Sécurité des réseaux informatiques

Partie 1 :

Principes et techniques de sécurité

Généralités et bases de cryptographie

2015-2016 Virginie Van den Schrieck

Table des matières

Préambule

Ce syllabus est destiné à servir de support au cours de Sécurité des Réseaux informatiques en 3ème Technologie de l'Informatique. Ce cours a été concu sur base du livre de référence de Stallings and Brown, Computer Security, Principles and Practice [3]. Dans sa version actuelle, ce syllabus reprend la structure et les éléments de ce livre correspondant au cahier des charges du cours, éléments qu'il traduit, synthétise et enrichit d'apports extérieurs.

Les étudiants intéressés par un tour d'horizon plus complet et plus détaillé que ce qui est vu dans le cadre de ce cours sont donc invités à consulter l'ouvrage d'origine ainsi que les autres références mentionnés en bibliographie.

Chapitre 1

Généralités

L'objectif de cette section introductive est d'identifier les concepts regroupés sous le terme « sécurité informatique ». Tout d'abord, nous réfléchirons à ce qu'est la sécurité informatique, et sur quels éléments (actifs) elle s'applique. Nous examinerons dans quelle mesure ces actifs sont menacés et par quoi, et comment il est possible de contrer ces menaces.

1.1 Concepts de sécurité informatique

Le livre NIST Computer Security Handbook [1] définit la sécurité informatique comme suit :

Définition

La **sécurité informatique** est la protection fournie à un système d'information automatisé pour atteindre les objectifs de préservation de l'intégrité, de disponibilité et de confidentialité des ressources du système d'information (à savoir les éléments matériels, logiciels, les firmwares, les données et les télécommunications).

Trois objectifs clés apparaissent dans cette définition [2][3] :

1. **La confidentialité** : Sous ce terme se cachent deux éléments : d'une part, la confidentialité des données, qui assure que les informations privées ou confidentielles ne sont pas dévoilées à des personnes non autorisées, et d'autre part, la protection de la vie privée, qui assure que chacun peut contrôler la manière dont ses informations personnelles sont diffusée. En résumé, la confidentialité assure que l'information n'est connue que des entités communicantes.
2. **L'intégrité** : D'une part, l'intégrité des données assure que les informations et les programmes ne sont modifiés que de manière autorisée et contrôlée, d'autre part, l'intégrité des systèmes assure que le système

fonctionne de manière conforme, sans manipulation non autorisée, délibérée ou non. En résumé, l'objectif d'intégrité assure que l'information n'a pas été modifiée entre sa création et son traitement.

3. **La disponibilité** : Assure que le système fonctionne de manière adéquate et que le service est bien à disposition des utilisateurs. L'information est toujours accessible et ne peut pas être perdue.

Ces trois objectifs sont regroupés dans le modèle du **triangle CIA** (voir figure 1.1), un de modèles les plus connus en raison de sa simplicité et de ses très larges possibilités d'application.

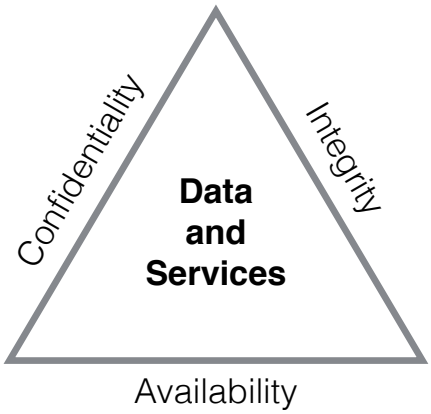

FIGURE 1.1 – Triangle CIA

Bien que ces trois concepts forment une base largement reconnue, on leur associe souvent des concepts additionnels pour compléter le tableau. Par exemple :

- **L'authenticité** : Le fait d'être authentique et digne de confiance, de manière vérifiable. Cela couvre la confiance en la validité d'une transmission, d'un message, ou en la source d'un message, et implique la vérification de l'identité des utilisateurs ainsi que l'origine des données arrivant dans le système.
- **La tracabilité** (accountability) : Il faut pouvoir tracer les actions pour identifier celui ou ce qui en est à l'origine. Cela comprend par exemple la non répudiation, l'isolation de failles, la détection et la prévention d'intrusions, etc. Puisque les systèmes ne peuvent être entièrement sécurisés, il faut pouvoir tracer l'origine d'une faille de sécurité jusqu'à son responsable.

1.2 Terminologie

Nous allons à présent définir les termes principaux utilisés tout au long du cours.

Définition

Une **ressource**, ou actif (asset en anglais), est un élément que les utilisateurs et les propriétaires souhaitent protéger.

Les ressources informatiques peuvent être de différents types :

- **Hardware** : Les ordinateurs et les appareils de traitement ou de stockage des données, ainsi que les appareils de communications
- **Software** : Les applications, les utilitaires systèmes et les systèmes d'exploitation
- **Les données** : Les fichiers et les bases de données contenant les informations métier, ainsi que les informations liées au processus de sécurité lui-même (ex : fichier de mots de passe)
- **Les réseaux et moyens de communications** : Les liens des réseaux LAN et WAN, les routeurs, les switches, etc.

Il y a également lieu de tenir compte des ressources non informatiques : Les **individus** et, également, **la réputation** : réputations des personnes, mais aussi réputation de l'entreprise.

Pour protéger ces ressources, il y a lieu de mettre en place une politique de sécurité :

Définition

Une **politique de sécurité** est un ensemble de règles et de pratiques qui spécifie ou régulent la manière dont un système ou une organisation fournit des services de sécurité pour protéger les ressources critiques et sensibles du système.

Une politique de sécurité est donc composée de contre-mesures techniques, mais également d'un ensemble de règles et de procédures à l'intention de tous les intervenants dans le système.

La permière étape dans une stratégie de sécurité, après l'inventaire des ressources à protéger, est d'identifier les points faibles de ces ressources :

Définition

Une **vulnérabilité** est une faille ou une faiblesse dans le conception, l'implémentation, l'opération ou la gestion d'un système qui peut être exploitée pour violer la politique de sécurité du système.

Ces vulnérabilités peuvent être classifiées en plusieurs catégories :

- **La corruption d'un système** : le système effectue de mauvaises opérations ou donne de mauvaises réponses (exemple : corruption d'une base de données).
- **Les fuites** : Par exemple, une personne peut réussir à obtenir des informations auxquelles elle n'est pas autorisée à accéder.

– **Les indisponibilités** : un système peut devenir inaccessible ou trop lent pour être utilisable.

Remarquez que chacune de ces vulnérabilités correspond à la mise en défaut d'un des objectifs de sécurité du triangle CIA.

A chaque vulnérabilité correspond une **menace** qui est capable de l'exploiter.

Définition

Une **menace** représente un danger potentiel de sécurité pour une ressource, par l'exploitation d'une vulnérabilité

Définition

Une **attaque** est la réalisation d'une menace. Il s'agit d'une tentaive délibérée d'éviter un service de sécurité et de violation de la politique de sécurité d'un système.

Les attaques peuvent être classifiées de différentes manières. Elles peuvent être **actives**, c'est-à-dire que l'attaque essaie d'altérer le système et ses opérations, ou **passives**, lorsque l'attaquant essaie d'obtenir ou d'utiliser des informations du système sans affecter les ressources. On peut également les classer selon leur origine : Les attaques **internes** viennent d'une entité à l'intérieur du périmètre de sécurité, par un attaquant disposant d'un accès autorisé aux ressources, mais l'utilisant à mauvais escient. Les attaques **externes** viennent de l'extérieur du périmètre de sécurité, et sont par un attaquant ne disposant pas d'accès autorisé.

Pour contrer les attaques, il faut mettre en place des **contre-mesures**, afin de prévenir, de détecter, ou, le cas échéant, de récupérer de chaque type d'attaque.

Définition

Une **contre-mesure** est une action, un appareil, une procédure ou une technique qui réduit une menace, une vulnérabilité ou une attaque l'éliminant ou en la prévenant, en minimisant le danger qu'elle cause ou encore en la découvrant et en la rapportant de manière à ce qu'une action corrective puisse être prise.

Enfin, le dernier terme à définir est le concept de risque.

Définition

Un risque est la perte attendue, exprimée en terme de la probabilité pour qu'une menace spécifique exploite une vulnérabilité particulière avec un résultat négatif particulier.

Les risques diffèrent bien sûr selon les systèmes, et selon les catégories d'utilisateurs. Ainsi, un fournisseur d'accès Internet n'aura pas la même définition des risques qu'un simple utilisateur. Leurs actifs sont différents, leurs enjeux sont différents, donc leurs définition des risques seront différentes. Les risques peuvent porter, entre autres, sur les abus d'utilisation, sur la conformité, sur les données, sur les humains, sur le matériel, le logiciel ou encore la réputation. L'analyse du risque de sécurité est donc une procédure délicate, dans laquelle interviennent beaucoup de facteurs.

Une manière d'estimer le risque est de prendre en compte ces facteurs multiples dans la formule suivante :

Risque = Probabilité d'un incident x Impact

L'impact de l'incident est mesuré en termes des trois objectifs de sécurité, c'est-à-dire la confidentialité, l'intégrité et la disponibilité (CIA).

La sécurité absolue n'existe pas, et l'existence de toute entreprise repose sur une prise de risques, plus ou moins contrôlée. Dans le cadre de la sécurité, il s'agit de diminuer les risques mais en tenant compte du contexte budgétaire : Le coût des mesures de sécurité ne doit pas dépasser la valeur des biens à protéger.

1.3 Menaces, attaques et actifs

Le RFC4949 [4] liste quatre types de conséquences liés à des menaces, et liste les attaques qui en résultent.

1.3.1 La divulgation non autorisée (unauthorised disclosure)

La divulgation non autorisée d'information est une menace à l'objectif de confidentialité. Quatre attaques peuvent mener à cette situation :

1. **L'exposition (exposure)** : Il s'agit de la mise à disposition , délibérée ou non, d'informations sensibles à des personnes n'ayant pas l'autorisation d'y accéder (par exemple : publication sur le web de numéros de carte de crédits).
2. **L'interception** : Une personne ou un système non autorisé intercepte des informations lors de leur transmission entre source et destination

3. **L'inférence** : Un attaquant réussit à accéder à des informations sensibles de manière indirecte, en raisonnant sur les caractéristiques des communications.
4. **L'intrusion** : Un attaquant accède à des informations sensibles en coutournant les protections de sécurité du système.

1.3.2 La tromperie (deception)

La tromperie consiste à transmettre de fausses informations à une entité autorisée et à lui faire croire qu'il s'agit de données conformes. Les types d'attaques correspondantes sont :

1. **Masquerade** : Un attaquant obtient accès à un système et peut effectuer des opérations malicieuses en se faisant passer pour une entité autorisée.
2. **Falsification** : il s'agit d'altérer ou de remplacer des données valides, ou encore d'introduire de fausses données dans un fichier ou une base de données.
3. **Répudiation** : Un utilisateur nie avoir reçu, posséder ou avoir envoyé des données.

1.3.3 Les perturbations (disruption)

Les perturbations sont une menace pour la disponibilité et/ou l'intégrité des systèmes. Elles peuvent être causées par les attaques suivantes :

1. **Neutralisation (incapacitation)** : Cette attaque mène à la destruction physique ou à l'endommagement du matériel, et nuisent à la disponibilité du système. Il s'agit typiquement d'attaques basées sur des logiciels malicieux (malware) tels que les chevaux de Troie, les virus ou les vers, qui désactivent un système ou, au miminum, certains de ses services.
2. **Corruption** : Cette attaque cible l'intégrité d'un système. Les malwares agissant dans cette optique s'arrangent pour que le système attaqué fonctionne d'une manière non prévue. Un utilisateur ayant usurpé des droits d'accès peut également accéder et modifier un système, en plancant par exemple des portes dérobées.
3. **Obstruction** : Cette attaque interrompt un service en perturbant les opérations du système. Cela peut se faire en coupant les liens de communications par exemple, ou en surchargeant le système par un excès de trafic réseau.

1.3.4 Usurpation

Une situation d'usurpation apparait quand une entité non autorisée prend le contrôle d'un service ou des fonctions d'un système. Il s'agit d'une menace à l'intégrité du système. Les attaques qui mène à cette situation sont :

1. **Détournement (misappropriation)** : Une entité contrôle sans autorisation, de manière logique ou physique, les ressources d'un système. Un exemple d'une telle attaque est l'attaque de déni de service distribué (DDoS), où des malwares installés sur des machines sont utilisés comme plateformes de lancement d'attaques.
2. **Mauvais usage (misuse)** : Cette attaque a pour conséquence qu'un composant d'un système effectue une opération ou un service au détriment de la sécurité du système. Cela arrive par exemple lorsqu'un hacker ou un malware a accès à un système et peut désactiver les fonctions de sécurité.

1.3.5 Menaces et types d'actifs

Les menaces listées ci-dessus s'appliquent à des actifs (assets en anglais), ou ressources informatiques. Ces actifs peuvent être classifiés en quatre catégories : Hardware, software, données ou bien lignes et réseaux de communications. Chaque catégorie d'actifs est plus ou moins impactée par un type d'attaque, lié donc à un objectif de sécurité. Ainsi, le matériel est particulièrement sensible aux attaques visant la disponibilité, telles que le vol ou la destruction accidentelle ou non. Pour les données, bien qu'elles soient vulnérables sur les trois aspects, l'emphase est souvent mis sur la confidentialité. Les réseaux, quant à eux, sont sensibles d'une part à des attaques passives (observation du contenu, analyse du traffic), d'autre part aux attaques actives, via le reju, les modifications de messages, le déni de service, ...

Le tableau 1.1 fournit, pour chaque type d'actifs, des exemples d'attaques menaçant chacun des objectifs du triangle CIA.

1.4 Principes de conception de systèmes sécurisés

Malgré des années de recherche et développement, il n'existe toujours pas à l'heure actuelle de techniques de conception et d'implémentation qui excluent les failles de sécurité et empêchent les actions non autorisées. Il est néanmoins possible de dresser une liste de principes de conception reconnus permettant de guider le développement de mécanismes de protection. De tels principes sont listés dans [5], et nous les reprenons ici. Les intitulés sont en anglais, car souvent difficilement traduisibles.

– **Economy of mechanisms** : Les systèmes complexes sont ceux qui, logiquement présentent le plus d'opportunités de failles. Il est donc

	Disponibilité	**Confidentialité**	**Intégrité**
Hardware	Vol ou destruction d'équipement	Vol d'un CD ou d'un DVD non encrypté	
Software	Suppression d'un programme, bloquant l'accès aux utilisateurs	Copie non autorisée du logiciel	Modification d'un programme, le faisant crasher ou modifiant son comportement
Données	Suppression de fichiers	Accès non autorisé aux données, ou analyse statistique révélant des données sous-jacentes	Modification de fichiers existants ou création de données
Réseaux et communications	Destruction de messages ou blocage des moyens communications	Lecture des messages ou observation des caractéristiques de trafic	Modification de messages, réordonnancement, duplication ou fabrication de messages

TABLE 1.1 – Exemples d'attaques selon les types d'actif et l'objectif de sécurité

conseillé de toujours rechercher des solutions simples, claires et de taille restreintes, pour limiter justement ces risques, mais également pour simplifier les problèmes de gestion des configurations ou des mises à jour. Ce principe rejoint celui rasoir d'Occam, ainsi que le principe KISS (Keep It Simple et Stupid), variante de l'adage « Pourquoi faire compliqué quand on peut faire simple ».

- **Fail-safe default** : Toutes les décisions d'accès devraient adopter l'exclusion comme comportement par défaut, comportement par dessus lequel ont rajoutées, au cas par cas, les permissions spécifiques. Une erreur dans un tel système résulte en un refus d'accès, qui est simple à identifier et donc à solutionner. A l'inverse, avec l'acceptation par défaut, une erreur autorisant un accès non autorisé aura tendance à passer inaperçue, et le risque couru est plus grand.
- **Complete mediation** : Chaque accès à un système doit être validé auprès du mécanisme de contrôle d'accès. Cela implique que les systèmes doivent éviter l'utilisation de caches pour ce type de décision. Un exemple typique violant ce principe est celui du contrôle d'accès aux fichiers : Une fois qu'un fichier a été ouvert, il n'y plus de vérification ultérieure des permissions, alors que cela devrait être fait à chaque accès, donc à chaque lecture ou écriture.
- **Open design** : Les éléments de conception d'un système de sécurité devraient être publiques plutôt que secrètes. Par exemple, les algorithmes cryptographiques doivent être connus, ce sont les clés d'en-

cryption, elles gardées secrètes, qui assurent la sécurité des systèmes cryptographiques.

- **Separation of privilege** : Lors de la conception d'un système, lorsque les différents éléments ont des besoins d'accès différents à des ressources privilégiées, il faut séparer ces éléments afin de leur attribuer des privilèges séparés.
- **Least privilege** : Ce principe est complémentaire du précédent. Il spécifie que chaque élément d'un système, processus ou utilisateur ne doit bénéficier que des privilèges minimaux suffisants pour effectuer sa tâche.
- **Least common mechanism** : Ce principe stipule que, dès la conception, il faut minimiser les fonctionnalités partagées par les différents utilisateurs ou les différents éléments d'un programme. Par exemple, si un programme corrompt un élément partagé (cela peut être le système de fichiers), tous les programmes sont impactés.
- **Psychological acceptability** : Les mécanismes de sécurité ne doivent pas interférer sans raison avec les tâches de l'utilisateur, tout en remplissant les demandes en termes de contrôle d'accès. Un mécanisme de sécurité qui complique la tâche de l'utilsateur, ce dernier pourrait tout simplement l'ignorer ou le désactiver. Le mécanisme de sécurité ne doit donc pas faire obstruction à l'utilisateur, et, idéalement, correspondre à l'idée qu'à l'utilisateur du besoin en sécurité, afin que ce dernier l'accepte.
- **Isolation** : Ce principe s'applique dans trois contextes.
  1. Les systèmes à accès publique doivent être isolés des systèmes critiques
  2. Les processus et les fichiers des utilisateurs doivent être isolés de ceux des autres, à moins que le partage ne soit explicitement demandé.
  3. Les accès aux mécanismes de sécurité doivent également être isolés.
- **Encapsulation** : L'encapsulation est une forme d'isolation appliquées aux fonctionnalités orientées-objet. En limitant et en contrôlant la manière dont on accède, interagit et modifie les objets, on s'assure de protéger la structure interne de l'objet.
- **Modularity** : Dans le cadre de la sécurité, cela fait référence au développement séparé de modules protégés (cfr principe du Least Common Mechanism), qui est complémenté par le partage des fonctions et des services de sécurité, comme par exemple les librairies cryptographiques. Il est plus sûr de disposer d'une seule implémentation partagée (et validée) d'un protocole de sécurité, plutôt que de devoir développer de multiples versions du protocole, ce qui multiplie le risque d'apparition de failles.

- **Layering** : Ce principe stipule d'utiliser plusieurs couches de protection afin qu'une panne, une attaque ou un manquement à la sécurité au niveau d'une couche ne laisse pas le système sans protection. Ce principe est également appelé « défense en profondeur ».
- **Least astonishment** : Ce principe stipule qu'un programme ou une interface utilisateur doit toujours se comporter conformément à l'idée que s'en fait un utilisateur. Par exemple, un mécanisme d'autorisation doit être suffisament transparent pour que l'utilisateur ait une bonne compréhension des objectifs de sécurité auxquels le mécanisme répond.

1.5 Stratégies de sécurité informatique

Lorsqu'il s'agit de mettre au point une stratégie de sécurité, il faut considérer trois étapes :

1. **Définir** la politique de sécurité : Qu'est-ce que le mécanisme de sécurité doit faire ?
2. **Implémenter** cette politique : Quels mécanismes mettre en place ?
3. **Valider** cette politique : Comment s'assurer que le mécanisme fonctionne et correspond à la politique de sécurité demandée ?

1.5.1 Politiques de sécurité

Pour définir une politique de sécurité, il faut considérer plusieurs facteurs. D'une part, il faut définir les biens à protéger, et leur valeur. Un inventaire régulier des avoirs est donc indispensable, aussi bien pour garantir leur existence que pour assurer le bon fonctionnement des mesures mises en place. L'inventaire reprendra aussi idéalement, pour chaque avoir, le nom d'une (et une seule) personne responnable.

La valeur des avoirs peut être définie en termes des trois objectifs de sécurité (CIA), comme nous l'avons vu plus haut.

Ensuite, il faut soigneusement identifier les vulnérabilités du système, les attaques potentielles qui les exploitent, et enfin, la probabilité d'occurence de ces attaques et l'impact qu'elles auront. La probabilité d'occurence peut par exemple se mesurer en terme de fréquence d'incident : Une attaque se répétant quotidiennement est évidemment plus probable qu'une attaque survenant tous les dix ans. Cette mesure doit être combinée à l'impact qu'aura l'attaque, c'est-à-dire le coût du sinistre lorsqu'il se produit. Certaines attaques peu probables peuvent avoir des impacts catastrophiques mettant en cause l'existence même de l'entreprise.

Cette analyse combinée de la probabilité et de l'impact des menaces permet de donc définir les risques, comme nous l'avons déjà vu plus haut. Il est important d'être exhaustif, car même une attaque peu probable aujourd'hui peut constituer un risque réel plus tard.

Pour chaque vulnérabilité identifiée, il est possible de mettre en place une contre-mesure. Cependant, le choix des mécanismes à mettre en place nécessite de faire des compromis :

- d'une part, **entre la facilité d'utilisation et la sécurité** : les mécanismes de sécurité impactent souvent la convivialité d'utilisation des systèmes informatiques. Ainsi, les contrôles d'accès imposent, entre autres, aux utilisateurs de retenir des mots de passe, ou encore, les firewalls peuvent impacter la qualité des communication et empêcher le passage de certains types de traffic.
- d'autre part, **entre le coût de la sécurité et le coût des pertes et de la procédure de récupération** en cas de panne ou d'attaque.

La politique de sécurité constitue donc un **point de décision stratégique important** pour une entreprise. Et puisque chaque choix est un renoncement, la politique de sécurité implémentée ne peut souvent pas couvrir l'ensemble des risques identifiés. La mise en place d'une politique de sécurité nécessite de faire le choix des risques contre lesquels il faut se prémunir, ce qui a logiquement pour conséquence d'établir des **risques résiduels**, pour lesquels il n'est pas possible d'agir, financièrement ou matériellement. Il est donc tout aussi important d'identifier et de documenter ces risques résiduels.

1.5.2 Mise en place des contre-mesures

L'implémentation des mécanismes de sécurité implique quatre groupes d'actions complémentaires :

1. **La prévention** : Un mécanisme de sécurité idéal est celui qui ne laisse passer aucune attaque. Cela n'est bien sûr pas possible en réalité, mais la prévention permet néanmoins de restreindre de manière efficace le nombre d'attaques potentielles.
2. **La détection** : Puisqu'il n'est pas toujours possible d'empêcher des attaques, les responsables de la sécurité doivent tout faire pour pouvoir détecter les attaques dès que possible. C'est par exemple le rôle des systèmes de détection d'intrusion.
3. **La réaction** : Une fois une attaque détectée, le système doit pouvoir réagir de manière adéquate à l'attaque pour interrompre cette dernière et minimiser les pertes subies.
4. **La récupération** : Puisque les pertes sont parfois inévitables, il est important de définir permettant de récupérer ce qui a été perdu. Les systèmes de backup permettent ainsi la récupération de donées dont l'intégrité a été compromise.

1.5.3 Validation

Lors de la mise en place d'un système de sécurité, son utilisateur suppose qu'il effectue bien les fonctions de protection qu'il prétend assurer. Cette sup-

position se base sur les spécifications données par le fabriquant. Néanmoins, il ne faut jamais rien prendre pour acquis, et il est nécessaire d'évaluer et de valider la manière dont le système fonctionne.

En plus de l'analyse et de la conception de la politique de sécurité et en plus de l'implémentation de celle-ci, il est tout aussi important de **définir des procédures d'évaluation** afin de valider le bon fonctionnement des contre-mesures mises en place.

1.6 Connaître son ennemi

Dans le cadre d'une analyse de sécurité, il est important de comprendre l'adversaire, de réfléchir comme lui, de connaître sa motivation, ses méthodes, ses moyens afin de pouvoir s'en protéger efficacement. Cette section est largement inspirée de [6], chap. 1, section 8.

1.6.1 Les types d'attaquants

Les attaques automatisées

Parmi les attaques automatisées figurent les virus, trojans et autres vers circulant sur Internet. Ces attaques exploitent des vulnérabilités qui, souvent, ont déjà été répertoriées mais pour lesquelles des mesures correctives appropriées n'ont pas encore été appliquées. Il importe donc de se tenir à jour en suivant les publications de vulnérabilités, de mettre à jour les systèmes avec les correctifs de sécurité, de tenir les signatures des antivirus à jour, de détecter les comportement anormaux et bloquer la propagation de ces attaques.

Un bon suivi permet donc de réduire fortement le risque liés à ces vulnérabilités documentées, mais il reste toujours le risque lié aux attaques non encore documentérs (Zero Day Exploit).

Les Script Kiddies

Parmi les attaquants potentiels figurent des « amateurs plus ou moins éclairés », étudiants en mal d'expérimentation, adolescents passionés, etc., qui expérimentent divers scripts et méthodes permettant d'attaquer des sites. Ils utilisent soit des scripts existants, soit développent leurs propres solutions. Indépendamment de leur niveau de compétences, ils partagent une stratégie commune, à savoir la recherche et l'exploitation de vulnérabilités spécifiques sur des cibles aléatoires. C'est cette recherche aléatoire qui est dangereuse.

Le vandale

Le vandale est un attaquant qui, pour le plaisir, pour montrer ses capacités ou pour se forger une réputation, n'hésitera pas à détruire la cible qu'il

aura réussi à attaquer.

L'expert

L'expert tente de mettre en évidence les risques liés à l'utilisation d'Internet en réalisant des intrusion. Il ne cherche pas à détruire, et ne recherche pas le profit direct. Parfois, ils finissent par après par se trouver un emploi dans le domaine de la sécurité.

