Création d'un boîtier UTM: Unified Threat Management

Frédéric BORDI, Mohamed DJOUDI,
Cédric LESEC, Guillaume ROUVIERE

10 mai 2009

Table des matières

Première partie

Introduction

Dans le cadre de notre première année de master informatique il nous a été demandé de faire un travail d'études et de recherches sur un sujet proposé. Nous avons choisi la création d'un boîtier de type UTM[1], qui a pour but de filtrer le flux en assurant la sécurité du réseau local via différents modules regroupés. Le choix de ce sujet nous est apparu comme étant une évidence compte tenu du domaine dans lequel nous aimerions nous spécialiser : les réseaux et leur sécurisation.

En premier lieu, il nous fallu observer ce qui est proposé sur le marché des UTM afin de nous renseigner sur les différentes fonctionnalités. À partir de ce travail, nous avons pu définir le cahier des charges, contenant les différents modules que nous allions devoir installer et configurer. Cette liste comprend pare-feu, antivirus, antispam, filtrage d'URL[2], IDPS[3] ainsi que divers serveurs mandataires servant, entre autres, à filtrer les données et à garder un historique des actions effectuées sur le réseau.

Nous avons ensuite effectué un travail de recherche afin de trouver le système d'exploitation et les différents logiciels nécessaires au boîtier ainsi que la documentation associée lorsque celle-ci existe. S'ensuivit une longue phase de configuration de chacun des différents modules au terme de laquelle nous avons developpé l'interface, un site intranet herbergé sur le boîtier offrant la possiblité de modifier la configuration de chaque module.

Vous trouverez dans la suite de ce rapport une présentation des boîtiers UTM, suivie de l'ensemble du travail effectué tout au long de ce TER, depuis le choix de l'architecture, jusqu'à la description de chaque module[4].

[1] Unified Threat Management : traitement unifié de la menace

[2] URL : adresse permettant de localiser une page

[3] Intrusion Detection and Prevention Systems : Système de détection et prévention des intrusions

[4] Le détail des configurations est disponible dans les annexes

Deuxième partie

Développement

Chapitre 1

Présentation générale

1.1 Situation de la sécurité informatique actuelle

L'explosion d'Internet entraîne la venue d'applications web riches qui amplifient les vecteurs d'attaques. En effet, ces applications ne sont pas dépourvues de vulnérabilités qui sont sans cesse traquées et exploitées par les pirates.

Viennent s'ajouter à cela virus, vers, spywares et autres *malwares*[1] dont le developpement croît aussi vite que les systèmes d'information. En 2008, l'éditeur Allemand G Data à recensé 894 250 nouveaux programmes malveillants, soit sept fois plus qu'en 2007.

99% des systèmes visés sont équipés avec Windows©. Les États-Unis et la Chine constituent 80% des attaques informatiques. La France, quant à elle, est en cinquième position des attaques de type *phishing*[2] avec $2,52\%$ derrière le Canada, l'Allemagne la Chine et les États-Unis. L'année dernière, le gouvernement français contre-attaque en décidant de répondre offensivement aux actes de malveillance.

1.2 Présentation des boîtiers UTM

Afin de faire face aux menaces émanant de l'Internet, la simple installation d'un pare-feu ou d'un antivirus n'est plus suffisante. En effet, l'augmentation et la diversité des attaques de plus en plus ingénieuses nous poussent à utiliser des outils plus spécialisés. Or, la mise en place intelligente de ces outils requiert une réflexion préalable et un savoir-faire certain quant à leur organisation. Ceci peut vite devenir un point bloquant pour les utilisateurs non connaisseurs ou pour certaines entreprises. Il est donc nécessaire de réfléchir à l'élaboration d'une solution centralisant ces outils sur un même boîtier (appliance). Un boîtier UTM, en français, Unité de Traitement de la Menace s'avère être un bon compromis. Il s'agit d'un boîtier regroupant l'ensemble des outils (nécessaires) pour pouvoir lutter efficacement contre les dangers de l'Internet. Dans la plupart des cas, ils sont placés entre l'accès Internet et le réseau privé. Les avantages de ce type de

[1] Logiciel malveillant développé dans le but de nuire á un système informatique

[2] C'est une technique qui repose sur l'usurpation d'identité. Le pirate se fait passer pour un tiers de confiance dans le but de soutirer des informations confidentielles. Cette technique est basée sur l'ingénierie sociale.

boîtier sont nombreux. Ils permettent de surveiller de façon continue un réseau informatique. Ils constituent la solution idéale pour les petites entreprises qui ne disposent pas de spécialistes en sécurité. La centralisation des modules facilite l'administration et la configuration de la sécurité du réseau.

1.3 Objectif du TER

L'objectif principal de notre TER est donc de réaliser notre propre boîtier UTM afin qu'il soit le plus efficace possible. Pour cela, nous nous appuyons sur une architecture de sécurité modulaire et cohérente. Notre UTM doit être constitué au minimum d'un pare-feu, d'un antivirus, d'une sonde de détection et de prévention des intrusions, d'un antispyware, d'un filtrage d'URL et d'un antispam. Des fonctionalités devront pouvoir y être ajoutées telles qu'un module pour le VPN (Virtual Private Network) par exemple. Cependant, nous garderons à l'esprit que l'intégration de multiples technologies sur un seul boîtier ne doit pas se faire au détriment des performances et de la cohérence de notre politique de sécurité. Par la suite, nous y inclurons un système de supervision du réseau et nous élaborerons une interface web de paramétrage afin de rendre accessible la configuration du boîtier aux utilisateurs ; mais, avant de mettre en place l'architecture du boîtier, il faut au préalable le configurer en tant que passerelle.

Chapitre 2

Architecture et configuration initiale

2.1 Présentation de l'architecture

Chaque module doit être indépendant des autres. C'est sur ce constat que nous nous sommes appuyés pour définir le type d'architecture de notre boîtier. Il était essentiel pour nous de pouvoir ajouter, modifier ou supprimer des modules sans avoir à reconfigurer les autres, et ce pour plusieurs raisons. Premièrement, notre politique est de proposer un boitier intéressant un large public. Deuxièmement, Le fait qu'un utilisateur refuse d'employer de manière ponctuelle ou définitive l'un des modules ne doit pas l'empêcher d'utiliser le boîtier. Enfin, point essentiel : la panne de l'un des modules ne doit en aucun cas affecter le fonctionnement des autres. En effet, dans le cadre d'une architecture en cascade où chaque module s'appuie sur les autres, la moindre erreur ponctuelle entraînerait un dysfonctionnement total du boîtier.

Dans une architecture de réseau "classique", on retrouve généralement un modem de type "box" (freebox, livebox etc. ...). Notre boîtier se positionnera alors entre la box et le réseau local et les utilisateurs du réseau se connecteront alors à notre boîtier plutôt qu'à la box.
Voici une illustration de cette organisation sur le réseau :

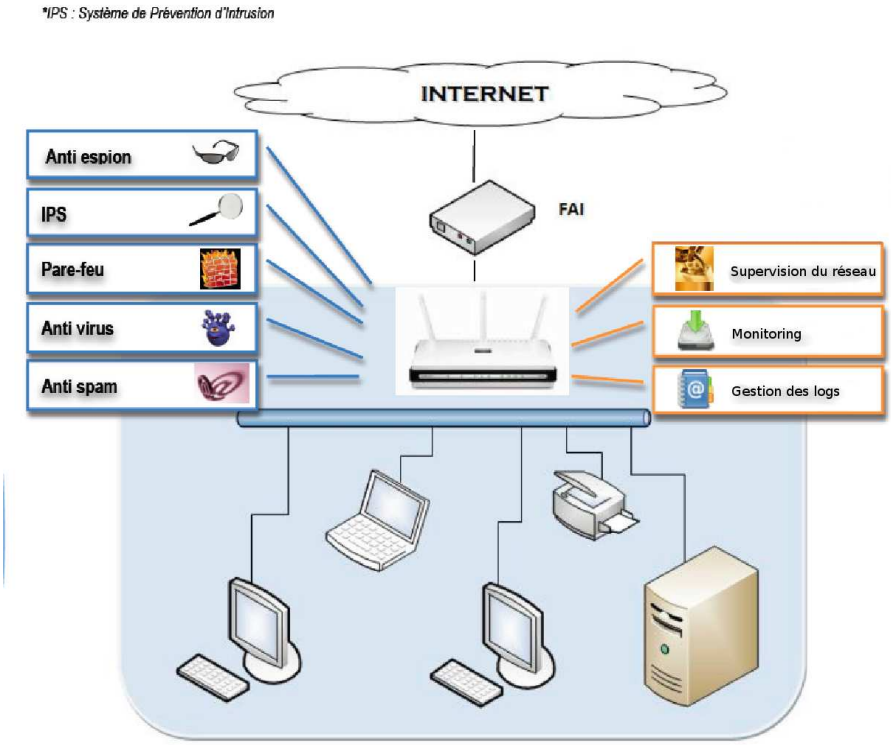

FIG. 2.1 – Architecture du boîtier au sein du réseau local

L'architecture intérieure quant à elle est plus complexe. Une illustration est présentée plus loin. Sur la gauche, les entrées dans le boîtier UTM, et sur la droite, les sorties. Le carré rouge entourant tout le boîtier représente le pare-feu. Celui-ci filtre les entrées et sorties du boîtier selon leur type, provenance, destination etc... Sur les entrées, on voit que la plupart sont redirigées vers les ports d'écoute des serveurs mandataires. C'est de cette facon qu'on leur envoie le flux précis sur lequel ils travaillent. Certains serveurs mandataires utilisent un module de filtrage comme l'antivirus ou l'antispam. À la sortie, la sonde de détection des intrusions filtre le contenu juste avant que le pare-feu ne le fasse à son tour, ensuite, le pare-feu s'occupe du routage en translatant les adresses IP. Tout le contenu de ce schéma sera expliqué plus en détails dans la description de chacun des modules.

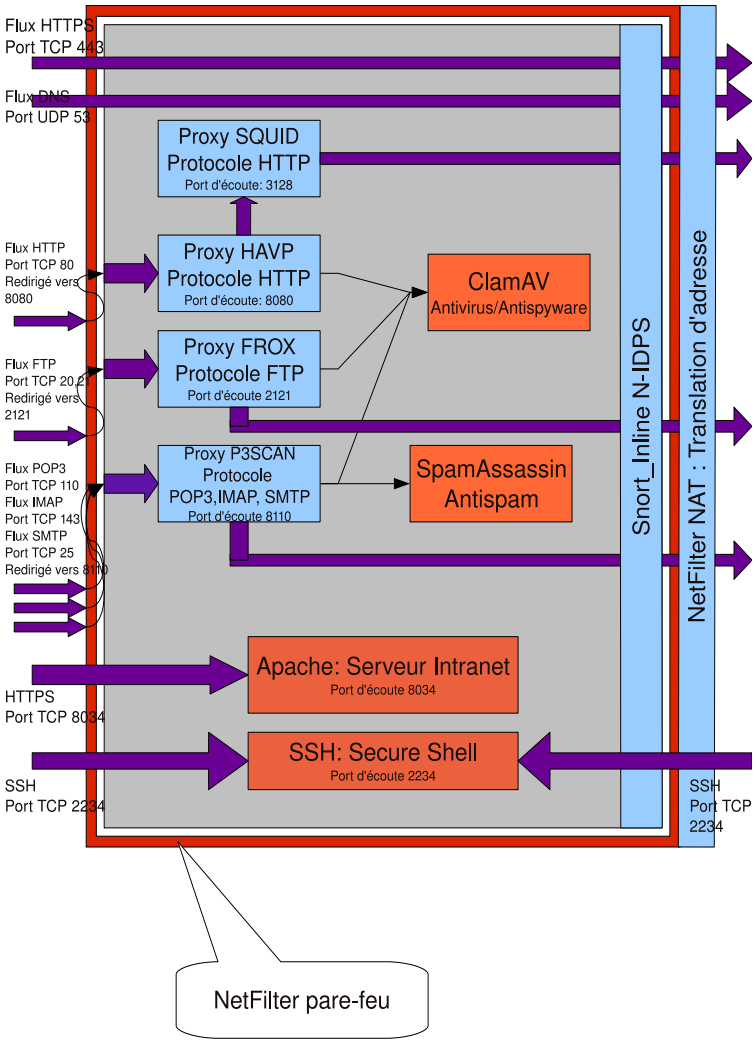

FIG. 2.2 – Architecture interne du boîtier

2.2 Configuration initiale

2.2.1 Problématique

Avant de commencer tout travail sur les modules, il nous fallait une machine faisant office de boîtier. Comme nous l'avons vu dans la partie architecture, les utilisateurs du réseau local se connectent directement au boîtier et non plus à leur box. Cela paraissait plus pratique pour travailler de pouvoir configurer le boîtier en point d'accès WiFi plutôt que d'utiliser un réseau ethernet. Nous

avons donc choisi l'un de nos ordinateurs portables équipé d'une carte WiFi et d'une carte ethernet comme première machine pour travailler.

