

Formation   
Linux – Sécurité

Formation continue  
Collège de Maisonneuve  
2030, boulevard Pie IX, bureau 430,  
Montréal (Québec) H1V 2C8  
514.251-7131 poste 4800

Document proposé par Louis Savard, M. Ing. TI  
Dernière révision : Février 2015

# Table des matières

[Table des matières 3](#_Toc410888430)

[Introduction 5](#_Toc410888431)

[Processus de classification de l'information 7](#_Toc410888432)

[Gnu Privacy Guard(GPG) 9](#_Toc410888433)

[Gestion des accès aux fichiers 11](#_Toc410888434)

[Les listes de contrôle d'accès (ACLs) *Access Control Lists* 15](#_Toc410888435)

[Chiffrement et certificats 18](#_Toc410888436)

[IPsec 22](#_Toc410888437)

[Authentification 27](#_Toc410888438)

[Mots de passe 30](#_Toc410888439)

[Contrôle d’accès 33](#_Toc410888440)

[Haute disponibilité 40](#_Toc410888441)

[Laboratoire- Répartition de charge 49](#_Toc410888442)

[Transport Layer Secure - Secure Socket Layer 53](#_Toc410888443)

[Certificat numérique 55](#_Toc410888444)

[Exercice SSL (1) 58](#_Toc410888445)

[Exercice SSL (2) 62](#_Toc410888446)

[Sécure Shell (SSH) 69](#_Toc410888447)

[Activité sur le protocole SSH 72](#_Toc410888448)

[Modules PAM 80](#_Toc410888449)

[Outils réseaux 85](#_Toc410888450)

[Pare-feu sous Linux 91](#_Toc410888451)

[*iptables* 100](#_Toc410888452)

[*pfsense* 110](#_Toc410888453)

[Outils d’intrusion et de détection 125](#_Toc410888454)

[Références 131](#_Toc410888455)

[Annexe A … Analyse de risque 132](#_Toc410888456)

[Annexe B … Introduction à la cryptographie 135](#_Toc410888457)

[Annexe C … nmap – Résumé des principaux commutateurs 142](#_Toc410888458)

[Annexe D … Exercice - Mise en place de NAT sous Linux avec iptables 148](#_Toc410888459)

[Annexe E … Politique de sécurité 155](#_Toc410888460)

[Annexe F… Modification du mot de passe *root* sous Linux 164](#_Toc410888461)

[Annexe G… Sécuriser un serveur Linux 165](#_Toc410888462)

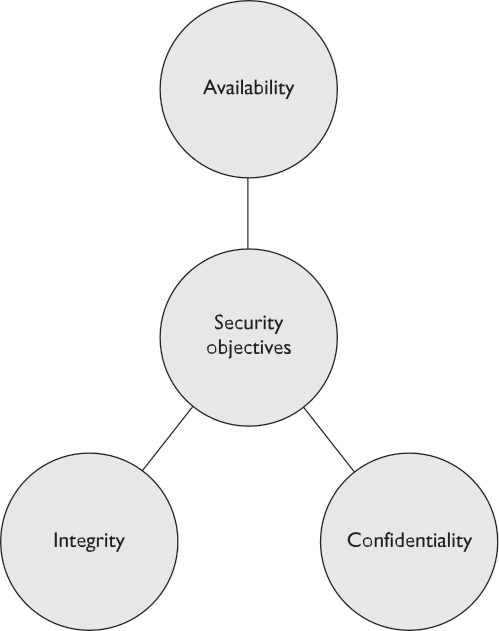
# Introduction

**Principes de base**

* Le degré de sécurité est proportionnel au degré de solidité du maillon le plus faible;
* Le niveau de sécurité est inversement proportionnel à la facilité d’utilisation;
* La sécurité se définit comme étant la gestion du risque;
* La sécurité n’est pas un ensemble de technologies mais plutôt un processus.
* La sécurité n’est pas seulement une réponse technique.

**Les trois grands (*the big 3*):**

* **Confidentialité** (niveau nécessaire du secret)  
  Seules les personnes autorisées ont accès aux informations qui leur sont destinées. Tout accès indésirable doit être empêché.
* **Intégrité** (exactitude et fiabilité)  
  Les données doivent être celles que l'on attend, et ne doivent pas être altérées de façon fortuite, illicite ou malveillante. En clair, les éléments considérés doivent être exacts et complets.  
  et
* **disponibilité** (fiabilité et accès en temps opportun)   
  Le système doit fonctionner sans faille durant les étendues d'utilisation prévues et garantir l'accès aux services et ressources installées avec le temps de réponse attendu.



D'autres aspects peuvent aussi être considérés comme des objectifs de la sécurité, tels que :

* La **traçabilité**  
  Garantie que les accès et tentatives d'accès aux éléments considérés sont tracés et que ces traces sont conservées et exploitables.
* **L'authentification**  
  L'identification des utilisateurs est fondamentale pour gérer les accès aux espaces de travail pertinents et maintenir la confiance dans les relations d'échange.
* **La non-répudiation et l'imputation**  
  Aucun utilisateur ne doit pouvoir contester les opérations qu'il a réalisées dans le cadre de ses actions autorisées, et aucun tiers ne doit pouvoir s'attribuer les actions d'un autre utilisateur.

**Risque, menaces, vulnérabilité et impact**

* **Risque**Combinaison d'une menace et des pertes qu'elle peut engendrer: c'est-à-dire de la potentialité de l'exploitation de vulnérabilité par un élément menaçant et de l'impact sur l'organisme.
* **Menace**Attaque possible d'un individu ou d'un élément naturel sur des biens (ici, des informations) entraînant des conséquences potentielles négatives. Elle est souvent caractérisée par une expertise de l'attaquant, ses ressources disponibles et sa motivation. Exemple: un développeur modifie le code source en vue de détournement de fonds (grande expertise et forte motivation si les flux financiers sont importants), vol d'un ordinateur portable lors d'un déplacement (peu d'expertise nécessaire),...
* **Vulnérabilité**Caractéristique d'une entité qui peut constituer une faiblesse ou une faille au regard de la sécurité de l'information. Ces vulnérabilités peuvent être organisationnelle (ex: pas de politique de sécurité), humaine (ex: pas de formation des personnels), logicielles ou matérielles (ex: utilisation de produits peu fiables ou non testés),...Exemple: les fichiers sur les ordinateurs portables ne sont pas protégés en lecture.
* **Impact**Conséquence sur l'organisme de la réalisation d'une menace. L'impact peut être exprimé financièrement, ou dans une échelle qui dépend du contexte.

On remarquera que la notion de risque dépend de l'impact: une menace ayant une grande probabilité de se concrétiser, mais ayant un impact nul constitue un risque nul. Dans le langage courant, l'acceptation du mot "risque" peut être un peu différente et ne pas intégrer la notion d'impact.

# Processus de classification de l'information

**Objectifs de la classification de l'information**

Toutes les données ne possèdent la même valeur à une organisation.

1. Certaines données sont plus précieuses pour les gens qui prennent des décisions stratégiques, car il les aide à prendre à long terme ou à courte portée des décisions de la direction d'entreprise.
2. Certaines données, telles que les secrets commerciaux, les formules, et de nouvelles informations sur les produits, sont si précieuses que sa perte pourrait créer un important problème pour l'entreprise sur le marché en créant un embarras publique ou en provoquant un manque de crédibilité.

Avantages de classification de l'information

* Démontre un engagement organisation pour la protection des données ;
* Permet d'identifier quelles sont les informations les plus sensibles ou vitales à une organisation ;
* Supporte les principes de confidentialité, d'intégrité et de disponibilité que ce qui a trait aux données ;
* Permet d'identifier les protections pouvant s'appliquer aux informations qui pourraient être nécessaires pour des raisons juridiques, réglementaires ou de conformité.

**Terminologie**

* **Non classés**  
  Renseignements désignés comme étant ni sensible ni classifiées.
* **Sensible mais non classifié (SBU)**  
  Informations désignées comme un secret mineur mais pourrait ne pas créer de graves dommages en cas de divulgation.
* **Confidentiel**  
  La divulgation non autorisée de ces informations pourrait causer des dommages à l’entreprise.
* **Secret**  
  La divulgation non autorisée de ces informations pourrait causer de graves dommages à l’entreprise.
* **Top Secret**  
  Le plus haut niveau de classification de l'information. La divulgation non autorisée de ces informations secrètes causera un préjudice exceptionnellement grave pour l’entreprise.

Autre méthode de classification de l’information

* **Publique**  
  Informations semblables aux informations non classifiées.  
  Cette information ne devrait probablement pas être divulguée. Toutefois si elle est divulguée il ne devrait pas avoir d’impact sérieux ou négatif pourl’entreprise.
* **Sensible**  
  Information qui nécessite un niveau de classification plus élevé que des données normales. Cette information doit être protégée pour sa confidentialité ainsi que la perte d'intégrité (en raison d'une modification non autorisée).
* **Privée**  
  Information qui est considéré comme de nature personnelle et est réservée pour une utilisation seulement à l’intérieur de l’entreprise.   
  Sa divulgation pourrait nuire à la société ou ses employés.   
  Par exemple, les échelles de salaire ou encore les informations médicales sont considérées comme privées.
* **Secrète**  
  Information qui est considéré comme très sensible et qui est destinée à un usage interne uniquement.  
   Sa divulgation non autorisée pourrait impact négatif sérieux sur l’entreprise.   
  Par exemple, des informations sur le développement de nouveaux produits, des secrets commerciaux et des négociations de fusion sont considérées comme confidentielles.

**Critères de classification**

* **Valeur**  
  Numéro un des critères les plus couramment utilisés pour la classification des données dans le secteur privé.
* **Âge**  
  La classification de l'information pourrait être réduit car cette valeur diminue avec le temps. Dans certains cas, certains documents classifiés sont automatiquement déclassifiés après qu’une période de temps prédéterminée s'est écoulée.
* **Vie utile**  
  Si une information devient obsolète en raison de la venue de nouvelles informations, de changements importants au niveau de l’entreprise ( ou tout autres raisons) cette information est souvent déclassifiée.
* **Informations personnelles**  
  Si une information est personnellement associé à un ou des individus spécifiques ou est protégée par une loi de la vie privée elle devrait classée.   
  Par exemple, les données d'enquête qui révèle des noms pourraient avoir besoin de demeurer classée.

**Procédure pour classifier les informations**

1. Identifier l'administrateur / le dépositaire.
2. Préciser les critères sur la façon de classer et d'étiqueter l'information.
3. Classer les données selon son propriétaire et doit faire l'objet d'un examen par un superviseur.
4. Définir et documenter les exceptions à la politique de classification.
5. Spécifier les contrôles qui seront appliquées à chaque niveau de classification.
6. Préciser les modalités de résiliation pour la déclassification de l'information ou  
   pour le transfert de la garde de l'information à une autre entité.
7. Créer un programme de sensibilisation des entreprises sur le classement  
   contrôles.

# Gnu Privacy Guard (GPG)

GNU est un projet de système d'exploitation libre lancé en 1983 par Richard Stallman, puis maintenu par le projet GNU. Son nom est un acronyme récursif qui signifie en anglais « GNU’s Not UNIX » (littéralement, « GNU n’est pas UNIX »). Il reprend les concepts et le fonctionnement d’UNIX1.

Le système GNU permet l’utilisation de tous les logiciels libres, pas seulement ceux réalisés dans le cadre du projet GNU.

**Historique**

Le projet GNU est un projet informatique dont les premiers développements ont été réalisés en janvier 1984 par Richard Stallman pour développer le système d’exploitation GNU. Le projet est maintenu par une communauté de hackers organisée en sous-projets. Chaque brique du projet est un logiciel libre utilisable de par sa nature dans des projets tiers, mais dont la finalité est de s’inscrire dans une logique cohérente avec l’ensemble des sous-projets en vue de la réalisation d’un système d’exploitation complet et entièrement libre, et avec pour stratégie, l’utilisation de l’existant.

C’est ainsi que la première version fonctionnelle du système GNU est construite en 1992 avec l’utilisation du noyau Linux, un projet développé indépendamment du projet GNU par Linus Torvalds. Mais si la « rencontre GNU/Linux » permet l’assemblage d’un système libre, le développement d’un micro-noyau reste aujourd’hui l’un des objectifs techniques du projet.

Le projet est soutenu par la *Free Software Foundation*, en assurant notamment sa protection légale par la gestion de ses droits d'auteurs. Les objectifs et la philosophie du projet sont par ailleurs définis dans le manifeste GNU, lequel représente l’acte fondateur du mouvement du logiciel libre. Le projet GNU s’inscrit enfin dans une démarche sociale en plaçant les fondements philosophiques du mouvement devant les objectifs techniques du projet

**GNU/Linux**

|  |  |
| --- | --- |
| GNU and Tux.svg | GNU/Linux (souvent appelé Linux) est une variante du système d'exploitation GNU fonctionnant avec le noyau Linux. Le projet GNU avait originellement prévu le développement du noyau *Hurd* pour compléter le système, mais au début des années 1990, *Hurd* ne fonctionnait pas encore et son développement rencontrait encore des difficultés. L’arrivée du noyau Linux permit l’utilisation du système GNU sur les ordinateurs animés par des microprocesseurs de la famille Intel x86, en favorisant sa large diffusion par la complémentarité des projets. |
|  |  |

**Mouvement du logiciel libre**

Le mouvement du logiciel libre milite pour que les utilisateurs de l'informatique gagnent la liberté garantie par le logiciel libre. Ce dernier donne aux utilisateurs le contrôle de leur informatique. Le logiciel non libre, au contraire, place les utilisateurs sous le pouvoir de son développeur. Voir la vidéo d'explication.

Le terme « logiciel libre » signifie que les utilisateurs sont libres d'exécuter, de copier, de distribuer, d'étudier, de modifier et d'améliorer les logiciels.

Le logiciel libre est affaire de liberté, pas de prix. Pour comprendre ce concept, vous devez penser à « liberté d'expression », pas à « entrée libre ».

Plus exactement, cela signifie que l'utilisateur d'un programme libre jouit des quatre libertés essentielles :

* la liberté d'exécuter le programme comme il le souhaite, pour n'importe quel usage (liberté 0) ;
* la liberté d'étudier le fonctionnement du programme et de l'adapter à ses besoins (liberté 1) – une condition préalable est d'accéder au code source ;
* la liberté d'en redistribuer des copies pour aider les autres (liberté 2) ;
* la liberté d'améliorer le programme et de rendre publiques ses améliorations pour que toute la communauté en bénéficie (liberté 3) – une condition préalable est d'accéder au code source.

Les développements de l'usage de la technologie et du réseau ont rendu ces libertés encore plus importantes aujourd'hui qu'elles ne l'étaient en 1983.

Actuellement, le mouvement du logiciel libre va bien au-delà du développement du système GNU.

Il est possible d’avoir plus d’information sur le site de la **Free Software Foundation**.  
<http://www.gnu.org/licenses/licenses.fr.html>

# Gestion des accès aux fichiers

**Les types d'usagers**

Les permissions d'accès pour un fichier peuvent être positionnés par propriétaire, groupe, et pour les autres sur la base de permissions en lecture (r), écriture (w) et exécution (x). Les droits d'accès sont attribués aux fichiers ou aux répertoires.

Chaque usager fait partie d'un et d'un seul groupe. En retour, un même groupe peut contenir plusieurs usagers.

Par exemple, l'administrateur d'un système Linux peut avoir réparti ses usagers comme suit:

* les usagers etud00, etud01 et etud02 dans le groupe etudiants;
* les usagers bob et joe dans le groupe support;
* les usagers arthur et berta dans le groupe comptable.

**Remarques**

* le terme *user* désigne le propriétaire d’un fichier;
* Le terme *group* désigne tous les usagers du groupe dont fait partie le propriétaire;
* le terme *other* désigne tous les autres usagers, ceux qui font partie d'un groupe différent de celui du propriétaire.

**Affichage des permissions : ls –l**

Les droits sur un item sont affichées lorsqu'on utilise l’option -l.   
ls –l f1

On obtient alors comme résultat:  
-rwxr-xr-x 1 etud01(user) eda(group) 52 Apr 14 08:52 f1

La chaîne de caractères qui représente les droits  
(ici: *-rwxr-xr-x*) a la forme: *fuuugggooo*, où l'on a:

|  |  |
| --- | --- |
| f | un caractère identifiant certains types de fichiers, par exemple  - ordinaire d répertoire l lien symbolique c, b type de périphérique |
| uuu | droits pour le propriétaire (*user*) |
| ggg | droits pour le groupe (*group*) |
| ooo | droits pour les autres (*other*) |

**Droits concernant les fichiers**

Les droits pour chaque type d'usager sont exprimées par trois caractères:

|  |  |
| --- | --- |
| r | droit en lecture (*read*) donc permission d'afficher, de copier le fichier |
| w | droit en écriture (*write*) donc permission de modifier, de déplacer, de renommer, d'effacer le fichier, de le remplacer |
| x | droit d'exécuter |

L'ordre de ces caractéristiques est important, et l'absence d'un de ces accès est représenté par le tiret (–).

Par exemple, les droits du fichier test22 sont –rwx r-x --x et nous indiquent que:

* il s'agit d'un fichier ordinaire;
* le propriétaire (*user*) a les toutes les droits sur f1 (rwx);
* les membres du groupe (*group*) possèdent les droits de lecture et d'exécution sur f1 (r-x);
* les autres (*other*) autres ont le droitd'exécuter f1 seulement (--x).

**Droits concernant un répertoire**

Les répertoires possèdent eux aussi des droits. En général, la signification des différents types d'accès est:

|  |  |
| --- | --- |
| r | droit de visualiser le contenu du répertoire (avec ls) |
| w | droit de modifier le contenu du répertoire |
| x | droit de se déplacer dans le répertoire (avec cd) |

Pour simplifier, on peut dire que les restrictions sur les permissions d'un répertoire sont habituellement exprimées sous l'une de ces formes:

* Aucun accès (- - -) veut dire interdiction de voir, exécuter, modifier.   
  Retirer les accès sur un de vos répertoires permet du même coup de verrouiller tout ce qui se trouve immédiatement "dessous".
* Accès lecture/exécution (r – x) permet de voir les items que contient le répertoire et de s'y déplacer, mais empêche de modifier quoi que ce soit.

**Changement des droits**

La commande chmod permet de changer les droits d'un fichier ou d'un répertoire.   
Il y a deux formats possibles pour la commande:

* Forme numérique  
  chmod nombre fichier1 fichier2…
* Forme symbolique  
  chmod [u|g|o] [+|-|=] [r|w|x] fichier1 fichier2 …

Pour utiliser la forme numérique de la commande chmod, il faut spécifier un nombre de trois chiffres.   
Chacune des valeurs (en base 8) s'adresse respectivement au type *user*, *group* et *other*.   
Chaque valeur représente un ensemble d'accès que l'on désire donner au fichier (ou répertoire).

|  |  |
| --- | --- |
| **Accès** | **Valeur associée** |
| **- - -** | **0** |
| **- - x** | **1** |
| **- w -** | **2** |
| **- w x** | **3** |
| **r - -** | **4** |
| **r - x** | **5** |
| **r w -** | **6** |
| **r w x** | **7** |

Exemples:

* chmod 751 jonathan  
  donne les droits rwx r-x --x au fichier toto
* chmod 700 t\*  
  donne les droits rwx --- --- aux items débutant par t
* chmod 664 \tmp\\*  
  donne les droits rw- rw- r-- aux items dans \tmp
* chmod 400 denis julie  
  donne les droits r-- aux fichiers jean et paul

Notez qu’avec la forme numérique, on doit forcément spécifier les accès pour les trois types d’usagers.

La forme symbolique, quant à elle, permet de changer de façon indépendante les permissions accordées au propriétaire, au groupe et aux autres. Elle permet également de modifier certains types d'accès sans toucher aux autres. Exemples:

* chmod u+x jean  
  l'accès x est ajouté aux permissions courantes pour soi-même.  
  (Le reste est n’est pas touché);
* chmod g+w,o-w jean  
  l'accès en écriture est ajouté pour le groupe et retiré pour le type "other" (tout le reste est inchangé);
* chmod u=rx,g=x jean  
  les droits pour le propriétaire (user) seront r-x, et celles du groupe seront –x;  
  les droits en place pour les autres restent comme avant.
* chmod u+r,g-rw, o= jean  
  l'accès en lecture est ajouté aux droits en place pour le propriétaire;  
  les accès r et w sont retirés au groupe (l'accès x reste ce qu'il était);  
  toutes permissions retirées pour les autres.
* chmod u+x,u-r jean etienne  
  l'accès x est ajouté et l'accès r est retiré pour les fichiers jean et etienne.
* chmod g-r+wx jean  
  le droit r est retiré au groupe et les droits w et x sont ajoutés.

Notez que seul le propriétaire d'un item peut changer les droits qui lui sont associées.

**La commande *umask***

La commande *umask* est un outil de gestion des droits sur la création des fichiers et des répertoires. Il permet de gérer une certaines sécurité et une restriction des droits par défaut. En définissant un *umask*, on retire automatiquement l’*umask* des droits défini sur la création des futurs fichiers ou répertoires.

Il est possible de définir les droits associés aux fichiers lors de leur création.   
On retrouve la valeur de son *umask*, simplement en tapant la commande umask dans un terminal.   
Pour la modifier, il suffit d'exécuter cette commande :

umask 002 (par exemple)

Avec cet exemple de *umask* positionné à 002, les fichiers créés auront par défaut les droits rw-rw-r--, cela s'explique (un peu d'algèbre booléen) :

Quand on crée un fichier, les droits de celui-ci sont 666 (soit rw-rw-rw) + NON masque.  
Si on note ça en bit, cela fait :

*110 110 110 pour les rw-rw-rw-  
000 000 010 pour le masque de 2  
------------------  
110 110 100 pour 666 et Non masque*

Résultat :  
rw- rw- r--

Quand on crée un répertoire c'est le même principe mais les droits de celui-ci sont de 777 + NON masque

Ainsi, dans certains cas, il est plus intéressant de changer la valeur de *umask* que de faire des chmod à répétition ensuite.

**Changement de propriétaire**

Le propriétaire d'un item peut attribuer un nouveau propriétaire à un item avec la commande chown (change owner). Par exemple, si le fichier fich-jean appartient à l'usager etud00, ce dernier peut faire:

chown etud05 fich-jean

À partir de ce moment, l'usager etud05 devient le propriétaire du fichier. L'usager etud00 a dorénavant les permissions suivantes sur fich-jean:

* celles du groupe (*group*), s'il fait partie du même groupe que etud05
* celles des autres (*other*) sinon

Notez que le fichier reste où il était (par exemple dans le répertoire personnel de etud00).

Cette commande est irréversible, puisque seul le propriétaire peut la faire.

# Les listes de contrôle d'accès (ACLs) *Access Control Lists*

Les listes de contrôle d'accès (ACLs) vous permettent de fournir différents niveaux d'accès aux dossiers et aux fichiers pour différents utilisateurs. Un des dangers que les acls essayent d'éviter, est de permettre aux utilisateurs de créer des dossiers avec la permission 777.   
Ceci touche directement la sécurité du système. (Ubuntu n'installe pas des acls par défaut, vous devez les installer et configurer le système de fichiers de sorte que les acls soient en activées).   
Les ACLs sont un outil important que les administrateurs doivent comprendre et montrer aux utilisateurs comment s’en servir de manière sécuritaire.

**Les Commandes acl**

Le processus pour changer des acls est assez simple mais parfois les implications sont beaucoup plus complexes. Il y a quelques commandes qui vous aideront à faire les changements pour des acls sur différents fichiers et dossiers personnels.

getfacl file or directory

Cette commande énumérera toutes les acls appliquées au fichier ou dossier. Par exemple si un utilisateur (tom) crée un dossier et donne des droits d'ACL à un autre utilisateur le résultat ressemblera à:

getfacl myfile  
*# file: myfile  
# owner: tom  
# group: tom  
user::rw-  
user:sue:rwx  
group::rw-  
mask::rwx   
other::r--*

*getfacl* affiche le propriétaire du fichier ou dossier ainsi que les utilisateurs qui ont été ajoutés avec des acls comme sue dans l'exemple. Il fournit également les droits pour un utilisateur. Dans l'exemple sue a les droits rwx sur le fichier *myfile*.

La commande *setfacl* est utilisée pour créer ou modifier les droits acl. Par exemple, si vous vouliez changer l'ACL pour mike sur un fichier vous devriez utiliser la commande:

setfacl -m u:mike:rwx file or directory

L’option -m permet de modifier l’acl et le "u" est pour le user qui est spécifiquement nommé, "mike", suivi par les droits et le fichier ou le répertoire. Si on remplace "u" par "g" on changera alors les acls du groupe.

setfacl -m g:sales:rw file or directory

Si vous voulez configurer un répertoire de sorte que tous les fichiers qui sont créés hériteront des acls du dossier, vous devriez utiliser l'option « d » avant user ou group.

setfacl -m d:u:mike:rw directory

Pour supprimer les droits ACL, on utilise l’option "x".

setfacl -x u:mike file or directory

**Laboratoire**

Ce laboratoire vous aidera à apprendre comment gérer les ACL.

Création d’un groupe nommé comptabilité.   
groupadd comptabilité

* Créez 3 usagers, joe, jim et jane et rendez les membres du groupe comptabilité.
* Éditez le fichier */etc/group* et vérifiez qu’ils sont bien placés dans le groupe comptabilité, séparés par des virgules.
* Créez le répertoire */home/comptabilité*, donnez au groupe comptabilité l’accès au répertoire /*home/comptabilité* et contrôle total à joe.  
  chown root:accounting /home/comptabilité  
  setfacl –m u:joe:rwx /home/comptabilité

Se connecter en tant que Joe pour créer les dossiers, les fichiers et donner les permissions

mkdir /home/comptabilité/ventes  
mkdir /home/comptabilité/recherche  
touch /home/comptabilité/ventes/raports\_ventes  
touch /home/comptabilité/ventes/retours

Ajouter du texte aux 2 fichiers et configurer ces acls comme ci-dessous.

setfacl -m u:jim:rwx /home/comptabilité/ventes/raports\_ventes  
setfacl -m u:jane:r /home/comptabilité/ventes/retours  
setfacl -m g:comptabilité:r /home/comptabilité/ventes

**Test des accès**

Se connecter comme jim et jane et essayer de lire et écrire sur les fichiers créés.

**Remarque**  
Avec la configuration ci-dessus, Jim et Jane ne peuvent pas accéder aux fichiers à cause de la restriction de leur groupe COMPTABILITÉ qui a seulement LECTURE.   
Il faut ajouter au groupe conptabilité l’autorisation x (exécuter) pour permettre aux membres du groupe de se déplacer dans le répertoire ventes.

setfacl –m g:comptabilité:rx /home/comptabilité/ventes

Avec cette configuration :

* Jim peut LIRE et ÉCRIRE sur rapports\_ventes et seulement LIRE sur retours.
* Jane peut seulement LIRE les 2 fichiers.

# Chiffrement et certificats

La cryptographie est une des disciplines de la cryptologie s'attachant à protéger des messages (assurant confidentialité, authenticité et intégrité) en s'aidant souvent de secrets ou clés. Elle se distingue de la stéganographie qui fait passer inaperçu un message dans un autre message alors que la cryptographie rend un message inintelligible à autre que qui-de-droit.

**Le chiffrement symétrique**

Le chiffrement symétrique (aussi appelé chiffrement à clé privée ou chiffrement à clé secrète) consiste à utiliser la même clef pour le chiffrement que pour le déchiffrement.

Le chiffrement consiste alors à effectuer une opération entre la clé privée et les données à chiffrer afin de rendre ces dernières inintelligibles. Ainsi, le moindre algorithme (tel qu'un OU exclusif) peut rendre le système quasiment inviolable (la sécurité absolue n'existant pas).

Toutefois, dans les années 40, Claude Shannon démontra que pour être totalement sûr, les systèmes à clefs privées doivent utiliser des clefs d'une longueur au moins égale à celle du message à chiffrer. De plus le chiffrement symétrique impose d'avoir un canal sécurisé pour l'échange de la clé, ce qui dégrade sérieusement l'intérêt d'un tel système de chiffrement.

Ainsi, dans les années 20, Gilbert Vernam et Joseph Mauborgne mirent au point la méthode du one time pad (traduisez méthode du masque jetable), basée sur une clé privé générée aléatoirement, utilisée une et une seule fois, puis détruite. Ainsi à la même époque le Kremlin et la maison blanche étaient reliés par le fameux téléphone rouge, c'est-à-dire un téléphone dont les communications étaient cryptées par une clé privée selon la méthode du masque jetable. La clé privée était alors échangée grâce à la valise diplomatique (jouant le rôle de canal sécurisé).

Quelques algorithmes de chiffrement symétrique très utilisés :

* Chiffre de Vernam (le seul offrant une sécurité théorique absolue, à condition que la clé ait au moins la même longueur que le message, qu'elle ne soit utilisée qu'une seule fois à chiffrer et qu'elle soit totalement aléatoire);
* DES;
* 3DES;
* AES;
* RC4;
* RC5;
* MISTY1.

**Le chiffrement asymétrique**

Le principal inconvénient d'un cryptosystème à clefs secrètes provient de l'échange des clés. En effet, le chiffrement symétrique repose sur l'échange d'un secret (les clés). Ainsi, se pose le problème de la distribution des clés: Pour un groupe de n personnes utilisant un cryptosystème à clés secrètes, il est nécessaire de distribuer n x (n-1) / 2 clés. Le principe du chiffrement à clé publique Le paradigme de chiffrement à clés publiques est apparu en 1976, avec la publication d'un ouvrage sur la cryptographie par Whitfield Diffie et Martin Hellman.

Dans un cryptosystème asymétrique (aussi appelé cryptosystème à clés publiques), les clés existent par paires (on parle souvent de bi-clés):

* Une clé publique pour le chiffrement ;
* Une clé secrète pour le déchiffrement.

Ainsi, dans un système de chiffrement à clé publique, les utilisateurs choisissent une clé aléatoire dont ils sont seuls connaisseurs (il s'agit de la clé privée). À partir de cette clé, ils déduisent chacun automatiquement un algorithme (il s'agit de la clé publique). Les utilisateurs s'échangent cette clé publique au travers d'un canal non sécurisé.

Lorsqu'un utilisateur désire envoyer un message à un autre utilisateur, il lui suffit de chiffrer le message à envoyer au moyen de la clé publique du destinataire (qu'il trouvera par exemple dans un serveur de clés tel qu'un annuaire). Ce dernier sera en mesure de déchiffrer le message à l'aide de sa clé privée (qu'il est seul à connaître).

Ce système est basé sur une fonction facile à calculer dans un sens (appelée fonction à trappe à sens unique ou en anglais one-way trapdoor function), mais qui est mathématiquement très difficile à inverser sans la clé privée (appelée trappe).

Pour faire une image avec le monde réel, il s'agit pour un utilisateur de créer une clé aléatoire (la clé privée), puis de fabriquer un grand nombre de cadenas (clé publique) qu'il dispose dans un casier accessible par tous (le casier joue le rôle de canal non sécurisé). Pour lui faire parvenir un document, chaque utilisateur peut prendre un cadenas (ouvert), fermer une valisette contenant le document grâce à ce cadenas, puis envoyer la valisette au propriétaire de la clé publique (le cadenas). Seul le propriétaire sera alors en mesure d'ouvrir la valisette avec sa clé privée.

Le problème consistant à se communiquer la clé de déchiffrement n'existe plus, les clés publiques pouvant être envoyées librement. Le chiffrement par clés publiques permet donc à des personnes de s'échanger des messages cryptés sans pour autant posséder de secret en commun. En contrepartie tout le challenge consiste à s'assurer que la clé publique que l'on récupère est bien celle de la personne à qui l'on souhaite faire parvenir l'information chiffré ! 'intercepteur" ou "attaque par le milieu".

**Intérêt d'une clé de session**

Les algorithmes asymétriques (entrant en jeu dans les cryptosystèmes à clé publique) permettent de s'affranchir de problèmes liés à l'échange de clé via un canal sécurisé. Toutefois, ces derniers restent beaucoup moins efficaces (en terme de temps de calcul) que les algorithmes symétriques. Ainsi, la notion de clé de session est un compromis entre le chiffrement symétrique et asymétrique permettant de combiner les deux techniques.

Le principe de la clé de session est simple : il consiste à générer aléatoirement une clé de session de taille raisonnable, et de chiffrer celle-ci à l'aide d'un algorithme de chiffrement à clef publique (plus exactement à l'aide de la clé publique du destinataire). Le destinataire est en mesure de déchiffrer la clé de session à l'aide de sa clé privée. Ainsi, expéditeur et destinataires sont en possession d'une clé commune dont ils sont seuls connaisseurs. Il leur est alors possible de s'envoyer des documents chiffrés à l'aide d'un algorithme de chiffrement symétrique.

**Algorithme de Diffie-Hellman**

L'algorithme de Diffie-Hellman (du nom de ses inventeurs Diffie et Hellman) a été mis au point en 1976 afin de permettre l'échange de clés à travers un canal non sécurisé. Il repose sur la difficulté du calcul du logarithme discret dans un corps fini.

**Attaque Man-in-the-middle**

L'algorithme de Diffie-Hellman est cependant sensible à l'attaque Man in the middle (traduit parfois en attaque de l'intercepteur ou attaque par le milieu).

Deux algorithmes de cryptographie asymétrique très utilisés :

* RSA (chiffrement et signature);
* DSA (signature).

**Signature électronique**

Le paradigme de signature électronique (appelé aussi signature numérique) est un procédé permettant de garantir l'authenticité de l'expéditeur (fonction d'authentification), ainsi que de vérifier l'intégrité du message reçu. La signature électronique assure également une fonction de non-répudiation, c'est-à-dire qu'elle permet d'assurer que l'expéditeur a bien envoyé le message (autrement dit elle empêche l'expéditeur de nier avoir expédié le message).

Qu'est-ce qu'une fonction de hachage ?

Une fonction de hachage (parfois appelée fonction de condensation) est une fonction permettant d'obtenir un condensé (appelé aussi haché) d'un texte, c'est-à-dire une suite de caractères assez courte représentant le texte qu'il condense. La fonction de hachage doit être telle qu'elle associe un et un seul haché à un texte en clair (cela signifie que la moindre modification du document entraîne la modification de son haché). D'autre part, il doit s'agir d'une fonction à sens unique (one-way function) afin qu'il soit impossible de retrouver le message original à partir du condensé.

Ainsi, le haché représente en quelque sorte l'empreinte digitale (en anglais finger print) du document.

Les algorithmes de hachage les plus utilisés actuellement sont :

* **MD5 (MD signifiant *Message Digest*)**, créant une empreinte digitale de 128 bits.  
  Il est courant de voir des documents en téléchargement sur Internet accompagnés d'un fichier MD5, il s'agit du condensé du document permettant de vérifier l'intégrité de ce dernier)
* **SHA (pour *Secure Hash Algorithm*)**, pouvant être traduit par Algorithme de hachage sécurisé) créant des empreintes d'une longueur de 160 bits.  
  Lors de la réception du message, il suffit au destinataire de calculer le haché du message reçu et de le comparer avec le haché accompagnant le document. Si le message (ou le haché) a été falsifié durant la communication, les deux empreintes ne correspondront pas.

**Le scellement des données**

L'utilisation d'une fonction de hachage permet de vérifier que l'empreinte correspond bien au message reçu, mais rien ne prouve que le message a bien été envoyé par celui que l'on croit être l'expéditeur.

A réception du message, il suffit au destinataire de déchiffrer le sceau avec la clé publique de l'expéditeur, puis de comparer le haché obtenu avec la fonction de hachage au haché reçu en pièce jointe. Ce mécanisme de création de sceau est appelé scellement.  
Ainsi, pour garantir l'authentification du message, il suffit à l'expéditeur de chiffrer (on dit généralement signer) le condensé à l'aide de sa clé privée (le haché signé est appelé sceau) et d'envoyer le sceau au destinataire.

# IPsec

**Concepts généraux**

Le développement des réseaux, qu’ils soient internes ou externes, accroît la demande en sécurité. Les données transitant sur un réseau doivent être sécurisées et protégées contre toutes attaques extérieures, écoutes réseaux, accès non autorisés, interceptions de paquets IP, etc. Afin de remédier à ce problème de sécurité l’IETF (*Internet* *Engineering Task Force*) a rédigé la RFC 2401 abordant le problème de la sécurité sur un réseau IP en 1998.

La confidentialité, l'intégrité, l'authentification des données entre deux hôtes est gérée par un ensemble de normes et protocoles. Le protocole IPSec rassemble toutes ces normes ouvertes.

Alice émet des données qu’elle va chiffrer avant de les faire transiter sur le réseau à destination de Bob; Bob  décode les données après réception de celles-ci.

IPSec est un standard permettant de réaliser la sécurité des communications au niveau de la couche réseau du modèle OSI.

Ce modèle est fondamentalement différent des systèmes ou la sécurité des données se réalise au niveau de la couche application comme dans les modèles SSL ou TLS par exemple. SSL et TLS permettent de réaliser une sécurité au niveau de la couche transport. L’application cliente ou serveur doit avoir été conçu pour. L’avantage d’IPSec est que n’importe quelle application (même non écrite pour) pourra utiliser de manière transparente les fonctionnalités de sécurité fournies par l’hôte sur lequel il réside. La mise en tunnel (*tunneling*) IPSec peut être employé pour protéger un ou plusieurs flux de données entre hôtes internes, accès distant ou tout autres connexions devant être sécurisée.

Le modèle IPSec est déjà présent dans la plupart des piles IPv4. Il sera totalement intégré à IPV6. Le déploiement d’un tel système est aisé car il est transparent pour les applications déjà existantes.

**La mise en tunnel IPSec**

La mise en tunnel IPSec peut être employée pour protéger un ou plusieurs flux de données entre:

* des hôtes internes,
* un accès distant   
    
  ou
* toutes autres connexions devant être sécurisée.

Lorsqu’un paquet de données est envoyé sur le réseau, la mise en tunnel l’encapsule dans un autre paquet. De cette manière le paquet est sécurisé contre toutes attaques extérieures (écoutes, rejeu, …).

**Les modes IPSec**

Une communication entre deux hôtes protégée par IPSec est susceptible de fonctionner suivant deux modes différents:

* le mode transport  
  Communication IPSec bout en bout  
  et  
  le mode tunnel  
  Communication IPSec au moyen d’une passerelle de sécurité.

Le mode transport offre essentiellement une protection aux protocoles de niveau supérieur. Le mode transport est utilisé par les stations et par les serveurs pour sécuriser les flux de bout en bout. Dans le mode transport, l’entête du paquet IP sécurisé n’est pas modifié et est expédié en clair sur le réseau.

Le mode tunnel permet quant à lui d'encapsuler des datagrammes IP dans d'autres datagrammes IP dont le contenu est protégé. L'intérêt majeur du mode tunnel est qu'il rend la mise en place de passerelles de sécurité qui:

* traitent toute la partie IPSec d'une communication  
  **et**
* transmettent les datagrammes épurés de leur partie IPSec à leur destinataire réel.

Le mode tunnel, surtout utilisé par les équipements de base de réseau, va permettre de faire du *tunneling*. Le paquet d’origine est complètement encapsulé dans un nouveau paquet IP. Les données du paquet IPSec sont en fait le paquet d’origine en entier et non pas les données du paquet d’origine. On est réellement dans un mode tunnel où tout le paquet IP de départ est compris dans la sécurité (AH ou ESP).

Il est également possible d'encapsuler une communication IPsec en mode tunnel ou transport dans une autre communication IPsec en mode tunnel, elle-même traitée par une passerelle de sécurité, qui transmet les datagrammes après suppression de leur première enveloppe à un hôte traitant à son tour les protections restantes ou à une seconde passerelle de sécurité

**Les protocoles**

IPSec fournit deux protocoles de base:

* AH (Authentication Header)  
  et
* ESP (Encapsulation Security Paylod)

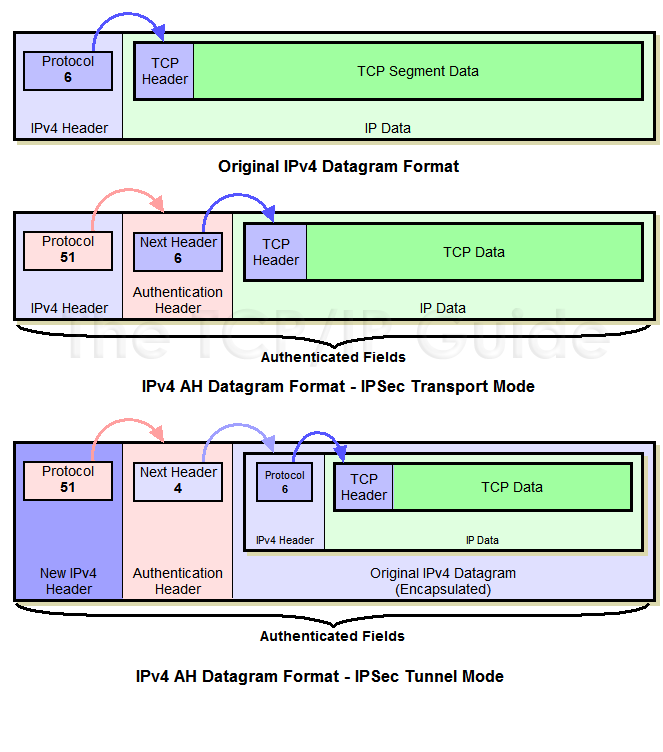
**AH (Authentication Header)**

Ce mode permet uniquement de réaliser l’authentification et la vérification de l’intégrité des paquets. Aucune confidentialité dans les données n’est assurée. Les spécifications de ce mode sont indiqués dans la RFC 2402.

**Format du paquet AH**

Le format du paquet IP est le suivant. Le paquet est composé de trois parties distinctes:

* l’entête traditionnel qui permet d’aiguiller le paquet.
* le AH est en fait le résultat du hachage des informations suivantes:  
  entête IP,   
  autres informations d’entêtes ainsi qu’une clé secrète utile au hachage
* les données d’origine du paquet IP.



