ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL

Facultad de Ingeniería de Electricidad y Computación

"DESARROLLO DEL CONJUNTO DE APLICACIONES PARA DETECCIÓN DE ATAQUES EN REDES DE DATOS IPv4/IPv6 UTILIZANDO PYTHON"

INFORME DEL PROYECTO DE GRADUACIÓN

Previa la obtención del Título de:

INGENIERO EN TELEMÁTICA

Presentado por:

Melgar Jara Erick Santiago

GUAYAQUIL – ECUADOR

AÑO 2015

AGRADECIMIENTO

A mis padres por su constante apoyo para culminar esta meta.

A mi director del proyecto de graduación Ing. José Patiño por su ayuda en la revisión del informe del proyecto de graduación.

A la empresa Sudamericana de Software por su apoyo y brindarme la facilidad para culminar el proyecto.

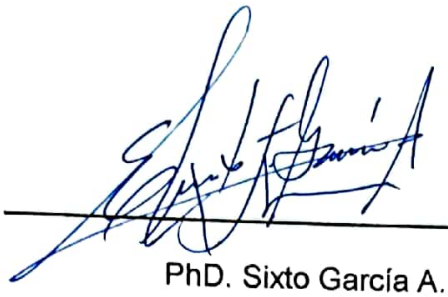
DEDICATORIA

A Dios.

A mis padres y a mis hermanos que fueron quienes han sido mi apoyo y me han sabido aconsejar en los momentos necesarios.

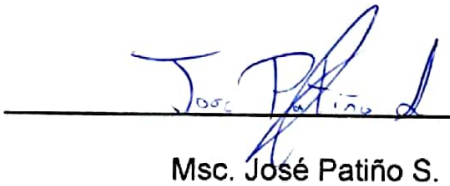
Erick Melgar Jara

TRIBUNAL DE SUSTENTACIÓN



PhD. Sixto García A.

SUBDECANO SUBROGANTE DE LA FIEC



Msc. José Patiño S.

DIRECTOR DEL PROYECTO DE GRADUACIÓN

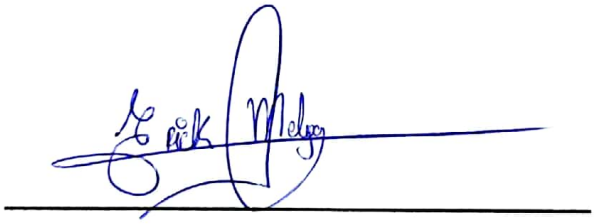
Msc. Albert Espinal S.

MIEMBRO DEL TRIBUNAL

DECLARACIÓN EXPRESA

"La responsabilidad del contenido de este Informe, nos corresponde exclusivamente; y el patrimonio intelectual del mismo a la ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL".

(Reglamento de exámenes y títulos profesionales de la ESPOL)



Erick Santiago Melgar Jara

RESUMEN

El complejo mundo de la seguridad informática presenta novedades día a día y requiere que el auditor de red se encuentre en constante crecimiento profesional. Los auditores de seguridad suelen disponer de un kit de herramientas o aplicaciones para ejercer sus proyectos.

El siguiente trabajo comprende en realizar una herramienta para dichos auditores de red los cuales necesitan detectar los múltiples ataques producidos por personas ajenas que desean perjudicar a los usuarios obteniendo su información o manipularla para otros fines.

El usuario final al utilizar dicha herramienta podrá lograr detectar ataques comunes que se encuentran en las redes IPv4/IPv6 como por ejemplo “Man In The Middle” o la Denegación de Servicios.

ÍNDICE GENERAL

ABREVIATURAS Y SIMBOLOGÍA

ARP	Address Resolution Protocol
DNS	Domain Name Server
DoS	Deny Of Service
EUI	Extended Unique Identifier
FTP	File Transfer Protocol
IANA	Internet Assigned Numbers Authority
IP	Internet Protocol (Protocolo de Internet).
ISP	Internet Service Provider
LAN	Local Area Network
MAC	Media Access Control
MITM	Man In The Middle
NDP	Neighbor Discovery Protocol
RA	Router Advertisement
SLAAC	Stateless Address Auto configuration
TCP	Transport Control Protocol
UDP	User Datagram Protocol
VoIP	Voice Over IP

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

INTRODUCCIÓN

Las amenazas contra la seguridad basadas en la red han provocado robos de identidad y fraude financiero generalizados. El correo no deseado, los virus y el spyware causan graves problemas a empresas y consumidores.