Une fois la machine choisie, il nous fallait décider d'un système d'exploitation, léger, stable et efficace. Dans la mesure où tout le travail allait être effectué via le protocole SSH[1], que le boitier n'aurait pas d'écran et devrait rester léger, nous avons choisi de ne pas installer de serveur graphique. Ce choix nous restreignait par la suite dans les logiciels utilisés puisque ceux-ci devaient pouvoir se piloter intégralement en lignes de commandes. Cette recherche nous a amenés à un système répondant à toutes nos attentes : Debian 4.0 ETCH. Système que nous avons remplacé plus tard par sa toute nouvelle version 5.0 Lenny.

La première chose à mettre en place pour travailler, une fois le système d'exploitation installé, fut la fonction de routage, principale fonction du boîtier et, qui plus est, nécessaire à notre travail. Nous avons donc commencé par l'installation et la configuration du WiFi. Après une longue et pénible période consacrée à la reconnaissance de notre carte WiFi par le système, il s'est avéré qu'elle ne pouvait pas prendre en charge le mode point d'accès. Nous avons donc changé de machine pour une tour équipée d'un processeur Pentium III 500Mhz, 8Go de disque dur, et de 3 interfaces ethernet. Ce changement fut finalement intéressant car cette machine beaucoup moins puissante que la précédente nous a obligés à garder un système léger et peu gourmand en ressources. Suite à ce petit incident, nous avons abordé la configuration du réseau. Le boitier se fait automatiquement allouer une adresse IP par la box, depuis laquelle il aurait accès à Internet. Sur les autres interfaces, les utilisateurs connectés au boîtier devaient se faire allouer une adresse IP et voir leurs requêtes vers Internet aboutir. Il fallait donc configurer ce routage. Pour cela, il nous fallait configurer les interfaces réseaux, un serveur DHCP pour l'allocation des adresses IP et, enfin, gérer le routage des paquets.

2.2.2 Configuration

Comme nous l'avons vu précédemment, l'une des interfaces est connectée à la box tandis que les autres sont dédiées aux utilisateurs. Il fallait donc en configurer une avec une adresse automatique et les autres avec des adresses fixes. Cette configuration s'effectue depuis le fichier /etc/network/interfaces[2]. Elle est heureusement relativement rapide à prendre en main, puisque celle-ci fut régulièrement modifiée pendant notre travail.

Le serveur DHCP a pour rôle d'attribuer une adresse IP à tout utilisateur se connectant au réseau. Il lui indique aussi le serveur DNS[3] et la passerelle par défaut[4] à utiliser. L'installation, facile et rapide, précède la modification de deux fichiers pour configurer la plage d'adresses allouables, le serveur DNS à communiquer ainsi que différents paramètres.

[1] Secure Shell : protocole plus amplement décrit dans la partie qui lui est consacrée

[2] disponible en annexe

[3] Domain Name Server : type de serveur stockant les associations entre adresses URL et IP permettant de joindre les sites directement par leur nom plutôt que par leur adresse

[4] adresse à laquelle envoyer les paquets lorsqu'aucune autre ligne de la table de routage ne correspond à l'adresse du destinataire

Enfin, il faut activer le routage des paquets du réseau local vers Internet et inversement. Pour cela, on doit modifier le fichier /etc/sysctl.conf pour autoriser le transfert. On doit ensuite gérer le NAT[5], c'est-à-dire faire correspondre aux adresses IP non-uniques et non-routables du LAN[6], une ou des adresses externes et routables. En l'occurence, nous avons utilisé le module Masquerade d'iptables[7] afin de masquer les adresses du LAN de telle sorte que d'un point de vue extérieur, quel que soit l'ordinateur faisant une requête depuis le LAN, l'adresse IP externe visible depuis Internet est la même pour tous.

[5]Network Address Translation
[6]Local Area Network : réseau local
[7]Vu plus en détail dans la partie pare-feu

Chapitre 3

Description technique des modules du boîtier

3.1 Pare-feu

Un pare-feu est un logiciel permettant de filtrer les paquets entrants et sortants dans le boîtier en fonction de différents paramètres tels que, entre autres, adresses et ports de la source ou destination, protocole utilisé, état de la communication (établie, tentative de connexion, etc. ...). En dehors de ce rôle principal, le pare-feu a aussi d'autres caractéristiques moins connues. Ainsi, c'est aussi le pare-feu qui se charge du routage, ou des redirections de ports. La redirection de port est très utile en alliance avec un ou des serveurs mandataires. En effet, ceux-ci sont généralement en écoute sur un port particulier. Le pare-feu aura donc pour rôle de rediriger toute arrivée sur un port (prenons le 80 pour le HTTP) vers celui sur lequel le serveur mandataire écoute (8080 pour le serveur mandataire HTTP). Ainsi tout le flux envoyé sur le port 80 sera redirigé vers le 8080 par exemple. Le noyau de linux intègre depuis sa version 2.4 un pare-feu nommé Netfilter. Celui-ci est extrêmement complet et permet de gérer un nombre très important de paramètres grâce aux différents modules Conntrack. Il se pilote intégralement via la commande iptables, et donc, sans interface graphique. Ce pare-feu est utilisé par tous les administrateurs réseau et constitue la réference en la matière, c'est pourquoi nous l'avons choisi.

Netfilter installe des accroches (ou hooks) dans la gestion des paquets réseau du noyau linux. Ces accroches sont spécifiques à chaque protocole réseau. Lorsqu'elles sont chargées, les tables de Netfilter inscrivent des chaînes sur ces accroches. Chacune de ces chaînes peut contenir des règles qui déterminent le traitement à effectuer sur les paquets qui traversent l'accroche. Ce sont ces règles que l'on va pouvoir modifier via la commande iptables.

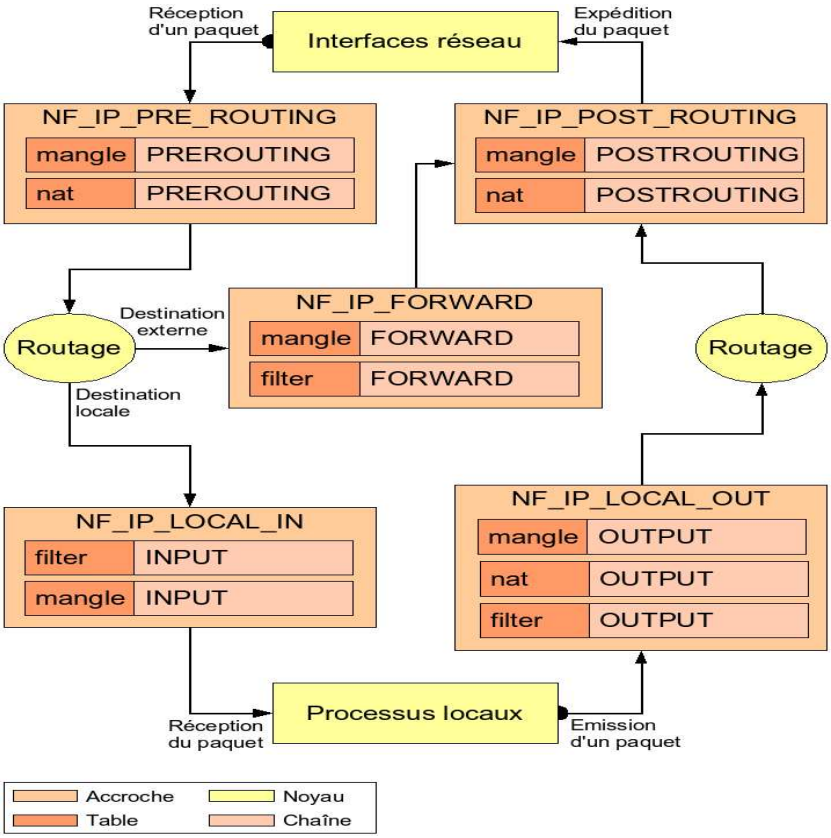

FIG. 3.1 – Système de chaînage netfilter

Netfilter est composé de trois tables : filter, NAT, et mangle. Filter sert a filtrer les paquets à l'entrée et à la sortie des processus destinataires via les chaines INPUT et OUTPUT, ainsi que ceux destinés à être routés vers une autre destination via la chaîne FORWARD. La table NAT est responsable de la traduction d'adresses et de ports et utilise les chaînes PREROUTING, POSTROUTING et OUTPUT. C'est cette table qui va nous permettre de masquer les IP des utilisateurs à l'extérieur du LAN, ou encore de router les paquets vers des ports différents pour les faire passer par un serveur mandataire écoutant un port particulier. Enfin, la table mangle est utilisée pour marquer des paquets. Marquage qui peut par la suite être utilisé par d'autres applications pour temporiser les paquets et faire de la qualité de service par exemple [AJ07].

Pour la configuration du pare-feu, nous avons opté pour une sécurisation maximale tout en gardant une possibilité d'actions relativement confortable. Ainsi, il est possible de surfer (via HTTP et HTTPS), envoyer et recevoir des mails simples ou sécurisés par SSL ou encore télécharger depuis un FTP[1]. Parallèlement, les tentatives d'établissement de connexion de l'extérieur vers l'intérieur sont

[1]File Transfer Protocol

rejetées si elle n'ont pas été demandées. Nous avons aussi fait en sorte que notre boîtier soit résistant au Ping Scanning[2] et à l'usurpation d'adresse IP[3]. Par exemple, une IP privée ne sera pas acceptée si elle vient de l'interface connectée à Internet. Pour cela, il nous a fallu rejeter les demandes de ping faites par diffusions (broadcast et multicast) et vérifier la cohérence des adresses par rapport aux interfaces utilisées. Si l'utilisateur du boîtier souhaite toutefois ouvrir d'autres ports, il en a la possibilité via l'interface graphique[4].

3.2 Antivirus

3.2.1 Antivirus

Un antivirus permet, comme son nom l'indique, de bloquer et supprimer les virus. La plupart de ces virus arrivent par le biais d'Internet sur les machines hôtes. Il paraissait donc indispensable d'intégrer la protection contre ces menaces dans notre boîtiers UTM. Par virus nous entendons les "vers" et "chevaux de troie", les "spywares" n'entrant pas dans cette catégorie. Notre première tâche était donc d'installer un antivirus car ces derniers représentent une menace directe pour le matériel.

3.2.2 Antispyware

L'antispyware doit être capable de détecter les spywares (en français logiciel espions). Ces derniers disposent eux aussi de signatures mais sont principalement destinés à récolter des informations privées diverses des utilisateurs de la machines infectée. Le spyware envoie aussi ces données sur des serveurs distants. Donc, en plus d'un antispyware, pour se prémunir au maximum il faudra interdire l'accès aux urls[5] connues pour être celles utilisées par les spywares.

3.2.3 ClamAV

Présentation

L'antivirus que nous avons choisi est ClamAV [Cla]. En effet, de nombreux antivirus existent sur le marché mais rares sont ceux totalement libres et gratuits. ClamAV[6] fait partie de ceux-là. Il a une grande particularité : il s'exécute en ligne de commande ou à travers une librairie insérée dans un programme. De plus de nombreux modules sont construits autour pour toutes sortes d'applications. Il s'adapte à un principe UNIX connu : KISS (Keep It Simple, Single) ce qui implique des applications spécialisées, simples d'emploi, en lignes de commandes et qui peuvent servir de base à des systèmes plus complexes.

[2] Technique consistant à effectuer des diffusions de ping dans un réseau pour détecter les machines présentes

[3] aussi appelée IP address spoofing

[4] Se reporter à la section concernée pour plus de détails

[5] adresses Internet

[6] Il tire son nom de "clam" qui en anglais veut dire "moule". Les moules filtrent l'eau de mer pour se nourrir, et ClamAV filtre les fichiers pour identifier les virus.

Entre autres, il peut scanner les virus dédiés au système Windows©ce qui lui confère un atout. ClamAV utilise une base de signatures qui contient en fait des sections de code (binaire) spécifiques aux virus. Il prend l'exécutable et le compare à l'aide de machine à automate fini à sa base de données[7]. Ce principe lui donne une rapidité d'exécution importante, ce qui dans notre cas est non négligeable. Certains virus avancés, comme les virus polymorphiques[8], nécessitent une approche différente. La plupart des scanneurs antivirus essayent d'émuler le code pour les détecter. Cette technique s'avère inefficace. ClamAV propose une détection basée sur des algorithmes dédiés. Cette méthode rend le "scann" plus long en général, mais permet de détecter beaucoup plus précisement et rapidement ce genre de virus. Pour le moment, les algorithmes doivent être insérés au niveau des applications utilisant la librairie de ClamAV[9]. Mais cet atout devrait être applicable plus concrètement dans un futur proche. Aussi, il peut regarder à l'intérieur des archives compressées suivant un nombre de niveaux de compression défini. Notamment, il gère les fichiers zip, RAR, CHM, tar, SIS, gzip, bzip2, PST, HTML, PDF, MS Office, JPEG, GIF, RIFF, etc. ...[10][Koj07]

Autre atout, il dispose dans sa base de signatures de spyware ce qui fait de lui un antispyware aussi. En effet, un spyware dispose lui aussi d'une signature propre tout comme un virus, constituant en fait une section de code binaire identifiable.