L'activiste

L'activiste est un attaquant expérimenté qui choisit ses cibles sur base d'une motivation politique.

L'espion

Il s'agit d'un professionnel des intrusions disposant d'outils et de méthodes sophistiqués, qui cherche à capturer de l'information sans laisser de traces. Son plan d'attaque peut se dérouler sur de longues périodes.

Le cybercrime organisé

Le cybercrime est constitué d'organisations d'attaquants bien organisés poussés par des motivations financières. Ces organisations cherchent par exemple à récupérer des informations liées à des cartes de crédit.

Le cyberterrorisme

Il s'agit là aussi d'organisation regroupant des experts bien organisés, dont l'objectif est cette fois le terrorisme, donc la destruction.

1.6.2 Méthodes d'attaque

Les attaquants travaillent de deux manières différentes : Soit ils utilisent des cibles aléatoires, soit des cibles spécifiques. Les deux manières peuvent être complémentaires. Typiquement, les script kiddies ou les automates préféreront les cibles aléatoires, puisque leur but est de trouver une cible, quelle qu'elle soit, pour en exploiter les vulnérabilités. Une fois la cible compromise, elle peut être utilisée par l'intermédiaire de portes dérobées pour mener d'autres attaques, éventuellement contre des cibles spécifiques cette fois. Les cibles aléatoires servent donc pour les attaques par rebond.

Les attaques visant des cibles spécifiques nécessitent une préparation minutieuse. La première phase consiste en effet à récolter un maximum d'information utiles : DNS, WHOIS, rapports annuels des soiété, annuaires, mais aussi le social engineering, c'est-à-dire l'extraction d'informations sensibles

auprès de personnes disposant de droits d'accès (ex : employés, secrétaires, ...).

1.6.3 Outils privilégiés

Les outils les plus facilement accessibles pour le recueil d'informations sur des cibles potentielles sont les suivants :

- WHOIS : Il s'agit d'un service de recherche offert par les RIR permettant d'obtenir des informations sur les IP, les noms de domaine, ainsi que sur les personnes responsables.
- Mailing list : Certains groupes de discussion permettent la consultation des archives de la mailing list, parmi lesquelles se retrouvent parfois des descriptifs détaillés de problèmes contenant des informations techniques précises
- Social engineering : Les help desk, les personnes de contact ou les responsables dont le nom est repris dans les rapports annuels sont des cibles potentielles pour l'obtention de renseignements
- Scanning de port : Les outils de scanning de port permettent de lister les services ouverts sur des machines en fonction de la manière dont elles répondent à des messages TCP spécifiques (ouverture de connexion par ex.). Les scans automatisés à grande échelle peuvent être facilement repérés, mais il est également possible de les ralentir pour passer au travers des mailles des filets de protection (IDS, firewall, ...). `sscan` et `nmap` sont deux outils parmi d'autres qui permettent de faire du scan de port.
- OS Fingerprinting : Les outils de fingerprinting permettent, sur base de la manière dont répond un système à des sollicitations, d'identifier le type d'OS et les services qui y tournent ainsi que leurs versions. En comparant ces données avec les failles connues, il est possible de cibler les attaques sur les vulnérabilités spécifiques de la machine.

Chapitre 2

Bases de cryptographie

Parmi les outils les plus importants en sécurité informatique se trouvent les algorithmes cryptographiques. Nous allons ici explorer rapidement les principaux algorithmes disponibles et étudier leurs champs d'application respectifs et leurs propriétés, sans pour autant creuser les aspects mathématiques sous-jacents qui sortent du cadre de ce cours.

Nous allons tout d'abord aborder le chiffrement symétrique, utilisé pour fournir la confidentialité, puis nous examinerons les fonctions de hachage et leur utilisation dans le cadre de l'authentification des messages. Nous analyserons ensuite le chiffrement asymétique, ou à clé publique, qui permet la gestion des clés et les signatures digitales.

Ce chapitre est très largement inspiré de Stallings, Computer Security, Principles and Practice [3], chapitre 2.

2.1 Le chiffrement symétrique

2.1.1 Principes

Le chiffrement symétrique est la technique universelle permettant d'assurer la confidentialité des échanges ou du stockage de données. Jusqu'à la fin des années 70 et l'apparition du chiffrement à clé publique, il s'agissait d'ailleurs de la seule méthode de chiffrement connue.

Définition

Le **chiffrement symétrique** est composé de cinq éléments, illustrés par la figure 2.1 :

- Le message, en clair,
- L'algorithme de chiffrement, qui applique des substitutions et des transformations diverses au message,
- Une clé secrète, qui est appliquée à l'algorithme de chiffrement et de qui dépend la combinaison de substitutions et de transformations appliquées au message,
- Le message chiffré, qui est le message inintelligible produit par l'algorithem. Ce résultat dépend à la fois du message original et de la clé de chiffrement. Pour un message donné, deux applications d'un même algorithmes avec deux clés différentes donneront deux messages chiffrés différents.
- L'algorithme de déchiffrement, qui est typiquement l'algorithme de chiffrement appliqué à l'envers. Il prend le message chiffré et la clé, et restitue le texte original en clair.

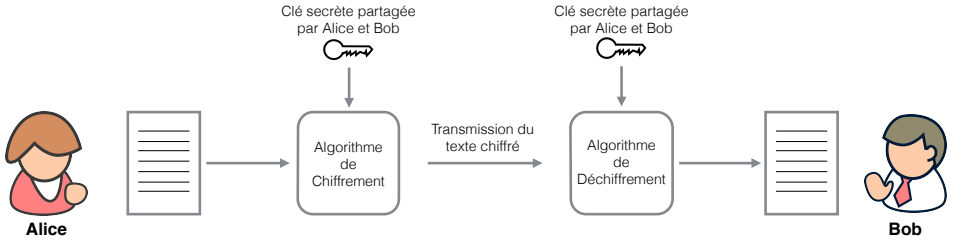

FIGURE 2.1 – Modèle simplifié du chiffrement symétrique

Il y a deux conditions essentielles pour que le chiffrement symétrique soit considéré comme sûr :

1. Un algorithme de chiffrement fort. Dans l'idéal, un attaquant connaissant l'algorithme ainsi que plusieurs combinaisons message en clair / message chiffré ne doit pas pouvoir reconstituer la clé et être capable de déchiffrer d'autres messages chiffrés.
2. L'émetteur et le receveur doivent avoir obtenu leur copie de la clé secrète de manière sécurisée, et doivent la garder en sécurité. Il est évident qu'un attaquant arrivant à mettre la main sur la clé secrète peut alors déchiffré toutes les données chiffrées.

Le chiffrement symétrique est vulnérable à deux types d'attaque. D'une part, les **attaques cryptanalytiques** se basent sur la nature de l'algorithme et sur l'observation ou la connaissance de certaines caractéristiques du message pour déduire soit la clé, soit le contenu du message chiffré. En cas de

succès, le résultat est catastrophique puisque tous les messages passés ou futurs chiffrés de la même manière sont également compromis.

D'autre part, le chiffrement symétrique est également vulnérable aux **attaques brute-force**. Le principe est d'essayer toutes les clés possibles sur un message chiffré jusqu'à ce qu'une traduction compréhensible soit trouvée. En moyenne, il faut essayer la moitié des clés possibles pour arriver à un résultat. Une des difficultés de l'attaque brute-force est qu'il faut pouvoir détecter qu'une tentative donne effectivement le résultat souhaité. Dans le cas d'un texte, c'est relativement facile, mais si le message initial a été compressé ou s'il s'agit de données numériques, la recherche est plus difficile à automatiser et nécessite une connaissance préalable du type de contenu attendu.

2.1.2 Le chiffrement symétrique par bloc

Il est possible de procéder de deux manières pour chiffrer un texte : Par bloc ou par flux. Le chiffrement par bloc consiste à découper le message en blocs de taille fixe, et à chiffrer ces blocs les uns après les autres. Le résultat de chaque étape est un bloc chiffré de taille fixe. Le message chiffré est alors la concaténation de tous ces blocs chiffrés. Les algorithmes DES et AES fonctionnent selon ce principe.

Data Encryption Standard

Adopté par le National Instute of Standards and Technology (NIST) en 1977, le standard DES propose un algorithme de chiffrement DEA (Data Encryption Algorithm) à appliquer à des blocs de 64 bits et à une clé de 56 bits. Les blocs chiffrés produits sont de 64 bits.

Les vulnérabilités potentielles de DES se situent à deux niveaux :

- D'une part, au niveau de l'algorithme lui-même, via les attaques cryptanalytiques. Malgré des années de tentatives et des études approfondies sur les propriétés de l'algorithme, aucune faiblesse n'a été découverte à ce jour.
- D'autre part, au niveau de la longueur de la clé. Avec une clé de 56 bits et la vitesse des processeurs actuels, DES est de nos jours vulnérable aux attaques brute-force.

DES dans sa version de base n'est donc plus fiable, mais il existe heureusement des alternatives. Une première possibilité est de chiffrer plusieurs fois un message en utilisant plusieurs clés différentes. C'est le principe de **Triple DES (3DES)**, qui applique donc trois fois le DEA standard avec deux ou trois clés différentes. La taille de clé résultante est donc de 112 ou 168 bits. 3-DES est un mécanisme de chiffrement adéquat actuellement, d'une part grâce à la résistance bien connue de l'algorithme DEA aux attaques cryptanalytiques, et d'autre part, grâce à sa clé de 168 bits qui lui

permet de résister aux attaques brute-force. Sa faiblesse réside par contre dans le temps de calcul, puisqu'il faut appliquer trois fois DES. DES a été initalement conçu pour une optimisation hardware, mais s'adapte mal au calcul en software. De plus, DES et 3DES utilisent tous les deux des blocs de 64 bits, ce qui, actuellement, n'est plus optimal en termes de sécurité et d'efficacité.

Advanced Encryption Standard

AES (Advanced Encryption Standard) a été conçu en 1997 pour surmonter les faiblesses de DES et 3DES. Les spécifications pour sa conception étaient qu'il devait avoir un niveau de sécurité égal ou supérieur à 3DES, tout en améliorant nettement son efficacité. En plus de cela, AES utilise des blocs de 128 bits de longueur, et supporte des longueurs de clés de 128, 192 et 256 bits.

2.1.3 Le chiffrement symétrique par flux

A l'inverse du chiffrement par bloc qui découpe le message en blocs de taille fixe, le chiffrement par flux travaille de manière continue, en chiffrant typiquement un octet (ou un bit, ou plusieurs octets) à la fois. La figure 2.2 montre le processus de chiffrement par flux. On y constate la présence d'un générateur de nombres pseudo-aléatoires. Ce générateur reçoit la clé en données, et produit un flux d'octets de 8 bits apparemment aléatoires, appelé en anglais *keystream*. L'idée est que ce flux ne peut pas être prédit sans avoir connaissance de la clé secrète. Ce flux est ensuite combiné, octet par octet et en utilisant l'opération logique XOR, au flux de données en clair pour produire le flux de données chiffré.

Le chiffrement par flux peut être aussi sûr que le chiffrement par bloc de taille comparable, à condition que le générateur de nombres pseudo-aléatoires soit correctement conçu. L'avantage du chiffrement par flux est qu'il est généralement plus rapide et nécessite moins de lignes de code que le chiffrement par blocs. Le choix de l'un ou l'autre se fera en fonction du contexte d'utilisation.

2.2 Les fonctions de hachage et l'authentification des messages

Le chiffrement permet de garantir la confidentialité des échanges, et protège contre les attaques passives telles que l'écoute des communication (eavesdropping). La protection contre les attaques actives telles que la falsification des données ou des transactions requiert d'autres mécanismes, permettant d'authentifier un message ou des données et donc de garantir leur intégrité.

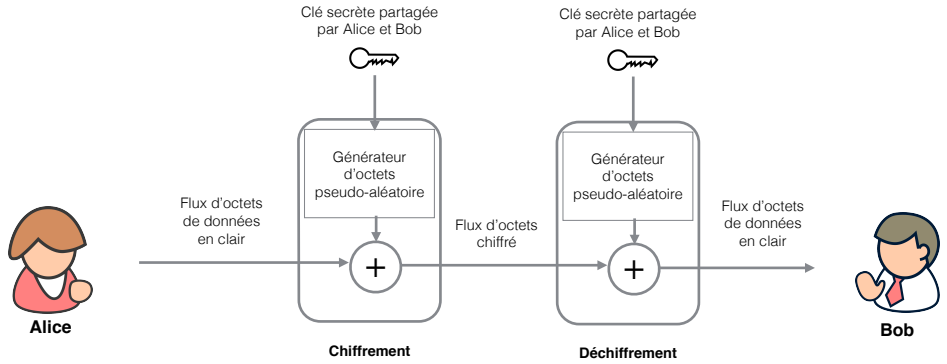

FIGURE 2.2 – Modèle simplifié du chiffrement symétrique par flux

L'authentification des données nécessite deux vérifications : D'une part, il faut s'assurer que le message vient bien de la source authentique, et d'autre part, il faut s'assurer que le contenu du message n'a pas été altéré. Dans certains cas, il peut également être souhaitable de valider la temporalité du message, c'est-à-dire vérifier qu'il n'a pas été artificiellement retardé et rejoué.

Une première idée pour assurer l'authentification de manière simple serait d'utiliser le chiffrement symétrique. En effet, puisque seul l'émetteur et le receveur partagent une même clé, à partir du moment où un message peut être décodé, on peut supposer que la source en était authentique. De plus, en cas de présence d'un mécanisme de détection d'erreur et d'un numéro de séquence, on pourrait s'assurer qu'aucune altération n'a été faite sur le message.

En pratique, le chiffrement symétrique seul n'est pas un outil adéquat pour l'authentification de données. Par exemple, dans le cas du chiffrement par bloc tel que décrit plus haut, il est possible pour un attaquant de mélanger les blocs chiffrés. Le receveur peut toujours déchiffrer le message, mais ce dernier sera probablement incompréhensible à cause du ré-ordonnancement.

2.2.1 L'authentification de message sans chiffrement

Nous allons à présent examiner différents moyens pour authentifier un message sans le chiffrer. L'idée est de générer un tag d'authentification et de l'ajouter à la fin du message lors de la transmission. Le message en lui-même n'est pas chiffré par ce mécanisme, et passe donc en clair à moins qu'un mécanisme de chiffrement complémentaire soit appliqué afin d'assurer la confidentialité en plus de l'authentification.

Dans certains cas, il peut être souhaitable de disposer de l'authentification sans assurer la confidentialité par le chiffrement. Par exemple, si le receveur est lourdement chargé, il ne peut pas nécessairement assurer le dé-

chiffrement de toutes les données reçues. Il pourrait alors se contenter de valider l'authenticité d'un échantillon des données, de manière aléatoire. Un autre exemple est l'authentification des programmes informatiques : Il est souvent préférable d'éviter de déchiffrer un programme à chacun de ses exécutions. Par contre, il est envisageable de vérifier l'authentification sur base du tag lorsqu'il faut s'assurer de l'intégrité du programme.

Code d'authentification de message

Le principe du code d'authentification de message (MAC pour Message Authentication Code) est d'utiliser une fonction qui génère un code sur base du message et d'une clé secrète, partagée entre la source et la destination. Le code est ajouté au message et transmis. La destination, qui possède également la clé secrète, recalcule le code et le compare avec celui reçu pour s'assurer de l'authenticité des données reçues. Ce processus est illustré à la figure 2.3

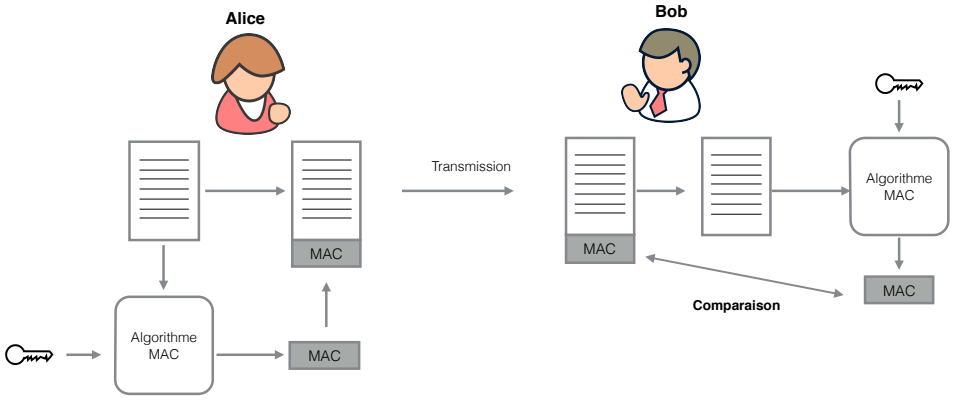

FIGURE 2.3 – Authentification de message par MAC

Avec ce mécanisme, tant que la clé reste secrète, les garanties suivantes sont fournies :

- Le receveur est sûr que le message n'a pas été altéré, puisqu'une altération se verrait dans le code. L'attaquant ne peut pas non plus modifier le code pour rendre le changement invisible puisqu'il ne dispose pas de la clé.
- Le receveur est sûr que le message vient de l'émetteur authentique, puisque lui seul peut calculer le code.
- Si le message inclut un numéro de séquence (ex : TCP), puisque ce numéro ne peut être altéré, le receveur est sûr que le message a été reçu conformément à la séquence.

DES peut être utilisé comme algorithme pour MAC, par exemple en chiffrant le message et en gardant les derniers bits du résultat (typiquement,

16 ou 24 bits). L'algorithme utilisé n'a pas besoin d'être symétrique puisque l'émetteur et le receveur effectuent la même opération de calcul du MAC.

Fonctions de hachage unidirectionnelles

Une alternative à MAC est l'utilisation de fonctions de hachage unidirectionnelles. Une fonction de hachage prend un message M en entrée et produit un « résumé »(digest) de taille fixe en sortie. A la différence de MAC, les fonctions de hachage n'utilisent pas de secret partagé. Il faut donc utiliser un mécanisme supplémentaire pour assurer l'authentification. On peut par exemple chiffrer le résumé du message par du chiffrement symétrique ou à clé publique. Cette dernière approche permet d'ailleurs de fournir une signature digitale. Il est également possible d'éviter le chiffrement, pour limiter les coûts de calcul. Cela peut se faire par exemple en utilisant un secret partagé, qui est concaténé au message lors de l'application de la fonction de hachage. Seul un receveur possédant le même secret peut reproduire et processus et valider l'authenticité du message.

Garanties de sécurité des fonctions de hachage

Le but d'une fonction de hachage est de fournir une empreinte digitale d'un bloc de données. Pour qu'une fonction de hachage H puisse être utilisée dans le cadre de l'authentification de données, elle doit posséder les propriétés suivantes :

- H peut être appliquées à un bloc de données de n'importe quelle taille
- H produit un résultat de longueur fixe
- H est relativement légère à calculer et peut être implémentée facilement aussi bien en hardware qu'en software
- Il ne doit pas être possible de retrouver le bloc de données initial sur base du résultat de l'application de H sur ce bloc
- Etant donné un message, il ne doit pas être possible de trouver un autre message produisant le même résultat avec la fonction de hachage (résistance faible aux collisions)
- Il ne doit pas être possible de trouver deux messages produisant le même résultat avec la fonction de hachage (résistance forte aux collisions).

Les trois premières propriétés sont d'ordre pratique. La quatrième permet l'unidirectionnalité : Il est facile de calculer le résultat d'une fonction de hachage, mais il doit être difficile de trouver le message initial sur base du « hash ». Les deux dernières donnent des garanties supplémentaires de résistance contre des attaques spécifiques, que nous ne détaillerons pas ici.

Remarquons qu'en plus de fournir l'authentification, le hachage permet également de garantir l'intégrité des données. Si n'importe quel bit du message est altéré pendant la communication, cela sera détecté lors de la vali-

dation.

Tout comme le chiffrement, les fonctions de hachage sont potentiellement vulnérables à la cryptanalyse et aux attaques brute-force. Pour ces dernières, un des critères déterminant va être la longueur du code de hachage produit. Par exemple, actuellement, l'algorithme de hachage MD5, produisant des codes de 128 bits, peut être « craqué »relativement facilement. Une autre fonction de hachage qui est plus fréquemment utilisé de nos jours est SHA (Secure Hash Algorithm). Trois versions de SHA ont été développées : SHA-1, avec 160 bits, n'est plus utilisable. SHA-2 propose des valeurs de hachages de longueur 256, 384 et 512. Les longueurs des valeurs de hachage sont toujours acceptables, mais des similarités structurelles entre les algorithmes SHA-1 et SHA-2 ont conduit à la standardisation d'un nouvel algorithme, SHA-3, disponible depuis 2012.

Nous avons discuté ici l'utilisation des fonctions de hachage dans le cadre de l'authentification de message, mais elles peuvent être utilisées pour d'autres applications de sécurité. Ainsi, elles permettent le stockage sécurisé des mots de passe dans le système d'exploitation, et peuvent également être utilisées pour la détection d'intrusion.

2.3 Le chiffrement à clé publique

Proposé pour la première fois par Diffie et Hellman en 1976, le chiffrement à clé publique est la première avancée révolutionnaire depuis des centaines d'années. L'innovation tient dans le fait que le chiffrement se fait sur base de fonctions mathématiques plutôt que sur des opérations sur des schémas de bits tels qu'utilisés en chiffrement symétrique. Plus important encore, la cryptographie à clé publique est **asymétrique**, nécessitant deux clés distinctes, ce qui a des impacts en termes de confidentialité, de distribution des clés et d'authentification.

Le chiffrement à clé publique, contrairement à une croyance répandue, n'est pas plus sécurisé que le chiffrement symétrique. La sécurité d'un chiffrement dépend essentiellement, d'une part, de la longueur de la clé, et d'autre part, de la complexité calculatoire nécessaire pour casser l'algorithme. Le chiffrement à clé publique n'est pas non plus destiné à remplacer le chiffrement symétrique. La distribution des clés, par exemple, est un problème délicat à gérer dans le chiffrement à clé publique. Les deux mécanismes sont en fait complémentaires, et le choix de l'un ou l'autre dépendra du contexte d'utilisatoin.

Le chiffrement à clé publique se base sur six éléments :

- Le message, en clair,
- L'algorithme de chiffrement, qui applique des substitutions et des transformations diverses au message,
- Une clé publique et une clé privée, qui sont choisies de telle sorte que

l'une sera utilisée pour le chiffrement et l'autre pour le déchiffrement. Comme son nom l'indique, la clé privée est gardée secrète par son propriétaire, tandis que la clé publique est diffusée publiquement de telle sorte que n'importe qui puisse l'utiliser.
- Le message chiffré, qui est le message inintelligible produit par l'algorithem. Ce résultat dépend à la fois du message original et de la clé de chiffrement. Pour un message donné, deux applications d'un même algorithmes avec deux clés différentes donneront deux messages chiffrés différents.
- L'algorithme de déchiffrement, qui est typiquement l'algorithme de chiffrement appliqué à l'envers. Il prend le message chiffré et la clé, et restitue le texte original en clair.

Les étapes principales du chiffrement sont les suivantes :

1. Les utilisateurs génèrent une paire clé privée/clé publique à utiliser pour le chiffrement et le déchiffrement de messages
2. Les utilisateurs placent une des deux clés dans un registre publique ou un fichier accessible. C'est la clé publique. L'autre clé est gardé privée.
3. Si Alice souhaite envoyer un message privé à Bob, elle chiffre le message avec la clé publique de Alice.
4. Quand Bob reçoit le message, il le déchiffre avec sa clé privée. Aucun autre destinataire ne peut déchiffrer le message, car seul Bob connaît la clé privée.

Cet exemple d'utilisation est illustré à la figure 2.4. Dans ce schéma, n'importe qui peut envoyer un message chiffré à Bob, et lui seul peut le déchiffrer. Ce système garantit la confidentialité de la communication.

La figure 2.5 montre un autre cas d'utilisation : Cette fois, Alice chiffre le message avec sa clé privée, et Bob le déchiffre avec la clé publique de Alice. En pratique, n'importe qui peut déchiffrer le message. Il n'est plus question ici d'assurer la confidentialité, mais bien l'authentification et/ou l'intégrité des données : N'importe qui peut s'assurer que c'est bien Alice qui a envoyé les données, et seule Alice peut modifier le message puisqu'elle est la seule à pouvoir effectuer le chiffrement.

En pratique, les systèmes à clés publiques peuvent être utilisés dans le cadre de différentes applications. Ces applications peuvent être regroupées en trois catégories :
- La signature numérique
- La distribution de clés symétriques
- Le chiffrement de clés secrètes

Les conditions à remplir par un algorithme de chiffrement asymétrique sont les suivantes :
- Il doit être possible de générer une paire (clé privée, clé publique) en un temps de calcul raisonnable

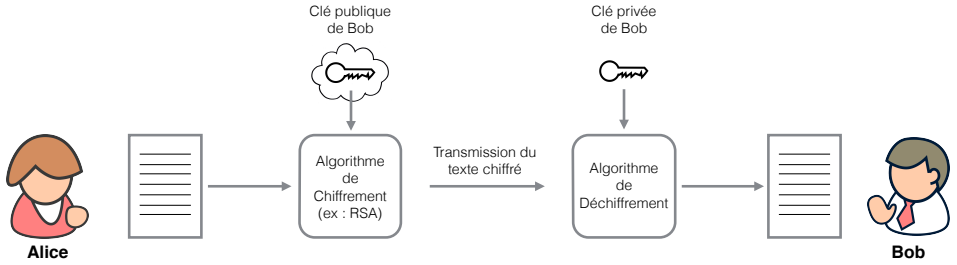

FIGURE 2.4 – Alice envoie un message chiffré à Bob avec la clé publique de ce dernier

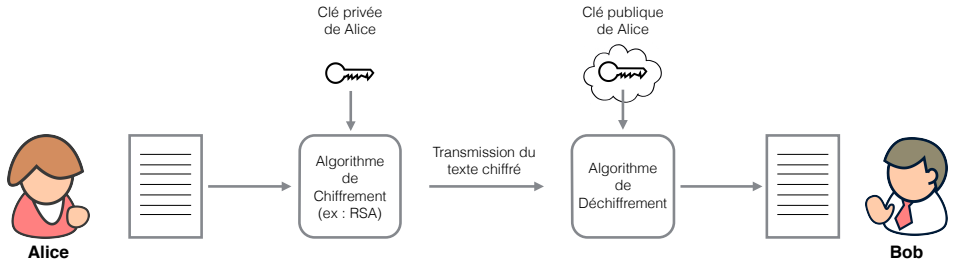

FIGURE 2.5 – Alice envoie un message chiffré à Bob avec la clé publique de ce dernier

- Il doit être possible de chiffrer un message sur base de la clé publique en un temps de calcul raisonnable
- Il doit être possible de déchiffrer un message chiffré avec la clé publique en un temps de calcul raisonnable en utilisant la clé privée
- Il doit être impossible en terme de temps de calcul pour un attaquant connaissant la clé publique de déterminer la clé privée
- Il doit être impossible en terme de temps de calcul pour un attaquant, connaissant la clé publique et un texte chiffré, de recomposer le message original
- Et enfin, pour certaines applications, il doit en plus être possible d'utiliser n'importe laquelle des deux clés pour le chiffrement, tandis que l'autre est utilisée pour le déchiffrement.

