Licence Professionnelle Sécurité et Qualité des Télécommunications
Université de Tours – Antenne de Blois

Sécurité dans les solutions Open Source

Julien VEHENT

-

Année 2004/2005

1

Introduction

La société Microgate.Pro est spécialisée dans l'intégration et la maintenance de serveurs Open Source. Cette SSII intervient principalement dans les PME, en particulier du secteur médical.

L'architecture serveur est assurée par la distribution GNU/Linux E-Smith SME. Cette dernière est optimisée et sécurisée pour permettre à l'administrateur d'installer et de configurer rapidement un serveur d'entreprise. Ses principaux intêrets sont la rapidité de déploiement mais également l'intégration de fonctionnalités avancée d'administration et de sécurité. Toutefois, cette dernière n'est pas exempte de défauts. On notera, en particulier, une segmentation excessive des fichiers de configuration qui ralentit énormément les opérations en console.

L'objectif de mon stage au sein de Microgate.Pro était donc de reprendre certaines fonctions des serveurs et de les réintégrer dans une architecture plus sécurisée et plus maîtrisable, tout en gardant la facilité et la rapidité de déploiement d'une distribution E-smith SME.
Enfin, il restait un aspect mal développé à travailler: le VPN. L'architecture SME utilisée, étant basée sur PPTP et IPSec, ne permet pas d'utiliser toute les fonctionnalités souhaitées.

Au cours de ce mémoire, je vais vous présenter quelques une des solutions que j'ai intégrées au pannel de Microgate.Pro. Nous verrons tout d'abord celles-ci sans les aspect de sécurité. Puis nous verrons quels choix ont été fait dans ce domaine et comment je les ais implémentés.

2

Solution de serveur de courriers

Les solutions de serveurs de courriers sont nombreuses et pour la plupart assez complètes, intégrant directement les outils antiviraux voir parfois antispams. Microgate.Pro déploit régulièrement ce type de serveurs via la distribution SME. Cette dernière implémente le logiciel « *Qmail* » avec les protocoles SMTP et POP3.

Ayant déjà effectué mon projet tutoré sur ce thème, je disposais déjà d'une petite expérience en abordant ce sujet. A la différence de la distribution SME, mon choix se porta plutôt sur le logiciel « *Postfix* » (*que j'ai détaillé dans le rapport de ce même projet tutoré*) dont les avantages en matières de scalabilité me le font préférer à « *Qmail* ».

Les objectifs

Hormis le transport du courrier sur le réseau internet, fonction évidente et sans réelle difficultés, cette solution doit remplir des critères d'accessibilité et de sécurité:

- Contrôle du relais

  L'ouverture d'un serveur au *relaying* entraîne non seulement une surcharge de la liaison réseau mais également un risque de résiliation de la ligne par le fournisseur d'accès à Internet.

- Contrôle des comptes utilisateurs

  La création/modification/suppression de comptes utilisateurs doit se faire le plus simplement possible. Il est en effet plus que probable qu'un non-informaticien gère

cette fonctionnalité une fois le serveur installé en entreprise.

- Inspection dynamique

  La possibilité d'inspecter systématiquement un courrier sur de nombreux critères afin de déterminer si oui ou non il est acceptable constitue la première des considérations sécuritaires de cette solution.

- Hébergement de plusieurs domaines

  Il est fréquent de voir une entité modifier le nom DNS qu'elle utilise. Il est alors important que la solution soit capable de récupérer les messages à destination non seulement du nouveau domaine mais également de l'ancien.

- Gestion des listes de diffusions

  Dés qu'une entreprise souhaite développer l'utilisation de sa messagerie, elle rencontre le besoin d'exploiter des listes de diffusions qui permettent, pour une adresse donnée, d'avoir autant de destinataires qu'on le souhaite.

- Interface de consultation

  Le « *Webmail* » est une interface de consultation via HTTPS, impliquant donc un serveur web (*en l'occurrence « Apache »*). Cette interface est déjà activée sur la distribution SME mais incomplète. Le projet « *Horde* » - développeur du webmail « *IMP* » - étant particuliérement dynamique, il convient de l'intégrer à cette solution. (*voir la plaquette relative en annexe*)

Les moyens techniques

Pour mettre en place cette solution, je dispose d'un site de test: celui de la société Microgate elle-même. En effet, c'est initialement une distribution SME qui occupe la fonction de serveur de courriers pour les domaines « microgate.fr » et « microgate.dyndns.org ».
Au niveau hardware, la tolérance de pannes est effectuée par un système de fichiers en RAID de niveau 1 (*mirroring des partitions*) et un onduleur électrique.

Au niveau réseau, le LAN est cablé en 100baseTX (*paires torsadées de catégorie 5*) et la connection au réseau Internet est assurée par une ligne ADSL de type 7000Mbps/1024Mbps (*down/up*).

Rappels sur les protocoles de courriers électroniques

Je ne vais revenir que succinctement sur ces deux protocoles que sont SMTP et IMAP étant donné qu'ils ont été détaillés dans mon rapport de projet tutoré.

Tout d'abord, le choix d'un protocole pour l'envoi des mails n'existe pas. **S**imple **M**ail **T**ransfert **P**rotocol (*SMTP*) est un standard depuis de nombreuses années et il ne semble pas se profiler à l'horizon de la communauté Internet un protocole de remplacement. On peut, par contre, constater que l'IETF publie régulièrement des écrits traitant de *SMTP*. Le dernier en date présentant un retour d'expérience pour l'exploitation de ce protocole dans un environnement mixte Ipv4/v6 (*RFC 3974 – 01/2005*).

A l'inverse, il existe plusieurs choix possible en ce qui concerne le protocole de réception du courrier. Le choix de IMAP (*Internet Message Access Protocol*) est principalement dû à sa capacité de gestion des dossiers et de classement des courriers dans ceux-ci. Il est ainsi possible de créer un dossier et d'y abonner des utilisateurs particuliers pour qu'ils puissent le consulter. De plus, IMAP limite la charge réseau requise lors de la consultation d'une boite à lettre (*Inbox*) car seuls les en-têtes sont envoyés au client de messagerie. C'est ensuite à chaque ouverte d'un message que le corps est téléchargé. Enfin, contrairement à son concurrent direct qu'est POP3, les discussions et améliorations relatives à IMAPv4 continue en particulier sur la mailing-list de l'Université de Washington.

LDAP: la gestion des comptes utilisateurs

Après le succès des NIS (*voir glossaire*) et beaucoup plus modérément des NIS+ par SUN, le CCITT (*Consultative Committee for International Telegraph and Telephone*) et l'ISO (*International Standard Organization*) se sont attelés à définir un standard d'annuaire. Ils créèrent le protocole X.500. Bien qu'ayant de très bonnes spécifications, les détails

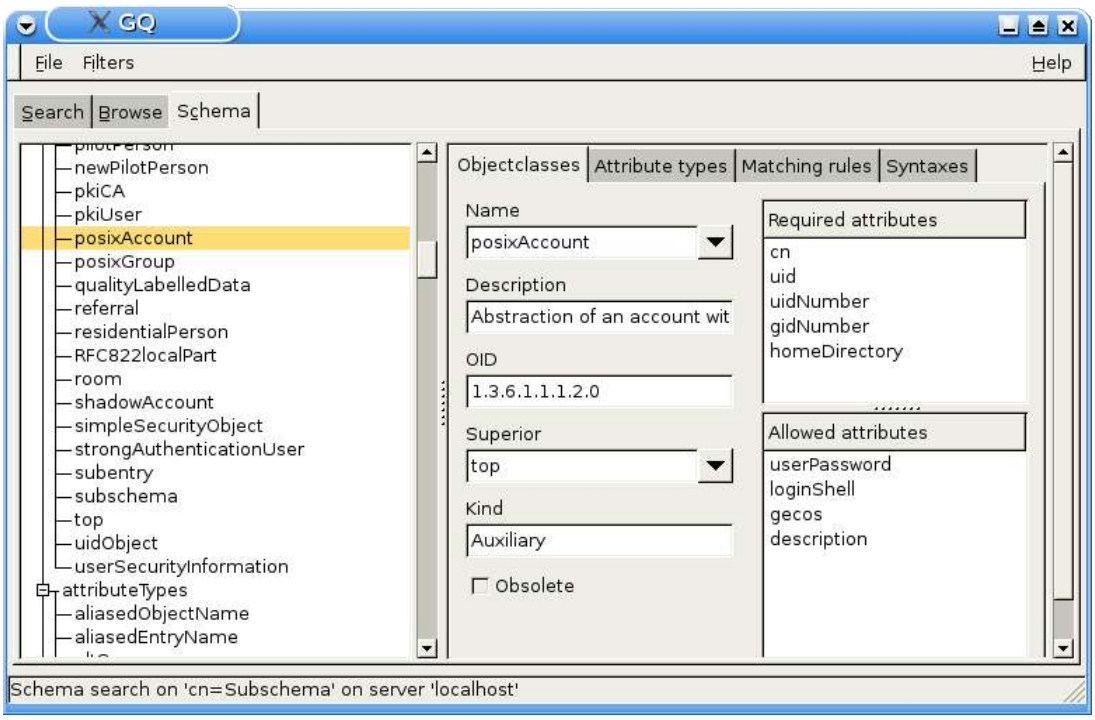
d'implémentations n'étaient en revanche pas très heureux que se soit au niveau de la rigidité des règles d'implémentation ou de la pile de protocole envisagée. C'est dans ces conditions qu'est née une version allégée de X.500 basée sur TCP/IP: **L**ightweight **D**irectory **A**cces **P**rotocol qui peut être définis comme étant un protocole spécialisé dans la manipulation d'annuaires et adapté aux réseaux et systèmes en exploitation sur l'Internet.

Dans l'architecture que nous étudions ici, l'objectif d'un annuaire LDAP est de concentrer en un point les enregistrements d'utilisateurs. L'annuaire ne sera pas répliqué ni même, pour le moment, interrogé autrement que via l'interface locale 127.0.0.1. Toutefois, de nombreuses applications savent interroger les annuaires LDAP et la présence de celui-ci au sein de la solution de serveur de courriers est un plus du point de vue technique comme commercial.

Le choix de déployer un annuaire LDAP s'est imposé lorsque j'ai travaillé sur la problématique de gestion des comptes utilisateurs. Il est, en effet, fastidieux de créer un utilisateur pour le serveur Cyrus-IMAP par la méthode en console décrite dans mon projet tutoré, et il est encore plus fastidieux de supprimer cet utilisateur ou d'en modifier les informations. Il fallait donc trouver une solution permettant à un quidam de gérer les inscriptions au serveur de courriers.

Le schéma utilisé pour gérer les utilisateur est nommé « User Account ». Il s'agit d'un schéma générique et conforme à POSIX intégré par défaut dans le logiciel de serveur LDAP OpenSource: *Slapd*.

En compléments des attributs requis, sont utilisés le « userPassword » (*chiffrés avec md5*) et l'attribut « loginShell » qui pointe sur le script '*/bin/false*' afin d'interdire toute connection shell aux utilisateurs. (*voir schéma page suivante*)



LDAP I Attributs du schéma PosixAccount

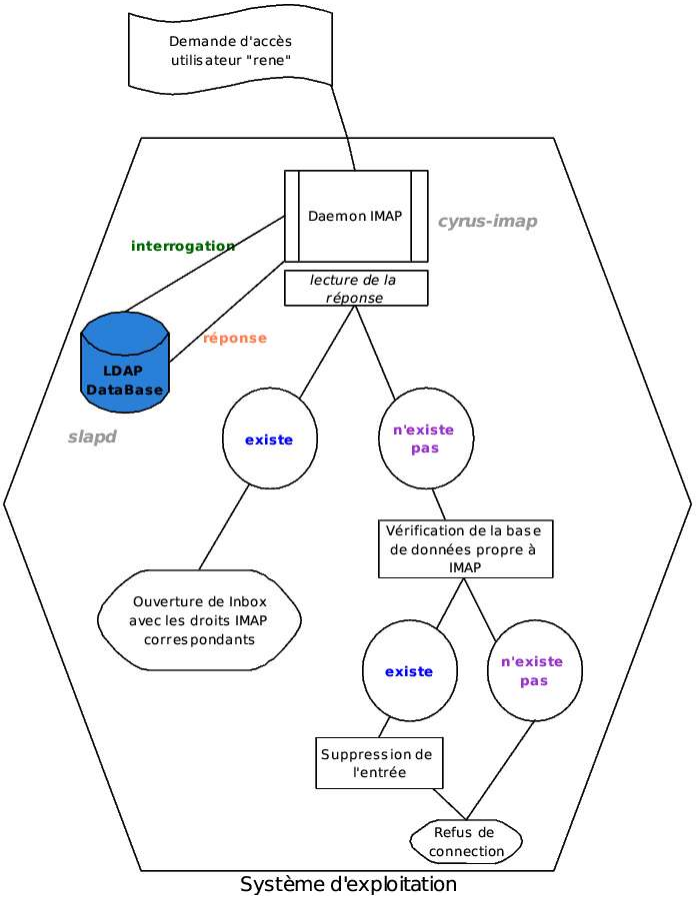
Synchronisation de LDAP avec Cyrus-IMAP

L'implémentation officielle de Cyrus-IMAP intègre la gestion des utilisateurs via LDAP mais pas la création des Inbox. Toutefois, il existe une solution créée par l'Université d'Athènes qui consiste en un patch pour Cyrus-IMAP permettant la création automatique d'une Inbox si elle n'existe pas.