Installation et configuration

Pour l'installation, rien de plus simple. En effet, vu que nous utilisons une installation de type Debian, il suffit de faire un appel au gestionnaire de package aptitude ou apt-get avec le nom de ClamAV. Les packages à installer sont : clamav, clamav-deamon, libclamav-dev[11]. Nous avons préféré, quand nous le pouvions, utiliser au maximum ce gestionnaire de package. Ainsi, les mises à jour et installations sont simplifiées. Au niveau de la configuration de ClamAV, nous avons laissé celle par défaut. Nous avons dû ensuite le configurer pour apporter des limites au logiciel (limites concernant les niveaux de descente dans les archives compressées, etc. ...).

Problème

Le problème que nous avons rencontré : nous ne pouvions scanner que des fichiers. Or, les données que nous souhaitions analyser étaient principalement des flux. D'où l'utilité de passer par un serveur "Proxy" ou "Mandataire", car ces derniers embarquent un cache stockant les fichiers.

[7] À ce jour, la base de données contient 549 481 signatures de virus.
[8] Ces virus sont appelés ainsi, car il change leur code ce qui rend impossible la détection par signature
[9] Voir HAVP plus loin
[10] Nous avons là juste un éventail parmi les types de fichiers les plus connus.
[11] libclamav-dev est utile pour scanner les flux : voir partie sur HAVP

3.3 Antispam

3.3.1 Les Courriels

Les e-mails ou courriels sont à l'origine de beaucoup de désagréments pour les utilisateurs. En effet bien que très pratiques, ils constituent une menace s'ils sont utilisés à mauvais escient. Ainsi les spams (ou pourriels[12]) constituent la plus imposante menace (mais la moins dangereuse). Ensuite, les virus contenus dans les pièces jointes sont certes moins courrants, mais bien plus dangereux. Il convient donc de contrôler au maximum ces éléments avec des antispams et antivirus.

3.3.2 Antispam : SpamAssassin

Présentation

SpamAssassin permet de détecter si un mail est un spam ou pas. Son utilisation est donc très importante. Le problème avec les spams, c'est qu'un programme ne peut pas être sûr qu'un courriel est non-sollicité. Il ne fait que des hypothèses. C'est pourquoi SpamAssassin s'occupe juste de renommer l'objet du mail par l'entête "**********SPAM**********"[13]

Installation et configuration

L'installation de SpamAssassin se fait évidemment par le biais de l'installeur de packages. La configuration porte surtout sur le niveau de détection du spam sur un courriel. En effet, SpamAssassin atribue un score à un mail en fonction de paramètres prédéfinis (présence de slogan, d'url, etc. ...). Ensuite, il suffit de définir un score limite en dessus duquel tout mail est considéré comme spam.

3.4 Serveurs mandataires

3.4.1 Serveur Mandataire HTTP : HAVP

Le protocole HTTP

Le protocole HTTP est le protocole utilisé pour la consultation de sites web avec un explorateur Internet. Il permet aussi le téléchargement simple d'images et de fichiers divers. Le port public par défaut de connexion des clients sur le serveur porte le numéro 80.

Présentation

HAVP [HAV] est un serveur Proxy spécialement dédié au scan antivirus sur des flux HTTP. C'est avec le HTTP que la majorité des virus se transmettent par l'Internet. HAVP est spécialement dédié à une utilisation avec ClamAV et plus exactement sa librairie. Il peut être couplé avec d'autre antivirus, mais ClamAV reste l'antivirus pour lequel il a été conçu. La particularité de HAVP est qu'il permet à l'antivirus de commencer à scanner les fichiers en cours de

[12] Courriel publicitaire non désiré
[13] On peut spécifier évidemment l'en-tête que l'on souhaite.

téléchargement pour limiter au maximum le temps d'attente sur la machine hôte.

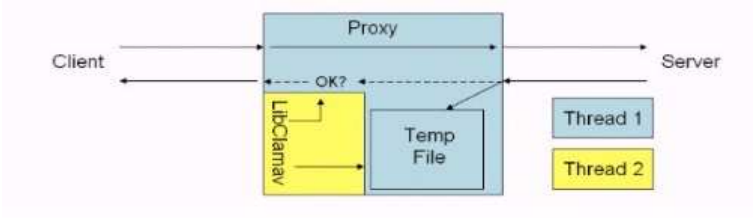

FIG. 3.2 – Principe de fonctionnement HAVP

Installation et configuration

Son installation est plus délicate car les packages ne sont pas présents dans les sources de base de la Debian "etch". Dans un premier temps, nous avons dû donc installer les packages pour la compilation puis télécharger les sources avant de les compiler. Ensuite, nous avons changé de version et sommes passés à la version toute nouvelle de Debian nommée "Lenny". L'installation s'en trouvait simplifiée, car maintenant disponible par le gestionnaire de package (le package à installer se nomme havp). Puis, il nous a fallu configurer le serveur de telle sorte qu'il utilise un proxy parent[14], définir une partition en "mandatory locking[15]" (indispensable pour le fonctionnement de HAVP) et enfin rediriger le port 80 des interfaces locales sur le port qu'écoute le serveur, c'est à dire 8080. De nombreuses autres configurations sont nécessaires, notamment définir l'antivirus choisi : c'est-à-dire ClamAV. Ensuite, il nous a fallu réfléchir aux limites du système et à la politique sécuritaire en matière du protocole HTTP. Ce que nous entendons par là, c'est tout simplement la taille maximum d'un fichier à scanner, à télécharger, le nombre de fichiers maximum qu'il est possible de télécharger en parallèle, le nombre de niveaux d'une archive que l'on va vérifier, etc. ...

[14]SQUID décrit plus loin

[15]Le mandatory locking permet d'allouer au système la charge de vérification des blocages en accès par plusieurs processus non coopératifs sur un fichier. Ainsi c'est le kernel linux qui va s'en charger. Pour faire ce montage, il nous a fallu modifier le fichier /etc/fstab qui contient toutes les informations de montages des périphériques.

3.4.2 Serveur mandataire FTP : FROX

Le protocole FTP

Le protocole FTP[16] sert au transfert de fichiers. Il a la particularité de se voir doté de deux canaux de communication. En effet, à la différence d'une connexion client/serveur classique (HTTP par exemple), FTP va avoir un canal de transfert des requêtes et commandes (sur le port public 21), et un canal de transfert des données (sur le port public 20). Ceci pose une difficulté connue pour les pare-feu et proxy. Il existe de plus deux types de FTP, le passif et l'actif. En mode actif, c'est le client qui détermine le port de connexion à utiliser pour permettre le transfert de fichiers, alors qu'en mode passif, c'est le serveur qui détermine ce port et le transmet au client. Seul le passif est utilisé à ce jour.[Com06]

Présentation

FROX[FRO] est un serveur Proxy spécialement dédié au protocole FTP. FTP est un protocole de transfert de fichiers, d'où la nécessité de scanner ces fichiers car ces derniers peuvent contenir des virus. Le cache d'un serveur FTP n'a d'utilité que pour scanner des fichiers car il est très rare contrairement à HTTP de télécharger plusieurs fois le même fichier. C'est un problème, car devant cette particularité, rares sont les proxys FTP et, de plus, rares sont les proxys FTP qui font antivirus (à ce jour, nous ne connaissons que Frox). L'installation est très simple car le Frox est accessible depuis le gestionnaire de packages. Il suffit ensuite de le configurer et là, c'est une autre difficulté qui se présente à nous.

Difficultés

De nombreuses difficultés sont apparues lors du taitement du protocole FTP. D'abord au niveau du Firewall, puis au niveau de Frox. La configuration de Frox contient de multiples particularités nécessitant de connaître le protocole FTP correctement. En outre, Frox est un projet abandonné, il n'y a donc pas beaucoup de documentation présente sur Internet et encore moins de communauté active récente. Nous n'avons pas réussi à le rendre opérationnel. En effet, nous avons apparemment réussi à le paramétrer mais le souci est intervenu lors de l'utilisation de certains clients FTP. Alors que le client en ligne de commande wget fonctionnait correctement, le client filezilla ne fonctionnait pas. Nous ne pouvons expliquer ces erreurs que par un problème interne à Frox. Nous avons choisi de ne pas vérifier ce protocole pour le moment.

3.4.3 Serveur mandataire de supervision : SQUID

Présentation

SQUID [SQU] est un serveur proxy régulièrement utilisé. Il fait office de référence en la matière. Il écoute le port 3128.

[16] File Transfert Protocol ou Protocole de Transfert de Fichier

Objectifs

Outre son cache, son utilité réside surtout dans la gestion des connexions. En effet, on peut décider avec un tel serveur de restreindre des connexions au niveau applicatif. C'est-à-dire restreindre tel ou tel type de vidéo téléchargée, tel ou tel créneau horaire, etc. ... C'est là sa principale utilité. En effet, les menaces peuvent être réduites si l'on interdit les accès au web non nécessaires ou à des heures où ils ne devraient pas avoir lieu. Un autre aspect est la gestion des logs[17], qui permettent d'inscrire dans la durée toutes les actions des utilisateurs du web. On a ainsi une trace de tout ce qui a été fait et on peut déceler une anomalie plus rapidement.

Installation et configuration

L'installation est très simple, car elle s'effectue comme précédemment par l'intermédiaire du gestionnaire de package. Mais la configuration est plus difficile, car le fichier de configuration contient environ 200 paramètres. Heureusement il est abondamment commenté, ce qui facilite cette configuration. Le lien entre HAVP et SQUID se fait simplement dans la configuration de HAVP en définissant un serveur mandataire parent (ou Proxy Parent).

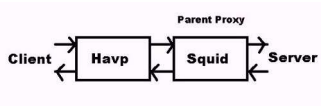

Fig. 3.3 – Chaînage HAVP - SQUID

Ainsi, on remarque que l'on rejoint bien l'architecture présentée au début. Le seul souci que l'on obtient en utilisant cette dernière, c'est que les logs enregistrés par SQUID ne permettront pas de connaître le client qui est à l'origine des requêtes. Heureusement, les logs de HAVP quant à eux, le permettront. Pour ce qui est des choix en matière de restrictions au niveau applicatif, nous avons décidé de le laisser aux futurs utilisateurs. Ils utiliseront le site de configuration[18] pour restreindre des contenus prédéfinis (images, vidéo, flash, etc. ...)[cB08].

3.4.4 Serveurs mandataires de gestion de courriel

Objectifs

Il existe trois protocoles principaux pour les courriels : IMAP, POP3 et SMTP. Les rendre plus sûrs est évidemment un objectif lié à notre boîtier UTM.

POP3

Présentation Le protocole POP3 se décline sous deux formes. Une version sécurisée (transferts cryptés) et une version non sécurisée. Il permet de récupérer ses mails sur un serveur Mail. Le POP3 sécurisé est, au même titre que le

[17] Fichier texte qui enregistre toutes les actions du logiciel
[18] Voir chapître sur le site intranet

HTTPS, non vérifiable. Ainsi nous ne occuperons que du proxy pour le POP3 non sécurisé.

P3Scan P3Scan est un proxy léger qui permet d'utiliser un antispam et un antivirus. Conçu plus spécialement pour utiliser SpamAssassin et ClamAV. Son installation se fait à travers le gestionnaire de package (paquet : p3scan), et sa configuration est très rapide.

IMAP

Présentation Le protocole IMAP permet de récupérer, au même titre que POP3, ses e-mails sur un serveur de courriels. Il n'existe pas à notre connaissance de serveurs mandataires destinés au protocole IMAP. P3Scan devrait en être pourvu dans un futur proche.

SMTP

Présentation Le protocole SMTP sert à la communication entre serveur Mail. Un utilisateur se sert de ce protocole pour envoyer un message directement au serveur de courriel. Il ne semble pas indispensable de vérifier ce protocole, vu qu'il n'y a pas de serveur courriel sur le réseau à protéger. Seulement, il ne faut pas que notre réseau soit source de nuisance pour autrui. C'est pourquoi nous avons décidé de contrôler ce flux de données.

Retour sur P3Scan P3Scan permet aussi de scanner ce protocole. Il utilise les outils SpamAssassin et ClamAV comme pour POP3 afin de détecter les malveillances.

Fonctionnement de P3SCAN

Installation et configuration Pour utiliser P3SCAN, il nous suffit en plus des configurations de rediriger les flux sur le port de P3SCAN à l'aide de iptables, quel que soit le protocole de ces flux (SMTP ou POP3). Ensuite, au niveau des configurations, il faut lui préciser l'antivirus et l'antispam à utiliser alors notre contrôle est opérationnel. La réflexion pour appliquer une politique de sécurité adéquate ne concerne pas directement P3SCAN mais plutôt les modules SpamAssassin et ClamAV décrits précédemment qu'il utilise.

Tests Nous avons pu facilement tester l'appel de P3Scan à l'antispam. En effet, simuler un spam ou en recevoir des vrais est très simple. Par contre, le test de l'antivirus n'a pu s'effectuer concrètement. En effet, tous nos fournisseurs de comptes mails sont pourvus par défaut d'antivirus. Il aura donc été impossible pour nous d'envoyer de fausses signatures de virus.