2.3.1 Algorithmes de chiffrement asymétriques

Nous allons présenter succintement quatre algorithmes permettant le chiffrement asymétrique. Chaque algorithme, en fonction de ses spécificités, peut convenir pour certaines des applications listées plus haut. Le tableau 2.1 montre les applications possibles avec chacun d'entre eux.

Un des premiers systèmes à clé publique est l'algorithme **RSA**, développé en 1977 au MIT. Il s'agit de l'algorithme le plus couramment utilisé et accepté

Algorithme	Signature numérique	Distribution de clé symétrique	Chiffrement de clé secrète
RSA	Oui	Oui	Oui
Diffie-Hellman	Non	Oui	Non
DSS	Oui	Non	Non
ECC	Oui	Oui	Oui

TABLE 2.1 – Applications des cryptosystèmes à clé publique

pour le chiffrement à clé publique. RSA effectue du chiffrement par bloc. Actuellement, la longueur de clé recommandée pour RSA est de 1024 bits.

Un autre algorithme connu est l'algorithme de **Diffie-Hellman**. Son but est de permettre l'échange d'un secret partagé entre deux utilisateurs, par exemple pour pouvoir effectuer du chiffrement symétrique par après.

L'algorithme **DSA**, pour Digital Signature Algorithm, fait partie du standard DSS (Digital Signature Standard), proposé par le NIST. DSS fournit un système de signature digitale sur base de SHA-1. Contrairement à RSA, il ne peut pas être utilisé pour le chiffrement ou l'échange de clé.

Enfin, le dernier mécanisme à clé publique que nous mentionnerons est la **cryptographie par courbe elliptique (ECC)**. L'objectif de ce mécanisme, relativement récent, est de fournir une alternative plus légère à RSA qui, avec l'augmentation de la longueur des clés, devient lourd à appliquer dans certains systèmes à lourde chare comme des sites de commerces électroniques. ECC semble fournir la même sécurité que RSA avec une charge de calcul amoindrie, mais étant donné que ECC n'est utilisé dans des produits que depuis peu de temps, le niveau de confiance qu'on lui accorde est moindre que celui de RSA.

2.4 Les signatures digitales

Les systèmes à clé publiques ont de nombreuses applications qui concernent d'une part la signature numérique, et d'autre part la gestion des clés. Nous allons ici détailler cette première applications, tandis que la section suivante s'attardera sur la problématique des clés.

Le chiffrement à clé publique peut donc être utilisé pour l'authentification, comme nous l'avons vu dans l'exemple de la figure 2.5, où Alice chiffre son message avec sa clé privée pour que Bob soit sûr que le message émane bien d'elle. Il est possible de simplifier et d'alléger le processus en se contentant de chiffrer non pas l'entièreté du message, mais une valeur de hachage calculée par exemple par SHA-512 (= SHA-2 avec une clé de 512 bits). Il n'est en effet pas important que le message circule en clair, puisque l'objectif à atteindre est l'authentification de la source.

Dans ce schéma simplifié, Alice va donc :

1. Calculer la valeur de hachage SHA-512 de son message,
2. Chiffrer cette valeur de hachage avec sa clé privée, créant une **signature numérique**
3. Envoyer le message original avec la signature numérique

A la réception du message, Bob va effectuer les opérations suivantes :

1. Calculer la valeur de hachage du message
2. Déchiffrer la signature d'Alice en utilisant sa clé publique
3. Comparer la valeur calculée avec la signature déchiffrée. Si les deux correspondent, Bob a la certitude que c'est Alice qui a bien envoyé le message.

Encore une fois, ce système ne fournit pas la confidentialité. Il garantit que le message arrive sans altération et a été généré par la bonne personne, mais rien n'indique qu'un attaquant n'a pas pu écouter le message en cours de transmission. Cela est valable aussi bien dans le cas d'une signature digitale partielle que dans le cas d'un chiffrement complet comme à la figure 2.5, puisque l'attaquant peut déchiffrer le message à l'aide de la clé publique d'Alice.

2.5 La gestion des clés

La cryptographie à clé publique est donc très utilisée dans le cadre de la gestion des clés. Nous allons aborder trois cas d'utilisation :

- La distribution sécurisée des clés publiques
- L'utilisation du chiffrement à clé publique pour la distribution de clés secrètes
- L'utilisation du chiffrement à clé publique pour créer des clés temporaires pour le chiffrement de messages

2.5.1 Les certificats de clé publique

Un des points faibles du chiffrement à clé publique est justement le fait qu'une des clés est publique. Chacun peut diffuser la clé publique qu'il a produit, mais rien ne garanti, en l'état actuel, que cette clé provient bien de la personne annoncée. Dans le cas de l'exemple de la figure 2.4, si un attaquant désire se faire passer pour Bob, il lui suffit de transmettre sa clé publique en prétendant qu'il est Bob, et recevoir et déchiffrer toutes les communications à destination de Bob.

La solution à ce problème est le certificat de clé publique. Il consiste à associer un identifiant utilisateur à la clé publique, et à faire chiffrer l'ensemble par un tiers de confiance. Le certificat contient en plus des informations sur le tiers, et sur la durée de validité du certificat. Ce tiers est typiquement

une autorité certificative (CA), telle qu'une agence gouvernementale ou une institution financière.

Le processus de vérification comporte plusieures étapes, montrées à la figure 2.6 :

1. L'utilisateur crée une paire de clés, une publique et une privée
2. L'utilisateur prépare un certificat non signé incluant son identifiant et sa clé publique
3. L'utilisateur fournit le certificat non signé à une CA de manière sécurisée (en main propre, via un formulaire avec vérification par email, ...)
4. L'autorité certificative crée la signature numérique comme suit :
   - L'autorité certificative utilise une fonction de hachage pour calculer le hash du certificat non signé
   - L'autorité certificative chiffre le hash avec sa clé privée
5. L'autorité certificative attaque la signature numérique au certificat non signé pour obtenir le certificat signé
6. L'autorité certificative renvoie le certificat signé à l'utilisateur
7. L'utilisateur peut à présent fournir le certificat à n'importe quel autre utilisateur.

Pour valider le propriétaire d'une clé publique, il faut donc partir du certificat, calculer le hash de la partie du certificat sans la signature, déchiffrer la signature numérique de l'autorité certificative, et comparer le résultat obtenu avec le hash.

Un des standards les plus répandus pour le formatage des certificats de clés publiques est le standard X.509.

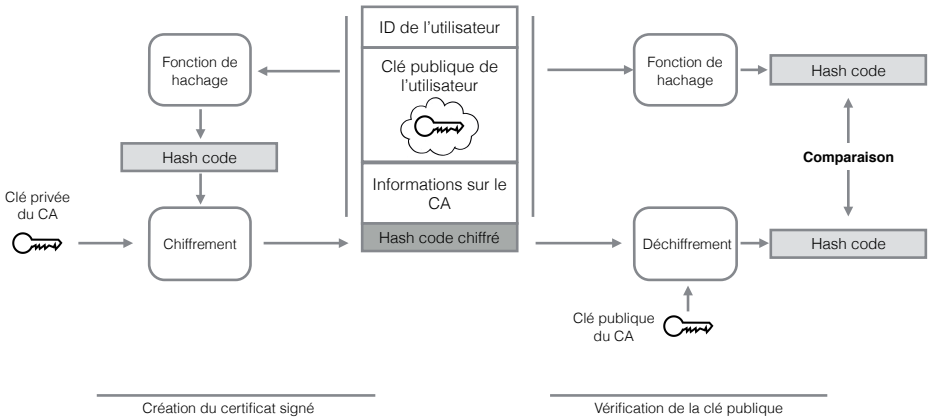

FIGURE 2.6 – Génération et validation d'un certificat de clé publique

2.5.2 L'échange de clés symétriques

Une des difficultés pratiques du chiffrement symétrique consiste à échanger une clé secrète partagée. Prenons l'exemple d'Alice qui souhaite créer une application de messagerie permettant d'échanger des emails de façon sécurisée avec n'importe qui ayant une connexion à Internet, en utilisant le chiffrement symétrique. Comment faire en sorte que deux interlocuteurs puissent se mettre d'accord sur le secret partagé, à savoir la clé ? Si les interlocuteurs sont dans la même pièce ou peuvent facilement s'échanger par exemple une clé USB, l'échange est possible, mais si Alice et Bob se trouvent sur deux continents différents, c'est évidemment plus compliqué.

Une solution à ce problème est l'utilisation de cryptographie à clé publique, comme par exemple l'algorithme d'échange de clés de Diffie-Hellman, que nous ne détaillerons pas ici.

2.5.3 Les enveloppes numériques

Une autre application utilisant le chiffrement à clé publique pour la protection d'une clé symétrique est le système d'enveloppe numérique, utilisée pour protéger un message sans concertation préalable de l'émetteur et du receveur sur une clé secrète partagée.

La clé secrète partagée sera en fait générée de manière aléatoire par l'émetteur, utilisée pour le chiffrement du message, et enfin, elle sera elle-même chiffrée via la clé publique du receveur avant d'être transmise en même temps que le message. L'enveloppe numérique est donc composée d'une part, du message chiffré par la clé secrète, et d'autre part, de la clé secrète elle-même, chiffrée avec la clé publique du receveur.

A la réception, le receveur déchiffre la clé secrète à l'aide de sa clé privée, et peut de la sorte déchiffrer le message.

Chapitre 3

Authentification de l'utilisateur

Dans le contexte de la sécurité informatique, l'authentification constitue souvent une brique essentielle du système, et la première ligne de défense contre des attaques. Elle est également à la base de la plupart des systèmes de contrôle d'accès et de la traçabilité des utilisteurs. L'authentification utilisateur est définie de la manière suivante dans [4] :

Définition

L'authentification est le processus de vérification de l'identité déclarée par ou pour une entité d'un système. Ce processus comporte deux étapes :

1. L'identification : Cette étape consiste à présenter l'identifiant que l'on revendique au système de sécurité
2. La vérification : Cette étape consiste à présenter ou générer de l'information d'authentification qui corrobore le lien entre l'entité et l'identifiant.

L'identification est donc le moyen par lequel un utilisateur fournit l'identité au système, tandis que l'authentification consiste à établir la validité de cette revendication.

Attention à ne pas confondre l'authentification de l'utilisateur et l'authentification d'un message. Cette dernière consiste bien à permettre à deux entités communicantes de vérifier que le contenu d'un message reçu n'a pas été altéré et que la source est authentique.

3.1 Principes

L'authentification des utilisateurs repose donc sur deux systèmes : Un premier système qui permet à un utilisateur de s'enregister, c'est-à-dire de se créer un identifiant. Ce processus d'enregistrement comprend une étape

de vérification de l'identité du nouvel utilisateur, la création d'une structure de données liant l'identité à un token, et la remise de ce token à l'utilisateur. Ce token peut être une clé de chiffrement, ou un mot de passe chiffré permettant à l'utilisateur de s'authentifier. Le second système permettra à un utilisateur enregistré de s'authentifier, sur base du token reçu lors de la phase d'enregistrement.

Ce token peut être de quatre types :

- Quelque chose connu par l'utilisateur (mot de passe, PIN, réponse à une question secrète, ...)
- Quelque chose possédé par l'utilisateur (smart card, clé physique, ...), souvent appelé **token**
- Quelque chose caractérisant l'utilisateur (reconnaissance d'empreinte digitale ou rétinienne, ...)
- Quelque chose que l'utilisateur fait (reconnaissance d'empreinte vocale, graphologie, rythme de frappe au clavier, ...)

Chaque méthode a bien entendu ses points forts et ses faiblesses. Un mot de passe peut être deviné ou volé, ou encore oublié par l'utilisateur. Un token peut être volé ou perdu. La biométrie pose également problème en termes de précision (faux positifs/négatifs), de coûts, ou encore d'acceptation par l'utilisateur.

3.2 Authentification par mot de passe

La technique de défense la plus utilisée contre les intrusions est le système de mot de passe, utilisés par la grande majorité des services en ligne. Le principe est que l'utilisateur fournit son identifiant et son mot de passe au système, et ce dernier compare le mot de passe avec celui associé à l'utilisateur pour fournir l'accès. Ce système d'identification permet d'assurer la vérification d'une part du droit d'accès de l'utilisateur, et d'autre part, des privilèges de ce dernier (anonyme, normal, super-utilisateur, ...). Le système fonctionne typiquement en maintenant une liste des mots de passe indexés par identifiant utilisateur. Souvent, le mot de passe n'est pas stocké en clair, mais est plutôt remplacé par sa valeur de hachage en utilisant une fonction de hachage unidirectionnelle.

Les mots de passe étant largement utilisés, ils sont la cible de nombreuses attaques, que nous décrivons ci-après ainsi que les contre-mesures associées :

- Attaque par dictionnaire hors ligne : En obtenant accès au fichier de mots de passe, il est possible de comparer les valeurs de hachages qu'il contient avec celles de mots de passe fréquemment utilisés. Si une correspondance est trouvée, le compte utilisateur correspondant peut être exploité. Les contre-mesures consistent à bloquer l'accès au fichier de mots de passe, à détecter les intrusions pour repérer les tentatives d'attaque, et à un changement rapide des mots de passe si le fichier est

compromis. Il faut bien sûr rajouter à ces mesures la sensibilisation des utilisateurs au choix d'un mot de passe robuste aux attaques par dictionnaire.

- Attaque sur un compte spécifique : Cette attaque cible un compte en particulier, et tente de s'y connecter en essayant des combinaisons jusqu'à ce que le mot de passe correct soit trouvé. Les contre-mesures consiste à bloquer les tentatives au delà de (par exemple) cinq échecs de connexion.
- Attaque par mot de passe populaire : Il s'agit ici d'essayer les mots de passe les plus utilisés sur un ensemble de comptes. Cette attaque est contrée par la sensibilisation des utilisateurs et par le scan des adresses IP des requêtes d'authentification afin de repérer des schémas d'attaque.
- Piratage de poste de travail : Cette attaque consiste à attendre qu'un poste se libère sans que l'utilisateur se soit déloggé. La contre-mesure consiste à délogger automatiquement l'utilisateur après une période d'inaction. Les systèmes de détection d'intrusion peuvent également détecter des changements de comportement de l'utilisateur du poste.
- Le social engineering : L'exploitation d'erreurs humaines peut permettre à un attaquant d'acquérir un mot de passe. Par exemple, un utilisateur qui a écrit son mot de passe, ou un attaquant qui obtient un mot de passe en se faisant passer pour un administrateur système, ou encore un mot de passe pré-configuré qui n'a pas été changé. Là encore, la solution réside dans la sensibilisation de l'utilisateur, dans la détection d'intrusion et dans la combinaison des systèmes à mot de passe avec d'autres mécanismes d'authentification.
- L'exploitation de l'usage multiple d'un mot de passe : Les utilisateurs ont souvent tendance à réutiliser leur mot de passe pour différents services. Des appareils réseaux sont également souvent configurés de la même manière, avec le même mot de passe. Les dommages en cas de compromission du mot de passe sont donc démultipliés.
- La surveillance électronique : L'écoute permet à un attaquant d'obtenir un mot de passe. Le fait que ce dernier soit chiffré n'est pas bloquant, car le mot de passe chiffré lui-même peut être exploité par une attaque de rejeu.

3.2.1 L'utilisation de mots de passe hachés

Souvent, les mots de passe hachés sont calculés en utilisant un « sel ». C'est typiquement le cas des systèmes Unix. Lors du calcul de la valeur de hachage, le mot de passe est associé avec une valeur de longueur fixe, appelée « sel »pour produire une valeur de hachage de longueur fixe. Pour limiter les attaques, l'implémentation de la fonction de hachage doit être relativement lente. Le résultat calculé est alors stocké dans le fichier de mot de passe, avec

la valeur du sel et l'identifiant utilisateur. Ce processus est montré à la figure 3.1.

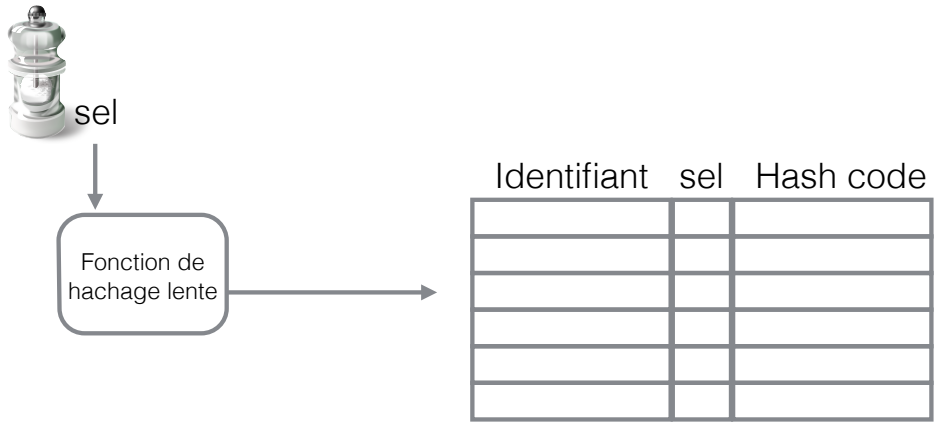

FIGURE 3.1 – Hachage d'un mot de passe Unix

Pour vérifier un mot de passe, le système utilise l'identifiant utilisateur pour récupérer le « sel », recalcule la valeur de hachage sur base du mot de passe entré et de ce sel, et compare le résultat avec le mot de passe haché stocké dans le fichier, comme montré à la figure 3.2

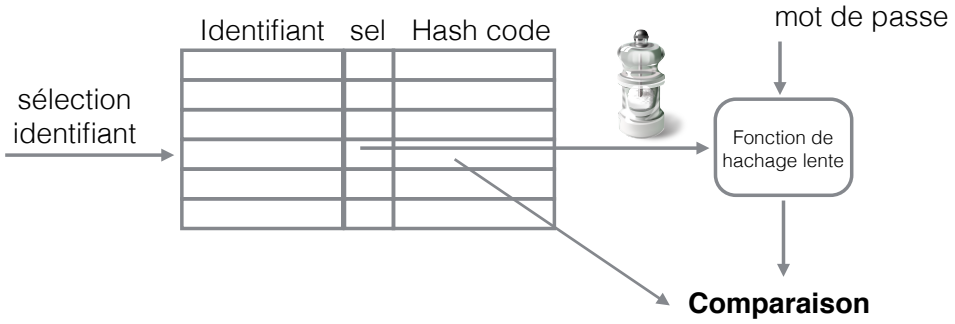

FIGURE 3.2 – Vérification d'un mot de passe Unix

L'utilisation du « sel »a trois objectifs :

- Elle empêche les mots de passes dupliqués d'être visible dans le fichier, puisqu'ils auront été hachés avec deux sels différents.
- Elle complexifie les attaques par dictionnaire, puisque, pour chaque supposition, l'attaquant doit appliquer la fonction de hachage autant de fois qu'il y a de valeur de sel.
- Elle rend presque impossible de savoir si une personne ayant des comptes sur plusieurs machines utilise le même mot de passe sur chacune d'elle.

Le système de mot de passe Unix est néanmoins exposé à une menace : Un utilisateur ayant accès à une machine peut récupérer le fichier de mots de passe et utiliser un logiciel « craqueur de mots de passe »pour retrouver les mots de passe.

A l'origine, Unix utilisait des mots de passe jusqu'à huit caractères en ASCII 7-bits, convertis en valeurs de 56 bits, et une valeur de sel de 12 bits. La routine de hachage `crypt` est basée sur DES, et consiste grosso modo à l'appliquer 25 fois, afin d'augmenter le temps de calcul. Ce système est depuis longtemps vulnérable aux attaques par dictionnaire, et la fonction de hachage actuellement utilisée sur les systèmes Linux, Solaris et FreeBSD est l'algorithme de hachage MD5 (similaire à SHA-1 mais moins sûr). Cet algorithme n'a pas de limitation sur la longueur du mot de passe, et utilise un sel jusqu'à 48 bits. Pour ralentir l'algorithme, le chiffrement MD5 utilise une boucle à mille itérations.

3.2.2 Craquage de mots de passe

Approches traditionnelles

L'approche traditionnelle pour craquer des mots de passe consiste donc à développer un dictionnaire de mots de passe possible, et de les comparer un par un avec le fichier de mots de passe. Chaque essai doit donc être combiné à toutes les valeurs de sel possibles pour le calcul des valeurs de hachage correspondantes. Des variations sur les mots du dictionnaire sont également possibles (ex : mots à l'envers, ajout de nombres ou de caractères spéciaux, etc.).

Pour optimiser cette attaque, il est possible de pré-calculer les valeurs de hachage potentielles pour toutes les combinaisons du dictionnaire. Cela génère un énorme fichier (appelé *rainbow table*), mais accélère sensiblement la recherche de mots de passe. Il a été montré qu'avec un fichier de 1.4 GB, il était possible de craquer 99,9% des mots de passes alphanumériques Windows en moins de 14 secondes [7]. Cette technique s'appelle le *compromis temps-mémoire*.

Approches modernes

Les attaques sur les mots de passe ont hélas continué à évoluer ces dernières décennies. Les pratiques de gestion de mots de passe s'améliorent, mais les techniques de craquage aussi, et ce, sur deux plans. Tout d'abord, l'évolution dans les puissances de calcul des ordinateurs permet aujourd'hui à de simples postes de travail d'effectuer des calculs nécessitant des supercalculateurs il y a à peine dix ans. Ensuite, les algorithmes de génération de mots de passe potentiels ont également évolué, et se basent notamment sur l'étude de larges bases de données de mots de passe permettant de caractériser les mots de passes en utilisation. De telles bases de données sont actuel-

lement disponibles. Ainsi, en 2009, une attaque par injection SQL contre un service de jeux en ligne a permis de révéler des millions de mots de passe en clair. D'autres datasets ont également été publiés depuis.

3.2.3 Contrôle d'accès au fichier de mots de passe

Une des façons d'empêcher les attaques sur les mots de passe est de limiter l'accès au fichier les contenant, et plus particulièrement à la partie contenant les valeurs de hachage. Souvent, les mots de passe hachés sont stockés dans un fichier séparé des identifiants utilisateurs, fichier appelé **shadow password file**. Les accès à ce fichier doivent bien entendu être soigneusement contrôlés.

Malgré le contrôle d'accès au fichier, il reste d'autres vulnérabilités mettant en péril ce fichier, comme l'exploitation de faille, la compromission d'un mot de passe par un autre moyen, ou encore le sniffing d'informations sur le réseau.

Le contrôle d'accès au fichier de mot de passe doit donc être associé avec d'autres mesures de protection.

3.2.4 Stratégies de sélection d'un mot de passe

Il est généralement admis qu'il faut imposer des contraintes sur le choix des mots de passe, sans quoi les utilisateurs ont tendance à les choisir trop courts ou trop faciles à deviner. A l'inverse, si l'on impose des mots de passe aléatoires, effectivement impossible à craquer, les utilisateurs risqueraient de les oublier ou, pire encore, de les noter.

Quatre techniques sont utilisées pour garantir l'utilisation de mots de passe d'un niveau de sécurité acceptable :

- La sensibilisation des utilisateurs : Les utilisateurs ne savent pas instinctivement quels mots de passe sont sécurisés ou non. Ils peuvent ainsi imaginer qu'un mot de passe inversé, ou avec des variations majuscules/minuscules sont sécurisés. Il est donc important de leur fournir des guidelines pour la selection de mots de passe. Une bonne technique consiste par exemple à utiliser la première lettre de chaque mot dans une phrase (ex : « Ma soeur Ernestine possede 24 chats Angora »donne MsEp24CA).
- La génération automatique de mots de passe : Comme mentionné plus haut, si les mots de passe aléatoires sont robustes, ils souffrent de ne pouvoir être facilement mémorisés par les utilisateurs. Il est néanmoins possible de générer des mots de passe formant des syllables prononcables, par exemple.
- La vérification des mots de passe : Cette stratégie consiste à périodiquement utiliser un craqueur pour vérifier si les mots de passe peuvent être devinés, et d'en informer l'utilisateur le cas échéant. Cette technique est néanmoins couteuse en temps.

– La vérification proactive des mots de passe : L'idée ici est de valider le mot de passe lors du processus d'enregistrement, et de le rejeter s'il n'offre pas suffisamment de garantie de sécurité.

3.3 Authentification par token

Les tokens sont des objets utilisés pour authentifier un utilisateur. Nous allons en examiner deux types : Les cartes à mémoire et les Smart Cards.

3.3.1 Les cartes à mémoire

Les cartes à mémoire peuvent stocker des informations, mais pas traiter de données. La plus courante est la carte banquaire avec bande magnétique stockant uniquement un code de sécurité (pouvant être facilement lue, mais hélas aussi reprogrammée). Il en existe d'autres avec une mémoire électronique interne.

Les cartes à mémoire peuvent être utilisées seule pour des contrôles d'accès physiques, comme par exemple les chambres d'hotel. Pour l'authentification, l'utilisateur doit en plus entrer un code d'identification personnel (PIN), par exemple pour un distributeur de billets de banque. Une carte mémoire associée à un PIN est plus sécurisée qu'un mot de passe seul, puisque l'attaquant doit alors avoir accès à la carte, et arriver à obtenir le code PIN.