Ainsi, lorsqu'un mail est reçu par le Daemon IMAP. Ce dernier interroge la base LDAP en utilisant comme champ CN (*Common Name*) la partie gauche de l'adresse email. Le serveur LDAP renvoie un status indiquant si oui ou non l'utilisateur existe. S'il existe, le Daemon IMAP vérifie l'existence de l'Inbox, la crée si elle n'existe pas et ajoute cette entrée dans sa base locale, puis classe le mail à l'intérieur.

Par la suite, lorsqu'un utilisateur souhaite accéder à ses messages, il émet une requête au prés du serveur IMAP. Celui-ci interroge encore une fois la base LDAP pour vérifier l'existence de l'utilisateur et son mot de passe (*via un challenge md5*). Si l'utilisateur existe, il ouvre l'Inbox. Si ce n'est pas le cas, le Daemon IMAP vérifie sa base locale pour

éventuellement supprimer une entrée périmée et refuse ensuite la connection. (*voir schéma ci-contre*)



LDAP II Processus de contrôle d'accès pour Cyrus-IMAP

Inspection dynamique

Comme nous l'avons vu dans les objectifs, la capacité de déterminer si un message est acceptable ou non est une fonctionnalité indispensable d'un serveur de courriers. Il appartient à l'administrateur de définir la notion d'acceptabilité en fonction de la politique de la société. Au sein de l'architecture Microgate.Pro, cette politique est définie par deux critères:

- ✔ L'inspection antivirale
- ✔ L'inspection antispam

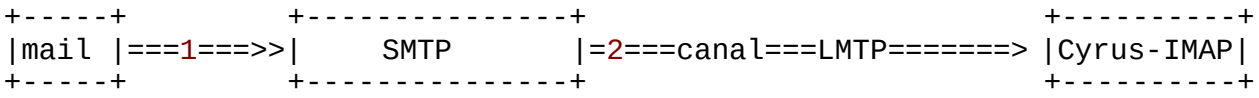
En ce qui concerne l'inspection antivirale, le critère est simple: le serveur doit refuser tout mail identifié comme étant un virus par la dernière révision connue du logiciel antiviral.

En ce qui concerne l'inspection antispam, il est beaucoup plus difficile de définir clairement une politique d'acceptabilité. La méthode choisie pour être déployée est l'analyse Bayesienne. Cette méthode, qui porte le nom de son concepteur (*M. Bayes, l'inventeur de l'algorithme utilisé ici était statisticien*), permet de comparer les éléments des nouveaux messages reçus avec les traits caractéristiques des messages rejetés et de déterminer s'ils comportent les mêmes attributs (*fréquence et position des mots dans le texte ou l'objet, nom ou adresse de l'expéditeur, etc.*). L'analyse Bayesienne est généralement beaucoup plus efficace que l'analyse par mots clés et surtout, elle produit beaucoup moins de faux positifs.

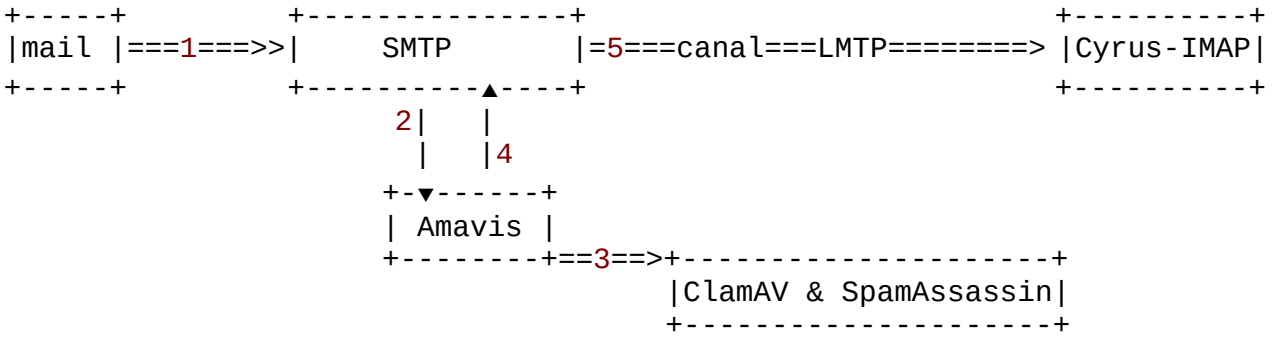
Afin d'implémenter l'inspection dynamique dans le serveur de courriers, je me suis basé sur trois logiciel:

- **Clam Antivirus**: Un puissant outil antivirus pour systèmes Unix. Basé sur le projet OpenAntivirus, ce logiciel détecte à l'heure actuelle plus de 34720 virus et ses mises à jours sont quotidiennes.
- **Spamassassin**: Ce logiciel écrit en Perl utilise l'inspection Bayesienne pour isoler les messages de spams au milieu des autres courriers. Il inclut de nombreux modules qui permette d'utiliser plusieurs types de détections.
- **Amavis**: Cet outil est lui aussi écrit en Perl. Il permet d'interfacer un MTA (*Mail Transport Agent*) comme Postfix avec des logiciels antiviraux et SpamAssassin. Amavis récupère les mails depuis le Daemon SMTP via une socket unix et les renvoie à ce même Daemon par une autre socket.

L'architecture initiale du serveur de courrier est la suivante:



Afin de mettre en place l'inspection dynamique, il convient de placer le Daemon Amavis avant la transmission sur le canal LMTP. Ainsi, on a:



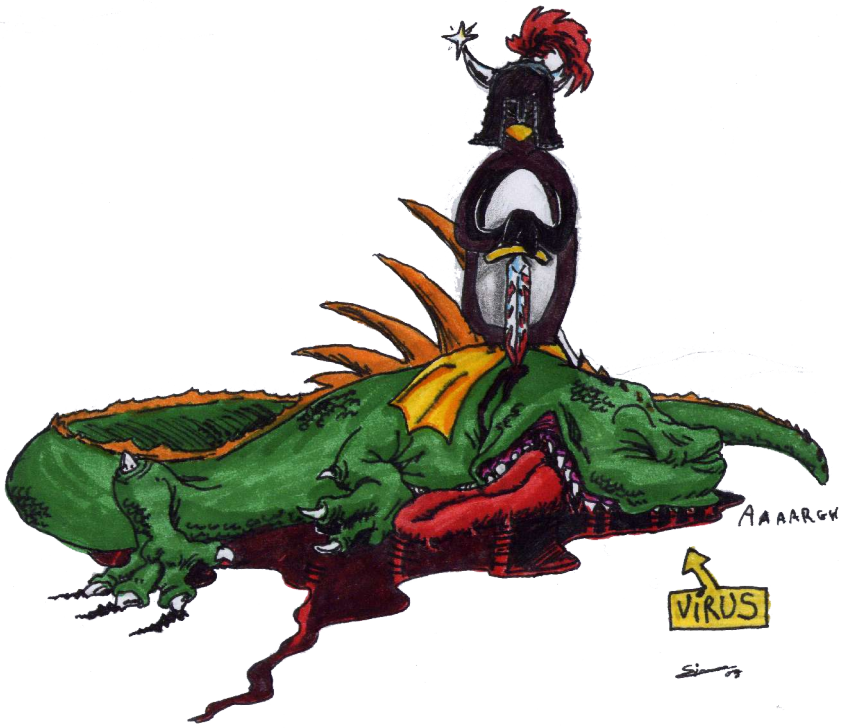
Le Daemon Master de Postfix, qui est chargé de lancer/stopper tous les processus relatifs au programme, dispose d'un processus nommé « *lmtp-amavis* ».

```
# ==========================================================================
# service    type  private unpriv  chroot  wakeup  maxproc command + args
# =>default         (yes)   (yes)   (yes)  (never)  (100)
# ==========================================================================
#smtp et amavis pour filtrage antiviral du contenu
lmtp-amavis unix     -      -       n       -       2    lmtp -o lmtp_d......
```

Extrait du fichier /etc/postfix/master.cf

Ce processus est appelé juste avant d'envoyer le message à Cyrus-IMAP. Il permet de transmettre le message au Daemon Amavis qui écoute sur le port TCP 10024. Evidemment, il faut un processus pour récupérer le message, nous avons donc un port TCP en écoute sur le 10025 pour que Postfix récupére le mail (*ou une information de non acceptabilité*) après l'inspection.

C'est ensuite au programme Amavis d'appeler les logiciels Spamassassin et ClamAV pour inspecter le mail et en déduire si, oui ou non, il a le droit de franchir la barrière. (*voir en annexe pour un exemple de message spam rejeté en quarantaine*)



3

Solution de serveur VPN

Les VPN, ou *Virtual Private Networks*, permettent aux entreprises de communiquer en temps réel et de manière sécurisée avec leurs filiales, partenaires ou collaborateurs, tout en utilisant des réseaux dits « peu sûrs » comme Internet. Cela revient à constituer de véritables réseaux privés de données, en se basant sur des infrastructures publiques de communication. Pour ce faire, les VPN s'appuient sur des mécanismes de cryptographie et de sécurité, ainsi que sur des protocoles d'encapsulations, afin de s'assurer que seules les personnes autorisées ont accès à l'information.

La technologie des VPN permet une utilisation dans différents contextes. On peut ainsi déployer :

- un VPN **Site à Site**

  utilisé pour connecter des sites distants

- un VPN **Client à Site**

  pour permettre aux postes nomades d'accéder aux ressources du système d'information

- un VPN **Extranet**

  pour permettre aux partenaires commerciaux d'accéder à des ressources précises

- un VPN **Serveur à Serveur**

  destiné à protéger les flux entre deux serveurs sensibles et distants

- un VPN **Client à Serveur**

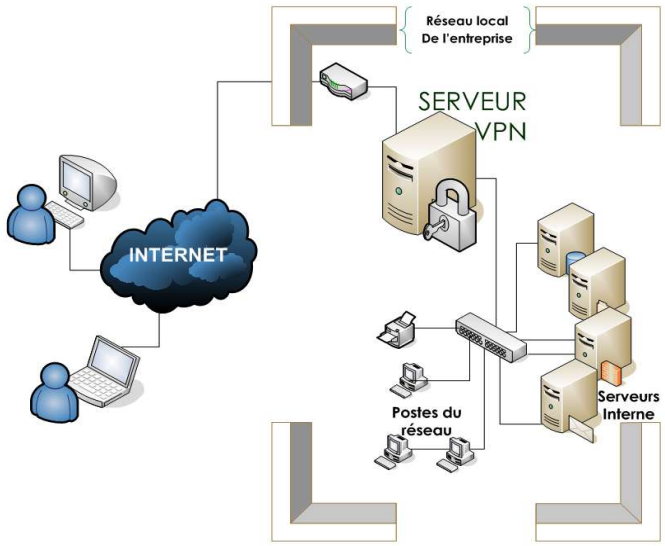
  permettant de sécuriser les échanges entre un serveur et des utilisateurs finaux

La distribution SME permet de créer un VPN **Client à Serveur**. Toutefois, en travaillant sur ce projet, je me suis orienté vers une solution **Client à Site**, plus souple car les applications utilisées par les nomades ne sont pas systématiquement disponible sur le serveur VPN. L'objectif était également de pouvoir adapter cette solution à un VPN **Site à Site** en cas de besoin.