Les services fournis par AH sont les suivants :

* Authentification de l’émetteur des paquets.
* Intégrité des données
* Détection d’intrusion dans un flux

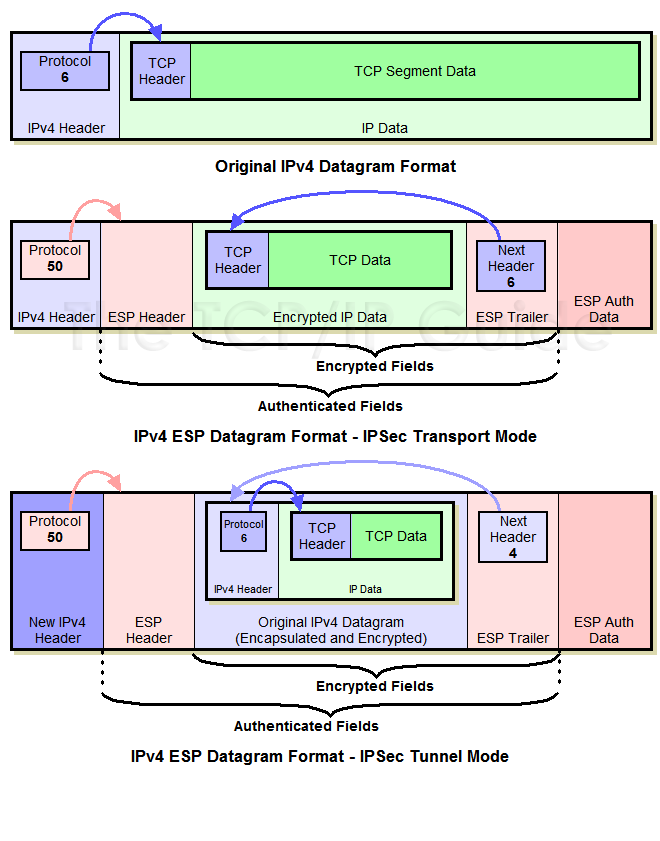
Le mode AH est moins consommateur de ressources machine que le mode ESP dont le rôle est de crypter les données.

**ESP (Encapsulation Security Paylod)**

Ce mode permet d’obtenir la confidentialité des données, l’authentification et l’intégrité des données. Ce protocole est défini dans la RFC 2406.

Les fonctionnalités d’ESP sont la confidentialité des données en cryptant les données avec un algorithme symétrique.  
Il est également possible en option de rajouter l’authentification et la vérification d’intégrité des données comme en AH avec *Keyed MD5* ou *Keyed SHA-1*.

ESP peut fonctionner également dans les deux modes IPSec:   
Mode Transport ou Mode Tunnel.



**Architecture de sécurité de sécurité**

**Associations de sécurité**

La RFC 2401 Security Architecture for the Internet Protocol) décrit le protocole IPSec au niveau le plus élevé. En particulier, elle indique ce qu'une implémentation est censée permettre de configurer en termes de politique de sécurité – c'est-à-dire quels échanges IP doivent être protégés par IPSec et, le cas échéant, quel(s) protocole(s) utiliser.

Sur chaque système capable d'utiliser IPSec doit être présente une base de données de politiques de sécurité (SPD ou Security Policy Database) dont la forme précise est laissée au choix de l'implémentation et qui permet de préciser la politique de sécurité à appliquer au système. Chaque entrée de cette base de données est identifiée par un index des paramètres de sécurité (SPI ou Security Parameters Index) unique de 32 bits choisi arbitrairement.

Une communication protégée à l'aide d'IPSec est appelée une association de sécurité (SA ou Security Association). Une association de sécurité repose :

* sur une unique application de AH  
  ou
* sur une unique application de ESP.

Ceci n'exclut pas l'usage simultané de AH et ESP entre deux systèmes ou par exemple l'encapsulation des datagrammes AH dans d'autres datagrammes AH mais plusieurs AS devront alors être activées entre les deux systèmes. En outre, une AS est unidirectionnelle. La protection d'une communication ayant lieu dans les deux sens nécessitera donc l'activation de deux AS. Chaque AS est identifiée de manière unique par un SPI, une adresse IP de destination (éventuellement adresse de diffusion ou de multi-diffusion) et un protocole (AH ou ESP). Les AS actives sont regroupées dans une SAD (Security Association Database).

La SPD est consultée pendant le traitement de tout datagramme IP, entrant ou sortant, y compris les datagrammes non-IPSec. Pour chaque datagramme trois comportements sont envisageables :

* le rejeter,
* l'accepter sans traitement IPSec  
  ou
* appliquer IPSec.

Dans le troisième cas (application d’IPSec), la SPD précise en outre quels traitements IPSec appliquer:

* ESP ou AH;
* mode tunnel ou transport;
* quel(s) algorithme(s) de signature et/ou chiffrement utiliser.

Les plus importants de ces termes sont récapitulés dans le tableau suivant:

* SPD  
  base de données définissant la politique de sécurité
* SA  
  une entrée de la SPD
* SAD  
  liste des SA en cours d'utilisation

Les règles de la SPD doivent pouvoir, si l'adminsitrateur du système le souhaite, dépendre des paramètres suivants:

* adresse ou groupe d'adresses IP de destination;
* adresse ou groupe d'adresses IP source;
* nom du système  
  (DNS complète, nom X.500 distingué ou général);
* protocole de transport utilisé  
  (typiquement, TCP ou UDP);
* nom d'utilisateur complet, comme [lsavard@cmaisonneuve.qc.ca](mailto:lsavard@cmaisonneuve.qc.ca)  
  (ce paramètre n'est toutefois pas obligatoire sur certains types d'implémentations)
* ports source et destination  
  (UDP et TCP seulement, le support de ce paramètre est facultatif)

Cependant, une politique commune pour le trafic vers un ensemble de systèmes permet une meilleure protection contre l'analyse de trafic qu'une politique extrêmement détaillée, comprenant de nombreux paramètres propres à chaque système particulier. Enfin, si l'implémentation permet aux applications de préciser elles-mêmes à quelle partie du trafic appliquer IPSec et comment, l'administrateur doit pouvoir les empêcher de contourner la politique par défaut.

# Authentification

L'authentification est la procédure qui consiste, pour un système informatique, à vérifier l'identité d'une personne ou d'un ordinateur afin d'autoriser l'accès de cette entité à des ressources (systèmes, réseaux, applications…). L'authentification permet donc de valider l'authenticité de l'entité en question.

L'identification permet donc de connaître l'identité d'une entité alors que l'authentification permet de vérifier cette identité.

**Définition**

L'authentification désigne le processus visant à confirmer qu'un commettant est bien celui qu'il prétend être. Il existe quatre facteurs d'authentification classiques qui peuvent être utilisés pour confirmer l'identité d'un commettant:

* Utiliser une information que seul le commettant connaît.
* Utiliser une information unique que seul le commettant possède;
* Utiliser une information qui caractérise le commettant dans un contexte donné;
* Utiliser une information que seul le commettant peut produire.

D'autres facteurs d'authentification peuvent parfois être utilisés comme les contraintes temporelles ou les capacités de localisation.

**Enjeu**

Le contrôle permanent de l'intégrité et de l'accès (usage, identité du destinataire, émetteur, propriétaire) à un contenu ou à un service constitue le fondement de la traçabilité des transactions.

Ce contrôle permet:

* la protection des intérêts supérieurs de l'État et du patrimoine informatique des entreprises, donc de leurs intérêts commerciaux.   
  Pour les entreprises, il s'agit de réduire le coût qui résulte d'attaques, de la perte de temps, de la perte d'informations, de l'espionnage, ou des fuites involontaires d'informations...
* le développement du commerce et des échanges électroniques. L  
  'authentification contribue à la facturation des services et contribue à la confiance dans l'économie numérique, condition indispensable du développement économique.
* la protection de la vie privée.   
  Les données personnelles véhiculées dans les systèmes d'information sont des données sensibles à protéger.

**Preuves**

Dans le cas d'un individu, l'authentification consiste, en général, à vérifier que celui-ci possède une preuve de son identité ou de son statut, sous l'une des formes (éventuellement combinées) suivantes:

* **Ce qu'il sait**   
  (mot de passe, numéro d'identification personnel).
* **Ce qu'il possède**  
  (acte de naissance, carte grise, carte d'identité, carte à puce, droit de propriété, certificat électronique, diplôme, passeport, carte Vitale, Token OTP, Carte OTP, Téléphone portable, PDA, etc.).
* **Ce qu'il est**   
  (photo, caractéristique physique, voire biométrie).
* **Ce qu'il sait faire**   
  (geste, signature).

**Méthodes de vérification**

La phase de vérification fait intervenir un protocole d'authentification. On en distingue trois sortes « familles »:

* **L'authentification simple**  
  l'authentification ne repose que sur un seul élément ou « facteur »   
  (exemple : l'utilisateur indique son mot de passe).
* **L'authentification forte**  
  l'authentification repose sur deux facteurs ou plus.  
  **L'Authentification unique**   
  (ou identification unique - en anglais *Single Sign-On* ou *SSO*)  
  méthode permettant à un utilisateur de ne procéder qu'à une seule authentification pour accéder à plusieurs applications informatiques (ou sites internet sécurisés).

Par ailleurs, l'authentification à transfert nul de connaissance ne considère que l'information de la véracité d'une proposition.

Authentification renforcée basée sur une *cryptocard*



**Facteur d'authentification**

Le facteur d'authentification est un facteur physique, cognitif ou biologique produisant une empreinte qu'un utilisateur peut prendre pour être authentifié par un système informatique. L'empreinte doit être personnelle à l'utilisateur et doit pouvoir être numérisée.

**Types de facteurs**

Quatre types de facteurs répondent à ces prérequis:

* **Facteur mémoriel** (ce qu'il sait)  
  Empreinte  
  une information qu'il a mémorisé.  
  Exemples  
  le nom de sa mère ou un mot de passe.
* **Facteur matériel** (ce qu'il possède)  
  Empreinte  
  une information contenue dans un objet qu'il utilise.  
  Exemples  
  une clé USB, un identifiant sur bande magnétique, un certificat numérique sur une carte à puce.
* **Facteur corporel** (ce qu'il montre)  
  Empreinte  
  une trace corporelle qu'il peut laisser quelque part.  
  Exemples  
  une empreinte digitale, les caractéristiques de sa pupille, sa voix.
* **Facteur réactionnel** (ce qu'il fait)  
  Empreinte  
  un geste qu'il peut reproduire.  
  Exemples  
  sa signature.

Le type de facteur le plus couramment utilisé sur Internet est le facteur mémoriel. La montée du phishing pousse les applications en ligne à surmonter l'authentification simple classique avec un deuxième facteur d'authentification (authentification forte).

# Mots de passe

**Notions de base**

Lorsqu’un utilisateur veut accéder à un système d'information, il doit habituellement dans un premier temps effectuer une procédure d'identification et d'authentification.

L'identification est une phase qui consiste à établir l'identité de l'utilisateur. Elle permet répondre à la question: « Qui êtes-vous? ». L'utilisateur utilise un identifiant (que l'on nomme compte d'accès, nom d'utilisateur ou *Login* en anglais) qui l'identifie et qui lui est attribué individuellement. Cet identifiant est unique.

L'authentification est une phase qui permet à l'utilisateur d'apporter la preuve de son identité. Elle intervient après la phase dite d'identification. Elle permet de répondre à la question: « Êtes-vous réellement cette personne? ».   
L'utilisateur utilise un authentifiant ou code secret que lui seul connait. Le code secret d'un utilisateur est une information personnelle qui ne doit en aucun cas être divulgués. Il est aussi communément nommé mot de passe.

Le mot de passe ne permet pas de donner un droit d'accès, il permet uniquement d'assurer l'imputabilité dans l'usage de ces droits d'accès. Lorsque deux personnes ou plus connaissent le mot de passe correspondant à une identité d'utilisateur, il s'agit d'une infraction à la sécurité (sauf en cas de disposition spécifique pour assurer la continuité d'un service). Cette disposition est alors clairement définie dans la charte d'usage des personnels.

**Les risques liés au mot de passe**

Les principaux risques liés au mot de passe sont sa divulgation et sa faiblesse.

* La divulgation d'un mot de passe est causée soit par négligence des règles (prêt de ses identifiants et authentifiants à un collègue ou à sa hiérarchie), soit par un acte de malveillance (craquage de mot de passe, cheval de Troie, hameçonnage).   
  Dans les deux cas, la responsabilité de l'utilisateur, propriétaire de l'identité usurpée, peut se voir engagée.
* La faiblesse d'un mot de passe constitue une faille sérieuse. Les techniques utilisées pour découvrir les mots de passe (craquage, ingénierie sociale) sont toujours efficaces lorsque ces derniers sont trop simplistes ou lorsqu'ils se rapportent à des éléments de la vie privée ou publique de la personne (prénom d'un enfant, marque de sa voiture, etc.).

**La politique des mots de passe dans une entreprise**

La sécurité de l'accès s'appuie sur la robustesse du mot de passe et sur notre capacité à le garder secret. Tous les utilisateurs doivent pour cela respecter un certain nomre de consignes.   
De plus l’administrateur réseau doit également imposer des règles à suivre.

La politique de mots de passe doit comporter:

* une définition de la robustesse (les critères qui suivent sont des exemples);
* une norme d’unicité;
* une durée minimale et maximale de vie;
* une mécanique de blocage en cas d’usage abusif;
* …

Le mot de passe doit aussi

* rester confidentiel et ne pas être accessible à proximité du poste de travail (post-it par exemple);
* être saisi par l'utilisateur lors de l'authentification et ne pas être enregistré au préalable   
  (comme cela est parfois proposé par les fureteur internet).

**Consignes aux utilisateurs**

L'utilisateur doit:

* modifier son mot de passe en cas de doute sur sa confidentialité   
  (après une attaque de type "Phishing" par exemple);
* signaler obligatoirement toutes difficultés ou risques éventuels liés à ces mots de passe à l'administrateur réseau;
* ne jamais communiquer ses mots de passe à quiconque - ni aux administrateurs ni aux supérieurs;
* chercher à exploiter le même mot de passe par domaine restreint (par exemple, éviter d'utiliser le même mot de passe pour se connecter à un forum quelconque ou à sa messagerie personnelle que celui utilisé pour accéder à la messagerie professionnel.

**Qu'est-ce qu'un bon mot de passe?**

Un mot de passe doit faire au moins 8 caractères, composé de lettres minuscules et majuscules, de chiffres et de caractères spéciaux. Il faut éviter les mots du dictionnaire, des noms propres (prénom, nom de famille, etc.).

**Exemples de robustesse**

* Variété de caractères  
  Majuscules: ABCDEFGHIJKLMNOPQRSTUVWXYZ  
  Minuscules: abcdefghijklmnopqrstuvwxyz  
  Chiffres: 0123456789  
  Spéciaux: ;:-\_=\|//?^&!.@$£#\*()%~<>{}[]
* Critères  
  minuscules : 2 au moins   
  majuscules : 2 au moins  
  chiffres : 2 au moins  
  caracteres speciaux AUTORISES : 2 au moins  
  longueur (en caracteres) : 8 au moins

# Contrôle d’accès

**Introduction**

Le principal objectif est de contrôler comment les usagers et les processus communiquent et interagissent avec les autres processus et les diverses ressources disponibles afin de protéger les propriétés des actifs.

* Confidentialité
* Intégrité
* Disponibilité

**Terminologie**

Le contrôle d’accès fait référence à deux concepts bien définis:

* Le sujet  
  Entité désirant accéder à un processus ou une ressource;
* L’objet  
  Entité passive à laquelle le sujet cherche à accéder.

**Concepts de base**

**Identification**Objectif  
Identifier de façon unique un sujet grâce à un identifiant.  
Fonctionnement:  
Chaque identifiant devrait être unique afin de s’assurer de l’imputabilité des usagers. Une méthode standardisée devrait être utilisée.Idéalement, l’identifiant ne devrait pas décrire la position du sujet. Cette information est unique et non secrète. Elle sert de référence dans la base des utilisateurs.

**Authentification**Objectif  
S’assurer que l’identité du sujet est bien celle qu’il prétend être.  
Fonctionnement:  
L'authentification consiste à l’identification de l’utilisateur, que ce soit une personne physique ou un service. Cette identification passe par la présentation de l’identité de l’utilisateur. Le contrôle de cette information consiste à vérifier un secret partagé entre l’utilisateur et le serveur d’authentification.

L’authentification repose sur un authentifiant (*credential*) que le sujet détient.

* Quelque chose que le sujet connaît (*something you know*)  
  Mot de passe, numéro d’identification personnel, …
* Quelque chose que le sujet possède (*something you have*)  
  Carte magnétique, carte à puce, …
* Quelque chose déterminant le sujet (*something you are*)  
  Biométrie: empreinte digitale, rétine, iris, …
* Quelque chose que le sujet sait faire (*something you can do*)  
  L’authentification peut donc être de plusieurs types :

L’authentification peut être…

* Statique  
  L’information transmise est alors la même lors d’authentifications successives.  
  Exemple: mot de passe type UNIX
* Dynamique  
  On passe alors par un challenge entre le serveur et l’utilisateur, ce qui permet d’avoir une information différente à chaque nouvelle authentification.  
  Exemples: calculette, carte à puce, badge...
* Physique  
  Elle est basée sur une ou plusieurs caractéristiques physiques d'un individu telles la reconnaissance vocale, les empreintes digitales, l'iris...

**Autorisation**Objectif  
Déterminer si le sujet authentifié peut poser l’action désirée sur l’objet spécifié.  
Fonctionnalités  
Une autorisation est le fait de déterminer quels sont les droits et/ou permissions d’un utilisateur.

Par exemple, après avoir ouvert une session, un usager peut essayer d’utiliser certaines commandes ou avoir un accès à un fichier. L’autorisation détermine alors si l’utilisateur peut ou non les utiliser. Dans certaines implémentations, l’identification et l’autorisation sont regroupés en une seule étape.

La politique de sécurité détermine les critères selon lesquels un sujet est autorisé à accéder à un objet:

* Rôles;
* Groupes;
* Localisation (physique ou logique);
* Heure de la journée, journée de la semaine;
* Type de transaction;
* …

Par défaut, l’accès à un objet devrait être refusé.

Deux concepts importants…

* Séparation des responsabilités (separation of duties)  
  Principe selon lequel au moins deux personnes sont requises afin de réaliser une tâche importante.
* Droit d’accès minimal (least privilege ou need-to-know basis)   
  Principe selon lequel une personne (ou un processus) qui a un tâche à effectuer possède les ressources et les privilèges minimaux pour la réaliser.

**Imputabilité (*Accounting*) et traçabilité**Objectif  
Attribuer un accès à un objet ou une opération à un sujet donné afin de s’assurer la traçabilité de toute violation ou tentative de violation d’une règle de sécurité.  
Fonctionnalités  
L'imputabilité consiste à enregistrer l'usage des droits des utilisateurs, les tentatives infructueuses et les accès illicites, afin de pouvoir attribuer les responsabilités dans le cadre de la non répudiation, ou détecter les anomalies et y remédier dans les meilleurs délais.

Au fur et à mesure que les systèmes d'information des entreprises se sont ouverts vers l'extérieur, les besoins de traçabilité ont augmenté. Pendant que les besoins d'accès des utilisateurs à leurs données et applications croissaient, la nécessité de contrôler, vérifier et imputer ces accès avec acuité s'est fait ressentir au niveau des organes de sécurité des entreprises.

Enfin, cette fonctionnalité consiste à mesurer les ressources qu’un utilisateur consomme, en terme d’échange réseau, de ressources système,... Cela sert en fait à journaliser un certain nombre d’informations sur l’utilisateur. Cela permet de connaître à la fois les services demandés par un utilisateur et la quantité de ressources requises.

**Protocoles d'authentification**

Un protocole d’authentification est un moyen de contrôle d’accès. Il comprend les trois A (AAA) :

* Authentification (Authentication)
* Autorisation (Authorization)
* Imputabilité (Accounting)

Les trois points définis ci-dessus sont importants pour une bonne gestion et une bonne sécurité d’un réseau. Ils devraient être disponibles à tout point d’entrée d’un réseau.

Tous les utilisateurs distants accèdent au réseau à partir d’un NAS (Network Access Server), aussi appelé Remote Access Server, ou Terminal Server. Un NAS est une interface qui accepte tout accès distant. Le NAS connecte les utilisateurs distants au réseau interne.

Après s’être connecté, l’utilisateur distant a accès à toutes les ressources du réseau interne (serveurs, partages, communication avec les autres utilisateurs, ...)

La mise en place des AAA au point d’entrée du réseau garantit un contrôle sur les connexions des utilisateurs au réseau, ainsi que sur ce qu’ils sont autorisés ou non à faire.

**RADIUS**

Le protocole RADIUS (Remote Authentication Dial-In User Service), est un protocole d'authentification standard, normalisé par l’IETF et défini par un certain nombre de RFC. (principalement 2138 et 2139).

RADIUS est un protocole client-serveur permettant de centraliser des données d'authentification.

Le protocole RADIUS repose principalement sur un serveur (le serveur RADIUS), relié à une base d'identification (base de données, annuaire LDAP, etc.) et un client RADIUS, appelé NAS (Network Access Server), faisant office d'intermédiaire entre l'utilisateur final et le serveur. L'ensemble des transactions entre le client RADIUS et le serveur RADIUS est chiffrée et authentifiée grâce à un secret partagé.

**Fonctionnement de RADIUS**

Un utilisateur envoie une requête au NAS afin d'autoriser une connexion à distance ;

* Le NAS achemine la demande au serveur RADIUS ;
* Le serveur RADIUS consulte la base de données d'identification afin de connaître le type de scénario d'identification demandé pour l'utilisateur.

Suite à un ACCESS-REQUEST, soit le scénario actuel convient, soit une autre méthodes d'identification est demandée à l'utilisateur.

Le serveur RADIUS retourne ainsi une des quatre réponses suivantes :

o **ACCESS-ACCEPT**l'identification a réussi ;

o **ACCES-REJECT**   
l'identification a échoué ;

o **ACCESS-CHALLENGE**le serveur RADIUS souhaite des informations supplémentaires de la part de l'utilisateur et propose un « défi » (en anglais « challenge ») ;

o **CHANGE PASSWORD**le serveur RADIUS demande à l'utilisateur un nouveau mot de passe.

La dernière version du protocole RADIUS est normalisée par l'IETF dans deux RFC : RFC 2865 (RADIUS authentication) et RFC 2866 (RADIUS accounting) de juin 2000

**TACACS (propriétaire Cisco)**

Le protocole TACACS+ (Terminal Access Controller Access Control System)

TACACS+ est la dernière version du protocole TACACS. TACACS+ utilise TCP pour son transport (contrairement à TACACS qui était basé sur l’UDP). Il utilise le port 49 (login). Il gère séparément les trois fonctions AAA contrairement à d’autres protocoles d’authentification. TACACS+ prévoit une implémentation pour chacune des trois, mais une configuration n'exige pas de toutes les utiliser

TACACS+ utilise la notion de session pour ses communications entre le client (NAS) et le serveur. Une session ne contient qu’un échange, ou d’authentification, ou d’autorisation, ou d’imputabilité. TACACS+ peut utiliser l’identifiant des sessions pour chiffrer les paquets.

**Authentification**

TACACS+ peut aussi bien utiliser des techniques d’authentification classiques type ouverture de session/mot de passe statique, ou bien des procédés plus évolués à base de challenge avec authentification réciproque.

Par exemple, lors d’une nouvelle connexion, le NAS émet un message START au serveur décrivant le type d’authentification à utiliser. En retour, le démon envoie un message REPLY. Ce type de message peut indiquer ou bien que l’authentification est terminée, ou bien qu’elle doit continuer, auquel cas, le client récupère l’information manquante et la retourne dans un message CONTINUE.

Le type de requête provenant du serveur peut être une demande GETDATA, GETUSER ou GETPASS. GETDATA est une requête générique de récupération d’information du profil utilisateur.

**Autorisation**

Lors d’un accès à un service particulier, le NAS ouvre une session d’autorisation. Cette session consiste juste en l’échange d’une paire de messages : REQUEST/RESPONSE. La requête décrit l’authentification pour l’utilisateur ou le processus qui demande l’accès au service.

La réponse du serveur contient un ensemble d’attributs pouvant restreindre ou modifier les actions du client, plutôt qu’une simple réponse affirmative de type oui/non.

**Imputabilité**

Les échanges utilisés lors de l’imputabilité sont similaires à ceux employés lors de l’autorisation (REQUEST/RESPONSE). Au démarrage et à la terminaison d’un service, on émet un paquet START et STOP.

De plus, le protocole TACACS+ propose l’émission de paquets UPDATE servant à confirmer qu’un service est en cours d’utilisation.

**Comparaison entre TACACS+ et RADIUS**

|  |  |  |
| --- | --- | --- |
|  | TACACS+ | **RADIUS** |
| **Protocole** | TCP : port 49 | UDP : port 1645 (1812 normalement) |
| **Chiffrement** | Chiffrement du paquet entier | Chiffrement du mot de passe |
| **Architecture AAA** | Les AAA sont indépendants | Autorisation liée à l’authentification |
| **Émission du profile** | Profil émis champs par champs à la demande du NAS | Profil global envoyé au NAS lors de la fin de l’authentification |
| **Protocoles supportés** | Support complet | Pas de NetBEUI |
| **Challenge / Réponse** | Bidirectionnel | Unidirectionnel |

**Modèles de contrôle d’accès**

Il existe trois principaux modèles d’accès…

* Modèle d’accès discrétionnaire (Discretionary Access Control ou DAC)  
  Modèle où le propriétaire d’un objet défini les droits des autres sujets.
* Modèle d’accès obligatoire (Mandatory Access Control ou MAC)  
  Modèle basé sur une hiérarchisation des sujets et des objets où le système s’assure du respect d’une politique très stricte.
* Modèle d’accès basé sur les rôles (Role-based Access Control ou RBAC)

**Modèle d’accès discrétionnaire**Le sujet créant le fichier en est le propriétaire.  
Le propriétaire du fichier peut accorder les droits aux autres sujets.  
(Aucun accès, Plein contrôle, Lecture, Écriture, Exécution, Suppression et Changement)  
Ce modèle permet toutes les actions sauf de modifier l’ACL et le propriétaire.  
Ce modèle est généralement implémenté avec des ACLs (Access Control Lists).  
Ce modèle est utilisé par Unix, Linux, Windows, Macintosh.

**Modèle d’accès obligatoire**Ce modèle d’accès ne permet pas aux sujets autant de liberté que le modèle DAC.   
Tous les sujets et objets possèdent une étiquette de sécurité (security label).  
Il existe une hiérarchisation des sujets (habilitation de sécurité ou security clearance en anglais) et des objets (classification) tel que confidentiel, secret, top secret, …

* Catégories  
  Afin de permettre d’implémenter le principe de droit d’accès minimal.
* Rôles  
  L’officier de sécurité défini les règles.

En fonction des niveaux de sécurité et des catégories

* L’administrateur configure le système
* Le système d’exploitation s’assure qu’aucune règle ne soit violée.

Ce modèle est utilisé dans des environnements où la classification et la confidentialité des informations sont essentielles.  
Il principalement est principalement utilisé par les militaires  
Il faut utiliser des versions particulières des systèmes d’exploitation tels que Trusted Solaris, SE Linux.

**Modèle d’accès basé sur les rôles (Role-based Access Control ou RBAC)**Ce modèle permet de définir les règles d’accès en fonction du rôle qu’un sujet a.

* Idéal pour le cas où la rotation du personnel est grande.  
  Ainsi, un rôle doit pouvoir effectuer un certain nombre d’opérations et de tâches.
* Complexe au niveau administratif.
* Les ACLs sont complexes à définir et à maintenir.  
  Problème de cohérence

# Haute disponibilité

**Les fondamentaux de la Haute Disponibilité**

Aujourd'hui un constat s'impose dans l'informatique…  
tout doit fonctionner à tout moment.

C'est d'autant plus vrai et critique en entreprise que l'activité de production (dans les services mais aussi l'industrie) dépend de plus en plus du système d'information de l’entreprise (CRM, logiciel logistique, site Web...) et de sa connexion au monde extérieur.

La haute disponibilité est un terme souvent utilisé en informatique, à propos d'architecture de système ou d'un service pour désigner le fait que cette architecture ou ce service a un taux de disponibilité convenable.

La disponibilité d'une application est inversement proportionnelle à son temps d'immobilisation total dans une certaine période (un mois en principe) et le temps d'immobilisation total est simplement la somme des durées de chaque panne ou interruption.

Par conséquent, pour augmenter la disponibilité d'un système,

* **il faut diminuer la durée des pannes,**
* réduire leur fréquence,
* ou les deux.

La haute disponibilité se caractérise par la dépendance que créent Internet et les nouvelles technologies, qui ne sont pas disponibles en permanence.

Il n'y a pas de norme en ce qui concerne la durée d'une interruption de service.   
Cela dépend du contexte et de la criticité de l'application.

Par exemple un système de navigation embarqué dans un avion sera conçu pour avoir une période d'indisponibilité de 5 minutes par an, alors que l'application de facturation d'une entreprise sera conçue pour une période d'indisponibilité d'une journée.

Deux moyens complémentaires sont utilisés pour améliorer la haute disponibilité:

* la mise en place d'une infrastructure matérielle spécialisée, généralement en se basant sur de la redondance matérielle. Est alors créé un agrégat (*cluster*) de haute-disponibilité (par opposition à un agrégat de calcul) : une grappe d'ordinateurs dont le but est d'assurer un service en évitant au maximum les indisponibilités;
* la mise en place de processus adaptés permettant de réduire les erreurs et d'accélérer la reprise en cas d'erreur.   
  ITIL contient de nombreux processus de ce type.

**Généralités**

On définit la haute disponibilité comme un **système permettant d'assurer une continuité opérationnelle d'un service sur une période donnée**.   
Pour mesurer la disponibilité, on utilise une **échelle qui est composée de 9 échelons.**   
Un service Hautement Disponible est **99% disponible** soit moins de **3,65 jours par an**.

Du point de vue de l'utilisateur, un service possède deux états:

* service approprié, c'est-à-dire conforme aux attentes;
* service inapproprié, c'est-à-dire non conforme aux attentes.

Les architectures hautement disponibles reposent sur deux composants qui sont :

* la redondance matérielle (cluster de haute disponibilité)   
  et
* la mise en place de processus de « basculement » de l’agrégat (partie logicielle).

Il existe plusieurs moyens de limiter les défaillances d'un service :

* la prévention des fautes   
  consistant à éviter les fautes en les anticipant.
* la tolérance aux fautes   
  dont l'objectif est de fournir un service conforme aux spécifications malgré les fautes en introduisant une redondance.
* l'élimination des fautes   
  visant à réduire le nombre de fautes grâce à des actions correctives.
* la prévision des fautes   
  en anticipation les fautes et leur impact sur le service.

**Mesure de la disponibilité**

Afin de calculer la disponibilité, les métriques suivantes sont utilisées:

* MTBF (*Mean Time Between Failure*)  
  mesure du temps estimé entre 2 défaillances d'un système.
* MTTR (*Mean Time to Resolution*)  
  mesure du temps estimé pour restaurer la fonctionnalité.

La formule de calcul de disponibilité est:

**Disponibilité = MTBF / (MTBF + MTTR)**

Le tableau suivant présente le temps d'indisponibilité (en anglais *downtime*) sur une base d'une année (365 jours) en fonction du taux de disponibilité.   
La disponibilité se mesure souvent en pourcentage

|  |  |  |  |
| --- | --- | --- | --- |
| **Disponibilité en %** | **Indisponibilité  par année** | **Indisponibilité  par mois** | **Indisponibilité par semaine** |
| 90 % ("un neuf") | 36,5 jours | 72 heures | 16,8 heures |
| 95 % | 18,25 jours | 36 heures | 8,4 heures |
| 98 % | 7,30 jours | 14,4 heures | 3,36 heures |
| 99 % ("deux neuf") | 3,65 jours | 7,20 heures | 1,68 heures |
| 99,5 % | 1,83 jours | 3,60 heures | 50,4 minutes |
| 99,8 % | 17,52 heures | 86,23 minutes | 20,16 minutes |
| 99,9 % ("trois neuf") | 8,76 heures | 43,2 minutes | 10,1 minutes |
| 99,95 % | 4,38 heures | 21,56 minutes | 5,04 minutes |
| 99,99 % ("quatre neuf") | 52,56 minutes | 4,32 minutes | 1,01 minute |
| 99,999 % ("cinq neuf") | 5,26 minutes | 25,9 secondes | 6,05 secondes |
| 99,9999 % ("six neuf") | 31,5 secondes | 2,59 secondes | 0,605 secondes |

L'amalgame est souvent fait, à tort, entre la haute disponibilité et le plan de reprise d'activité. Il s'agit de deux tâches différentes, complémentaires pour atteindre la disponibilité continue.

**Contrat de niveau de service   
Service Level Agreement (SLA)**

Le *service level agreement* (SLA) est un document qui définit la qualité de service requise entre un prestataire et un client.

Le service level agreement « garantie du niveau de service » ou plus simplement « convention de service » est donc un contrat (ou la partie du contrat de service) dans lequel on formalise la qualité du service en question. Dans la pratique, le terme SLA est quelquefois utilisé en référence aux modalités et/ou à la performance (du service) tel que défini dans le contrat.

Les spécifications techniques d’un SLA sont généralement décrites soit grâce à un SLS (*service level specification*) que l’on pourrait traduire en français par spécification de niveau de service ou grâce à un SLO (*service level objectives*) que l’on pourrait traduire en français par objectifs de niveau de service.

* Le SLS est techniquement un guide spécifiant certains critères que le service doit fournir.
* Le SLO fournit, quant à lui, le nécessaire permettant les mesures de la qualité des services telles que décrites dans le SLS.

**Les fautes ou failles**

L'origine des fautes peut être schématisée de la manière suivant :

**Origines physiques**elles peuvent être d'origine naturelle ou criminelle

* Désastre naturel (innondation, séisme, incendie)
* Environnement (intempéries, taux d'humidité de l'air, température) ;
* Panne matérielle ;
* Panne du réseau ;
* Coupure électrique.

**Origines humaines**elles peuvent être intentionnelles ou fortuites

* Erreur de conception (bogue logiciel, mauvais dimensionnement du réseau) ;

**Origines opérationnelles**elles sont liées à un état du système à un moment donné :

* Bogue logiciel ;
* Dysfonctionnement logiciel ;

**Tolérance aux pannes**

La capacité d'un système à fonctionner malgré une défaillance d'une de ses composantes est appelée tolérance aux pannes   
(parfois nommée tolérance aux fautes », en anglais *fault tolerance*).

Lorsqu'une des ressources tombe en panne, les autres ressources prennent le relais afin de laisser le temps aux administrateurs du système de remédier à l'avarie.   
En anglais le terme de « *Fail-Over Service* » (noté FOS) est ainsi utilisé.

**Sauvegardes**

Néanmoins, la mise en place d'une architecture redondante ne permet que de s'assurer de la disponibilité des données d'un système mais ne permet pas de protéger les données contre les erreurs de manipulation des utilisateurs ou contre des catastrophes naturelles telles qu'un incendie, une inondation ou encore un tremblement de terre.

Il est donc nécessaire de prévoir des mécanismes de sauvegardes, idéalement sur des sites distants, afin de garantir la pérennité des données.

Par ailleurs, un mécanisme de sauvegarde permet d'assurer une fonction d'archivage, c'est-à-dire de conserver les données dans un état correspondant à une date donnée.

**Internet et la Haute Disponibilité**

Dans de plus en plus d'entreprises, Internet se situe au cœur de l'activité et le besoin de disponibilité est donc constant. En effet, ce média est utilisé tout autant pour communiquer vers l'extérieur que pour supporter de nombreuses applications de l'entreprise (CRM, ERP, etc.) ou encore la téléphonie.

Il convient donc de distinguer le besoin de l'entreprise sur les deux niveaux

* services disponibles pour les clients   
  versus
* services nécessaires au fonctionnement interne.

L'un des exemples le plus parlant est celui des sites web des entreprises, qui sont aujourd'hui au centre de la communication et de la plupart des métiers des entreprises.

La haute disponibilité des sites web est organisée autour de différents axes qui peuvent être primordiaux:

* la redondance des matériels,
* les localisations des matériels,
* l'application de mises à jour de sécurité des applications serveurs,
* la sécurisation du réseau de l'entreprise,
* la disponibilité permanente de solution de sauvegarde/secours/reprise sur incident,
* le dimensionnement en puissance des matériels.

**Redondance des matériels**

La redondance est le mécanisme qui permet de dupliquer un ou plusieurs composants d'une architecture par un ou plusieurs éléments identiques.

Avoir n serveur(s) sur x site(s) permettra une redondance de l'information, avec un risque de panne divisé par x+n...

Toutefois il faut des systèmes qui permettent de basculer automatiquement d'un site à l'autre. Les systèmes les plus couramment mis en œuvre afin d'assurer cette redondance sont les clusters.Les clusters peuvent être actif/passif ou actif/actif.

* Le premier cas représente un groupe de machines de secours sur lequel on basculera l'infrastructure
* alors qu'un système actif/actif permettra d'avoir les deux systèmes en fonctionnement en parallèle; dans ce cas, un seul des deux matériels peut fonctionner en solo.

**Maintien des applications et mises à jour des logiciels**

Les applications peuvent présenter des bugs, la résolution par les mises à jour permet de corriger ces défauts. Ainsi on peut éviter que des personnes malveillantes explorent une faille qui permettrait l'accès aux informations de l'entreprise. Disposer de service de maintenance s'avère donc important et parfois compte-tenu des compétences techniques nécessaires.

**Reprise sur incident au cœur du dispositif**

Il s'agit d'un plan qui permet de reprendre une activité totale ou partielle suite à un sinistre survenu sur le système d'information. Le but de ce plan est de minimiser l'impact du sinistre sur l'activité de l'entreprise.

Les points essentiels dans un plan de reprise sont:

* la sauvegarde des équipements;
* la disponibilité de machines de secours;
* des solutions de secours, avec mode dégradé (en qualité de service) ex.: un lien de backup avec un débit plus petit.

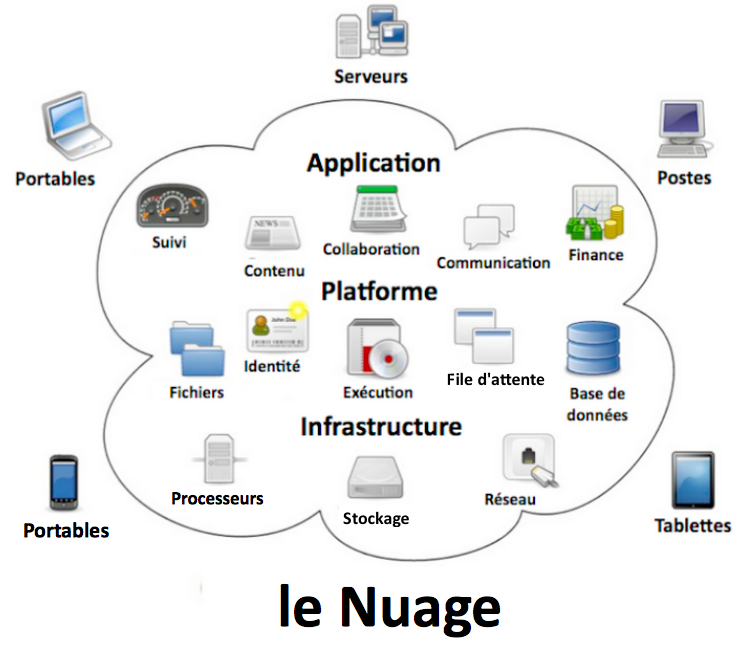
**Informatique en nuage** (*cloud computing*)

L’informatique en nuage, abrégé en cloud ou l’informatique en nuage désigne un ensemble de processus qui consiste à utiliser la puissance de calcul et/ou de stockage de serveurs informatiques distants à travers un réseau, généralement Internet.

Ces ordinateurs serveurs sont loués à la demande, le plus souvent par tranche d'utilisation selon des critères techniques (puissance, bande passante …) mais également au forfait.

**Principes - le Nuage**

Un nuage (en anglais *cloud*) est un ensemble de matériel, de raccordements réseau et de logiciels qui fournit des services sophistiqués que les individus et les collectivités peuvent exploiter à volonté depuis n'importe où dans le monde.   
L‘informatique en nuage est un basculement de tendance:  
au lieu d'obtenir de la puissance de calcul par acquisition de matériel et de logiciel, le consommateur se sert de puissance mise à disposition par un fournisseur via Internet.



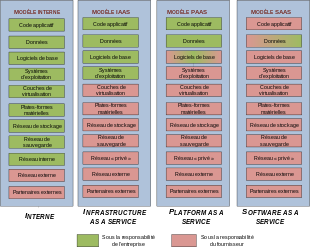
Les caractéristiques essentielles d'un nuage sont la disponibilité mondiale en libre-service, l'élasticité, l'ouverture, la mutualisation et le paiement à l'usage:

* ressources en libre-service et adaptation automatique à la demande  
  La capacité de stockage et la puissance de calcul sont adaptées automatiquement au besoin d'un consommateur. Ce qui contraste avec la technique classique des hébergeurs où le consommateur doit faire une demande écrite à son fournisseur en vue d'obtenir une augmentation de la capacité - demande dont la prise en compte nécessite évidemment un certain temps. En cloud computing la demande est automatique et la réponse est immédiate;
* ouverture  
  Les services de l’informatique en nuage sont mis à disposition sur Internet et utilisent des techniques standardisées qui permettent de s'en servir aussi bien avec un ordinateur qu'un téléphone ou une tablette;
* mutualisation  
  Elle permet de combiner des ressources hétérogènes (matériel, logiciel, trafic réseau) en vue de servir plusieurs consommateurs à qui les ressources sont automatiquement attribuées. La mutualisation améliore l'évolutivité et l'élasticité et permet d'adapter automatiquement les ressources aux variations de la demande;
* paiement à l'usage  
  La quantité de service consommée dans le nuage est mesurée, à des fins de contrôle, d'adaptation des moyens techniques et de facturation.

**Services**

Cette représentation des différents modèles de service montre comment les responsabilités sont théoriquement réparties suivant les modèles internes, IaaS, PaaS, SaaS.

* **IaaS** (*infrastructure as a service*)  
  C’est le **service de plus bas niveau**. Il consiste à **offrir un accès à un parc informatique virtualisé**. Des machines virtuelles sur lesquelles le consommateur peut installer un système d'exploitation et des applications. Le **consommateur est ainsi dispensé de l'achat de matériel informatique**. Ce service s'apparente aux services d'hébergement classiques des centres de traitement de données, et la tendance est en faveur de services de plus haut niveau, qui font davantage abstraction de détails techniques.
* **PaaS** (*platform as a service*)  
  Dans ce type de service, situé juste au-dessus du précédent, le **système d'exploitation et les outils d'infrastructure sont sous la responsabilité du fournisseur**. Le **consommateur a le contrôle des applications et peut ajouter ses propres outils**. La situation est **analogue à celle de l'hébergement web** où le consommateur loue l'exploitation de serveurs sur lesquels les outils nécessaires sont préalablement placés et contrôlés par le fournisseur. La différence étant que les systèmes sont mutualisés et offrent une grande élasticité - capacité de s'adapter automatiquement à la demande, alors que dans une offre classique d'hébergement web l'adaptation fait suite à une demande formelle du consommateur.
* **SaaS** (*software as a service*)  
  Dans ce type de service, des **applications sont mises à la disposition des consommateurs**. Les applications peuvent être **manipulées à l'aide d'un navigateur web** et le consommateur **n'a pas à se soucier d'effectuer des mises à jour, d'ajouter des patches de sécurité et d'assurer la disponibilité du service**.   
  Gmail est un exemple de tel service. Il offre au consommateur un service de courrier électronique et le consommateur n'a pas à se soucier de la manière dont le service est fourni.