3.5 Filtrage URL

3.5.1 Objectifs

L'objectif de ce filtre est de permettre une restriction des connexions sur des URLs bien connues pour être sources d'ennuis (notamment pour bloquer

les sites de spywares), et sur des URLs souhaitées par le futur utilisateur. En restreignant les urls, le "futur utilisateur" pourra être sûr que les spywares qui auront passé l'antispyware, ne pourront communiquer directement (si leur site est évidemment connu).

3.5.2 Retour sur HAVP

Il existe un module du serveur mandataire SQUID appelé SQUIDGuard qui permet de filtrer les URLs[19]. Seulement, HAVP le fait aussi. Du coup plutôt que d'installer un programme supplémentaire, nous avons décidé d'utiliser cette fonctionnalité d'HAVP. HAVP intègre des fichiers "Blacklist" et "Whitelist" qui permettent de spécifier respectivement les pages qui doivent être refusées avant d'être scannées et celles qui ne doivent pas être scannées du tout. Nous avons créé un petit script shell qui permet d'actualiser ces tables avec les listes publiques de sites référencés comme étant des sites de phishing ou de spywares. Nous pouvons agrandir ces tables avec d'autres sites pour constituer par exemple un contrôle parental.

3.6 Sonde de détection et de prévention d'intrusion

3.6.1 Présentation

La détection d'intrusion est le procédé de surveillance des évènements sur un réseau ou un ordinateur personnel. Ces évènements sont analysés afin de déceler des signes possibles d'incidents. On appelle incident des violations ou des menaces imminentes de violations de la politique de sécurité. Ces incidents sont dus à des attaques intentionnelles ou non. Une attaque peut être une injection de code SQL dans un formulaire Web, un scan de port, ou encore une tentative d'accès non autorisée à un système donné.

Une sonde de détection d'intrusion (Intrusion Detection System) est un logiciel automatisant le procédé de la détection d'intrusion. Ce type de sonde permet d'être alerté dans le cas d'une attaque. Une sonde de prévention d'intrusion (Intrusion Prevention System) possède les même capacités, mais peut aussi arrêter un éventuel incident. Les IPS se placent en coupure du réseau, comme pour un pare-feu ce qui leur permet de bloquer le trafic, tandis que les IDS peuvent se placer en parallèle. Le terme IDPS, fait référence aux deux technologies : les IDS et les IPS.

On distingue 4 grands types d' IDPS :

- **Host-IDPS** (IDPS basés hôtes)
  Ce genre d'IDPS traite le flux et les caractéristiques d'un hôte unique, par exemple les systèmes de journalisation ou les processus en cours. On les utilise généralement sur des sites critiques comme les serveurs en accès

[19] adresses Internet

public contenant des informations sensibles.

- **Network-IDPS** (IDPS basés réseaux)
  Les N-IDPS servent à surveiller le flux sur des segments précis du réseau. Ils analysent les protocoles de la couche réseau et application afin d'identifier des activités suspectes. Généralement, on les trouve sur la jonction entre deux réseaux, au niveau du pare-feu ou encore des routeurs.

- **Network Behavior Analysis system** (NBA)
  Les NBA, analysent le trafic réseau afin d'y déceler des flux inhabituels. Ce qui peut être une attaque de déni de service distribuée (DDoS), certaines formes de malwares tels que les vers ou les backdoors, ou alors des violations de la politique de sécurité.

- **IDPS sans fils** (wireless IDPS)
  Les IDPS sans fils sont utilisés pour la surveillance du trafic sur les réseaux Wifi. Ils peuvent identifier les activités suspectes sur la couche réseau mais pas sur les autres couches supérieures.

Dans notre cas on s'interessera uniquement au N-IDPS. En effet le but de notre sonde est d'analyser le trafic entrant et sortant du réseau interne. L'IPS ayant des fonctionalités plus intéressantes pour notre boîtier, nous opterons pour cette solution. Snort est logiciel libre qui est capable de mettre en relation le flux réseau avec des signatures tout comme un antivirus le ferait. Nous utiliserons une variante qui s'appelle Snort Inline qui est un projet à part entière qui permet de bloquer le trafic en cas d'attaque.

3.6.2 Détection d'intrusion

Snort Inline[Inl] tout comme Snort[SNO] est divisé en trois parties principales : la capture des paquets, les signatures, et les alertes. La capture des paquets se fait grâce à la librairie libipq de NETFILTER. Les signatures font l'association entre un flux donné et une action. Dès qu'il y a une correspondance entre un flux et une signature cela produit une alerte. Dans certains cas, il peut y avoir des alertes sans qu'il y ait d'attaques ce sont des faux positifs.

Positionnement de la sonde

Le choix de l'emplacement physique de la sonde sur le boîtier est primordial concernant son efficacité. La sonde peut se placer par exemple devant le pare-feu. Dans cette position, elle pourra détecter l'intégralité des attaques visant le réseau et pourra ainsi analyser le trafic que laisse passer ou non le pare-feu. Par contre, cette position pose un inconvénient majeur car Internet regorge d'activités malveillantes qui ne nous concernent pas forcément. Ce qui augmente considérablement la charge de travail car il faut alors faire le tri de faux positifs de vraies attaques. On placera notre sonde derrière le pare-feu car seuls les paquets qui arrivent sur le réseau interne nous intéressent. De cette façon, on peut analyser le trafic qui a été filtré par NETFILTER.

Une fois le paquet capturé via NETFILTER il subira toute une série d'opérations afin d'empêcher l'intrusion sur le réseau. La première consiste à réassembler les paquets à la manière du système d'exploitation cible grâce au pré-processeur frag3. En effet si un paquet trop grand est émis sur le réseau, il va être fragmenté en plusieurs paquets. Le motif d'une attaque peut donc se retrouver découpé en plusieurs morceaux qui ne se réassembleront pas de la même façon suivant la pile du système d'exploitation sur lequel est installé le NIDPS comparé au système cible.

Il existe toute une série de pré-processeurs activables ou non. Ils permettent d'étendre les fonctionnalités de Snort Inline. Par exemple, portscan permet d'enregistrer le début et la fin d' un scan de port provenant d'une adresse IP. Les pré-processeurs permettent aussi le décodage du paquet. Il s'agit d'éviter que l'attaque ne corresponde pas aux signatures définies à cause de la différence d'encodage de caractères par exemple. L'algorithme utilisé pour la recherche de chaînes de caractères est Aho-Corasick NFA car il a un bon rapport mémoire consommée / performances.

Les signatures ont un format bien spécifique. On distingue deux parties, l'en-tête et les options. L'en-tête est constituée de l'action à effectuer s'il y a une correspondance, le protocole, l'adresse IP, le masque réseau et le numéro de port de la source et la destination. Les actions que Snort permet d'effectuer sont les suivantes :

- log - journalise le paquet
- alert - génère une alerte puis journalise le paquet
- pass - ignore le paquet
- activate - génére une alerte puis active une autre signature dite dynamique
- dynamic - reste inactif jusqu'à son activation par une autre signature

Les actions suivantes ne sont possibles qu'avec Snort Inline :

- drop - renvoie le paquet à IPTABLES pour qu'il le jette et le journalise
- reject -
- sdrop - même action que drop sans la journalisation

Dans notre cas toutes les signatures ont pour action drop, ce qui nous permet de pouvoir supprimer les attaques.

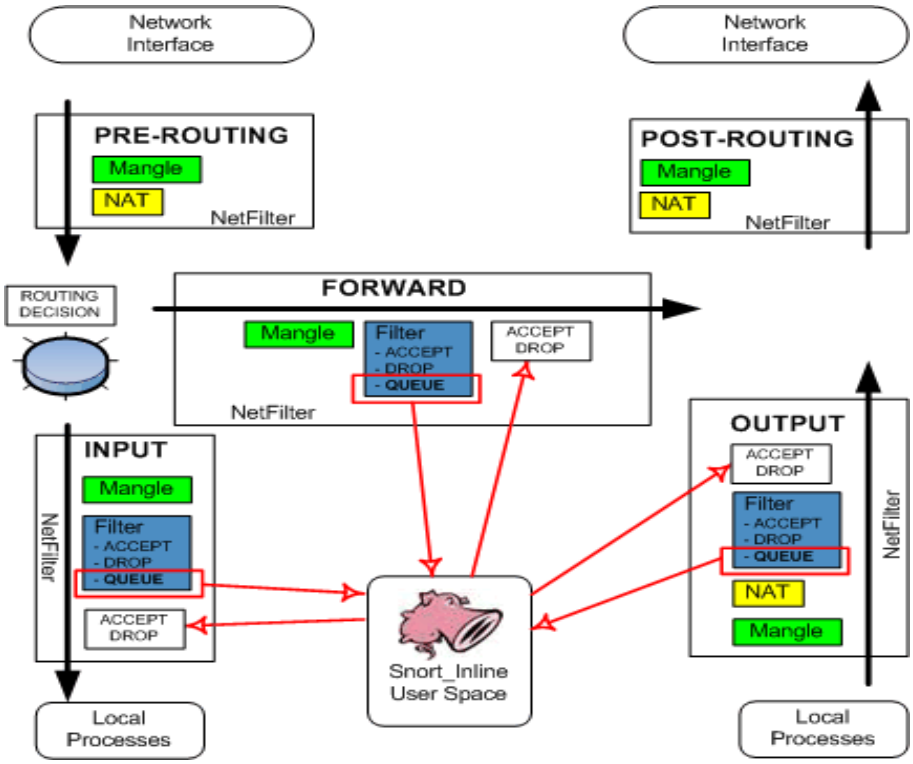

FIG. 3.4 – Fonctionnement de Snort Inline

Les options les plus intéressantes sont celles de type *payload*. Ce sont des options qui permettent de regarder les données à l'intérieur du paquet.

Les signatures Snort sont installées et mises à jour à l'aide du script oinkmaster. Les alertes sont activées sur deux modes, le mode full enregistre l'en-tête complète du paquet et le mode fast, plus réduit n'en garde que certains champs. Il y a deux types d'alertes, celles générées par les signatures et celles qui le sont par les pré-processeurs.

3.7 Authentification des postes : FreeRadius

3.7.1 Introduction

La plupart des intrusions systèmes proviennent de réseaux externes comme l'Internet mais il ne faut pas ignorer ceux qui proviennent de l'intérieur du réseau local.
Actuellement, les réseaux locaux sont évolués et l'accès à ces réseaux se fait via un réseau filaire ou sans fil. Si on sait parfaitement où commence et où finit un réseau filaire et qu'il faut se connecter sur une prise physique pour avoir une

chance de l'écouter, la difficulté avec les réseaux sans fil réside dans le fait que leur enveloppe est floue. Il n'y a pas de limites imposables et contrôlables. Une borne Wi-Fi émet dans toutes les directions et aussi loin que porte son signal. Bien souvent, ses limites dépassent les bâtiments de l'établissement, donc des espions peuvent s'installer et intercepter les communications ou s'introduire dans le réseau et l'utiliser à leur profit.
Parmi les aspects de sécurité du réseau local qu'il faut prendre en compte, on trouve l'autorisation d'un poste à effectuer certaines activités pour une durée déterminée, tous ces contrôles augmente considérablement la sécurité du système.
Des protocoles ont été développés pour pallier ces problèmes, parmi eux on trouve le protocole Radius (Remote Authentication Dial In User Service) qui est le plus répandu dans le monde des entreprises.

3.7.2 Radius

Le protocole Radius[Bor06] est en effet un protocole d'authentification, d'autorisation et de comptabilité, il est normalisé par l'IETF dans deux RFC : RFC 2865 (RADIUS authentification) et RFC 2866 (RADIUS accounting).
Il suit le modèle client / serveur, où chaque poste voulant entrer dans le réseau doist transmettre une requête d'accès à un client, on l'appelle NAS (Network Access Server), ce client se charge de demander les informations identifiant l'utilisateur, et effectue ensuite des échanges avec le serveur.

3.7.3 Fonctionnement

La première tâche du serveur Radius est d'authentifier les requêtes qui lui parviennent d'un poste client, c'est-â-dire d'engager des échanges avec lui pour obtenir la preuve qu'il est bien celui qu'il prétend être. On peut distinguer deux types généraux de preuves : par mot de passe ou bien via certificat électronique.
La deuxième tâche est de délivrer les autorisations, en général, il s'agit des attributs envoyés à l'équipement réseau (borne, commutateur) sur lequel est connecté le poste de travail.
La dernière composante de Radius est la comptabilité qui permet d'enregistrer un certain nombre d'indicateurs liés à chaque connexion (heure, adresse de la carte Ethernet du client,) à des fins de traitements divers et de contrôles.
Pour les réseaux sans fil, le serveur Radius possède une autre mission : amorcer les algorithmes de chiffrement de données, c'est-à-dire des communications que le poste de travail établira après la phase d'authentification.