Les inconvénients de ces cartes sont les suivants :

– Elles nécessitent un lecteur spécial, ce qui accroît les coûts d'utilisation et nécessite de garantir la sécurité à la fois du hardware et du software du lecteur.
– La perte d'un token empêche l'utilisateur d'accéder au système, et il y a un coût administratif de remplacement. La perte induit en plus le risque de voir le token utilisé par un attaquant, qui n'a alors plus qu'à deviner le PIN.
– Une mauvaise expérience-utilisateur : Bien que largement acceptées pour les distributeurs de billets, ces cartes le sont par contre moins pour l'accès à un ordinateur.

3.3.2 Les Smart Cards

Il existe de nombreux types de Smart Cards, qui peuvent être catégorisés selon quatre dimensions :

1. Les caractéristiques physiques : Ils peuvent être du format d'une carte banquaire, ou d'une clé, d'une calculatrice, ou tout autre objet portable
2. L'interface utilisateur : Certaines peuvent inclure un écran d'affichage et un clavier pour permettre l'interaction avec l'utilisateur

3. L'interface électronique : Les Smart Cards nécessitent une interface pour communiquer avec un lecteur compatible. Cela peut se faire par contact, ou sans contact (antenne)
4. Le protocole d'authentification : Il peut être soit statique, c'est-à-dire que l'utilisateur s'authentifie sur le token qui à son tour authentifie l'utilisateur sur le lecteur, à génération dynamique de mot de passe, c'est-à-dire que le token génère un nouveau mot de passe chaque seconde, ou encore à challenge/réponse. Dans ce dernier cas, l'ordinateur génère un challenge (suite de nombres), et le token génère une réponse sur base de ce challenge. Ce système peut par exemple utiliser la cryptographie à clé publique.

3.4 Authentification biométrique

Les systèmes d'authentification biométrique se base sur les caractéristiques physiques uniques des individus. Ces techniques, basées sur la reconnaissance de patterns, sont plus complexes et plus chères que les mots de passe et les tokens, et nécessitent encore d'être améliorées avant de pouvoir réellement être utilisées comme standards.

3.4.1 Caractéristiques physiques utilisées en biométrie

Les caractéristiques les plus couramment utilisées sont les suivantes :
- La reconnaissance faciale : La localisation des yeux, des sourcils, du nez, des lèvres, etc., ou l'utilisation d'une caméra infra-rouge pour détecter le système vasculaire sous-jacent du visage.
- Les empreintes digitales
- La géométrie de la main : Sa forme, la longueur et l'épaisseur des doigts, ...
- L'empreinte rétinienne : Le dessin formé par le réseau de veines de la rétine est unique et peut donc servir à l'identification. L'image de cette empreinte est récupérée en projetant un laser à base intensité ou une lumière infra-rouge dans l'oeil.
- L'iris : La structure de l'iris est également unique pour chaque individu
- La signature : Chaque individu a un style d'écriture unique, qui se réflète particulièrement dans la signature. Ce système est néanmoins peu robuste car une même personne ne produira pas deux fois la même signature.
- La voix : Ce système est plus stable que celui basé sur la signature, mais souffre néanmoins de variabilité entre deux échantillons d'un même utilisateur.

Contrairement aux systèmes par mot de passe ou par token, l'authentification biométrique n'est pas entièrement précise, et l'authentification d'un

utilisateur valide peut échouer occasionnellement. Parmi les systèmes exposés, la reconnaissance de l'iris est le plus fiable, mais coûte néanmoins le plus cher. La signature, la voix, la main et le visage ne sont par contre pas suffisamment précis pour être utilisés dans le cadre de l'authentification. Les empreintes digitales et rétiniennes offrent une meilleure précision pour un coût moyen.

3.5 Authentification à distance

La forme la plus simple d'authentification est l'authentification locale, avec communication directe entre l'utilisateur et le système. Il existe une forme plus complexe, qui est l'authentification à distance, avec l'utilisateur se trouvant sur une machine distante de celle sur laquelle il veut s'authentifier. Le processus d'authentification doit donc s'effectuer au travers d'un lien de communication, un réseau, voire Internet. Cela soulève donc de nouvelles problématiques sécuritaires, puisque les communications sont vulnérables à l'écoute, permettant à un attaquant de capturer un mot de passe, ou au rejeu, permettant à l'adversaire de rejouer une séquence d'authentification.

Ces menaces sont généralement contrées par des protocoles d'authentification sécurisées utilisant le principe du challenge/réponse.

3.6 Attaques ciblant l'authentification

Les systèmes d'authentification sont vulnérables aux attaques suivantes :

- Les attaques Client : Un adversaire essaye de réussir une authentification sans avoir accès à l'hôte ni intervenir dans les communications, en se faisant passer pour l'utilisateur légitime. Il s'agit par exemple de tentatives de deviner le mot de passe. Les contre-mesures consistent à choisir des mots de passe robustes, ou encore à limiter le nombre de tentatives.
- Les attaques Hôte : La machine cible est directement attaquée pour gagner accès au fichier stoquant les informations sur les mots de passe, les tokens ou les données biométriques.
- L'écoute : Dans le contexte des mots de passe, cela consiste à observer l'utilisateur, à retrouver une copie écrite du mot de passe. Le keylogging est une autre technique permettant d'arriver au même résultat. Avec les tokens, la menace est en l'occurence le vol ou la copie physique du token, et pour la biométrie, il s'agit de copier ou d'imiter les caractéristiques biométriques utilisées pour l'authentification.
- Le rejeu : Cette attaque consiste à ré-utiliser un échange d'authentification. La contre-mesure correspondante est l'utilisation d'un protocole à challenge/réponse.

- L'attaque par cheval de Troie : Cette attaque consiste à se faire passer pour une application d'authentification authentique afin de captuer le mot de passe utilisateur, ou ses caractéristiques biométriques.
- Le déni de service : L'attaquant tente de désactiver le système d'authentification en le surchargeant de demandes. Cette attaque peut cibler plus spécifiquement un utilisateur, en dépassant le nombre de tentatives permises pour l'authentification. L'utilisateur est alors bloqué.

Chapitre 4

Contrôle d'accès

4.1 Généralités

La plupart des concepts en sécurité informatique sont liés, d'une manière ou d'une autre, au contrôle d'accès. En effet, après avoir défini les avoirs à protéger, il faut en garantir la protection, ce qui passe le plus souvent par le contrôle de l'accès physique ou informatique à ces avoirs. Le RFC4949 [4] définit d'ailleurs la sécurité comme : « les mesures qui implémentent et assurent les services de sécurité dans un système informatique, en particulier celles qui assurent le service de contrôle d'accès. ». Plus spécifiquement, nous allons aborder le concept de contrôle d'accès en considérant la manière d'implémenter une politique de sécurité qui spécifie qui et quoi peut avoir accès à chaque avoir, chaque ressource du système, ainsi que le type d'accès qui est permis pour chacun.

4.1.1 AAA

Le contrôle d'accès à un système informatique est souvent modélisé selon le protocole AAA [8] : Authentication, Authorisation, Accounting.

- Authentication : Il s'agit de la vérification de la validité des éléments prouvant l'identité d'une personne ou d'une entité système. Comme mentionné au chapitre précédent, l'authentificaiton requiert deux étapes : L'identification, c'est-à-dire l'existence d'un identifiant pour le demandeur, et l'authentification elle-même, c'est-à-dire la vérification que cette identité correspond bien au demandeur effectif. La gestion des identifiants requiert un soin particulier, spécialement lorsqu'il s'agit d'un identifiant externe. L'acceptation d'un identifiant pour une personne externe à l'entreprise nécessite des mesures particulières, comme la rédaction d'un contrat stipulant les conditions d'accès, une évaluation du risque, et une réflexion sur les droits d'accès accordés. Les accès externes peuvent par exemple concerner des consultants, des sous-traitants, des stagiaires, du personnel de nettoyage, etc.

- Authorization : Il s'agit de vérifier si la personne ou l'entité identifiée bénéficie des autorisations requises pour accéder à la ressource spécifiée. Lors de la mise en place des autorisations, il y a lieu de définir et documenter le besoin en termes d'accès du demandeur, et d'appliquer une politique de blocage en cas d'absence de besoin, ou encore en cas d'inactivité prolongée.
- Accounting : Afin de prévenir et de pouvoir réagir aux usurpations d'identité, il faut prévoir une surveillance des accès aux ressources, par un processus d'audit ou par l'utilisation de logs (enregistrement de l'heure des connexions, des actions effectuées, etc.). Dans le cadre d'une gestion efficace des logs impliquant des timestamps dans l'enregistrement des événements, il est important de maintenir une synchronisation des horloges du systèmes.

4.2 Modèles de contrôle d'accès

Les politiques de contrôle d'accès peuvent être catégorisées en plusieurs types de politiques, qui ne sont pas mutuellement exclusives :

- Discretionary Access Control (DAC) : Cette politique indique que le contrôle d'accès est basé sur des règles fixant quels demandeurs sont ou non autorisés à effectuer une action donnée. Le terme « discrétionnaire »vient de ce qu'une entité peut décider de transmettre ses droits d'accès à une ressource dont elle est propréitaire à une autre entité.
- Mandatory Access Control (MAC) : Le contrôle d'accès est basé sur le niveau de sécurité de la ressource (secret, top secret, confidentiel, ...), qui est comparé aux droits attribué au demandeur d'accès. Le terme « mandatory »vient de ce qu'une entité ayant accès à une ressource ne peut transmettre ce droit à une autre entité.
- Role-Based Access Control (RBAC) : Les droits d'accès sont définis pour chaque catégorie d'utilisateur, correspondant à un rôle spécifique. Cette politique est relativement simple à mettre en place, puisqu'il suffit d'identifier les différents rôles, de définir les droits d'accès pour chacun d'eux, puis d'attribuer un (ou plusieurs) rôle(s) à chaque utilisateur

4.3 Sujets, objets et droits d'accès

Les éléments intervenants dans un système de contrôle d'accès sont au nombre de trois :

Définition

Un **sujet** est une entité pouvant accéder à des objets.

On identifie trois catégories de sujets : Le **propriétaire** de la ressource, des **groupes** d'utilisateurs ayant des droits d'accès spécifiques, et enfin le **reste du monde**, à savoir les utilisateurs ne faisant pas partie des deux catégories sus-nommées.

Définition

Un **objet** est une ressource dont l'accès est contrôlée.

Il peut s'agir de blocs de données, de pages, de fichiers, de morceaux de fichiers, de répertoires, d'une arborescence de répertoires, de boîtes mails, de programmes, ...

Définition

Un **droit d'accès** décrit la manière dont un sujet peut accéder à un objet

Les droits d'accès sont, par exemple :
- Lecture
- Ecriture
- Exécution
- Suppression
- Création
- Recherche

4.4 Exemple : Le contrôle d'accès aux fichiers UNIX

Parmis les systèmes de contrôle d'accès implémentant le modèle DAC, on retrouve le contrôle d'accès aux fichiers des systèmes UNIX.

4.4.1 Gestion des fichiers UNIX

Pour rappel, les fichiers en UNIX sont gérés par l'intermédiaire d'une structure spécifique appelée inode (index node). Chaque fichier est contrôlé par un et un seul inode, et un inode est associé à un et un seul fichier. Entre autres informations, l'inode contient les attributs du fichier, ainsi que les permissions qui y sont associés. Les inodes sont stockés sur le disque, dans une table ou liste d'inodes. Chaque fois qu'un fichier est ouvert, son inode est chargé dans la mémoire principale.

Les répertoires sont structurés de manière hiérarchique, chaque répertoire contenant des fichiers ou d'autres répertoires. Un répertoire est simplement une liste de noms associés à des pointeurs vers les inodes correspondants (et donc, chaque répertoire possède lui-même un inode).

4.4.2 Gestion des utilisateurs UNIX

Dans la plupart des systèmes UNIX, un utilisateur est associé à un identifiant numérique (user ID), à un groupe primaire, et éventuellement à d'autres groupes, chaque groupe étant identifié par un group ID.

Il existe un user ID particulier, appelé « superuser ». Cet utilisateur possède un accès non restreint à toutes les ressources du système, et il faut bien entendu être particulièrement précautionneux lorsqu'un programme est exécuté avec les autorisations de cet utilisateur.

4.4.3 Permissions d'accès aux fichiers

Lorsqu'un fichier est créé, un propriétaire lui est assigné, et le fichier est marqué avec le user ID de cet utilisateur. On lui associe également un group ID, qui est soit le groupe primaire du créateur, soit le groupe du répertoire parent, selon la valeur de la permission SetGID de ce répertoire.

En plus de ce userID et de ce groupID, douze bits de protection sont associés au fichier : neuf spécifient les permissions, et trois définissent des comportements spécifiques supplémentaires :

- les trois premiers bits indiquent les permissions en lecture, écriture et exécution pour le propriétaire. Dans le cas d'un répertoire, le premier bit autorise le listing, le second la création, le renommage et la suppression de fichiers, et le troisième donne le droit d'effectuer une recherche sur un nom de fichier ou de descendre dans ce répertoire.
- les trois suivants font de même pour les utilisateurs du groupe associé au fichier/répertoire
- les trois suivants indiquent les droits d'accès pour les autres utilisateurs, non compris dans les deux catégories plus haut.
- le dixième bit est le bit SetUID. Lorsque ce bit est mis, lorsqu'un utilisateur possédant les droits d'exécution sur le fichier l'exécute, le système lui alloue temporairement les droits de l'utilisateur créateur du fichier (effective user ID, par opposition au real user iD). Dans le cas d'un répertoire, ce bit est ignoré.
- le onzième bit est le bit SetGID, qui fonctionne de manière similaire au précédent, mais appliqué au groupe. Lorsqu'il est appliqué à un répertoire, il indique que les fichiers nouvellement créés seront associés au group ID de ce répertoire.
- le dernier bit est le « sticky bit ». Ce bit indiquait à l'origine au système qu'il fallait garder le contenu du fichier en mémoire après exécution,

mais il n'est plus utilisé. Cependant, dans le cas d'un répertoire, il indique que seul le propriétaire d'un fichier du répertoire peut renommer, déplacer ou supprimer ce fichier. C'est typiquement utile dans le cas de répertoire temporairement partagés.

4.4.4 Utilisation d'Access Control Lists en UNIX

Le modèle traditionnel de contrôle d'accès aux fichiers de UNIX convient dans certaines circonstances, lorsqu'il y a peu de groupes d'utilisateurs et que les besoins d'accès restent simples et correspondent au découpage en groupes. Néanmoins, si le système nécessite de combiner de nombreux groupes et d'associer chaque utilisateurs à plusieurs d'entre eux, la gestion devient rapidement impraticable. Une solution à ce problème est d'utiliser des listes de contrôle d'accès (ACL).

Par exemple, FreeBSD permet à l'administrateur d'assigner une liste de user et group IDs à un fichier en utilisant la commande `setfacl`. N'importe quel nombre d'utilisateurs et de groupes peut être associé à un fichier, chacun avec les trois bits de protections habituels.

4.5 Types de mesures de contrôle d'accès

Une politique de contrôle d'accès implique des mesures à différents niveaux et de différents types [6] :

- Découragement : Il s'agit de décourager un attaquant potentiel en complexifiant les attaques potentielles (par ex : clôtures pour limiter l'accès physique à une ressource)
- Prévention : Il s'agit d'éviter qu'un incident se produise, par exemple en installant un anti-virus qui détectera les virus avant infection.
- Correction : Ces mesures permettront de corriger ou rétablir des composants après un incident (par exemple, nettoyage d'un fichier infesté)
- Récupération : Mesure permettant le retour à un fonctionnement normal, par exemple après restauration de données backupées avant l'incident
- Détection : La détection d'un incident peut s'effectuer en utilisant un IDS (Intrusion Detection System), qui détectera des accès non autorisés, ou permettra l'analyse ultérieure d'un incident.
- Compensatoire : L'idéal est de mettre en place différentes mesures qui vont se recouvrir l'une l'autre, afin d'optimiser la couverture du système. Ainsi, en cas de défaillance d'une mesure, il y en aura toujours une autre pour assurer la protection du système.
- Directive : Il s'agit d'une mesure obligatoire, suivant une réglemnetation.

4.6 Bonnes pratiques

Afin de conserver un niveau de sécurité acceptable, il est recommandé d'exécuter un certain nombre d'actions à intervalles réguliers :

- Interdire l'accès aux utilisateurs non authentifiés
- Limiter et suivre l'usage des comptes particuliers (ex : admin)
- Bloquer ou retarder l'accès après un certain nombre d'essais infructueux
- Retirer immédiatement les comptes des personnes qui quittent la société
- Suspendre les comptes inactifs depuis un certain temps (1 ou 2 mois)
- Suivre le principe du « Least Privilege »
- Désactiver les services inutiles
- Remplacer les mots de passe par défaut
- Limiter et suivre les règles d'accès
- Forcer le changement régulier de mot de passe
- Définir et exiger des caractéristiques minimales pour le mot de passe
- Protéger les fichiers de logs
- ...

Chapitre 5

Sécurité des bases de données et du Cloud Computing

5.1 La sécurité des bases de données

Les bases de données organisationnelles concentrent généralemnet des informations sensibles en un seul système logiques. Typiquement, on y trouvera les données financières de l'entreprise, des enregistrements d'appels téléphoniques confidentiels, des informations sur les clients et les employés, des informations propriétaires sur les produits, ou encore, des dossiers médicaux.

Pour les entreprises, il est important de fournir l'accès à ces informations, soit à leurs employés, soit à leurs partenaires, ou encore à leurs clients. Néanmoins, de telles informations pourraient se retrouver la cible d'attaques malveillantes, nuisant à leur confidentialité. Il y a donc un besoin critique de sécurisation spécifiques de ces bases de données, et la sécurité des bases de données représente un pan important de la stratégie de sécurité des entreprises.

En pratique, on remarque une différence en termes d'évolution entre les techniques de sécurisation des bases de données et la fiabilité accrue de ces dernières. Différentes raisons ont été avancées par [9] :

- Un déséquilibre important entre l'évolution des DBMS et les techniques de sécurité existantes. La complexité des DBMS et l'ajout de nouvelles fonctionnalités et services ont ouvert la porte à de nouvelles vulnérabilités.
- Les interactions avec les bases de données se font le plus souvent à travers le langage SQL, qui est par nature bien plus complexe que d'autres protocoles tels qu'HTTP.
- Une entreprise n'a en général pas de personnel dédié à la sécurité des bases de données. Elles comprennent souvent des administrateurs de bases de données, mais l'aspect sécurité est souvent négligé en termes de personnes-ressource.

– Beaucoup d'environnements de bases de données sont hybrides (BDD, OS, applications, ...), complexiant d'autant la tâche du personnel de sécurité.

L'utilisation croissante de technologies hébergées dans le cloud ajoute un challenge supplémentaire au personnel de sécurité, puisque les bases de données ont alors délocalisées.

5.2 Les injections SQL

Les attaques par injections SQL sont une des menaces réseaux les plus fréquentes d'après plusieurs études récentes. Elles sont conçues pour exploiter la nature des applications Web dynamiques, qui fournissent du contenu et des composants dynamiques. Ce aspect dynamique entraîne des interactions avec un base de données contenant un volume d'informations variées, allant des données relatives aux cartes de crédits aux informations sur les meilleures ventes de chaussures. Les applications Web effectuent donc des requêtes SQL sur les bases de données. C'est par ce canal que les injections SQL vont agir, en envoyant des commandes SQL malicieuses au serveur de base de données.

5.2.1 Principes

Pour introduire ses requêtes SQL malicieuses dans le système, le hackeur exploite des vulnérabilités de l'application Web, véhiculant ses requêtes dans du traffic autorisé par le firewall.

La manipulation des requêtes SQL repose sur l'insertion de nouvelles requêtes au milieu des requêtes normales générées par l'application Web sur base de l'input de l'utilisateur.

Imaginons par exemple un script construisant une requête SQL comme ceci :

```
var ShipCity;
ShipCity = Request.form("ShipCity");
var sql = "select * from OrdersTable where ShipCity = ‘"
+ ShipCity + "’";
```

Le script a été conçu pour que l'utilisateur entre le nom d'une ville. Par exemple, si l'utilisateur introduit « Bruxelles », cela donnera la requête SQL suivante :

```
select * from OrdersTable where ShipCity = ‘Bruxelles’
```

Cependant, si, au lieu d'une ville, l'utilisateur introduit ceci :

```
Liege’; DROP table OrdersTable--
```

la requête devient alors :

```
select * from OrdersTable where ShipCity = 'Liege'; DROP table OrdersTable--'
```

Le point-virgule permet la séparation des deux instructions, tandis que le double tiret marque le début d'un commentaire, invalidant donc tout ce qui sera concaténé à la suite. Le résultat de l'injection est donc la suppression de la table OrdersTable.

5.2.2 Points d'entrée des attaques

Les vecteurs principaux d'attaques par injection SQL sont les suivants :

- **L'input utilisateur** : L'attaquant injecte des commandes SQL à travers les interfaces prévues pour que l'utilisateur d'une application Web puisse introduire des données, c'est-à-dire typiquement les formulaires, envoyés à l'application Web par des requêtes HTTP POST ou GET.
- **Les variables du serveur** : Il s'agit d'une série de variables issues du serveur Web, tels que les en-têtes HTTP, les en-têtes de protocoles, et les variables d'environnement. Les applications Web utilisent ces variables de plusieurs manières, par exemple pour logger les statistiques d'utilisation et identifier les navigateurs utilisés. En agissant sur les en-têtes des protocoles, un attaquant peut accéder à la base de données si ces derniers ne sont pas nettoyés avant utilisation par l'application.
- **Injection de second ordre** : Ce type d'injection, plus subtil, ne se base pas sur l'exécution directe d'une requête SQL malicieuse, mais plutôt sur son stockage dans le système, avec exécution potentielle ultérieure. Cette attaque nécessite une connaissance approfondie du système cible. Elle exploite aussi souvent des techniques de protection contre les injections SQL mal mises en place.
- **Cookies** : Les cookies servent pour les clients qui reviennent sur un site web précédemment visité, afin de pouvoir réutiliser la configuration définie lors de la première visite. En manipulant un cookie, un attaquant peut injecter une requête SQL dans le système.
- **Input utilisateur physique** : Un attaquant peut également arriver à insérer des requêtes SQL par un autre canal que les requêtes Web. Cela peut être des code-barres, des tags RFID, ou même des formulaires papier.

5.2.3 Types d'attaque

Il existe trois types d'attaques : Les attaques « inband », les attaques par inférence, et les attaques « out-of-band ».

Attaques Inband

Les attaques du premier type utilisent le même canal de communication pour l'injection du code SQL et pour la récupération des résultats. Les informations récupérées sont alors affichées directement dans l'application Web. Ces attaques regroupent entre autres :

- Les tautologies : Cette forme d'attaque injecte une condition qui est toujours vraie, et permet par exemple de passer outre une vérification de mot de passe. Dans le script suivant, l'application injecte le mot de passe pour créer la requête SQL permettant la vérification de l'authentification :
  `$query$ = 'SELECT info FROM user WHERE name=`
  `$_GET["name"] 'AND pwd = '$_GET["pwd"]'";`
  Supposons que l'attaquant introduise `' OR 1=1 - '` dans le champ nom. La requête résultante serait alors :
  `SELECT info FROM user WHERE name= ' ' OR 1=1 -- and pwd = ' '`
  La condition étant alors une tautologie, la requête va renvoyer la liste de tous les utilisateurs.
- Les commentaires de fin de ligne : Cette attaque consiste à insérer le marquer de commentaire - pour annuler le code SQL légitime qui suit ce qui est injecté. L'exemple plus haut utilise ce principe.
- Les requêtes « piggybackée » : Cette attaque injecte une nouvelle requête au milieu d'une requête légitime. Les exemples présentés jusqu'ici illustrent aussi cette technique.

Attaques par inférence

Dans ce type d'attaques, il n'y a pas de transfert de données en tant que tel, mais l'attaquant observe la manière dont le système réagit à l'attaque pour en déduire des informations sur son comportement.

- Les requêtes illégales ou logiquement incorrectes : Pour collecter de l'information sur le système de base de données, l'attaquant tente de faire afficher au serveur une page ou un message d'erreur. Ces derniers étant souvent trop descriptifs, ils fournissent des données précieuses pour les attaquants.
- Les injections SQL à l'aveugle : Cette attaque permet d'extraire de l'information même pour des applications limitant l'affichage de messages d'erreur. En insérant des requêtes SQL nécessitant des réponses True/False, l'attaquant peut observer la réaction du serveur (page normale ou contenu différent), et en déduire si la requête a été évaluée à True ou False.

Attaque Out-Of-Band

Dans ce type d'attaque, les données sont récupérées par un autre canal, comme par exemple par l'intermédiaire d'un email. Cela est utilisé quand l'accès au données est bien sécurisé, mais qu'il existe des failles dans la connectivité sortante du serveur de base de données.

5.2.4 Contre-mesures

Les attaques par injection SQL sont tellement fréquentes, et ont un impact potentiel si important qu'il est nécessaire de mettre en place plusieurs contre-mesures. Ces mesures peuvent être classifiées en trois catégories :

1. Le code défensif : Il s'agit d'améliorer les pratiques de codage des développeurs, pour réduire les menaces d'injection. Cela consiste notamment à nettoyer toutes les données entrées par l'utilisateur. Il est aussi possible d'utiliser des classes types SQL DOM, pour encapsuler les requêtes.
2. La détection : Plusieurs méthodes existent pour détecter les attaques par injection. Il est possible d'essayer de reconnaître un pattern d'attaque sur base de sa signature, d'observer les anomalies, ou encore, l'analyse du code pour trouver les vulnérabilités SQL.
3. La prévention lors de l'exécution : Cette technique consiste à vérifier les requêtes au runtime, pour voir si elles sont conformes au modèle de requête attendu.