Para el desarrollo del conjunto de aplicaciones se utilizó el lenguaje de programación llamado Python debido a su práctico uso de librería para captura de paquetes de red, por tal motivo este lenguaje es el preferido para las personas encargadas de la seguridad de la red.

Este proyecto apunta a obtener la experiencia técnica para defender las redes de las instituciones a las que nos encontremos evitando así problemas de seguridad de las mismas a futuro.

Finalmente se usó la mejor metodología para que el proyecto sea eficiente al momento de utilizarlo, se estudió el caso de requerimientos para ejecutarlo y la ilustración de la información del ataque para que el usuario sepa lo que está ocurriendo.

CAPÍTULO 1

GENERALIDADES

1.1. Descripción

En este proyecto, lo que haremos será crear una aplicación, desarrollada con el lenguaje de programación Python; al momento de activar dicha aplicación, esta será capaz de detectar si estamos siendo atacados en nuestra red, en cualquiera de los 5 ataques que hemos escogido, estos son: MITM en IPv6 con Neighbor Advertisement Spoofing, MITM en IPv4 con ARP Spoofing, MITM en IPv6 con SLAAC, MITM con Invalid MAC Spoofing y DNS Hijacking.

1.2. Antecedentes

Las amenazas contra la seguridad basadas en la red han provocado robos de identidad y fraude financiero generalizados. El correo no deseado, los virus y el spyware causan graves problemas a empresas y consumidores. Una infracción de seguridad puede causar un daño irreparable a la reputación o la imagen de marca de una compañía. En los Estados Unidos, los problemas con la seguridad en Internet amenazan con retrasar la adopción de un sistema de expedientes médicos electrónicos a nivel nacional. En la Unión Europea, la confianza de los consumidores con respecto a la seguridad en Internet y la protección de datos suponen una barrera a la expansión más rápida del comercio electrónico entre los estados miembros.

Los ataques actuales contra la seguridad de la información son un negocio rentable y a menudo están controlados por los sindicatos del crimen organizado. Un creciente número de sofisticados modelos comerciales de ciberdelincuencia, incluido el auge de empresas delictivas, se basa en la venta de herramientas y servicios para lanzar ataques contra la red, más que en la venta de la información obtenida con los ataques.

Las tecnologías referentes a la seguridad de la información en Internet siguen progresando, y está pasando de tener un enfoque pasivo y puntual basado en productos, a tener planteamientos activos de punta a punta basados en reconocimiento, contención y cuarentena. Además, los proveedores de servicios de Internet (ISP) están compitiendo en seguridad y los ISP dirigidos al consumidor ofrecen seguridad de Internet dentro de su paquete de servicios.

Los legisladores de todo el mundo están centrados en el estado de la infraestructura de la información. Los legisladores quieren asegurarse de que los usuarios de las redes emplean la mejor tecnología y prácticas de procesos para hacer que sus redes sean lo más seguras posibles. Los gobiernos y las empresas actualizan de forma continua sus estrategias de prevención de ataques, y se han formado asociaciones entre órganos públicos y privados con el fin de desarrollar planteamientos de seguridad voluntarios basados en el mercado. [13]

1.3. Objetivos

Objetivos Generales

- Mejorar los niveles de seguridad en redes IPv4/IPv6, detectando los ataques más comunes dentro de una red utilizando el lenguaje de programación Python.

Objetivos Específicos

- Diseñar e implementar una aplicación en Python, para la prevención del ataque Man In The Middle en IPv6 con Neighbor Advertisement Spoofing.
- Diseñar e implementar una aplicación en Python, para la prevención del ataque Man In The Middle en IPv4 con ARP Spoofing.
- Diseñar e implementar una aplicación en Python, para la prevención del ataque Man In The Middle en Ipv6 con SLAAC.
- Diseñar e implementar una aplicación en Python, para la prevención del ataque DoS en IPv4 con Invalid MAC Spoofing.
- Diseñar e implementar una aplicación en Python, para la prevención del ataque DNS Hijacking.