3.7.4 Types d'authentifications

Parmi les moyens que Radius utilise pour authentifier un poste de travail :
- Authentification avec l'adresse Ethernet (adresse MAC)
  L'adresse MAC de la carte Ethernet du poste de travail identifie ce dernier. Cette adresse MAC n'est pas une preuve absolue d'identité puisqu'il est relativement facile de la modifier et d'usurper l'identité d'un autre poste

de travail. Néanmoins, sur un réseau filaire, cette adresse peut être suffisante puisque, pour tromper le système d'authentification, il faudra tout de même pénétrer sur le site, connaître une adresse MAC valide et réussir à s'en servir. Même si on peut imaginer qu'une personne décidée peut y arriver, cette solution est suffisante si on considère qu'il s'agit là d'une première barrière. En revanche, si on souhaite une authentification très forte il faudra utiliser une autre méthode, à savoir 802.1X et EAP. Dans le cas du sans-fil, l'authentification par adresse MAC est fortement déconseillée comme unique moyen, car l'adresse MAC circule toujours en clair et n'importe qui, écoutant ce réseau, même sans accès physique, peut capter des adresses MAC et s'en servir très facilement pour s'authentifier. Cet inconvénient est moindre en filaire car le périmètre est complètement déterminé et une présence physique dans les locaux est nécessaire. Ce type d'authentification est appelé Radius-MAC ou encore MAC-based.

- Authentification par identifiant et mot de passe
  Plusieurs protocoles peuvent être mis en oeuvre pour assurer une authentification par identifiant et mot de passe. Cependant, il convient d'éliminer ceux pour lesquels le mot de passe circule en clair sur le réseau ou bien est stocké en clair dans la base de données. Le protocole 802.1X nous permettra de mettre en oeuvre des solutions (EAP/PEAP ou EAP/TTLS) qui permettront de résoudre ces problèmes. Comme les utilisateurs sont déjà confrontés à la nécessité de posséder de multiples mots de passe pour de multiples applications, il sera intéressant de réutiliser une base existante comme une base LDAP.

- Authentification par certificat électronique
  Ce type d'authentification consiste à faire présenter par le client un certificat électronique dont la validité pourra être vérifiée par le serveur.
  Il peut s'agir d'un certificat appartenant à un utilisateur. Dans ce cas, on parlera d'authentification par utilisateur, mais il peut également s'agir d'un certificat machine qui sera alors lié à la machine.

3.7.5 Principe de l'authentification 802.1X (EAP)

Cette authentification est plus compliquée à mettre en oeuvre mais plus sûre que Radius-MAC, tout d'abord, un logiciel particulier sera indispensable sur le poste de travail. Ce logiciel est appelé supplicant, c'est lui qui va envoyer vers Radius les éléments d'authentification (certificat, identifiant, mot de passe). Pour que le serveur Radius interroge sa base de données, il a besoin d'un identifiant qu'il utilise comme point d'entrée. Le serveur accepte ou refuse l'authentification et renvoie sa réponse au supplicant et à éventuellement l'équipement réseau (commutateur) sur lequel est connecté le poste de travail pour qu'il autorise les communications venant de ce poste.

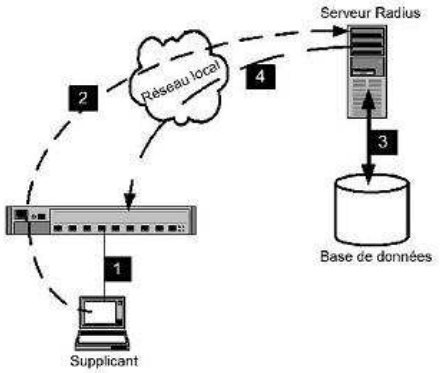

FIG. 3.5 – Cas général du fonctionnement de Radius

3.7.6 FreeRadius

FreeRadius[Fre] est une implémentation de Radius élaborée, c'est un projet Open Source sous licence GPL.
FreeRadius est très répandu dans le monde et doit son succès à sa compatibilité avec un grand nombre de standards couvrant les systèmes d'exploitation, les protocoles et les bases d'authentification.

3.7.7 Problèmes rencontrés lors de la mise en oeuvre

Le temps nous a manqué pour mettre en oeuvre un serveur Radius vu le nombre de technologies qu'il fallait installer avec :

- OpenSSL pour générer les certificats
- Annuaire LDAP
- Configurer des supplicants et les tester
- Impossible de réalisé une interface graphique qui englobe toutes les configurations possibles car il existe un très grand nombre de paramètres et de fichiers de configuration des différentes composantes.

Chapitre 4

Monitoring du boîtier et interface

4.1 Monitoring

Le monitoring est une activité de surveillance et de mesure d'une activité informatique, dans notre cas nous voulons récupérer des mesures de différentes ressources de la machine (le boîtier UTM). MRTG[Cal][MRT] est un logiciel qui permet de construire des pages HTML avec des graphiques représentant les débits du réseau, et aussi permet de superviser les ressources système comme connaître la RAM disponible, la charge CPU, la taille du SWAP occupé, etc. ... Il génère un graphique quotidien, hebdomadaire, mensuel et annuel, ce qui permet une bonne vue d'ensemble. Il est ainsi très aisé de détecter une saturation du boîtier, que se soit au niveau du trafic réseau que de la charge du processeur.

4.1.1 Fonctionnement

La récupération des données se fait en utilisant le protocole SNMP pour tout ce qui concerne le trafic réseau, et du répertoire /proc qui est en fait un espace RAM dans lequel il y a une infinité de compteurs, d'indicateurs, de variables et drapeaux ce qui permet de récupérer beaucoup d'informations sur la machine. Le protocole SNMP (Simple Network Management Protocol) est destiné à récupérer des informations sur des équipements informatiques, à distance ou localement.Il est utilisé généralement par les administrateurs réseau pour gérer les équipements du réseau, superviser et diagnostiquer des problèmes matériels à distance. Dans notre cas on s'en sert seulement pour récupérer les données locales (le boîtier UTM), tout en interdisant tout accés distant à ces données, puisqu'elles sont des informations sensibles et que la machine serait vulnérable si elles étaient récupérées par une personne malveillante. MRTG récupère ces données et en crée les diagrammes graphiques.

4.1.2 Installation et configuration

En utilisant le gestionnaire de package apt-get on installe les packets suivants : snmp, mrtg et snmpd.

La configuration est simple, il faut autoriser l'accès en lecture des données snmp et changer quelques paramètres dans snmpd.conf.

La configuration du MRTG se fait en quatre étapes :

- Création du fichier de configuration par cfgmaker.
- Choix des données à afficher.
- Génération du fichier html avec indexmaker.
- Exécution de mrtg touts les cinq minutes (utilisant crontab) pour actualiser les données.

4.1.3 Objectif

Ce module permet de procurer plus de confort à l'administrateur, et l'aide aussi à choisir une stratégie plus convenable pour son réseau, il peut alors :

- Surveiller les différentes ressources du boîtier, évitant ainsi sa surcharge
- Surveiller les débits
- Avoir une idée globale des variations du trafic réseau pendant de longues périodes

4.2 Site intranet

4.2.1 Présentation

Le but du site web est de permettre aux futurs utilisateurs de configurer facilement leur boîtier UTM sans nécessiter une grande expérience en informatique. L'autre solution est la connexion en SSH sur la machine. Cette dernière n'est pas accessible facilement à une personne ne disposant que de bases en informatique, c'est pourquoi le site web est primordial.

Bien que le site semble être l'élément ayant le plus d'importance dans notre projet, il n'en est rien. En effet, la configuration des logiciels aura nécessité bien plus de travail que la programmation de ce site. Ce dernier permet d'apporter la touche finale à notre boîtier UTM, et c'est ainsi qu'il gagne son importance.

4.2.2 Maquette

La réalisation d'une maquette est intervenue lors de la conception et de l'installation des applications. Nous sommes partis du principe que chaque application devait être configurée par le site. Donc, nous avons chacun oeuvré à la conception de cette maquette. Ce que nous entendons par maquette est en fait des dessins représentant l'interface, et permettant surtout de se poser les questions quant aux manières de configurer les diverses fonctions du boîtier UTM.

4.2.3 Exécution de scripts

Pour que le site puisse modifier les caractéristiques interne du boîtier UTM, il nous a fallu concevoir des scripts exécutables qui effectuent eux-mêmes les opérations. L'exécution de script aura nécessité d'installer le programme sudo.

Il permet de définir des droits root[1] temporaires à des utilisateurs ou à des groupes. En effet, les commandes à exécuter par le site sont des commandes qui ne peuvent l'être que par un utilisateur root. Donc, nous avons mis en place des scripts shell (en dehors de l'espace défini pour le site évidemment), qui appelent des commandes en mode sudo. Le site quand à lui appele ces script en PHP avec la fonction "exec". Ensuite, il n'a plus fallu que créer tous les scripts nécessaires. Ceci nous donne une relative sécurité dans le sens où si quelqu'un arrive à pirater le site, il ne pourra en aucun cas exécuter des choses non définies dans le site.

4.2.4 Résultat

Cryptage HTTPS

Le résultat obtenu est concluant. Nous pouvons à l'aide de ce site modifier les caractéristiques internes du boîtier UTM. La connexion au site s'effectue en HTTPS depuis le réseau local. Nous préservons ainsi le boîtier UTM des piratages internes au réseau. En effet, dans quatre-vingts pour cent des cas les menaces directes sur un système sont internes à la structure. Ce cryptage n'est présent que pour permettre à l'administrateur qui va disposer de ce boîtier UTM d'être sûr que personne ne peut le reconfigurer derrière lui. La mise en place de ce cryptage s'est effectuée simplement par configuration d'Apache[2]. Le seul souci qui, en fait n'en est pas un, est que le protocole HTTPS nécessite une authentification auprès d'une structure internationnale de contrôle. Cette société délivre des certificats valides pour SSL. Or nous ne pouvions demander ce certificat auprès d'une telle structure dans le cadre de ce TER. Nous en avons donc créé un par le biais du logiciel OpenSSL, mais les navigateurs Internet spécifient le site comme un site qui veut s'identifier lui même. Ce souci peut être contourné facilement en demandant à l'explorateur d'ignorer le certificat.

Les fonctionnalités du site

Les fonctionnalités implantées dans le site sont les suivantes :
- Arrêt/Redémarrage de l'UTM
- Configuration du parefeu (ajout/suppression de ports de connexion du réseau local vers l'Internet).
- Configuration du serveur proxy Squid (filtrage des connexions).
- Configuration du filtre URL.

L'interface

Nous avons voulu garder une interface assez sobre. Quand l'utilisateur se connecte au site, il doit bien évidemment entrer un nom d'utilisateur et un mot de passe (cf figure Page de connexion).

[1] Super-utilisateur, utilisateur pouvant effectuer toutes les opérations sur le système

[2] Le serveur Apache dispose de nombreux "modes" qui permettent de lui ajouter des fonctionnalitées. Par exemple : le cryptage SSL pour l'HTTPS

FIG. 4.1 – Page de connexion

Il accède ensuite à l'interface de configuration. Les pages de la configuration du parefeu, du serveur mandataire (proxy), et du filtre URL sont présentées ci-après.