Les caractéristiques de l’informatique en nuage intéressantes pour les entreprises sont :

* la réduction du coût total de possession des systèmes informatiques;
* la facilité d'augmenter ou de diminuer les ressources;
* la possibilité de décharger les équipes informatiques des entreprises.

# Laboratoire- Répartition de charge

Une des composantes importantes de la sécurité informatique est la disponibilité : sécuriser un site web ne consiste pas seulement à éviter les accès indésirables ou à conserver des copies de sauvegardes, mais aussi à s’assurer que les services soient toujours accessibles.

Peu importe la performance ou la fiabilité d’un serveur, celui-ci ne sera jamais tout-à-fait à l’abri d’une panne. La meilleure assurance de continuité de service, dans ce contexte, consiste à en créer une copie conforme qui pourra être utilisée au besoin. Le terme utilisé pour décrire ce dédoublement est ***redondance***.

Il y a deux cas d’usage de la redondance. Le premier cas, qui est le plus simple, est ce qu’on appelle une architecture de ***basculement*** (en anglais, « *failover* ») : dans le cas où le serveur original tombe en panne ou doit être arrêté, sa copie prendra automatiquement le relais.

Dans le deuxième cas, on utilise les deux serveurs alternativement afin que l’ensemble des requêtes soit réparti équitablement entre eux. On nomme ce procédé la ***répartition de charge*** (ou en anglais « network load balancing » ou NLB).

Le basculement est utile uniquement dans le cas où un serveur s’arrête, volontairement ou involontairement. Toutefois, en temps normal, ce serveur seul parvient à répondre seul à l’ensemble des requêtes. La répartition de charge est quant à elle souvent utilisée lorsqu’un seul serveur aurait du mal à répondre à la demande et qu’en mettre deux permet de maintenir des performances acceptables. À noter, une architecture à répartition de charge peut très bien inclure des fonctionnalités de basculement; mais l’inverse est faux.

**Configuration**

Nous allons configurer deux serveurs web en équilibrage de charge conformément au schéma fonctionnel suivant :

Web1

Requêtes

Proxy

SQL

Web2

Dans cette configuration, on n’accède plus directement aux serveurs HTTP: l’adresse du site web est en fait celle d’un serveur proxy, et c’est lui qui se chargera de rediriger les requêtes vers un ou l’autre des serveurs.

Attention, le proxy n’est pas une passerelle entre deux réseaux. C’est simplement un serveur qui redirige les requêtes qu’il reçoit vers deux autres serveurs, alternativement. Notre réseau est conforme à la topologie suivante :

Web1

Web2

proxy

SQL

Passerelle

internets

On installera le programme ***HAProxy*** sur une machine virtuelle *TurnKey Core* (une distribution de linux très légère et simple à installer) pour assurer le service de répartition de charge. Évidemment, il faut s’assurer que les deux serveurs web sont identiques, et qu’ils le resteront malgré les changements qu’on pourra y apporter. Pour ce faire on les maintiendra synchronisés avec le service ***rsync***, qu’on lancera régulièrement avec une tâche *cron*.

**Synchronisation des serveurs http**

La première étape consiste à monter un second serveur HTTP identique au premier. Dans *VirtualBox*, la manière la plus simple consiste à *exporter comme une application virtuelle* notre serveur original, puis ensuite à créer une nouvelle machine virtuelle à partir de cette application exportée (menu *Importer une application virtuelle*).

On se retrouve ensuite avec deux serveurs identiques : notre serveur original et sa copie. On doit maintenant installer le service *rsync* sur l’original pour en faire un serveur *rsync*, et le serveur web copié sera pour sa part un client *rsync*. Ainsi, si on doit faire des modifications, on devra les faire sur le serveur original et non sur la copie.

**Partager la racine du serveur et démarrer le service *rsync***

Sur le serveur Apache original, partager le répertoire ***/var/www/*** dans *rsync* en ajoutant les lignes suivantes dans le fichier ***/etc/rsyncd.conf*** (si ce fichier n’existe pas, créez-le):

pid file = /var/run/rsyncd.pid

log file = /var/log/rsyncd.log

[webroot]

path = /var/www

Ensuite, lancer le service avec la commande suivante:

sudo rsync --daemon --config /etc/rsyncd.conf

Vous remarquerez que *rsync* n'est pas démarré via une commande de type « sudo service … ». Ceci a pour effet que pour l'arrêter, il faut utiliser la commande ***kill.***

Autre chose à remarquer, le répertoire partagé est complètement ouvert: dans une situation réelle, il faudrait évidemment restreindre les accès en définissant des permissions avec le paramètre ***auth users***.

**Configurer le client *rsync***

Sur le nouveau serveur, supprimez le contenu de ***/var/www*** puis lancez la commande suivante:

rsync -r IP\_DU\_SERVEUR\_ORIGINAL::webroot /var/www

Pour que par la suite la synchronisation se fasse automatiquement, définissez une tâche *cron* appropriée.

**Mise en place du proxy de répartition de charge**

Installez un serveur à partir de la distribution *TurnKey Core* disponible sur le site <http://www.turnkeylinux.org/core> .

Après avoir mis à jour la liste de dépôts logiciels avec la commande ***apt-get update***, installez *haproxy* avec la commande suivante :

apt-get install haproxy

Pour que *haproxy* soit automatiquement activé au démarrage du serveur, assurez-vous d’avoir la ligne suivante dans le fichier ***/etc/default/haproxy :***

ENABLED=1

Le fichier de configuration principal est ***/etc/haproxy/haproxy.cfg***. Remplacez son contenu par les lignes suivantes :

global

daemon

maxconn 256

defaults

mode http

timeout connect 5000ms

timeout client 50000ms

timeout server 50000ms

frontend public

bind ADRESSE\_DU\_PROXY:80

default\_backend prive

backend prive

server web1 ADRESSE\_SRV\_HTTP\_1:80 maxconn 32

server web2 ADRESSE\_SRV\_HTTP\_2:80 maxconn 32

listen admin

stats enable

stats auth admin:abc-123

bind ADRESSE\_DU\_PROXY:8080

Les sections ***global*** et ***defaults*** permettent de définir des paramètres qui auront un effet sur l’ensemble du service *haproxy*.

La section ***frontend*** définit l’adresse IP à laquelle le serveur attendra les requêtes entrantes (*bind*), et vers quels serveurs ces requêtes seront redirigées (*default\_backend*).

La section ***backend*** spécifie la liste des serveurs sur lesquels *haproxy* répartit les requêtes entrantes.

La section ***listen*** permet de définir d’autres comportements du serveur; ici, on l’utilise pour désigner une adresse où il sera possible de visualiser des statistiques de fonctionnement de *haproxy*. Celles-ci seront accessibles à l’URI ***haproxy?stats*** sur le serveur; ainsi, si le serveur proxy est à l’adresse 192.168.1.136, la page des statistiques sera à l’URL *http://192.168.1.136:8080/haproxy?stats.*

# Transport Layer Secure Secure Socket Layer

Transport Layer Security (TLS), et son prédécesseur Secure Sockets Layer (SSL), sont des protocoles de sécurisation des échanges sur Internet. Le protocole SSL a été développé à l'origine par Netscape. L'IETF, en a poursuivi le développement en le rebaptisant Transport Layer Security (TLS). On parle parfois de SSL/TLS pour désigner indifféremment SSL ou TLS.

TLS (ou SSL) fonctionne suivant un mode client-serveur. Il permet de satisfaire aux objectifs de sécurité suivants :

* l'authentification du serveur;
* la confidentialité des données échangées (ou session chiffrée);
* l'intégrité des données échangées;
* de manière optionnelle, l'authentification du client (mais dans la réalité celle-ci est souvent assurée par le serveur).

Le protocole est très largement utilisé, sa mise en œuvre est facilitée du fait que les protocoles de la couche application, comme HTTP, n'ont pas à être profondément modifiés pour utiliser une connexion sécurisée, mais seulement implémentés au-dessus de SSL/TLS, ce qui pour HTTP a donné le protocle HTTPS.

Un groupe de travail spécial de l'IETF a permis la création du TLS et de son équivalent en mode UDP, le DTLS. Depuis qu'il est repris par l'IETF, le protocole TLS a connu trois versions, TLS v1.0 en 1999, TLS v1.1 en 2006 et TLS v1.2 en 2008.

**Présentation**

Au fur et à mesure qu'Internet se développait, de plus en plus de sociétés commerciales se mirent à proposer des achats en ligne pour les particuliers. L'offre se mit à croître régulièrement, mais le chiffre d'affaires dégagé par le commerce électronique resterait modeste tant que les clients n'auraient pas une confiance suffisante dans le paiement par carte bancaire. Une des façons de sécuriser ce paiement fut d'utiliser des protocoles d'authentification et de chiffrement tels que SSL. La session chiffrée est utilisée pour empêcher un tiers d'intercepter des données sensibles transitant par le réseau : numéro de carte lors d'un paiement par carte bancaire, mot de passe lorsque l'utilisateur s'identifie sur un site…

Avec un système SSL, la sécurité a été sensiblement améliorée et les risques pour le client grandement réduits, comparés à l'époque où le paiement par internet était encore une technologie émergente. Bien que, comme tout système de chiffrement, le SSL/TLS ne pourra jamais être totalement infaillible, le grand nombre de banques et de sites de commerce électronique l'utilisant pour protéger les transactions de leurs clients peut être considéré comme un gage de sa résistance aux attaques malveillantes.

En 2009, TLS est utilisé par la plupart des navigateurs Web. L'internaute peut reconnaître qu'une transaction est chiffrée à plusieurs signes :

* l'URL dans la barre d'adresse commence par https et non http (https://...);
* affichage d'une une clé ou un cadenas, dont l'emplacement varie selon le navigateur : généralement à gauche de la barre d'adresse mais aussi dans la barre inférieure de la fenêtre;
* les navigateurs peuvent ajouter d'autres signes, comme le passage en jaune de la barre d'adresse (cas de Firefox).

Il existe quelques cas très spécifiques où la connexion peut être sécurisée par SSL sans que le navigateur n'affiche ce cadenas, notamment si le webmaster a inclus la partie sécurisée du code HTML au sein d'une page en http ordinaire, mais cela reste rare. Dans la très grande majorité des cas, l'absence de cadenas indique que les données ne sont pas protégées et seront transmises en clair.

# Certificat numérique

Un certificat électronique (aussi appelé certificat numérique ou certificat de clé publique) peut être vu comme une carte d'identité numérique. Il est utilisé principalement pour identifier et authentifier une personne physique ou morale, mais aussi pour chiffrer des échanges.

Il est signé par un tiers de confiance qui atteste du lien entre l'identité physique et l'entité numérique (Virtuel).

Le standard le plus utilisé pour la création des certificats numériques est le X.509.

**Différents types**

Les certificats électroniques respectent des standards spécifiant leur contenu de façon rigoureuse. Les deux formats les plus utilisés aujourd'hui sont :

* X.509 dont plusieurs RFC définissent les propriétés et usages [citation nécessaire] ;
* OpenPGP, défini dans la RFC 48802.

La différence notable entre ces deux formats est qu'un certificat X509 ne peut contenir qu'un seul identifiant, que cet identifiant doit contenir de nombreux champs prédéfinis, et ne peut être signé que par une seule autorité de certification. Un certificat OpenPGP peut contenir plusieurs identifiants, lesquels autorisent une certaine souplesse sur leur contenu, et peuvent être signés par une multitude d'autres certificats OpenPGP. Ce qui permet alors de construire des toiles de confiance.

**Utilité des certificats**

Les certificats électroniques sont utilisés dans différentes applications informatiques dans le cadre de la sécurité des systèmes d'information pour garantir :

* la non-répudiation et l'intégrité des données avec la signature numérique. ;
* la confidentialité des données grâce au chiffrement des données ;
* l'authentification ou l'authentification forte d'un individu ou d'une identité non-physique.

**Authentification par certificat numérique**

Dans la majorité des cas, l'utilisateur authentifie le serveur TLS sur lequel il se connecte. Cette authentification est réalisée par l'utilisation d'un certificat numérique X.509 délivré par une autorité de certification (AC). Des applications web peuvent utiliser l'authentification du poste client en exploitant TLS. Il est alors possible d'offrir une authentification mutuelle entre le client et le serveur. Le certificat client peut être stocké au format logiciel sur le poste client ou fourni par un dispositif matériel (carte à puce, token USB). Cette solution permet d'offrir des mécanismes d'authentification forte.

OpenSSL contient le service SSL (Secure Sockets Layers) [ couche de sockets sécurisée] qui est un procédé de sécurisation des transactions effectuées via Internet. Avec la collaboration de Mastercard, Bank of America, MCI et Silicon Graphics Netscape a mis au point le standard SSL. Fondé sur un procédé de cryptographie par clef publique, il garantit la sécurité de la transmission de données sur internet. Son principe consiste à établir un canal de communication sécurisé (chiffré) entre deux machines (un client et un serveur) après une étape d'authentification.

**OpenSSL**

Le système SSL est indépendant du protocole utilisé, ce qui signifie qu'il peut aussi bien sécuriser des transactions faites sur le Web par le protocole HTTP que des connexions via le protocole FTP, POP ouIMAP. En effet, SSL agit telle une couche supplémentaire, permettant d'assurer la sécurité des données, située entre la couche application et la couche transport (protocole TCP par exemple).

De cette manière, SSL est transparent pour l'utilisateur (entendez par là qu'il peut ignorer qu'il utilise SSL). Par exemple un utilisateur utilisant un navigateur internet pour se connecter à un site de commerce électronique sécurisé par SSL enverra des données chiffrées sans aucune manipulation nécessaire de sa part.

Actuellement la quasi intégralité des navigateurs supporte désormais le protocole SSL. Netscape Navigator affiche par exemple un cadenas verrouillé pour indiquer la connexion à un site sécurisé par SSL et un cadenas ouvert dans le cas contraire, tandis que Microsoft Internet Explorer affiche un cadenas uniquement lors de la connexion à un site sécurisé par SSL. Google offre une connexion sécurisée (https) pour son service de courrier électronique « gmail .»

Pour tout serveur web sécurisé par SSL l’URL commence par https://, où le "s" signifie secured (sécurité).

**Fonctionnement de SSL 2.0**

La sécurisation des transactions par SSL 2.0 est basée sur un échange de clés entre client et serveur. La transaction sécurisée par SSL se fait selon le modèle suivant :

* Dans un premier temps, le client se connecte au site marchand sécurisé par SSL et lui demande de s'authentifier. Le client envoie également la liste des cryptosystèmes qu'il supporte, triée par ordre décroissant selon la longueur des clés.
* Le serveur de réception de la requête envoie un certificat au client, contenant la clé publique du serveur, signée par une autorité de certification (CA), ainsi que le nom du cryptosystème le plus haut dans la liste avec lequel il est compatible (la longueur de la clé de chiffrement - 40 bits ou 128 bits - sera celle du cryptosystème commun ayant la plus grande taille de clé).
* Le client vérifie la validité du certificat (donc l'authenticité du marchand), puis crée une clé secrète aléatoire (plus exactement un bloc prétenduement aléatoire), chiffre cette clé à l'aide de la clé publique du serveur, puis lui envoie le résultat (la clé de session).
* Le serveur est en mesure de déchiffrer la clé de session avec sa clé privée. Ainsi, les deux entités sont en possession d'une clé commune dont ils sont seuls connaisseurs. Le reste des transactions peut se faire à l'aide de clé de session, garantissant l'intégrité et la confidentialité des données échangées.

**Certificat SSL**

* Un certificat SSL est composé de deux éléments:
* Une clef privée que seule la personne ayant créée le certificat possède.   
  Cette clef doit être protégée du vol !   
  Si quelqu'un en fait une copie, il peut dupliquer votre site, et duper les internautes.
* Une clef publique que tout le monde peut télécharger.
* La méthode de fonctionnement du cryptage SSL est simple:
* Lorsque le contenu est crypté à l'aide de la clef privé, seule la clef publique peut le décoder: Certitude que le contenu a été créé par le propriétaire du certificat.
* Lorsque le contenu est crypté à l'aide de la clef publique, seule la clef privé peut le décoder: Certitude que le contenu ne peut être lu que par le propriétaire du certificat.

**Autorités de certification**

Les autorités de certification sont des organismes enregistrés et certifiés auprès d'autorités publiques ou de gouvernance de l'internet qui établissent leur viabilité comme intermédiaire fiable. Ces organismes diffusent leurs propres clés publiques. Étant certifiées fiables ces autorités sont en contact direct avec les principaux producteurs de système d'exploitation et de navigateurs internet (tels que Mozilla Firefox, Internet Explorer, Google Chrome...) qui incluent nativement les listes de clés des autorités de certification3. C'est cette relation qui est à la base de la chaine de confiance. Ces clés sont appelées clés publiques racines ou clé root, elles sont utilisées pour identifier les clés publiques d'autres organismes. Les autorités de certification vérifie que la personne ayant créé un certificat SSL pour un domaine est bien le propriétaire du domaine. Une fois l'identité du créateur vérifiée, l'autorité de certification signe le certificat serveur à l'aide de sa clef privé.

Les systèmes d'exploitations et navigateurs Web disposent des clefs publiques des autorités de certification. Il leur est ainsi possible de contrôler les certificats SSL des sites Web et autres serveurs.

* Quelle autorité de certification pour quel besoin?
* Vous disposez d'un serveur Internet d'entreprise: vous devez utiliser une autorité de certification payante.
* Vous disposez d'un serveur Internet personnel: vous pouvez utiliser l'autorité de certification gratuite CACert.
* Vous disposez d'un serveur Intranet: vous devez créer votre propre autorité de certification

# Exercice SSL (1)

**Générer et signer un certificat**

OpenSSL est une boite à outils open source qui implémente SSL/TLS ainsi qu'une bibliothèque de cryptographie.

Il va nous permettre ici de générer un certificat auto-signé. Ce type de certificat est très utile dans un cadre d'utilisation personnelle, par exemple pour implémenter les variantes sécurisées des protocoles tels que HTTP, IMAP ou SMTP qui deviendront alors HTTPS, IMAPS et SMTPS.

A noter : lors de l'utilisation sur un site Web, le navigateur affichera un message d'avertissement sur la connexion en HTTPS. Ceci est du au fait que, comme son nom l'indique, le certificat utilisé est auto-signé, c'est à dire non certifié par une autorité de confiance telle que Verisign par exemple. C'est pour cela qu'il n'est pas conseillé d'utiliser un certificat auto-signé pour un site publique par exemple, mais plutôt dans un cadre restreint.

**Installer OpenSSL**

Pour installer OpenSSL, on fait appel au gestionnaire d’installation  
Apt-get install openssl

**Créer le certificat**

La création du certificat se déroule en trois étapes :

1. création de la clé privée
2. création d'une requête de certification
3. auto-signature

Dans le cas où le certificat sera certifié par une CA (autorité de certification) la procédure est la même, sauf que la requête de certification sera envoyée au CA au lieu de faire l'auto-signature.

**Créer la clé privée**

La création de la clé privée se fait grâce à la commande genrsa d'OpenSSL avec de quelques options :

* genrsa  
  création d'une clé avec l'algorithme RSA
* -des3 (facultatif)  
  indique la création d'une clé avec mot de passe (pass phrase)
* -rand  
  réservoir de données aléatoires
* -out  
  fichier de sortie
* 1024  
  taille de la clé, en bits

Par exemple :  
openssl genrsa -des3 -out maisonneuve.net.key 1024

Generating RSA private key, 1024 bit long modulus  
..................++++++  
..........................++++++  
e is 65537 (0x10001)  
Enter pass phrase for maisonneuve.net.key:  
Verifying - Enter pass phrase for maisonneuve.net.key:

Le fichier de sortie maisonneuve.net.key est à copier dans le répertoire /usr/local/ssl/private/ pour une utilisation ultérieure.

Cette clé doit rester secrète, il faut donc aussi réduire les droits sur ce fichier (lecture uniquement pour le propriétaire) :  
cp example.com.key /usr/local/ssl/private/  
chmod 400 /usr/local/ssl/private/maisonneuve.key

**Créer la requête de certification**

La création de la requête de certification se fait grâce à la commande req d'OpenSSL avec un certain nombre d'options :

* eq  
  gestion des requêtes de certification
* -new  
  nouvelle demande
* -key  
  clé privée à utiliser
* -out  
  fichier de sortie

La création d'une nouvelle demande de certification implique le renseignement d'un certain nombre d'informations sur le demandeur, sachant qu'ils ne sont pas forcément obligatoires (par exemple, Organizational Unit Name). Si vous ne voulez pas remplir un champ, mettre un espace.

A noter : l'information Common Name doit correspondre à l'URL que va saisir l'utilisateur, si le certificat est destiné à un usage Web ou au nom complet du serveur si autre utilisation (par exemple, imap.example.com)

La *pass phrase* demandée correspond au mot de passe fourni lors de la création de la clé privée.

Par exemple :  
openssl req -new -key maisonneuve.net.key -out maisonneuve.net.csr

Enter pass phrase for example.com.key:  
You are about to be asked to enter information that will be incorporated into your certificate request.  
What you are about to enter is what is called a Distinguished Name or a DN.  
There are quite a few fields but you can leave some blank  
For some fields there will be a default value, If you enter '.', the field will be left blank.  
-----  
Country Name (2 letter code) [AU]:CA  
State or Province Name (full name) [Some-State]:Quebec  
Locality Name (eg, city) []:Montreal  
Organization Name (eg, company) [Internet Widgits Pty Ltd]:Maisonneuve  
Organizational Unit Name (eg, section) []:info  
Common Name (eg, YOUR name) []:www.maisonneuve.net  
Email Address []:lsavard@maisonneuve.net

Please enter the following 'extra' attributes to be sent with your certificate request  
A challenge password []:  
An optional company name []:

Cette commande produit le fichier maisonneuve.csr qui peut être utilisé dans deux cas :

* utilisé pour l'auto-signature
* envoyé à une autorité de certification tierce

**Auto-signer le certificat**

La création du certificat auto-signé se fait grâce à la commande x509 d'OpenSSL avec un certain nombre d'options :

* x509  
  gestion des certificats
* -req  
  pour passer un certificat
* -days  
  validité du certficat, en jours
* -in  
  fichier contenant la demande de certification
* -signkey  
  auto-signature
* -out  
  nom du certificat de sortie

La *pass phrase* demandée correspond au mot de passe fourni lors de la création de la clé privée.

Par exemple :  
openssl x509 -req -days 1 -in maisonneuve.net.csr -signkey maisonneuve.key -out maisonneuve.net.cert

Signature ok  
subject=/C=CA/ST=Quebec /L=Montreal/O= Maisonneuve/OU=info [/CN=www.maisonneuve.net/emailAddress=lsavard@maisonneuve.net](mailto:/CN=www.maisonneuve.net/emailAddress=lsavard@maisonneuve.net)  
Getting Private key  
Enter pass phrase for maisonneuve.net.key:

Le fichier de sortie maisonneuve.net.cert est à copier dans le répertoire /usr/local/ssl/certs/ pour une utilisation ultérieure.  
cp maisonneuve.net.cert /usr/local/ssl/certs/

# Exercice SSL (2)

**Serveurs de certificats**

**Introduction**

Un certificat est, pour une entreprise informatisée, ce qu’est un passeport pour un individu.

Un certificat est émis par une Autorité de Certification (AC ou CA en anglais). On retrouve parmi les plus importante AC *VeriSign*, *Entrust* ou encore *Thawte*. L’autorité de certification a pour mission, après vérification de l’identité du demandeur du certificat à émettre, de signer et de garder à jour le certificat du demandeur.

L’autorité de certification (AC) peut opérer la gestion des certificats par elle-même ou bien peut décider de déléguer l’hébergement de la clé privée d’un certificat à un opérateur de certification (OC). L’AC contrôle l’opérateur selon les procédures établies dans ce que l’on désigne comme la déclaration des pratiques de certification.

Ainsi, l’infrastructure de gestion des clés permet alors de s’assurer que:

* les données transmises ne sont pas modifiées lors du transfert (intégrité)
* les données proviennent bel et bien de l’émetteur présumé   
  (signature et imputabilité)

Ces données peuvent être des informations personnelles et /ou des informations secrètes telles qu’un numéro de carte bancaire.

L’utilisation d’un certificat se fait à l’aide du protocole HTTP (non sécurisé), avant l’établissement d’une session HTTPS (sécurisé). Ce sont les protocoles TLS (*Transport Layer Secure*) ou SSL (*Socket Secure Layer*) qui permettent, par la suite, le chiffrement des données sur HTTP lors de la transaction de données à travers un réseau.

Les clés en jeu sont:

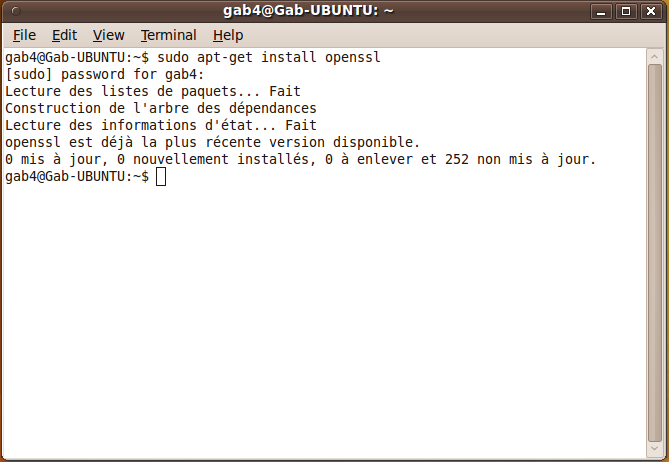
* celles créées de l’émetteur et du destinataire,
* la clé temporaire (ou de session) créée par l’émetteur à partir de la clé du destinataire  
  et finalement
* les clés du certificat de l’AC.

La clé privée ne doit pas être dévoilées en clair, donc on utilise un conteneur de clés. Les données qui seront chiffrées par la clé publique pourront être lues par une clé privée associée.

Dans ce qui suit, il est expliqué comment se définir en tant qu’autorité de certification (AC) pour pouvoir distribuer des certificats à des entités logiques ou physiques. Il peut être pratique de faire cela en entreprise si vous avez divers environnements de pré-production car cela pourrait coûter très cher en certificats.

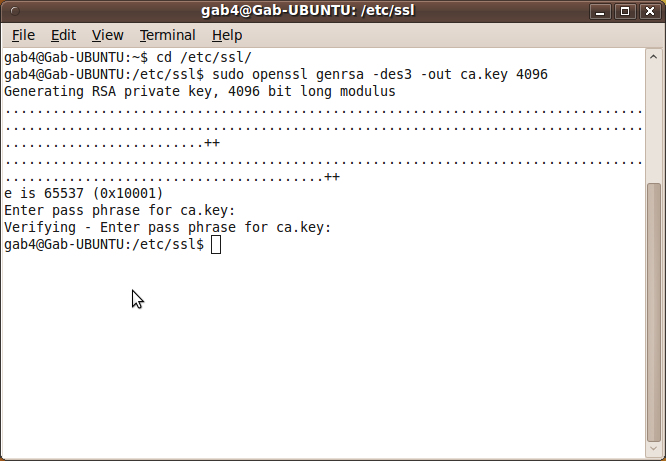
En tant qu’environnement de test, nous allons simulé le cas d’un entreprise en se désignant comme Autorité de certification sous Linux (Ubuntu) et publié le certificat en tant que AC reconnue à travers l’entreprise en utilisant Windows Server 2008.

**Partie 01  
Se définir en tant qu’Autorité de Certification sous Linux**



Premièrement, il faut s’assurer d’avoir une machine Linux fonctionnelle et endurcie (sécuritaire).   
Ensuite, ouvrir à une invite de commande (Terminal) et faire la commande ci-dessus.

**sudo apt-get install openssl**Cette commande installe le service openSSL et crée le répertoire /etc/ssl.



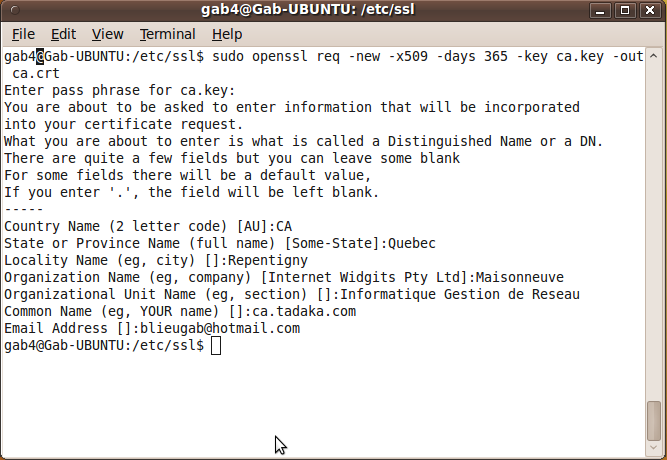
Résumé des commandes de la capture d’écran ci-haut…

Se déplacer ensuite dans */etc/ssl/*.

sudo openssl démarre openSSL avec les droits de root.   
  
Le reste de la commande sont des paramètres pour openSSL.  
genrsa –des3 –out ca.key 4096   
Il est indiqué à openSSL de générer une clé RSA (Rivest, Shamir, Adelman) de 4096 bits qui utilise l’algoruthme de chiffrement 3DES (Triple Data Encryption Standard).   
La clé générée est sauvegardée dans ca.key.   
Il faut entrer un mot de passe afin d’être certain de garder le tout secret… faute de quoi une personne malintentionnée pourrait usurper votre identité en tant qu’AC.

La clé est maintenant créée.

Il faut maintenant créer le certificat public CA.   
Lors de cette étape le système va nous interroger pour plusieurs informations qui feront partie du certificat.



Cette fois encore plus de paramètres *openSSL* sont utilisés.   
Voici leur signification….

En fait il est demandé à *openSSL* un nouveau certificat de type X509 qui sera valide pendant 365 jours (1 an). Il est défini que le fichier *ca.key* sera la clé privée et que le tout sera sauvegardé dans ca.crt.   
La durée de vie de ce certificat sera d’un an (une expiration d’un an peut être trop courte pour une entreprise).

X509 est une norme cryptographique. X.509 établie entre autres les formats standard de certificats électroniques et un algorithme pour la validation de chemin de certification

**Partie 02  
Installation du certificat sur un serveur Windows**

L’Autorité de Certification est maintenant établie et prête à être utilisé dans un environnement informatique. Dans ce qui suit, nous allons copier le contenu de *ca.crt* sur notre contrôleur de domaine sous Active Directory.

Au préalable, en utilisant la commande nano ca.crt, nous pouvons voir le contenu de notre clé privée. Le contenu d’une clé chiffrée tel que nous voyons dans la capture ci-dessous n’est tout simplement pas compréhensible par l’humain.

Toutefois un ordinateur en est totalement capable.



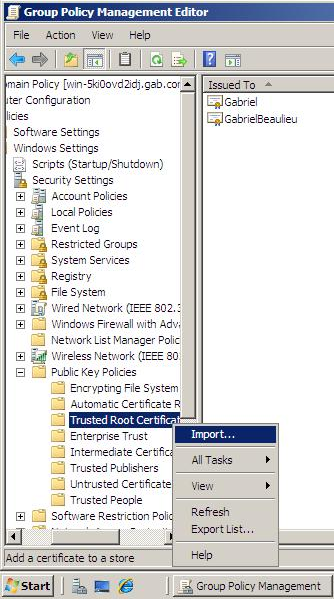
Copiez le contenu de ce fichier sur le contrôleur de domaine. Ensuite, renommer le fichier texte dans le but de changer l’extension de *.txt* à *.crt* pour que ce soit utilisable comme un certificat SSL. Renommer le tout *ca.gab.crt*.

Importer par la suite le Certificat de votre AC dans Active Directory

Il faut utiliser les stratégies de groupes (*group policy*) pour établir ce certificat comme certificat de confiance pour le domaine.   
Ouvrir le menu démarrer et sélectionner Outils d’administration – Gestion des stratégies de groupe (*Start, Administration tools, Group Policy Management)*.

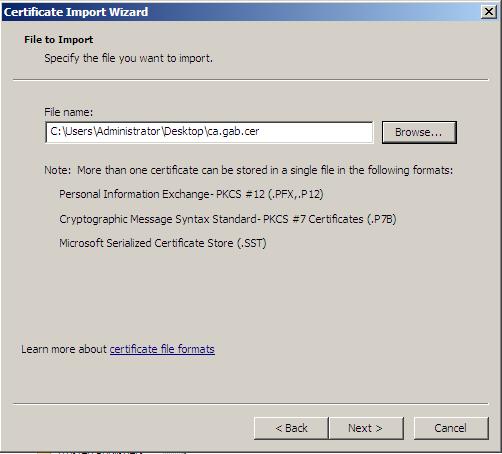
Ensuite, cliquer, à l’aide du bouton droit de la souris, sur Stratégie de domaine par défat (*Default Domain Policy*) et sélectionner le bouton Éditer (*Edit*).

De là, dérouler Configuration de l’ordinateur -> Stratégies -> Paramètres Windows -> Paramètres de sécurité -> Stratégies de clés publiques -> Autorités de certification fiables (*Computer Configuration -> Policies -> Windows Settings -> Security Settings -> Public Key Policies -> Trusted Root Certification Authorities*).



Cliquer, à l’aide du bouton droit de la souris, sur Importer (*Import*)...

L’assistant d’importation de certificat apparaît. Sélectionner le certificat et confirmer l’importation.



Cliquez sur Suivant (*Next*).

Il y a deux options… choisir Par défaut.

Le certificat apparaît désormais en tant que certificat de confiance dans la console Gestion des stratégies de groupe (*Group Policies Management*).

Vous avez maintenant une autorité de certification disponible pour usage dans votre environnement!

**Vidéos complémentaires**

<http://www.youtube.com/watch?v=rcrNK7k10Yw>  
<http://www.youtube.com/watch?v=BKjtMLFZ764>

Texte original : Gabriel Beaulieu – 2010

# Sécure Shell (SSH)

**Présentation**

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

Le protocole SSH a été conçu avec l'objectif de remplacer les différents programmes rlogin, telnet, rcp, ftp et rsh.

**Le protocole**

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

* La première version permet de se connecter à distance à un ordinateur afin d'obtenir un shell ou ligne de commande. Cette version souffrait néanmoins de problèmes de sécurité dans la vérification de l'intégrité des données envoyées ou reçues, la rendant vulnérable à des attaques actives. En outre, cette version implémentait un système sommaire de transmission de fichiers, et du port tunneling.
* La version 2 qui était à l'état de draft jusqu'en janvier 2006 est déjà largement utilisée à travers le monde.  
  Cette version est beaucoup plus sûre au niveau cryptographique, et possède en plus un protocole de transfert de fichiers complet, le SSH file transfer protocol.

Habituellement le protocole SSH utilise le port TCP 22. Il est particulièrement utilisé pour ouvrir un shell sur un ordinateur distant. Peu utilisé sur les stations Windows (quoiqu'on puisse l'utiliser avec PuTTY, mRemote, cygwin ou encore OpenSSH), SSH fait référence pour l'accès distant sur les stations Linux et Unix.

SSH peut également être utilisé pour transférer des ports TCP d'une machine vers une autre, créant ainsi un tunnel. Cette méthode est couramment utilisée afin de sécuriser une connexion qui ne l'est pas (par exemple le protocole de récupérations de courrier électronique POP3) en la faisant transférer par le biais du tunnel chiffré SSH.

Il est également possible de faire plusieurs sauts entre consoles SSH, c'est-à-dire ouvrir une console sur un serveur, puis, de là, en ouvrir une autre sur un autre serveur.

**SSH avec authentification par clés**

Avec SSH, l'authentification peut se faire sans l'utilisation de mot de passe ou de phrase secrète en utilisant la cryptographie asymétrique. La clé publique est distribuée sur les systèmes sur lesquels on souhaite se connecter. La clé privée, qu'on prendra le soin de protéger par un mot de passe, reste uniquement sur le poste à partir duquel on se connecte. L'utilisation d'un « agent ssh (en) » permet de stocker le mot de passe de la clé privée pendant la durée de la session utilisateur.

Cette configuration profite aussi à SCP et à SFTP qui se connectent au même serveur SSH.

**OpenSSH**

OpenSSH (*OpenBSD Secure Shell*) est un ensemble d'outils informatiques libres permettant des communications sécurisées sur un réseau informatique en utilisant le protocole SSH.

**Présentation technique**

OpenSSH est à la fois une brique logicielle du système OpenBSD1 et l'implémentation SSH la plus utilisée, y compris pour GNU/Linux. L'équipe du projet OpenSSH maintient cependant une version multiplate-forme sous l'appellation Portable OpenSSH2.

La suite logicielle OpenSSH inclut les outils suivants :

* SSH  
  Un remplaçant pour les clients rlogin et telnet  
  ssh utilisateur@exemple.com
* SCP  
  Un remplaçant pour le client rcp  
  scp utilisateur@exemple.com:~/toto
* SFTP  
  Un remplaçant pour le client ftp  
  sftp utilisateur@exemple.com
* SSHD  
  Le daemon SSH
* Sftp-server  
  Le daemon remplaçant le serveur FTP. sftp-server est un sous-système lancé par sshd quand celui-ci reçoit une demande de connexion d'un client sftp
* SSH-KEYGEN  
  Programme de génération, gestion et conversion des clés RSA, DSA et DSA basées sur les courbes elliptiques. Par défaut, la clé privée générée par ssh-keygen est un fichier texte sans extension, la clé publique équivalente est un fichier texte d'extension .pub. Ces paires de clés servent à l'authentification des hôtes (attribut HostKey du serveur sshd) et à l'authentification des utilisateurs   
  (attribut IdentityFile du client ssh).
* SSH-AGENT  
  Agent d'authentification.
* SSH-ADD  
  Agent de gestion des clés privées de l'utilisateur. ssh-agent et ssh-add font office de trousseau de clés.
* SSH-KEYSCAN  
  Utilitaire de récupération et vérification des clés publiques d'hôtes distants.
* SSH-KEYSIGN  
  Agent d'authentification servant si l'option HostbasedAuthentication est activée sur le serveur (option désactivée par défaut).
* SSH-COPY-ID  
  Utilitaire servant à déposer sa clé publique sur un serveur distant.

OpenSSH utilise la cryptographie asymétrique comme mécanisme d'authentification. À ce titre, OpenSSH gère les clés RSA, les clés DSA et les clés DSA basées sur les courbes elliptiques. OpenSSH reconnaît aussi les certificats X509 et les fichiers au format PKCS#12.

Si le modèle de sécurité n'emploie pas d'IGC, OpenSSH adoptera la méthode du Trust On First Use, où chaque apprentissage de nouvelle clé publique de serveur OpenSSH sera soumise à approbation de l'utilisateur. En cas d'acceptation après vérification de l'empreinte de la clé publique soumise, OpenSSH enregistrera cette clé dans un fichier known\_hosts afin de permettre sa vérification les fois suivantes. ssh-keyscan est responsable de cette vérification et de l'enregistrement des clés de confiance.

# Activité sur le protocole SSH

**Communications sécurisés entre systèmes distants**

**Introduction**

Le protocole SSH (Secure Shell) permet d’accéder à distance un système informatique, la plupart du temps tournant sur la plateforme Unix. Il assure une communication sécurisée puisque les trames sont encryptées dès le début de la transaction. Il est donc impossible de sniffé la communication avec des outils tel que WireShark afin d’en soutirer des informations privées. Il s’agit d’un choix par excellence pour l’administration à distance d’un serveur. En plus de permettre l’administration à distance, il est possible de transférer des fichiers en utilisant le protocole SSH. Cette fonction vise à remplacer l’insécure et vieillissant protocole FTP.

La distribution la plus utilisée du protocole SSH est la suite OpenSSH. Elle est inclut dans la très grande majorité des distributions GNU/Linux et présente le plus haut taux de compatibilité. La suite OpenSSH inclut, entre autres, les outils suivants, que nous allons utilisés :

• **ssh**, le client principal;

• **sshd**, le serveur;

• **scp**, une version de « cp » qui utilise le protocole SSH afin de transmettre des fichiers sur le réseau de manière complètement encryptée;

• **ssh‐keygen**, Génération de paires de clés privées/publiques.

Ces outils sont très probablement déjà installés sur votre système Linux. À vous de vérifier. Sinon, utiliser l’outil de gestions de paquetages de votre distribution afin de les installer (yum, apt‐get, portage, etc.).

Afin d’établir une connexion, l’ordinateur distant doit exécuter **sshd** (Secure Shell Daemon). Ce processus a comme tâche d’écouter sur le port 22 (par défaut) pour de nouvelles connexions, authentifier l’utilisateur distant, et fournir une console d’accès, selon les politiques du système. Sur une grande majorité des distributions GNU/Linux, le serveur sshd est démarré automatiquement à l’initialisation du système. Sa configuration est régit par le fichier

**sshd\_config** situé dans le répertoire **/etc/ssh/**.



Figure 1. Établissement d'une connexion SSH vers un serveur distant.

La figure 1 représente un exemple de connexion à l’aide du protocole SSH. Suite à l’établissement de la connexion, l’utilisateur obtient une invite de commande de la même façon que s’il se serait connecté physiquement au système à l’aide d’un clavier. Il hérite sur la machine distante des droits et privilèges de l’utilisateur avec lequel il s’est connecté (soit « olivier » dans l’exemple présent). Il est donc fortement conseillé d’interdire à l’utilisateur root de se connecter à distance directement. Malgré le fait que SSH soit un protocole extrêmement sécuritaire, il existe certaines techniques qui permettent l’usurpation d’identité. Ainsi, si un pirate réussi à se connecter en tant que root, il posséderait tous les moyens nécessaires afin de compromettre votre système. Tel que démontré dans la figure 1, il n’est pas nécessaire que l’utilisateur local soit le même que l’utilisateur distant. Ainsi, roger peut se connecter en que robert sur le serveur X, en tant que julie sur le serveur Y et bien sûr en tant que roger sur le serveur Z. Notez que la syntaxe ssh [*Username*@*Hostname*](mailto:Username@Hostname)n’a absolument rien à voir avec la syntaxe d’une adresse de courriel.

Une fois la connexion établie, toutes les ressources du système distant sont disponibles, exactement comme si l’utilisateur était devant le système même. Il est donc possible de gérer les services, administrer les comptes, surveiller les processus, etc. C’est ce qui fait du protocole

SSH un excellent choix pour l’administration à distance.

**La configuration**

**Le fichier /etc/ssh/sshd\_config**

Ce fichier contient la configuration du serveur SSH. Sa configuration est donc très importante. Elle reste toutefois simple pour une configuration de base.

Port 22

Protocol 2

LoginGraceTime 2m

PermitRootLogin no

MaxAuthTries 6

RSAAuthentication yes

PubkeyAuthentication yes

AuthorizedKeysFile .ssh/authorized\_keys

PasswordAuthentication yes

UsePAM yes

Figure 2. Exemple d’un fichier de configuration *sshd\_config.*

Référez‐vous à l’URL <http://www.openbsd.org/cgi>‐bin/man.cgi?query=sshd\_config ou tapez *man sshd\_config* afin d’obtenir la liste complète des paramètres disponibles accompagnés d’une explication de ceux‐ci.