5.3 Le contrôle d'accès aux bases de données

Les systèmes de gestion de bases de données fournissent généralement un système de contrôle d'accès, permettant l'accès à la base de données aux utilisateurs authentifiés bénéficiant des droits d'accès ad-hoc. En plus de l'accès à la base de données en général, le système de gestion permet également de contrôler plus finement l'accès en définissant des droits s'appliquant sur des portions de la base de données.

Les systèmes de gestion commerciaux ou open source fournissent un contrôle d'accès soit par rôle (RBAC), soit discrétionnaire (DAC). Ils supportent plusieurs types de politiques administratives, dont :

1. Administration centralisée : Un petit nombre d'utilisateurs privilégiés attribuent ou révoquent les droits d'accès
2. Administration par le propriétaire : Le propriétaire d'une table attribue ou révoque les accès à cette table
3. Administration décentralisée : En plus d'attribuer les accès à la table, un propriétaire peut également déléguer ses droits d'autorisation à

d'autres utilisateurs, leur permettant alors d'également attribuer à leur tour ces droits à d'autres.

Un système de contrôle d'accès à une base de données peut attribuer des droits permettant la création, l'insertion, la suppression, la mise à jour, la lecture et l'écriture. La granularité peut également être très fine, et dans certains cas, il est même possible de donner des droits d'accès en fonction du contenu d'une entrée de la table (par exemple, un employé du service du personnel peut consulter les informations sur le salaire des employés uniquement jusqu'à un certain montant).

5.3.1 Définition des accès en SQL

SQL fournit deux commandes pour la gestion des accès : GRANT et REVOKE. La syntaxe exacte peut différer d'une version à l'autre, mais elle suit en général le pattern suivant :

```
GRANT  {privileges|role}
[ON   table]
TO {user|role|PUBLIC}
[IDENTIFIED BY password]
[WITH GRANT OPTION]
```

Par exemple, l'instruction suivante autorise l'utilisateur tartempion à faire des requêtes sur n'importe quelle table de la base de données :

```
GRANT SELECT ON ANY TABLE TO tartempion
```

Pour révoquer ce droit, il faut utiliser la commande suivante :

```
REVOKE SELECT ON ANY TABLE FROM tartempion
```

L'option `GRANT OPTION` permet d'attribuer le droit de donner des droits d'accès à un autre utilisateur. On a alors un modèle d'autorisations en cascade. Ce système peut être complexe lors de la révocation des droits. Considérons par exemple la figure 5.1, tirée de [3] :

Dans cette figure, les droits sont attribués en cascade au départ de l'utilisateur Ann. La chronologie de ces droits d'accès est représentée par les timestamps indiqués sur les flèches. La question qui se pose est de définir le comportement du système lorsque Bob révoque les droits d'accès de David : Est-ce que les droits d'accès doivent être retirés à David, ainsi que, logiquement, à tous ceux qui ont reçu leurs droits d'accès de ce même David ? Ou bien ne devrions-nous pas prendre en compte le fait que Chris a également attribué le droit d'accès à David ?

La convention suivie par la plupart des implémentations est la suivante : Lorsqu'un utilisateur A révoque un droit d'accès, tous les droits d'accès qui

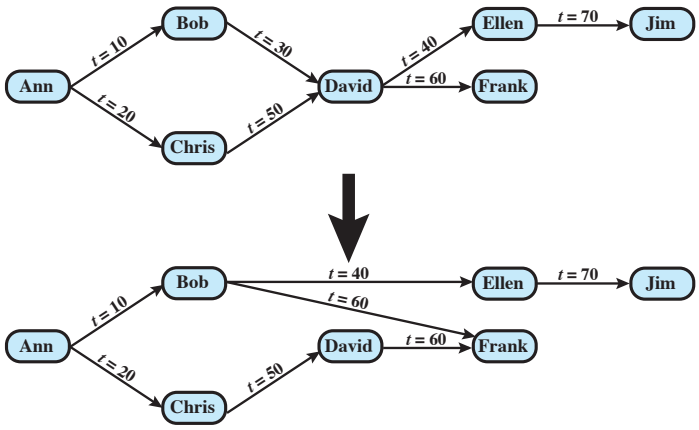

Figure 5.18 Bob Revokes Privilege from David, Second Version

FIGURE 5.1 – Révocation de privilèges dans un système d'autorisations en cascade

en découlent suite au système en cascade sont également révoquée, sauf si ils avaient malgré tout existé sans le droit d'accès attribué par A.

Dans la figure 5.1, lorsque Bob révoque le droit d'accès de David, tous les droits d'accès attribués par David avant que Chris ne lui donne également l'accès sont révoqués, à savoir les droits d'accès de Ellen et Jim. Le droit d'accès de Frank est maintenu, puisqu'il est postérieur à l'attribution du droit d'accès à David par Chris.

5.3.2 Contrôle d'accès par rôle

Le système d'accès par rôle convient particulièrement bien aux bases de données, vu qu'un système de base de données supporte parfois des douzaines d'applications, et qu'un utilisateur donné peut utiliser plusieurs applications pour effectuer des tâches très diverses, avec des privilèges différents. Il serait donc trop complexe de donner les droits d'accès au cas par cas.

Dans un système d'accès discrétionnaire, nous pouvons identifier trois catégories d'utilisateurs :

1. Le propriétaire de l'application, qui possède les objets de la base de données utilisée par l'application
2. L'utilisateur final, qui opère sur les objets d'une base de données à travers l'application, mais sans posséder ces objets
3. L'administrateur, qui est un utilisateur responsable d'une partie ou de toute la base de données.

Pour un système de rôle (RBAC), nous pouvons émettre des hypothèses sur ces trois catégories. Une application est en effet associée à un certain nombre de tâches, et pour chacune de ces tâches, il est possible de définir un ou plusieurs rôles spécifiant les accès nécessaires. L'utilisateur de l'application peut assigner des rôles aux utilisateurs finaux. Les administrateurs ont responsables des rôles plus généraux ou plus sensibles, incluent la gestion des composants physiques et logiques de la DB (ex : gestion des utilisateurs, des fichiers, de la sécurité, ...)

Une base de données RBAC doit donc fournir les fonctionnalités suivantes :

- Création et suppression de rôles
- Définition d'une permission pour un rôle
- Assignation et révocation d'un rôle à un ou plusieurs utilisateurs

Nous prendrons en exemple la gestion des rôles dans SQL Server. Ce système gère trois types de rôles : Les *fixed server roles*, les *fixed database roles*, et les *user-defined roles*. Les deux premiers types sont fixes, c'est-à-dire qu'ils sont pré-configurés et ne peuvent être supprimés ou modifiés. Il est juste possible d'ajouter ou de supprimer des utilisateurs à ces rôles.

La première catégorie a trait à la gestion au niveau du serveur, tandis que la seconde opère au niveau des bases de données individuelles. Le troisième

type permet la création de rôles pour définir des droits d'accès à des portions spécifiques de la base de données.

5.4 L'inférence

L'inférence, lorsqu'on parle de sécurité des bases de données, consiste à effectuer des opérations autorisées et d'en déduire des informations non accessibles.

En simplifiant, deux techniques peuvent être utilisées pour effectuer de l'inférence :

- Analyser les dépendances fonctionnelles entre des attributs dans une tables ou à travers plusieurs tables
- Regrouper des vues ayant les mêmes contraintes

Nous allons voir ça au travers d'un exemple, une fois encore tiré de Stallings, Computer Security, Principles and Practice [3]. La figure 5.2(a) montre une table avec quatre colonnes, sur laquelle sont définies deux vues, montrées à la figure 5.2(b). Ces deux vues sont définies en SQL comme suit :

```
CREATE view V1 AS CREATE view V2 AS
SELECT Position, Salary SELECT Name, Department
FROM Employee FROM Employee
WHERE Department = "strip" WHERE Department = "strip"
```

Les utilisateurs de ces vues n'ont pas l'autorisation d'accéder à la relation entre Name et Salary. Il n'est pas possible d'utiliser des dépendances fonctionnelles pour déduire cette relation, c'est-à-dire qu'il n'est pas possible de se baser sur une relation entre Name et Salary pour déduire Salary sur la base de Name et d'éventuellement d'autres informations.

Néanmoins, supposons que les deux vues ont été créées avec une contrainte telle que Item et Cost ne peuvent être atteinte ensemble. Un utilisateur qui connaît la structure de la table Inventory et qui sait que que les vues gardent le même ordre de ligne que la table Employee peut alors regrouper les deux vues pour construire la table montrée à la figure 5.2(c), violant dès lors la contrainte précitée.

Il existe deux approches pour gérer cette menace :

1. Détection d'inférence durant la conception de la DB : Cette approche altère la structure de la base de données ou change le contrôle d'accès pour éviter l'inférence, par exemple en supprimant des dépendances entre données en séparant une table en plusieurs tables, ou en utilisant un contrôle d'accès avec des rôles plus précis. Ces techniques ont tendance à réduire la disponibilité au système.
2. Détection d'inférence au moment des requêtes : La détection de violation est effectuée à la volée, lors des requêtes. Si une tentative est détectée, la requête est rejetée ou altérée.

Name	Position	Salary ($)	Department	Dept. Manager
Andy	senior	43,000	strip	Cathy
Calvin	junior	35,000	strip	Cathy
Cathy	senior	48,000	strip	Cathy
Dennis	junior	38,000	panel	Herman
Herman	senior	55,000	panel	Herman
Ziggy	senior	67,000	panel	Herman

(a) Employee table

Position	Salary ($)
senior	43, 000
junior	35,000
senior	48,000

Name	Department
Andy	strip
Calvin	strip
Cathy	strip

(b) Two views

Name	Position	Salary ($)	Department
Andy	senior	43,000	strip
Calvin	junior	35,000	strip
Cathy	senior	48,000	strip

(c) Table derived from combining query answers

Figure 5.8 Inference Example

FIGURE 5.2 – Exemple d'inférence

Dans les deux cas, il s'agit d'un problème difficile, toujours sujet à recherche.

Dans notre exemple, une solution serait de séparer la table Employee en trois tables : Employee, Salaries et Employee-Salary. Les deux premières peuvent être accessibles à un rôle peu sensible, tandis que la troisième n'est accessible qu'à des rôles ayant des autorisations spécifiques. Néanmoins, imaginons qu'on souhaite ajouter un attribut start_date à la table Salaries. Cette information peut facilement être associée à un employé par quelqu'un d'interne à l'entreprise, capable d'observer ou d'investiguer l'historique des employé. Cette information permet alors de reconstruire la dépendance entre les deux tables. Il serait alors plus sûr d'ajouter cette information à la table Employee.

5.5 Le chiffrement des bases de données

La base de données étant typiquement la ressource la plus critique d'une entreprise, elle doit être protégée par plusieurs niveaux de sécurité, tels que des firewalls, des mécanismes d'authentification, des systèmes de contrôle d'accès généraux et au niveau de la base de données, etc. Pour les données particulièrement sensibles, il est possible d'ajouter du chiffrement. Néanmoins, il y a deux désavantages à cette pratique :

- La gestion des clés : Les utilisateurs autorisés doivent disposer de la clé de déchiffrement. Or, ces utilisateurs peuvent être nombreux, ce qui complexifie la gestion sécurisée de cette clé.
- L'inflexibilité : Le chiffrement de la base de données complexifie la recherche de données.

Le chiffrement peut être appliqué à différents niveaux : Sur la table complète, au niveau des enregistrements, au niveau des colonnes, ou encore au niveau des champs.

5.6 Cloud Computing

La tendance actuelle dans beaucoup d'organisations, appelée Enterprise Cloud Computing, est de déplacer une grande partie des opérations IT dans une infrastructure connectée à Internet. Cette tendance fait émerger un certain nombre de problèmes de sécurité, spécialement dans le domaine de la sécurité des bases de données.

Une définition du Cloud COmputing est donnée par le NIST :

Définition

Le Cloud Computing est un modèle permettant un accès réseau pratique, à la demande et ubiquiste à un pool partagé de ressources informatiques (réseau, serveurs, stockage, applications et services), qui peut être rapidement provisionné et déployé avec un effort de gestion ou une interaction avec le fournisseur minimal.

Différents modèles et caractéristiques du Cloud Computing peuvent être utilisés. Ils sont montrés à la figure 5.3.

On y retrouve cinq caractéristiques essentielles du Cloud Computing :

1. Un accès réseau large : Les fonctionnalités sont accessibles par des mécanismes standards permettant l'utilisation de plateformes clients hétérogènes (PC, laptops, smartphones, ...)
2. Une élasticité rapide : Le Cloud Computing permet d'ajouter et de retirer des ressources au système en fonction de l'évolution des besoins.
3. Un service mesuré : L'utilisation de chaque service peut être monitorée, contrôlée et reportée afin de permettre un suivi précis pour le fournisseur et le client.
4. Un self-service à la demande : Le client peut automatiquement ajuster les ressources, tels que le stockage disque ou le temps serveur, sans interaction avec un opérateur humain.
5. Un pooling des ressources : L'ensemble des ressources du Cloud est regroupée de manière à pouvoir être attribuée ou réattribuée entre les utilisateurs, avec un certain degré de liberté par rapport à la localisation précise des ressources (malgré certaines contraintes possibles)

La figure 5.3 définit également trois modèles de service :

1. Software as a Service (SaaS) : La ressource disponible dans le Cloud est une application logicielle. Ce modèle suit le modèle bien connu des Services Web. L'application est typiquement accessible à travers une interface Web, depuis différents types d'appareils. GMail est un exemple de ce service.
2. Platform as a Service (PaaS) : La ressource fournie est une plateforme sur laquelle une organisation peut développer et faire tourner ses propres applications. Google App Engine est un exemple de PaaS.
3. Infrastructure as a Service (IaaS) : La ressource est ici au niveau de l'infrastructure, c'est à dire qu'il s'agit de machines, de matériel et/ou d'OS virtualisés, qui peuvent être controllés au travers d'une API. Windows Azure et Amazon EC3 sont deux exemples de IaaS.

Enfin, nous avons également quatre modèles de déploiement :

1. Cloud publique

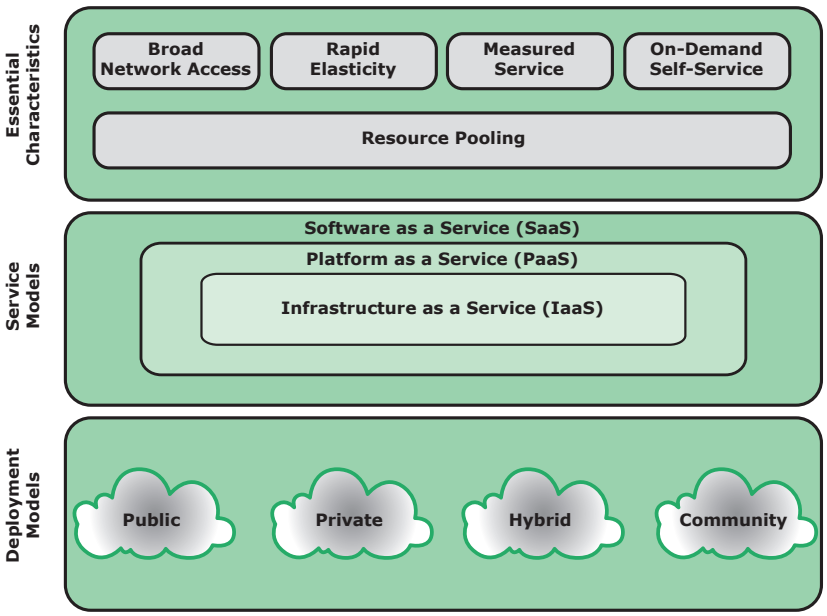

Figure 5.11 Cloud Computing Elements

FIGURE 5.3 – Eléments du Cloud Computing

2. Cloud privé
3. Cloud communautaire
4. Cloud hybride

Chacun de ces modèles varie au niveau des responsabilités, au niveau du fournisseur ou de la/les organisation(s) utilisant le Cloud.

5.7 Risques de sécurité et contre-mesures liés au Cloud Computing

Le contrôle de sécurité dans le cloud est relativement similiaire à celui de n'importe quel environnement IT, avec des risques supplémentaires spécifiques. Le risque principal vient de ce que l'organisation perd une certaine partie du contrôle sur les ressources, services et applications. Elle doit néanmoins assurer la tracabilité au niveau des politiques de sécurité et de confidentialité.

Les menaces principales listées par la Cloud Security Alliance sont les suivantes :

- Abus et usage malicieux du Cloud Computing : Il est relativement facile de souscrire à des ressources Cloud, et d'en détourner l'usage par exemple pour du spamming ou des attaques de déni de service. Il faut donc que les fournisseurs instaurent un système de souscription strict et surveillent l'activité du Cloud de manière approfondie.
- Les interfaces et API non sécurisées
- Les attaques internes : Les entreprises déléguant la gestion des ressources au fournisseur de Cloud n'ont pas de contrôle sur le personnel de ce dernier. Il faut donc être strict au niveau du contrat en ce qui concerne la politique de contrôle d'accès interne aux ressources du client.
- Les problèmes de technologies partagées : Les ressources du Cloud sont partagées entre diverses organisations. La séparation et l'isolation des ressources s'effectue à travers l'utilisation de machines virtuelles, mais il reste néanmoins des vulnérabilités dans ce modèle.
- Perte ou fuites de données : Il s'agit d'une des menaces les plus critiques pour les clients. Il faut donc prévoir des mesures de contrôle d'accès strict, un éventuel chiffrement (aussi pour le transit), et une gestion sévère des clés.
- Piratage de comptes ou de services : Les contre-mesures consistent à utiliser une autentification à deux niveaux, à monitorer de près l'activité, et à bien établir et comprendre les politiques et SLA du fournisseur.

5.8 Cloud Security As A Service

Les fournisseurs de service Cloud peuvent également fournir des services de sécurisation pour permettre de prendre en charge une partie de la politique de sécurité d'une entreprise. Cela comprend l'authentification, les anti-virus, les anti-malware/spyware, la détection d'intrusion, la gestion des événements, etc. On y retrouve également un élément important d'une politique de sécurité : Le Business Continuity and Disaster Recovery. Il s'agit des mesures visant à conserver la continuité des services lors d'une attaque, ou, en tout cas, à rétalir aussi vite que possible les services ou les ressources pour minimiser l'impact des événements. Cela inclut entre autres les backups et la possibilité de relocaliser rapidement les ressources.

Chapitre 6

Les malwares

Les malwares, ou « malicious softwares », sont une des catégories de menaces de sécurité informatiques les plus importantes. Ces menaces s'appliquent envers les softwares aussi bien applicatifs qu'utilitaires (éditeurs, compilateurs,...), ou encore au niveau du noyau de l'OS. Ces menaces s'appliquent également au niveau de sites ou de serveurs Web, ou encore par l'intermédiaire de spam ou autre messages.

Une définition possible des malwares est :

Définition

Un malware est un programme qui est inséré dans un système, généralement de manière dissimulée, avec l'intention de compromettre la confidentialité, l'intégrité ou la disponibilité des données, des applications, du système d'exploitation, ou de toute autre ressource de la victime.

Nous allons ici passer en revue les caractéristiques des malwares, en les classifiant d'une part sur base de leur moyen de propagation, et d'autre part sur base des actions que ces malwares effectuent sur le système infecté.

6.1 Types de malwares

La table représentée aux figure 6.1 et 6.2 reprend et définit les principaux malwares rencontrés de nos jours.

6.2 Mode de propagation

Le mode de propagation est la première caractéristique des malwares que nous allons aborder. Nous allons étudier trois types de propagation : Via du contenu infecté (les virus), via l'exploitation de vulnérabilités (les vers), et via des techniques de social engineering (spam et chevaux de troie).

Name	Description
Advanced persistent threat	Cybercrime directed at business and political targets, using a wide variety of intrusion technologies and malware, applied persistently and effectively to specific targets over an extended period, often attributed to state-sponsored organizations.
Adware	Advertising that is integrated into software. It can result in pop-up ads or redirection of a browser to a commercial site.
Attack Kit	Set of tools for generating new malware automatically using a variety of supplied propagation and payload mechanisms.
Auto-rooter	Malicious hacker tools used to break into new machines remotely.
Backdoor (trapdoor)	Any mechanisms that bypasses a normal security check; it may allow unauthorized access to functionality in a program, or onto a compromised system.
Downloaders	Code that installs other items on a machine that is under attack. It is normally included in the malware code first inserted on to a compromised system to then import a larger malware package.
Drive-by download	An attack using code in a compromised web site that exploits a browser vulnerability to attack a client system when the site is viewed.
Exploits	Code specific to a single vulnerability or set of vulnerabilities.
Flooders (DoS client)	Used to generate a large volume of data to attack networked computer systems, by carrying out some form of denial-of-service (DoS) attack.
Keyloggers	Captures keystrokes on a compromised system.
Logic bomb	Code inserted into malware by an intruder. A logic bomb lies dormant until a predefined condition is met; the code then triggers an unauthorized act.
Macro Virus	A type of virus that uses macro or scripting code, typically embedded in a document, and triggered when the document is viewed or edited, to run and replicate itself into other such documents.
Mobile Code	Software (e.g., script, macro, or other portable instruction) that can be shipped unchanged to a heterogeneous collection of platforms and execute with identical semantics.
Rootkit	Set of hacker tools used after attacker has broken into a computer system and gained root-level access.
Spammer Programs	Used to send large volumes of unwanted e-mail.

FIGURE 6.1 – Terminologies des malwares (d'après Stallings, Computer Security, Principles and Practice [3])

Spyware	Software that collects information from a computer and transmits it to another system by monitoring keystrokes, screen data and/or network traffic; or by scanning files on the system for sensitive information.
Trojan horse	A computer program that appears to have a useful function, but also has a hidden and potentially malicious function that evades security mechanisms, sometimes by exploiting legitimate authorizations of a system entity that invokes the Trojan horse program.
Virus	Malware that, when executed, tries to replicate itself into other executable machine or script code; when it succeeds the code is said to be infected. When the infected code is executed, the virus also executes.
Worm	A computer program that can run independently and can propagate a complete working version of itself onto other hosts on a network, usually by exploiting software vulnerabilities in the target system.
Zombie, bot	Program activated on an infected machine that is activated to launch attacks on other machines.

FIGURE 6.2 – Terminologies des malwares (d'après Stallings, Computer Security, Principles and Practice [3])

6.2.1 Les virus

Un virus est un morceau de programme qui peut infecter d'autres programmes ou tout type de contenus exécutables (ex : macros Office, documents PDF), en les modifiant. Les virus sont apparus au début des années 80. L'analogie avec les virus biologiques vient de la capacité de ces derniers à exploiter la machinerie d'une cellule pour la forcer à produire des centaines de répliques du virus original. Le virus informatique procède exactement de la même manière, en incluant dans son code les instructions pour se répliquer. Lorsque le virus se retrouve intégré à un programme, dès qu'il est en contact avec un nouvel exécutable, il se copie et s'intègre à ce nouvel hôte, se propageant ainsi de proche en proche. Le virus peut également se propager sur d'autres machines par l'intermédiaire de clés USB ou d'échange par pièce jointe d'email, entre autres.

Les virus sont généralement composés de trois parties :

1. Un mécanisme/vecteur d'infection
2. Un élément déclencheur (bombe logique)
3. Une action, comme par exemple la destruction de données.

Dans leur cycle de vie, ils traversent typiquement quatre phases distinctes :

1. Une phase dormante (facultative) : Le virus attent d'être actionné par un événement (une date, la présence d'un autre programme ou fichier, ou le dépassement d'un quota disque)
2. Une phase de propagation : Le virus se copie dans d'autres programmes ou d'autres zones du système. La copie peut ne pas être entièrement

identique pour éviter la détection. Chaque copie continue alors la phase de propagation.

3. La phase de déclenchement : Le virus est activé et se prépare à effectuer l'action qui lui est attribuée. Là aussi, de nombreux événements peuvent déclencher cette phase, dont entre autres le nombre de duplications que le virus a pu effectuer.

4. La phase d'exécution : L'action est effectuée. Elle peut être inoffensive, comme un message sur l'écran, ou nocive, telle que la destruction de programmes ou de fichiers.

La figure 6.3(a) montre le pseudo-code d'un virus simple. Le virus commence par sélectionner une cible, non encore infectée (vérification du code de début de virus). Il se copie au fichie-cible, puis effectue la vérification de la condition. Enfin, si cette condition est remplie, il effectue son action. Ce type de virus est facile à détecter, car il modifie la taille du fichier infecté. Une variante, montré à la figure 6.3(b), permet au virus de conserver la taille du fichier en compressant l'exécutable original.

6.2.2 Les vers

Un ver est un programme qui cherche de manière active de nouvelles machines à infecter, et chaque nouvelle machine infectée sert de point de départ pour une nouvelle campagne d'attaques. Les vers exploitent les vulnérabilités software dans les programmes clients ou serveurs, afin de gagner l'accès à de nouveaux systèmes. Ils utilisent les connexions réseaux, mais aussi les médias partagés comme les clés USB ou les CDs ou DVDs de données. Les vers peuvent aussi se propager dans des macros ou des scripts attachés à des emails. Comme les virus, les vers ont une phase de propagation, ainsi qu'une ou plusieurs actions associées.

Les vers utilisent typiquement les moyens suivants pour se répliquer :

- Les emails ou les messageries instantanées, via les pièces jointes
- Le partage de fichiers
- La possibilité d'exécution à distance, en utilisant des failles de sécurité
- Les possibilités de transfert ou d'accès à distance à des fichiers
- Les possibilités de connexion à distance, ce qui permet au ver de se copier lui-même sur le système distant en utilisant le login de l'utilisateur victime

Pour découvrir de nouvelles cibles, les vers utlisent les stratégies de scan suivantes :

- Scan aléatoire : Chaque hôte compromis scanne un espace d'IP aléatoire, en utilisant un seed différent à chaque fois. Cette technique produit un important volume de trafic, qui peut impacter la connectivité avant même que l'attaque ne soit lancée.