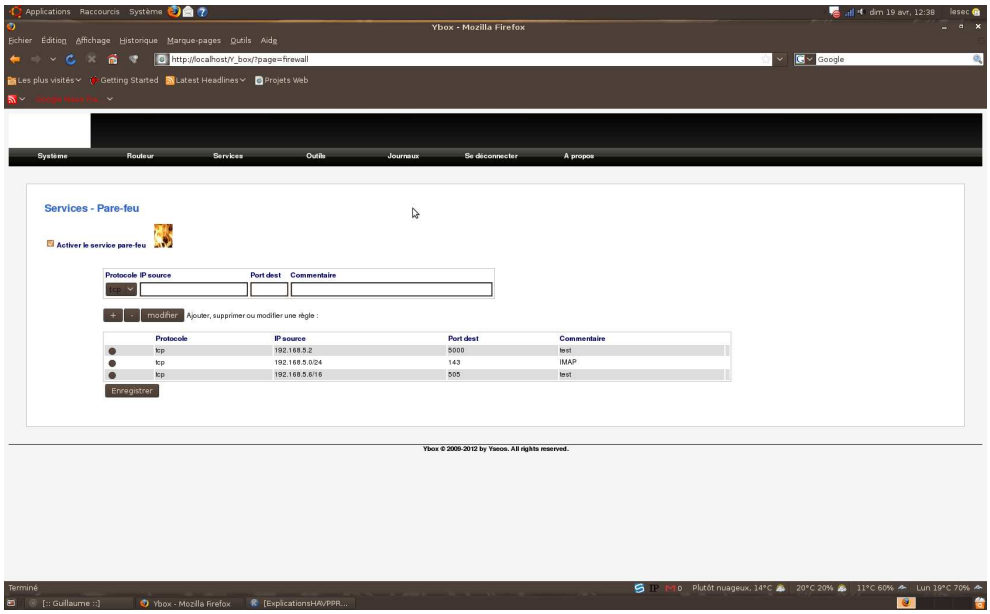

FIG. 4.2 – Page de configuration du parefeu

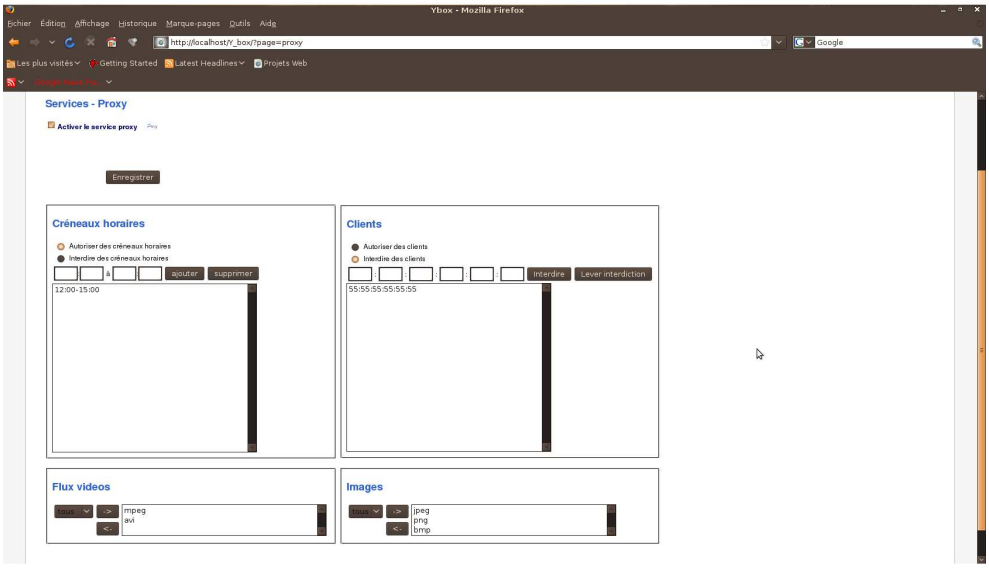

FIG. 4.3 – Page de configuration du serveur mandataire

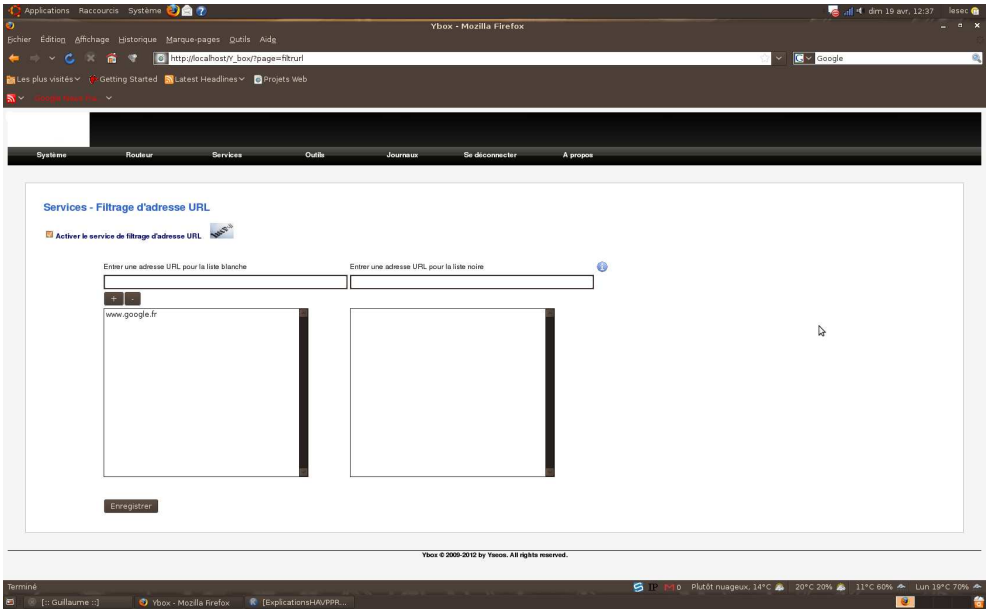

FIG. 4.4 – Page de configuration du filtre URL

4.3 SSH

4.3.1 Présentation

Le protocole SSH permet de se connecter au travers d'un réseau à une machine distante, et de lancer des applications sur cette machine par le biais de l'interpréteur de commande. Il est l'héritier de "telnet", sauf qu'à la différence de ce dernier les instructions circulent encryptées sur le réseau. Ceci lui confère une certaine notoriété dans le monde UNIX et même Windows©.

4.3.2 Objectif

L'objectif premier de cette interface était la configuration du boîtier UTM à distance. En effet, nous ne disposions pas décrans pour cette machine et le travail à plusieurs en parallèle aurait été compromis si nous n'avions pas opté pour cette solution. Ce fut donc un outil très utile pour la réalisation de ce projet. Il est par contre trop compliqué pour être utilisé par un utilisateur non-confirmé d'où la création de l'interface web.

4.3.3 Installation

Pour utiliser ce protocole, nous avons dû installer un serveur SSH sur la machine. Le client SSH, quant à lui, est la plupart du temps installé avec nos système linux. Nous avons configuré le serveur pour qu'il écoute le port TCP 2234 [3] tout simplement car ce dispositif écoute aussi sur l'interface reliée à l'Internet et il fallait induire en erreur tout potentiel pirate informatique.

[3] Le serveur écoute par défaut le port numéro 22.

Chapitre 5

Discussion

Nous avons tout au long de ce projet tenté de trouver le meilleur compromis entre efficacité, performance et sécurité. C'est là que réside le gros de notre travail et qu'intervient cette fameuse "politique de sécurité". Nous n'avons pas pu dans tous les cas trouver le meilleur compromis.

Par exemple, le problème principal de ClamAV est que ce dernier ne peut pas scanner indéfiniment des niveaux de compression imbriqués (une archive qui contient elle-même une archive qui contient elle-même une archive, etc. ...). Nous avons dû le limiter dans la configuration à huit niveaux d'archivage, sinon le temps de "scannage" devient trop important. Ainsi, si un virus se trouve au neuvième ou dixième niveau, il ne sera pas détecté par ClamAV. Ceci n'est pas réellement un problème de ClamAV mais un problème récurrent à tous les antivirus et le choix cornélien entre la sécurité et la performance. Le même inconvénient survient avec le serveur mandataire HAVP. HAVP ne scanne que des fichiers dont la taille est inférieure à une limite fixée. Si un fichier dépasse cette limite et que ce dernier est vérolé, alors le virus pourra s'introduire sur l'ordinateur hôte. On peut évidemment mettre une taille infinie mais il ne faut pas oublier les limites physiques du boîtier UTM et le problème que pourrait créer ce choix. Voici donc une autre faille. On peut bien évidemment essayer d'atténuer ces failles, voire de les supprimer, mais cela se fait au détriment des performances et du confort pour l'utilisateur.

L'installation compliquée de certains modules, peut amener à complexifier leurs mises à jour et ainsi ajouter des failles de sécurité[1]. Ce qui est le cas pour l'IPDS Snort Inline. En effet, il requiert l'installation de plusieurs logiciels qui nécessitent une bonne configuration. Une fois installé et configuré, outre de grandes capacités de détection, il dispose toutefois de quelques limites.

1. Afin d'obtenir un niveau de détection d'intrusion élevé, il est nécessaire de combiner l'usage de plusieurs sondes différentes.
2. Un N-IDPS ne peut pas analyser les protocoles sans fils et vice versa.
3. Ils ne permettent pas non plus de détecter les attaques dissumulées dans les flux chiffrés tel que le VPN, le protocole HTTPS et les sessions SSH.

[1] Pour éviter le plus possible d'être victime de problème sécuritaire, il est nécéssaire de maintenir à jour ses logiciels

De plus, ils sont susceptibles d'être eux-mêmes la cible de certaines attaques telles que le déni de service.

Dans d'autres cas, c'est le côté transparent du boîtier UTM qui peut être compromis. L'authentification par adresse MAC (Radius-MAC) ne dépend pas de la machine ou du système d'exploitation du poste du travail. Il suffit de le brancher et l'équipement réseau le détecte et enclenche automatiquement le protocole Radius. Par contre, dans le cas du sans-fil, ce "branchement" correspond à la phase d'association du poste sur la borne. La difficulté réside dans la configuration des postes clients que l'on veut authentifier avec Radius-802.1X. Il faudra pour cela que chaque poste dispose d'un logiciel appelé "supplicant"[2]. De plus, pour un poste sans fil, il faudra vérifier que la carte Wi-Fi est bien compatible avec WPA[3].

En dehors de ces points très spécifiques et qui résultent d'une étude complète, nous sommes convaincus de l'efficacité de notre boîtier UTM.

[2] Aujourd'hui, les versions les plus récentes de Windows©et de Mac OS©disposent de ce logiciel en standard. Sous Linux, la situation dépend de la distribution qui inclue ou non un supplicant. Dans tous les cas il est possible d'installer Xsupplicant ou wpa_supplicant.

[3] Il faudra faire attention et vérifier qu'il sagit de "WPA-Enterprise" et pas seulement de "WPA-PSK", parfois appelé "WPA-Home", qui peut être considéré comme une version simplifiée n'utilisant pas de serveur Radius

Troisième partie

Conclusion

Au terme de ce projet, nous sommes satisfait de regarder en arrière et de voir le travail accompli. En effet, nous sommes allés de l'étude de chaque outil jusqu'à la mise en oeuvre de ceux-ci sous diverses contraintes. Cela nous a permis d'acquérir une connaissance solide du système linux, ainsi qu'une trés bonne expèrience de l'administration et de la sécurité des réseaux en général. Malgré certains moments difficiles, nous ressortons enchantés d'avoir travaillé sur un projet aussi enrichissant dans le domaine où nous avons choisi de nous spécialiser.

L'objectif de réaliser un boîtier UTM unifiant un ensemble de modules pour sécuriser un réseau a été accompli. Même si aucun système de sécurité ne peut garantir une protection totale, un boîtier UTM constitue un filtre contre de nombreuses menaces et a pour grand avantage de ne nécessiter aucune ou peu de modifications de la part de l'utilisateur pour être effectif. Modifications pouvant, qui plus est, être appliquées depuis l'interface sans connaissances particulières en informatique. Tous ces atouts font des boîtiers UTM un système de protection encore méconnu du grand public mais qui risque fort de connaître une importante croissance dans les années à venir au vu de son utilisation grandissante.

Si le temps alloué pour le TER avait été plus important, nous aurions aimé continuer notre travail en implémentant une utilisation de VPN[4] entre des réseaux locaux physiquement séparés. Le principe étant que deux réseaux locaux distants, tous deux possédants notre boîtier UTM, puissent communiquer entre eux comme s'ils étaient sur le même réseau local de façon transparente. Ce genre d'utilisation se révélant être très pratique par exemple pour une entreprise dont plusieurs bureaux travaillants ensembles.

[4]Virtual Private Network

Quatrième partie

Glossaire et bibliographie

Client : Dans un réseau, un client est l'élément qui envoie des requêtes au serveur (exemple : un navigateur web).

DOS : Attaque visant à rendre inutilisable un matériel comme un serveur (Deny Of Service).

Filezilla : Client FTP avec interface graphique qui permet de télécharger des fichiers.

FTP : Protocole de transfert de fichier via un réseau (File Transfer Protocol).

HTTP : Protocole de transfert de page web via un réseau (HyperText Transfer Protocol).

HTTPS : Version cryptée de HTTP.

IMAP : Protocole de récupération de courriels sur un serveur mail

IP : Protocole définissant le service d'acheminement de paquets sans connexion et au mieux dans l'Internet (Internet Protocol)

Logs : Fichier texte qui enregistre toutes les actions effectuées par un programme.

Man in the middle : Attaque durant laquelle l'attaquant se fait passer pour le serveur au niveau du client et au client pour le serveur et, ainsi lui permettant de lire, insérer ou modifier des données même si la communication est cryptée.

POP3 : Protocole de récupération de courriels sur un serveur mail

Phishing : Attaque visant à faire croire à la victime qu'elle s'adresse à un tiers de confiance (site Internet bancaire par exemple) afin de lui soutirer des renseignement confidentiels (hameçonnage).

Serveur : Programme applicatif fournissant des services à des clients au travers d'un réseau.

SMTP : Protocole standard pour le transfert de mail d'un terminal à un autre (Simple Mail Transfer Protocol).

Spoofing ou IP Spoofing : Attaque visant à usurper l'identité d'un hôte sur un réseau (IP Spoofing : usurpation de l'adresse IP).

SSH : Alternative à TELNET permettant d'assurer la confidentialité d'une session à l'aide de cryptage (Secure SHell).

TCP : Protocole standard de la couche transport fiabilisant IP (Transmission Control Protocole).

TELNET : Protocole TCP/IP standard pour l'accès à distance (en ligne de commande).

URL : Adresse permettant de localiser une ressource (page web) dans l'Internet.

UTM : Type de boîtier (objet de ce projet) permettant une gestion unifiée des menaces de l'Internet (Unified Threat Management).

VPN : Réseau privé transitant par un réseau public comme Internet (Virtual Private Network).

Wget : Client FTP en ligne de commande qui permet de télécharger des fichiers.

WPA et WPA2 : Wi-Fi Protected Access est un mécanisme pour sécuriser les réseaux sans-fil de type Wi-Fi.

Bibliographie

[AJ07] Olivier Allard-Jacuin. Initiation à netfilter, le firewall de linux, par l'exemple. *hakin9*, 2007.

[Bor06] Serge Bordères. *Authentification réseau avec Radius-802.1X, EAP, FreeRadius*. Eyrolles, 2006.