Démarrage du service

Afin que votre système puisse accepter les connexions entrantes, vous devez vous assurez que le processus **sshd** est en cours d’exécution.

Selon votre distribution :

* **service sshd start**
* **/etc/init.d/sshd start**
* **/etc/init.d/ssh start**

Vérifiez que le serveur est bel et bien en cours d’exécution et qu’il accepte les connexions entrantes :

* **netstat ‐ntpl|grep sshd**

Si vous obtenez un résultat comme le suivant :

tcp 0 0 0.0.0.0:22 0.0.0.0:\* LISTEN 15397/sshd

Félicitation ! Votre serveur est prêt.

**Transfert s de fichiers**

La suite OpenSSH nous fournit un précieux outil permettant de transférer des fichiers sur le réseau de la même manière que la commande **cp** dans un environnement GNU/Linux. Il s’agit de **scp,** qui signifie « **s**ecure **c**o**p**y ». La syntaxe de cette commande est un amalgame de **ssh** et **cp**;

* **scp FichierLocalSource** [**username@hostname:**](mailto:username@hostname:)**/destination /distante/**

Ainsi, la commande

* **scp /home/phillipe/Pomme.zip** [**guy@206.167.44.100:/home**](mailto:guy@206.167.44.100:)**/guy/**

enverrait le fichier local *Pomme.zip* dans le dossier */home/guy/* de la machine distante *206.167.44.100* en tant que l’utilisateur *guy*.

Cette commande fonctionne de façon bidirectionnelle. Alors, la commande

* **scp r** [**guy@206.167.44.100:/home/gu**](mailto:guy@206.167.44.100:)**y/légumes/ /home/phillipe/**

copierait le dossier */home/guy/légumes/* stocké sur le système distant *206.167.44.100* ainsi que tout son contenu vers le répertoire local */home/phillipe/*.

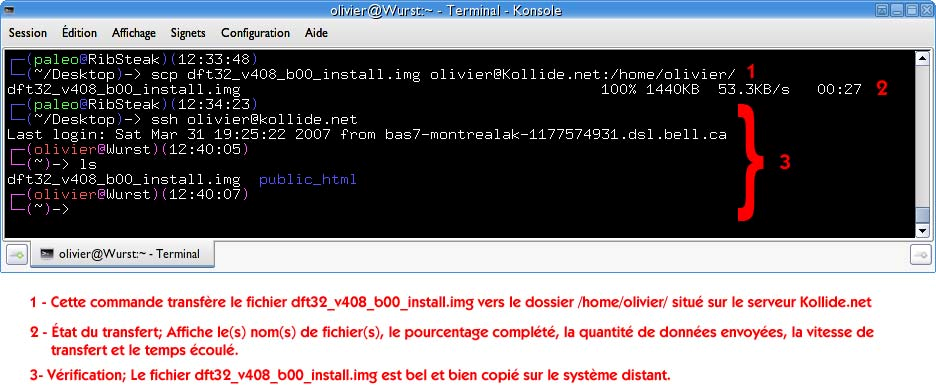


Figure 3. Copie d'un fichier local vers un système distant.

**Authentification n par paire s de clés privée s/publique s**

L’authentification est un principe de base lorsque nous parlons de protocoles réseau. Celle‐ci est nécessaire afin de s’allouer les droits et permissions sur un système distant qui sont attribué à l’utilisateur avec lequel nous nous authentifions. Il s’agit donc d’une cible de choix pour les pirates qui désirent pénétrer dans votre infrastructure. S’ils sont en mesure de cracké votre protocole d’authentification, cela leur ouvre la porte a bien plus. La méthode d’authentification la plus répandue, et pourtant l’une des moins sécure, est l’authentification par simple mot de passe. Cette méthode est très vulnérable à différentes attaques, surtout lorsque le protocole utilisé n’utilise aucune encryptions, tel que HTTP ou FTP. Le mot de passe est alors transféré en clair sur le réseau. Une simple attaque de sniffing suffit à compromettre votre infrastructure. De par sa nature extrêmement sécuritaire, SSH propose différentes méthode d’authentification. La plus sécure d’entre tous est l’authentification par paires de clés privées/publiques. Le principe est simple et facile à appliquer. Tout d’abord, le client doit posséder sa propre paire de clés. Cette paire lui est unique et il doit, selon les normes d’utilisation de cette méthode, être le seul individu à en posséder une copie. Cette paire est donc divisée en deux parties; la clé publique ainsi que la clé privée. Le serveur SSH distant doit posséder une copie de la clé publique et l’associer à l’utilisateur en question. Lors de la connexion, le client informe le serveur qu’il veut procéder à une authentification par clés. Le client ouvre alors sa clé privée et le serveur vérifie si elle correspond à la clé publique qu’il possède. Si oui, alors l’autorisation est allouée. La clé est transférée de manière encryptée, alors il est impossible de sniffer celle‐ci. De plus, pour une sécurité encore plus accrue, il est possible de protéger la clé privée par mot de passe. Ainsi, même si un individu s’empare de votre clé privée, elle lui est inutile sans le mot de passe. Lors de l’ouverture de la clé privée, si celle‐ci est protégée par mot de passe, le client SSH demandera alors à l’individu de le taper. Notez que ce mot de passe n’est pas transférer sur le réseau, il ne sert que pour l’ouverture de la clé privé.

**Génération d’une paire de clés**

C’est l’outil **ssh‐keygen** qui nous permet de générer ces paires de clés. Afin de générer une paire de clés utilisant la forte encryption RSA d’une force de 1024 bits :

* **ssh‐keygen –t rsa –b 1024**

Vous obtiendrez alors le résultat suivant :

Generating public/private rsa key pair.

Enter file in which to save the key (/home/olivier/.ssh/id\_rsa): Created directory '/home/olivier/.ssh'.

Enter passphrase (empty for no passphrase):

Enter same passphrase again:

Your identification has been saved in /home/olivier/.ssh/id\_rsa. Your public key has been saved in /home/olivier/.ssh/id\_rsa.pub.

The key fingerprint is:

60:22:6b:cb:cb:c7:a7:39:76:79:4b:67:29:7e:8c:68 [olivier@RibSteak](mailto:olivier@RibSteak)

Figure 4. Création d’une paire de clés de type RSA.

Vous possédez donc maintenant votre clé privé, enregistré tel qu’indiqué dans la figure 3 dans le fichier */home/olivier/.ssh/id\_rsa*. Évidemment, *olivier* sera remplacé par votre propre nom d’utilisateur.

La clé publique quant à elle est enregistrée dans */home/olivier/.ssh/id\_rsa.pub*. C’est le contenu de ce fichier que vous devrez donner au serveur distant.

**Ajout de la clé publique sur le serveur**

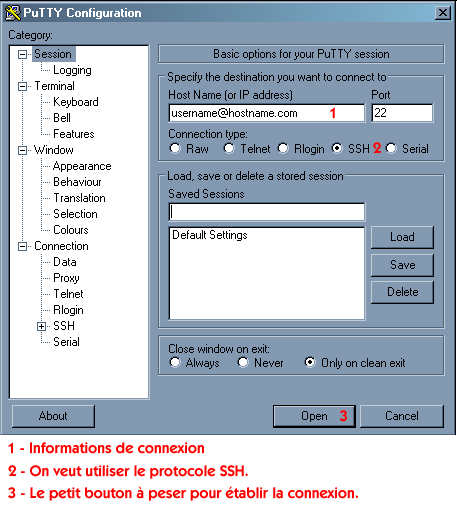
Afin que l’authentification par clés fonctionne correctement, vous devez ajouter votre clé sur le serveur afin que ce dernier la reconnaisse. Ce procédé consiste à copier le fichier *id\_rsa.pub* (ou son contenu) dans un fichier nommé *authorized\_keys* situé dans le dossier *.ssh/* dans le répertoire utilisateur. La méthode la plus recommandé pour copier sa première clé publique est d’utiliser l’utilitaire **scp**. Ensuite, il suffit de modifier le fichier *authorized\_keys* manuellement.



Le fichier *authorized\_keys* peut contenir plusieurs clés publiques afin que différents utilisateurs puissent partager le même compte sans partager un mot de passe commun. Pour se faire, vous devez simplement ajouter manuellement une nouvelle ligne pour chaque clé à ajouter.

**Client Windows**

Le client SSH par excellence pour Windows est sans aucun doute PuTTY. Il est gratuit et libre. Malgré le fait qu’il supporte une tonne de fonctionnalités, son utilisation est très simple.



**Activités**

Dans chaque équipe, initialisé deux postes de travail roulant sur GNU/Linux et un environnement Windows. Assurez‐vous qu’OpenSSH soit installé sur les deux postes GNU/Linux. Téléchargez l’utilitaire PuTTY sur le poste Windows

Dans un environnement complètement GNU/Linux :

1. Créez un compte utilisateur temporaire sur le poste B. À l’aide du poste A, connectez‐ vous au poste B en utilisant le compte utilisateur nouvellement créé et l’authentification par mot de passe.
2. Dans votre session SSH, utilisez la commande **wall** afin de transmettre un message à tous les utilisateurs connectés. Le message est‐il reçu localement sur le poste B ?
3. Copier le répertoire */etc/* du poste A au complet vers le répertoire */tmp/* du poste B en utilisant l’utilitaire **scp**.
4. Générez une paire de clés privée/publique et faite en sorte que l’authentification par clés fonctionne.
5. ransférer à nouveau le répertoire */etc/* vers le dossier */tmp/* sur le poste B à l’aide de l’utilitaire **scp**. Devez‐vous entrez un mot de passe à présent ?

**Utilisation de SSH à travers Internet**

Établissez une connexion SSH au serveur Atelier.Kollide.net sur le port 443 en utilisant les informations reçu en début d’atelier. Afficher le contenu du fichier Bravo situé dans votre répertoire personnel.

Dans un environnement mixte Windows (client) et GNU/Linux (serveur):

1. À l’aide de l’utilitaire PuTTY sur Windows, connectez‐vous à votre serveur SSH sur le poste B en utilisant l’utilisateur créé précédemment.
2. Si le temps le permet : Faites des recherches afin d’utiliser l’authentification par paires de clés privés/publiques avec PuTTY. Indice : **puttygen.exe**

# Modules PAM

**Introduction**

Tous les systèmes basés sur Unix possèdent un système d'authentification reposant sur les mots de passe. À partir de 1995, la bibliothèque PAM a été créée et a fournie une API pour les services relevant de l'authentification permettant à un administrateur système de “jouer” avec l'authentification sur le système à l'aide de modules. Nous allons voir comment, à l'aide de Linux-PAM, améliorer l'authentification des utilisateurs sur notre système Debian.

Le système PAM (*Pluggable Authentication Modules*) est, à la base, une suite de librairies partagées qui permet à l'administrateur de choisir comment les applications authentifieront les utilisateurs. PAM est donc une collection de modules qui forment une barrière entre un service de votre système et son utilisateur. L'utilisation de ces modules varie énormément allant du rejet de connexion de certains utilisateurs de certains groupes UNIX à l'implémentation de limites quant aux ressources de votre système.

La plupart des UNIX commerciaux actuels intègre ce système ainsi que pratiquement toutes les distributions Linux. Le pouvoir et la souplesse qu'offre PAM est une aubaine pour les développeurs d'applications pour systèmes Linux car cela leur évite de coder une interface d'authentification à part, ils leur suffit d'écrire un module PAM et ainsi les administrateurs pourront configurer leur application simplement avec toutes celles utilisant PAM. Ceci rend également la vie plus facile à l'administrateur qui ne devra connaitre que PAM et non tous les systèmes d'authentification pour chaque service mis en place.

**Le fichier de configuration de Linux-PAM**

Nous verrons comment PAM s'applique à un système GNU/Linux et plus particulièrement à Debian. Solaris et d'autres systèmes UNIX commerciaux ont une configuration un peu différente basée sur un seul fichier /etc/pam.conf. Cependant le système reste le même, sous Linux chaque service utilisant PAM aura son propre fichier de configuration. Sur la plupart des systèmes GNU/Linux, ces fichiers de configurations se situent dans le répertoire /etc/pam.d et sont nommés selon le nom du service. Par exemple, le fichier de configuration “login” se nomme /etc/pam.d/login. Jetons un œil au fichier pour voir à quoi cela ressemble :

* *auth required /lib/security/pam\_securetty.so*
* *auth required /lib/security/pam\_nologin.so*
* *auth sufficient /lib/security/pam\_ldap.so*
* *auth required /lib/security/pam\_unix\_auth.so try\_first\_pass*
* *account sufficient /lib/security/pam\_ldap.so*
* *account required /lib/security/pam\_unix\_acct.so*
* *password required /lib/security/pam\_cracklib.so*
* *password required /lib/security/pam\_ldap.so*
* *password required /lib/security/pam\_pwdb.so use\_first\_pass*
* *session required /lib/security/pam\_unix\_session.so*

**Les tâches gérées par PAM**

La première chose à noter est que la première colonne du fichier indique le type de tâche que PAM va gérer parmis les quatre types suivant : auth (gestion des authentification), account (gestion des comptes), password (gestion des mots de passes) et session (gestion des sessions). Tandis que de nombreux modules supportent plus d'une structure (pam\_unix par exemple les utilisent tous), d'autres, comme pam\_craklib, ne sont destinés qu'à une seule (dans le cas de pam\_craklib seule la structure 'password' sera utilisée). La compréhension de ces quatres structures est très importante pour une utilisation efficace de PAM. Voyons comment ils fonctionnent :

* ***authentication management*** (authentification)  
  Ceci permet à l'utilisateur de s'authentifier et de définir les droits du compte.  
  Les modules utilisant ce système demandent en général un mot de passe.
* ***account management*** (gestion de compte)  
  Structure qui permet la gestion des comptes utilisateurs. Par exemple, des restrictions peuvent être définies selon les horaires ou la charge du serveur.
* ***session management*** (gestion de session)  
  Ce sont les tâches à effectuer en début et fin de chaque session. Par exemple, démarrer un agent ssh, monter des disques, etc …
* ***password management*** (gestion des mots de passe)  
  Mise à jour du jeton d'authentification du compte d'un utilisateur, soit parce qu'il a expiré soit parce que celui-ci désire le modifier.

**Contrôle de réussite des modules PAM**

La colonne du milieu contient un mot clé qui détermine ce que la librairie PAM doit faire en cas de réussite ou d'echec du module. L'application n'est pas au courant du succès ou des échecs des modules dans /etc/pam.d, elle reçoit uniquement une réponse sommaire de la librairie PAM. Voici une présentation de ces quatres mots-clés :

* ***required***  
  Si un module “required” retourne un status qui n'est pas “success”, l'opération échouera mais seulement après que tous les modules en dessous soient invoqués.   
  La réussite d'au moins un des modules “required” est nécessaire.
* ***requisite***  
  Si un module “requisite” échoue, l'application en est tout de suite informée et aucun autre module n'est invoqué.   
  La réussite de tous les modules “requisite” est nécessaire.
* ***sufficient***  
  La réussite d'un seul module “sufficient” est suffisant. Tous les modules “sufficient” qui suivent ne seront pas invoqués.   
  Il faut noter, toutefois, que si un module “required” échoue avant un “sufficient”, l'opération échouera ignorant les autres modules “sufficient”.
* ***optional***  
  La réussite d'au moins un des modules “optional” est nécessaire si aucun autre n'a réussi.

**L'empilement des modules (*stacking*)**

Dans la colonne la plus à droite, nous voyons le chemin où est stocké le module. Dans notre exemple, quatre modules différents sont listés pour le type “auth”. En anglais le terme “stacking” est employé dans le jargon PAM pour indiquer le fait qu'on empile les modules. Ceci sous-entend que l'ordre dans lequel passent les modules est important (nous le verrons un peu plus bas). De plus, on peut également passer des options aux modules comme c'est le cas dans notre exemple avec le module “pam\_unix” qui prend l'argument “try\_first\_pass”. 3.5. Fonctionnement de l'empilage des modules

Maintenant que nous avons vu comment se présentait un fichier de configuration, attardons-nous un peu plus sur notre exemple et plus précisement sur le type “auth” et sa pile d'exécution en détaillant ce qui pourrait se produire en fonction de différents scénarios.

Dans notre fichier d'exemple, nous avons quatre modules empilés pour le type auth :

*auth required /lib/security/pam\_securetty.so  
auth required /lib/security/pam\_env.so  
auth sufficient /lib/security/pam\_ldap.so  
auth required /lib/security/pam\_unix.so try\_first\_pass*

Comme les modules sont invoqués dans l'ordre, voici ce qui va arriver :

1. Le module “pam\_securetty” va vérifier son fichier de configuration, /etc/securetty, afin de vérifier que le terminal utilisé est listé dans le fichier. Si cela n'est pas le cas, les connexions en tant qu'utilisateur root ne seront pas permises. Vu que le statut est à “required”, le module appelera les trois autres modules de la pile mais même si tous les autres sont ok, la connexion échouera. Notons que si le module avait le statut “requisite”, l'opération aurait été soldé par un échec et aucun des autres modules n'auraient été invoqué, peu importe de leur statut.
2. Le module “pam\_env” va fixer les variables d'environnement d'après le fichier /etc/security/pam\_env.conf.
3. Le module “pam\_ldap” va demander à l'utilisateur un mot de passe et vérifier ensuite le répertoire ldap indiqué dans le fichier /etc/ldap.conf pour authentifier l'utilisateur. Si cela échoue, l'opération peut réussir tout de même si le module “pam\_unix” authentifie l'utilisateur. Si “pam\_ldap” authentifie l'utilisateur, “pam\_unix” ne sera pas invoqué.
4. Le module “pam\_unix”, dans ce cas, ne demandera pas de mot de passe. L'argument “try\_first\_pass” lui indique d'utiliser le mot de passe fourni dans le module précédent (ici “pam\_ldap”). Il essaiera d'authentifier l'utilisateur en utilisant les appels systèmes getpw\*. Si “pam\_unix” échoue et que “pam\_ldap” avait échoué, l'opération échouera. Si “pam\_ldap” échoue mais “pam\_unix” est correct, l'opération sera exécutée.

**Les modules PAM disponibles**

Vous pouvez trouver sur le site http://www.kernel.org/pub/linux/libs/pam/modules.html une liste de modules disponibles pour PAM. Elle est assez impressionnante et vous permettra d'expérimenter toute sorte de configuration pour augmenter le degré d'authentification de votre système.

**PAM dans la vraie vie**

Si l'on cherche à comprendre comment fonctionne PAM pour un cas concret, il faut commencer par analyser le service associé au programme que l'on cherche à étudier. Prenons l'exemple de GDM, le gestionnaire de session de Gnome.

GDM a sur la majorité des distribution un service PAM associé en le fichier /etc/pam.d/gdm :

#%PAM-1.0  
auth required pam\_env.so  
auth substack system-auth  
auth optional pam\_gnome\_keyring.so  
auth required pam\_nologin.so  
account include system-auth  
password include system-auth  
session optional pam\_console.so  
session optional pam\_gnome\_keyring.so auto\_start  
session include system-auth

Comme vous le voyez, le service gdm est complet et définit une pile pour chaque groupe de gestion. Si l'on se concentre sur le groupe auth, nous avons le classique pam\_env.so qui va mettre en forme les variables d'environnement en fonction du fichier /etc/security/pam\_env.conf. D'un point de vue général, /etc/security est l'endroit où l'on trouve la configuration des modules PAM.

Ensuite nous avons une règle de type command-service qui va faire appel à system-auth/auth. Il s'agit d'une commande de type substack ce qui implique que si cela se passe mal dans ce service, les deux règles suivantes s'exécuteront tout de même. A savoir l'optionnel pam\_gnome\_keyring qui dérouille le gestionnaire de mots de passe de Gnome, et pam\_nologin en contrôle required qui empêche l'utilisateur de se connecter lorsqu'existe le fichier /etc/nologin.

Maintenant jetons un œil au service system-auth qui est le plus intéressant car c'est lui que finalement, par des include ou des substack, tous les services appellent à un moment ou à un autre. Ici j'ai limité la liste de règles au groupe de gestion auth:

#%PAM-1.0  
auth required pam\_env.so  
auth sufficient pam\_tcb.so shadow fork nullok prefix=$2a$ count=8  
auth sufficient pam\_ldap.so use\_first\_pass  
auth requisite pam\_deny.so

...

Le déroulement du service pour auth est donc :

* pam\_env, encore une fois.
* sufficient pam\_tcb.   
  Sur d'autres distributions pam\_tcb sera plutôt pam\_unix. le système TCB est un système qui vient remplacer, de manière plus sécurisée, le vénérable mécanisme basé sur shadow. Comme aucun mot de passe n'est fournit à ce module, il va chercher à en demander un à l'utilisateur et vérifier si ce mot de passe lui plaît. Si c'est le cas, l'exécution de la pile s'arrête là.
* pam\_ldap,  
  Si pam\_tcb échoue, le module LDAP tente sa chance. Le mot clef use\_first\_pass indique au module de ne pas redemander de mot de passe mais d'utiliser celui qui lui a été fournit par pam\_tcb. Sufficient indique que si le module LDAP valide ce mot de passe, l'exécution s'arrête là.
* Arrive enfin le module pam\_deny qui récupère la balle et qui ne sait faire qu'une seule chose : échouer.

Introduction

Configuration

Exemples d’utilisation

# Outils réseaux

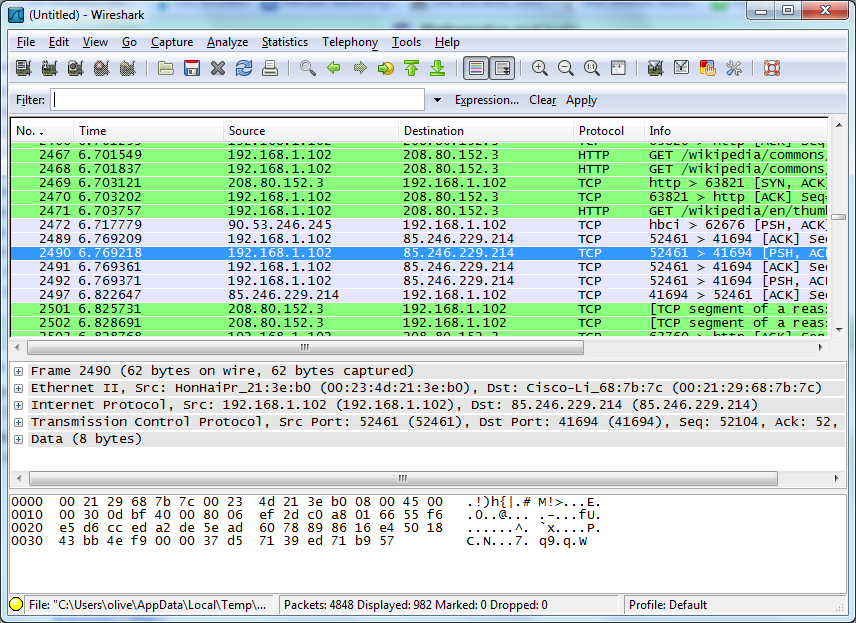
**Inspection de trames avec *wireshark***

wireshark est une application qui permet d'analyser le contenu des trames qui circulent sur un réseau. Une de ses grandes forces lui vient de sa capacité à reconnaître un très grand nombre de protocoles et à les présenter de manière structurée à l'utilisateur.   
On s'en sert généralement pour :

* Observer le contenu de certains paquets spécifiques;
* Observer les conversations entre deux hôtes sur le réseau;
* Visualiser l'ensemble des messages pour un protocole spécifique;
* Dresser des statistiques sur l'utilisation d'un réseau.

La manière la plus simple d'utiliser wireshark est de le lancer sur un hôte: on pourra alors inspecter tout ce qui passe sur l'interface de cet hôte. Depuis la page de départ, sélectionnez dans la section « Interface List » l’interface réseau sur laquelle vous voulez lancer l’inspection.

La fenêtre principale se divise en 3 sections :



Les trois sections de l'interface dépendent l'une de l'autre: chacune contient des informations qui ne peuvent pas être interprétées toutes seules.

* La fenêtre du haut est la liste des paquets, avec leurs informations sommaires réparties dans chacune des colonnes:   
  IP source, IP cible, protocole du niveau supérieur, informations sur le contenu…
* La fenêtre du milieu contient les informations spécifiques au paquet sélectionné, en hiérarchie, chaque niveau d'encapsulation étant précédé d’un « + » et pouvant être déployés pour révéler des informations supplémentaires sur son contenu.
* La fenêtre du bas affiche le contenu brut (en octets) du paquet représenté en hexadécimal à gauche et en ASCII à droite.

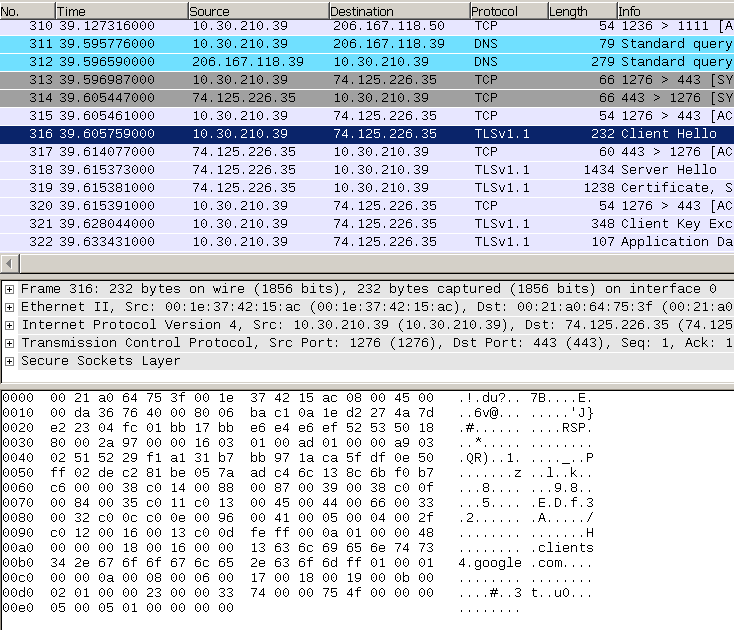
On peut modifier l'apparence de certaines informations via le menu ***Edit -> Preferences***; notamment, la disposition des sections de l'interface, la typographie et les couleurs, mais aussi la résolution des noms au moment de la capture.

**Sauvegarde et chargement de fichier pcap**

**La liste des paquets**

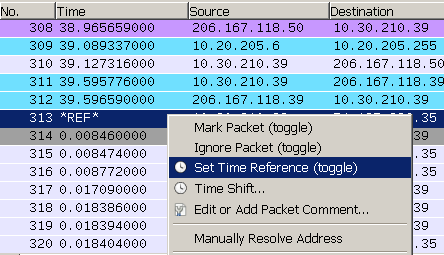


Lorsqu'on sélectionne un paquet dans la fenêtre du haut, son contenu apparaît dans le panneau du milieu et du bas (la section "détails" et "données", respectivement).

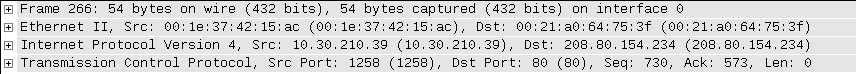


À l'aide du menu contextuel, on peut faire plusieurs choses :

* Marquer un ou plusieurs paquets (CTRL-M), pour ensuite passer de l'un à l'autre (CTRL-SHIFT-B et CTRL-SHIFT-N);
* Définir un paquet comme référence temporelle pour tous les autres paquets qui le suivent



**Les détails du paquet**



Sélection d'une section va la souligner dans le panneau des données

**nmap**

nmap est un balayeur de ports *open source* qui permet l’exploration réseau et obtenir des informations sur le système d’exploitation d’un ordinateur distant. Ce logiciel est de plus en plus utilisé par les administrateurs réseaux car l’audit des résultats de nmap fournit des indications bien précises d’un réseau.

Il est conçu pour…

* détecter les ports ouverts,
* identifier les services hébergés   
  et
* obtenir des informations sur le système d'exploitation d'un ordinateur distant.

Ce logiciel est devenu une référence pour les administrateurs réseaux car l'audit des résultats de *nmap* fournit des indications sur la sécurité d'un réseau.

Pour balayer les ports d'un hôte distant, *nmap* utilise diverses techniques d'analyse qui s'appuient sur des protocoles tels que TCP, IP, UDP ou ICMP.   
De même, il se fonde sur les réponses qu'il obtient à des requêtes particulières pour obtenir une empreinte de la pile IP, souvent propre au système qui l'utilise. C'est par cette méthode qu'il peut reconnaitre la version d'un système d'exploitation ainsi que la version des services (aussi appelés daemons) en écoute.

**Note importante**Puisqu'un balayage de ports peut être considéré comme une tentative d'intrusion ou une attaque potentielle il est préférable de spécifier, lors des balayages l'hôte scanme.nmap.org comme cible.

**Exemples de balayages**

Voici quelques exemples d'utilisation de nmap, du plus simple au un peu plus complexe et ésotérique.

De véritables adresses IP et noms de domaine sont utilisés pour rendre les choses plus concrètes. Vous devez les substituer avec celles de votre propre réseau.  
Bien qu’il est généralement admis que balayer les protocoles d'autres réseaux soit ou devrait être illégal, certains administrateurs de réseau n'apprécient pas les balayages non sollicités de leur réseau et peuvent s'en plaindre.

La meilleure approche est donc d'obtenir d'abord leur autorisation.

Pour des raisons de tests, il est autorisé de balayer l'hôte *scanme.nmap.org*.   
Cette permission inclus seulement les balayages avec Nmap et non pas l'essai d'exploits ou d'attaques de dénis de service.   
Afin de préserver la bande passante, Il est suggéré de ne lancer qu'une douzaine de balayages sur cet hôte au maximum par jour. En cas d'abus de ce libre-service de cible de balayage, il serait fermé et Nmap afficherait le message suivant :   
*Failed to resolve given hostname/IP: scanme.nmap.org*.   
Ces permissions s'appliquent aussi à l'hôte *scanme2.nmap.org*, à *scanme3.nmap.org*, et ainsi de suite, même si ces hôtes n'existent présentement pas.

**Identification du système d’exploitation d’une machine (distante ou non)  
nmap -O 127.0.0.1**

Si nmap n’arrive pas à identifie la version de votre système d’exploitation, il est possible d’afficher une liste des systèmes qui pourraient potentiellement correspondre :  
**nmap -O --osscan-guess 127.0.0.1**

**Balayage générique**Cette option balaye tous les ports réservés TCP sur la machine *scanme.nmap.org* .   
L'option -v active le mode verbeux.  
**nmap -v scanme.nmap.org**

**Détection des hôtes sur un réseau**Si on souhaite identifier toutes les hôtes présents sur un réseau.  
**nmap -sP 192.168.0.0/10**

**Visualisation des ports ouverts sur un hôte**Un balayage de ports permet de voir comment votre machine est vue depuis l’extérieur, quels sont les services qui sont accessibles.   
Par contre un balayage de ports permet un pirate informatique de voir les services qui tournent à distance afin de trouver une faille sur l’un de ces services et comme ça il pourra aller plus loin dans l’attaque.  
**nmap 127.0.0.1**

Pour balayer un port précis. Ici, c’est le port HTTP :  
**nmap –p 80 127.0.0.1**

**Balayage furtif d’un réseau**Lance un balayage furtif (*stealth SYN scan*) contre chaque machine active parmi les 255 machines du réseau de “classe C” sur lequel *scanme* réside.   
Il essaie aussi de déterminer le système d'exploitation sur chaque hôte actif.   
Cette démarche nécessite les privilèges de *root* puisqu'on utilise un *SYN scan* et une détection de SE.  
**nmap -sS -O scanme.nmap.org/24**

**Détermination les démons en fonction sur une partie d’un réseau**Lance une recherche des hôtes et un balayage TCP dans la première moitié de chacun des 255 sous-réseaux à 8 bits dans l'espace d'adressage de classe B 198.116.  
Cela permet de déterminer si les systèmes font tourner *sshd*, DNS, *pop3d*, *imapd* ou le port 4564.   
Pour chacun de ces ports qui sont ouverts, la détection de version est utilisée pour déterminer quelle application est actuellement lancée.  
**nmap -sV -p 22,53,110,143,4564 198.116.0-255.1-127**

**Balayage aléatoire pour un service donné**Demande à nmap de choisir 100 000 hôtes de façon aléatoire et de les balayer dans le but de trouver les serveurs Web (port 80). L'énumération des hôtes est désactivée avec **-P0** puisque envoyer en premier lieu quelques probes pour déterminer si un hôte est actif est inutile lorsque vous ne cherchez à tester qu'un port sur chaque hôte.   
**nmap -v -iR 100000 -P0 -p 80**

**Balayage spécifique pour un service donné**Cette procédure balaye 4 096 adresses IP à la recherche de serveurs Web (sans leur envoyer un *ping* au préalable) et sauvegarde la sortie en format *grepable* et XML.  
**nmap -PN -p80 -oX logs/pb-port80scan.xml -oG logs/pb-port80scan.gnmap 216.163.128.20/20**

# Pare-feu sous Linux

Pare-feu sous Linux

*iptables*

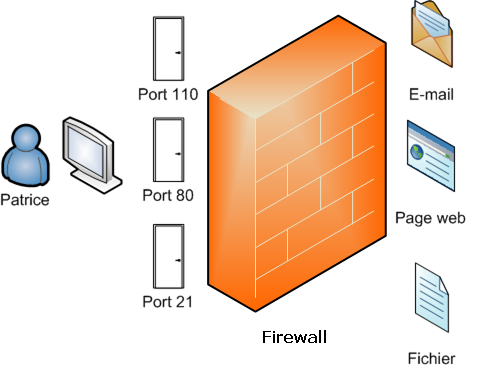
Introduction

Le **pare-feu le plus populaire** utilisé sous Linux est *iptables*. Il propose une **interface en ligne de commande** permettant de configurer *netfilter*.  
Il permet d'établir un certain nombre de **règles pour dire par quels ports on peut se connecter à un hôte** (ordinateur) mais aussi à quels **ports on a le droit de se connecter**.

**Netfilter** est un module du noyau Linux qui **offre la possibilité de contrôler**, **modifier et filtrer les paquets IP**, et de **suivre les connexions**. Il fournit ainsi les **fonctions de pare-feu**, **de partage de connexions internet** et d'**autorisation** d**u trafic réseau**.

iptables est un logiciel libre de l'espace utilisateur Linux grâce auquel l'administrateur système peut configurer les chaînes et règles dans le pare-feu en espace noyau (et qui est composé par des modules Netfilter).

Différents programmes sont utilisés selon le protocole employé : **iptables est utilisé pour le protocole IPv4**, **Ip6tables pour IPv6,** **Arptables pour ARP** (Address Resolution Protocol) ou encore **Ebtables**, spécifique aux **trames Ethernet**.



Par exemple, si on veut empêcher toute connexion FTP (considérant que FTP n'est pas sûr), il est possible de bloquer le port 21 (port TCP utilisé par FTP).  
En général, la technique ne consiste pas à bloquer certains ports mais plutôt à bloquer par défaut **tous** les ports et à en autoriser seulement quelques-uns.

Le but d'un pare-feu est d'empêcher que des programmes puissent communiquer sur le réseau sans autorisation. Aujourd'hui, même sous Windows (depuis Windows XP SP2), un pare-feu est intégré par défaut tant le problème est important.

Avoir un pare-feu ne prémunit pas contre les virus (bien que sous Linux, ils demeurent rares).

En revanche, cela rend la tâche **particulièrement difficile** aux pirates qui voudraient accéder à un hôte visible sur le réseau.

L’objectif de l’installation d’un pare-feu est de bloqu**er par défaut les ports et de n'autoriser seulement ceux dont nous avons besoin**, qui sont considérés comme « sécures » et qui sont utilisés.

Par exemple, le port TCP 80 utilisé pour le web est un port sûr que l’on peut activer.

**Notez**, et cela est important, **qu'il** **y a des portes d'entrée et des portes de sortie sur un** **hôte** (ce ne sont pas nécessairement les mêmes).

*iptables* existe aussi pour *ipv6*, pour cela il suffit d'utiliser la **commande ip6tables** au lieu de *iptables*.

Utilisation  
*iptables* est un programme extrêmement puissant… mais tout aussi complexe.

C’est pourquoi *iptables* n’est **paramétrable seulement en tant que *root***.

Pour la suite des opérations, il est donc **recommandé de basculer en superutilisateur** dès à présent :

**$ sudo su**

**La table *filter***

*iptables* **fonctionne selon un système de tables**, ces **tables sont composées de chaînes**.

Dans le cadre de la configuration et de l'utilisation de *netfilter* comme pare-feu, c'est la **table Filter qui est utile**, elle permet de **filtrer les paquets réseaux**.

Chaque paquet entrant est analysé afin de déterminer notamment sa source et sa destination.

Elle est composée de trois sortes de chaînes…

**INPUT**  
Permet d'**analyser** **les paquets entrants**.   
Si le paquet est adressé au poste, il est confronté au filtre INPUT.

**FORWARD**  
Permet d'**analyser** et d'**autoriser** **les trames à passer d'une interface à une autre**, seulement dans le cadre d'une interface réseau servant de passerelle.

**OUTPUT**  
Permet d'**analyser** **les paquets sortants**.   
Si le paquet sort du poste, il passera par la chaîne OUTPUT.

À cette table, peuvent être affectées des politiques (*policy*) : **DROP**, **LOG**, **ACCEPT**, **SNAT**, **DNAT** et **REJECT**.

**iptables –L ou afficher les règles**

*iptables -L* **permet d’afficher les règles qui régissent actuellement le pare-feu**.

**# iptables -L**

Chain INPUT (policy ACCEPT)  
 target     prot opt source               destination           
 Chain FORWARD (policy ACCEPT)  
 target     prot opt source               destination           
 Chain OUTPUT (policy ACCEPT)  
 target     prot opt source               destination

Nous verrons dans un second document la section FORWARD (la portion NAT ou redirection du trafic).

**Par défaut, les règles sont vides**.   
Il y a trois tableaux mais qui pour le moment ne contiennent aucune ligne.   
Par ailleurs, à chaque fois les **mots policy ACCEPT** sont présents, cela signifie que **tout le trafic est accepté**.

Le **pare-feu est tout simplement inactif et il ne bloque rien**.

S’il y a **déjà des règles inscrites** pour le pare-feu (ce qui ne devrait pas être), il est **possible de les réinitialiser**.  
Ne pas faire que **s’il on est certain de vouloir le faire**. En effet, sur un hôte partagé, il est possible quelqu'il déjà configuré le pare-feu et il serait dommage de saboter tout son travail.

**# iptables –F**Attention… Cette **commande réinitialise toutes les règles *iptables***.

**Le principe des règles**

Voici ce que cela pourrait donner lorsqu'on aura établi des règles… par exemple pour la section INPUT :

**# iptables -L**

Chain INPUT (policy DROP)  
target  prot opt  source destination           
ACCEPT tcp   --   anywhere anywhere tcp dpt:www   
ACCEPT tcp   --   anywhere anywhere tcp dpt:ssh   
ACCEPT tcp   --   anywhere anywhere tcp dpt:pop3

Première chose à savoir : **l'ordre des règles est important**.

En effet, *iptables* les **lit de haut en bas** et la **position de ces règles influe sur le résultat final**.   
Donc, les règles sont numérotées.

Pour afficher les numéros de lignes, ajoutez à la commande le commutateur --line-numbers

**# iptables -L --line-numbers**

Chain INPUT (policy DROP)  
Num  target prot  opt  source       destination           
1 ACCEPT      tcp   --   anywhere              anywhere            tcp dpt:www   
**2 ACCEPT      tcp   --   anywhere           anywhere        tcp dpt:ssh**3  ACCEPT      tcp  --   anywhere              anywhere            tcp dpt:pop3

Ainsi, la règle filtrant SSH est la règle numéro 2.

Chaque ligne correspond à une **règle différente qui permet de filtrer ou non une adresse IP ou un port**.   
Parmi les colonnes intéressantes, on note :

* **target** – **ce que fait la règle**.   
  Ici c'est ACCEPT, c'est-à-dire que cette ligne autorise un port et / ou une adresse IP;
* **prot** – **le protocole utilisé** (tcp, udp, icmp).   
  Il est bien de se rappeler que TCP est celui auquel on a le plus recourt.  
  ICMP quant à lui permet à l’hôte de répondre aux requêtes de type « *ping* »;
* **source** – **IP de source**.   
  Pour INPUT, la source est l'ordinateur distant qui se connecte à l’hôte;
* **destination** – **IP de destination**.   
  Pour OUTPUT, c'est l'ordinateur auquel l’hôte se connecte;
* ***la dernière colonne*** – indique le **port après les deux points** « : ».   
  Ce port est affiché en toutes lettres, mais avec -n vous pouvez obtenir le numéro correspondant.

Sur cet exemple, seuls les ports HTTP, SSH et POP3 (courriel) sont autorisés en entrée.   
Personne ne peut se connecter à la machine par un autre biais.  
En effet, si on regarde bien, par défaut le pare-feu est configuré pour qu'il ignore tous les autres paquets: (policy DROP).

**L’ajout et la suppression des règles**

Voici les principales commandes à connaître.

* **-A chain  
  ajoute une règle en fin de liste** pour la chaine indiquée (INPUT ou OUTPUT, par exemple).
* **-D chain rulenum**  
  **supprime la règle n° rulenum** pour la chaine indiquée.
* **-I chain rulenum**  
  **insère une règle au milieu de la liste** à la position indiquée par *rulenum*.   
  Si **aucune position** rulenum n’est indiquée, la **règle sera insérée en premier**, tout en haut dans la liste.
* **-R chain rulenum  
  remplace la règle n° rulenum** dans la chaine indiquée.
* **-L**  
  **Liste les règles**.
* **-F chain**  
  **vide toutes les règles de la chaine indiquée**. Cela revient à supprimer toutes les règles une par une pour cette chain.
* **-P chain regle**  
  **modifie la règle par défaut** pour la chain… cela permet de dire, par exemple, que par défaut tous les ports sont fermés, sauf ceux que l'on a indiqués dans les règles.

De manière générale, l'ajout d'une règle se passe suivant ce schéma:

iptables -A (chain) -p (protocole) --dport (port) -j (décision)

Remplacez…

* ***chain*** par la section qui vous intéresse (INPUT ou OUTPUT),
* ***protocole*** par le nom du protocole à filtrer (TCP, UDP, ICMP…)   
  et enfin
* ***décision*** par la décision à prendre:   
  ACCEPT pour accepter le paquet,   
  REJECT pour le rejeter   
  ou bien
* DROP pour l'ignorer complètement.

Il existe également les paramètres LOG, SNAT et DNAT.

Le mieux est de découvrir comment on ajoute une règle par une série d'exemples.