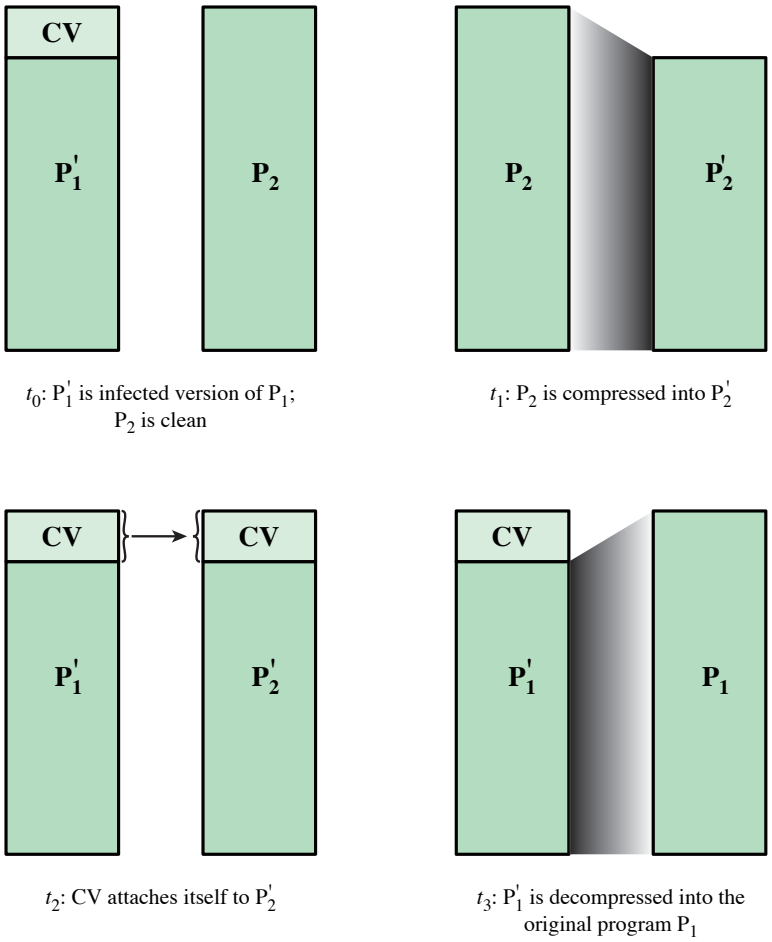

Figure 6.2 A Compression Virus

FIGURE 6.3 – Exemple de virus (Tiré de Stallings, Computer Security, Principles and Practice [3])

- La hit-list : L'attaquant compile une liste d'IPs de machines potentiellement vulnérables. Il commence ensuite par infecter les premières machines de la liste, qui à leur tour se chargent de l'infection d'un sous-ensemble de la liste.
- Le scan topologique : Le ver utilise les informations trouvées sur la machine infectée pour trouver d'autres hôtes à scanner
- Le réseau local : Si l'infection a pu passer un firewall, l'hôte tente alors de trouver des cibles sur le réseau local.

6.2.3 Spam et chevaux de Troie

Le troisième type de propagation consiste à tromper les utilisateurs pour qu'ils compromettent eux-même leur système sans le savoir, en répondant à un spam ou en permettant l'installation et l'exécution d'un programme ou un script Cheval de Troie.

Le Spam

L'utilisation de l'email comme moyen de communication privilégié a permis l'émergence de l'envoi d'emails non sollicités, appelés Spam. Des évaluations récentes avancent le chiffre d'une proportion de 90% d'email non sollicités parmi l'ensemble des emails envoyés. Cela a pour conséquence d'énormes coûts tant au niveau de l'infrastructure réseau, pour relayer un tel traffic, qu'au niveau des utilisateurs, pour filtrer les mails légitimes du spam. Des produits anti-spam sont apparus, et, comme dans d'autres domaines de la sécurité, il en résulte une lutte technologique entre les spammeurs et les anti-spams.

La croissance dans le volume de spam aurait cependant tendance à se stabiliser. Une hypothèse possible est le développement d'attaques alternatives via les réseaux sociaux. Ces services contiennent en effet de nombreuses potentialités d'exploitations d'un nouveau type.

L'origine des spams est de moins en moins souvent localisée en un point précis, mais tend plutôt à prendre source depuis des réseaux de machines infectées, appelés botnets.

Les spams peuvent être de la simple publicité ou des tentatives d'escroquerie, mais ils peuvent également servir de vecteurs de propagation pour d'autres malwares, via des pièces jointes. Le spam peut également servir de point de départ à une attaque par phishing, en redirigeant l'utilisateur vers un site web reproduisant une interface de connexion à un service connu.

Les chevaux de Troie

Définition
Un cheval de Troie est un programme ou un utilitaire utile ou en apparence utile, contenant des instructions cachées qui, une fois exécutées, effectuent des actions nocives.

Un cheval de Troie peut être utilisé pour accomplir des actions indirectement, action que l'attaquant ne pourrait pas effectuer de lui-même, comme par exemple l'accès à des informations personnelles. Le Cheval de Troie peut alors par exemple scanner les fichiers de l'utilisateur cible, retrouver l'information souhaitée et en envoyer une copie à l'attaquant via un formulaire Web, un email ou un message texte.

Les chevaux de Troie peuvent par exemple se faire passer pour des utilitaires anti-virus dernier cri, ou pour des mises à jour de sécurité importantes, afin de convaincre la cible de l'installer.

6.3 Types d'actions

Une fois que le malware est actif sur le système, il peut effectuer différentes actions sur ce dernier. Certains ne font rien, ou font des actions insignifiantes et ont pour but principal de se propager, mais souvent, ils sont vecteurs d'actions aggressives.

6.3.1 La corruption de systèmes

La destruction de données

Un premier type d'action est la destruction pure et simple de données. Le virus Chernobyl en est un exemple. Il visait les système Windows 95 et 98, et, lorsque déclenché, à une certaine date, il supprimait des données sur le système en écrivant des 0 sur le premier megaoctet du disque dur, ce qui corrompait donc le système entier.

Une alternative à la destruction de données est le chantage : Certains malwares chiffrent les données, et le maître chanteur demande donc à la victime de lui envoyer une rançon en échange de la clé de déchiffrement.

Les dommages physiques

Un autre type de corruption de système consiste à endommager des équipements physiques. Cela peut se faire par exemple en ré-écrivant le code du BIOS du système cible, ou en visant la partie logicielle des systèmes de contrôle de processus industriels (ex : centrifugeuse dans le programme iranien d'enrichissement d'uranium).

6.3.2 Les agents d'attaque

Un autre type d'action effectué par les malwares consiste à détourner le système attaqué de son usage premier et à l'utiliser comme agent pour d'autres attaques. Le système est alors appelé « bot », pour robot, ou encore zombie. De cette manière, il devient bien plus difficile de remonter jusqu'à la source de l'attaque. Les bots, et les réseaux de bots (botnets) sont typiquement utilisés pour :

1. Lancer des attaques de type Déni de Service Distribué (DDoS)
2. Pour l'envoi de spam
3. Pour sniffer le traffic
4. Pour le keylogging
5. Pour la propagation de nouveaux malwares
6. Pour l'automatisation de clics publicitaires
7. Pour la manipulation de jeux ou de paris en ligne

A la différence des vers qui sont indépendants et autonomes, les bots peuvent être contrôlés à distance via le réseau. Cela se faisait à l'origine en utilisant les réseaux IRC, mais les botnets plus récents évitent ces mécanismes et préfèrent des canaux de communications plus classiques comme HTTP.

6.3.3 Le vol d'information

Le troisième type d'action effectué par le malware est le vol d'informations, tel que, par exemple, celui du login et du mot de passe de la victime sur des sites de banking, de jeu ou autre. Ces actions peuvent éventuellement, mais plus rarement, viser des documents ou des informations de configuration à des fins de reconnaissance ou d'espionnage.

Le vol de login/mot de passe peut s'effectuer à l'aide d'un keylogger, qui capture les frappes de l'utilisateur au clavier. D'autres spywares plus génériques permettent d'obtenir le même résultat.

Le phishing est une autre technique qui consiste à attirer, depuis un spam, une victime potentielle sur un site web reproduisant une interface de connexion d'un service connu, et à obtenir les informations de connexion en la trompant.

D'autres attaques, plus complexes, peuvent viser des cibles plus spécifiques, comme par exemples de grandes organisations.

6.3.4 La furtivité

La dernière catégorie d'actions effectuées par les malwares concerne les actions furtives, c'est-à-dire les techniques utilisées par les attaquants pour

cacher leur présence sur un système. Il s'agit typiquement des backdoors, qui sont un point d'entrée secret non autorisé à l'intérieur d'un système. De nos jours, il s'agit typiquement d'un service réseau ouvert sur un port non standard, auquel l'attaquant peut se connecter et effectuer des commandes sur le système.

Un autre type de malware furtif est le rootkit, qui est installé sur un système afin de couvrir un accès root illicite, en dissimulant autant que possible sa présence. L'attaquant a alors un accès complet au système, et peut donc changer les programmes ou les fichiers, surveiller les processus ou les accès réseaux, et obtenir une backdoor à la demande.

6.4 Les contre-mesures

Les contre-mesures permettant de contrer les malwares sont connues sous le terme « anti-virus », car ils furent initialement développés pour contrer les infections virales. Néanmoins, ils protègent actuellement contre la plupart des malwares mentionnés dans ce chapitre.

La solution idéale contre les infections par malware est évidemment la prévention. C'est néanmoins difficile à appliquer, bien qu'il soit possible de diminiuer la propabilité d'infection en sécurisant les systèmes et en sensibilisant les utilisateurs pour qu'ils évitent les risques d'infection.

La première action à entreprendre en prévention est donc de s'assurer que tous les systèmes soient à jour et de s'assurer que les patchs de sécurité soient appliqués, afin de minimimser les vulnérabilités du système. La seconde est de configurer soigneusement le contrôle d'accès aux applications et aux données sensibles du système. Et la troisième est donc de former les utilisateurs pour qu'ils puissent reconnaître et éviter les situations les exposant à des malwares.

Si la prévention échoue, il faut alors pouvoir :

1. Détecter l'infection et localiser le malware
2. Identifier le malware
3. Retirer le malware et toute trace de son passage sur tous les systèmes infectés

Parfois, la suppression du malware n'est pas possible. Il faut alors supprimer le fichier infecté. En fonction de la complexité du malware, cela peut impliquer de nettoyer complètement le système de stockage de données et de tout réinstaller proprement.

Chapitre 7

Attaques par Déni de Service (DoS)

7.1 Généralités

Les attaques par déni de service (DoS), et plus particulièrement les attaques distribuées (DDoS) existent depuis longtemps, et leur puissance a accru de manière importante au cours des années. Ainsi, en 2010, un ensemble de site Web ayant coupé leurs liens avec des sites controversés liés à Wikileaks ont été la cible d'attaques sans précédent impactant leur joignabilité.

Avec la croissance de la bande passante des liens Internet, les attaques sont passées d'un modeste 200 MBps en 2002 à 100GBps en 2010, voire même 300GBps dans le cas de l'attaque Spamhous en 2013. Des attaques de l'ordre de 50GBps sont actuellement suffisantes pour dépasser la capacité de n'importe quelle cible, y compris les IXP ou les serveurs de noms DNS, mais des attaques de moindre importance sont déjà étonnamment efficaces. Les attaques DDoS peuvent en outre parfois servir de diversion, par exemple dans le cas de systèmes banquaires, pour pouvoir par ailleurs mener une réelle attaque sur les switches ou les réseaux de paiement ATM.

Une attaque par déni de service peut se définir comme suit (NIST) :

Définition

Un déni de service (DoS) est une action qui empêche ou complique l'usage légitime de réseaux, systèmes ou applications en saturant les ressources telles que le CPU, la mémoire, la bande passante ou l'espace disque

Il y a donc trois catégories de ressources qui sont ciblées par ces attaques :

– **La bande passante réseau** : En saturant les liens réseaux avec d'énormes volumes de trafic, l'attaque empêche les paquets des uti-

lisateurs légitimes d'accéder au service
- **Les ressources système** : Une attaque DoS visant les ressources système utilise un nombre limité de paquets réseaux, mais ces paquets sont spécialement destinés à consommer les ressources systèmes (par exemple, les buffers, les tables de connexion ouvertes, etc.). C'est le cas de l'attaque par SYN spoofing Une autre attaque de ce type consiste à envoyer des paquets spécifiques (poison packets) déclenchant un bug dans la partie réseau du système, le faisant crasher. Les attaques *ping of death* et *teardrop* fonctionnent selon ce principe.
- **Les ressources applicatives** : Par exemple, une attaque visant un serveur web consistera à envoyer un grand nombre de requêtes valides, consommant les ressources du serveur et l'empêchant de répondre à d'autres requêtes. Les requêtes utilisées peuvent être des requêtes SQL coûteuses, qui chargent le serveur de manière importante (*cyberslam*). Il est également possible d'utiliser le principe des poison packets pour faire crasher le serveur.

Les attaques DoS peuvent aussi être caractérisées par le nombre de systèmes qui sont utilisés pour diriger le trafic vers la cible. A l'origine, une seule ou quelques machines étaient utilisées, directement sous le contrôle de l'attaquant. Cela fonctionne typiquement pour les attaques visant les bugs des serveurs et de systèmes. Les attaques nécessitant de gros volumes de trafic sont par contre souvent déclenchées au départ de plusieurs systèmes, en utilisant des formes distribuées ou amplifiées de déni de service.

7.1.1 Attaques DoS classiques

L'attaques DoS la plus simple consiste donc à surcharger la capacité de la connection à l'internet de la cible. L'attaquant doit donc disposer d'un réseau avec une large bande passante, plus large que celle de la cible. La forme la plus simple de l'attaque consiste par exemple à envoyer massivement des requêtes ping (ping flood) vers la cible. Ces requêtes consomment l'ensemble des ressources du lien de la cible, et l'excédent est jeté par le routeur de bordure. Le trafic légitime a dès lors peu de chance d'atteindre le réseau cible, et aura de forte probabilité d'être lui aussi jeté au niveau de ce routeur.

La première faiblesse de cette attaque est que l'adresse de l'attaquant peut facilement être retrouvée en observant l'IP source des paquets ICMP. La seconde faiblesse est qu'en général, les paquets ICMP déclenchent une réponse, qui sera à son tour renvoyée vers la source de l'attaque, dégradant les ressources du réseau de départ et accroissant dès lors les chances de voir l'attaque détectée.

Les attaques DoS peuvent être améliorée en utilisant le spoofing d'adresse source : En utilisant les raw sockets, l'attaquant peut forger des paquets IP en usurpant n'importe quelle adresse source. La source de l'attaque est alors plus difficile à détecter, et mieux encore, les paquets ICMP en réponse seront

dirigés vers les adresses sources usurpées, qui peuvent éventuellemnet être des machines valides qui renverront à leur tour des messages d'erreur vers la cible, accroissant alors l'attaque.

La solution au spoofing d'adresse consiste à ce que les fournisseurs d'accès filtrent, à l'entrée de leur réseau, les adresses IP sources non contenues dans le range d'adresse attribuées au client. Malheureusement, de nombreux ISP n'implémentent pas ce filtrage, et les attaques DoS avec spoofing d'adresses continuent à se produire.

Un autre type d'attaque DoS classique est le SYN Spoofing. Cette attaque vise à saturer la pile TCP des systèmes visés. Pour cela, l'attaquant génère un grand nombre de demande de connexions TCP (paquet SYN), avec une adresse source spoofée. Le serveur cible enregistre chacune de ses demandes et envoie le paquet SYN+ACK à l'adresse source spoofée. Si cette adresse correspond à un système valide, il répondra par un RST pour annulé la demande de connexion. Néanmoins, dans le cas où il n'y a pas de système derrière l'adresse source la demande de connexion restera ouverte dans la table des connexions TCP. L'attaque par SYN Spoofing se poursuivant, le taux d'occupation de la table est maintenu malgré les time-out sur les entrées, et le serveur visé reste injoignable.

Il y a une grande différence au niveau du trafic entre l'attaque par SYN spoofing et l'attaque par ping flooding mentionnée plus haut. L'attaque par SYN spoofing nécessite comparativement beaucoup moins de traffic, et la bande passante nécessaire est beaucoup plus basse que la capacité du lien réseau, puisque la cible est la table des connexions TCP et pas le lien lui-même.

Ces deux attaques sont les premières attaques DoS apparues sur l'Internet, mais d'autres variantes se sont développées depuis. Nous allons d'abord explorer d'autres types d'attaques par flooding, puis étudier les attaques distribuées et les attaques au niveau applicatifs avant de détailler les mécanismes permettant d'amplifier les DoS. Nous terminerons en abordant les contre-mesures.

7.2 Attaques par flooding

Les attaques par flooding sont caractérisées par le protocole réseau utilisé pour l'implémentation. Le principe commun est de causer la congestion du lien réseau du serveur visé pour bloquer le trafic légitime à destination de ce serveur et le rendre indisponible.

N'importe quel type de paquet réseau peut être utilisé pour ce type d'attaque. La seule contrainte est que le type de paquet ait la capacité d'arriver jusqu'à la cible. Typiquement, ces attaques utilisent des paquets de type ICMP, UDP ou TCP SYN.

7.2.1 ICMP flood

Nous avons déjà parlé des attaques utilisant les ping `echo request`. C'est l'exemple classique d'attaque par ICMP Flooding. L'avantage des paquets ICMP est qu'ils ont été conçus pour permettre la gestion et le monitoring par les administrateurs du réseau. Ils constituent donc un outil de diagnostic puissant. Actuellement, beaucoup d'organisation restreignent l'utilisation des paquets ICMP et les filtrent lors des passages par les firewalls. Les attaquants ont donc commencé à utiliser d'autres types de paquets ICMP, en préférant ceux qui sont indispensables au bon fonctionnement de la pile TCP/IP, et qui seront donc plus souvent autorisés par les firewalls. De tels paquets ICMP sont par exemple les ICMP `time exceeded` et `destination unreachable`.

Un autre avantage du flood ICMP est que les paquets peuvent être assez volumineux, puisqu'ils peuvent contenir un extrait d'un paquet pour lequel une erreur est reportée. Cela permet d'occuper la bande passante de manière efficace.

7.2.2 UDP flood

Une alternative à l'utilisation d'ICMP est l'envoi massif de segments UDP vers un port particulier, comme par exemple le service de diagnostic echo, activé par défaut sur de nombreux serveurs. Si le service tourne, le serveur renvoie un segment UDP en réponse. Dans le cas contraire, le segment est jeté, mais peut donner lieu à l'envoi d'un message ICMP en retour, ce qui accroit d'autant le volume de l'attaque.

Si l'attaque vient d'une source unique, le spoofing d'adresse source est généralement utilisé. Si l'attaque vient d'un botnet, les adresses des systèmes compromis peuvent être gardées, puisqu'il sera de toute façon difficile de remonter à la source de l'attaque.

7.2.3 TCP SYN flood

Outre les floods UDP et ICMP, il est également possible de créer une attaque de Déni de Service en utilisant des segments TCP au système cible. Une première possibilité est d'envoyer un flux important de demandes de connexion TCP. Cette attaque est appelé SYN flooding, car elle vise à surcharger la bande passante, au contraire du SYN spoofing, qui vise à saturer la table des connexions au niveau du système.

Il est également possible d'utiliser des segments TCP de données. Ces segments sont rejetés par la cible, mais cela n'a pas d'importance encore une fois, puisque l'objectif est toujours la congestion du lien.

Ces techniques d'attaques sont limitées par le volume de données que la source peut générer, s'il s'agit d'une source unique. Dans ce cas, il est en plus possible d'identifier l'attaquant (modulo utilisation du source spoofing). Assez logiquement, des techniques d'attaques distribuées se sont donc

développées, afin de pouvoir augmenter sensiblement le volume de données généré et de dissimuler l'origine de l'attaque.

7.3 DDoS

Dans les attaques par déni de service distribuées, l'attaquant s'arrange pour infecter une série de machines intermédiaires à l'aide de malwares par lesquels il est alors possible d'installer des agents permettant le contrôle du système et le contrôle du lancement de l'attaque. De tels systèmes intermédiaires sont appelés **zombies**. Un ensemble de zombies est appelé un **botnet**.

Afin d'éviter de pouvoir remonter à la source d'une attaque, les communications entre l'attaquant et le botnet (comportant éventuellement plusieurs niveaux de contrôle) peuvent être chiffrées.

Il est évident que pour éviter de se retrouver impliqué involontairement dans de telles attaques DDoS, il faut déployer toutes les précautions nécessaires pour éviter l'infection par malware.

7.4 DoS applicatives

Lorsque l'objectif du DoS est d'impacter un système et non un lien, les techniques les plus efficaces consistent à forcer la cible à exécuter des opérations coûteuses en ressources. Cela permet, en visant une application spécifique et en utilisant une faible bande passante, d'augmenter facilement le taux d'occupation de la cible. Nous allons examiner deux types d'attaques DoS applicatives.

7.4.1 SIP flood

L'attaque SIP flood exploite le fait que, dans le protocole SIP, une simple requête INVITE envoyée à un proxy peut déclencher des opérations relativement coûteuses, puisque le proxy doit utiliser le DNS pour retrouver l'IP du serveur proxy de la destination, avant de transférer la requête à ce dernier. En envoyant un flux de requêtes INVITE, l'attaquant surcharge le proxy de deux manières : en occupant la bande passante du lien, et en surchargeant le CPU du proxy. En outre, la destination est également impactée par l'attaque, puisqu'elle recevra de multiples appels VoIP simultanément.

7.4.2 Attaques HTTP

HTTP peut également servir de support à des attaques DoS, et ce, de deux manières.

HTTP flood

Une première possibilité est de générer une attaques DDoS en bombardant un serveur web de requêtes HTTP. Les requêtes peuvent être spécialement conçues pour occuper un maximum de ressource au niveau du serveur, par exemple en demandant expressément un fichier de grande taille, que le serveur doit lire sur le disque, charger en mémoire, puis découper en morceaux afin de pouvoir être envoyé sur le réseau. Cela consomme donc des ressources au niveau de la mémoire, du CPU et de la bande passante. Une variante, appelée **recursive HTTP flood** ou **spidering**, consiste à envoyer une requête à partir d'un lien donné, et de suivre tous les liens sur la page reçue de manière récursive.

Slowloris

Slowloris est une attaque consistant à occuper les ressources d'un serveur web à partir d'une faible bande passante, en envoyant des requêtes HTTP partielles qui ont pour effet d'occuper les sockets du serveur. Lorsque les requêtes Slowloris ont réussi à monopoliser tous les sockets TCP et les threads associés, les autres utilisateurs ne sont plus en mesure d'accéder au serveur web. Slowloris continue à envoyer continuellement des morceaux de requêtes HTTP, à intervalles inférieurs au timeout du serveur web, afin de garder la connexion HTTP ouverte.

Pour éviter ce type d'attaque, il est possible de limiter le nombre de connexions entrantes venant d'un même hôte. Cela ne fonctionne bien sûr pas en cas d'attaque DDoS. Il est aussi possible de faire varier le timeout du serveur en fonction du nombre de connexions actives.

7.5 Attaques par réflexion et amplification

Les attaques par réflexion et amplification sont une variante de l'attaque DDoS dans laquelle on utilise comme relais, non pas des zombies, mais des systèmes fonctionnant normalement. Une fois encore, c'est le spoofing d'adresse source qui permet la mise en place de cette attaque DoS. L'attaquant envoie un paquet avec une adresse source spoofée à un service donnée sur un serveur quelconque. Le serveur répond au paquet, en l'envoyant au propriétaire légitime de l'adresse spoofée, qui constitue la cible effective de l'attaque. En envoyant une série de telles requêtes à différents serveurs, les réponses seront toutes redirigées vers la cible, saturant donc la bande passante de cette dernière.

Ces attaques sont simples à déployer, parce qu'elles utilisent des mécanismes réseaux et des systèmes légitimes pour véhiculer l'attaque. Il existe deux attaques dérivant de ce principe : L'attaque par réflexion, et l'attaque par amplification.

7.5.1 Attaques par réflexion

L'attaque par réflexion se base donc sur des relais intermédiaires pour rediriger un flux vers une cible. Les relais seront choisis pour leur large bande passante, afin de pouvoir accroitre le volume de traffic vers la cible, mais également pour noyer l'attaque dans le traffic régulier du réseau intermédiaire. Le service intermédiaire visé sera choisi pour sa capacité à générer des réponses de taille plus large que les requêtes, ce qui permet d'augmenter le volume de traffic de l'attaque. Les services UDP classiques peuvent être utilisés à cette fin, comme par exemple le service echo, le DNS ou SNMP. Dans le cas du DNS, en envoyant des requêtes spécifiques pour des Resource Records de grande taille, il est possible d'obtenir des réponses jusqu'à 512 octets (et plus pour IPv6) avec une simple requête de 60 octets. La récursion DNS peut également être exploitée pour amplifier la taille de la réponse.

Une autre variante de cette attaque utilise les paquets TCP SYN, ce qui a pour effet de génerer un flux de SYN+ACK en direction de la cible, surchargeant là aussi le lien. En pratique, n'importe quel service TCP fait également l'affaire. C'est d'ailleurs ce qui rend ce type d'attaque très difficile à détecter, puisque les requêtes de connexion TCP ont l'air légitimes en apparence. C'est uniquement sur base d'un IDS que les requêtes avortées en provenance d'une source spécifique peuvent être détectées comme attaque DoS. En cas de DDoS, c'est encore plus problématique, car tous les flux ne peuvent être détectés.

Clairement, la seule possibilité de bloquer ce type d'attaque est de prévenir l'utilisation d'adresses source spoofée, en implémentant des filtres aux frontières des ISPs afin de n'accepter que les paquets correspondant bien aux ranges d'IP des clients de ces derniers. Hélas, ces filtres ne sont pas largement déployés, ce qui laisse donc la porte ouverte à d'autres attaques DoS.