Le dernier impératif étant de se baser sur une solution Open Source afin de rester dans les termes de la licence GPL (*General Public Licence*), c'est tout naturellement que je me suis orienté vers le logiciel OpenVPN qui permet d'établir un VPN SSL.

Les objectifs

Cette solution de serveur VPN doit permettre de monter un tunnel sécurisé de n'importe où dans l'entreprise (*en pratique: un serveur indépendants sur le réseau)* afin de donner l'accès aux clients à toutes leurs applications Intranet.



VPN I Architecture de type Client à Site

Techniquement, il faut avoir la possibilité de *broadcaster* le réseau en couche 3 de manière complètement transparente pour l'utilisateur final. Il est également important de limiter, voir exclure, les modifications réseaux nécessaires à l'implémentation d'un serveur VPN (proxies, translations d'adresses, firewall, ...).

De plus, nous devons être capable de garantir un bon niveau de contrôles d'accès. Il est fréquent que les serveurs VPN soit les cibles des pirates qui recherchent un point d'accès au réseau local de l'entreprise.

Enfin, la partie cliente du logiciel ne doit pas nécessiter de connaissance particulière afin de fonctionner. Les utilisateurs ne sont pas informaticiens.

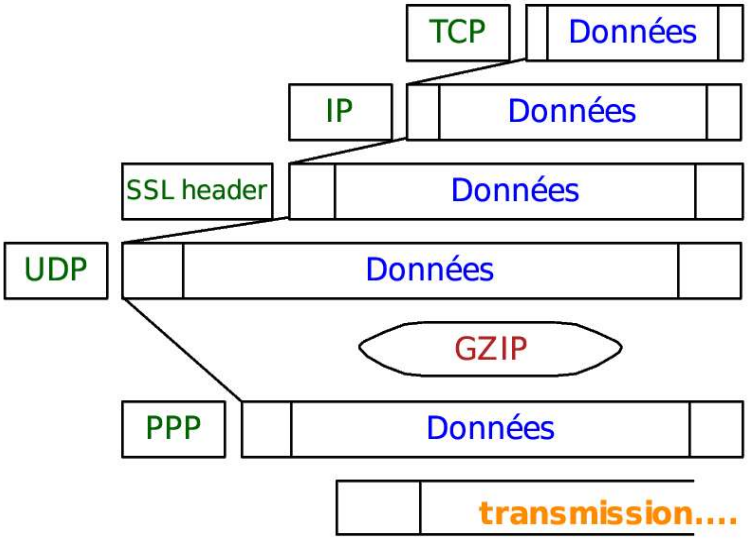
Architecture

Sur un système GNU/Linux classique, l'implémentation logicielle d'un serveur VPN via OpenVPN demande peu de composants. Les principaux outils sont la suite OpenSSL (*pour les outils cryptographiques*) et, évidemment, le logiciel OpenVPN. Nous verrons OpenSSL et l'utilisation des certificats X.509 plus en détails dans la cinquième partie de ce mémoire.

OpenVPN est un projet libre (sous licence GPL) qui fournit une solution complète pour déployer des VPN SSL. Le logiciel assure toute la procédure d'authentification et d'encapsulation. Il travaille sur les couches 2 et 3 du modèle OSI et peut utiliser TCP comme UDP. Dans notre configuration, c'est UDP qui sera utilisé car les développeurs d'OpenVPN le recommande pour une encapsulation de TCP (*majoritairement utilisé dans les applications réseaux*).
Pour le chiffrement, c'est le protocole TLS qui est utilisé. La description détaillée de ce protole est dans mon rapport de projet tutoré.
La compression des données transmises dans le tunnel est assurée par l'algorithme GZIP et le protocole de couche 2 de transmission des trames est PPP (*Point To Point Protocol*).

L'encapsulation est présentée ci-contre :

Celle-ci se fait via un périphérique de type « *TAP* ». Ce dispositif permet au système d'exploitation de ré-écrire les trames *Ethernet* pour les modifier et leurs rajouter les en-têtes nécessaires au transport dans le tunnel VPN. *TAP* est en réalité un pilote de bas niveau (*kernel*) qui encapsule les trames et les transmets ensuite au pilote de la carte réseau physiquement présente dans la machine.



VPN II Encapsulation des trames pour le VPN/SSL

Sur la machine serveur, les périphériques *TAP* et *ETH* (*interface ethernet physique sous GNU/Linux*) sont « *bridgés* » pour permettre à ces deux interfaces de communiquer entre elles directement sur la couche 2 du modèle OSI.
Cette méthode permet à un ordinateur distant du LAN d'un site de travailler comme s'il était présent sur le site. Les paquets sont encapsulés comme vu ci-dessus puis transmis sur le réseau Internet. Le serveur qui récupère ces trames enlève la capsule (*à partir du SSL Header et en dessous*) et les retransmets à l'interface réseau du serveur VPN tels qu'ils étaient à leurs départ. Ainsi, les paquets IP ne sont absolument pas modifiés.
Une application représentative de cette technique est qu'un client qui travaille en VPN sur un site distant peut « *pinger* » n'importe quelle entités de ce site, la transparence est complète.

Cette architecture de type Client à Site a la capacité d'évoluer vers une architecture Site à Site. Il suffit pour cela de mettre en vis à vis deux serveurs VPN et d'établir une connection entre eux. Les deux réseaux privés ne formeront alors plus qu'un seul.

Authentification

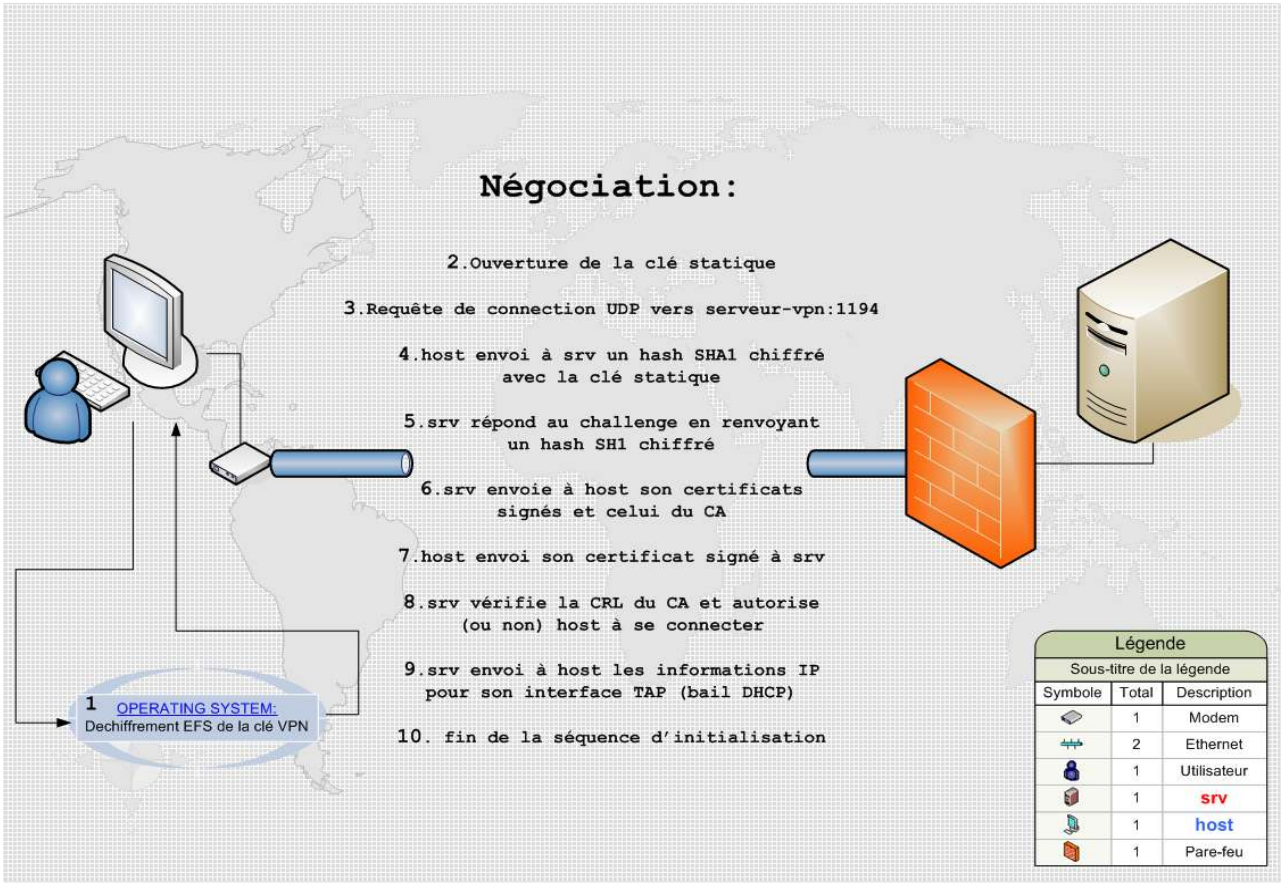
Une telle solution nécessite une protection forte. L'utilisation d'un certificat X.509 pour identifier le client est un premier point dans la procédure d'authentification. Comme nous le verrons dans la présentation de la PKI de Microgate.Pro, les clés privés des certificats clients sont protégés par un mot de passe que nous forçons en suivant le schéma « #aaaa11 ». Il est donc nécessaire de connaître ce mot de passe pour initialiser une connection au serveur VPN.
De plus, les serveurs VPN comme leurs clients vérifient à chaque initialisation de connection que les certificats utilisés par les deux entités ont bien été émis par l'autorité de certificats de Microgate.Pro.
Enfin, la révocation d'un certificat client est assurée par la PKI, les serveurs VPN synchronisant cette CRL toutes les heures (*voir chapitre 5*).

Si ce niveau de sécurité est suffisant pour des infrastructures non critiques, il ne l'est pas pour l'application que nous recherchons ici. C'est pourquoi un second niveau de sécurité à

été mis en place. Il consiste en une clé de 2048 bits émise par le serveur VPN et propre à celui-ci que possède chaque client. Cette clé permet d'initialiser la connection au serveur VPN et ceci avant même l'échange des certificats signés.

Enfin, les clients étant systématiquement des systèmes Microsoft basés sur un noyau NT5.0/NT5.1, nous utilisons le protocole EFS (*Encrypting File System*) pour protéger la clé d'un VPN. Ce troisième niveau de sécurité permet de s'assurer que seul un utilisateur désigné (*celui qui demande le chiffrement*) aura accès à la clé statique du serveur VPN.
L'avantage de EFS est que l'application cliente qui utilise des données chiffrées avec ce système n'a pas besoin d'implémenter un protocole de déchiffrement particulier. C'est le système d'exploitation qui déchiffre et envoi ensuite les données à l'application.

En conclusion de ce chapitre, voici comment se passe une ouverture de session VPN :



VPN III Negociation d'une session VPN/SSL de type Client à Site

4

Sécurité et Optimisation des serveurs

Dans ce chapitre, je vais décrire quels sont les modifications effectuées sur les systèmes d'informations pour en assurer la sécurité et l'optimisation. Nous verrons, dans une première approche, la sécurité du système d'exploitation en particulier au travers du système *PaX*. Ensuite, c'est à la problématique de la sécurité réseau des systèmes GNU/Linux que nous nous intéresserons. Ces modifications ont été apportées à tous les serveurs sur lesquels j'ai travaillé. Elles sont, de plus, intégrés dans les différents scripts d'installations.