[Cal] Christian Caleca. http ://christian.caleca.free.fr/snmp/mrtg_2.htm.

[cB08] Jean-François Bouchaudy. *Linux Administration Tome 3*. EYROLLES, 2008.

[Cla] ClamAV. http ://www.clamav.net/.

[Com06] Douglas Comer. *TCP/IP Architecture, protocoles et application*. PEARSON, 2006.

[Fre] FreeRadius. http ://www.freeradius.org.

[FRO] FROX. http ://frox.sourceforge.net/.

[HAV] HAVP. http ://www.server-side.de/.

[Inl] Snort Inline. http ://snort-inline.sourceforge.net/.

[Koj07] Thomas Kojm. Clamav : le modèle de la moule appliqué à la virologie informatique. *hakin9*, 2007.

[MRT] MRTG. http ://oss.oetiker.ch/mrtg/doc/index.en.html.

[SNO] SNORT. http ://www.snort.org/.

[SQU] SQUID. http ://www.squid-cache.org/.

Cinquième partie

Annexes

Cahier des charges: Boîtier Unified Threat Management

Frédéric BORDI, Mohammed DJOUDI,
Cédric LESEC, Guillaume ROUVIÈRE

10 mai 2009

Table des matières

Table des figures

0.1 Présentation : Unified Threat Management

0.1.1 Qu'est ce qu'un UTM ?

Un UTM (comprendre Unified Threat Management ou Gestion Unifiée des Menaces) est destiné comme son nom l'indique à sécuriser un accès réseau sur tous les axes. C'est à dire gérer les divers problèmes de sécurité comme les intrusions, les virus, les spams, etc. ... Il réalise la plupart du temps un pont entre un réseau à sécuriser et un réseau non-sécurisé en utilisant un pare-feu, un antivirus, et bien d'autres modules décrits plus loin.

0.1.2 Utilisation de l'UTM

Notre UTM sera destiné à protéger un réseau privé des menaces d'Internet. Il devra s'utiliser de façon transparente, c'est à dire sans nécessiter de configuration poussée de la part des hotes du réseau privé. Nous allons privilégier la conception du réseau privé en WIFI. Ainsi ce boitier se placera entre le modem (box telle que livebox, freebox, neufbox, etc. ...) et les ordinateurs hotes et ces derniers se connecterons au réseau en WIFI.

0.1.3 Pourquoi ce choix ?

Nous avons choisi ce sujet, car nous souhaitons tous les quatres faire du réseau, de son administration et de sa sécurité un objectif de profession. Travailler sur les boîtier UTM, qui sont la nouvelle tendance en matière de sécurité réseau, nous permettra d'apréhender plus facilement les problèmes de notre futur métier. De nombreux UTM existent, mais les versions libres et gratuites sont moins présentes. Le but de ce TER sera pour nous, de nous approprier les divers outils des UTM en concevant le notre, et d'essayer d'innover en apportant d'autres fonctionnalités lorsque celle de base seront opérationnelles.

0.2 Cahier Des Charges

0.2.1 Fonctionnalités à inclure

La première fonctionnalité indispensable à inclure est celle du pare-feu. En effet, pour protéger efficacement le sous réseau un pare-feu efficace doit être configuré. Pour cela nous allons utiliser IPTables, l'interface en ligne de commande du pare-feu de linux : NETFILTER. Grâce à cela, nous allons pouvoir définir une zone "démilitarisée", plus communément connue sous le nom de DMZ. Cette zone est telle que seul les paquets potentiellement autorisés seront succeptible de pénétrer. Ensuite, dans cette DMZ nous allons implanter des fonctionnalité de serveur mandataire, d'anti-virus, d'anti-spam, ainsi qu'un IPS (Intrusion Prevention System), ce dernier étant destiné à analyser les flux de données transitant par le routeur et à couper la connexion si une menace se fait sentir. Un annuaire LDAP sera aussi déployé, afin de gérer les identifiants de connexions. Il y aura de plus un serveur web apache pour modifier et paramétrer le tout, ainsi qu'un serveur ssh pour accéder à l'ordinateur. Voici un schéma de ce projet :

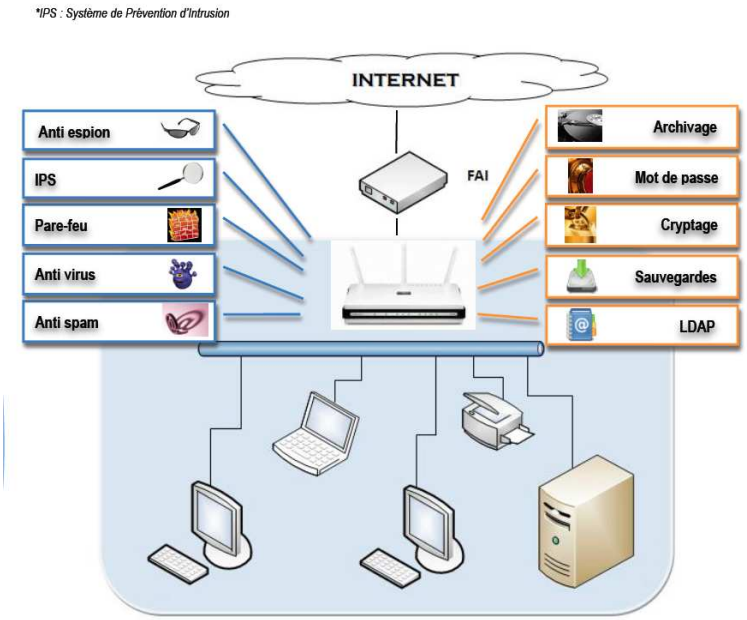

Fig. 1 – Représentation de l'UTM

Donc en récapitulant, on obtient :

- Configuration/Installation du matériel, et des serveurs DHCP, DNS, Apache, et SSH pour faciliter la connexion et le travail sur la machine même et à distance
- Configuration/Installation du Pare-Feu et définition des politiques de sécurité
- Configuration/Installation du ou des serveur mandataires
- Configuration/Installation des Anti-virus
- Configuration/Installation de l'IPS
- Configuration/Installation de l'Anti-Spam
- Configuration/Installation d'un annuaire LDAP
- Création de l'interface web
- Création de notre distribution Linux
- Configuration/Installation des autres fonctionnalités facultatives

Le choix de notre distribution linux de base est Debian, car les mises à jour peuvent être automatisées et c'est un système que nous connaissons bien.

0.3 Objectifs

0.3.1 Répartition

Ensembles Des Taches

Voici le diagramme de Gantt qui donne un aperçu des étapes du projet.

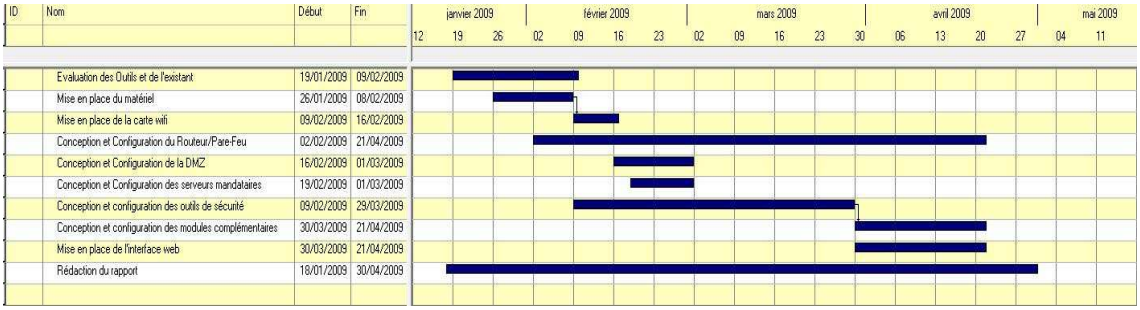

Fig. 2 – Diagramme de Gantt

La répartition des tâches n'est pas fixée entre les membres de l'équipe. En effet, chacun se voit attribué une nouvelle tâche à effectuer, lorsqu'il a fini la précédente. Le but étant d'insérer le plus de modules possible. Un fort travail en équipe est de toute manière nécessaire car en sécurité informatique plusieurs regards valent mieux qu'un.

Réunions

Nous avons décidé pour le moment de nous réunir les lundi, mercredi et vendredi de chaques semaines. Nous utilisons ces réunions pour travailler ensemble aussi et mettre en commun le travail réalisé. Un compte-rendu sera effectué par semaine.

0.3.2 Objectifs Primaires

L'objectif primaire est de concevoir notre distribution de linux intégrant les fonctionnalités de base retrouvées sur n'importe quel UTM du marché, avec une interface web pour la paramétrer. Bien entendu, la principale difficulté est d'apprendre comment tout coordonner, tout correctement paramétrer et répondre aux besoins de sécurité. Ces fonctionnalités sont détaillées à la section cahier des charges.

0.3.3 Objectifs Secondaires

Notre objectif secondaire sera d'ajouter des fonctionnalités supplémentaires facultatives telles que des serveur de fichiers, et l'encryptage de fichiers. En effet, il serait utile que cet UTM dérive son utilité en ne s'occupant plus seulement des aspects sécurité à proprement parlé. Ces objectifs seront abordés si nous en avons le temps.

0.3.4 Premières Difficultés

Les premières difficultés que nous rencontrons déjà sont dues au matériel, notamment les cartes wifi. Ces dernières doivent, en plus d'être compatibles linux, être paramètrable en point d'accès. Cet aspect devrait être résolu rapidemment.

Compte-rendu des réunions

Frédéric BORDI, Mohammed DJOUDI,
Cédric LESEC, Guillaume ROUVIÈRE

10 mai 2009

0.1 Réunions de la semaine 19/01/2009 au 25/01/2009

0.1.1 Objectifs

Nous avions lors de cette séance pris contact avec le tuteur de projet. Nous avions établi les axes du projet, commencé des recherches sur les outils à utiliser, et planifié les prochaines réunions.

0.1.2 Travaux effectués

- Petite étude sur les UTM du marché et les diverses distribution linux.
- Définition de l'architecture de base de notre UTM
- Commencement des recherches sur le pare-feu, l'antivirus, l'IPS.

0.2 Réunions de la semaine 26/01/2009 au 01/02/2009

0.2.1 Objectifs

Lors de ces séances, nous voulions mettre en place le matériel pour pouvoir commencer à travailler.

0.2.2 Travaux effectués

- Installation d'une version allégée de Debian sur plusieurs de nos ordinateurs portables.
- Détection de problèmes avec les cartes WIFI : le mode point d'accès des cartes wifi n'est pas fonctionnel sur toutes les cartes. À cela, vient s'ajouter des problèmes de compatibilité de drivers avec le système Linux.
- Paramétrage d'un des ordinateurs portables en pont Internet, un peu comme une "box" Internet[1].

0.3 Réunions de la semaine 02/02/2009 au 08/02/2009

0.3.1 Objectifs

Notre objectif était de résoudre les problèmes de matériels coût que coût.

0.3.2 Travaux effectués

- Récupération d'un vieil ordinateur : installation de trois carte réseau ethernet classiques et de la version "light" de Debian : décision de se passer du WIFI pour le coup.
- Configuration d'un ordinateur portable comme nous le souhaitions, avec le mode Ad-HOC au lieu du mode point d'accès pour la carte WIFI.
- Choix de passer par des serveurs mandataires ou proxy pour les diverses analyse antivirale.
- Analyse du firewall nommé netfilter[2] intégré au noyau linux.

[1] Livebox, Freebox, etc. ...

[2] configurable via le logiciel iptables

0.4 Réunions de la semaine 09/02/2009 au 15/02/2009

0.4.1 Objectifs

Nous avions essayer d'implanter les première fonctionnalitées de notre UTM, c'est à dire le firewall, l'antivirus, et l'IPS.

0.4.2 Travaux effectués

- Étude des politiques de sécurité en matière de parefeu
- Firewall analysé, et commencé. Il fallait le tester.
- Le proxy antivirus mis en place avec le filtre URL, il restait à mettre en place la liaison avec le proxy, rediriger le flux dessus et enfin le tester.
- Comprehension de l'IPS , il n'était pas encore mis en place, mais les règles de sécurité étaient en train d'être définies.

0.5 Réunions de la semaine 16/02/2009 au 22/02/2009

Pas de réunions cette semaine pour cause de vacances.

0.6 Réunions de la semaine 23/02/2009 au 01/03/2009

0.6.1 Objectifs

Nous voulions faire la migration vers la dernière version de Debian récemment sortie et l'installation de SNORT.

0.6.2 Travaux effectués

- Nouvelle version de Debian sortie (lenny) : ré-installation du système.
- 1ère version du Firewall fini.
- Début de l'installation et configuration de SNORT (IPS).
- Conception du script pour copier l'adresse de serveur DNS vers la configuration du serveur DHCP.

0.7 Réunions de la semaine 02/03/2009 au 08/03/2009

0.7.1 Objectifs

Les objectifs étaient la réinstallation des programmes, l'installation de SNORT et de MRTG ce dernier permettant sur le long terme, d'améliorer la qualité de service.