**# iptables -A INPUT -p tcp --dport ssh -j ACCEPT**

Cela ajoute à la section INPUT (donc, pour le trafic entrant) une règle sur les données reçues via le protocole TCP sur le port de SSH (vous pouvez remplacer ssh par le numéro du port, soit 22).   
Lorsque votre ordinateur recevra des données en TCP sur le port de SSH, celles-ci seront acceptées ; cela vous permettra donc de vous connecter à distance à votre PC via SSH.

Vous pouvez faire de même avec d'autres ports :

**#** **iptables -A INPUT -p tcp --dport www -j ACCEPT**

… pour le Web (80).

**#** **iptables -A INPUT -p tcp --dport imap4 -j ACCEPT**

… pour les courriels, etc.

Si vous **ne précisez pas de port (en omettant la section dport), tous les ports seront acceptés** !

Autoriser les pings

En plus d'autoriser le trafic sur ces ports, il est intéressant d'autoriser le protocole ICMP (pour pouvoir faire un ping) sur tous ces derniers:

**#** **iptables -A INPUT -p icmp -j ACCEPT**

Comme il n’est pas indiqué de section *--dport*, cette règle s'applique à tous les ports – **mais pour les pings (ICMP) uniquement** !

L’ordinateur répondra alors aux requêtes *ICMP echo* (*ping*) pour indiquer qu'il est bien en vie.

Les règles *iptables* pour INPUT devraient maintenant ressembler à ceci :

**#** **iptables -L**

Chain INPUT (policy ACCEPT)  
target      prot  opt  source  destination           
ACCEPT tcp   --   anywhere              anywhere             tcp dpt:www   
ACCEPT  tcp   --   anywhere              anywhere             tcp dpt:ssh   
ACCEPT tcp   --   anywhere              anywhere             tcp dpt:imap2   
ACCEPT icmp  --   anywhere              anywhere

**Autoriser les connexions locales et déjà ouvertes**

Pour l'instant, nos règles sont encore un peu trop restrictives et pas vraiment utilisables   
(vous risquez de ne plus pouvoir faire grand-chose).

Je vous propose d’ajouter deux règles pour « assouplir » un peu votre pare-feu et le rendre enfin utilisable.

**#** **iptables -A INPUT -i lo -j ACCEPT**

**#** **iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT**

Ces deux règles utilisent des options un peu différentes. Voici quelques explications.

1. La première règle autorise tout le trafic sur l'interface de loopback locale grâce à -i lo. Il n'y a pas de risque à autoriser votre ordinateur à communiquer avec lui-même, d’autant plus qu’il en a parfois besoin.
2. La seconde règle autorise toutes les connexions qui sont déjà à l'état ESTABLISHED ou RELATED.   
   En clair, elle autorise toutes les connexions qui ont été demandées par votre PC.   
   Là encore, cela permet d'assouplir le pare-feu et de le rendre fonctionnel pour une utilisation quotidienne.

**Refuser toutes les autres connexions par défaut**

Il reste un point essentiel à traiter car, **pour l'instant, ce filtrage ne sert à rien**. En effet, nous avons indiqué quelles données nous autorisions, mais **nous n'avons pas dit que toutes les autres devaient être refusées** !

Changez donc la **règle par défaut pour DROP** par exemple :

**#** **iptables -P INPUT DROP**

*iptables* devrait maintenant indiquer que par défaut tout est refusé, sauf ce qui est indiqué par les lignes dans le tableau :

**#** **iptables -L**

Chain INPUT (policy DROP)  
target      prot  opt  source destination           
ACCEPT      tcp   --   anywhere  anywhere            tcp dpt:www  
ACCEPT      tcp   --   anywhere anywhere            tcp dpt:ssh   
ACCEPT      tcp   --   anywhere  anywhere            tcp dpt:imap2   
ACCEPT      icmp  --   anywhere  anywhere

Le **filtrage est radical**. Nous n'avons pas autorisé beaucoup de ports et il se pourrait que vous vous rendiez compte que **certaines applications n'arrivent plus à accéder à l'internet** (normal, leur port doit être filtré).

À vous de savoir quels ports ces applications utilisent pour modifier les règles en conséquence.  
Au besoin, pensez à faire de même pour les règles de sortie (OUTPUT).

Annexe A  
Gérer ses règles iptables. Proprement.

On voit régulièrement des administrateurs sauvegarder leurs règles iptables dans un script init. Bien que cela fonctionne, ce n'est pas la manière la plus élégante.

Un alternative est la sauvegarde des règles existantes à l'aide de la commande iptables-save.

Cette commande renvoie le résultat sur la sortie standard, redirigeant donc la sortie vers un fichier.

**#** **iptables-save > /etc/iptables**

Une option intéressante…  
le -c, qui permet d'afficher combien de fois la règle a été appliquée (en bytes et en paquets).

Pour charger les règles, il suffit d'utiliser la commande iptables-restore en fournissant le fichier de sauvegarde en entrée.

**#** **iptables** **-restore < /etc/iptables**

Ajoutez le -c pour prendre en compte les compteurs (s'ils ont été sauvegardés). Si jamais des règles sont déjà chargées, vous n'avez même pas besoin de purger (flusher), la commande le fait pour vous par défaut (-n pour les garder).

Pour restaurer automatiquement les règles au démarrage, on pourrait créer un petit script init.

Il est préférable de directement lancer la restauration avant que l'interface ne soit active (up). Cela permet d'éviter tout oubli lorsquel’on efface les règles puis l’activation des interfaces.

De plus, on gagne ainsi la possibilité de créer des règles personnalisées pour chaque interface.

Si les règles sont indépendantes des interfaces, on peut toujours toujours lancer la restauration en avant l’activation de la boucle locale.

Fichier */etc/network/interfaces*

**iface eth0 inet static**

**pre-up iptables-restore < /etc/iptables**

**address 543.454.233.42**

**netmask 255.255.255.0**

**gateway 543.454.233.254**

**auto eth0**

# *iptables*

**Netfilter et Iptables**

Le pare-feu dans un environnement Linux est composé de plusieurs modules implémentés en son noyau et l’ensemble est nommé Netfilter. Son rôle est de filtrer et /ou transformer les paquets qui circulent dans le trafic entrant et sortant de la machine. Il utilise différents outils de configuration intégrés selon le protocole employé. Étant donné le rôle central que joue le protocole IPv4 dans les communications réseaux, l’outil IPtables est souvent qualifié comme étant lui-même le pare-feu Linux. Dans les faits, c’est une interface « ligne de commandes » qui permet à l’utilisateur de configurer Netfilter et sert de commande à tout faire.

Tous les paquets testés passent à travers des tables. Chacune d’entre elles agit comme une file d’attente et est dédiée à un type de traitement des paquets. Aux tables sont associé ou non cinq chaînes de base prédéfinies : PREROUTING, INPUT, FORWARD, OUTPUT et POSTROUTING. Chaque chaîne contiendra à son tour une cible (target) qui spécifie une action à effectuer sur le paquet comme par exemple ACCEPT, DROP, REJECT, LOG, DNAT, SNAT ou MASQUERADE. Ensuite elle peut contenir ou non des règles définies par des options de correspondances ou d’opérations constituant ainsi un filtre spécifique. L’ordre des règles est important puisque chaque paquet traverse systématiquement les chaînes examinant si une règle lui correspond ou non. Si c’est le cas la règle sera appliquée et le traitement arrêtera. Sinon le paquet passera à la suivante. Si aucune règle n’a arrêté le paquet la cible par défaut de la chaîne sera appliquée. L’administrateur peut créer autant de chaînes et de règles qu’il le désire. IPtables permet de consulter ou modifier les tables, les chaînes et les règles selon nos besoins.

Netfilter est constitué de 4 tables qui lui permettent de remplir ses fonctions.

1. La première table (MANGLE) par laquelle passe le paquet permet de lui apporter des modifications, leur appliquant ainsi un traitement spécial. Ceci peut influencer d’autres règles contenues dans les tables NAT et de filtrage. Elle contient les cinq chaînes prédéfinies.
2. La deuxième table est celle qui gère la translation d’adresse réseau (NAT). Elle permet de configurer la gestion de l’adressage lorsqu’on désire utiliser des adresses privées au sein du réseau interne. Or pour router les paquets de données sur un réseau externe, ceux-ci doivent être dotés d’une adresse publique. Le nating permet donc d’associer une adresse publique à une adresse privé. Elle contient les chaînes suivantes :  
   … chaîne PREROUTING (pré-routage) : principalement utilisé pour le DNAT (destination NAT ) dans lequel on change l'adresse de destination des paquets provenant du réseau public en une adresse privée du réseau local.   
   … chaîne POSTROUTING (post-routage) : utilisé pour le SNAT (source NAT) et qui consiste à changer une adresse privée source d’un paquet sortant avec une adresse source publique.   
   … chaîne OUTPUT : utilisé pour traduire les adresses des paquets générés par le pare-feu.
3. La table de filtrage (FILTER) sert principalement de fonctions de pare-feu. Elle spécifie les règles de trafic en indiquant par quels ports de la machine le paquet peut transiter. Elle est la table par défaut si la chaine n’est associée à aucune table. Elle a trois chaînes prédéfinies :  
   … chaîne FORWARD : filtre et transfère les paquets destinés à une machine derrière le pare-feu. Donc qui le traversent.   
   … chaîne INPUT : filtre les paquets destinés au pare-feu.  
   … chaîne OUTPUT : filtre les paquets qui ont pour origine le pare-feu.
4. La table (RAW) est principalement utilisée pour marquer les paquets qui doivent ou non être vérifiés par le système de traçage de connexion. Celui-ci permet de connaître l’état d’une connexion spécifique et ainsi de contrôler plus facilement qui ou quoi est autorisé à démarrer de nouvelles sessions.
5. L’outil de configuration de pare-feu IPtables est sans doute le plus célèbre de l’environnement Linux. Il est largement supporté par la communauté qui vante sa souplesse et sa simplicité de configuration. Les différentes combinaisons qu’il permet de réaliser font de lui un des outils les plus puissants du marché.

Configuration du pare-feu

Dans ce document, nous allons configurer notre pare-feu de la manière suivante

* On bloque tout le trafic entrant par défaut.
* On autorise au cas par cas:   
  le trafic appartenant ou lié à des connexions déjà établies   
  et   
  le trafic à destination des serveurs (web, ssh, etc.) que nous souhaitons mettre à disposition.

Pour voir les règles exitantes

**$ sudo su**

**$ iptables -L**

En entrant la commande iptables -L une liste de vos règles actuelles est affichée.

Si vous (ou un logiciel) n'avez encore jamais touché à *iptables*, les chaines sont vides, et vous devriez voir ce qui suit…

**Chain INPUT (policy ACCEPT)**

**target prot opt source destination**

**Chain FORWARD (policy ACCEPT)**

**target prot opt source destination**

**Chain OUTPUT (policy ACCEPT)**

**target prot opt source destination**

Pour l'instant, tout passe dans toutes les directions (*policy ACCEPT*).

Pour cette configuration basique, seul le trafic entrant (chaine input) nous intéresse.

Par défaut, iptables -L n'affiche que la table « filter ».

Pour consulter les autres tables, vous devez ajouter l'option -t suivie de « nat », « mangle » ou « raw ».

Pour la configuration d'un pare-feu la table "filter" est toutefois la seule nécessaire.

Attention…

Si vous avez modifié la règle par défaut pour le blocage (iptables -P INPUT DROP voir plus bas) et que vous entrez iptables -F vous bloquerez tous les accès … y compris celui en cours.

Ceci est particulièrement problématique sur un hôte sur laquelle vous accédez à distance.

Si vous avez déjà modifié la configuration et que vous voulez la réinitialiser, entrez…

**$ iptables –F**

**$ iptables –X**

**Autoriser le trafic entrant d'une connexion déjà établie**

Pour permettre à une connexion déjà ouverte de recevoir du trafic..

**$** **iptables -A INPUT -m conntrack --ctstate ESTABLISHED -j ACCEPT**

Le module conntrack permet de paramètrer dans quel état se trouve une connexion.

Pour permettre le trafic entrant sur un port spécifique…

Pour permettre le trafic entrant sur le port 22 (traditionnellement utilisé par SSH, vous devrez indiquer à *iptables* tout le trafic TCP sur le port 22 de votre interface réseau.

**$** **iptables -A INPUT -p tcp -i eth0 --dport ssh -j ACCEPT**

Cette commande ajoute une règle (-A) à la chaine contrôlant le trafic entrant INPUT, pour autoriser le trafic (-j ACCEPT), vers l'interface (-i) eth0 et à destination du port (--dport) SSH (on aurait pu mettre 22).

Maintenant vous pouvez vérifier vos règles iptables :

**$ iptables –L**

**Chain INPUT (policy ACCEPT)**

**target prot opt source destination**

**ACCEPT all -- anywhere anywhere state RELATED,ESTABLISHED**

**ACCEPT tcp -- anywhere anywhere tcp dpt:ssh**

Maintenant, acceptons tout le trafic web (www) entrant…

**$** **iptables -A INPUT -p tcp -i eth0 --dport 80 -j ACCEPT**

En regardant nos règles, nous avons…

**$ iptables -L**

**Chain INPUT (policy ACCEPT)**

**target prot opt source destination**

**ACCEPT all -- anywhere anywhere state RELATED,ESTABLISHED**

**ACCEPT tcp -- anywhere anywhere tcp dpt:ssh**

**ACCEPT tcp -- anywhere anywhere tcp dpt:www**

Nous avons exceptionnellement autorisé le trafic tcp pour ssh et les ports web mais, comme nous n'avons rien bloqué, tout le trafic passe quand même.

**Bloquer le trafic**

Maintenant que nous avons fini avec les autorisations, il faut maintenant bloquer le reste. Nous allons en fait modifier la politique par défaut (policy) de la chaine INPUT.

Cette décision (DROP) s'applique lorsqu'aucune règle n'a été appliquée à un paquet.

Donc, si la tentative de connexion n'est permise par aucune des règles précédentes, elle sera rejetée.

**$** **iptables -P INPUT DROP # Attention… À ne pas utiliser sur un serveur distant !**

**$** **iptables -L**

**Chain INPUT (policy DROP)**

**target prot opt source destination**

**ACCEPT all -- anywhere anywhere state RELATED,ESTABLISHED**

**ACCEPT tcp -- anywhere anywhere tcp dpt:ssh**

**ACCEPT tcp -- anywhere anywhere tcp dpt:www**

Un autre moyen de procéder est l'ajout en fin de chaine d'une règle supprimant les paquets (les paquets autorisés par les règles précédentes n'atteindraient pas celle-ci), à partir de la commande iptables -A INPUT -j DROP.  
Il faudrait alors faire attention à la position des futures règles.

**Autoriser le trafic local**

Un p'tit problème de notre configuration est que même l'interface locale (*loopback*) est bloquée.

Nous pourrions avoir écrit les règles de rejet seulement pour *eth0* en spécifiant -i eth0 mais nous pouvons aussi ajouter une règle pour *loopback*.

Par exemple, nous pourrions l'insérer en seconde position…

**$** **iptables -I INPUT 2 -i lo -j ACCEPT**

Pour lister les règles plus en détail…

**$** **iptables -L -v -n**

**Autoriser les requêtes ICMP (*ping*)**

Il peut être utile de valider les réponses aux requêtes *ping*, ne serait-ce que pour s'assurer que le poste est toujours en activité.

**$** **iptables -A OUTPUT -p icmp -m conntrack --ctstate NEW,ESTABLISHED,RELATED -j ACCEPT**

On autorise les pings

**$** **iptables -A INPUT -p icmp -j ACCEPT**

**Supprimer une règle**

Si vous vous êtes trompé dans la création d'une règle et que cela vous bloque une connexion, vous pouvez supprimer une seule entrée plutôt que de tout réinitialiser.

Tout d'abord vous listez l'ensemble de vos régles avec l'affichage des lignes :

**$** **iptables -L --line-numbers**

Voici un exemple de ce que cela peut retourner …

**Chain INPUT (policy DROP)**

**num target prot opt source destination**

**1 DROP icmp -- anywhere anywhere**

**2 ACCEPT tcp -- anywhere anywhere tcp dpt:ssh**

**3 ACCEPT tcp -- anywhere anywhere tcp dpt:www**

**4 ACCEPT tcp -- anywhere anywhere tcp dpt:webmin**

**Chain FORWARD (policy ACCEPT)**

**num target prot opt source destination**

**Chain OUTPUT (policy ACCEPT)**

**num target prot opt source destination**

**1 ACCEPT tcp -- anywhere anywhere tcp spt:www**

**2 ACCEPT tcp -- anywhere anywhere tcp spt:12345**

Je souhaite supprimer la ligne 2 de la chaîne OUTPUT

**$** iptables -D chaine numéro\_de\_ligne

**$** **iptables -D OUTPUT 2**

**Sauvegarder vos règles**

**$** **iptables-save -c**

**Appliquer les règles au démarrage**

Vous avez testé vos règles alors il reste à les appliquer au démarrage.

**$** **service iptables-persistent**

Note  
Si le service n’est pas installé il faut le faire à l’aide de la commande apt-get install iptables-persistent.

Le service *iptables-persistent* prend les arguments…

* save pour sauvegarder les règles,
* flush pour vider toutes les règles   
  et
* reload pour les recharger depuis les fichiers précités.

**Annexe A  
Script iptables**

Ce script est un exemple, il est à adapter selon ses besoins.

Il peut toutefois être utilisé pour une utilisation courante, il offre une plutôt bonne "protection".

Attention…

iptables -A INPUT -m pkttype –pkt-type broadcast -j DROP peut poser problème au DHCP "le broadcast : pour trouver et dialoguer avec un serveur DHCP, la machine va simplement émettre un paquet spécial de broadcast (broadcast sur 255.255.255.255 avec d'autres informations comme le type de requête, les ports de connexion…) sur le réseau local.

Lorsque le serveur DHCP recevra le paquet de broadcast, il renverra un autre paquet de broadcast (n'oubliez pas que le client n'a pas forcement son adresse IP et que donc il n'est pas joignable directement) contenant toutes les informations requises pour le client" cette commande peut donc empêcher le pc de recevoir son ip par le DHCP

iptables.sh

**#!/bin/bash  
IPTS=iptables**

**# Vidage des tables et des règles personnelles  
$IPTS -t filter –F  
$IPTS -t filter -X**

**# Interdire toutes connexions entrantes et sortantes  
$IPTS -t filter -P INPUT DROP  
$IPTS -t filter -P FORWARD DROP  
$IPTS -t filter -P OUTPUT DROP**

**# Traitement interface local  
$IPTS -A INPUT -i lo -j ACCEPT  
$IPTS -A OUTPUT -o lo -j ACCEPT**

**# Ne pas casser les connexions établies  
$IPTS -A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT  
$IPTS -A OUTPUT -m state --state RELATED,ESTABLISHED -j ACCEPT**

**# Authoriser HTTP et HTTPS  
$IPTS -t filter -A OUTPUT -p tcp --dport 80 -j ACCEPT  
$IPTS -t filter -A OUTPUT -p tcp --dport 443 -j ACCEPT  
$IPTS -t filter -A INPUT -p tcp --dport 80 -j ACCEPT  
$IPTS -t filter -A INPUT -p tcp --dport 443 -j ACCEPT**

**# On rejette les balayages (scans) XMAS et NULL.  
$IPTS -A INPUT -p tcp --tcp-flags FIN,URG,PSH FIN,URG,PSH -j DROP  
$IPTS -A INPUT -p tcp --tcp-flags ALL ALL -j DROP  
$IPTS -A INPUT -p tcp --tcp-flags ALL NONE -j DROP  
$IPTS -A INPUT -p tcp --tcp-flags SYN,RST SYN,RST -j DROP**

**# Courriels SMTP POP3S (Modifier les ports en fonction de ses besoin)  
$IPTS -t filter -A INPUT -p tcp --dport 465 -j ACCEPT  
$IPTS -t filter -A OUTPUT -p tcp --dport 465 -j ACCEPT  
$IPTS -t filter -A INPUT -p tcp --dport 993 -j ACCEPT  
$IPTS -t filter -A OUTPUT -p tcp --dport 993 -j ACCEPT**

**# Rejetter silencieusement tous les paquets de diffusion générale (broadcast)**

**# Attention: cfr. message ci-dessus)  
$IPTS -A INPUT -m pkttype --pkt-type broadcast -j DROP**

**# On journalise les paquets en entrée.  
$IPTS -A INPUT -j LOG**

**# On journalise les paquets forward.  
$IPTS -A FORWARD -j LOG   
exit 0**

# *pfsense*

|  |  |
| --- | --- |
| http://www.generation-linux.fr/public/dec09/pfsense-logo.jpg | pfSense est une distribution logicielle permettant de réaliser une passerelle réseau à partir d'un serveur x86. Il date de 2004 à partir d’un fork de m0n0wall par Chris Buechler et Scott Ullrich. |

Très fréquemment rencontré dans les PME et les petites structures, pfSense offre une solution complète de routage, filtrage, VPN et partage de connexion. Il est basé sur pf, et intègre un grand nombre de composants tiers : serveur DHCP/DNS, serveur de temps, proxy web, monitoring... La configuration se fait entièrement via une interface web.

pfSense est disponible sous licence BSD. Un support officiel est proposé par la société BSD Perimeter.

pfSense peut être installé sur un simple ordinateur personnel comme sur un serveur. Basé sur PF (packet filter), comme iptables sur GNU/Linux, il est réputé pour sa fiabilité.

Après une installation en mode console, il s'administre ensuite simplement depuis une interface web et gère nativement les VLAN (802.1q).

**Présentation**

*pfSense* ne fait pas seulement un pare-feu, il offre également toute une panoplie de services réseaux.

* Pare-feu  
  Le pare-feu est celui de *FreeBSD*, à savoir *PacketFilter*.
* Table d'état  
  La table d'état (*State table*) contient les informations sur les connexions réseaux.   
  Cela permet d'avoir un aperçu des connexions et surtout de créer des règles comme par exemple sur le nombre de connexion maximum pour un hôte.
* Translation d'adresses réseaux (NAT)  
  Permet de joindre un réseau privé vers l'extérieur.
* VPN  
  Permet la création de VPN IPsec, OpenVPN ou PPTP.
* Serveur DHCP.
* Serveur DNS et DNS dynamiques.
* Portail Captif.
* Redondance et équilibrage de charge.
* Graphes pour la charge système et réseaux.

Des logiciels complémentaires s'installent grâce à un système d’ajout. Ces derniers sont configurés pour s'intégrer à l'interface web. Comme exemple d’ajout on trouve *nut* (pour le monitoring des onduleurs) et FreeRADIUS.

Le nombre d’ajout augmente régulièrement.

Il est aussi possible d'installer d'autres paquets que ceux proposés par l'interface, en ligne de commande.

La version 2.0 finale de *pfSense* est arrivée fin décembre 2011.

**Installation du service**

pfSense est un pare-feu à état (*statefull firewall*) basé sur FreeBSD, gratuit et à code ouvert comme pare-feu ou aiguilleur (*router*). En plus de sa puissance et flexibilité, la distribution inclut beaucoup de fonctionnalités.

**Mise en place de la machine virtuelle**

Lors de la création de la machine virtuelle…

Inscrire *pfsense* comme nom de la machine virtuelle et Virtalbox fera le reste.

Accepter les paramètres par défaut.

Avant de démarrer la machine virtuelle une première fois...

Modifier les paramètres suivants :

* Interface 1 (carte 1)   
  Mode d'accès réseau = Accès par pont (cela sera notre interface du réseau interne (LAN)).
* Interface 2 (carte 2)   
  Mode d'accès réseau = Accès par point (cela sera notre interface du réseau externe (WAN)).

Note:

Ne pas oublier d'insérer l'image iso dans le lecteur CD/DVD.

**Mise en place de *pfsense***

Démarrer la machine virtuelle.

Note:

Les touches de déplacement et la touche TAB permettent de se déplacer entre les éléments)

* Sélectionner I pour installer le logiciel

À l'aide de l'interface texte

* Sélectionner les paramètres standards (*Accept these settings*)
* Sélectionner l'installation (*Install pfsense*)
* Sélectionner le disque par défaut
* Formater le disque
* Sélectionner les paramètres proposés (*Use this Geometry*)
* Confirmer le formatage
* Sélectionner le partitionnement du disque (*Skip this step*)
* Sélectionner le gestionnaire de démarrage (*Skip this step*)
* Sélectionner la partition 1: ...
* Confirmer (*OK*)
* Sélectionner les sous-partition (*Accept and Create*)
* Sélectionner le type de noyau (*Symetric multiprocessing kernel*)
* Redémarrer

Avant le redémarrage

Retirer de la machine virtuelle l'image iso du lecteur cd/dvd.

Au redémarrage

* Entrer n (pour non) à la demande de la configuration VLAN
* Sélectionner la première interface *em0* comme interface interne (Réseau interne (LAN))  
  Nommer l'interface em0
* Sélectionner le second interface *em1* comme interface externe (Réseau externe (WAN))  
  Laisser le nom de l'interface vide  
  *pfsense* confirme vos deux interfaces
* Entrer *y* (pour oui) pour confirmer

**Utilisation de pfSense**

À l'aide d'une seconde machine virtuelle

* Modifier les paramètres de la machine virtuelle:  
  Mode d'accès réseau = Accès par point (cela sera notre interface du réseau local (LAN)).

Démarrer un fureteur et entrer l’URL suivante  
<http://192.168.1.1> (adresse de l’interface interne par défaut)

Pour se connecter à l'interface graphique

Nom d'usager: *admin*

Mot de passe: *pfsense*

**Personnalisation de *pfsense***

Sélectionner *System – General setup*

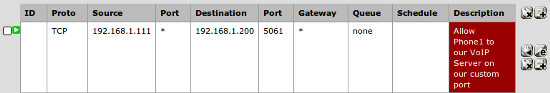
Modifier les paramètres si désiré

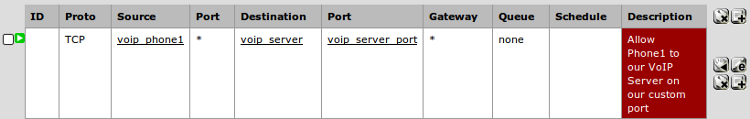
**Principales configurations**Outils offert sous le menu *firewall*

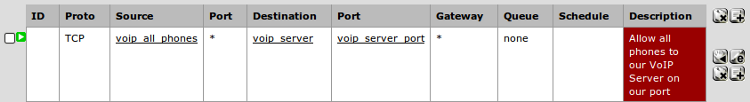
**Création d’un alias**

Dans *pfsense* un alias est utilisé pour associer ou regrouper des éléments sous un nom commun. Ce nom est utilisable dans certaines zones, lors de la création de règles par exemple. Un alias (ou “groupe”) peut être constitués de ports, d'adresses IP ou de réseaux.

Ces éléments, de différentes natures, ne peuvent pas être mélangés au sein d'un même alias.

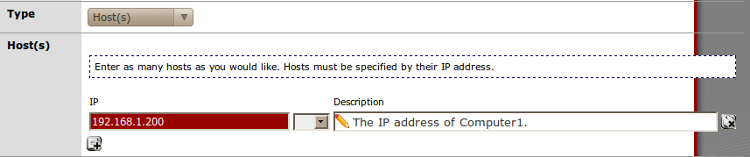




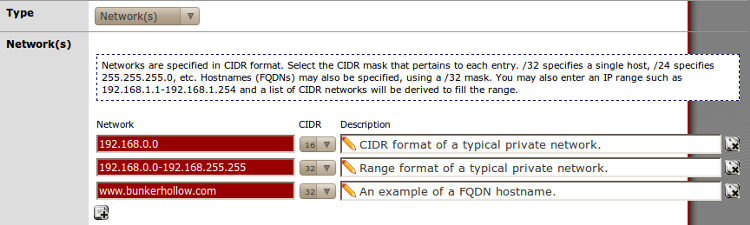


**Création d’un alias pour un hôte**

Hôte ayant le nom de computer1



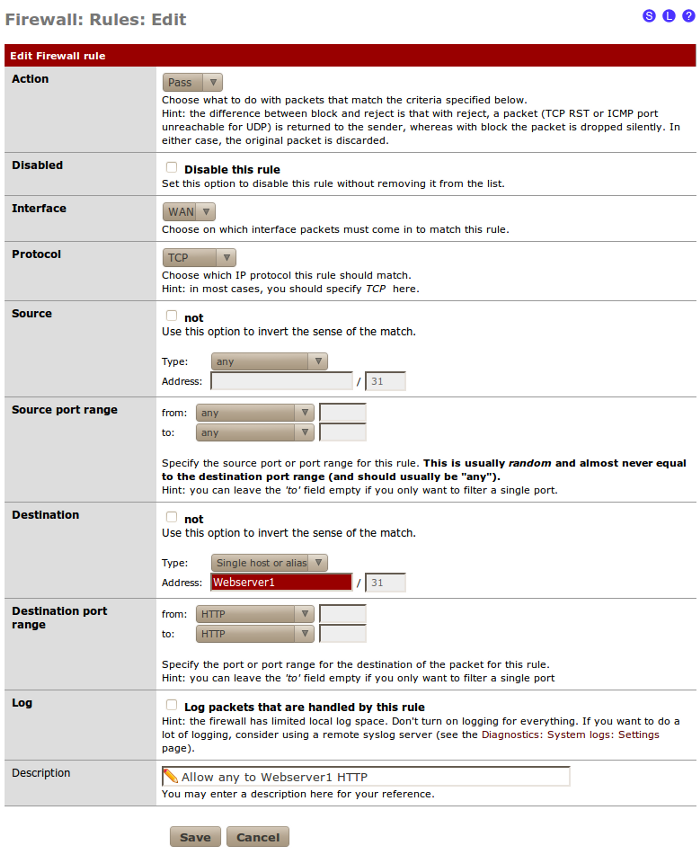
**Création d’un alias de réseau ou de groupe de réseau**



**Création d’un alias de port ou de groupe de ports**



**Création d’une règle de pare feu**



**Ordre des règles du pare-feu**

Les règles avec pfsense sont toujours évaluées de haut en bas.

La première règle qui correspond est exécutée et le reste des règles sont ignorées.

De manière générale, il est de bon usage de placer les règles les plus restrictives en premier et les règles les plus génériques en dernier.

Pour réorganiser l’ordre des règles, sélectionner une règle et cliquez sur le bouton de déplacement appropriés.

**Annexe**

Télécharger l’image pfsense 1.2 sur http://pfsense.com.

Pour se connecter à l'interface graphique

Nom d'usager: admin

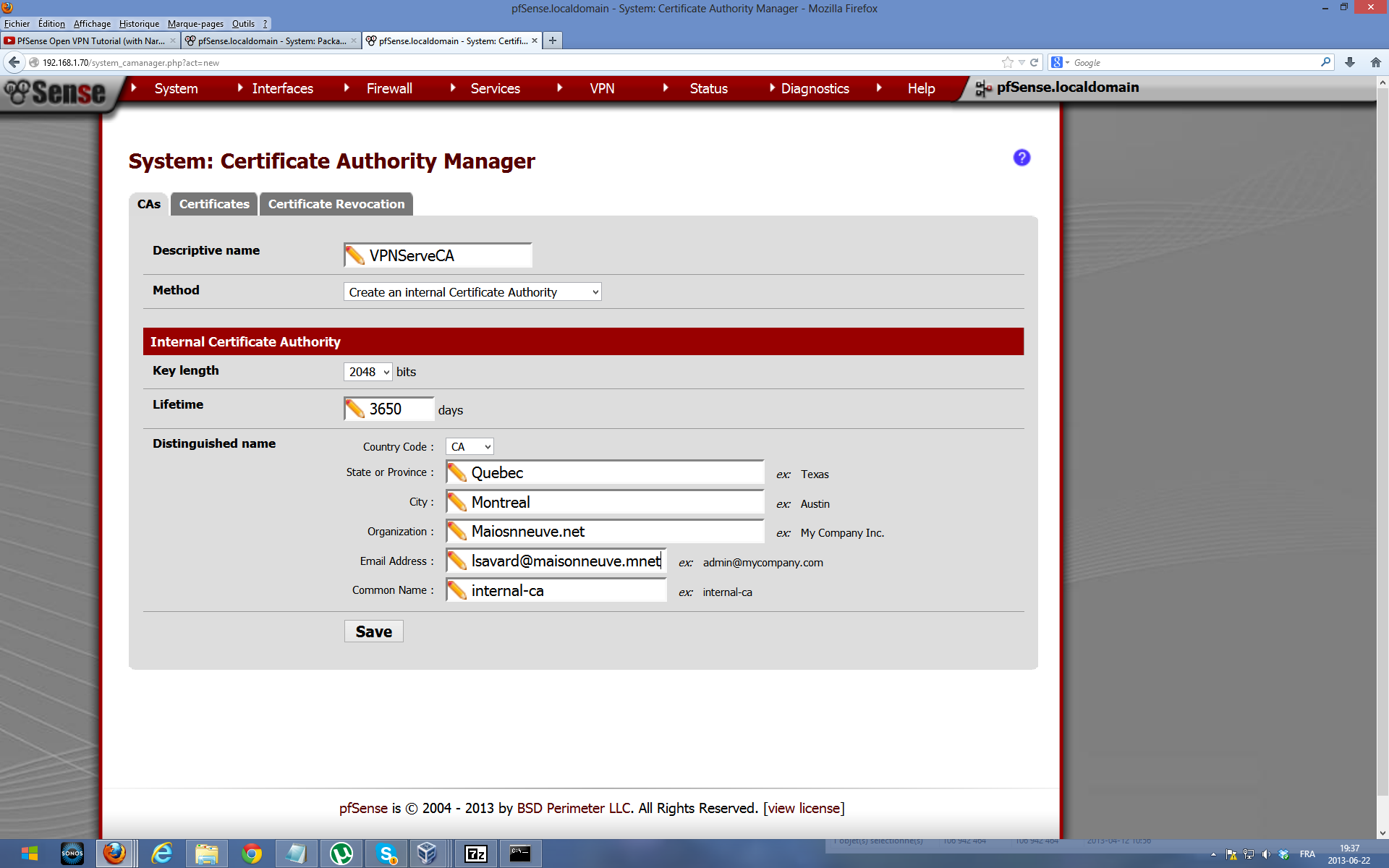
Mot de passe: pfsense

**Installation du client VPN**

System – Packages  
Available packages  
OpenVPN Client Export Utility

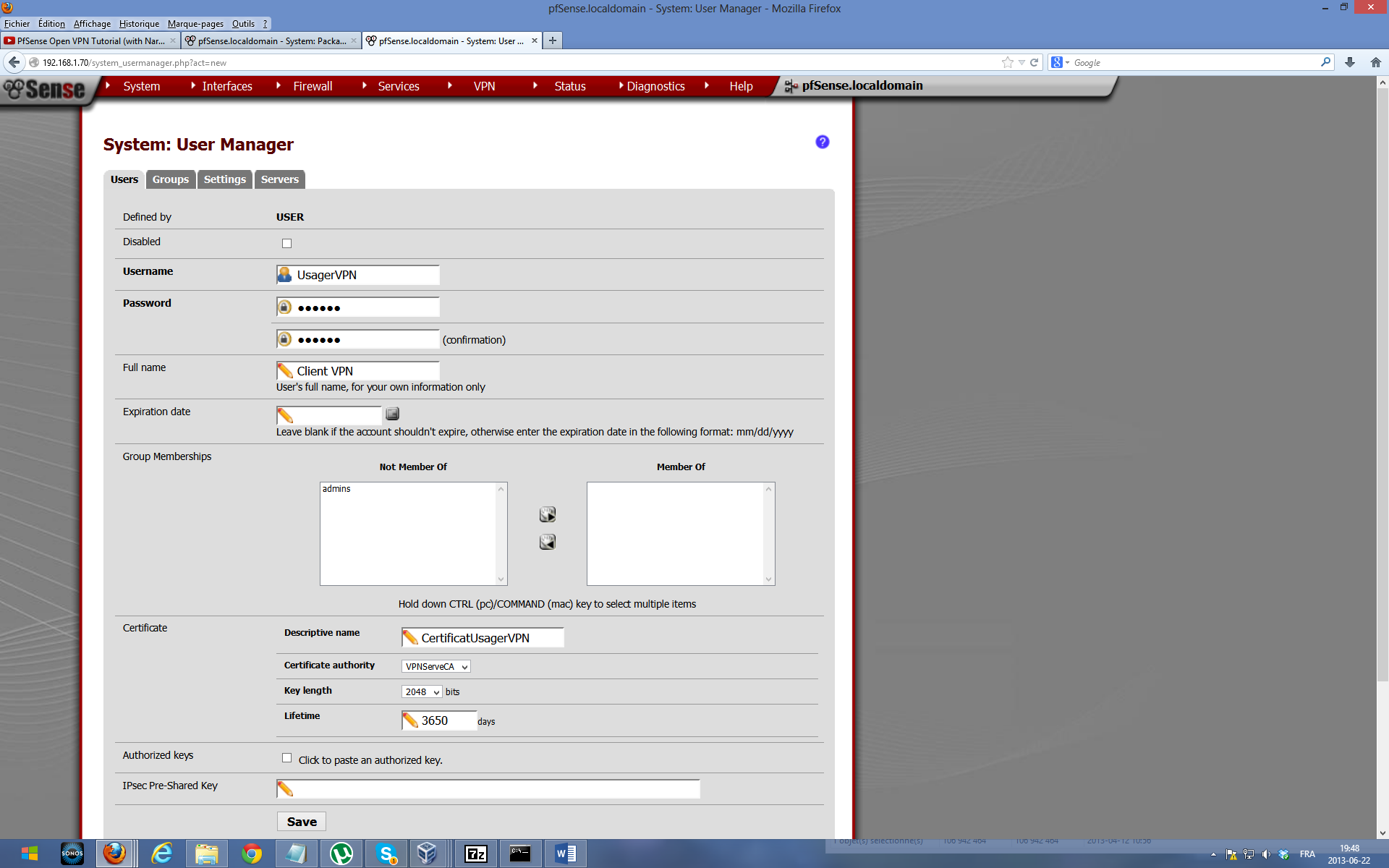
**Installation de l’autorité de certification**