Au niveau du système d'exploitation

La distribution utilisée durant ces douze semaines de stage (*et également à titre privé*) est une Debian Net Install. La particularité du projet Debian est que ses développeurs essaient toujours de fournir un très bon niveau de sécurité. Les outils de gestion des packages intégrent le contrôle PGP (*Pretty Good Privacy*) et de nombreux outils empêchent par défaut les attaques les plus connus (*IP spoofing, Denial Of Service, ...*).
De plus, les fonctionnalités les plus sûrs de stockage de mot de passes sont utilisées (*shadow password entre autres*) et le système est pré-configuré pour fournir une quantité importante de *logs* (*journaux systèmes*).
Enfin, l'intêret d'utiliser une version Net Install est qu'à la fin de l'installation du système, aucun service inutile n'est déployé. L'espace disque utilisé se limite à 400Mo et les services écoutant sur le réseau se comptent sur les doigts d'une main.

Le seul aspect non réellement maîtrisé est le noyau. Bien que ce dernier soit très souvent mis à jour et que les versions stables soient réellement éprouvées et exemptes de

bugs, ce dernier reste tout de même sensibles aux attaques les plus pointues, comme le « *Buffer Overflow* » (*voir glossaire*).

Le projet *GrSecurity* fournit une réponse à ce type d'attaques. Il consiste en un patch applicable au noyau (*qu'il faut ensuite recompiler*).
L'une des fonctionnalités de *GrSecurity* qui nous intéressera ici est le projet *PaX.* Ce projet tend a développer une méthode permettant de prévenir, et donc d'empêcher, les attaques de types *buffer overflow.* Pour ce faire, *Pax* utilise trois principes pour la gestion de la pile :

- ✔ Introduire/exécuter un code arbitraire
- ✔ Exécuter le code existant en dehors de la pile originale
- ✔ Exécuter le code existant dans la pile originale mais avec des données arbitraires

Une fois le noyau recompilé avec *PaX (en dur uniquement, il n'est pas possible de le compiler en tant que module)* tous les programmes sont protégés par les trois méthodes ci-dessus. Bien qu'il soit théoriquement possible de « *bypasser » PaX*, aucun écrit ne montre une réalisation pratique de l'exploitation d'une faille au travers de *PaX*. On peut donc en conclure qu'il fournit un bon niveau de sécurité (*avec une certaine réserve toutefois, la résistance d'un système informatique étant du domaine de l'éphémère*).

Au niveau réseau

La sécurité au niveau réseau est assurée sur deux niveaux:

- ✔ au niveau noyau avec le firewall Netfilter
- ✔ au niveau applicatif via les outils GNU/Linux

Tout d'abord, le firewall Netfilter/Iptables. Ce dernier est utilisé pour filtrer les paquets sur les couches 3 et 4 du modèle OSI. Ce firewall est puissant et rapide car il travaille directement dans le noyau Linux, évitant ainsi les remontées systématiques dans la couche applicative comme le fond les firewall logiciels classiques. Toutefois, bien que des outils existent pour en simplifier l'administration, il n'est pas facile de créer et administrer un nombre important de règles. C'est pourquoi nous utilisons Netfilter/Iptables comme rempart de

premier niveau avec des règles basiques mais efficaces. L'exemple ci-dessous concerne le serveur de courriers de Microgate:

```
srvlinux-mailbox:~# iptables -L -v
Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
pkts bytes target  prot opt in   out  source    destination
2823 1850K ACCEPT  all  --  lo   any  anywhere  anywhere
0    0     ACCEPT  icmp --  any  any  anywhere  anywhere     icmp echo-request state
                                                              RELATED,ESTABLISHED
0    0     ACCEPT  icmp --  any  any  anywhere  anywhere     icmp echo-request state NEW
                                                               limit: avg 10/min burst 5
0    0     ACCEPT  icmp --  eth0 any  anywhere  anywhere     icmp echo-reply
1007 129K  ACCEPT  udp  --  eth0 any  anywhere  anywhere     udp spt:domain
133  11229 ACCEPT  tcp  --  eth0 any  anywhere  anywhere     tcp dpt:ssh
899  797K  ACCEPT  tcp  --  any  any  anywhere  anywhere     tcp dpt:smtp
1551 73736 ACCEPT  tcp  --  any  any  local/24  anywhere     tcp dpt:imap2
252  88278 ACCEPT  tcp  --  eth0 any  anywhere  anywhere     tcp dpt:https
36   13428 ACCEPT  tcp  --  eth0 any  ocsp.microgate.fr  anywhere  tcp spt:https
2346 313K  DROP    all  --  eth0 any  anywhere  anywhere

Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination

Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
pkts bytes target     prot opt in     out     source               destination
2823 1850K ACCEPT     all  --  any    lo      anywhere             anywhere
5638 2802K ACCEPT     all  --  any    eth0    anywhere             anywhere
```

C'est règles autorisent les paquets pour les protocoles SMTP, SSH, IMAP et HTTPS à entrer sur la machine. Elles limitent également les entrées de ping à 10/minutes et autorisent les retours de requêtes DNS. Une règle supplémentaire permet d'autoriser une connection HTTPS vers l'autorité de certificats pour synchroniser la CRL. Hormis ces règles, tous les paquets entrants sont rejetés.

Il n'y a pas de règle de *forwarding* car notre serveur n'assure pas de fonctions de routage de paquets.

Enfin, le serveur est autorisé à sortir comme il l'entend. La limite étant fixée sur les paquets entrants. Par exemple, si notre serveur est compromis, le pirate ne pourra pas ouvrir de session SSH sur une autre machine du réseau sans abaisser le firewall, cette opération nécessitant le mot de passe Root.

Le deuxième niveau de sécurité réseau est en réalité un ensemble d'opérations décrites dans le chapitre « *Linux General Security* » du livre Securing and Optimizing Linux:

RedHat Edition de Gerhard Mourani. Ces opérations permettent d'assurer un niveau de sécurité élémentaire et générique pour tout serveur Linux déployé. Elles sont énumérées ci-dessous:

- ➔ Fermeture du compte Root au terme de 7200 secondes d'inactivité
- ➔ Suppression des services inutiles (*en particulier dans inetd*)
- ➔ Immuabilité des principaux fichiers systèmes (*passwd, shadow, host, etc...*)
- ➔ Purge régulière du fichier bash_history
- ➔ Centralisation des logs sur un serveur
- ➔ Contrôle de l'exécution des scripts init.d
- ➔ Suppression des bits SUID
- ➔ Protection contre le hacking ICMP
- ➔ Protection contre les TCP SYN Cookie
- ➔ Protection contre l'IP Spoofing
- ➔ Protection contre les paquets impossibles (*martian log*)
- ➔ Interdiction des comptes RHosts

Ces mesures de sécurité, appliquées à tous les serveurs de Microgate.Pro, assurent une protection efficace contre la plupart des attaques, référencées ou non.

5

Public Key Infrastructure

En étudiant les besoins en sécurité de Microgate.Pro et, par là même, les solutions implémentables chez ses clients, il apparut que nous disposions d'une marge de manoeuvre assez limitées.
En effet, les entreprises faisant appel à Microgate.Pro ne sont pas particuliérement sensibles aux problématiques de sécurité (*tant qu'elles n'en font pas les frais tout du moins*). Il fallait donc trouver une solution suffisamment transparente et commune à de nombreux protocoles afin à la fois de limiter les coups de développement mais également d'utiliser sa récurrence au sein des solutions comme un argument d'indispensabilité.

Le protocole TLS (*Transport Layer Security, aussi appelé SSL3.1*) répond à ces besoins, il fût donc choisi pour être la base de la politique de sécurité. L'une des premières étapes de mon travail sur le thème de la sécurité via TLS fût donc d'étudier une solution stable et solide pour intégrer une Autorité de Certificats Racine (*RootCA*)à la structure de Microgate.Pro.

Déployer une PKI en local a de nombreux avantages. Nous pouvons ainsi gérer tous les certificats qui sont utilisés dans les parcs maintenus par les administrateurs de Microgate.Pro. Cela signifie avoir la possibilité nous seulement de délivrer de nouveaux certificats mais également d'en révoquer et d'informer les différentes entités de ces révocations.

Rapidement, la solution qui apparut comme la plus mûre fût la suite de logiciels OpenSSL, basée sur un système GNU/Linux. Cette suite intègre tous les outils pour implémenter, éprouver et gérer une PKI. Ce que nous appellerons le « *Toolkit OpenSSL* »

comprend les outils nécessaires à la génération des certificats sur de nombreux champ X.509, la révocation et la diffusion des CRL (*Certificates Revocation List*), la gestion des formats *PEM* et *DER*, les outils d'interrogation *OCSP* (*Online Certificate Status Protocol*) et de nombreux outils complémentaires dont je ne parlerais pas ici.

Le projet OpenSSL est diffusé sous licence GPL et est particulièrement bien documenté. J'ai d'ailleurs eu l'occasion de discuter avec de nombreux administrateurs, que ce soit sur les listes de discussions ou sur des sites divers, qui m'ont fait profiter de leurs expériences sur certains points obscurs.

Architecture

L'architecture Hardware du *RootCA* est commune à celle présentée dans les chapitres précèdents. De mêmes pour les considérations de sécurité étudiées au chapitre 4. En réalité, c'est en cherchant à sécuriser le *RootCA* que j'ai construit cette politique.

Les règles Netfilter/Iptables sont particulièrement strictes. Ainsi, seul l'accès HTTPS est autorisé et seulement depuis le réseau local. En effet, nous avons besoin de cet accès pour répliquer la CRL générée toute les heures sur le composant Apache du serveur de courriers (*qui dispose, lui, d'un accès depuis l'Internet*). De plus, le *RootCA* est utilisé comme bibliothèque pour conserver les *HowTos* que j'ai écris pendant mon stage. Cette bibliothèque utilise elle aussi le port 443 (*https*).

Propriétés de l'Autorité

Le point de montage */var* est répliqué sur deux disques physiques en temps réel. Les données relatives au *RootCA* sont stockées dans le répertoire */var/certificats (A partir de maintenant nous utiliserons ce répertoire comme racine de travail).* La paire de clés du RootCA est basée sur l'algorithme RSA. D'une longueur de 2048 bits, elle garantissent que même avec un super-calculateur, il faudrait des années pour la factoriser.
La clé privée est conservée dans le répertoire « *ca* », comme toutes les informations relatives

à l'autorité de certificats.

Ce dossier contient les fichiers suivants:

```
authority:/var/certificats/ca# l
total 33
-r--------  1 root daemon 1679 2005-05-02 13:00 ca.key
-rwxr-x---  1 root daemon 1830 2005-05-02 13:00 ca.pem
-rw-r--r--  1 root root   1818 2005-06-03 16:12 index.txt
-rw-r--r--  1 root root     21 2005-06-03 16:12 index.txt.attr
-rw-r--r--  1 root root     21 2005-06-03 16:11 index.txt.attr.old
-rw-r--r--  1 root root   1678 2005-06-03 16:11 index.txt.old
drwxr-x---  2 root daemon  576 2005-06-03 16:12 newcerts
-rw-r--r--  1 root root      3 2005-06-03 16:12 serial
-rw-r--r--  1 root root      3 2005-06-03 16:07 serial.old
```

PKI I Fichiers relatifs à l'autorité de certificats

Outre les fichier « *ca.key* » et « *ca.pem* » qui constitue les indispensables clé privée et certificat signé, ce répertoire contient des fichiers spécifiques à l'attribution et la révocation de certificats.

Le fichier « *index.txt* » contient des informations sur les certificats signés par l'autorité:

```
authority:/var/certificats/ca# head -n 2 index.txt

R 150430110521Z  050513155313Z 01 /C=FR/ST=Indre-et-Loire/L=Tours/O=Resgate
                    Security Department/CN=srvlinux-mailbox.microgate.fr/
                                    emailAddress=security@microgate.fr

V 150430120656Z  02 /C=FR/ST=Indre-et-Loire/L=Tours/O=Resgate      Security
        Department/CN=ocsp.microgate.fr/emailAddress=security@microgate.fr
```

➔ Le premier certificat qui apparaît dans cette liste (*tronquée aux deux premières entrées*) est révoqué. La lettre « R » en début de ligne l'indique, suivent ensuite les dates d'émission et de révocation, le numéro de série et les champs X.509 obligatoires.

➔ Le deuxième certificat est valide (*la première lettre est un « V »*). les champs sont les mêmes à l'exception bien sûr de la date de révocation qui n'est pas présente.