7.5.2 Attaques par amplification

Les attaques par amplification varient de celles par réflexion par le fait qu'un message initial va arriver à déclencher de multiples paquets en réponse, redirigés vers l'adresse IP spoofée. Par exemple, en dirigeant le message initial vers une adresse destination de broadcast dans un réseau, tous les hôtes de ce réseau peuvent potentiellement répondre. La seule difficulté est de trouver un service utilisé par tous les hôtes du réseau intermédiaire. Le ping `echo request` en est un bon exemple. Le programme DoS `smurf`, populaire pendant un temps, utilisait notamment ce mécanisme. Le service UDP `echo` fonctionne également, et fut utilisé par le programme `fraggle`. Les services TCP, quant à eux, ne peuvent utilisés. Etant orienté-connection, ils ne fonctionnent pas en broadcast.

La prévention contre ce type d'attaque consiste à bloquer les messages de broadcast initiés à l'extérieur du réseau de destination. On peut également

limiter les services réseaux `ping` et `echo`

7.6 Protection contre les attaques DoS

La protection contre les attaques par déni de service contient quatre lignes de défenses :

1. La prévention et l'anticipation : Prévenir une attaque DoS est difficile, car elle requiert une action conjointe des acteurs de l'Internet. Il s'agit notamment du filtrage d'adresses IP spoofées, et le blocage des broadcasts externes. L'anticipation consiste à prévoir les DoS et à fournir des mécanismes permettant aux services de survivre pendant une tentative d'attaque. Pour cela, il faut disposer de ressources de réserve, et avoir une stratégie de backup permettant le déploiement rapide à la demande.
2. La détection et le filtrage : Lorsqu'une attaque se produit, il est important d'arriver à la détecter le plus tôt possible afin d'en minimiser l'impact. Des systèmes de détection d'intrusion peuvent effectuer cette détection, en recherchant des patterns inhabituels de trafic. Lorsqu'une attaque est détectée, les paquets concernés sont filtrés. La collaboration avec le fournisseur d'accès est importante, il faut donc pouvoir le joindre par un autre médium que le réseau durant l'attaque.
3. Le traçage et l'identification de la source : Il est important, pour éviter des attaques ultérieures, de remonter à la source de l'attaque.
4. La réaction à l'attaque : Une fois que l'attaque a eu lieu et a été contrôlée, il s'agit de restaurer le système et de réparer les dégâts qui ont été effectués.

Chapitre 8

La détection d'intrusion

L'intrusion constitue un problème de sécurité capital. Le RFC2828 [10] définit l'intrusion comme suit :

Définition

Une intrusion est un événement en sécurité, ou une combinaison de plusieurs événements qui constitue un incident au cours duquel un intrus gagne ou essaie de gagner l'accès à un système sans en avoir l'autorisation.

Des exemples d'intrusion sont par exemple le vol d'un login pour se connecter à une machine, ou encore, dans le cas d'un utilisateur enregistré, l'acquisition de privilèges non autorisés. Comme nous l'avons vu, certains logiciels, appelés malwares, peuvent égalemnet effectuer des intrusions dans des systèmes.

8.1 Les intrus

Les hackers ou crackers sont des personnes cherchant à s'introduire de manière non autorisée dans des systèmes. Ils constituent une menace importante à la sécurité. Ils peuvent agir de l'extérieur de l'organisation ciblée, aussi bien qu'à l'intérieur. Ils peuvent choisir des cibles au hasard, mais également viser des cibles spécifiques, auquel cas leur attaque sera nettement plus sophistiquées car conçue sur mesure, après investigation au sujet du système-cible. De telles attaques sont en général conçue pour passer outre les firewalls et autres défenses au niveau du périmètre de l'organisation, c'est pourquoi il faut prévoir des mécanismes de sécurité à plusieurs niveaux (défense en profondeur). Cela peut se réaliser entre autre par l'utilisation d'un système de détection d'intrusion, comme nous allons le voir.

Il est important de comprendre les motivations des attaquants afin de concevoir une stratégie de sécurité sur mesure. On peut ainsi classifier les

attaquants en quatre catégories :

- Les cyber-criminels : Leur objectif est le gain financier. Ils l'obtiennent via du vol d'indentité, d'accès à des organismes financiers, de l'espionnage, du vol de données ou des demandes de rançon sur des données.
- Les activistes : ils sont motivés par des causes politiques ou sociales. Le but de leurs attaques est de promouvoir leur cause et la rendre publique. Ils utilisent des défacements de sites web, des DoS, ou du vol et de la distribution de données pour nuire à la réputation de la cible.
- Les organisations sponsorisées par l'état : Responsables des APT (Advanced Persistent Threats), attaques d'espionnage ou de sabotage d'activités, de plus en plus répandues et s'étendant sur de longues périodes.
- Les autres : c'est-à-dire les attaquants non repris dans les trois premières catégories. On retrouve entre autres les hackers classiques, motivés par le challenge technique ou par l'estime mutuelle de la communauté.

Dans chacune de ces catégories, on retrouve différents niveaux d'expertise. On peut également les classifier :

1. Les apprentis : hackers avec un niveau technique minimal, qui utilisent de préférence des toolkits existants. Ils constituent la plus grande partie des attaquants. Ils sont relativement faciles à contrer, étant donné leur niveau et la faible complexité de leurs attaques. On les appelle également **script-kiddies**.
2. Les compagnons : hackers avec des compétences techniques suffisantes pour modifier et étendre les toolkits existantes afin d'exploiter de nouvelles vulnérabilités, ou pour adapter les attaques à la cible. Ils sont plus difficiles à contrer.
3. Les maîtres : hackers avec des niveaux techniques élevés, capables de découvrir de nouvelles catégories de failles ou d'écrire de nouvelles toolkits puissantes.

En fonction de l'objectif de l'attaque et du niveau de l'attaquant, les intrusions peuvent être bénignes à sérieuses, allant jusqu'au vol de données critiques ou à la mise hors service de systèmes complets.

Les systèmes de détection et de prévention d'intrusion sont conçus pour aider à contrer les menaces liées à ces attaques. Ils peuvent être relativemetn efficace contre des attaques relativement simples et documentées. Ils ne pourront par contre pas faire grand chose contre les attaques complexes et spécifiques lancées par des criminels ou des intrus sponsorisés par des états (APT), notamment parce que ces attaquants sont susceptibles d'exploiter des vulnérabilités **zero-day**, non connues dans la sphère publique. Les IDS et les IPS doivent donc faire partie de toute une stratégie de défense en profondeur incluant également du chiffrement, des audits, des contrôles d'authentification et d'autorisation, et une gestion active de la sécurité des applications et des OS.

8.2 Etapes d'une intrusion

Malgré leur variété, il est possible d'identifier des étapes récurrentes dans les intrusions :

1. Le choix de la cible et la collecte d'information : L'attaquant commence par choisir une cible et par récupérer l'information disponible publiquement à propos du système visé. Ces informations peuvent être techniques ou non. Les actions possibles sont l'exploration du site web, l'utilisation d'outils réseaux comme le DNS ou la base de données WHOIS, ou encore le scanning de réseaux avec nnmap. La collecte d'informations permet ainsi d'identifier des vulnérabilités potentielles, comme par exemples des services ou des plugins CMS vulnérables.

2. L'accès initial : L'attaquant peut utiliser une vulnérabilité réseau, craquer un mot de passe, utiliser un malware ou effectuer du social engineering pour accéder au système cible.

3. L'acquisition de privilèges : Une fois l'accès initial effectué, l'attaquant essaye d'obtenir un accès privilégié au système, en utilisant des vulnérabilités logicielles ou en installant un sniffer pour capturer des mots de passe.

4. La collecte d'informatoin ou l'exploitation du système : L'attaquant peut alors accéder et modifier les informations ou les ressources du systèmes, ou bien naviguer vers un autre système-cible. Cela peut aller du scan de fichiers pour retrouver une information au transfert de documents, ou encore à l'obtention de mots de passe vers d'autres machines du réseau.

5. La pérénnisation de l'accès : L'installation de backdoors ou autres malwares, ou encore l'ajout de crédences d'authentification permet à l'attaquant de maintenir l'accès pour des attaques ultérieures.

6. La dissimulation des traces : cette ultime étape permet à l'attaquant d'effacer les traces de son passage en désactivant ou en modifiant les logs, en supprimant les preuves de ses activités et en utilisant des rootkits et autres mesures dissimulatrices des fichiers et du code installé.

8.3 La détection d'intrusion

La détection d'intrusion est définie dans le RFC2828 [10] comme suit :

Définition

La détection d'intrusion est un service de sécurité qui surveille et analyse les événements système dans le but de trouver des tentatives d'accès non autorisées aux ressources du système, et de fournir des avertissements en temps réel ou presque.

Un système de détection d'intusion (IDS) comporte trois composants logiques :

1. Les senseurs : Responsables de la collecte des données. Ces données peuvent être des paquets réseaux, des fichiers de logs ou des traces d'appels systèmes. Les senseurs récoltent et transmettent ces informations à l'analyseur.
2. L'analyseur : Il reçoit les données depuis un ou plusieurs senseurs, les analyse, et indique sur cette base si une intrusion s'est produite.
3. L'interface utilisateur : Permet à l'utilisateur de voir le résultat de l'analyse, ou de contrôler le comportement du système.

Le nombre de senseurs utilisés par l'IDS et la localisation de l'analyseur permet de définir plusieurs types d'IDS :

- HIDS (Host-based IDS) : Surveille un hôte unique sur base des événements se produisant sur cet hôte, tels que les identifiants des processes et les appels systèmes qui sont effectués.
- NIDS (Network-based IDS) : Surveille le trafic réseau de segments ou d'appareils spécifiques, et analyse les protocoles réseaux, transport ou applicatifs pour identifier de l'activité suspecte.
- IDS hybridee : Combine les information de plusieurs senseurs, souvent à la fois HIDS et NIDS, en un analyseur central capable d'efficacement identifier et répondre à de l'activité intrusive.

L'objectif rempli par les IDS est, premièrement, de détecter les intrusions et d'identifier les attaquants suffisament rapidement pour l'éjecter du système avant que des dégats aient été produits. A défaut de pouvoir éviter les dégats, une détection précoce permet de limiter l'ampleur des dommages, et donc d'accélerer le temps de rétablissement du bon fonctionnement du système. Deuxièmement, un IDS efficace peut décourager les intrusions. Enfin, troisième avantage, l'utilisation d'IDS permet la collecte d'informations à propos des techniques d'intrusions, informations qui permettent de renforcer les mesures de prévention.

Le principe de base d'un IDS est que le comportement d'un intrus est différent du comportement d'un utilisateur légitime. En analysant les événements, l'IDS essaie de retrouver les symptômes d'un comportement anormal. Néanmoins, pour certains comportements, il est difficile de dire s'ils sont liés à de l'activité malicieuse ou légitime. Il y a donc risque d'erreurs, de deux types :

– Les faux positifs : Un comportement légitime est identifié comme étant une intrusion. Le problème des faux positifs est que quand ils sont trop fréquents, l'équipe d'administration a tendance à les ignorer.
– Les faux négatifs : Un comportement malicieux est considéré comme légitime. Le danger est alors évidemment qu'une intrusion passe inaperçue.

La configuration d'un IDS est donc délicate, et l'ajustement des paramètres doit être soigneusement pensé pour éviter tant les faux négatifs que les faux positifs.

8.4 Approches analytiques

Les analyseurs des IDS peuvent exploiter les données des senseurs de deux manières pour détecter les intrusions :

1. Par la recherche d'anomalies : Cette technique implique une collecte de données relative au comportement légitime des utilisateurs du système, pendant une période donnée. Une fois cette collecte effectuée, le système compare le comportement observé au comportement enregistré et détermine s'il est légitime ou malicieux.
2. Par l'application d'heuristiques ou reconnaissance de signatures : Cette autre technique utilise une base de données d'éléments permettant d'identifier une intrusion (signature), et, pour chaque comportement observé, elle vérifie s'il existe une signature correspondant à ce comportement. Si c'est le cas, une intrusion est signalée. Cette approche, contrairement à la précédente, ne permet d'identifier que les attaques connues.

8.5 Host-based IDS

Les HIDS ajoutent une couche de sécurité software spécialisée aux systèmes vulnérables ou sensibles tels que les serveurs de base de données ou les systèmes administratifs, afin de détecter et signaler les intrusions dès que possible.

L'avantage primaire des HIDS est qu'il peut détecter à la fois les attaques internes et les attaques externes, ce qui n'est pas possible avec des NIDS ou des firewalls.

Les senseurs utilisés par les HIDS peuvent être :

– Les traces d'appel système : Ces derniers fonctionnent particulièrement bien pour détecter les intrusions dans les systèmes Unix, mais sont beaucoup moins efficace en Windows, à cause de l'usage des DLLs qui empêchent de savoir précisément quel processus a effectué quel appel système.

- Les fichiers de logs : Déjà intégrés au système, ils sont simples à consulter. Néanmoins, les attaquants peuvent modifier ces fichiers pour cacher leurs actions.
- Les checksums d'intégrité des fichiers : Une autre approche possible est d'enregistrer les checksums des fichiers sensibles, et de vérifier régulièrement les éventuels changements. La difficulté est qu'il faut tenir compte des modifications légitimes à ces fichiers. Le logiciel *Tripwire* utilise cette approche.
- Les accès au registre : Sur les systèmes Windows, il est également possible de surveiller les programmes accédant au registre.

Les analyseurs utilisés dans les HIDS peuvent utiliser les deux approches mentionnées plus haut :

1. La détection d'anomalie : Utilisée principalement sous Unix, elle consiste généralement en l'étude du comportement en termes d'appels système.
2. La détection par signature : Cette approche est très largement utilisée dans les HIDS, puisqu'elle est à la base des logiciels anti-virus/anti-malwares. Elle est également intégrée dans les systèmes de mail, les proxy web, les firewalls et les NIDS.

8.6 Network-based IDS

Un NIDS surveille le trafic à des endroits spécifiques du réseau, en temps réel ou presque, afin de détecter des traces d'intrusion. Il peut analyser les paquets au niveau de la couche réseau, transport ou applicative.

Les NIDS sont souvent localisés dans le périmètre de sécurité d'une organisaiton, incorporés ou associés au firewall. Ils peuvent analyser soit la forme du trafic, soit le contenu, afin de détecter les activités suspectes. Néanmoins, avec l'usage de plus en plus répandu du chiffrement, l'accès au contenu réseau est de moins en moins possible.

Deux types de senseurs peuvent être utilisés : Des senseurs actifs, ou inline, qui sont insérés dans un segment réseau et sont traversés par l'ensemble du trafic (par exemple au niveau d'un firewall), ou bien des senseurs passifs, qui analysent une copie du trafic réseau. Cette deuxième option permet de ne pas impacter le délai de transmission des paquets.

Les senseurs peuvent être localisés à différents endroits. La figure 8.1 montre différentes options possibles.

1. Au niveau du firewall externe : Cela permet de voir les attaques de l'extérieur qui ont pu pénétrer à l'intérieur du périmètre défensif, ce qui met donc en évidence les faiblesses du firewall. L'IDS peut également surveiller les attaques à destination de la DMZ.
2. A l'extérieur du réseau : Le senseur voit passer tout le trafic non filtrer, et peut donc documenter l'ensemble des attaques (nombre et types).

3. Au niveau du backbone du réseau donnant accès aux services principaux : Le senseur surveille une large partie du trafic réseau, et a de plus une visibilité sur des activités malicieuses en provenance de l'intérieur du réseau

4. Au niveau des LAN ou des machines spécifiques : Cela permet de surveiller les intrusions sur les systèmes critiques en limitant les coûts.

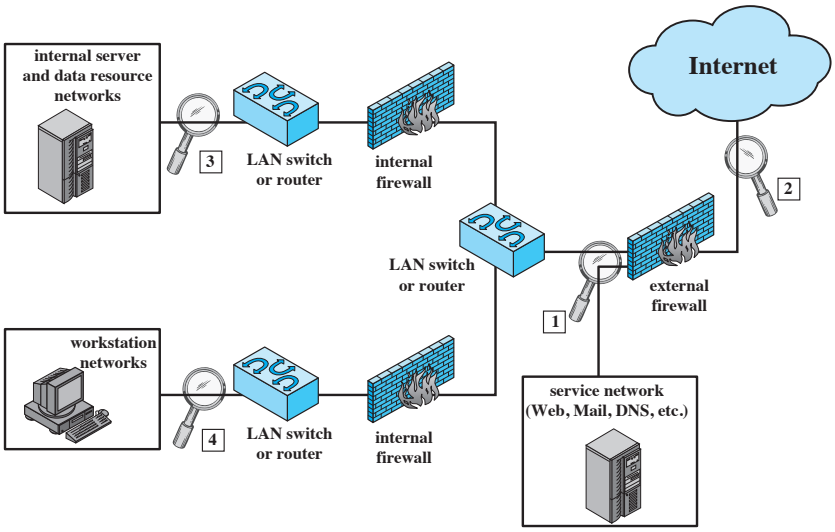

Figure 8.5 Example of NIDS Sensor Deployment

FIGURE 8.1 – Localisations possibles des senseurs (Tiré de Stallings, Computer Security, Principles and Practice [3])

Les NIDS peuvent utilisé la détection d'anomalie, consistant alors à surveiller les patterns de trafic (bande passante consommée ou type de trafic en fonction des périodes de la journée). De la sorte, ils peuvent détecter les attaques DoS, les scans réseaux ou encore les attaques de vers.

La détection par signature, quant à elle, consistera à repérer des attaques spécifiques, comme par exemple, dans le cas de la couche réseau, des attaques utilisant ICMP, ou encore des paquets avec des adresses IP spoofées. Pour la couche Transport, les scans de port ou les attaques SYN flood sont identifiables relativement facilement.

8.7 Les IDS hybrides

Les IDS hybrides, ou distribuées, consistent en un ensemble de senseurs répartis au niveau des hôtes et à divers endroits du réseau, qui transmettent les données collectées à un analyseur centralisé. Ayant une vue d'ensemble sur l'ensemble de l'organisation, l'analyseur peut alors effectuer une détection d'intrusion efficace.

8.8 Les honeypots

Un autre composant participant à la détection d'intrusion est le **honeypot**. Un honeypot est un leurre, c'est-à-dire une machine spécialement conçue pour attirer des attaques, avec pour objectifs :

1. de distraire un attaquant de l'accès aux systèmes critiques
2. de collecter de l'information à propos de l'activité de l'attaquant
3. d'encourager l'attaquant à rester sur le système suffisament longtemps pour permettre à l'administrateur de réagir

Le honeypot étant une machine sans valeur productive, toute tentative de communication ou d'accès à cette machine est considérée comme suspecte.

Il existe deux types de honeypots. Les honeypots à faible interaction sont des logiciels qui vont imiter le comportement de services réels, mais sans en être eux-même. Les honeypots à forte interactions sont par contre des services réels, instrumentés et déployés en fonction de cet objectif spécifique.

Tout comme les NIDS, les honeypots peuvent être placés à différents endroits du réseau, chaque endroit ayant ses avantages et ses inconvénients. Relevons qu'un élément de première importance à considérer est le risque encourru au cas où le honeypot tombe, puisque l'attaquant dispose alors d'une plate-forme privilégiée pour accéder au réseau.

Chapitre 9

Les Firewalls

9.1 Motivations

Alors qu'à l'origine de l'Internet, les réseaux étaient des moyens de communication fiables et les différents interlocuteurs dignes de confiance, il n'est plus possible actuellement d'exposer son réseau vers l'extérieur sans courir des risques majeurs de sécurité. Par ailleurs, l'accès internet est une ressource à présent indispensable, et aucune organisation ne peut s'en passer. L'accès internet est donc tout autant un besoin qu'une menace.

La protection individuelle des hôtes est certes incontournable, mais elle n'est néanmoins pas suffisante pour protéger l'ensemble du réseau, sans parler des complexités de mises à jour. Il faut donc également disposer d'un moyen de contrôler et réguler tout le trafic entrant dans une organisation. Ce contrôle peut être réalisé par un firewall, installé à la bordure entre le réseau et l'Internet afin de réaliser un périmètre de sécurité. En forcant tout le trafic à passer au travers de ce firewall, on dispose d'un point central pour l'implémentation de politiques de sécurité et pour la surveillance et l'audit de l'ensemble du trafic.

9.2 Caractéristiques et politique d'accès

L'objectif des firewalls est d'assurer les points suivants :

1. Tout le trafic de l'intérieur et de l'extérieur (et vice-versa) doit passer à travers le firewall, ce qui implique de bloquer physiquement tout accès au réseau autre que via ce firewall.
2. Seul le trafic autorisé par les politiques de sécurité doit être autorisé à passer.
3. Le firewall lui-même doit être sécurisé contre les intrusions. Le système qui l'abrite doit donc être correctement sécurisé.

La définition des politiques d'accès implémentées par les firewalls est donc d'une importance critique. Cette définition se fait sur base de l'analyse des risques effectuée au niveau de l'organisation, et sur base de cette analyse, on définit le type de trafic qui est nécessaire pour le bon fonctionnement de l'organisation.

La politique d'accès d'un firewall peut se baser sur de nombreux éléments différents : L'adresse IP, le protocol applicatif, l'identité de l'utilisateur, ou encore le type d'activité ou la tranche horaire considérée. Un firewall est un outil indispenable, permettant d'empêcher les intrusions dans le système et de simplifier la gestion de la sécurité en offrant un point d'observation sur l'ensemble du trafic réseau. En plus, il peut héberger des services supplémentaires comme des systèmes de prévention d'intrusion (IPS), ou encore une passerelle NAT ou un point d'entrée IPSec.

Par contre, ils ont également leurs limitations. Ils ne peuvent bien entendu pas intervenir dans le cas d'attaque utilisant d'autres points d'entrée, évitant donc le firewall lui-même. Un exemple est le cas d'une borne wifi interne mal configurée, accessible de l'extérieur. Ils ne protègent pas non plus des menaces internes à l'organisation.

9.3 Types de firewalls

Un firewall peut surveiller le trafic réseau à différents niveaux, allant du bas-niveau (IP) au haut niveau (applications). Il peut également fonctionner comme un filtre positif, c'est-à-dire n'acceptant que des paquets répondant aux règles spécifiées, ou bien comme un filtre négatif, rejetant les paquets répondants aux règles. Nous allons ici examiner quatre types de firewalls.

9.3.1 Les firewalls stateless

Un firewall dit stateless analyse chaque paquet sur base d'un ensemble de règles, qui définit si le paquet doit être transmis ou jeté. Les règles sont basées sur l'adresse IP source ou destination, les numéros de port utilisés, le champ protocole dans l'en-tête IP, ou encore l'interface réseau sur laquelle le paquet a été reçu. Lorsque le paquet ne répond à aucune règle, il est important de spécifier la politique à appliquer dans ce cas. Les deux politiques par défaut possibles sont l'acceptation ou le rejet. Bien que la première soit plus confortable pour les utilisateurs, qui ont plus de liberté dans le trafic qu'ils souhaitent envoyer, la seconde est plus conservative et sera donc en général préférée.

Rule	Direction	IP Src	IP Dst	Protocol	Dst port	Action
1	In	External	Internal	TCP	25	Permit
2	Out	Internal	External	TCP	>1023	Permit
3	Out	Internal	External	TCP	25	Permit
4	In	External	Internal	TCP	>1023	Permit
5	Either	Any	Any	r6c5	Any	Deny

L'exemple ci-dessus montre une configuration possible d'un firewall pour le trafic SMTP. Les règles sont appliquées par ordre croissant, et sont les suivantes :

1. Autoriser le mail entrant depuis une source externe
2. Permettre la réponse à une connecion SMTP entrante
3. Autoriser le mail sortant depuis une source interne
4. Permettre la réposne à une connexion SMTP sortante
5. Bloquer par défaut tout autre type de trafic

Cette configuration pose plusieurs problèmes. Premièrement, la règle 4 permet à tout trafic entrant vers un port supérieur à 1023 de pénétrer dans le réseau, ce qui laisse de nombreuses possibilités d'attaque. Il faudrait donc rajouter dans chacune de ces règles le port source autorisé, à savoir, pour cette règle 4, le port 25. Deuxièmement, tel que spécifié, rien ne garanti que les paquets venant de l'extérieur depuis un port TCP 25 viennent réellement d'un serveur SMTP. Pour s'assurer qu'un tel trafic ne vient qu'en réponse depuis une requête interne, il est possible d'utiliser les flags TCP pour n'autoriser que les paquets ayant le flag ACK mis. De la sorte, les machines extérieures sont dans l'impossibilité d'ouvrir une connexion TCP depuis un port 25.

Bien que les firewall stateless aient l'avantage de la simplicité, ils souffrent de nombreuses faiblesses. Ils ne peuvent ainsi pas exploiter les données des couches supérieures, et ne peuvent alors pas repérer les attaques employant des vulnérabilités au niveau applicatif. Les logs seront également d'intérêt limité, pour la même raison. Ensuite, en raison de leur mode de configuration, les erreurs de configuration ne sont pas rares et laissent autant de portes d'entrée pour les attaques. Enfin, les firewalls stateless sont généralement eux-aussi vulnérables à des attaques, telles que, par exemple :

- Le spoofing d'adresse IP : Un attaquant peut tenter de pénétrer le réseau en utilisant une adresse IP interne à ce dernier. Le firewall peut détecter cette attaque en vérifiant l'interface d'entrée sur laquelle le paquet est reçu.
- Les attaques par source routing : Un attaquant peut essayer d'utiliser le source routing pour bypasser les mesures de sécurité. Le firewall peut néanmoins bloquer tous les paquets utilisant cette option.
- Les attaques par petits fragments : En fragmentant le paquet IP, l'attaquant peut espérer bluffer le firewall et l'empêcher d'analyser l'en-tête

TCP en s'arrangant pour que ce dernier soit réparti sur plusieurs fragments.

9.3.2 Les firewalls stateful

Les firewalls stateless considèrent les paquets indépendamment les uns des autres, sans regarder le contexte dans lequel ils se trouvent. Un firewall stateful va permettre d'être plus spécifique, en examinant le contexte de la connexion TCP dans laquelle chaque paquet (s'il contient du TCP) se trouve. Pour cela, il va garder en mémoire des informations à propos des connexions TCP qui le traversent. Pour reprendre l'exemple du serveur SMTP, la règle 4, avec un firewall stateful, pourra spécifier que les paquets venant de l'extérieur depuis un port 25 doivent faire partie d'une connexion qui aura été initiée depuis l'intérieur.