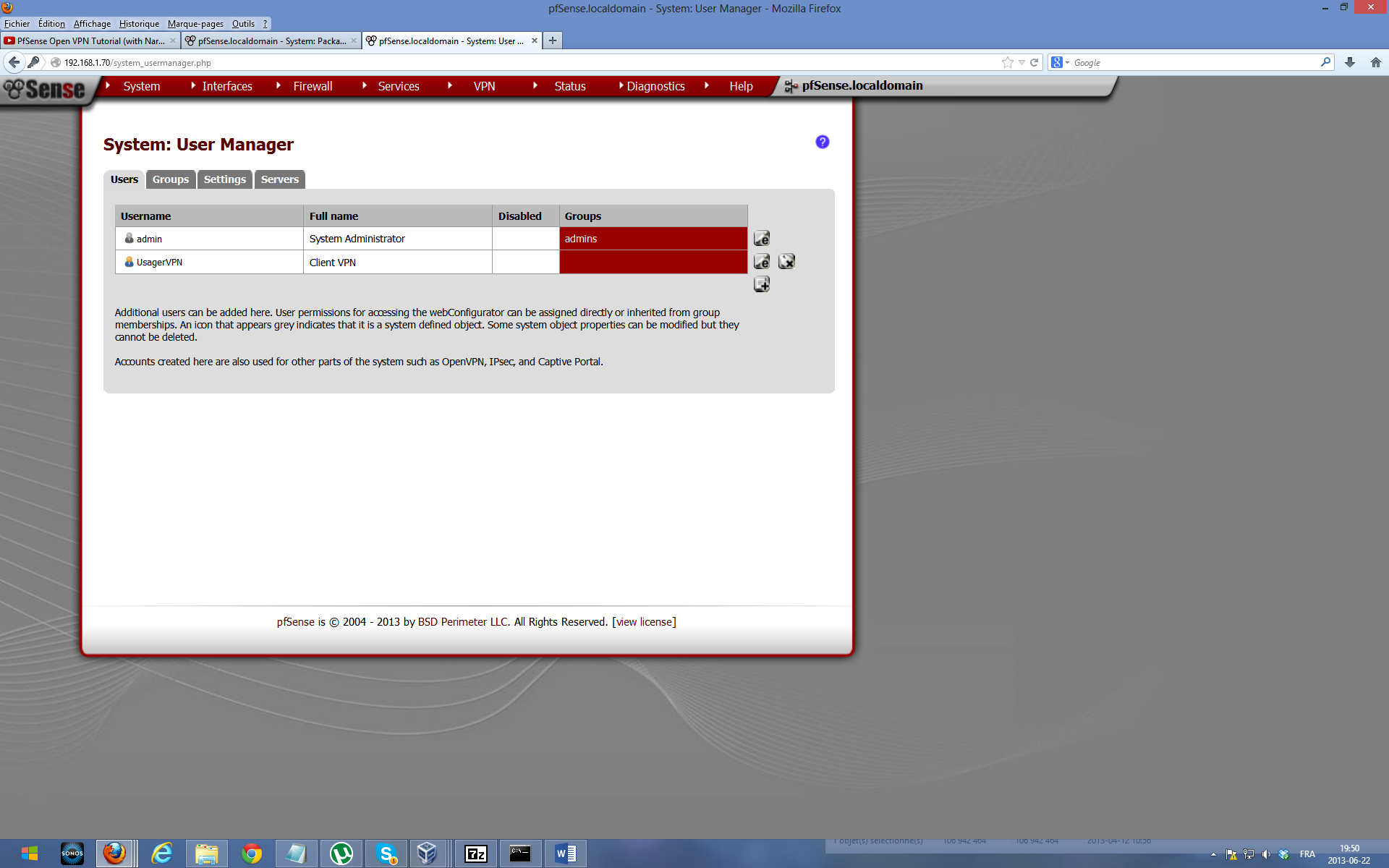
System – CertManager



**Création d’un usager**

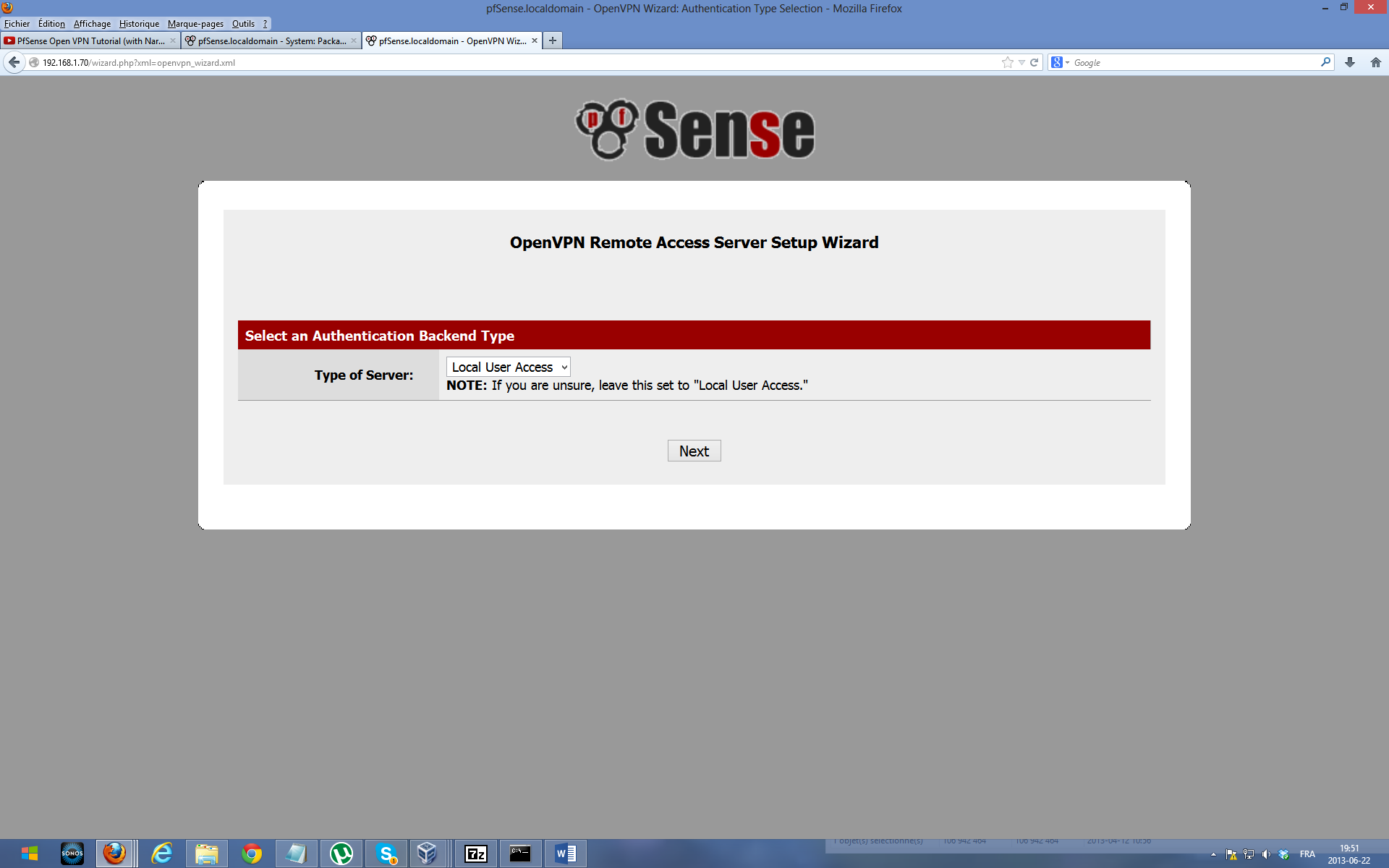
System – User Manager

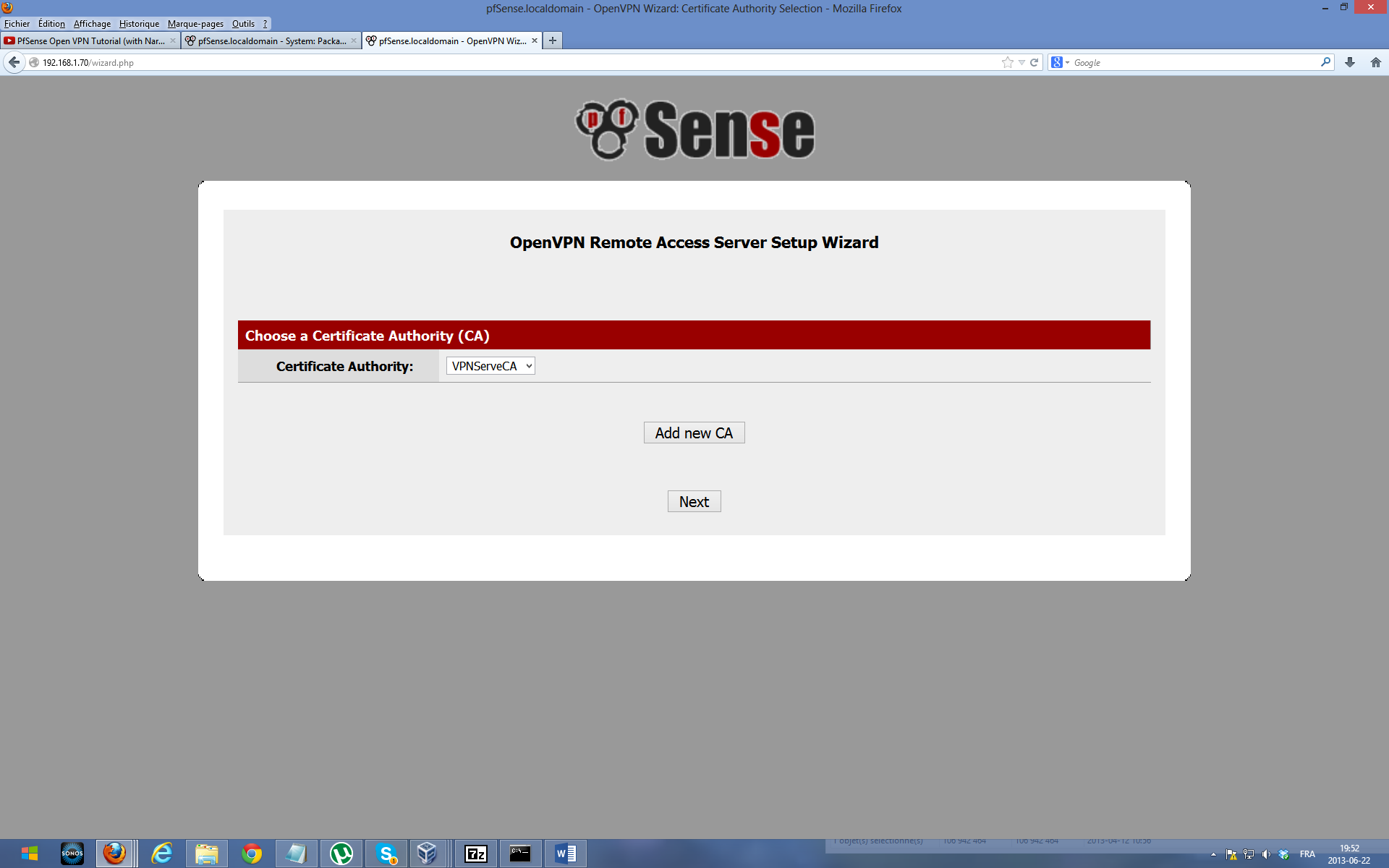




Mise en place du VPN

VPN – OpenVPN… Wizzard



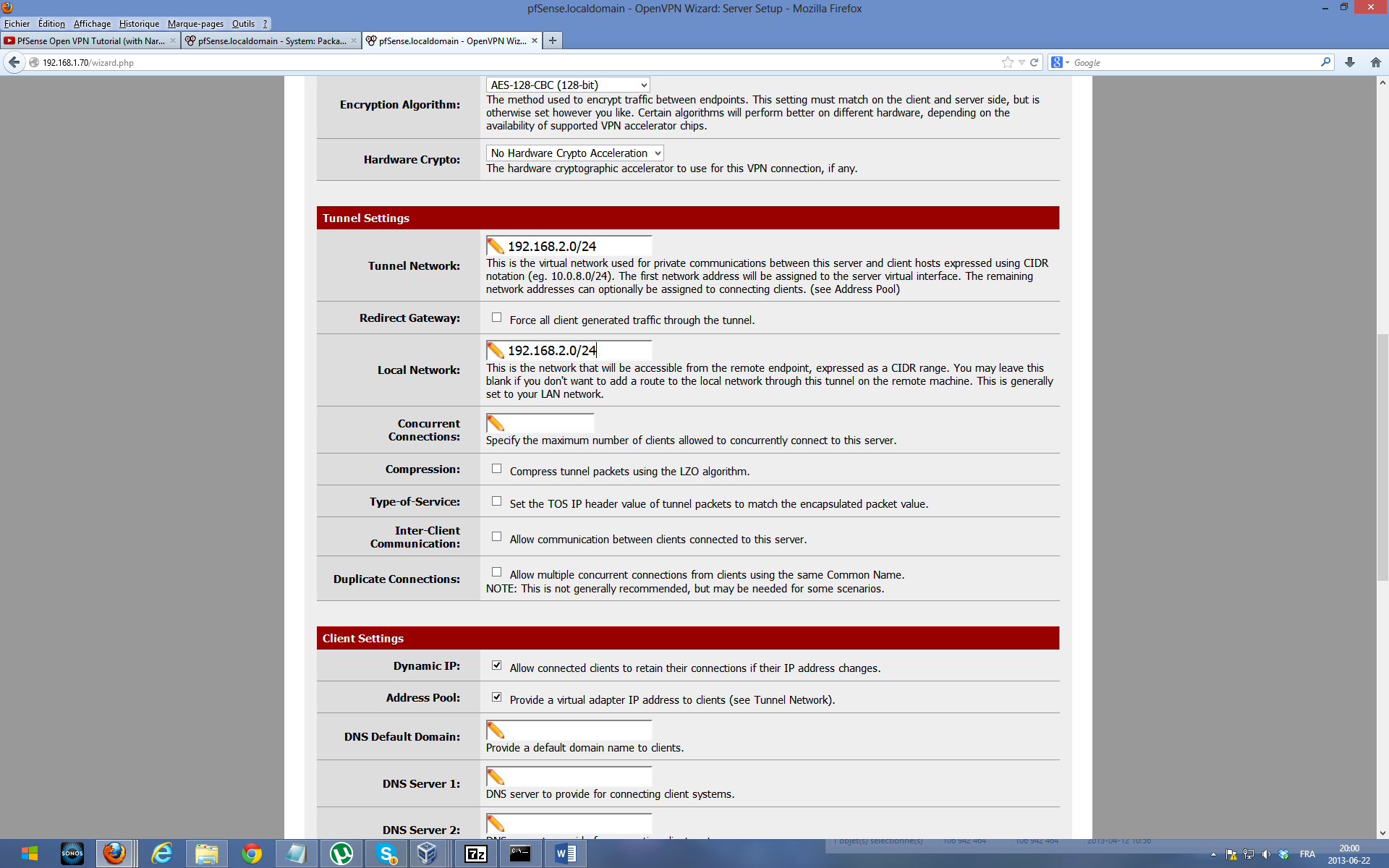


Add new certificate

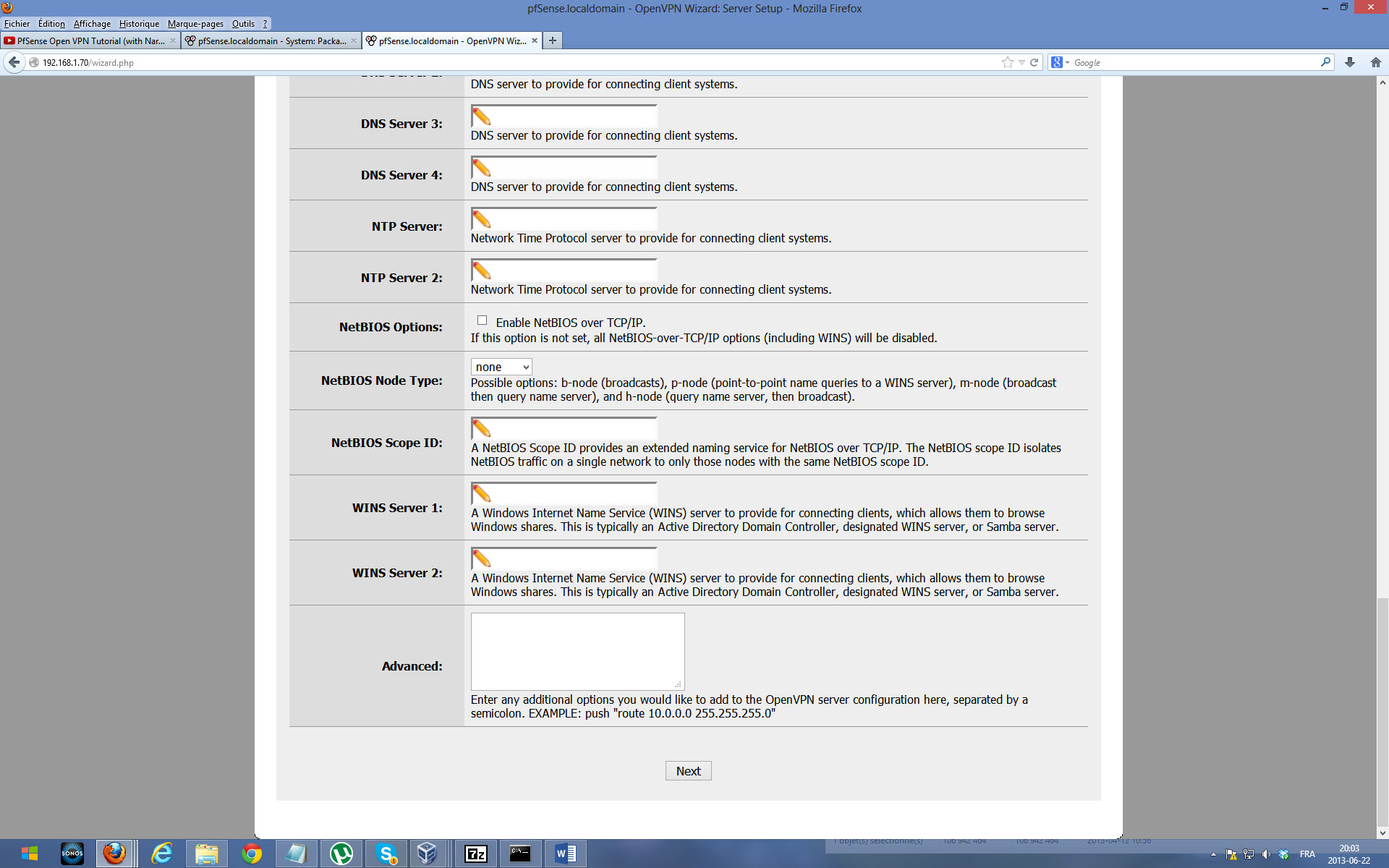




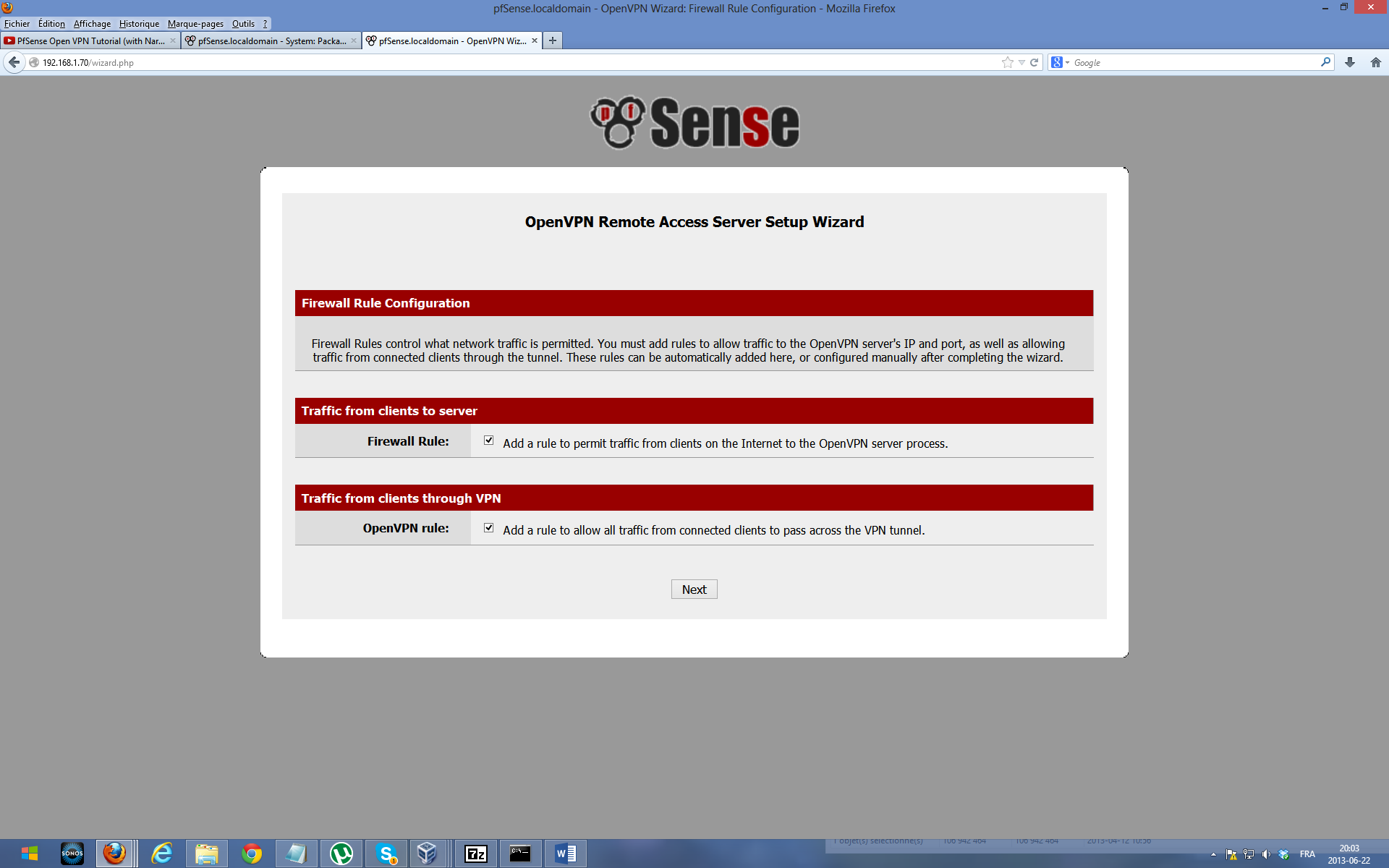
…



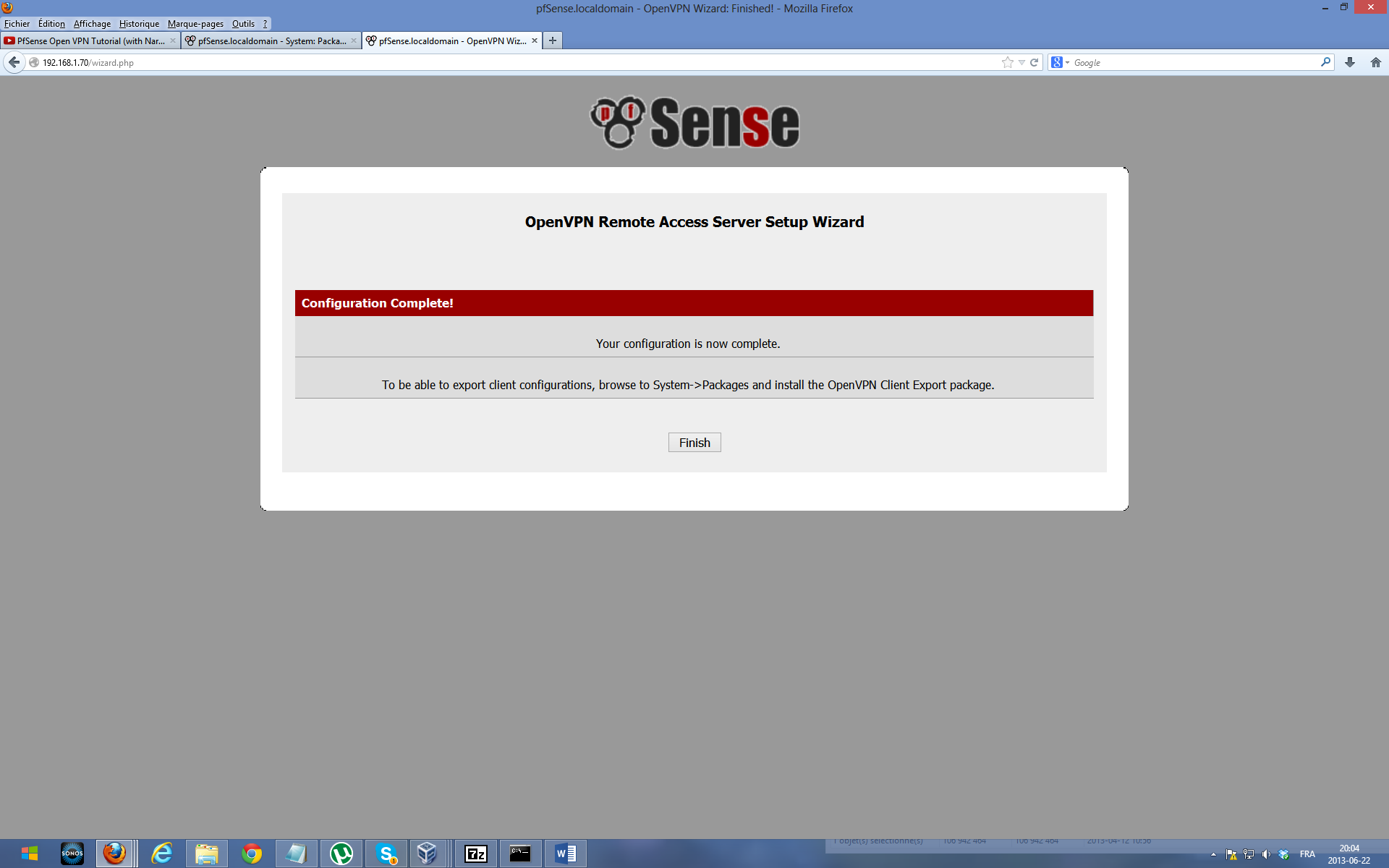
…

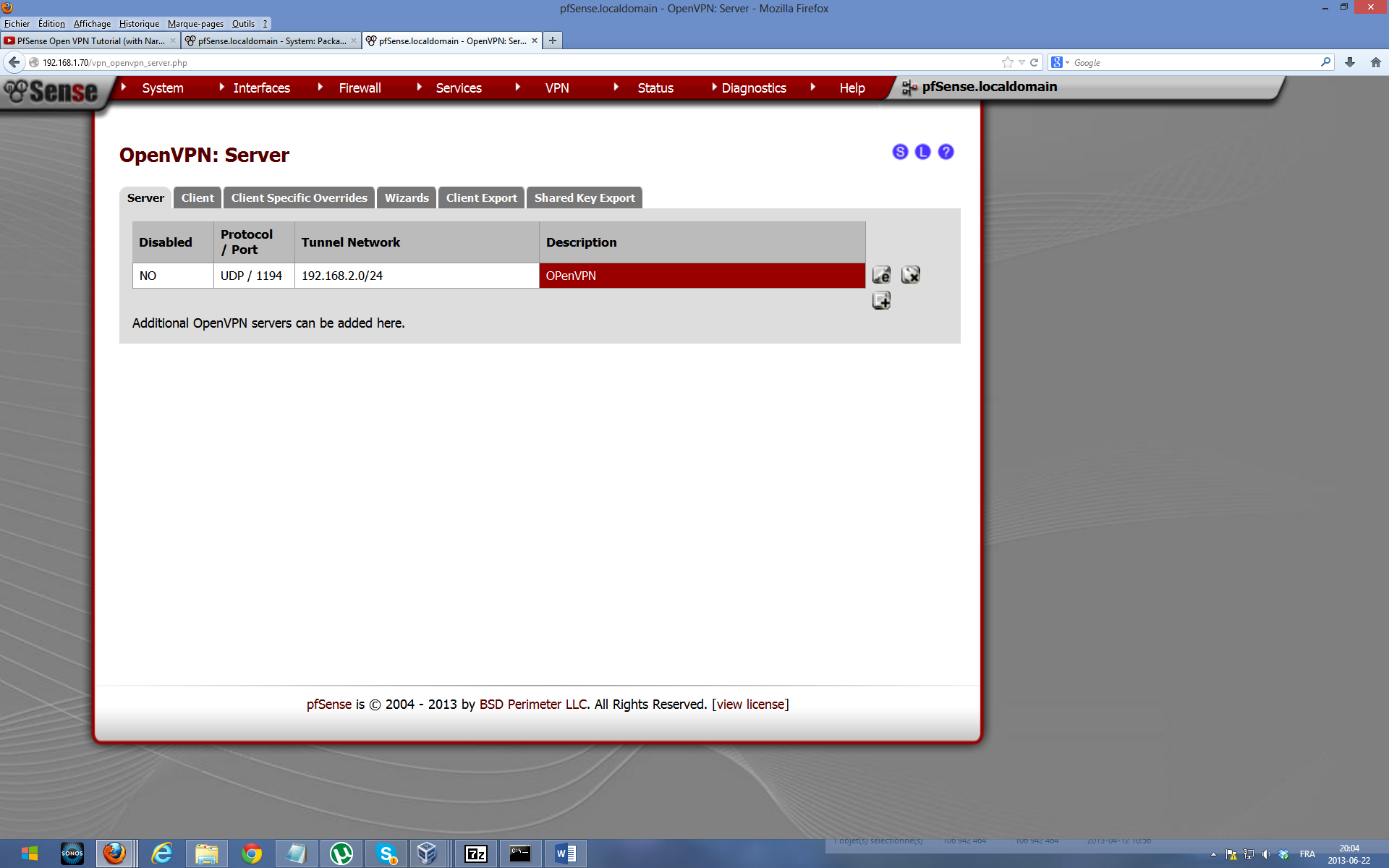


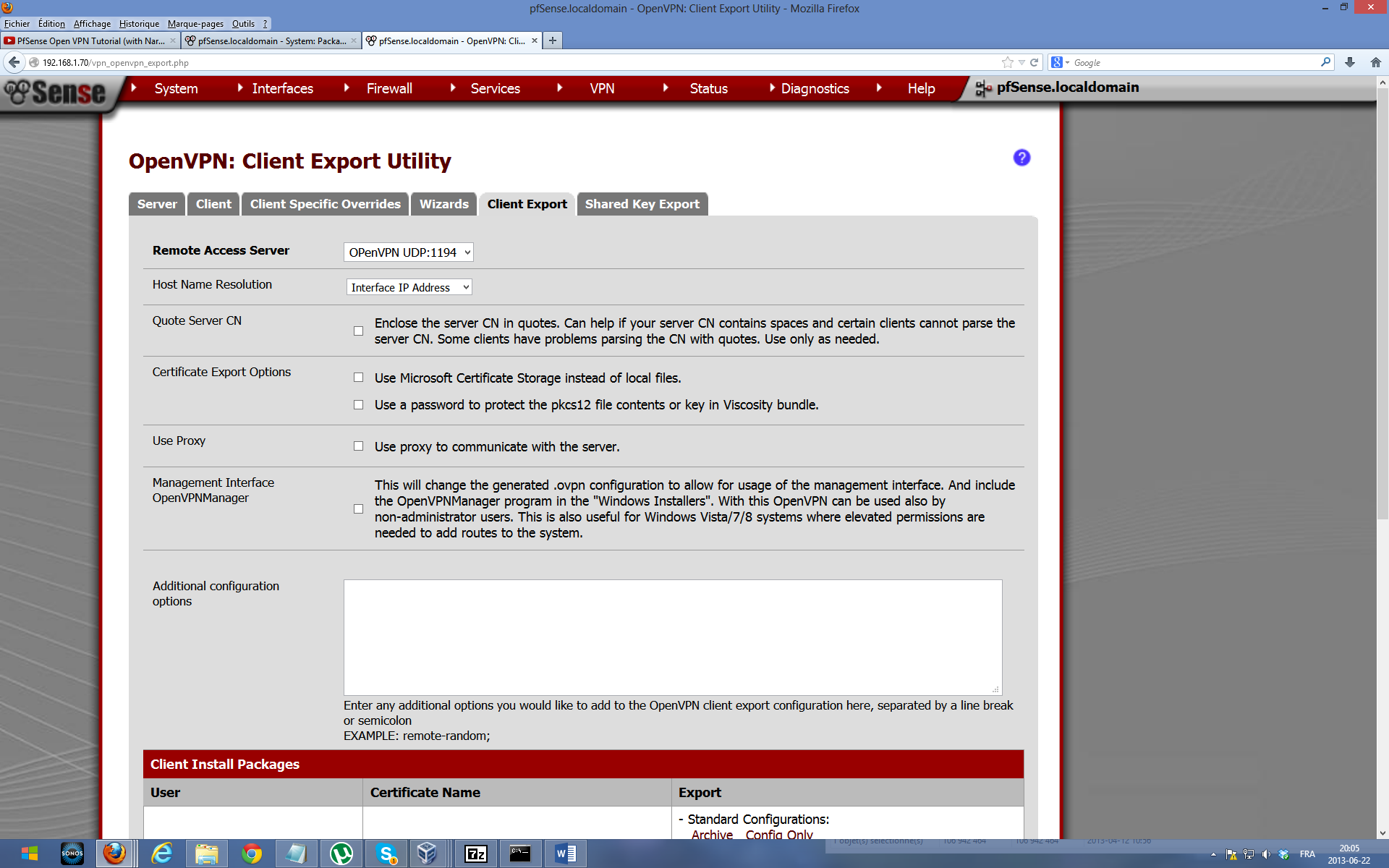
Next…



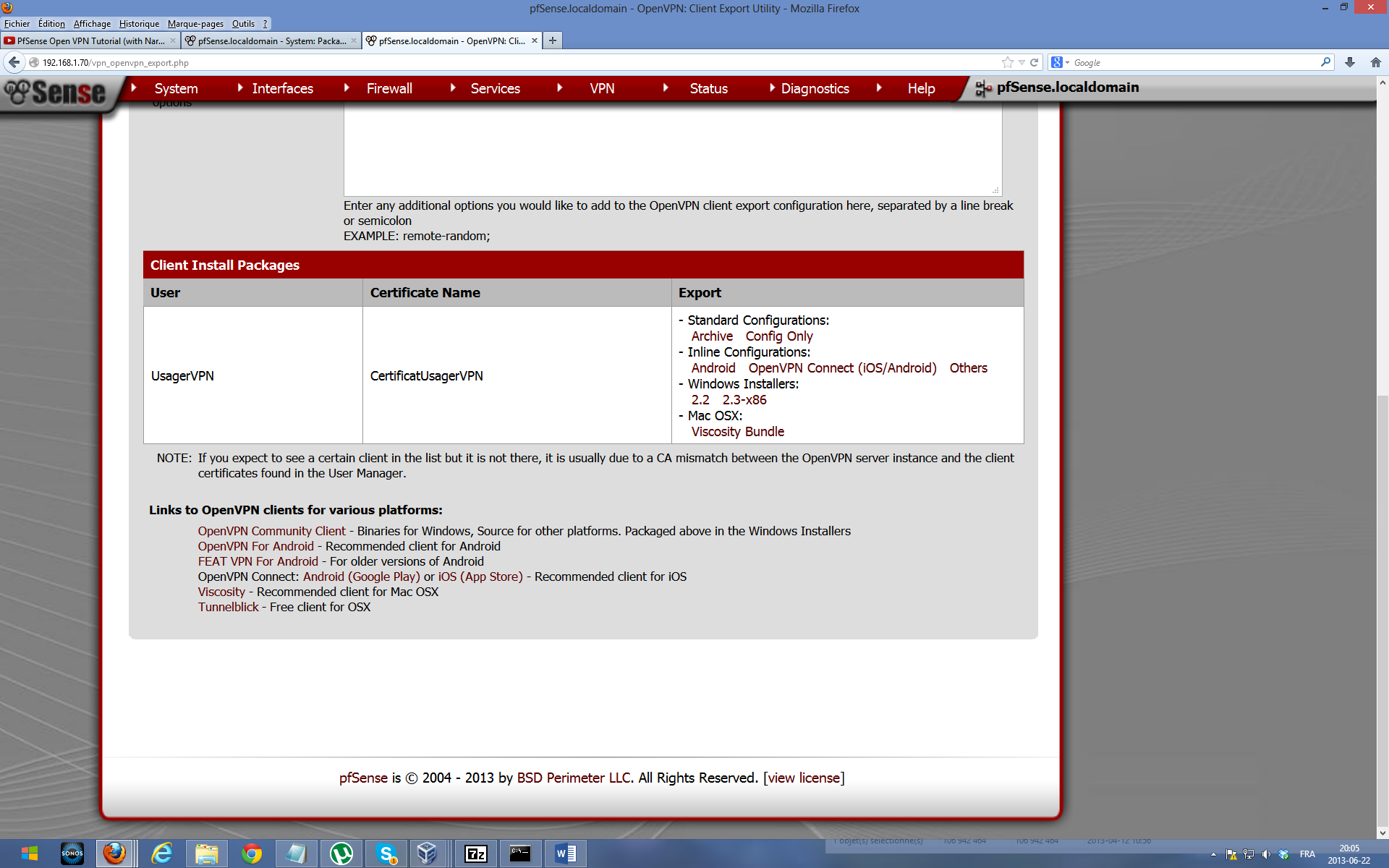
Next…



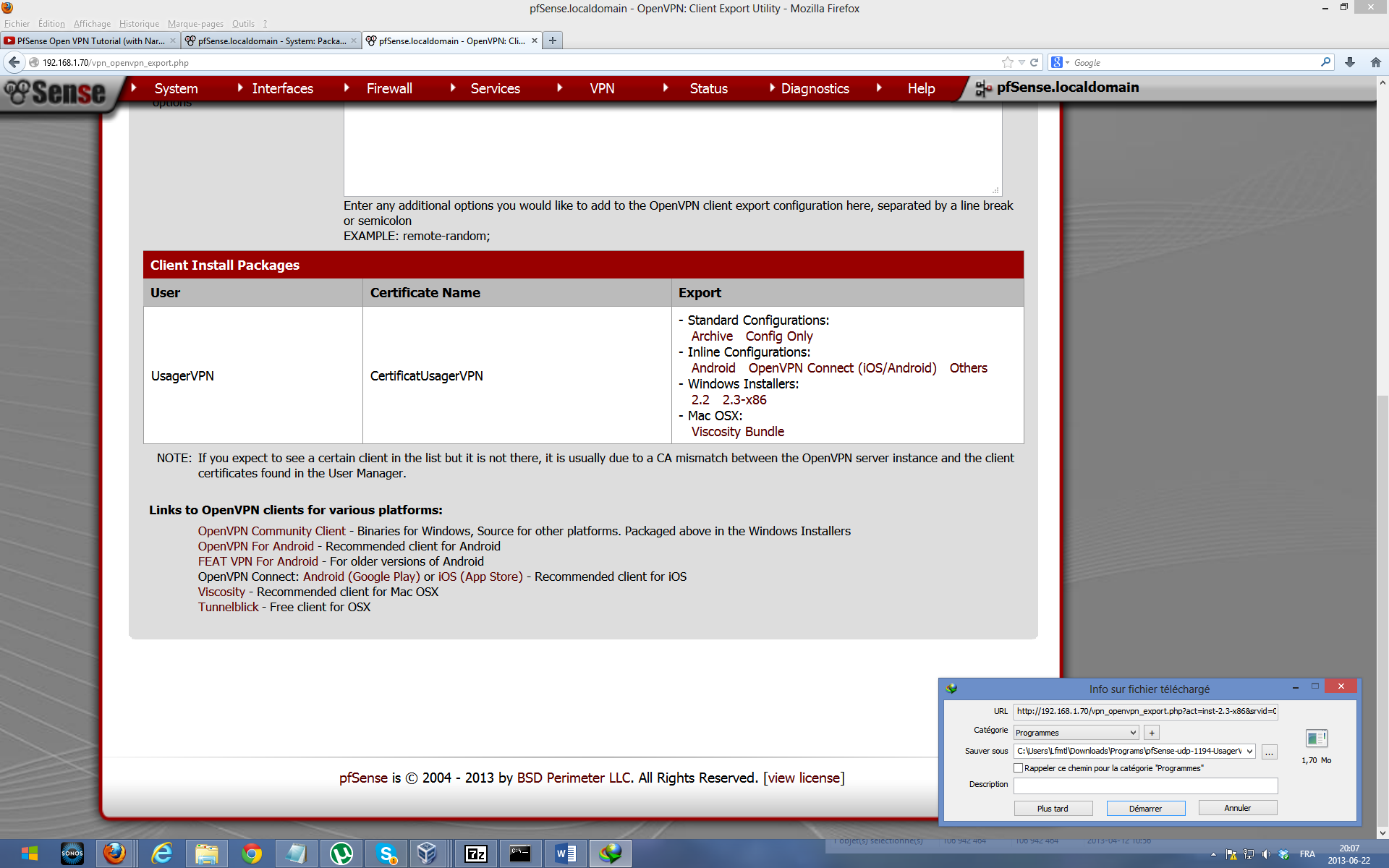




…



**Installation du client (suite)**



Vidéos YouTube

<http://www.youtube.com/watch?v=VdAHVSTl1ys>  
<http://www.youtube.com/watch?v=02vlrVNbe70>  
<http://www.youtube.com/watch?v=odjviG-KDq8>

# Outils d’intrusion et de détection

**snort, un système de détection d’intrusions (IDS)**

La plupart des IDS ont, à la base, un mode de fonctionnement assez simple : il s'agit de capturer l'ensemble des paquets qui passent par l'interface réseau d'un hôte et d'appliquer sur ceux-ci un ensemble de règles afin de détecter les communications suspectes.   
En somme un IDS est similaire à une application telle que *wireshark*; cependant les IDS sont conçues pour être beaucoup plus performantes, disposer de mécanismes de filtrage plus spécifiques à la détection de menaces et disposer de fonctionnalités plus poussées de stockage, d'alertes et de journalisation.

*snort* est un moteur de IDS développé sous licence libre par *SourceFire*, qui le commercialise dans le cadre de solutions plus complètes et « clé en mains ». De nombreuses applications libres sont cependant conçues pour fonctionner avec *snort*. Comme exemples… pour permettre le stockage d'alertes dans une base de données relationnelle - *snort-mysql*, pour fournir une interface graphique à l'utilisateur - *snorby*, *squert*, *acidbase* ou encore pour télécharger automatiquement les règles de détection les plus à jour – *pulledpork* et *oinkmaster*.

Comme pour wireshark, un système snort doit être placé à un endroit stratégique sur le réseau en fonction des besoins de surveillance. Dans le cadre de ce cours nous allons nous familiariser avec les fonctionnalités élémentaires de *snort* en mode "hôte", c'est-à-dire sur un hôte unique à une seule interface.

Pour ceux qui sont intéressés à une installation plus complète, vous pouvez essayer une distribution de linux assez récente vouée à la détection et la prévention d'intrusions nommée *SecurityOnion*.

**Prérequis:**

Un hôte *Debian* (*TurnkeyLinux*)

Sur votre hôte en mode superutilisateur, lancez l'installation de *snort* et *wireshark*. Le service snort démarre automatiquement à l'installation mais stoppez-le:

apt-get install snort wireshark

service snort stop

Au moment de l'installation, donnez le nom de votre interface et l'adresse du réseau sur lequel celle-ci se trouve, par exemple ***eth0*** et ***10.80.n.0/24***.

**Étape 1 : analyse de captures, règles sur mesure**

En plus de pouvoir écouter ce qui se passe sur une interface réseau, *snort* peut aussi analyser des captures *wireshark*; afin d'en comprendre le fonctionnement nous allons utiliser tout d'abord cette fonctionnalité.

Tout d'abord, créer un fichier nommé ***/etc/snort/local.rules*** et ajoutez-y la ligne suivante …

log icmp any any -> any any (msg:"PING Detecte!!"; sid:1000001)

Avec un fichier de captures (par exemple ***smallFlows.pcap***), lancez la commande suivante:

snort -c local.rules -r smallFlows.pcap

Suite à cette commande *snort* génère un fichier lisible par *Wireshark* (le "log") dans le répertoire***/var/log/snort.***

Modifez ensuite la règle comme suit :

alert icmp any any -> any any (msg:"PING Detecte!!"; sid:1000001)

Après exécution de la commande, *snort* génère maintenant un fichier nommé ***alert*** dans le répertoire.

Lancez la commande suivante, puis regardez ensuite le contenu de ***/var/log/snort/alert***:

snort -c local.rules -r exercices.pcap -A fast

**Étape 2 : capture et filtrage, règles sur mesure**

Pour lancer *snort* sur une interface, il suffit d'omettre la référence au fichier:

snort -c local.rules

Pour tester la capture lancez un *ping* sur l’hôte. Allez ensuite voir le résultat dans ***/var/log/snort/alert***

Pour abréger les messages d'alerte:

snort -c local.rules -A fast

Pour afficher les messages sur la ligne de commande:

snort -c local.rules -A console

**Syntaxe des règles :**

ACTION PROTO IPSRC PORTSRC -> IPDST PORTDST OPTIONS

***Actions***

|  |  |
| --- | --- |
| **alert** | Génère une alerte et ensuite une entrée dans le journal |
| **log** | Génère seulement une entrée dans le journal |
| **pass** | Ignore le paquet |
| **activate** | Définit un critère d’activation pour une autre règle |
| **dynamic** | Sera appliquée seulement si une règle *activate* a été déclenchée; génère une entrée dans le journal |
| **drop** | Bloque le paquet et génère une entrée dans le journal |
| **reject** | Comme une règle DROP, mais envoie en plus un TCP reset à l’origine |
| **sdrop** | Bloque sans journalisation |

***Protocoles***

Les protocoles peuvent être IP, TCP, UDP ou ICMP, en minuscule, par exemple:

alert tcp any any -> any any (msg:"Un paquet TCP entre")

***Adresses***

Les adresses peuvent être formulées normalement (sans masque) ou conformément à la notation CIDR, ce qui permet de spécifier des sous-réseaux, ou encore avec ***any***. On peut aussi mettre plus d’une adresse entre crochets et séparées par des virgules :

alert tcp 192.168.1.23 any -> any any (msg:"L’hôte 23 envoie un paquet TCP")

alert tcp any any -> 10.30.20.0/24 any (msg:"Paquet TCP pour n’importe quel hôte du réseau")

alert tcp [10.30.210.55,10.30.210.101] any -> any any (msg:"Paquet TCP de l’hôte 55 ou 101")

***Ports***

Comme les adresses, on peut utiliser ***any***; sinon on désigne les ports par leur valeur numérique. Pour spécifier plusieurs ports on peut les séparer d’une virgule ou définir des intervalles avec « : »

alert udp any 53 -> any any (msg:"Réponse DNS reçue")

alert udp any !53 -> any 53 (msg:"Requête DNS reçue")

alert tcp any any -> any 80,443 (msg:"Requête HTTP(S) reçue")

alert tcp any any -> any 100:500 (msg:"Paquets vers un port entre 100 et 500")

***Direction***

On peut avoir deux opérateurs : un unidirectionnel (***->***) ou un bidirectionnel (***<>***). Ce dernier permet de déclencher la règle pour les paquets entrants et sortants.