Ensuite, toujours dans le répertoire « *ca* », nous avons le répertoire « *newcerts* » qui contient une copie des certificats signés émis par l'autorité.
Enfin, le fichier « *serial* » contient un entier positif qui est la valeur du numéro de série du prochain certificat qui sera signé par l'autorité.

Certificats

Sur ce *RootCA*, actuellement en production, la génération des certificats se fait sur quatre critères selon que le certificats demandé est destiné à un CA, un Serveur, un Client ou à un Responder OCSP (*dernier point de ce chapitre*).

Pour générer un certificat signé, il faut tout d'abord posséder une paire de clé et un fichier *CSR* (*Certificate Signing Request*). Le *ToolKit OpenSSL* permet de créer ces fichiers. Le fichier CSR est ensuite soumis à l'autorité. Le *RootCA* ajoute les champs que nous allons voir ci-dessous et certifie l'identité de son propriétaire en apposant un hash de l'ensemble du certificat chiffré avec le « *ca.key* ».
Les champs ajoutés lors de la création des certificats signés sont les suivants:

Pour un CA (*qu'il soit racine ou non*):

```
[CA_ROOT]
nsComment                       = "Autorite de Certificats Racine"
subjectKeyIdentifier            = hash
authorityKeyIdentifier          = keyid,issuer:always
basicConstraints                = critical,CA:TRUE,pathlen:1
keyUsage                        = keyCertSign, cRLSign
```

Les deux derniers champs spécifie que le propriétaire de ce certificat est une autorité qui est capable de signer des certificats et des CRL.

En ce qui concerne les certificats Serveur:

```
[SERVER_RSA_SSL]
nsComment                       = "Certificat Serveur SSL"
subjectKeyIdentifier            = hash
authorityKeyIdentifier          = keyid,issuer:always
issuerAltName                   = issuer:copy
```

```
basicConstraints                = critical,CA:FALSE
nsCertType                      = server
extendedKeyUsage                = serverAuth
authorityInfoAccess             = OCSP;URI:http://ocsp.microgate.fr
crlDistributionPoints           = URI:http://ocsp.microgate.fr/current
```

Les sept premiers champs parlent d'eux mêmes. Remarquons que ce type de certificat ne peut pas signer d'autre certificats (*argument* `basicConstraints`). Il est important de noter la présence des champ « *authorityInfoAccess* » et « *crlDistributionPoints* » relatifs au protocole OCSP. L'argument URI (*Uniform Ressource Indicator)* indique à quelle adresse les requêtes OCSP doivent être envoyée.

Maintenant, les certificats clients:

```
[CLIENT_RSA_SSL]
nsComment                       = "Certificat Client SSL"
subjectKeyIdentifier            = hash
authorityKeyIdentifier          = keyid,issuer:always
issuerAltName                   = issuer:copy
basicConstraints                = critical,CA:FALSE
keyUsage                        = digitalSignature, nonRepudiation
nsCertType                      = client
extendedKeyUsage                = clientAuth
```

Ces derniers sont limités à une simple fonction d'identification du propriétaire. A ce titre, il ne peuvent pas signer de certificats ni contrôler un statut sur le répondeur ocsp.

Enfin, les certificats pour répondeur OCSP:

```
[OCSP]
nsComment                       = "OCSP Responder"
subjectKeyIdentifier            = hash
authorityKeyIdentifier          = keyid,issuer:always
basicConstraints                = critical,CA:FALSE
extendedKeyUsage                = OCSPSigning
crlDistributionPoints           = URI:http://oscp.microgate.fr/current
```

Ce type de certificats possède l'argument `OCSPSigning` qui autorise son propriétaire à signer des réponses OCSP et à les diffuser à l'adresse spécifiée dans le dernier argument.
La présentation d'un certificat X.509 signé est détaillé dans mon rapport de projet tutoré, je

ne reviendrais donc pas dessus.

Gestion de la révocation

La nécessité de connaître le statut d'un certificat s'est faite ressentir lors du déploiement de la solution VPN. Il est, en effet, primordial de pouvoir retirer l'accès à un utilisateur qui aurait, par exemple, quitté la société propriétaire du serveur VPN.
En elle-même, la révocation d'un certificat n'est pas une tâche complexe. Il suffit de disposer d'un accès au *RootCA* et de lancer la commande :

```
#openssl ca -revoke $NAME/$NAME-signed-cert.pem -config ssl/openssl.cnf
```

où *$NAME* est le CN (*Common Name – voir le fichier « index.txt »*) du certificat signé.
Cette commande édite le fichier « *index.txt* » que nous avons vu précédemment.

Une fois cette tâche effectuée, la difficulté consiste en la diffusion de ces révocations. Pour cela, il existe deux méthodes: CRL (*Certificates Revocation List*) et OCSP (*Online Certificates Status Protocol*).

Certificates Revocation List

CRL est la méthode classique de diffusion des révocations de certificats X.509. Elle consiste en un fichier monolithique qui contient les identifiants des certificats révoqués. Ce fichier est généré au format PEM et signé par le *RootCA*. Il est donc impossible de falsifier une CRL autrement qu'en compromettant ce même *RootCA*.

Cette méthode possède deux avantages:

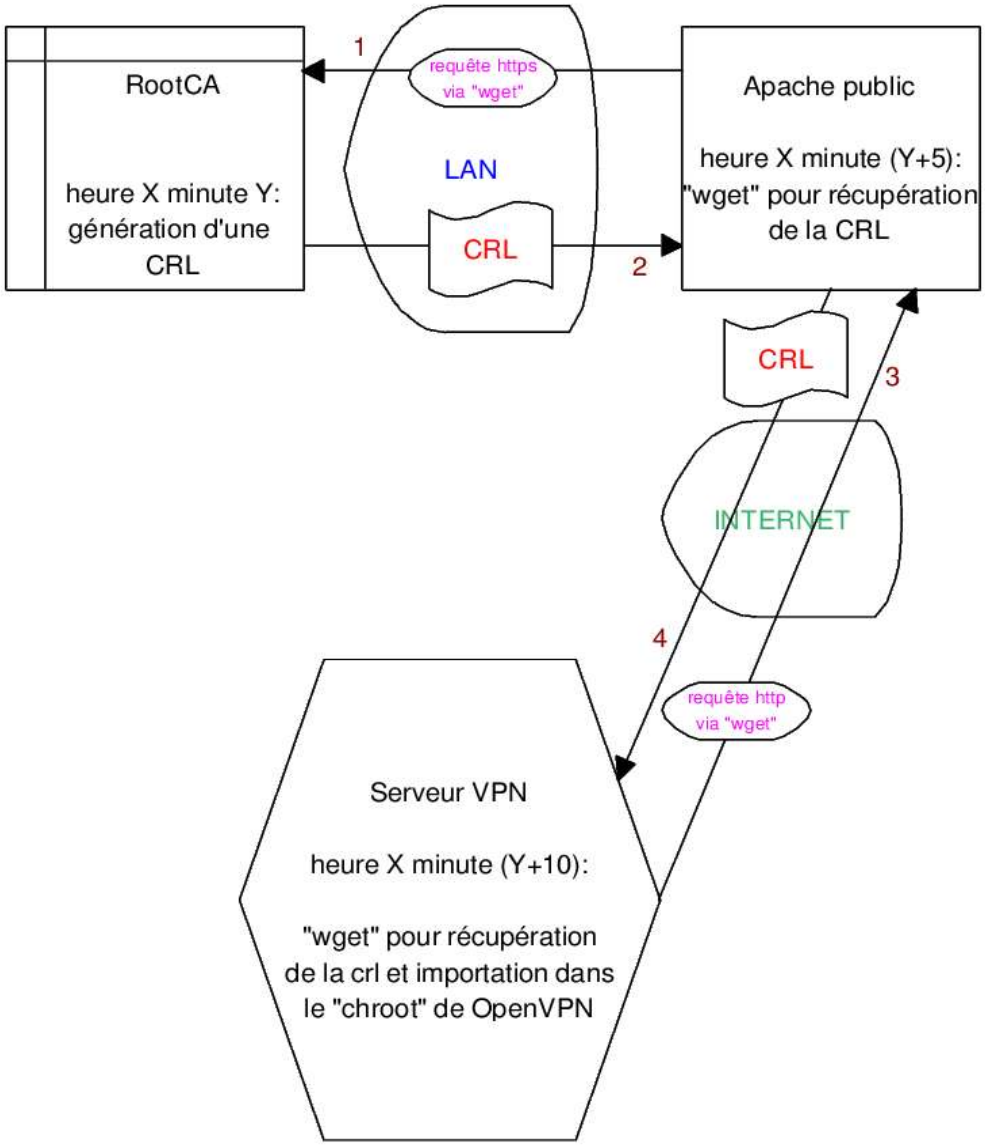
- ✔ Elle est très simple à déployer, une simple ligne de commande lancée toute les heures peut créer le fichier.
- ✔ Elle est interprétée par la quasi totalité des protocoles utilisant la technologie des PKI.

et possède également un inconvénient majeur:

- ✗ Une CRL doit être répliquée sur TOUS LES SERVEURS qui appartiennent à la PKI pour que le contrôle de la révocation soit effectué. Il n'est pas possible d'interroger une CRL au travers

d'un réseau.

Cette limitation oblige sa publication toutes les heures sur un serveur Apache pour que les serveurs de la PKI puissent se synchroniser. Si le serveur Apache est compromis (*les failles Apache sont parmis les plus prisées*), alors le contrôle de la révocation est compromis. Certaines applications vont mêmes jusqu'à couper leurs fonctionnement si elle ne disposent pas d'une CRL valide.



PKI II Mécanisme de publication d'une CRL sur l'architecture Microgate.Pro

Online Certificates Status Protocol

Il s'agit d'un protocole servant à déterminer l'état de révocation actuel d'un certificat numérique sans avoir recourt aux CRL. OCSP permet aux applications de déterminer l'état d'un certificat donné. OCSP peut être utilisé pour satisfaire à des exigences de fourniture d'information de révocation plus ponctuelles qu'il n'est possible de le faire avec les CRL.

Un client OCSP émet une requête d'état de certificat sur le port 80 (*HTTP*) d'un serveur OCSP et suspend l'acceptation du certificat en attendant une réponse du serveur.

Une requête OCSP contient les données suivantes:

- ➔ Version du protocole
- ➔ service requis
- ➔ identifiant du certificat cible
- ➔ extensions optionnelles qui peuvent être traitées par un répondeur OCSP

Lors de la réception d'une requête OCSP, un répondeur détermine si

- ✔ Le message est correctement formulé
- ✔ le répondeur est en mesure de fournir le service
- ✔ la requête contient les informations nécessaires au répondeur

Si au moins l'une des conditions précédentes n'est pas remplie, le serveur OCSP produit un message d'erreur sinon il fournit une réponse complète.

Toutes les réponses complètes sont numériquement signées. La clé utilisée pour la signature appartient à un répondeur désigné par le *RootCA* qui possède un certificat particulier de type OCSP (*voir section Certificats*).
Une réponse est composée du nom du répondeur, une valeur de statut pour chacun des certificats de la requête et une signature effectuée par la clé privée du certificat OCSP.

Il existe trois valeurs pour le status des certificats:

good revoked unknown

- ➔ La valeur *good* signifie une réponse positive quant au statut recherché. Cette réponse signifie que le certificat n'est pas révoqué.
- ➔ Le statut *revoked* indique que le certificat a été révoqué (soit de façon permanente soit temporaire).
- ➔ Le statut *unknown* indique que le répondeur ne dispose pas d'informations concernant le certificat faisant l'objet de la requête.

Au sein de la PKI Microgate.Pro, le répondeur OCSP est hebergé sur le *RootCA*. C'est le logiciel *OCSPD* fournit par le site Openca.org qui est utilisé. Les répondeurs OCSP scalables sont rares car ce protocole ne connais pas un grand succès. *OCSPD* est stable même face à un grand nombre de requêtes. La possibilité de le *chrooter,* de l'exécuter sous un compte non Root ou encore de spécifier le nombre de processus à lancer lui assure un bon niveau de sécurité.

