement à certains outils comme Pagekite ou Serveo ou un sous domaine est directement attribué, ici il faudrait déjà en avoir un, ce qui rend pour ainsi dire le travail un peu plus complexe.

Comme particularité, contrairement à d'autres logiciels ou applications du genre, SISH n'utilise que le SSH pour les échanges et la communication avec le serveur de ce type en local, ce qui définit le type de tunnel. L'outil est récent et devrait peut-être par la suite avoir des mis-à-jour ou des corrections pour faciliter son utilisation et son déploiement très complexe.

Références :

https://github.com/antoniomika/sish

https://progsoft.net/fr/software/sish

https://www.wikipedia.org/

https://www.speedcheck.org/fr/wiki/tunneling/

https://golangrepo.com/repo/antoniomika-sish-go-network

https://leviwheatcroft.github.io/selfhosted-awesome-unlist/sish.html