***Options***

Les options permettent de définir des critères supplémentaires en ce qui concerne les champs contenus dans les entêtes des protocoles, des chaînes de caractères détectées dans le message encapsulé, ou encore des traitements supplémentaires à faire subir à la trame qui a déclenché la règle. Il y a plus de 80 options possibles. Quelques options utiles:

|  |  |
| --- | --- |
| **content** | Définit une chaîne de caractères à rechercher dans le paquet. On peut en mettre plusieurs, *snort* recherchera les paquets qui contiennent toutes les chaînes de caractères spécifiées, peu importe l’ordre |
| **offset** | Donne le nombre d’octets qui précèdent le contenu cherché dans le paquet |
| **depth** | Spécifie la distance maximale, à partir du début, où le contenu est recherché, afin de dire à *snort* de ne pas chercher dans l’ensemble du "payload" |
| **distance** | Spécifie une distance minimale entre les deux éléments recherchés |
| **within** | Spécifie une distance maximale entre les deux éléments recherchés |
| **uricontent** | Permet de rechercher une chaîne de caractères dans une URI/URL demandée |
| **flags** | Filtrage par "flag" de l'entête TCP |

Pour toutes les options, référez-vous à la documentation (<http://manual.snort.org/node27.html>).

**Étape 3 : Service snort, règles sur mesure**

Déplacer le fichier des règles dans ***/etc/snort/rules/.***

Lancer le service avec la commande suivante:

service snort start

**Configuration du service**

Le service *snort* utilise les paramètres de configuration du fichier ***/etc/snort/snort.conf***.

**Variables**

Dans les règles il est possible de spécifier des variables au lieu de donner explicitement des adresses ou des ports. On définit les valeurs de ces variables dans *snort.conf* à l’aide des mots-clé ***ipvar*** ou ***portvar***, selon la syntaxe suivante :

ipvar NOM VALEUR

Par exemple si on a plusieurs serveurs web aux adresses 10.20.30.40 et 10.20.30.41, on pourra avoir la directive suivante :

ipvar HTTP\_SERVERS [10.20.30.40,10.20.30.41]

Ce qui permettra par la suite de définir une règle en utilisant cette variable, comme suit :

alert any any -> $HTTP\_SERVERS any (…)

**Préprocesseurs**

Les préprocesseurs sont des composantes de *snort* qui interviennent avant l'application des règles comme telles. Ils servent à transformer les paquets entrants avant l'application des règles, car les paquets peuvent avoir été modifiés (volontairement ou non) de manière à ce que les règles ne puissent plus s'appliquer. Deux exemples:

Imaginaons une attaque caractérisée par la présence de la chaîne d'octets cd 80 e8 d7 ff ff a2 d7. On crée donc l’alerte suivante:

alert tcp any any -> any any (content:"|cd 80 e8 d7 ff ff a2 d7|"; ...)

Que se passera-t-il alors si la trame IP contenant cette chaîne est fragmentée, et que l'on retrouve les 4 premiers octets dans un fragment et 4 autres dans le fragment suivant? Réponse: la règle ne s'appliquera pas.

Deuxième exemple, prenons une règle qui détecte les tentatives de se connecter sur l'interface d'administration *Wordpress* à partir d'un hôte à l'extérieur du réseau local:

alert tcp !$HOME\_NET any -> any 80 (uricontent:"/wp-admin"; ...)

Une attaque pourrait contourner cette règle en demandant l’URL suivant :

http://www.abc.com/%77%70%2Dadmin

Cette URL serait interprétée correctement par le serveur, mais pas détectée par *snort*...

La solution à ce genre de problèmes consiste à faire un prétraitement aux paquets entrants avant de les envoyer au moteur de règles. Dans le cas des deux exemples précédents, on utilisera respectivement les préprocesseurs ***frag3*** et ***http\_inspect.***

Aussi, lorsque certaines attaques sont caractérisées par des patterns de communication qui ne peuvent pas nécessairement être détectés par des règles (par exemple les messages de la couche 2 ou des séquences d’évènements), on utilisera un préprocesseur.

Quelques préprocesseurs activés par défaut dans *snort*:

**sfPortscan** : Détection de balayage de ports, conçu pour reconnaître ***nmap***

**ftp\_telnet\_protocol** : Normalisation et réassemblage des flux *FTP* et *telnet*

**dns** : Détection d’exploitations de quelques vulnérabilités reconnues

**arpspoof** : Détecte les messages ARP suspects

**ssh** : Détection de certaines attaques classiques sur le service *ssh*

**reputation** : Ajout de fonctionnalité de « liste noire » à *snort*

Pour plus d’informations sur les préprocesseurs, référez-vous à la documentation (<http://manual.snort.org/node15.html>)

**Étape 4: Service *snort*, règles téléchargées (*oinkmaster*)**

Installation du programme :

apt-get install oinkmaster

Dans ***/etc/oinkmaster.conf***, spécifier l’URL du site d’où les règles doivent être téléchargées. Dans ce cas-ci on choisira celles de *Emerging Threats* :

url = http://rules.emergingthreats.net/open/snort-2.9.0/emerging.rules.tar.gz

Lors du téléchargement des règles, le programme fait automatiquement une sauvegarde des anciennes règles; il faut cependant créer le répertoire où cette sauvegarde sera faite :

mkdir /etc/snort/backup

Pour lancer le téléchargement des règles à jour, lancez la commande suivante :

oinkmaster -o /etc/snort/rules -b /etc/snort/backup

On automatise habituellement cette procédure de mise-à-jour à l’aide d’une tâche ***cron***.

**Exercices**

1. Faire une règle (la plus précise possible) qui détecte les tentatives de se connecter comme ***root*** en ftp.
2. Faire une règle qui détecte toutes les tentatives de se connecter au port 9292.
3. Faire une règle qui détecte toutes les tentatives d’ouvrir l’URL /icinga/ à partir d’une adresse autre que faisant partie du réseau local.
4. Faire une règle qui détecte les scans nuls/xmas.

# Références

* <http://www.picsi.org/fiche_259.html>
* <http://fr.wikipedia.org/wiki/GNU>
* <https://www.gnu.org/home.fr.html>
* <http://commentcamarche.com>
* <http://fr.wikipedia.org/wiki/Transport_Layer_Security>
* <http://fr.wikipedia.org/wiki/Certificat_%C3%A9lectronique>
* <http://fr.wikipedia.org/wiki/Secure_Shell>
* <http://fr.openclassrooms.com/informatique/cours/reprenez-le-controle-a-l-aide-de-linux/iptables-le-pare-feu-de-reference>
* <http://www.planet-libre.org/index.php?post_id=15665&go=external>
* <http://www.inetdoc.net/guides/iptables-tutorial/introduction.html>
* <http://doc.ubuntu-fr.org/iptables>
* <http://wiki.ubuntu.com/IptablesHowTo>
* <http://www.inetdoc.net/guides/iptables-tutorial/howtoread.html>
* <http://profdinfo.com/web/420-128-LG/Section%204.html#43>
* <http://www.inetdoc.net/guides/iptables-tutorial/statemachine.html>
* <http://www.packtpub.com/article/pfsense-configuring-nat-firewall-rules>
* <http://xenod.free.fr/index.htm>
* <https://doc.pfsense.org/index.php/Main_Page>
* <https://forum.pfsense.org/index.php?topic=20973.0>
* <https://prezi.com/aluporbplcft/pfsense/>
* <http://fr.wikipedia.org/wiki/Nmap>
* <http://blog.nicolargo.com/2007/08/nmap-le-scanneur-de-reseau.html>
* <https://ssi.ac-strasbourg.fr/bonnes-pratiques/recommandations/lidentification-et-lauthentification/>
* <https://www2.univ-mlv.fr/~monestie/doku.php?id=cume_2008:linuxetud:linuxpam>
* <http://blog.moncoindunet.fr/linux/openssl-creer-un-certificat-auto-signe/>
* <http://fr.openclassrooms.com/informatique/cours/securiser-son-serveur-linux>

# Annexe A … Analyse de risque

Le risque est au cœur de nos préoccupations quotidiennes, mais qu’est-ce que l’analyse de risques?  
Plus précisément…

* Qu’est-ce qu’un risque?   
  Qu’est-ce qu’un danger?
* Comment faire une analyse de risques?
* Quels sont les outils et les méthodologies?
* Comment identifier des scénarios de risques?
* Comment négocier des objectifs pour évaluer les risques?
* Comment définir des barrières de protection?
* Comment établir un plan d’actions?

**Introduction et définitions**

On entend régulièrement le terme risque et pourtant sa compréhension et son utilisation par le grand public sont le plus souvent erronées. La confusion vient en effet de la définition des mots risques et dangers.  
Dans un premier temps, afin de cerner le sujet, voici quelques exemples de risques dont on parle souvent:

* les effets secondaires d’un médicament ou d’une opération chirurgicale;
* les accidents de la circulation;
* la contamination de sang, de rivières, de l’environnement …;
* les accidents dans la pratique d’un sport ou d’une activité;
* l’explosion d’une usine de produit dangereux;
* le crash d’un avion ou le déraillement d’un train;
* le mauvais (défaut de) fonctionnement d’un appareil, d’un service ou d’un système;
* la mauvaise exécution d’une tâche;
* les cataclysmes naturels;
* les prises de position à la bourse (risques financiers);l
* ’intrusion de virus informatiques (risques informatiques);
* …

Bien que ces risques soient de natures très différentes, leur analyse procède d’une même démarche théorique. Cependant, des différences d’interprétations et de vocabulaire sont couramment observées dues essentiellement à des écoles de pensées liées à des métiers particuliers. L’approche que nous allons étudier émane de la « science du danger ».

**Définition du « risque »**  
D’une part, les scientifiques, qui ont créé la science du danger appelée « cindynique » et d’autre part, les professionnels tels que les assureurs, en donnent une même définition :

**R associe D \* P \* G \* A**

Ce qui veut dire en clair…  
Le **risque** est l’association d’un **danger**, de sa **probabilité**, de sa **gravite** et de son **acceptabilité**.

Le risque est un ensemble de quatre éléments indissociables. Un peu comme une équation mathématique. Cependant, ces quatre éléments ne sont pas de même nature. C’est pour cela qu’il n’est pas possible de mettre un signe = entre R et les autres éléments.  
Comme il est dit dans l’introduction, attention à l’amalgame entre les mots risque et danger.   
Pour éviter de définir le risque par rapport au danger et le danger par rapport au risque (débouclage de définition, le danger étant une potentialité de risque), il est nécessaire de donner une définition originale du danger:  
Le danger est défini par un ensemble de processus qui déroule l’enchaînement d’événements conduisant à un évènement non souhaité (ENS) pouvant avoir un impact, en général destructeur, sur une ou plusieurs des quatre cibles possibles:

* un ou des individus;
* une ou des populations;
* un ou des écosystèmes;
* un ou des systèmes matériels ou symboliques.

Définitions des autres termes utilisés dans la définition du risque:

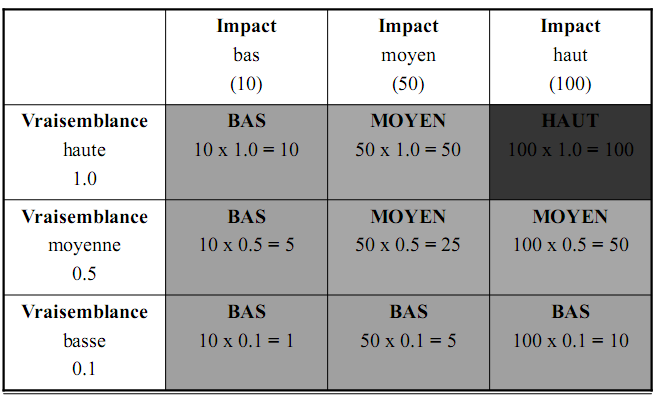
* **Probabilité**  
  Elle est définie par la probabilité d’enchaînement des événements conduisant à l’ENS.   
  Dans certain cas, on peut aussi utiliser le terme fréquence.
* **Gravité**  
  Elle est définie par l’effet des ENS sur les cibles.
* **Acceptabilité**  
  Elle est définie par l’acceptabilité de l’ENS par les acteurs dont les cibles. Nous constatons ainsi que les définitions révèlent des niveaux différents. Il est possible de définir scientifiquement le danger, sa probabilité bien sûr et sa gravité mais il n’est pas possible de définir scientifiquement son acceptabilité car à ce niveau, les subjectivités individuelles et collectives sont prépondérantes. « Le risque est un construit individuel et social ».

* **Processus**  
  C’est toute transformation dans le temps, l’espace, la forme (ou nature), de matière, d’énergie, d’information. Un processus s’établit entre un objet processeur, ou système processeur, ou source, ou système source et un objet « processé », ou système « processé », ou puits, ou système cible, par échange d’un flux de matière, d’énergie, d’information.

Définition de « l’Analyse de Risques »

En reprenant chacun des termes de la définition du risque proposée ci-dessus, même si elle est réductrice, on peut donner une première définition de l’analyse de risques (A.R.).L’A.R. en 5 étapes :

1. Traiter le **danger** et pour cela identifier les processus de dangers c'est-à-dire l’enchaînement d’événements issus de systèmes sources de dangers et pouvant conduire à des ENS.
2. Représenter l’enchaînement des événements conduisant à l’ENS, conduit à des représentations du type arbres logiques ou réseaux.
3. Afin de déterminer la **gravité** des ENS on détermine leur impact sur les cibles. Celui -ci peut être immédiat mais aussi différé traduisant des états de la cible dans le temps. Certains de ces états différés sont difficiles à prévoir, d’où le principe de précaution.
4. La détermination de l’acceptabilité se fait par négociation de tous les acteurs concernés dont les cibles. Dans certains cas, des limites peuvent être imposées par une réglementation (cas du nucléaire) ou par une règle spécifique (cas des installations classées).  
   Parmi les outils possibles voici les grilles Gravité \*Probabilité. Ce sont des outils simples et assez faciles à mettre en œuvre. Elles permettent de situer les scénarios d’accidents et de les hiérarchiser. On en trouvera un exemple ci-après.



# Annexe B … Introduction à la cryptographie

Depuis les débuts de l’écriture, l’être humain a cherché à transmettre des messages de manière codée et secrète. Depuis, cette discipline, qu’on appelle la **cryptographie**, a considérablement évolué au cours des siècles.

Ainsi, à partir d’un message lisible et compréhensible - donc formé avec un certain alphabet et une grammaire - on applique une série de transformations qui ont pour but d’obtenir un message dit codé ou chiffré. Si deux interlocuteurs connaissent le moyen de coder (et l’inverse décoder) le message, alors ils pourront communiquer de manière secrète.

La cryptographie a trois objectifs :

1. **Assurer la confidentialité**  
   Ceux qui interceptent le message et qui ne détiennent ni les codes ni les algorithmes pour le décoder ne peuvent comprendre son contenu. On peut donc transmettre des informations confidentielles.
2. **Assurer l’authenticité**  
   Est-ce que l’expéditeur est bien celui qui prétend être l’auteur?
3. **Assurer l’intégrité**  
   Un message intercepté par une tierce partie peut être transformé et ré-écrit.

Exemples …

**Confidentialité**On dit souvent: le nerf de la guerre, c’est l’information. Les séries historiques (les Rois maudits, Borgia, les Tudors, Rome, les Liaisons dangereuses …) en sont truffées d’exemple, où les intrigues, entre divers clans, familles, sont ponctués de drames de toutes sortes.   
Imaginons aussi les chaînes de commandement au sein d’une armée, où les ordres doivent être transmis d’un groupe à un autre: il ne faut pas que ces ordres puissent être compris des troupes ennemies.

**Authenticité**Un ordre est envoyé par un empereur à toutes ses trompes dispersées dans diverses campagnes. Est-ce que cet ordre a bien été envoyé par l’empereur? Ou est-ce une tactique des clans ennemis pour semer la pagaille?

**Intégrité**Depuis un certain temps, une rumeur court à l’effet qu’un agent au sein d’un service de renseignement (CIA par exemple) serait en réalité un agent-double. Un jour, un message est envoyé aux services secrets prouvant que monsieur X est cette taupe. Monsieur X, qui s’avère être un machiavélique agent-double, intercepte le message, et modifie son nom par celui d’un collègue, innocent, qui sera accusé à sa place.

La cryptographie a marqué l’imagination de nombreux écrivains et lecteurs, mais son application, dans la vie de tous les jours et technologique que nous connaissons est bien réelle.

Sans la cryptographie …

* Nous ne pourrions pas faire des achats avec une carte de crédit sur internet;
* Nos mots de passes échangés entre un poste de travail et un serveur seraient exposés;
* Nous ne pourrions pas échanger des informations confidentielles et privées sur le web.

C’est pourquoi, dans les réseaux et dans l’informatique, en général, la cryptographie est indispensable.

Le principe de la cryptographie est le suivant:

1. Celui qui rédige le message transforme son contenu en utilisant une méthode spécifique, le rendant ainsi incompréhensible;
2. Le message ainsi codé (on dit aussi: chiffré) est envoyé au destinataire;
3. Le destinataire utilise la même méthode pour déchiffrer le message.

On appelle **chiffre** cette méthode utilisée pour chiffrer et déchiffrer un message. Ce chiffre peut être aussi associé à un algorithme.

Il existe trois types de méthodes cryptographiques:

1. Les algorithmes de chiffrement faibles;
2. Les algorithmes de cryptographie symétrique;
3. Les algorithmes de cryptographie asymétrique.

**Les débuts de la cryptographie  
le chiffrement faible**

L’exemple le plus célèbre mais aussi le plus simple est celui du chiffre de César. Il consiste à changer toutes les lettres d’un message par une autre et ce selon un certain décalage.

Ce chiffre ainsi nommé car il était utilisé par Jules César pour communiquer avec ses armées. Cette méthode consiste simplement à décaler les lettres de l’alphabet d’un nombre donné de caractères: par exemple, si le décalage vaut 3, alors A est remplacé par D, B par E, C par F… X par A, Y par B, etc.

Avec le chiffre de César, si quelqu’un qui intercepte le message connaît la méthode de chiffrement, il aura tout de même besoin d’une information supplémentaire pour déchiffrer le message, soit la valeur de décalage utilisée. On appelle **clé** cette information essentielle à l’interprétation du chiffre. Évidemment, si on sait qu’un message est codé grâce au chiffre de César, il est assez simple de le déchiffrer même sans disposer de la clé car il n’existe que 26 clés possibles.

Exercice…   
Deux par deux, coder-décoder un message à l’aide du chiffre de César.

Un autre exemple est l’utilisation d’une table de transposition. (A est remplacé par X, B par E, C par W, D par J, E par Z, etc… , tout cela est fait de manière arbitraire, l’important est que les deux interlocuteurs aient la même table de correspondance. Même sans cette table, un message peut être facilement décodé par l’analyse de la fréquence des caractères. Sachant que le E revient souvent dans la langue française, il est facile de déduire que si un Z se retrouve à plusieurs reprises dans un texte, alors ce Z sera très probablement un E.

Ces méthodes de chiffrement montrent donc des limites. Par contre, l’un d’entre eux, le **chiffre de Vigenère**, eût longtemps la réputation d’être indéchiffrable.

Durant environ 300 ans (du 16e au 19e siècle), le chiffre de Vigenère est demeuré indéchiffrable, c’est-à-dire qu’il n’existait aucune méthode pour décrypter les messages à moins d’avoir la clé. Ce chiffre consiste à utiliser un chiffre de César différent pour chaque lettre de l’alphabet:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Lettre** | A | B | C | D | E | F | etc. |
| **Décalage** | 0 | 1 | 2 | 3 | 4 | 5 |

On applique ce décalage sur le message original en fonction d’une clé spécifique qui détermine quel décalage sera appliqué sur quelle lettre. Par exemple, si on a le message « Le ciel est bleu » et la clé « abri », on appliquera la clé sur le message comme suit :

Original: **L E C I E L E S T B L E U**

Clé: **A B R I A B R I A B R I A**

Décalage: **0 1 17 8 0 1 17 8 0 1 17 8 0**

* A est la 1ère lettre de l’alphabet, donc le L initial sera encodé comme L (décalage de 0 appliqué sur L)
* B est la 2e lettre de l’alphabet, donc le E suivant sera encodé comme F (décalage de 1 appliqué sur E)
* etc.

Le message final sera donc **L F T Q E M V A T C C M U**

Afin de simplifier, on peut utiliser la table suivante …

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |  | Message original | | | | | | |  |  |  |  |  |  |  |  |  |  |
|  |  | **A** | **B** | **C** | **D** | **E** | **F** | **G** | **H** | **I** | **J** | **K** | **L** | **M** | **N** | **O** | **P** | **Q** | **R** | **S** | **T** | **U** | **V** | **W** | **X** | **Y** | **Z** |
|  | **A** | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z |
|  | **B** | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z | A |
|  | **C** | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z | A | B |
|  | **D** | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z | A | B | C |
|  | **E** | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z | A | B | C | D |
|  | **F** | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z | A | B | C | D | E |
|  | **G** | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z | A | B | C | D | E | F |
|  | **H** | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z | A | B | C | D | E | F | G |
|  | **I** | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z | A | B | C | D | E | F | G | H |
|  | **J** | J | K | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z | A | B | C | D | E | F | G | H | I |
|  | **K** | K | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z | A | B | C | D | E | F | G | H | I | J |
|  | **L** | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z | A | B | C | D | E | F | G | H | I | J | K |
| **Clef** | **M** | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z | A | B | C | D | E | F | G | H | I | J | K | L |
|  | **N** | N | O | P | Q | R | S | T | U | V | W | X | Y | Z | A | B | C | D | E | F | G | H | I | J | K | L | M |
|  | **O** | O | P | Q | R | S | T | U | V | W | X | Y | Z | A | B | C | D | E | F | G | H | I | J | K | L | M | N |
|  | **P** | P | Q | R | S | T | U | V | W | X | Y | Z | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O |
|  | **Q** | Q | R | S | T | U | V | W | X | Y | Z | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P |
|  | **R** | R | S | T | U | V | W | X | Y | Z | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q |
|  | **S** | S | T | U | V | W | X | Y | Z | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R |
|  | **T** | T | U | V | W | X | Y | Z | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S |
|  | **U** | U | V | W | X | Y | Z | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T |
|  | **V** | V | W | X | Y | Z | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U |
|  | **W** | W | X | Y | Z | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V |
|  | **X** | X | Y | Z | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W |
|  | **Y** | Y | Z | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W | X |
|  | **Z** | Z | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y |

Le chiffre de Vigenère est facile à briser par analyse de fréquences.

Exercice …  
Deux par deux, Coder/décoder un message par chiffre de vigenère.

**Les méthodes symétriques**

Les méthodes que nous venons de voir sont des techniques symétriques. C’est à dire qu’elles reposent sur une clé secrète. Cette même clé sert à coder et décoder un message. Pour décrypter le message, il faut savoir quel est l'algorithme et quelle est la clé.

L’avènement des techniques numériques et des ordinateurs, dont la puissance de calcul décuple les rendements, a permis la découverte de certains algorithmes de cryptographie symétrique utilisés entre autres dans les technologies sans-fil:   
**RC4** et **AES**.

* **WEP** et **WPA** utilisent RC4. RC4 est considéré peu sûr.
* **AES** est approuvé par la NSA et est la base du **WPA2**.   
  L’attaque sur AES semble, pour l’instant, impossible, sauf par **force brute**.

**Une clé très prisée**

Même si on utilise des chiffres de plus en plus complexes, prenant 300 ans à déchiffrer avec les ordinateurs les plus puissants, le code ne sert plus à rien si la clé tombe entre les mains de ceux à qui elle n'est pas destinée. Ainsi pendant des centaines et des centaines d'années, de Jules César à la machine Enigma utilisée par les sous-mariniers allemands durant la seconde guerre mondiale, **le problème de la confidentialité de la clé est demeuré central**… jusqu'à 1976, où deux mathématiciens ont eu une idée géniale.

**Cryptographie asymétrique**

Comme nous l'avons vu, un encodage normal ne requiert qu’une clé. L’utilisateur A encode un message avec sa clé, et l’utilisateur B utilise la même clé pour décoder le message. Le problème est que:

* A et B doivent trouver un moyen de s’échanger la clé;
* A doit faire confiance à B de ne pas diffuser sa clé.

En 1976, *Whitfield Diffie* et *Martin Hellman* proposent un système selon lequel on utiliserait deux clés différentes: une pour encoder le message et une autre pour le décoder. Comme le fait d'encoder un message ne pose pas de problème de confidentialité particulier, cette clé peut être diffudée à n'importe qui: c'est la **clé publique**. La clé qui sert à décoder le message, elle, doit cependant demeurer secrète: c'est la **clé privée**. Les deux clés fonctionnent exclusivement l'une avec l'autre, comme un cadenas et sa combinaison.

*Diffie* et *Hellman* ont proposé le concept mais ce ne sont pas eux qui l'ont mis en pratique les premiers. L'algorithme de chiffrement, de déchiffrement et de génération des paires de clés associées a été mis au point en 1978 par trois cryptologues: *Ronald Rivest*, *Adi Shamir* et *Leonard Adleman*. L'algorithme se nomme **RSA**, des noms de ses trois inventeurs.

Ces deux clés, privées et publiques, fonctionnent ensemble. Seuls les messages encodés avec une clé publique pourront être décodés par la clé privée. *Ainsi, la clé privée reste sur un serveur et ne sort jamais de celui-ci*.

Pour mieux comprendre le chiffrement à clés publiques et privées, on peut prendre l’analogie d’un petit coffre-fort qui fonctionnerait comme suit: ce coffre-fort contient deux serrures. La première serrure, lorsqu’utilisée avec la bonne clé, verrouille le coffre-fort de manière à sécuriser son contenu. Il est impossible d’ouvrir le coffre-fort sans la deuxième clé et la deuxième serrure.

Donc, Josée envoie le coffre-fort, ouvert, par camion, à Johnny. Elle prend soin d’inclure, avec le coffre-fort, la première clé tout en conservant de manière secrète la deuxième.

À la réception du convoi, Johnny dépose une lettre confidentielle à Josée. Il prend la clé qui venait (la première, vous vous souvenez) avec le coffre, le ferme et utilise la première clé, ce qui a pour effet de verrouiller le tout.

Finalement, Josée reçoit le coffre. Elle prend sa deuxième clé et ouvre le coffre et prend connaissance de son contenu.

Dans cet exemple:

1. La première clé est la clé publique;
2. La deuxième clé est la clé privée;
3. Même si quelqu’un intercepte le colis, et même s’il fait une copie de la clé #1 qui vient avec le coffre, il ne pourra pas ouvrir le coffre une fois ce dernier verrouillé.

# Annexe C … nmap – Résumé des principaux commutateurs

**Spécifications des cibles**Les cibles peuvent être spécifiées par:

* des noms d'hôtes (*scanme.nmap.org*);
* des adresses IP (*192.168.10.100*)
* des adresses de réseaux (*cmaisonneuve.qc.ca/24*, *192.168.10.0/24*);
* dés étendues spécifiques (*10.0-255.0-255.1-254*);
* etc.

**nmap -iL <inputfilename>**Lecture de la liste des hôtes/réseaux cibles à partir du fichier**nmap -iR <num hosts>**Choix des cibles au hasard

**nmap --exclude <host1[,host2][,host3],...>**Exclusion des hôtes/réseaux du scan

**namp --excludefile <exclude\_file>**Exclusion des hôtes/réseaux des cibles à partir du fichier

**Découverte des hôtes:**

**nmap -sL**Liste des hôtes – Liste simplement les cibles à balayer

**nmap -sP**Balayage de ping – Détermination des hôtes en ligne

**nmap -P0 ou nmap -PN**Considère que tous les hôtes sont en ligne -- évite la découverte des hôtes

**nmap -PS/PA/PU [portlist]**Découverte TCP SYN/ACK ou UDP des ports en paramètre

**nmap -PE/PP/PM**Découverte de type requête ICMP echo, timestamp ou netmask

**namp -PO [num de protocole]**Ping IP (par type)

**nmap -n/-R**Ne jamais résoudre les noms DNS/Toujours résoudre [résout les cibles actives par défaut]

**namp --dns-servers <serv1[,serv2],...>**Spécifie des serveurs DNS particuliers

**Techniques de balayage**

**nmap -sS/sT/sA/sW/sM**Balaye TCP SYN/Connect()/ACK/Window/Maimon

**nmap -sN/sF/sX**Balaye TCP Null, FIN et Xmas

**nmap -sU**Balaye UDP

**nmap --scanflags <flags>**Personnalise les flags des balayages TCP

**nmap -sI <zombie host[:probeport]>**Idlescan (Balayage passif)

**nmap -sO**Balaye des protocoles supportés par la couche IP

**nmap -b <ftp relay host>**Balaye par rebond FTP

**nmap --traceroute**Détermine une route vers chaque hôte

**nmap --reason**Donne la raison pour laquelle tel port apparait à tel état

**Spécifications des ports et ordre de balayage**

**namp -p <étendue de ports>**Ne balaye que les ports spécifiés  
Exemple:  
namp -p22; -p1-65535; -pU:53,111,137,T:21-25,80,139,8080

**namp –F**Fast - Ne balaye que les ports listés dans le fichier nmap-services

**namp –r**Balayage séquentiel des ports en ne mélangeant pas leur ordre

**namp --top-ports <nombre>**Balaye <nombre> de ports parmi les plus courants

**namp --port-ratio <ratio>**Balay <ratio> pourcent des ports les plus courants

**Détection de service/version**

**namp –sV**Teste les ports ouverts pour déterminer le service en écoute et sa version

**namp --version-light:**Limite les tests aux plus probables pour une identification plus rapide

**namp --version-intensity <niveau>**De 0 (léger) à 9 (tout essayer)

**namp --version-all**Essaie un à un tous les tests possibles pour la détection des versions

**namp 1/**Affiche des informations détaillées du balayage de versions (pour débogage)

**Script da balayage**

**namp -sC:**équivalent de --script=safe,intrusive

**namp --script=<lua scripts>**<lua scripts> est une liste de répertoires ou de scripts séparés par des virgules

**namp --script-args=<n1=v1,[n2=v2,...]>**Passer des arguments aux scripts

**namp --script-trace**Montre toutes les données envoyées ou recues

**namp --script-updatedb**Met à jour la base de données des scripts. Seulement fait si -sC ou --script a été aussi donné.

**Détection de système d'exploitation:**

**namp –O**Active la détection d'OS

**namp --osscan-limit**Limite la détection aux cibles prometteuses --osscan-guess: Détecte l'OS de façon plus agressive

**namp --osscan-guess**Devine l'OS de façon plus agressive

**Temporisation et performance**

Les options qui prennent un argument de temps sont en milisecondes a moins que vous ne spécifiiez 's' (secondes), 'm' (minutes), ou 'h' (heures) à la valeur (e.g. 30m).

**namp -T[0-5]**Choisit une politique de temporisation (plus élevée, plus rapide)

**namp --min-hostgroup/max-hostgroup <msec>**Tailles des groupes d'hôtes à balayer en parallèle

**namp --min-parallelism/max-parallelism <msec>***Parallélisation* des paquets de tests (probes) --min\_rtt\_timeout/max-rtt-timeout/initial-rtt-timeout <msec>: Spécifie le temps d'aller-retour des paquets de tests

**namp --min\_rtt\_timeout/max-rtt-timeout/initial-rtt-timeout <msec>**Spécifie le temps d'aller-retour des paquets de tests

**namp --min-rtt-timeout/max-rtt-timeout/initial-rtt-timeout <time>**Précise le *round trip time* des paquets de tests.

**namp --max-retries <tries>**Nombre de retransmissions des paquets de tests des scans de ports.

**namp --host-timeout <msec> délai d'expiration du scan d'un hôte --scan-delay/--max\_scan-delay <msec>**Ajuste le délai de retransmission entre deux paquets de tests

**namp --scan-delay/--max-scan-delay <time>**Ajuste le delais entre les paquets de tests.

**Évasion pare-Feu/IDS et usurpation d'identité**

**namp -f; --mtu <val>**Fragmente les paquets (en spécifiant éventuellement la MTU)

**namp -D <decoy1,decoy2[,ME],...>**Obscurci le balayage avec des leurres

**namp -S <IP\_Address>**Usurpe l'adresse source

**namp -e <iface>**Utilise l'interface réseau spécifiée

**namp -g/--source-port <portnum>**Utilise le numéro de port comme source

**namp --data-length <num>**Ajoute des données au hasard aux paquets émis

**namp --ip-options <options>**Envoi des paquets avec les options IP spécifiées.

**namp --ttl <val>**Spécifie le champ durée de vie (*time-to-live*) IP

**namp --spoof-mac <adresse MAC, préfixe ou nom du fabriquant>**Usurpe une adresse physique (MAC)

**namp –badsum**Envoi des paquets TCP/UDP avec une somme de contrôle erronée.

**Sortie**

**namp -oN/-oX/-oS/-oG <file>**Sortie dans le fichier en paramètre des résultats du balayage au format normal, XML, s|<rIpt kIddi3 et Grepable, respectivement

**namp -oA <basename>**Sortie dans les trois formats majeurs en même temps

**namp –v**Rend Nmap plus verbeux (-vv pour plus d'effet)

**namp -d[level**Sélectionne ou augmente le niveau de débogage (significatif jusqu'à 9)

**namp --packet-trace**Affiche tous les paquets émis et reçus

**namp –iflist**Affiche les interfaces et les routes de l'hôte (pour débogage)

**namp --log-errors**Journalise les erreurs/alertes dans un fichier au format normal

namp --append-output  
Ajoute la sortie au fichier plutôt que de l'écraser

**namp --resume <filename>**Reprend un scan interrompu

**namp --stylesheet <path/URL>**Feuille de styles XSL pour transformer la sortie XML en HTML

**namp –webxml**Feuille de styles de références de Insecure.Org pour un XML plus portable

**namp --no\_stylesheet**Nmap n'associe pas la feuille de styles XSL à la sortie XML

**Divers**

**namp -6**Active le scan IPv6

**namp –A**Active la détection du système d'exploitation et des versions

**namp --datadir <dirname>**Spécifie un dossier pour les fichiers de données de Nmap

**namp --send-eth/--send-ip**Envoie des paquets en utilisant des trames Ethernet ou des paquets IP bruts

**namp –privileged**Suppose que l'utilisateur est entièrement privilégié -V: Affiche le numéro de version

**namp –unprivileged**Suppose que l'utilisateur n'a pas les privilèges d'usage des raw socket

**namp –h**Affiche ce résumé de l'aide

**Exemples**

**nmap -v -A scanme.nmap.org  
nmap -v -sP 192.168.0.0/16 10.0.0.0/8  
nmap -v -iR 10000 -P0 -p 80**

# Annexe D … Exercice - Mise en place de NAT sous Linux avec iptables

**(Distribution CentOS)**

La plupart des entreprises possèdent un nombre limité d'adresses IP « aiguillables » (routable) de l'extérieur provenant de leur fournisseur d’accès Internet. De ce fait, les administrateurs réseau doivent trouver d'autres moyens pour partager l'accès aux services Internet sans donner d'adresses IP publiques limitées à chaque nœud sur le réseau local.

L'utilisation d'adresses IP privées est la manière la plus utilisée afin de permettre à tous les nœuds d’un réseau local d'accéder correctement aux services du réseau (interne et externe).   
Les passerelles de périphérie (comme les serveurs mandataires NAT) permettent de relayer des transmissions sortantes vers Internet ou d’aiguiller des paquets provenant de l’extérieur vers un nœud du réseau local.

**Configuration des interfaces**

|  |  |
| --- | --- |
| **aa.aa.aa.aa** |  |
| **bb.bb.bb.bb** |  |
| **cc.cc.cc.cc** |  |
| **dd.dd.dd.dd** |  |
| **ee.ee.ee.ee** |  |
| **ff.ff.ff.ff** |  |

Ce tutoriel démontre, étape par étape, comment mettre en place la translation d'adresses réseau (NAT) sous Linux avec *iptables*.   
À la fin de l’exercice votre système va fonctionner comme une passerelle et pourra fournir un accès Internet à de multiples hôtes du le réseau local (LAN) en utilisant une seule adresse IP publique.  
   
**Exigences matérielles et/ou logiques**1. Un serveur physique ou virtuel avec 2 (deux) cartes d'interface réseau (NIC) ;  
2. Une distribution Linux ;  
3. *iptables*.  
4. Un client pour vérifier le bon fonctionnement de la passerelle

**Définitions de base**

* **aa.aa.aa.aa**  
  est l’adresse IP du réseau étendu (WAN)  
  et   
  **bb.bb.bb.bb**   
  est le masque de sous-réseau du réseau étendu (WAN);
* **cc.cc.cc.cc**  
  est l'adresse IP du réseau local (LAN) (par exemple 192.168.0.1 ou 10.0.0.1)   
  et   
  **dd.dd.dd.dd** (LAN)  
  est le masque de sous-réseau du réseau local (par exemple 255.255.255.0) ;
* **ee.ee.ee.ee**   
  est la passerelle par défaut pour la connexion Internet;
* **ff.ff.ff.ff**est l’adresse du serveur DNS ;
* eth0 est le nom de l’interface réseau reliée au réseau étendu (WAN);
* eth1 est le nom de l’interface réseau reliée au réseau local (LAN).

**Mise en place étape par étape**

**Étape 1  
Vérification si les deux interfaces réseau sont bien reconnues par Linux et sont entièrement fonctionnelles**

Pour visualiser si les interfaces réseau sont fonctionnels …

**dmesg | grep eth0  
dmesg | grep eth1**

Le résultat de la commande peut varier, mais dans la plupart des cas, il devrait ressembler à ce qui suit:

*eth1: RealTek RTL8139 au 0xe0830000, 00:30:04 f: 3b: af: 45, IRQ 19  
eth1: Identifié 8139 type de puce 'RTL-8100B/8139D'  
eth0: link up, 100Mbps, full-duplex, lpa 0x41E1*

Une sortie similaire devrait être pour les cartes réseau eth0 et eth1.

Pour vérifier que les cartes réseau sont reconnues par Linux en tant que dispositifs de réseautage…

**ifconfig eth0  
ifconfig eth1**

Le résultat de la commande peut varier, mais dans la plupart des cas, il devrait ressembler à ce qui suit:

*eth0 Link encap: Ethernet HWaddr 00:50:56: C0: 00:08  
 adr inet6: fe80 :: 250:56 ff: fec0: 8/64 Portée: Lien  
 UP BROADCAST RUNNING MULTICAST MTU: 1500 Metric: 1  
 RX paquets: 0 erreurs: 0: 0 overruns: 0 trame: 0  
 TX paquets: 41 erreurs: 0: 0 overruns: 0 porteuses: 0  
 collisions: 0 txqueuelen: 1000  
 Octets reçus: 0 (0.0 b) TX bytes: 0 (0.0 b)*

**Étape 2   
Configuration de l'interface WAN (eth0) pour obtenir une connexion Internet**

**ifconfig eth0 aa.aa.aa.aa netmask bb.bb.bb.bb**Exemple  
**ifconfig eth0 10.30.242.200 netmask 255.255.255.0**

Vous pouvez également modifier les paramètres de l’interface du réseau étendu (WAN) à l’aide du fichier de configuration ***/etc/sysconfig/network-scripts/ifcfg-eth0*** .

L’adresse IP du réseau étendu (WAN) et le masque doivent être fournie par votre fournisseur d’accès Internet.

**Étape 3  
Mise en place des paramètres de l’interface réseau étendu qui s'appliqueront après le démarrage du serveur**  
   
Pour les distributions telles que RedHat, Fedora, Centos le fichier de configuration eth0 est situé dans ***/etc/sysconfig/network-scripts/ifcfg-eth0****.*   
Pour éditer des fichiers de configuration utiliser le traitement de texte *nano*.

Après avoir édité ***/etc/sysconfig/network-scripts/ifcfg-eth0*** le résultat devrait ressembler à ceci…

*DEVICE = eth0  
ONBOOT = yes  
BOOTPROTO = static  
IPADDR = aa.aa.aa.aa # EX. 10.30.242.200  
NETMASK = bb.bb.bb.bb # EX. 255.255.255.0  
GATEWAY = ee.ee.ee.ee # EX. 10.30.242.200  
HWADDR = f 00:30:04: 3b: af: 45 # adresse MAC (entrée facultative)*

**Étape 4  
Configuration de l'interface LAN (eth1) pour obtenir une connexion Internet**   
Après avoir édité ***/etc/sysconfig/network-scripts/ifcfg-eth1*** le résultat devrait ressembler à ceci…

*DEVICE = eth1  
ONBOOT = yes  
BOOTPROTO = static  
IPADDR = cc.cc.cc.cc # EX. 192.168.125.1  
NETMASK = dd.dd.dd.dd # EX. 255.255.255.0  
HWADDR = 00:50:08 d: D1: 24: db # Adresse MAC de l’interface réseau local  
(entrée facultative)*

**Étape 5 (facultative)  
Définition des noms de domaine des serveurs**

Après avoir édité ***/etc/resolv.conf*** le résultat devrait ressembler à ceci

*nameserver 203.145.184.13  
nameserver 203.145.184.12*

**Étape 6  
Activation du la redirection IP**

**cd / proc/sys/net/ipv4/   
echo 1 > ip\_forward** [[1]](#footnote-1)

**Étape 7  
Mise en place du service NAT avec iptables**  
   
Pour supprimer les éventuelles règles existantes de chaque table *iptables [[2]](#footnote-2)*.

**Iptables -F  
iptables -t nat -F  
iptables -t mangle -F**

|  |  |
| --- | --- |
| Commutateur | Signification |
| -t | spécifie la table qui sera l’objet de l’action : (nat pour la translation et mangle pour le type de paquets) |
| -F | signifie *Flush* (vider) le contenu existant. |
| Important: Linux est sensible à la casse (MAJ / min) lors du passage de paramètres. | |

Pour activer la fonctionnalité de NAT...

**Iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE  
iptables -A FORWARD -i eth1 -j ACCEPT**

|  |  |
| --- | --- |
| Commutateur | Signification |
| -t | spécifie la table qui sera l’objet de l’action : (nat pour la translation et mangle pour le type de paquets) |
| -F | signifie *Flush* (vider) le contenu existant. |
| -A | signifie *Append* (ajout) |
| POSTROUTING | modifie le paquet avant sa sortie du système après être passé par le module d’aiguillage |
| -o | Indique l’interface de sortie (out) |
| MASQUERADE | indique le camouflage de l’adresse IP source avec l’interface par lequel les paquets IP sortent |
| FORWARD | signifie faire suivre |
| -I | Indique le nom de l’interface qui reçoit les paquets |
| -j | précise l’action qu’il faut faire si le paquet correspond à la règle |
| ACCEPT | signifie accepter |
| Important: Linux est sensible à la casse (MAJ / min) lors du passage de paramètres. | |

Dans la table du NAT (-t nat), ajouter une règle (-A) après le routage (POSTROUTING) pour tous les paquets qui sortent par eth0 (-o eth0) qui stipule de camoufler la connexion (-j MASQUERADE).  
POSTROUTING permet aux paquets d'être modifiés lorsqu'ils quittent le périphérique externe de la passerelle.

Il est ensuite préférable de redémarrer le service *iptables* afin de s’assurer que les modifications ont été prises en compte.