Toutefois, ce protocole connais un nombre assez important de limitations. Il est tout d'abord très sensible au DoS (*Denial Of Service*). De plus, les logiciels qui intègrent un requêteur OCSP sont rares (*Mozilla étant le seul que j'ai trouvé*). OpenVPN ne l'intégrant pas, j'ai dû me contenter de l'utilisation des CRL. Enfin, un répondeur OCSP statut « *good* » pour un certificat non existant. Son rôle est, en effet, de statuer sur la révocation et non sur l'existence (*cette limitation s'applique également aux CRL*).

Ces limitations font de ce protocole un outil interessant mais incomplet. Il est cependant possible que, dans les années qui viennent, l'initiale RFC 2560 qui définit OCSP soit revue et améliorée. En attendant, l'utilisation conjointe d'OCSP et des CRL fournit une solution de gestion de la révocation des certificats X.509 efficace.

6

Conclusion

Ce projet à évolué en même temps que ma compréhension de ses différents composants mais également en même temps que les besoins et moyens de Microgate.Pro. Il y a une grande différence entre connaître l'existence d'un protocole et le déployer dans une entreprise qui possède ses propres contraintes techniques, économiques et humaines. Ce stage m'a permis de mieux évaluer cette différence.

Certains sujets que j'ai eu l'occasion d'aborder n'ont volontairement pas été présenté ici car au fil de leurs déploiements ils ont été abandonnés pour diverses raisons. La plus fréquente étant qu'ils ne convenaient pas à ce que nous recherchions, c'est-à-dire une structure à la fois solide, scalable et commercialement attrayante.

J'ai acquis une bonne expérience concernant, tous d'abord, les solutions de **P**ublic **K**ey **I**nfrastructure et de **V**irtual **P**rivate **N**etwork, mais également et surtout une meilleure vision de ce qu'est la Sécurité Informatique appliquée en Entreprise du point de vue de l'administrateur comme de l'utilisateur.

C'est ce dernier aspect qui m'a d'ailleurs orienté vers la PKI, ceci afin de fournir un bon niveau de sécurité sans pour autant incommoder l'utilisateur. La réalisation de cette PKI, à l'image de son importance dans ce mémoire, m'aura occupé un tiers de mon temps de stage. Cette dernière est aujourd'hui parfaitement fonctionnelle et je continuerais d'en assurer la maintenance pendant le passage de relais aux administrateurs de Microgate.Pro, cet été.

GLOSSAIRE

ISO

L'ISO *(Provient du grec isso qui signifie égal. Il ne s'agit pas d'un acronyme)* représente le symbole de l'*Organisation internationale de normalisation* ou International organization for standardization en anglais. Il s'agit d'une organisation internationale, composée de représentants des organismes de normalisation nationaux, qui produit des normes internationales dans des domaines industriels et commerciaux. Le secrétariat central de l'ISO est situé à Genève, en Suisse. Il assure aux membres de l'ISO le soutien administratif et technique, coordonne le programme décentralisé d'élaboration des normes et procède à leur publication.

NIS

Network Information Service (*Service d'information du réseau*): service qui permet à certaines informations d'être connues par toutes les machines disponibles sur le réseau. Ce service est géré dans la bibliothèque standard de la libc Linux. Il est considéré par la suite comme étant le "NIS traditionnel".

NIS+

Network Information Service (*Plus...*) : version de NIS améliorée. NIS+ a été conçu par Sun Microsystems Inc. pour remplacer NIS, avec un niveau de sécurité supérieur et une meilleure gestion pour les grosses installations.

Buffer Overflow

Le principe de ce type d'attaque est de profiter de l'accès à certaines variables du programme, souvent par le biais de fonctions telles `scanf()` (analyse de chaîne de caractères) ou `strcpy()` (copie de chaîne de caractères) en langage C, qui ne contrôlent pas la taille de la chaîne à enregistrer dans un tampon, afin d'écraser la mémoire du processeur jusqu'à l'adresse de retour de la fonction en cours d'exécution. On peut ainsi choisir quelles seront les prochaines instructions exécutées par le processeur. Le code introduit est généralement exécuté avec les droits du programme détourné.

ANNEXES

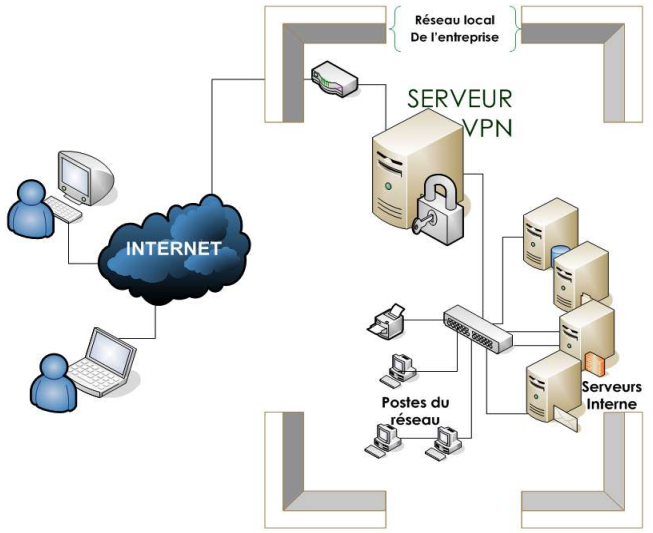
Solution VPN sécurisé

accèdez à votre réseau local depuis l'extérieur

La solution VPN (Virtual Private Network) permet à un ordinateur situé à l'extèrieur du réseau local d'accèder aux ressources interne de l'entreprise.

Ainsi, au travers d'une connection internet classique (type ADSL), il est possible de travailler sur le réseau de la même façon que si l'on était présent dans l'entreprise.

Partage sur le réseau, site Intranet, base de donnée, application professionnelle, ...
Tous ces outils informatiques habituellement limités aux réseaux locaux sont disponible de l'extérieur via une connection VPN.



Cette solution s'adresse principalement à deux catégories d'utilisateurs:

- ✔ Les nomades (ex: commerciaux,...) qui utilisent leurs ordinateurs portables ou qu'ils se trouvent.
- ✔ Les cadres et dirigeants désireux d'accèder à leurs données de chez eux ou de n'importe quel poste extérieur.

Sur le poste externe, le VPN nécessite simplement l'installation d'un petit logiciel. Une fois configuré il n'est pas plus difficile de se connecter au VPN que de se connecter à Internet.

Coté sécurité, Microgate.Pro gére les attributions et les révocations de Certificats Numériques. Ces certificats garantissent l'identité d'un utilisateur et sont protégés par un mot de passe fort.

De plus, cette technologie utilise la cryptographie pour interdire toute possibilités d'interception des données qui transitent entre l'utilisateur et le réseau de son entreprise.
Enfin, une clé secréte et spécifique à chaque serveur VPN est obligatoire pour s'y connecter.

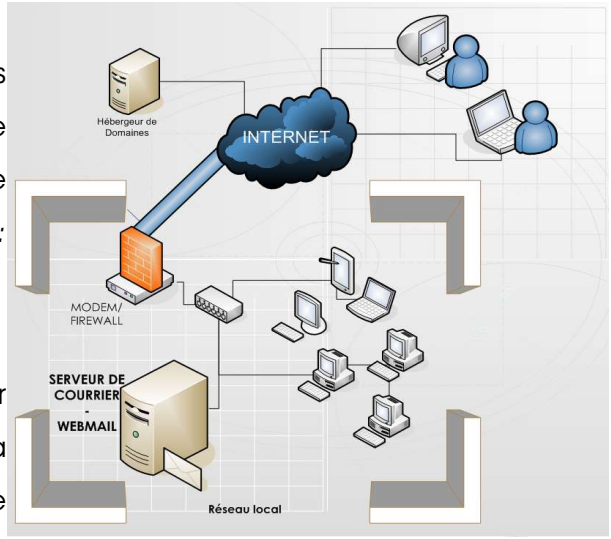
Solution Serveur de Courriers

système complet de gestion de vos boites e-mails

Le courrier électronique est devenu un outil standard de communication. Cette solution a été créée dans l'optique de fournir un serveur de courrier a **coût réduit** tout en intégrant des services de haut niveau. Elle comprend:

- ✔ L'envoi et la réception de courrier électronique sur un domaine que vous possédez. (*ex: direction@votre_entreprise.com*).
- ✔ Une **protection Anti-Virale** et **Anti-Spam** qui inspecte vos e-mails entrant et sortants et met en quarantaine ceux contenant des virus ou du spam.
- ✔ La gestion simplifiée des comptes utilisateurs via une interface web.
- ✔ Une machine serveur basée sur des Logiciels Libres (GNU/Linux) et conçues pour résister aux attaques informatiques comme aux pannes matérielles.

Cette solution intégre la gestion des "listes de diffusions". Ces dernières vous permettent de spécifier plusieurs destinataires pour une seule adresse e-mail (*ex: contact@votre_entreprise.com*).



Vous aurez également la possibilité de consulter votre courrier de n'importe quel point d'accès à Internet comme si vous étiez sur votre lieu de travail.

Microgate.Pro assure l'étude, le déploiement, la maintenance et la formation à l'utilisation des comptes e-mails.

Enfin, si vous souhaitez un **haut niveau de disponibilité**, Microgate.Pro prend en charge vos mails en cas de défaillance de votre connection Internet. *(Nous contacter pour plus d'informations)*

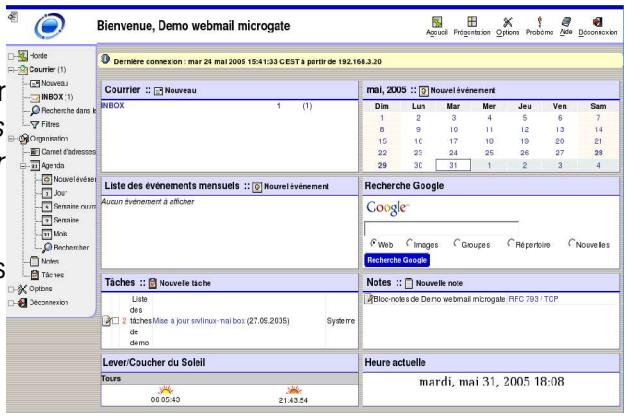
Solution Webmail

toute la puissance du travail en groupe sur votre navigateur Internet

La suite de logiciels du projet Horde/IMP vous permet d'**accèder très rapidement** aux fonctionnalités avancées des logiciels de travail en groupe.

Ce Webmail travaille conjointement avec votre serveur de courriers électronique et intègre les fonctionnalités suivantes:

Courrier:



- ✔ Gestion avancée de votre courrier électronique (*filtrage par listes blanche/noire, classement par dossier, ...*)
- ✔ Recherche sur de nombreux critères dans toute votre boite électronique

Organisation:

- ✔ Gestion avancée de votre emploi du temps (*par jour, semaine, mois; onglet de recherche; alarme de rappel;...*)
- ✔ Partage de votre emploi du temps avec vos collaborateurs
- ✔ Organisation de réunion/évènements intégrant :
  _contrôle de disponibilité
  _accusé de réception et d'acceptation
- ✔ Gestion des tâches par catégorie

L'accés au Webmail se fait par une adresse Internet, utilisable en interne de votre réseau comme de n'importe quel accès Internet externe.

Sur le plan de **la sécurité**, la Solution Webmail intègre deux protocoles:

1. Le protocole "https" de sécurisation des pages web. Ainsi tout ce que vous faites sur le Webmail est privé et sécurisé.

2. Le protocole "PGP" qui vous permet de certifier à vos interlocuteurs l'intégrité des messages que vous leurs destinez.

+==+
ANALYSE DES LOGS DU SERVEUR DE COURRIERS
+==+

Passage du mail au daemon Amavis via une socket Unix

Jun 6 17:08:09 srvlinux-mailbox amavis[29342]: (29342-06-2) LMTP::10024 / var/lib/amavis/amavis-20050606T160806-29342: <thsimon@hathway.com> -> <info@microgate.fr> Received: SIZE=1459 BODY=8BITMIME from srvlinux-mailbox.microgate.fr ([127.0.0.1]) by localhost (srvlinux-mailbox.microgate.fr [127.0.0.1]) (amavisd-new, port 10024) with LMTP id 29342-06-2 for <info@microgate.fr>; Mon, 6 Jun 2005 17:08:09 +0200 (CEST)

Début de l'inspection bayesienne

Jun 6 17:08:09 srvlinux-mailbox amavis[29342]: (29342-06-2) Checking: <thsimon@hathway.com> -> <info@microgate.fr>

Somme des "hits": on arrive ici a 7.166
Les "hits" sont énumérés dans le "tests=...."