0.7.2 Travaux effectués

- ré-installation des serveurs mandataires antivirus HTTP (en apt-get) et filtre URL. Flux redirigé et testé.
- continuation de l'installation de SNORT (IPS).
- test concluant pour l'antivirus HTTP (HAVP et Squid).

- Modification du firewall pour intégrer l'antivirus sans changer la politique voulue.
- Début de l'étude de MRTG.

0.8 Réunions de la semaine 09/03/2009 au 15/03/2009

0.8.1 Objectifs

Cette semaine, il était prévu d'installer le proxy FTP, la continuation de l'installation de SNORT et MRTG, et la mise en place du serveur mandataire pour les courriels.

0.8.2 Travaux effectués

- ré-installation des serveurs mandataires antivirus FTP : Problème de connection.
- continuation de l'installation de SNORT (IPS), des règles et de sa base de données.
- installation du serveur proxy pop3 antivirus et antispam P3scan. Détection des problèmes de connexion du réseau WIFI-ETUD-UM2 et du serveur pop.gmail.com (port différent du 110, il utilise en fait une version sécurisée de POP).
- Optimisation du Firewall en plusieurs fichiers afin de bien délimiter les aspects de ce dernier (filtre, NAT, etc. ...).
- Conception du script pour le blacklistage de sites Internet potentiellement dangereux.

0.9 Réunions de la semaine 16/03/2009 au 22/03/2009

0.9.1 Objectifs

Les objectifs de cette semaine étaient la validation des proxy des protocoles de courriels, la résolution du problème de FROX et de SNORT, et la fin de la mise en place de MRTG.

0.9.2 Travaux effectués

- test de P3SCAN en ce qui concerne la protection antivirale et antispam.
- Fin du paramétrage de MRTG.
- Résolution de problème de SNORT (snort_inline) : Finalement, installation de l'avant dernière version de SNORT compatible avec notre machine.
- Recherche d'un proxy IMAP.
- Analyse de FROX. Fonctionnement defectueux avec certain client FTP. Du coup, choix de ne pas l'utiliser.

0.10 Réunions de la semaine 23/03/2009 au 29/03/2009

0.10.1 Objectifs

Nous définissions cette semaine toutes les méthodes d'installation des modules précédents au travers de scipts shell. Nous mettions en place un serveur apache HTTPS pour, par la suite, mettre en place le site web de configuration de l'UTM.

0.10.2 Travaux effectués

- Conception des scripts d'installation
- Mise en place du HTTPS pour le serveur Web, ce dernier permettant de ne pas compromettre l'accès par une malveillance interne au réseau local.

0.11 Réunions de la semaine 30/03/2009 au 05/04/2009

0.11.1 Objectifs

Cette semaine il était prévu de concevoir l'interface du site web qui permettra de configurer l'UTM.

0.11.2 Travaux effectués

- Conception du site web pour configurer l'UTM simplement. Il nous a fallu réfléchir comment implanter les actions possibles pour ne pas compromettre les objectifs de sécurité aportés par le boitier.
- Mise en place des "logs"
- Tests et optimisations de l'antispam

0.12 Réunions de la semaine 06/04/2009 au 12/04/2009

0.12.1 Objectifs

Les objectifs étaient de l'ordre de l'effacement des logs periodiquement et de l'exécution de script shell ou python depuis le site Internet.

0.12.2 Travaux effectués

- Exécution de script depuis le site web et mise en place du sudo pour ne pas ajouter de faille de sécurité.
- Mise en place de l'effacement des logs périodiquement
- Désactivation de l'envoie des mails par la crontab
- Récupération des adresses Internet pour mettre à jour SNORT

0.13 Réunions de la semaine 13/04/2009 au 19/04/2009

Pas de réunions cette semaine pour cause de vacances. Les objectifs de travaux personnels durant cette période étaient la rédaction du rapport et la conception du site Web.

0.14 Réunions de la semaine 19/04/2009 au 26/04/2009

0.14.1 Objectifs

Les objectifs à l'ordre de cette semaine étaient l'amélioration de l'interface du site web et le travail autour du rapport.

0.14.2 Travaux effectués

- Rédaction du rapport : mise en commun des parties et amélioration du plan.
- Conception du site web : mise en place de l'excution de scripts pour la modification des propriétées du serveur mandataire, parefeu et filtre URL.

0.15 Réunions de la semaine 27/04/2009 au 03/05/2009

0.15.1 Objectifs

Nos objectifs cette semaine étaient d'avoir des fonctionalités disponible sur le site, et la remise du rapport étant iminente, finir le rapport cette semaine pour le corriger la semaine suivante.

0.15.2 Travaux effectués

- Rédaction du rapport : amélioration des parties.
- Conception du site web : résolution de problèmes et amélioration.

```
ddns-update-style none;

option domain-name "sbhome";
option domain-name-servers 162.38.101.51;

default-lease-time 600;
max-lease-time 7200;

authoritative;

log-facility local7;

subnet 192.168.5.0 netmask 255.255.255.0 {
  range 192.168.5.2 192.168.5.200;
  option domain-name "sbhome";
  option routers 192.168.5.1;
  option broadcast-address 192.168.5.255;
}

subnet 192.168.6.0 netmask 255.255.255.0 {
  range 192.168.6.2 192.168.6.200;
  option domain-name "sbhome";
  option routers 192.168.6.1;
  option broadcast-address 192.168.6.255;
}
```

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

# The loopback network interface
auto lo eth1 eth2
iface lo inet loopback

# The primary network interface
allow-hotplug eth0
iface eth0 inet dhcp

iface eth1 inet static
address 192.168.5.1
netmask 255.255.255.0

iface eth2 inet static
address 192.168.6.1
netmask 255.255.255.0
```

```
# !/bin/sh
ILAN=eth1
ILAN2=eth2
INET=eth0
LAN=192.168.5.0/24
LAN2=192.168.6.0/24
NET=0.0.0.0/0
HNET=192.168.5.1
HNET2=192.168.6.1
LOCALHOST=127.0.0.1
HQ=192.168.2.1/32

#Activer,
#echo 1 > /proc/sys/net/ipv4/ip_FORWARD

#ajout du module ip_conntrack_ftp pour gerer les connexions FTP
modprobe ip_conntrack_ftp

#Ignore les ping envoyes par broadcasts ou multicasts. Souvent utilise pour eviter les "Ping Scanning"
echo 1 > /proc/sys/net/ipv4/icmp_echo_ignore_broadcasts

#Emp?che l' "ip spoofing". Le noyau verifie que la concordance IP/Interface soit coherente. Par
exemple une IP privee ne sera pas accepter si elle vient par l'interface connecter a internet.
echo 1 > /proc/sys/net/ipv4/conf/all/rp_filter

#suppression de toutes les regles de toutes les chaines de la table filter
iptables -t filter -F
iptables -t filter -X

#interdiction par defaut de tous les paquets
iptables -t filter -P INPUT DROP
iptables -t filter -P OUTPUT ACCEPT
iptables -t filter -P FORWARD DROP

#autorisation de la boucle locale ? lo ?
iptables -t filter -A OUTPUT -o lo -s $LOCALHOST -d $LOCALHOST -j ACCEPT
iptables -t filter -A INPUT -i lo -s $LOCALHOST -d $LOCALHOST -j ACCEPT

#on autorise tout les connexions depuis le r?seau local (connect? ? eth1) ? la box
iptables -t filter -A INPUT -i $ILAN -s $LAN -d $LOCALHOST -j ACCEPT
iptables -t filter -A INPUT -i $ILAN2 -s $LAN2 -d $LOCALHOST -j ACCEPT
iptables -t filter -A OUTPUT -o $ILAN -s $LOCALHOST -d $LAN -j ACCEPT
iptables -t filter -A OUTPUT -o $ILAN2 -s $LOCALHOST -d $LAN2 -j ACCEPT
```

```
# !/bin/sh
ILAN=eth1
ILAN2=eth2
INET=eth0
LAN=192.168.5.0/24
LAN2=192.168.6.0/24
NET=0.0.0.0/0
HNET=192.168.5.1
HNET2=192.168.6.1
LOCALHOST=127.0.0.1
HQ=192.168.2.1/32

echo "Declaration des regles du firewall"

#autorisation de toute connexion entrante d?ja ?tablie par un client sur le LAN
iptables -A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT

#autorisation de toute connexion deja ?tablie en forward du LAN vers NET ou inversement
iptables -A FORWARD -m state --state RELATED,ESTABLISHED -j ACCEPT

#autorisations n?cessaires au protocole ftp : le client sur le LAN initialise la connexion sur le port
21 du serveur,
#celui-ci communique depuis 2 ports publics 20 et 21
#il faut aussi ajouter le module ip_conntrack_ftp au kernel pour g?rer les ports priv?s cr??s par la
suite:
#voir modprobe ip_conntrack_ftp au d?but du fichier

iptables -A FORWARD -p tcp --dport 21 -i $ILAN -o $INET -m state --state NEW,ESTABLISHED,RELATED -j
ACCEPT
iptables -A FORWARD -p tcp --dport 21 -i $ILAN2 -o $INET -m state --state NEW,ESTABLISHED,RELATED -j
ACCEPT

iptables -t filter -A INPUT -i $ILAN -p tcp --dport 8110 -j ACCEPT
iptables -t filter -A INPUT -i $ILAN2 -p tcp --dport 8110 -j ACCEPT

iptables -t filter -A INPUT -i $ILAN -p tcp --dport 8034 -j ACCEPT
iptables -t filter -A INPUT -i $ILAN2 -p tcp --dport 8034 -j ACCEPT

iptables -t filter -A INPUT -i $ILAN -p tcp --dport 8080 -j ACCEPT
iptables -t filter -A INPUT -i $ILAN2 -p tcp --dport 8080 -j ACCEPT

#autorisation les requette http (80), https (443), DNS (53) et NTP(network time protocol, 123) depuis
le r?seau local vers le net

iptables -t filter -A FORWARD -i $ILAN -o $INET -p tcp --dport 443 -j ACCEPT
iptables -t filter -A FORWARD -i $ILAN2 -o $INET -p tcp --dport 443 -j ACCEPT

iptables -t filter -A FORWARD -i $ILAN -o $INET -p udp --dport 53 -j ACCEPT
iptables -t filter -A FORWARD -i $ILAN2 -o $INET -p udp --dport 53 -j ACCEPT

iptables -t filter -A FORWARD -i $ILAN -o $INET -p udp --dport 123 -j ACCEPT
iptables -t filter -A FORWARD -i $ILAN2 -o $INET -p udp --dport 123 -j ACCEPT

iptables -t filter -A FORWARD -i $ILAN -o $INET -p tcp --dport 995 -j ACCEPT
iptables -t filter -A FORWARD -i $ILAN2 -o $INET -p tcp --dport 995 -j ACCEPT

iptables -t filter -A FORWARD -i $ILAN -o $INET -p tcp --dport 465 -j ACCEPT
iptables -t filter -A FORWARD -i $ILAN2 -o $INET -p tcp --dport 465 -j ACCEPT


# on n'autorise le protocole SSH que depuis l'adresse HQ (2234)
iptables -t filter -A INPUT -p tcp --dport 2234 -j ACCEPT

#autorisations des pings (prot icmp) en sortie de box, en entr?e de box seulement depuis le LAN
#et transfert des pings seulement dans le sens LAN -> NET
#iptables -A OUTPUT -p icmp -j ACCEPT
iptables -A INPUT -p icmp -i $ILAN -j ACCEPT
iptables -A INPUT -p icmp -i $ILAN2 -j ACCEPT
iptables -A FORWARD -p icmp -i $ILAN -j ACCEPT
iptables -A FORWARD -p icmp -i $ILAN2 -j ACCEPT
```

```
# !/bin/sh
ILAN=eth1
ILAN2=eth2
INET=eth0
LAN=192.168.5.0/24
LAN2=192.168.6.0/24
NET=0.0.0.0/0
HNET=192.168.5.1
HNET2=192.168.6.1
LOCALHOST=127.0.0.1
HQ=192.168.2.1/32

echo "configuration des regles nat"


## ********************************* N A T ****************************
##**********************************************************************


#configuration de la table nat pour la translation des adresses
#initialisation
iptables -t nat -F
iptables -t nat -X

#translation des adresses priv?es
iptables -t nat -A POSTROUTING -s $LAN -o $INET -j MASQUERADE
iptables -t nat -A POSTROUTING -s $LAN2 -o $INET -j MASQUERADE

iptables -t nat -A PREROUTING -i $ILAN -p tcp --dport 80 -j REDIRECT --to-port 8080
iptables -t nat -A PREROUTING -i $ILAN2 -p tcp --dport 80 -j REDIRECT --to-port 8080

iptables -t nat -A PREROUTING -i $ILAN -p tcp --dport 110 -j REDIRECT --to-port 8110
iptables -t nat -A PREROUTING -i $LAN2 -p tcp --dport 110 -j REDIRECT --to-port 8110
iptables -t nat -A PREROUTING -i $ILAN -p tcp --dport 25 -j REDIRECT --to-port 8110
iptables -t nat -A PREROUTING -i $ILAN2 -p tcp --dport 25 -j REDIRECT --to-port 8110
```