**service iptables restart**

**Étape 8  
Configuration des clients du réseau local (LAN) pour accéder à Internet à partir de la passerelle**

*Adresse IP du même réseau que cc.cc.cc.cc  
MSR : dd.dd.dd.dd  
Serveur DNS : ff.ff.ff.ff  
 Passerelle cc.cc.cc.cc*   
Exemple:  
*Adresse IP: 192.168.0.7  
MSR: 255.255.255.0  
Serveur DNS: 209.160.67.13  
Passerelle: 192.168.0.1*

**Pour en faire un peu plus…**

S’il existe un serveur sur le réseau interne que l’on souhaite le rendre disponible de manière externe, il faut utiliser la cible -j DNAT de la chaîne PREROUTING dans NAT pour spécifier une adresse IP et un port de destination où les paquets entrants demandant une connexion à votre service interne peuvent être retransmis.

Par exemple, si on souhaite retransmettre des requêtes HTTP au serveur HTTP Apache dédié sur 172.31.0.23, il faut saisir la commande suivante :

**iptables -t nat -A PREROUTING -i eth0 -p tcp --dport 80 -j DNAT --to 172.31.0.23:80**

Cette règle spécifie que la table NAT utilise la chaîne intégrée PREROUTING pour retransmettre les requêtes HTTP entrantes exclusivement à l'adresse IP de destination listée de 172.31.0.23.

**Lancement d'un service au démarrage (ex : httpd)**

**Vérification de l'état du service *iptables***

**chkconfig --list | grep httpd**

Le résultat de la commande peut varier, mais dans la plupart des cas, il devrait ressembler à ce qui suit:

*iptables 0:arrêt 1:arrêt 2:arrêt 3:arrêt 4:arrêt 5:arrêt 6:arrêt*

**Activation du service *iptables* au démarrage**

**chkconfig iptables on** (pour tous les niveaux)

**chkconfig --level5** (pour le niveau 5)

Le résultat de la commande peut varier, mais dans la plupart des cas, il devrait ressembler à ce qui suit:

*iptables 0:arrêt 1:arrêt 2:arrêt 3:arrêt 4:arrêt 5:arrêt 6:arrêt*

# Annexe E … Politique de sécurité

**(document tiré du site autourdupc.com)**

**Le besoin de sécurité**

L'informatique et les réseaux de télécommunications sont devenus des outils de travail indispensables pour les tâches professionnelles. Le bon fonctionnement doit en être assuré afin de garantir la liberté de communication.

« Bon fonctionnement » signifie comme conditions (entre autres):

* la protection des systèmes et des données nominatives est une obligation légale que l'Internet ne fait que renforcer;
* la fiabilité des logiciels et matériels;
* la performance et disponibilité du service   
  Rien de plus gênant et problématique que de ne pas pouvoir utiliser un service quand il le faut;
* la protection des informations stockées et échangées (Intégrité et confidentialité);
* la protection des accès aux systèmes (droits d'accès);
* la confiance dans l'identité des « correspondants » avec lesquels échanger des informations (garantie d'authentification et assurance de non « mascarade d'identité »).

Jusqu'au début des années 1980, la centralisation des moyens informatiques (quelques gros serveurs par établissement par exemple) et la quasi absence de communications avec l'extérieur permettait de facilement garantir les objectifs ci-dessus grâce à une administration (équipe) centralisée bien identifiée.

Les temps ont changé: le système d'Information d'une entreprise est aujourd'hui réparti, avec des « morceaux d'intelligence » partout et interconnectés entre-eux par le réseau.

Le « bon fonctionnement » de cette informatique distribuée, peut être perturbé.

En effet, si la généralisation de la connexion à Internet des entreprises et l'utilisation de plus en plus intense des systèmes et applicatifs « communiquants » offrent des possibilités nouvelles et prometteuses, elles introduisent également un certain nombre de risques dont il faut prendre conscience, en mesurer les conséquences éventuelles, et en connaissance de cause prendre les mesures adéquates (voir la section *Les risques*).

Qui dit mesures dit coûts (humains et financiers) et il y a comme toujours un compromis à trouver entre coût et sécurité.

Bien sur, il y a des coûts supplémentaires induits mais qui seront maitrisés en définissant raisonnablement une politique de sécurité avant même d'entamer la mise en œuvre technique.

Il n'est pas question de prêcher la mise en œuvre d'une « forteresse » mais d'un plan raisonnable.

En contrepartie, l'absence de sécurité pourrait bien coûter plus cher à terme, à la vue des conséquences que cela peut entrainer.

Le coût le plus important pour la mise en œuvre d'une politique de sécurité ne vient pas des matériels et logiciels supplémentaires nécessaires, mais des ressources humaines, car il faut assurer une surveillance régulière sans laquelle il n'y a pas de sécurité, et ceci est une charge importante.

La surveillance signifie (entre autres activités) :

* l’analyse des journaux, fichiers de traces, etc…;
* le passage régulier d'outils de vérification d'intégrité des systèmes;
* la vérification régulière de la « solidité » des mots de passe;
* la réflexion régulière sur la politique de sécurité.

**Les risques**

Le réseau d'un établissement peut être raccordé au réseau d'une entreprise dispersée géographiquement, lui-même maillon d'un réseau ouvert à des millions de machines, l'Internet (via, par exemple, des serveurs Web).

La population ayant accès aux services Internet ne fait qu'augmenter et dépasse plusieurs millions de personnes.

Or, d'après certaines études statistiques, il y aurait dans cette population 6 « malveillants » pour 10 000 utilisateurs. Le terme « malveillant » indiquant ici les personnes « décidées » à détruire, corrompre ou voler l'information. Mais il y a une autre catégorie, en plus grand nombre cette fois qui par esprit « ludique », par simple curiosité ou fréquemment par maladresse pourraient mettre en péril l'intégrité et la fiabilité des systèmes ainsi que la confidentialité des informations.

C'est une problématique douloureuse, et cependant deux certitudes:

* Il est impensable de contrôler le comportement de millions d'individus extérieurs à un établissement;
* l'outil réseau associé à l'Internet est aujourd'hui indispensable et il est donc hors de question aujourd’hui d'en priver la totalité d'une entreprise.

Parmi les risques, notons:

* **Destruction ou corruption d'informations**Ceci peut occasionner des pertes de temps considérables au personnel chargé de restaurer l'état initial des systèmes et bases de données.   
  De plus, si des sauvegardes ne sont pas régulièrement assurées, cela peut devenir catastrophique.
* **Déni de service (diminution de la qualité du service)**La qualité de l'informatique et du réseau sont devenus une pièce maitresse pour le fonctionnement d'une entreprise; cette qualité doit être garantie et maintenue.   
  Ceci peut entrainer une perte de confiance dans l'outil informatique et réseau mais aussi une perte d'argent par les retard accumulés si les performances de l'outil informatique se dégradent.
* **Vol d'informations confidentielles**Le travail scientifique doit être protégé car l'espionnage industriel est une réalité.  
  Les données financières doivent aussi être protégées, ainsi que l'accès aux fichiers confidentiels.
* **Modifications de données de haute importance**Ce risque est encore plus pernicieux que le précédent.   
  Il concerne des modifications stratégiques mais non détectables dans l'immédiat.   
  Il y a une certaine gravité si ces modification ne sont découvertes que trop tardivement (après plus d'une période de rotation des sauvegardes).
* **Mascarade d'identité**Cela peut porter un lourd préjudice à l'image de marque d'un établissement, quant aux relations ultérieures avec des partenaires**.**
* **Rebonds**Dans le cadre d'une attaque en règle, utilisation d'un système local à un établissement pour rebondir en cascade vers un autre site.   
  Vis à vis de ce dernier, l'établissement est responsable car étant le dernier dans la chaine.
* **Utilisation frauduleuse de ressources**Intrusion extérieure (ou intérieure) sur une machine sans autorisation et utilisation de ressources réservées (calcul, espace disque, logiciels, ...).   
  La responsabilité de l'établissement est engagée en cas d'utilisation de logiciels piratés, de stockage de contenus illicites, etc.

**On ne peut pas ignorer la sécurité.**

**Se protéger soi-même est un devoir.**

**Politique de sécurité**

Avant d'aborder la technique --> réfléchir, avoir une vision globale

En sachant :

* que la sécurité à 100% n'existe pas, (pas de paranoïa donc)
* qu'il y a nécessairement un compromis entre la valeur du «protégé» et le coût de la protection,
* qu'il faut savoir quoi protéger (nécessité d'un «audit»),
* qu'il faut œuvrer à la mise en place de moyens raisonnables,
* et que le but est d'être suffisamment «dissuasifs».

**Un peu de bon sens**

* La sécurité c'est 20% de technique et 80% de bons sens!
* La sécurité est très certainement d'abord une affaire de direction.   
  Son élaboration et sa mise en œuvre ne doit pas reposer sur une seule personne!   
  Tout le monde est concerné, (utilisateurs, administrateurs, etc.).   
  Trop de personnels considèrent que la sécurité informatique n'est pas leur problème. L'expérience démontre au contraire qu’un nombre important des incidents de sécurité trouvent leur source dans le comportement indélicat et non responsable des utilisateurs de base.
* Les hackers savent exploiter à merveille la moindre brèche laissée béante!
* La sécurité c'est comme une chaine   
  Un seul maillon faible fragilise l'ensemble ! Cela peut signifier qu'une équipe, ou même un seul individu qui n'applique pas les règles peut anéantir les efforts de toute la communauté.   
  Elle doit être vue et pensée de façon globale, si l'on désire une certaine cohérence.
* Pour des raisons d'efficacité, il est souhaitable qu'au niveau de l'établissement une personne (le correspondant de sécurité) ait une vision globale de l'ensemble afin:
  + de pouvoir détecter des incohérences notoires et de proposer si nécessaire des évolutions;
  + de veiller, en relation avec les autres administrateurs, à l'application des règles;
  + de pouvoir coordonner les actions de « riposte » en cas d'incident.
* Quand on dit global cela signifie entre autres:
  + les aspects physiques…  
    protection contre le vol, les incendies, les inondations,   
    protection des accès physique,   
    protection des supports de données (sauvegardes),  
     etc.   
    et
  + les aspects logiques/logistiques…   
    sensibilisation,   
    formations,   
    administration serveurs et réseaux,   
    etc.
* La sécurité ne doit pas être vue comme une contrainte mais comme un ensemble de règles librement consenties.
* Elle doit faire l'objet d'un consensus, et être entérinée par les instances de l'établissement (conseils) afin d'avoir toutes les chances d'être ensuite applicable et appliquée.
* La politique en matière sécurité doit être connue de tous.   
  Il faut sensibiliser sur les risques et leurs implications. L'ensemble des personnels est concerné.
* La politique en matière sécurité ne doit pas dépendre de la technique. Au contraire la technique utilisée dépendra de la politique choisie.

**Éléments de politique sécurité**

Une approche pour définir une politique de sécurité:

* bien connaître son environnement dans son intégralité;
* recenser ce qu'il faut protéger et à quel niveau;
* savoir d'où peuvent venir les menaces;
* évaluer / découvrir les risques majeurs encourus  
  tous les laboratoires ou services n'ont pas le même profil, en tenir compte, etc.;
* établir un catalogue des actions concrètes à entreprendre et des solutions à utiliser;
* établir une liste des règles pour l'administration et l'utilisation des systèmes et du réseau;
* définir des procédures qui permettront:   
  de s'assurer du bon suivi de cette politique de sécurité;  
  de vérifier que les règles en vigueur sont bien appliquées;  
  d'appliquer des moyens de pression / rétorsion dans le cas contraire;
* d'organiser la réaction en cas d'incidents;  
  s’assurer que tout ce qui a été décidé est tenable.

**Finalité de la politique de sécurité informatique**

Les finalités d'une politique sécurité s'articulent autour de 5 axes:

* sensibiliser aux risques pesant sur les systèmes d'information et aux moyens disponibles pour s'en prémunir;
* créer une structure chargées d'élaborer, de mettre en œuvre des règles consignes et procédures cohérentes pour assurer la sécurité des systèmes informatiques;
* promouvoir la coopération entre les différents services et unités de l’entreprise pour l'élaboration et la mise en œuvre des règles consignes et procédures définies;
* susciter la confiance dans le système d'information de l'établissement;
* faciliter la mise au point et l'usage du système d'information pour tous les utilisateurs autorisés de l'établissement.

**Informations nécessitant une protection**

Les informations sensibles non classifiées  
Il s'agit d'informations pour lesquelles le non-respect de la confidentialité, la disponibilité ou l'intégrité mettrait en cause la responsabilité du propriétaire ou du dépositaire, ou causerait un préjudice à eux-mêmes ou à des tiers.

Voici quelques exemples de critères permettant de qualifier une information de « sensible », parmi lesquels :

* informations nominatives (fichiers employés, étudiants, …);
* informations relevant du secret de la vie privée:   
  dossiers du personnel et médicaux;
* informations ne présentant pas un caractère de secret mais qui restent soumises à l'obligation de réserve ou de discrétion professionnelle;
* informations constitutive du patrimoine (scientifique, industriel, technologique, …);
* informations commerciales.

Il existe également des mentions de sensibilité (confidentiel / diffusion restreinte) qui peuvent être assorties d'une mention spécifique caractérisant le domaine à protéger   
(personnel / professionnel / industriel / université, etc.)

**Quelques vérités simples**

Rien ne sert…

* de se payer un super coffre-fort pour protéger quelques pacotilles et de laisser l'accès libre à une cave remplies de grands crus classés!   
  … on se trompe d’objectif, on se sait pas ce qu'il faut réellement protéger.
* de construire des remparts pour se protéger de l'aviation!  
  … cela montre que l'on ne sait pas d'où peuvent venir les attaques.
* d'utiliser un marteau pilon pour écraser une mouche!  
  … disproportion des moyens avec ce qu'il faut protéger;   
  … la sécurité doit avoir un coût raisonnable.
* d'acheter une porte blindée et d'oublier de fermer la fenêtre!  
  … la sécurité c'est une chaine, si un maillon est faible tout casse;  
  … une cohérence doit être assurée;  
  … surtout elle doit être vue globalement.
* d'employer un (et un seul) « gourou prêchant des formules secrètes » et de contraindre … les enfants à assister à la messe;  
  … la sécurité doit être simple et comprise (un minimum) par tous;  
  … la convivialité ne doit pas trop en souffrir;  
  … suffisamment de liberté (ouverture) doit être accordée.

La politique de sécurité des systèmes informatiques est un ensemble de règles définissant le comportement des utilisateurs.   
L'objectif de cette politique est la préservation et la sécurisation de l'intégralité du système informatique avec ses bases de données.

Notre politique de sécurité se base sur les normes et les documents suivants :

* RFC 2196  
  *Site Security Handbook*;
* FIPS PUB 191  
  *Federal Information Processing Standards Publication 191  
  Standard for: Guideline for the Analysis of Local Area Network Security*;
* Publications professionnelles concernant la sécurité des systèmes informatiques.

Les règles définies par la politique de sécurité concernent tout le processus du traitement et de consultation des informations, indépendamment des moyens de collecte et du traitement des données.

Nous menons une activité de conseil auprès de la direction informatique des entreprises désirant adopter une politique de sécurité informatique. Ainsi, nous conseillons la direction informatique à toutes les étapes de l'analyse nécessaire à la mise en place d'une politique de sécurité. L'acceptation par la direction générale des règles de la politique de sécurité est l'un des éléments les plus importants de la réussite de la politique de sécurité.

Dans le processus de la conception et de l'intégration de la politique de sécurité, il est important de designer une personne appartenant à l'entreprise. Cette personne, en général un informaticien de haut niveau, va être le responsable de la sécurité des informations circulant dans le système informatique.

Les étapes de la conception et de l'intégration de la politique de sécurité:

1. L'estimation de la valeur des informations traitées par le système informatique.  
   C'est une étape essentielle de la conception d'une politique de sécurité.   
   Dans le cadre de cette étape, nos spécialistes vont créer une base de données contenant les informations relatives aux différentes sortes d'informations traitées par le système informatique ainsi que les applications et le matériel utilisés pour leur traitement.  
   Les informations collectées pendant cette étape permettront de réaliser les autres étapes de la mise en place de la politique de sécurité.
2. La valorisation et la classification des informations stockées et traitées.  
   Les informations collectées lors de la première étape sont classifiées selon leur valeur pour l'entreprise. Les critères dont nous tenons compte sont notamment le degré de confiance, les conséquences de la transmission d'une information donnée à une personne non autorisée, les conséquences de la perte ou de la modification d'une information donné.  
   L'analyse conduite permet de classifier les informations traitées par le système informatique en 5 groupes :

* les informations non classifiées (qui sont publiées);
* les informations non classifiées (protégées, qui ne seront pas publiées);
* les informations classifiées;
* les informations confidentielles;
* les informations secrètes.

Même si la valorisation des informations pourrait présenter un risque dans certains cas, elle est un élément indispensable de la création d'un système de protection des données.

1. La définition de la meilleure circulation des informations à l'intérieur comme à l'extérieur de l'entreprise.  
   En possédant les données relatives aux différentes sortes de l'information du système, nous devons analyser les sens de circulation des informations.   
   Cette analyse permet de créer le schéma de la circulation des informations, pour en déterminer les points faibles. Aussi, cette analyse nous permet de diviser le système informatique en " domaines " en indiquant les sens de la circulation des informations.
2. L'analyse du risque:   
   une classification en catégories de risques avec les moyens permettant de les éviter.  
   Cette classification permet l'établissement d'un plan d'action. Celui-ci nous permettra d'utiliser la quasi-totalité des moyens existants pour prévenir les risques les plus probables. Lors de cette analyse de risque, nous attribuons des degrés de priorité aux diverses menaces qui peuvent avoir une conséquence sur le fonctionnement du système informatique de votre entreprise.  
   Suite à cette analyse, nous rédigeons un document opérationnel. Voici les informations que comporte ce document:

* la description du risque potentiel;
* les conséquences potentielles;
* le coût de réparation des dommages;
* la probabilité de la réalisation du risque;
* les actions prioritaires;
* la description des actions préventives;
* le coût des éléments de sécurisation.

1. L’élaboration d'une méthode de protection des informations adaptée aux spécificités du système informatique de l'entreprise.  
   La réalisation de cette étape commence par la définition des trois principaux niveaux de sécurité :

* le niveau de la procédure;
* le niveau physique;
* le niveau logique.

L'intégration de la méthode de sécurité est réalisée à chaque niveau en utilisant les meilleurs moyens de sécurisation (physiques, techniques, et juridiques).  
L'élaboration d'une « police de sécurité », c'est-à-dire d'un ensemble de documents décrivant la politique de sécurité.   
Outre la partie générale de la police de sécurité qui comprend les normes relatives aux méthodes d'une utilisation sécurisée du système informatique, la police de sécurité comprend les documents destinés aux utilisateurs finaux et aux groupes d'utilisateurs, établis en fonction des tâches concrètes accomplies par chacune de ces personnes. Chaque salarié devrait approuver avoir pris connaissance de ce document.  
Une police de sécurité bien conçue remplit trois fonctions principales:

* elle doit comporter les règles concernant chaque utilisateur;
* elle devrait pouvoir être utilisée comme source des connaissances en matière des meilleures méthodes de travail dans le système informatique;
* elle constitue une référence en termes de l'intégration de la politique de sécurité dans l'entreprise.

1. L'intégration d'une politique de sécurité suppose deux principaux modes d'action.

* L'installation du matériel et des logiciels indispensables au maintient d'un haut niveau physique de sécurité des données stockées et traitées par le système informatique.
* La formation des futurs utilisateurs du système informatique sécurisé.   
  En effet, les utilisateurs du système informatique devraient être formés aussi bien au niveau général qu'au niveau pratique.

1. L'évolution de la politique de sécurité et la maintenance du système des protections doivent être contrôlées régulièrement.  
   Avec le temps, de nouvelles menaces apparaissent et des modifications sont apportées au système informatique (le remplacement du matériel ou des logiciels, l'intégration des nouvelles fonctionnalités, la rotation du personnel).   
   La politique de sécurité doit en permanence prendre en compte tous ces changements, sinon elle deviendra complètement inutile.

# Annexe F… Modification du mot de passe *root* sous Linux

**Changer le mot de passe root perdu sur Debian**

Si vous avez perdu le mot de passe *root* et que vous avez un accès physique à votre machine, voici la procédure à suivre pour le réinitialiser.

* Redémarrer votre machine;
* Lorsque Grub apparaît, **sélectionnez la deuxième ligne**, mode de dépannage (*rescue mode*);
* **Appuyez sur la touche** **e** (pour éditer);
* **Placez votre curseur à la fin de la ligne commençant par Linux  
  Remplacez ro par rw**  
  et  
  **Ajoutez** **init=/bin/bash**;
* Appuyez ensuite sur **CTRL+X** pour démarrer;

Vous êtes maintenant connecté en tant que *root*;

* **Remonter la partition en écriture** avec la commande suivante  
  **mount -o remount,rw /**
* **Taper la commande** **passwd** pour changer le mot de passe root.

Référence  
<http://www.ingesys.fr/2011/07/20/changer-le-mot-de-passe-root-perdu-sur-debian-squeeze/>

# Annexe G… Sécuriser un serveur Linux

Ce guide va donner les grandes lignes comment sécuriser un serveur et donc offrir une introfuction aux thématiques de la sécurité informatique. En quoi est-ce important ? Par définition, un serveur est ouvert sur le monde, un minimum de sécurité est donc intéressant afin de se prémunir des attaques les plus simplistes.

La marche à suivre sera donc la suivante…

* présentation des failles;
* présentation des outils pour y pallier.

Point de vue matériel, il faudra…

* un serveur embarquant une distribution Debian (pour les autres distribs, le principe sera le même, mais les commandes risquent d'être différentes);
* un accès root (en ssh par exemple) et une console.

Tout ce document utilise la ligne de commande afin de pouvoir être accessible à tous. Comme il faudra éditer des fichiers, s’assurer d'avoir un éditeur de fichier en ligne de commande (nano est très simple).

**Filtrer le trafic à partir d’un pare-feu**

**Présentation**

Le pare-feu est l’élément indispensable pour sécuriser un serveur. Il va en effet filtrer tout le trafic en n’autorisant que les échanges permis par l’administrateur. Sans pare-feu correctement paramétré, tous les trafics sont plus ou moins permis (c’est-à-dire qu’un attaquant peut faire ce qu’il veut) et ce genre de faille est détectable par un simple balayage de ports.

Or, le noyau Linux offre déjà un pare-feu à l’utilisateur, qu’il est possible de configurer via le logiciel iptables (normalement contenu dans /sbin/iptables).

Voici le détail du fonctionnement d’un pare-feu - relativement simple. Un firewall analyse tout le trafic et vérifie si chaque paquet échangé respecte bien ses règles (critères de filtrage). Donc, il suffit de spécifier de bonnes règles pour interdire tout trafic superflu.

Les critères peuvent être divers (filtrer les ports, les protocoles, les adresses IP, etc). De base, la première étape sera de spécifier les règles sur les ports. Il faut être le plus strict possible quant au choix des règles; c’est pourquoi, par défaut, tout pare feu se règle en premier lieu en bloquant tout, absolument tout. Ensuite, il suffira d’ouvrir (autoriser le trafic) certains ports qu’il est souhaité utiliser (par exemple pour un serveur web, il faudra ouvrir le port 80 afin que le site web soit accessible).

Pour plus de souplesse, les règles seront écrites sous forme de script *bash*. Petite mesure de prudence si on a ouvert une session sur un hôte distant (à l’aide du protocole ssh), il faut être certain de ne pas se bloquer l’accès ou - le cas échéant - de pouvoir redémarrer l’hôte. Sinon, récupérer son serveur sera compliqué !

**Déclaration des règles**

La commande iptables -L -v permet de consulter les règles courantes

iptables -L -v

**Création du script**

nano /etc/init.d/firewall

Le fichier comporte les éléments qui suivent.

#!/bin/sh

On efface les règles précédentes pour partir sur de bonnes bases…

:

iptables -t filter –F

iptables -t filter -X

On bloque par défaut tout le trafic   
(si on est en ssh, bien entendu, ne pas exécutez le script tout de suite)…

iptables -t filter -P INPUT DROP

iptables -t filter -P FORWARD DROP

iptables -t filter -P OUTPUT DROP

On ne ferme pas les connexions déjà établies…

iptables -A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT

iptables -A OUTPUT -m state --state RELATED,ESTABLISHED -j ACCEPT

Les paramètres -m et --state indiquent de ne pas fermer les connexions qui sont déjà établies.

On autorise le *loopback* (il ne faut pas se bloquer soi-même !)…

iptables -t filter -A INPUT -i lo -j ACCEPT

Attention: iptables -t filter -A OUTPUT -o lo -j ACCEPT  
 le commutateur *–i* ne peut pas s’utiliser avec *lo*.

Tout est bloqué, il ne reste plus qu’à ouvrir les ports utilisés.

**Ouverture des ports utilisés**

Voici en détail les paramètres de *iptables*.

|  |  |
| --- | --- |
| Commutateur | Signification |
| -t | indique la table qui sera l’objet de la commande  (la valeur par défaut est la table FILTER) |
| -A | sert à indiquer le sens du trafic: INPUT (entrant) ou OUTPUT (sortant) |
| -p | indique le protocole visé  (TCP ou UDP en principe) |
| --dport et  –sport | désigne respectivement le port destination et leport source  (ici le serveur donc principalement dport) |
| -j | traitement sur le paquet  (ACCEPT et DROP pour respectivement accepter et refuser le paquet) |

Ainsi, une règle simple aura la forme suivante…

iptables -t *filter* -A *INPUT/OUTPUT* -p *protocole* --dport *port* -j ACCEPT

Afin de permettre un échange, il faut toujours ouvrir le port dans les deux sens (INPUT et OUTPUT).

On permet le trafic vers le serveur web (port 80)…

iptables -t filter -A OUTPUT -p tcp --dport 80 -j ACCEPT

iptables -t filter -A INPUT -p tcp --dport 80 -j ACCEPT

Il ne reste qu’à spécifier toutes les règles nécessaires.

Voici un petit tableau (il s’agit de données par défaut).

|  |  |  |
| --- | --- | --- |
| Service | Port d’écoute | Protocole |
| ssh | 22 | TCP |
| web/HTTP | 80 | TCP |
| FTP | 20 et 21 | TCP |
| SMTP | 25 | TCP |
| POP3 | 110 | TCP |
| IMAP | 143 | TCP |
| DNS | 53 | TCP et UDP |

Cas particulier du *ping*  
La commande *ping* est basé sur le protocole ICMP qui n’a pas de port prédéfini.

iptables -t filter -A INPUT -p icmp -j ACCEPT

iptables -t filter -A OUTPUT -p icmp -j ACCEPT

**Démarrage du firewall**

Enfin, il faut lancer le pare-feu…

chmod +x /etc/init.d/firewall

/etc/init.d/firewall

Il est important de tester abondamment que tout se passe bien. Il est notamment possible d’utiliser l’utilitaire *nmap* afin de vérifier qu’il n’y a pas plus de ports ouverts que voulu.

Il est important de charger ce script au démarrage de la machine afin qu’un simple reboot ne vous laisse pas sans protection :

update-rc.d firewall defaults

**Exemple de script**

Ci-dessous, un exemple de script de base autorisant le minimum vital pour un serveur web (HTTP, FTP, mail et résolution de DNS).

1 #!/bin/sh

2

3 # Réinitialise les règles

4 sudo iptables -t filter -F

5 sudo iptables -t filter -X

6

7 # Bloque tout le trafic

8 sudo iptables -t filter -P INPUT DROP

9 sudo iptables -t filter -P FORWARD DROP

10 sudo iptables -t filter -P OUTPUT DROP

11

12 # Autorise les connexions déjà établies et localhost

13 sudo iptables -A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT

14 sudo iptables -A OUTPUT -m state --state RELATED,ESTABLISHED -j ACCEPT

15 sudo iptables -t filter -A INPUT -i lo -j ACCEPT

16

17 # Autorise ICMP (Ping)

18 sudo iptables -t filter -A INPUT -p icmp -j ACCEPT

19 sudo iptables -t filter -A OUTPUT -p icmp -j ACCEPT

20

21 # Autorise le protocole SSH

22 sudo iptables -t filter -A INPUT -p tcp --dport 22 -j ACCEPT

23 sudo iptables -t filter -A OUTPUT -p tcp --dport 22 -j ACCEPT

24

25 # Autorise le protocole DNS

26 sudo iptables -t filter -A OUTPUT -p tcp --dport 53 -j ACCEPT

27 sudo iptables -t filter -A OUTPUT -p udp --dport 53 -j ACCEPT

28 sudo iptables -t filter -A INPUT -p tcp --dport 53 -j ACCEPT

29 sudo iptables -t filter -A INPUT -p udp --dport 53 -j ACCEPT

30

31 # Autorise le protocole HTTP

32 sudo iptables -t filter -A OUTPUT -p tcp --dport 80 -j ACCEPT

33 sudo iptables -t filter -A INPUT -p tcp --dport 80 -j ACCEPT

34

35 # Autorise le protocole FTP

36 sudo iptables -t filter -A OUTPUT -p tcp --dport 20:21 -j ACCEPT

37 sudo iptables -t filter -A INPUT -p tcp --dport 20:21 -j ACCEPT

38

39 # Autorise le protocole SMTP

40 iptables -t filter -A INPUT -p tcp --dport 25 -j ACCEPT

41 iptables -t filter -A OUTPUT -p tcp --dport 25 -j ACCEPT

42

43 # Autorise le protocole POP3

44 iptables -t filter -A INPUT -p tcp --dport 110 -j ACCEPT

45 iptables -t filter -A OUTPUT -p tcp --dport 110 -j ACCEPT

46

47 # Autorise le protocole IMAP

48 iptables -t filter -A INPUT -p tcp --dport 143 -j ACCEPT

49 iptables -t filter -A OUTPUT -p tcp --dport 143 -j ACCEPT

50

51 # Autorise le protocole NTP (horloge du serveur)

52 sudo iptables -t filter -A OUTPUT -p udp --dport 123 -j ACCEPT

**Configuration avancée**

Le firewall, outre que c’est l’outil de base de tout système de sécurité, permet des manipulations plus poussées que filtrer des ports. Je vais vous montrer quelques exemples.

**Inondation (flood) ou déni de service**

Ce genre d’attaque vise à surcharger la machine de requête.   
Il est possible de s’en prémunir pas mal directement au niveau du pare-feu.

Ajouter les lignes suivantes au fichier /etc/init.d/firewall…

iptables -A FORWARD -p tcp --syn -m limit --limit 1/second -j ACCEPT

Le drapeau (flag) *TCP syn* engendre des demandes de connexions et le but de la prochaine règle est donc de les limiter à une par seconde (champs limit).  
Il est cependant déconseillé de monter au-delà de la seconde (sous peine de gêner le contrôle de flux et la récupération d’erreur de TCP).

On peut faire de même avec les protocoles UDP et ICMP…

iptables -A FORWARD -p udp -m limit --limit 1/second -j ACCEPT

iptables -A FORWARD -p icmp --icmp-type echo-request -m limit --limit 1/second -j ACCEPT

Il est à noter cependant que ce type d’attaque permet de faire tomber le serveur mais pas d’en prendre l’accès. C’est pourquoi le risque d’en être la cible est assez mince. En général, un logiciel simple anti-intrusion comme *fail2ban* (voir plus loin) sera suffisant.

Balayage de ports

On peut aussi limiter un tant soit peu le scan de ports (qui consiste à tester tous vos ports afin de détecter ceux qui sont ouverts). Pour ce faire une règle de ce genre irait…

iptables -A FORWARD -p tcp --tcp-flags SYN,ACK,FIN,RST RST -m limit --limit 1/second -j ACCEPT

C’est un peu le même principe que ci-dessus. Sachant qu’une connexion TCP en bon et due forme requiert trois paquets avec trois drapeaux différents, on voit tout de suite la finesse de cette règle qui peut travailler paquet par paquet.

Notez que cette règle basique n’est pas très efficace, c’est une protection de base.

**Bannir une IP**

Si on repère dans les journaux (logs) ou autre une adresse IP suspecte, il est possible de la bannir aisément au niveau du pare-feu à partir de la commande…

iptables -A INPUT -s *adresse\_ip* -j DROP

Notez cependant qu’il n’est pas conseillé de bannir les IP à tour de bras.

**Se prémunir contre les intrusions**

Actuellement, le pare-feu va bloquer toutes tentatives de connexions sur les ports fermés. Mais qu’en est-il des ports ouverts ? Afin de contrôler plus précisément ce qui se passe dessus, le pare-feu n’est pas suffisant et il faudra devoir utiliser d’autres outils, appelés IDS (*Intrusion Detection System*) et IPS (*Intrusion Prevention System*).   
Ces deux catégories de logiciels vont - comme leur nom l’indique - surveiller toute tentative d’intrusion sur le serveur.

La démarche qui va suivre va vous montrer comment réagir à chaque étape d’une tentative intrusion classique, à savoir :

1. le scan de port (plus généralement, la collecte d’informations) afin de trouver les vulnérabilités;
2. les attaques simples, témoins d’une faible sécurité;
3. l’intrusion (via des techniques spécialisées);
4. l’installation d’un moyen pour ouvrir une session sur le serveur à volonté   
   (si l’attaquant parvient jusqu’ici avec succès, on peut dire que la machine lui appartient).

***portsentry* (balayeur de ports)**

Cet utilitaire permet de bloquer en temps réel la plupart des scans de port connus (même très discrets et échappant aux règles de filtrage du firewall basiques).   
Toutefois, il ne faut pas paniquer si un serveur est la cible d’un simple balayage de port, cela sera monnaie courante, et si le serveur est bien protégé, le pirate passera sa route.

*portsentry* est donc utile si on veut compliquer la tâche de l’attaquant…

Apt-get update

Apt-get upgrade

apt-get install portsentry

Pour le configurer…

nano /etc/portsentry/portsentry.conf

* Dans la section TCP wrappers…  
  Mettre en commentaire les lignes *KILL\_HOSTS\_DENY="ALL: $TARGET$ "*
* Dans la section Dropping routes…  
  Décommenter la ligne *KILL\_ROUTE="/sbin/iptables -I INPUT -s $TARGET$ -j DROP"*

Ainsi, *portsentry* ajoutera une règle dans le pare-feu (iptables) pour rejeter les paquets en cas de scans.

Il faut redémarrer portsentry (il faut le lancer deux fois, pour TCP et UDP) :

portsentry –audp

portsentry –atcp

Vous pouvez tester tout ça avec *nmap* (si vous voulez tester en local, il vous faut modifier le fichier *portsentry.ignore* en enlevant le *localhost*).

Si vous souhaitez que vos réglages restent même après un nouveau lancement de portsentry, vous devrez modifier le fichier */etc/portsentry/portsentry.ignore.static*.

***fail2ban* (brute-force, dictionnaire, déni de service)**

Les ports ouverts sur un hôte sont à priori sans grande protection et sujet à des attaques simples telles que la tentative de connexion par brute-force ou par dictionnaire (par exemple, tester toutes les combinaisons de mots de passe pour se ouvrir une session avec ssh), le déni de services (surcharger le serveur de requêtes) ou - plus bêtement - la recherche d’utilisateurs sans mot de passe... Si un hôte est infiltrée aussi facilement l’attaquant sera vraiment content.

*fail2ban* est un petit utilitaire qui se base sur les journaux de l’hôte afin de chercher des actions suspectes répétées (par exemple, des erreurs de mot de passe) dans un laps de temps donné. S’il en trouve, il bannira l’adresse IP de l’attaquant via *iptables*. Ce type de logiciel est indispensable, car, bien que léger, il offre une bonne protection contre les attaques basiques indiquées ci-dessus.

Pour l’installation …

Apt-get update

Apt-get upgrade  
apt-get install fail2ban

Pour la configuration …

nano /etc/fail2ban/jail.conf

Le niveau de protection peut être modulé via les champs suivants (notez que la configuration par défaut suffit normalement):

* **bantime**  
  temps de bannissement des IP suspectes;
* **maxretry**  
  nombre de tentatives de connexion permise avant bannissement.

Dans la partie JAILS figure tous les services que fail2ban surveillera.   
Si on a modifié les ports par défaut, il faut les indiquer là aussi.   
Par exemple avec ssh :

nano /etc/fail2ban/jail.conf

ctrl+w => chercher [ssh]

port : indiquer le port

Enregistrez et quittez.

Enfin, pour recharger la nouvelle configuration …

/etc/init.d/fail2ban restart

***snort* (détection d’intrusions)**

Le problème quand on commence à sécuriser est de savoir s’arrêter à un moment donné. *snort* est un outil très puissant, pouvant en fait détecter la plupart des attaques qui échapperaient à un utilitaire comme *fail2ban*.   
Bien entendu, il ne servira pas dans 90 % des cas et comme ce n’est qu’un outil de détection, il faudra prendre les mesures nécessaires s’il détecte une intrusion.   
Enfin, comme il analyse le trafic en temps réel, cela ralentit forcément un peu les flux.   
L’installer n’est donc pas indispensable, cela dépend du degré de sécurité recherché !

Voici deux sites comme référence…

* http://www.trustonme.net/didactels/187.html
* http://doc.ubuntu-fr.org/snort

***rkhunter* (rootkit et backdoors)**

Dernier volet de cette section intrusion, les *backdoors*.   
Si par malheur un attaquant arrive à prendre possession d’un hôte, il y a fort à parier qu’il y laisse une porte dérobée qui lui permettrait d’en reprendre le contrôle plus tard, ainsi qu’un *rootkit* pour la dissimuler: l’attaquant maintient ainsi un accès frauduleux à cet hôte.

*rkhunter* est un utilitaire qui est chargé de détecter d’éventuels *rootkits* sur un serveur. Il est relativement léger (s’exécute une fois par jour par défaut).

Pour l’installer …

apt-get update

apt-get upgrade

apt-get install rkhunter

Il est conseillé de modifier un peu la configuration …

nano /etc/default/rkhunter  
  
REPORT\_EMAIL: indiquez un mail pour recevoir des alertes de Rkhunter;  
CRON\_DAILY\_RUN: mettez « yes » pour une vérification quotidienne de l’hôte via cron.

Notez que *rkhunter* se trompe parfois en déclarant comme infectés des fichiers sains (« faux positifs »), donc il faut être critique à l’égard des rapports. Par contre, s’il s’avère que l’alerte est justifiée, cela signifie que l’on a affaire avec un *rootkit* ainsi qu’une faille de sécurité qui a été découverte et exploitée. Méfiance donc !

**Surveiller les logs**

La plupart des logiciels cités plus haut enverront des notifications par courriel en cas d’alerte. Cependant, surveiller les journaux est important, car ils reflètent la « vie » du serveur. Les journaux les plus intéressants sont notamment:

* /*var/log/auth.log*   
  qui contient toutes les tentatives d’accès au serveur.   
  Il peut être utile de filtrer le contenu, par exemple:   
  cat /var/log/auth.log | grep authentication failure;
* */var/log/message* et */var/log/syslog*   
  contienent un peu de tout (erreurs, bugs, informations, etc);
* */var/log/fail2ban*   
  est le log d’alerte de *fail2ban*. Cherchez notamment :   
  cat /var/log/fail2ban | grep ban;
* */var/log/snort/alert* vous indiquera les logs d’alertes de *snort*;
* */var/log/rkhunter* pour voir les rapports quotidiens de *rkhunter*.   
  Faites attention aux erreurs trouvées, ce n’est pas bon signe (même si le risque de faux positifs existe ici).

***logwatch***

Il est aussi possible d’utiliser des utilitaires qui vous simplifient un peu ce travail de lecture des logs. logwatch notamment permet de résumer plusieurs logs afin de ne vous retourner que des anomalies si possible. Cela évite un long et fastidieux travail de recherche.

Pour l’installer …

apt- get update

apt-get upgrade

apt-get install logwatch

Pour le configurer …

nano /usr/share/logwatch/default.conf/logwatch.conf

Spécifiez l’option « MailTo » car logwatch envoie ses résumés de logs par mail.

Il va normalement s’exécuter tous les jours (ls -l /etc/cron.daily/ | grep logwatch pour s’en assurer).

Il peut aussi être intéressant de suivre l’état du réseau et du système (monitoring) afin de détecter par exemple une brusque montée en charge, synonyme de problèmes.

**Du bon usage de son serveur**

Le bon sens …  
90 % des problèmes informatiques relèvent de l’utilisateur. C’est pourquoi, avant même de penser à sécuriser sa machine, il faut garder en mémoire quelques règles de bon sens:

* interdire les utilisateurs sans mot de passe (ce sont d’énormes failles potentielles);
* toujours choisir de bons mots de passe:   
  8 caractères minimum,   
  pas de mot qui se trouve dans le dictionnaire,   
  si possible des chiffres, des majuscules, des symboles...   
  au besoin, un outil comme *pwgen* vous en génèrera automatiquement des bons (apt-get install pwgen);
* maintenir son système à jour (apt-get update et apt-get upgrade);
* toujours utiliser *ssh* pour l’accès à distance (et non *telnet* ou des services graphiques, sauf s’ils sont en tunnel à travers *ssh*).

**Configurer les logiciels**

Il peut aussi être bon d’améliorer un peu la sécurité des logiciels installés sur un hôte car il seront en première ligne pour traiter les paquets autorisés.

*ssh*

En premier lieu, il faut regarder du côté de *ssh*, puisque c’est tout de même un accès direct à votre machine.

Pour ce faire...

nano /etc/ssh/sshd\_config

... et il est conseillé de changer les champs suivants :

* **Port** : le port par défaut est 22... et n’importe quel attaquant le sait.   
  Modifier le port force à effectuer un balayage (ou équivalent) avant de réfléchir à attaquer (attention de bien changer le port au niveau du pare-feu);
* **PermitRootLogin**  
  Mettre à « no » afin d’interdire le login en root;
* **AllowUsers**   
  ndique une liste d’utilisateur autorisé à se connecter via ssh.   
  Cela peut être utile si on a des utilisateurs qui ne sont pas censés se connecter sur l’hôte.

Et on redémarre …

/etc/init.d/ssh restart

1. Pour activer la redirection d'IP, il est également possible d’exécuter la commande suivante :

   **sysctl -w net.ipv4.ip\_forward=1**

   Si cette commande est exécutée via une invite du shell, le paramètre est alors oublié après un redémarrage.

   Il est aussi possible de définir la retransmission de façon permanente en éditant le fichier ***/etc/sysctl.conf***. Il faut alors trouvez et modifiez la ligne suivante, en remplaçant 0 par 1 :

   **net.ipv4.ip\_forward = 0**

   Enfin, il faut exécuter la commande suivante pour activer les changements dans le fichier ***sysctl.conf*** :  
   **sysctl -p /etc/sysctl.conf** [↑](#footnote-ref-1)
2. Les commandes suivantes peuvent être aussi utilises :  
   **iptables --flush   
   iptables --table nat –flush  
   iptables --delete-chain** [↑](#footnote-ref-2)