Jun 6 17:08:09 srvlinux-mailbox amavis[29342]: (29342-06-2) spam_scan: hits=7.166 tests=DRUGS_ERECTILE,DRUG_ED_CAPS,HTML_30_40,HTML_MESSAGE,INFO_TLD,MIME_HTML_ONLY,SUBJECT_DRUG_GAP_C

Déplacement du mail en quarantaine

Jun 6 17:08:09 srvlinux-mailbox amavis[29342]: (29342-06-2) local delivery: <> -> <spam-quarantine>, mbx=/var/lib/amavis/virusmails/spam-a81395b6d116dcb56881a87d273b253b-20050606-170809-29342-06-2.gz

Modification de l'entête du mail

Jun 6 17:08:09 srvlinux-mailbox amavis[29342]: (29342-06-2) SPAM, <thsimon@hathway.com> -> <info@microgate.fr>, Yes, hits=7.2 tag1=4.0 tag2=6.3 kill=6.3 tests=DRUGS_ERECTILE, DRUG_ED_CAPS, HTML_30_40, HTML_MESSAGE, INFO_TLD, MIME_HTML_ONLY, SUBJECT_DRUG_GAP_C, quarantine spam-a81395b6d116dcb56881a87d273b253b-20050606-170809-29342-06-2 (spam-quarantine)

[.....................]

Le daemon Amavis avise le daemon Master de Postfix via LMTP que le mail est rejeté

Jun 6 17:08:09 srvlinux-mailbox postfix/lmtp[4945]: 2696E3A2323: to=<info@microgate.fr>, relay=127.0.0.1[127.0.0.1], delay=6, status=bounced (host 127.0.0.1[127.0.0.1] said: 550 5.7.1 Message content rejected, UBE, id=29342-06-2 (in reply to end of DATA command))

[.............]

Le démon Master de Postfix avise le SMTP distant que le mail est rejeté
et s'aperçoit alors que l'émetteur n'existe pas....

Jun 6 17:08:13 srvlinux-mailbox postfix/smtp[5353]: CEB473A2B1F: to=<thsimon@hathway.com>, relay=mail.hathway.com[202.88.130.5], delay=4, status=bounced (host mail.hathway.com[202.88.130.5] said: 550 5.1.1 unknown or illegal alias: thsimon@hathway.com (in reply to RCPT TO command))

ANALYSE DU CONTENU DU MAIL EN QUARANTAINE

En analysant le contenu du mail dans sa cage de quarantaine, on voit bien qu'il possède tous les critères du spam: syntaxe HTML, intégration d'une URL douteuse, vente de produit miracle...

srvlinux-mailbox:/var/lib/amavis/virusmails# cat spam-a81395b6d116dcb56881a87d273b253b-20050606-170809-29342-06-2

```
Return-Path: <>
Delivered-To: spam-quarantine
X-Envelope-To: <info@microgate.fr>
X-Envelope-From: <thsimon@hathway.com>
X-Quarantine-id: <spam-a81395b6d116dcb56881a87d273b253b-20050606-170809-29342-06-2>
Received: from ?emailtraveller.com (unknown [221.214.231.252])
        by srvlinux-mailbox.microgate.fr (Postfix) with SMTP id 2696E3A2323
        for <info@microgate.fr>; Mon,  6 Jun 2005 17:08:03 +0200 (CEST)
Received: from 212.234.37.131
        (SquirrelMail authenticated user thsimon@hathway.com);
        by emailtraveller.com with HTTP id J85Gz006123763;
        Mon, 06 Jun 2005 15:11:28 +0000
Message-Id: <nx8RtZ.squirrel@212.234.37.131>
Date: Mon, 06 Jun 2005 15:11:28 +0000
Subject: Save your money buy getting CIALIS here
From: "Kaley Flores" <thsimon@hathway.com>
To: info@microgate.fr
User-Agent: SquirrelMail/1.4.3a
X-Mailer: SquirrelMail/1.4.3a
MIME-Version: 1.0
Content-Type: text/html; charset=iso-8859-1
Content-Transfer-Encoding: 8bit
X-Priority: 3 (Normal)
Importance: Normal
X-Spam-Status: Yes, hits=7.2 tag1=4.0 tag2=6.3 kill=6.3 tests=DRUGS_ERECTILE,
        DRUG_ED_CAPS, HTML_30_40, HTML_MESSAGE, INFO_TLD, MIME_HTML_ONLY,
        SUBJECT_DRUG_GAP_C
X-Spam-Level: *******

<html>
<body>
<table>
<tr>
<p>
You have not tried <a
href="http://blmacehkfgj.medkit.info/?difgjxssryblmzsvacehk">Cialls</a> yet?<br>
Than you cannot even imagine what it is like to be a real man in bed!<br>
The thing is that a great errrect1on is provided for you exactly when you want.<br>
[...]
</p>
</tr>
</table>
</body>
</html>
```

Déroulement d'une procèdure de contrôle OCSP via le navigateur Mozilla.

Les deux premiers paquets (17 & 20) sont la requête. La méthode POST de HTTP est utilisée, le corps du message est la valeur dinaire de l'encodae "DER de OCSPRequest" (voir RFC2560).

Les deux paquets suivants (22 & 23) constituent la réponse du répondeur OCSP situé à l'adresse ocsp.microgate.fr. Le format des champs data de la réponse est la valeur bianire de l'encodage DER "OCSPResponse".