En général, lorsqu'une application basée sur TCP crée une session avec un serveur distant, elle établit la commnunication sur le port distant du serveur, en général un port bien connu inférieur à 1024. Elle-même va utiliser comme port source un port dynamique, choisi dans l'intervalle 1024-65535. Avec un firewall stateless, le seul moyen de permettre ce genre de connexion est d'accepter le trafic entrant vers tous ces numéros de port dynamiques, ce qui ouvre la porte à de nombreuses vulnérabilités. A l'inverse, un firewall stateful va enregistrer tous les établissements de connexion depuis l'intérieur du réseau, créant ainsi un répertoire de connexions TCP autorisées. Lors de l'examen des paquets entrants, le firewall peut donc déterminer, sur base de ce répertoire, si le nouveau paquet appartient à une connexion existante ou non. Cela limite fortement les opportunités d'attaque.

9.3.3 Les gateway applicatives, ou proxies

Après les firewalls stateless, qui se basent sur les informations IP et TCP, et les firewalls stateful, qui mémorisent et exploitent en plus l'état des connexions TCP, nous allons à présent aborder les gateway applicatives, aussi dites proxies. Ces dernières servent de relai au niveau de la couche applicative.

Un proxy est donc un équipement intermédiaire qui va se positionner entre un hôte à l'intérieur du réseau protégé et son interlocuteur à l'extérieur du réseau. Au niveau des connexions TCP, il n'y a plus une seule connexion de bout en bout entre le client et le serveur, mais deux connexions, une du client au proxy, et une seconde du proxy au serveur. Le proxy va intercepter toutes les requêtes pour une application donnée, effectuer des opérations de filtrage, d'authentification ou d'optimisation, puis éventuellement relayer la requête à son destinataire initial.

Si le protocole utilisé dans la communication n'est pas implémenté ou accepté au niveau du proxy, la requête ne pourra être relayée. Cela permet

aux administrateurs réseaux d'être beaucoup plus sélectif sur le trafic accepté, puisqu'on se base sur l'application utilisée plutôt qu'uniquement sur les informations des couches réseau et transport. Un proxy permet également la surveillance et les logs du trafic au niveau applicatif.

Un inconvénient du proxy, par contre, est qu'il apporte un délai supplémentaire par le temps de traitement nécessaire aux opérations effectuées lors du relai.

Il existe différents types de proxies :

Les proxies Web/HTTP

Un proxy HTTP est placé de telle sorte qu'il intercepte toutes les requêtes Web des machines internes du réseau. Sur base de ces requêtes, il peut effectuer diverses opérations ou vérifications :

- Authentifier l'utilisateur
- Appliquer des filtres au niveau du contenu demandé (listes noires de sites web, filtrage des publicités,...)
- Vérifier si la page demandée n'est pas déjà dans la cache du proxy
- Utiliser de la compression pour accélérer la navigation
- Anonymiser la requête
- Surveiller et logger le trafic web
- ...

Un proxy peut être utilisé pour protéger les utilisateurs d'un système, de manière explicite ou non (proxy transparent, par ex. au niveau de l'ISP). Il peut également être utilisé pour contourner des politiques de sécurité (par exemple, dans certains pays qui bloquent des sites sur base de l'IP, il est possible d'utiliser des proxies pour y accéder malgré tout, ou à l'inverse, consulter du contenu limité à certains pays), ou pour anonymiser les requêtes effectuées, ce qui permet d'éviter la traçabilité.

Les reverse proxies Web

Alors qu'un proxy vise à protéger un client Web, un reverse proxy sera placé de telle sorte qu'il protège le serveur. Un reverse proxy joue donc le rôle de rempart, pour éviter les accès directs aux serveurs web de l'organisation. Ce rempart, par divers mécanismes permet à la fois d'optimiser l'accès aux ressources HTTP, mais également de protéger l'intérieur du réseau. Le reverse proxy sera ainsi typiquement plaçé dans une DMZ, avec contrôle strict des connexions entre lui-même et le ou les serveurs web internes.

Le reverse proxy peut effectuer diverses opérations lors du relai :

- Décharger le serveur Web en implémentant un système de cache (accélérateur Web/HTTP)
- Décharger le serveur Web en servant de terminateur SSL (déchiffrement des connexions)

- Sécuriser l'architecture des serveurs en centralisant toutes les connexions et en cachant l'architecture interne des services (par ex : réécriture d'URL)
- Optimiser les performances en réalisant de la compression
- Optimiser les performances en réalisant de la répartition de charge entre les serveurs Web internes.

Un exemple d'utilisation d'un reverse proxy est la montée en charge d'un serveur Apache fournissant à la fois du contenu statique (images) et du contenu dynamique. Pour décharger le serveur, il est possible de déléguer la tâche du contenu statique à une autre machine, avec le reverse proxy qui se charge de répartir les requêtes.

Les proxies mail

Au niveau des services mail, les proxies SMTP sont typiquement utilisés pour détecter et bloquer les attaques de spamming, aussi bien de l'intérieur que de l'extérieur.

Les proxies FTP

Le principe du proxy peut également être appliqué au protocole FTP, de même que le principe du reverse proxy. Cela permet ici aussi d'effectuer du filtrage sur base de la source ou de la destination, de faire de l'authentification, ou bien encore, dans le cas du reverse proxy, de limiter les commandes FTP autorisées.

Les proxies DNS

Dans le cas du DNS, un proxy permet de filtrer les requêtes DNS qui ne sont pas correctement formées, ou qui risquent de causer des problèmes de sécurité au niveau du serveur DNS.

9.3.4 Les passerelles au niveau circuit

Une variante du proxy consiste à se placer au milieu de la connexion TCP entre un client et un serveur, créant donc deux connexions séparées au lieu d'une, puis à se contenter de transférer les segments TCP sans examiner le contenu applicatif. Les fonctionnalités de sécurité consistent alors à filtrer les connexions au niveau Transport. Il s'agit somme toute d'un proxy TCP. Il est typiquement utilisé dans des environnements où les utilisateurs sont fiables, et peuvent donc bénéficier de liberté au niveau des connexions sortantes, ce qui permet d'alléger la charge d'analyse des segments, tandis que les connexions entrantes sont filtrées par un proxy classique.

9.4 Types d'installation d'un firewall

Il existe plusieurs possibilités d'implémentation de la fonction de firewall :

9.4.1 Le Bastion

Un Bastion, en sécurité, est un système identifié par l'administrateur comme étant un point de passage critique dans le réseau. Typiquement, c'est le point d'entrée et de sortie vers l'Internet. Un bastion sert généralement à héberger les proxies ou les passerelles niveau circuit. Ses caractéristiques sont les suivantes :

- La machine est entièrement sûre (OS sécurisée, applications à jour, ...), et les applications ont été limitées au strict nécessaire (proxies DNS, HTTP, etc.). Ces dernières sont de plus des versions légères des services proposés (une application de proxy mail est typiquement beaucoup plus légère et plus sécurisée qu'une application mail complète)
- Il peut fournir un mécanisme d'authentification avant l'accès aux services (proxies)
- Les proxies sont configurés pour fournir uniquement certains services pour chaque application, ou uniquement pour certains hôtes
- Les proxies loggent les informations à propos du trafic
- Les proxies sont indépendant les uns des autres, et peuvent être gérés de manière modulaire
- Les applications de proxy n'utilisent généralement pas le disque, ce qui permet de configurer la partie du système de fichiers contenant les exécutables en read-only, ce qui empêche l'installation de sniffers ou d'autres chevaux de troie.

9.4.2 Le firewall sur un hôte

Les firewall dédiés aux hôtes sont des modules logiciels permettant de sécuriser des machines individuelles. Un tel firewall permet de :

- Ajuster les règles de firewall à l'environnement de l'hôte (ex : un serveur)
- Fournir une protection indépendamment de la topologie et de l'emplacement de l'hôte dans le réseau

Utilisés en conjonction avec des firewalls stand-alone, les firewalls-hôtes fournissent une couche supplémentaire de protection. Ils peuvent soit protéger un serveur dédié à un service particulier, soit offrir une protection à un terminal utilisateur.

Dans le cas d'un serveur, le firewall sera configuré en fonction du service spécifique. Dans le cas d'un firewall utilisateur, le rôle du firewall sera plus généraliste, et devra d'une part d'empêcher les accès extérieurs à la machine, et d'autre part, filtrer et détecter le trafic éventuellement émis par un ver ou

tout autre malware. Par défaut, tout trafic entrant sera bloqué, et l'utilisateur devra sélectivement ouvrir les ports dont il souhaite explicitement autoriser l'accès depuis l'extérieur (ex : accès SSH,partage de fichiers, NTP, VNC, ...).

9.5 Localisation et configuration du firewall

Un firewall est destiné à faire barrière entre une source de trafic extérieure et un réseau interne. Cela laisse plusieurs possibilités de choix à l'administrateur pour le placement d'un ou plusieurs firewalls. Nous allons examiner plusieurs combinaisons possibles.

9.5.1 Le réseau DMZ

La figure 9.1 illustre une configuration réseau classique utilisant deux firewalls, un externe, à la frontière entre l'organisation et l'extérieur, et un interne, qui sépare le coeur de l'entreprise des systèmes qui doivent être accessibles de l'extérieur mais nécessitent néanmoins une protection. La zone entre les deux firewalls est appelée DMZ. Typiquement, la DMZ héberge les serveurs web, email ou encore DNS.

Le firewall externe protège donc l'accès aux systèmes en DMZ tout en leur garantissant la connectivité extérieure, et fournit aussi un premier niveau de protection pour le reste du réseau. Les firewalls internes auront trois objectifs :

1. Ils fournissent un filtrage plus restrictif que le firewall externe pour protéger les serveurs et les postes de travail des attaques extérieures.
2. Le firewall interne fournit une protection au réseau interne contre les attaques lancées depuis les systèmes DMZ (vers, rootkit et autres malwares), et inversément, il protège les systèmes DMZ des attaques lancées depuis le réseau interne.
3. Des firewalls internes peuvent aussi être utilisés pour protéger des portions du réseau interne les uns des autres.

9.5.2 Les VPNs

Dans l'environnement informatique distribué que nous connaissons aujourd'hui, les VPNs (réseaux privés virtuels) permettent d'interconnecter des machines à travers des réseaux non sécurisés (typiquement Internet), en utilisant des protocoles de sécurisation et une couche de chiffrement. Cette sécurisation est essentielle pour éviter l'espionnage des données critiques de l'entreprise ainsi que pour protéger l'accès au réseau de cette dernière à des personnes non autorisées.

Un VPN est typiquement utilisé dans deux scénarii :

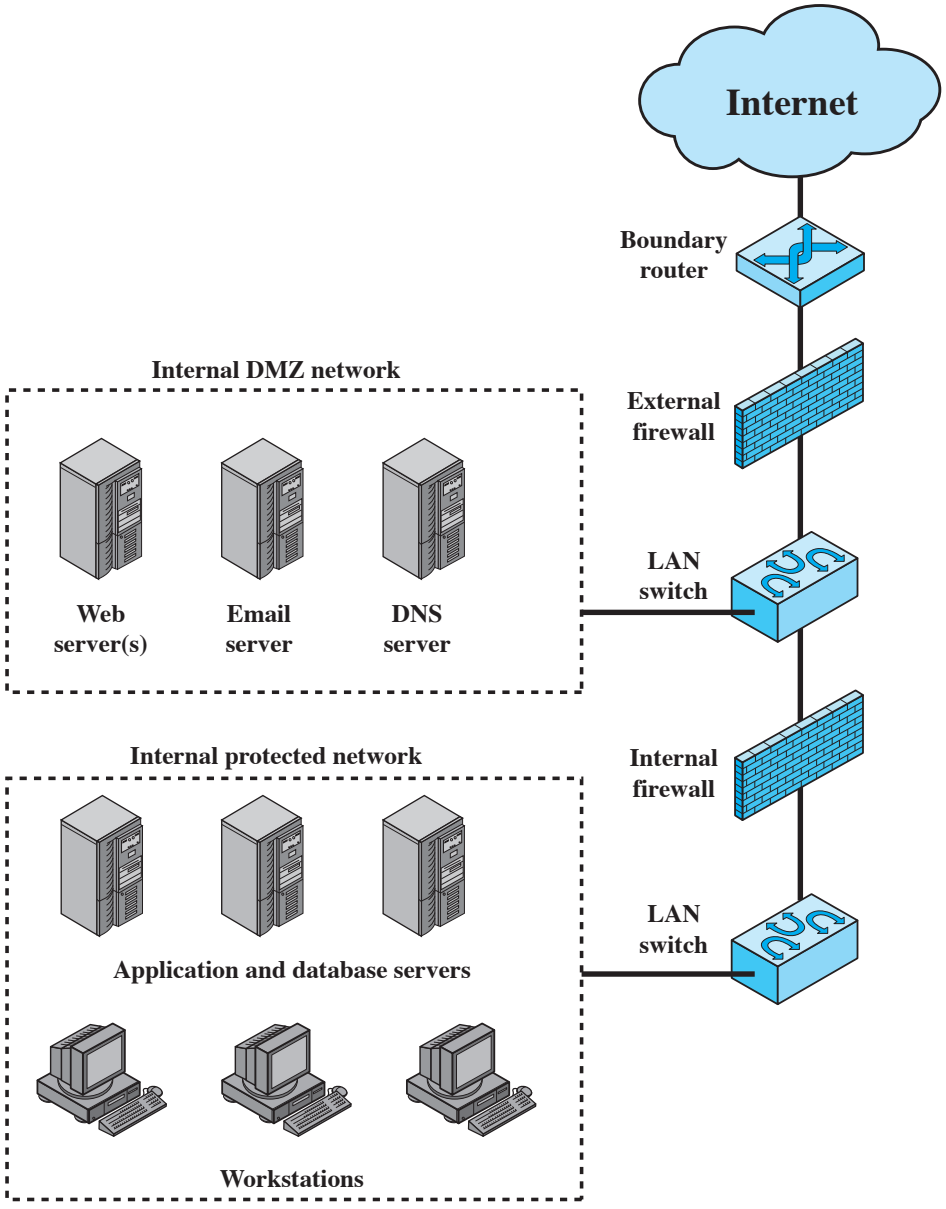

FIGURE 9.1 – Exemple de configuration de firewall (Tiré de Stallings, Computer Security, Principles and Practice [3])

- D'une part, pour interconnecter deux sites distants d'une même entreprise, sans devoir supporter les coûts élevés d'une ligne dédiée.
- D'autre part, pour permettre à des utilisateurs nomades d'accéder de manière sécurisée au réseau d'entreprise depuis divers endroits (domicile, client, conférence, ...).

Un VPN utilise des mécanismes de chiffrement et d'authentification dans les protocoles des couches inférieures. Ces mécanismes sont typiquement implémentés au niveau d'un routeur ou d'un firewall. Le plus commun, à savoir IPSec, fournit une sécurisation au niveau de la couche IP.

La figure 9.2 montre une organisation maintenant plusieurs LAN à différents endroits. Chaque LAN utilise de l'IP classique, mais IPSec est utilisé pour le traffic inter-site à travers Internet. Au niveau de chaque LAN, un Firewall sert de terminaison IPSec : il encapsule et décapsule le trafic IPSec pour permettre aux terminaux de communiquer en IP pur à l'intérieur des LAN. Un utilisateur nomade est également représenté en haut à gauche. Dans ce cas, le chiffrement IPSec est effectué au niveau du terminal par un logiciel ad-hoc.

Remarquons qu'au niveau des terminaisons IPSec des LANs, l'implémentation à l'intérieur du firewall n'est qu'un choix parmi d'autres. Il est également possible de faire le chiffrement au niveau d'un appareil séparé du firewall. Deux cas de figure sont alors possibles :

- L'appareil est en amont du firewall, à l'extérieur : Il n'est alors pas sécurisé par le firewall, et est donc vulnérable.
- L'appareil est en aval du firewall, à l'intérieur du LAN : Le trafic chiffré ne peut alors pas être analysé par le firewall, qui ne peut donc ni filtrer ni réaliser du contrôle d'accès sur ce trafic.

9.5.3 Les firewalls distribués

Les différents types de firewall peuvent être combinés entre eux pour former un système à firewalls distribués. On utilise aussi bien des firewall stand-alone que des firewalls-hôtes, tous contrôlés par la même entité administrative. Des outils adaptés permettent aux administrateurs du réseau de configurer les politiques de manière centralisée. Un aspect particulièrement critique de ce genre de système est la surveillance : un soin particulier doit être apporté à la collecte et à la consultation des logs de manière à surveiller le réseau dans son entièreté aussi bien qu'au niveau de chaque service spécifique.

La figure 9.3 montre un tel exemple de configuration :

- Une DMZ extérieure, isolée du réseau d'entreprise, est protégée par l'utilisation de firewalls-hôtes. En l'occurence, cette DMZ héberge les serveurs Web, ne contenant pas de données critiques, et les firewalls sont donc configurés spécifiquement pour ce service.

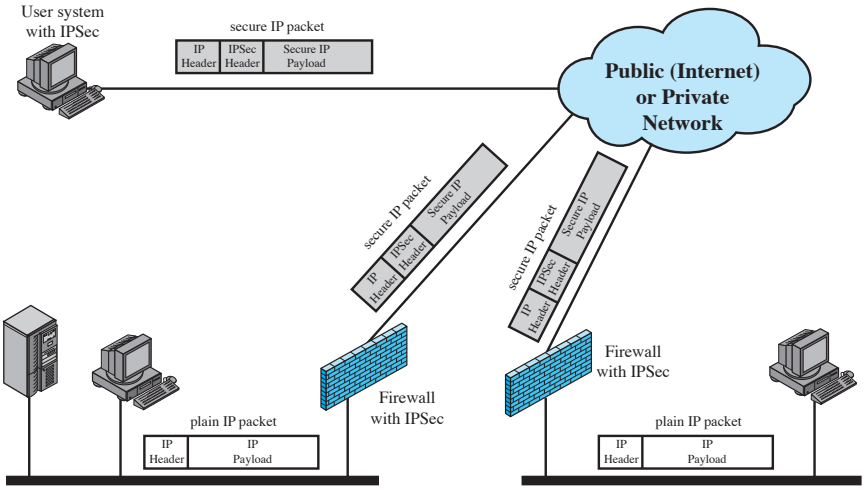

Figure 9.4 A VPN Security Scenario

FIGURE 9.2 – Un scénario d'utilisation d'un VPN (Tiré de Stallings, Computer Security, Principles and Practice [3])

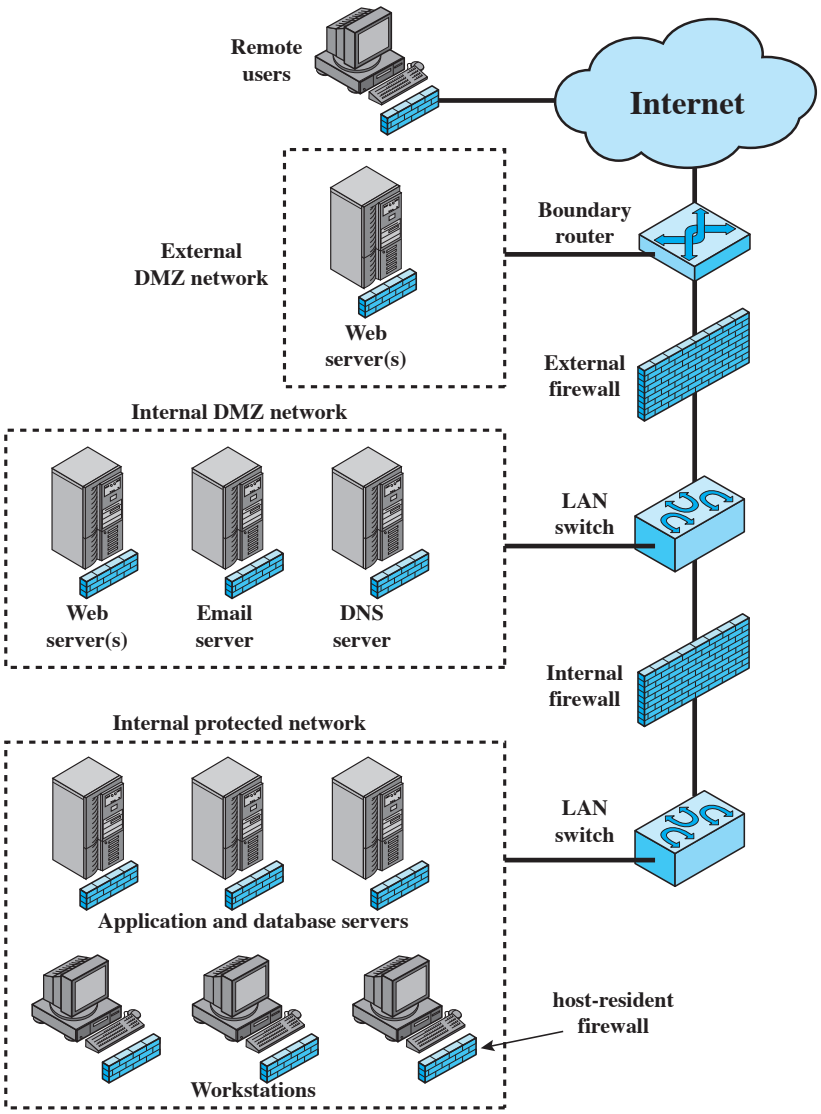

Figure 9.5 Example Distributed Firewall Configuration

FIGURE 9.3 – Système de firewalls distribués (Tiré de Stallings, Computer Security, Principles and Practice[3])

- Un firewall extérieur protège l'ensemble du réseau en aval de la DMZ externe. Il s'agit d'un firewall stand-alone.
- Une seconde DMZ, interne donc, contient les serveurs qui doivent communiquer avec l'extérieur, mais de manière très contrôlée. En plus d'être protégés par le firewall externe, ils disposent chacun d'un firewall intégré adapté à leur service propre.
- Un Firewall interne isole la partie la plus vulnérable du réseau de la DMZ interne. Il s'agit du réseau contenant les terminaux des employés ainsi que les serveurs applicatifs et de base de données, contenant donc les informations critiques. Là encore, chaque machine est protégée par un firewall-hôte.

9.5.4 Synthèse

Nous allons à présent regrouper les différentes topologies et localisations de firewall abordés dans les précédents paragraphes. Nous avons donc identifié les alternatives suivantes :

- Les firewall-hôtes : Il s'agit des logiciels de firewall personnels et des firewalls spécifiques aux serveurs.
- Les routeurs-firewall : Il s'agit de simples routeurs placés entre un réseau interne et un réseau externe, avec des capacités de filtrage avec ou sans état. Ils sont typiquement utilisés dans un cadre domestique ou pour des TPMEs.
- Bastion en ligne simple : c'est un firewall placé entre un routeur interne et un routeur externe. Il capture tout le trafic entre ces deux routeurs. Il peut implémenter du filtrage statefull ou des proxies applicatifs. Ce système sera utilisé pour des PMEs.
- Bastion simple en T : A la différence du bastion simple, il utilise trois interfaces, et peut donc contrôler l'accès à un troisième réseau de type DMZ où les serveurs visibles de l'extérieur seront placés.
- Deux bastions en ligne : Cette topologie se retrouve à la figure 9.1. Deux firewalls bastions encadrent la DMZ, permettant d'une part de la protéger de l'extérieur, et d'autre part, de protéger le réseau interne d'une éventuelle compromission de la DMZ. Par rapport au bastion simple en ligne, nous avons à présent une redondance physique au niveau du firewall, ce qui rajoute donc une ligne de défense supplémentaire. Cette configuration est utilisée dans des réseaux de grande taille.
- Deux bastions en T : Au lieu de prendre la DMZ en sandwich, le firewall externe la connecte sur une interface séparée de celle le connectant au firewall interne.
- Firewalls distribués : Il s'agit de configurations telles que celle de la figure 9.3.

9.6 Système de prévention d'intrusion

En plus de système de firewalling ou d'IDS, on peut également mettre en place des systèmes de prévention d'intrusion, ou IPS. Par rapport à un IDS qui fournit une détection de comportements suspects sur base de signatures, un IPS va permettre de réagir à une tentative d'intrusion supposée en bloquant l'activité présumée malicieuse.

Les IPS peuvent être de deux types :

1. HIPS (Host-Based IPS) : Installé sur une machine spécifique, un HIPS va se concentrer sur le traffic généré par les applications et sur les séquences d'appels systèmes pour tenter d'identifier l'activité d'un éventuel malware.
2. NIPS (Network-Based IPS) : Il s'agira ici d'un appareil dédié qui a la capacité de modifier ou de jeter des paquets suspects, coupant alors les connections TCP existantes.

Bibliographie

[1] National Institute of Standards and Technology. An introduction to computer security : The nist handbook, october 1995. Special publication 800-12.

[2] Renaud Dumont. INFO0045-2 : Cryptographie et sécurité informatique. http ://www.montefiore.ulg.ac.be/ dumont/pdf/crypto.pdf, 2009-2010.

[3] William Stallings and Lawrie Brown. *Computer Security : Principles and Practice*. Pearson, 3rd edition, 2014.

[4] R. Shirey. Internet Security Glossary, Version 2. RFC 4949 (Informational), August 2007.

[5] National Centers of Academic Excellence in Information Assurance/Cyber Defense. Ncae ia/cd knowledge units, 2013.

[6] Wendy Brevière. Syllabus de sécurité des réseaux informatiques. Ephec, Technologie de l'informatique.

[7] Philippe Oechslin. Making a faster cryptanalytic time-memory trade-off. In Dan Boneh, editor, *Advances in Cryptology - CRYPTO 2003*, volume 2729 of *Lecture Notes in Computer Science*, pages 617–630. Springer Berlin Heidelberg, 2003.

[8] Authorization Authentication and Accounting Working Group.

[9] R Ben-Natan. Data security, governance privacy : Protecting the core of your business. Guardium White Paper, www.guardium.com, 2006.

[10] R. Shirey. RFC 2828 - internet security glossary. http ://www.ietf.org/rfc/rfc2828.txt, May 2000.