Pour plus de lisibilité, les paquets d'acquittements, résolutions DNS et fin de session TCP ne sont pas présents ici.

```
No.     Time        Source                Destination        Protocol Info
17    0.525231    192.168.1.7           82.228.157.21        HTTP POST/HTTP/1.0

Frame 17 (162 bytes on wire, 162 bytes captured)
    [.............]
Ethernet II, Src: 00:04:75:d6:2c:34, Dst: 00:04:76:97:73:ac
    [.............]
Internet Protocol, Src Addr: 192.168.1.7 (192.168.1.7), Dst Addr: 82.228.157.21
(82.228.157.21)
    [.............]
Transmission Control Protocol, Src Port: 2799 (2799), Dst Port: http (80), Seq:
1, Ack: 1, Len: 108
    [.............]
Hypertext Transfer Protocol
    POST / HTTP/1.0\r\n
        Request Method: POST
        Request URI: /
        Request Version: HTTP/1.0
    Host: ocsp.microgate.fr:80\r\n
    Content-Type: application/ocsp-request\r\n
    Content-Length: 100\r\n
    \r\n

    [.............]
0030  ff ff b2 2f 00 00 50 4f 53 54 20 2f 20 48 54 54   .../..POST / HTT
0040  50 2f 31 2e 30 0d 0a 48 6f 73 74 3a 20 6f 63 73   P/1.0..Host: ocs
0050  70 2e 6d 69 63 72 6f 67 61 74 65 2e 66 72 3a 38   p.microgate.fr:8
0060  30 0d 0a 43 6f 6e 74 65 6e 74 2d 54 79 70 65 3a   0..Content-Type:
0070  20 61 70 70 6c 69 63 61 74 69 6f 6e 2f 6f 63 73    application/ocs
0080  70 2d 72 65 71 75 65 73 74 0d 0a 43 6f 6e 74 65   p-request..Conte
0090  6e 74 2d 4c 65 6e 67 74 68 3a 20 31 30 30 0d 0a   nt-Length: 100..
```

```
No.     Time        Source                Destination       Protocol Info
20    0.585131    192.168.1.7           82.228.157.21       HTTP Continuation
                                                            or non-HTTP traffic

Frame 20 (154 bytes on wire, 154 bytes captured)
      [.............]
    Protocols in frame: eth:ip:tcp:http:data
Ethernet II, Src: 00:04:75:d6:2c:34, Dst: 00:04:76:97:73:ac
      [.............]
Internet Protocol, Src Addr: 192.168.1.7 (192.168.1.7), Dst Addr: 82.228.157.21
(82.228.157.21)
      [.............]
Transmission Control Protocol, Src Port: 2799 (2799), Dst Port: http (80), Seq:
109, Ack: 1, Len: 100
    Source port: 2799 (2799)
    Destination port: http (80)
    Sequence number: 109    (relative sequence number)
    Next sequence number: 209    (relative sequence number)
    Acknowledgement number: 1    (relative ack number)
    Header length: 20 bytes
    Flags: 0x0018 (PSH, ACK)
        [.............]
    Window size: 65535
    Checksum: 0xb227 (incorrect, should be 0x6f4f)
Hypertext Transfer Protocol
    Data (100 bytes)

0000  00 04 76 97 73 ac 00 04 75 d6 2c 34 08 00 45 00   ..v.s...u.,4..E.
0010  00 8c 4f ae 40 00 80 06 f9 14 c0 a8 01 07 52 e4   ..O.@.........R.
0020  9d 15 0a ef 00 50 bf 4b ea 24 e8 2a 37 c0 50 18   .....P.K.$.*7.P.
0030  ff ff b2 27 00 00 30 62 30 60 30 3e 30 3c 30 3a   ...'..0b0`0>0<0:
0040  30 09 06 05 2b 0e 03 02 1a 05 00 04 14 e3 1c a2   0...+...........
0050  53 af 4b 08 b1 a0 2a 5a 0a 75 20 44 5a 6b 76 10   S.K...*Z.u DZkv.
0060  a7 04 14 26 55 24 d3 a2 08 27 96 39 09 c7 4a b9   ...&U$...'.9..J.
0070  d8 11 2e fc c6 d6 5b 02 01 03 a2 1e 30 1c 30 1a   ......[.....0.0.
0080  06 09 2b 06 01 05 05 07 30 01 04 04 0d 30 0b 06   ..+.....0....0..
0090  09 2b 06 01 05 05 07 30 01 01                     .+.....0..
```

```
No.     Time        Source                Destination    Protocol Info
22    0.743483    82.228.157.21         192.168.1.7          OCSP Response
                                                           [Unreassembled Packet]

Frame 22 (1454 bytes on wire, 1454 bytes captured)
    [.............]
    Protocols in frame: eth:ip:tcp:http:ocsp
Ethernet II, Src: 00:04:76:97:73:ac, Dst: 00:04:75:d6:2c:34
    [.............]
Internet Protocol, Src Addr: 82.228.157.21 (82.228.157.21), Dst Addr:
192.168.1.7 (192.168.1.7)
    [.............]
Transmission Control Protocol, Src Port: http (80), Dst Port: 2799 (2799), Seq:
1, Ack: 209, Len: 1400
    [.............]
Hypertext Transfer Protocol
    HTTP/1.0 200 OK\r\n
        Request Version: HTTP/1.0
        Response Code: 200
    Content-type: application/ocsp-response\r\n
    Content-Length: 1921\r\n
    \r\n
Online Certificate Status Protocol
[Unreassembled Packet: OCSP]
```

```
0000  00 04 75 d6 2c 34 00 04 76 97 73 ac 08 00   6e 74   urity Department
45 00   ..u.,4..v.s...E.                          0110  31 1a 30 18 06 03 55 04 03 13 11 6f 63 73
0010  05 a0 90 1b 40 00 33 06 00 94 52 e4 9d 15   70 2e   1.0...U....ocsp.
c0 a8   ....@.3...R.....                          0120  6d 69 63 72 6f 67 61 74 65 2e 66 72 31 24
0020  01 07 00 50 0a ef e8 2a 37 c0 bf 4b ea 88   30 22   microgate.fr1$0"
50 10   ...P...*7..K..P.                          0130  06 09 2a 86 48 86 f7 0d 01 09 01 16 15 73
0030  16 d0 40 e1 00 00 48 54 54 50 2f 31 2e 30   65 63   ..*.H........sec
20 32   ..@...HTTP/1.0 2                          0140  75 72 69 74 79 40 6d 69 63 72 6f 67 61 74
0040  30 30 20 4f 4b 0d 0a 43 6f 6e 74 65 6e 74   65 2e   urity@microgate.
2d 74   00 OK..Content-t                          0150  66 72 18 0f 32 30 30 35 30 35 31 36 31 35
0050  79 70 65 3a 20 61 70 70 6c 69 63 61 74 69   32 35   fr..200505161525
6f 6e   ype: application                          0160  34 33 5a 30 51 30 4f 30 3a 30 09 06 05 2b
0060  2f 6f 63 73 70 2d 72 65 73 70 6f 6e 73 65   0e 03   43Z0Q0O0:0...+..
0d 0a   /ocsp-response..                          0170  02 1a 05 00 04 14 e3 1c a2 53 af 4b 08 b1
0070  43 6f 6e 74 65 6e 74 2d 4c 65 6e 67 74 68   a0 2a   .........S.K...*
3a 20   Content-Length:                          0180  5a 0a 75 20 44 5a 6b 76 10 a7 04 14 26 55
0080  31 39 32 31 0d 0a 0d 0a 30 82 07 7d 0a 01   24 d3   Z.u DZkv....&U$.
00 a0   1921....0..}....                         0190  a2 08 27 96 39 09 c7 4a b9 d8 11 2e fc c6
0090  82 07 76 30 82 07 72 06 09 2b 06 01 05 05   d6 5b   ..'.9..J.......[
07 30   ..v0..r..+.....0                         01a0  02 01 03 82 00 18 0f 32 30 30 35 30 35 31
00a0  01 01 04 82 07 63 30 82 07 5f 30 82 01 08   36 31   .......200505161
a1 81   .....c0.._0.....                         01b0  35 32 35 34 33 5a 30 0d 06 09 2a 86 48 86
00b0  a1 30 81 9e 31 0b 30 09 06 03 55 04 06 13   f7 0d   52543Z0...*.H...
02 46   .0..1.0...U....F                         01c0  01 01 05 05 00 03 82 01 01 00 94 ee e2 fa
00c0  52 31 17 30 15 06 03 55 04 08 13 0e 49 6e   a8 cf   ................
64 72   R1.0...U....Indr                         01d0  99 c6 fc a6 28 92 f8 d7 aa ab 31 7d f2 23
00d0  65 2d 65 74 2d 4c 6f 69 72 65 31 0e 30 0c   d0 ca   ....(.....1}.#..
06 03   e-et-Loire1.0...                         01e0  14 b6 90 ac 32 4b 03 69 af 97 ab e7 0d af
00e0  55 04 07 13 05 54 6f 75 72 73 31 24 30 22   60 c3   ....2K.i......`.
06 03   U....Tours1$0"..                         01f0  ff 92 95 3a 1f f9 ea 65 5c da ca 63 c2 4c
00f0  55 04 0a 13 1b 52 65 73 67 61 74 65 20 53   88 61   ...:...e\..c.L.a
65 63   U....Resgate Sec                         0200  fa 6e f9 a2 0a 22 b8 5f 6d 49 cf 11 52 dd
0100  75 72 69 74 79 20 44 65 70 61 72 74 6d 65   af f2   .n..."._mI..R...
```

```
0210  3c d6 da 55 be 52 22 c6 6f a2 6d cd 9f 35   03e0  31 0e 30 0c 06 03 55 04 07 13 05 54 6f 75
9f b8   <..U.R".o.m..5..                          72 73   1.0...U....Tours
0220  7f d5 e2 99 5f e5 62 87 b8 9d d9 85 88 fa   03f0  31 24 30 22 06 03 55 04 0a 13 1b 52 65 73
e9 fc   ...._.b.........                          67 61   1$0"..U....Resga
0230  6e e0 12 4f 81 b3 08 00 77 43 a4 63 5b 9e   0400  74 65 20 53 65 63 75 72 69 74 79 20 44 65
71 60   n..O....wC.c[.q`                          70 61   te Security Depa
0240  84 a7 69 7e d1 15 c0 8c c8 94 bc da d5 a9   0410  72 74 6d 65 6e 74 31 1a 30 18 06 03 55 04
21 14   ..i~..........!.                          03 13   rtment1.0...U...
0250  dd cc 99 c1 91 2c 2e ce cf aa d6 67 bc 92   0420  11 6f 63 73 70 2e 6d 69 63 72 6f 67 61 74
42 cf   .....,.....g..B.                          65 2e   .ocsp.microgate.
0260  b7 19 66 e9 ed 10 ab 0e 5e cb f4 25 07 bc   0430  66 72 31 24 30 22 06 09 2a 86 48 86 f7 0d
6e 36   ..f.....^..%..n6                          01 09   fr1$0"..*.H.....
0270  88 fe 30 79 b5 4d bc a7 91 aa 62 a6 37 4c   0440  01 16 15 73 65 63 75 72 69 74 79 40 6d 69
8a f4   ..0y.M....b.7L..                          63 72   ...security@micr
0280  66 78 b4 b2 cf 14 54 d6 df 0b ef e3 a0 11   0450  6f 67 61 74 65 2e 66 72 30 82 01 22 30 0d
91 9b   fx....T.........                          06 09   ogate.fr0.."0...
0290  14 4d 0f a7 96 eb d5 7c 59 a6 c9 a9 90 13   0460  2a 86 48 86 f7 0d 01 01 01 05 00 03 82 01
49 57   .M.....|Y.....IW                          0f 00   *.H.............
02a0  b6 72 36 9d 06 51 d8 66 60 3b 44 d1 c7 31   0470  30 82 01 0a 02 82 01 01 00 c0 77 fa 3a b2
03 07   .r6..Q.f`;D..1..                          7e ac   0.........w.:.~.
02b0  56 4c 1a fd 3c 47 b5 21 82 ed 00 ff 5f 64   0480  87 c1 69 3c 38 79 8b 92 bb 48 48 90 dc 4b
47 57   VL..<G.!...._dGW                          1e 29   ..i<8y...HH..K.)
02c0  9c 63 c0 e6 b1 9e a1 63 72 34 a0 82 05 3b   0490  d6 cf fa 5c b8 f2 f6 f1 be fd 36 8c 36 52
30 82   .c.....cr4...;0.                          51 99   ...\......6.6RQ.
02d0  05 37 30 82 05 33 30 82 04 1b a0 03 02 01   04a0  06 b3 6d 3b 8b 81 c7 4e 53 87 b9 86 3f f0
02 02   .70..30.........                          52 de   ..m;...NS...?.R.
02e0  01 02 30 0d 06 09 2a 86 48 86 f7 0d 01 01   04b0  58 ad b3 fa da 9e 1a 9e 4e 93 40 88 09 f5
05 05   ..0...*.H.......                          a7 50   X.......N.@....P
02f0  00 30 81 a3 31 0b 30 09 06 03 55 04 06 13   04c0  a2 83 fc 48 10 64 f0 cf 51 3e ea c7 a6 38
02 46   .0..1.0...U....F                          94 a2   ...H.d..Q>...8..
0300  52 31 17 30 15 06 03 55 04 08 13 0e 49 6e   04d0  77 d8 58 20 a1 d7 6b 26 06 fb f3 c9 41 21
64 72   R1.0...U....Indr                          1a b7   w.X ..k&....A!..
0310  65 2d 65 74 2d 4c 6f 69 72 65 31 0e 30 0c   04e0  08 1e 8e 11 09 7c dd aa 93 28 0e 8c 68 1d
06 03   e-et-Loire1.0...                          20 eb   .....|...(..h. .
0320  55 04 07 13 05 54 6f 75 72 73 31 24 30 22   04f0  90 17 72 7e c9 12 8a 29 51 d9 51 02 e3 06
06 03   U....Tours1$0"..                          e0 6f   ..r~...)Q.Q....o
0330  55 04 0a 13 1b 52 65 73 67 61 74 65 20 53   0500  ca 41 ed f9 76 2f a8 21 30 a5 57 86 ad d1
65 63   U....Resgate Sec                          b4 c5   .A..v/.!0.W.....
0340  75 72 69 74 79 20 44 65 70 61 72 74 6d 65   0510  af 11 5e 48 a0 47 23 ed b0 0e 8a b9 8e 6b
6e 74   urity Department                          78 bd   ..^H.G#......kx.
0350  31 1f 30 1d 06 03 55 04 03 13 16 61 75 74   0520  8d 7e 75 22 3a f8 3e 84 74 0f 50 ab 23 e2
68 6f   1.0...U....autho                          75 15   .~u":.>.t.P.#.u.
0360  72 69 74 79 2e 6d 69 63 72 6f 67 61 74 65   0530  44 c9 72 0a 0d 89 1e 5e ea 3c 99 8e 2a dc
2e 66   rity.microgate.f                          2e 70   D.r....^.<..*..p
0370  72 31 24 30 22 06 09 2a 86 48 86 f7 0d 01   0540  16 fe 34 55 e5 59 9a c5 ba 2c 40 c4 d3 72
09 01   r1$0"..*.H......                          55 51   ..4U.Y...,@..rUQ
0380  16 15 73 65 63 75 72 69 74 79 40 6d 69 63   0550  cd 06 ff e1 c9 4c 82 93 a4 63 7c 88 cf 06
72 6f   ..security@micro                          ac e2   .....L...c|.....
0390  67 61 74 65 2e 66 72 30 1e 17 0d 30 35 30   0560  a1 f3 61 18 bc 04 89 5a cf fa db 3f f8 29
35 30   gate.fr0...05050                          bb 4e   ..a....Z...?.).N
03a0  32 31 32 30 36 35 36 5a 17 0d 31 35 30 34   0570  7e ee 47 05 c2 75 c5 de a9 02 03 01 00 01
33 30   2120656Z..150430                          a3 82   ~.G..u..........
03b0  31 32 30 36 35 36 5a 30 81 9e 31 0b 30 09   0580  01 73 30 82 01 6f 30 1d 06 09 60 86 48 01
06 03   120656Z0..1.0...                          86 f8   .s0..o0...`.H...
03c0  55 04 06 13 02 46 52 31 17 30 15 06 03 55   0590  42 01 0d 04 10 16 0e 4f 43 53 50 20 52 65
04 08   U....FR1.0...U..                          73 70   B......OCSP Resp
03d0  13 0e 49 6e 64 72 65 2d 65 74 2d 4c 6f 69   05a0  6f 6e 64 65 72 30 1d 06 03 55 1d 0e 04 16
72 65   ..Indre-et-Loire                          onder0...U....
```

```
No.     Time        Source                Destination    Protocol Info
23    0.750183    82.228.157.21         192.168.1.7      HTTP Continuation or
                                                           non-HTTP traffic

Frame 23 (657 bytes on wire, 657 bytes captured)
    [.............]
Ethernet II, Src: 00:04:76:97:73:ac, Dst: 00:04:75:d6:2c:34
    [.............]
Internet Protocol, Src Addr: 82.228.157.21 (82.228.157.21), Dst Addr:
192.168.1.7 (192.168.1.7)
    [.............]
Transmission Control Protocol, Src Port: http (80), Dst Port: 2799 (2799), Seq:
1401, Ack: 209, Len: 603
    [.............]
Hypertext Transfer Protocol
    Data (603 bytes)

0000  00 04 75 d6 2c 34 00 04 76 97 73 ac 08 00 45 00   ..u.,4..v.s...E.
[.............]
0280  15 07 76 51 7a a6 38 46 5a 0f 7a 73 5d a6 e0 3d   ..vQz.8FZ.zs]..=
0290  8f                                                .
```

Bibliographie

Les VPN
R. Corvalon, E. Corvalon, Y. Le Corvic; *Editions E. DUNOD*

Halte aux Hackers
S. Mc Clure, J. Scambray, G. Kurtz; *Editions Eyrolles*

Serveurs réseau Linux
Graig Hunt; *Editions Eyrolles*

Securing and Optimizing Linux
G. Mourani

Postfix (la référence)
Kyle Dent; *Editions O'Reilly*

Guide avancé d'écriture des scripts bash
Ecrit par la communauté Internet

Les Iptables Linux
G. N. Purdy; *Editions O'Reilly*

Transmission et réseaux
S. Lohier, D. Présent; *Editions E. Dunod*

.....et de trop nombreux sites Internet pour que je ne puissent tous les citer. Enfin, je souhaite remercier Simon Havard pour le superbe travail d'illustation qu'il a effectué sur la page de garde et les pages 9 & 18.