1.4. Justificación

El alcance de la frase "Seguridad Informática" se ha expandido de manera importante en la última década. El término ahora se extiende más allá de proteger los secretos de corporaciones grandes y gubernamentales para incluir al consumidor promedio. Las amenazas que evolucionan rápidamente están desafiando las prioridades y los procesos que usamos para proteger nuestras empresas. Cada día nuevas herramientas, técnicas, métodos, secuencias de comandos y malware desarrollados por hackers llegan al mundo con ferocidad siempre creciente. Por lo cual usualmente se utiliza la frase "Piense como el malvado" para lo cual se necesita estudiar cómo funcionan los ataques para poder desarrollar soluciones para los mismos.

Este proyecto apunta a obtener la experiencia técnica para defender las redes de las instituciones a las que nos encontremos evitando así problemas de seguridad de las mismas a futuro. Además se utilizara el lenguaje de programación llamado "Python" lo cual es uno de los más utilizados para el desarrollo de las aplicaciones contra seguridades de red.

Además se implementara un conjunto de aplicaciones que permitirá la detección de diversos ataques que se realizan a las redes IPv4 e IPv6 para poder evitar la infiltración de usuarios ajenos a nuestra red que tenga de propósito perjudicar la información que se desea enviar a través de la misma.

1.5. Metodología

La metodología se llevará a cabo con el desarrollo de las siguientes fases:

Descripción de la fase 1

Realizar una investigación de IPv4, características que nos ofrece el uso de IPv4; sin dejar de lado los problemas inminentes, vulnerabilidades a los que está sometido, sus niveles de seguridad, así como el acelerado agotamiento de direcciones, los problemas a nivel de arquitectura, entre otras limitaciones, que fueron factores motivantes para el nacimiento de IPv6.

Descripción de la fase 2

Realizar una investigación sobre IPv6, características de seguridad, así como detalles en su funcionamiento, para determinar aquellos puntos

débiles a los cuales está sometido debido a algunos aspectos como la transición a IPv6 y que no brindan una garantía en su seguridad.

Descripción de la fase 3

Realizar un estudio exhaustivo acerca del lenguaje de programación llamado Python, sus principales características, manejabilidad, esquemas, dimensiones, como sus comandos, sintaxis, funciones, entre otras. Así como sus diversos usos y alcances a nivel informático. Haciendo énfasis en la creación de aplicaciones para seguridad informática.

Descripción de la fase 4

Una vez finalizada la etapa anterior, se procederá a la investigación acerca de los ataques más comunes en redes basadas en IPv4/IPv6. Analizando íntimamente paso a paso como cada uno de ellos se logra infiltrar en la red; así como el alcance perjudicial que puede tener el ataque. Investigar los parámetros de vulnerabilidad que aprovecha cada tipo de ataque, para poder realizar una detección a tiempo. Al final se escogerán cinco de los ataques más comunes que se presentan en una red basada en IPv4 e IPv6.

Descripción de la fase 5

Una vez entendido el funcionamiento de los diferentes tipos de ataques en redes de datos IPv4/IPv6 anteriormente mencionados. Se continuará con el diseño de, la arquitectura de la aplicación, de su interfaz gráfica, de su operatividad; para un ataque específico. Y repitiendo este proceso para los cinco ataques que vamos a tratar en esta investigación. Almacenando todas estas aplicaciones en una solo suite.

Descripción de la fase 6

Implementar la suite, que contiene cada una de las aplicaciones para la detección de los ataques escogidos previamente.

Descripción de la fase 7

En base a lo realizado, procederemos a evaluar cada una de las aplicaciones, ventajas, desventajas y limitaciones; y en base a esta serie de parámetros poder elaborar nuestras conclusiones.

CAPÍTULO 2

MARCO TEÓRICO

2.1. Historia de las redes de datos

En la década de los 50´s ya teníamos computadores que funcionaban con circuitos integrados, que condensaban millones de transistores semiconductores, lo que nos daba la facilidad de procesar cantidades mucho más grades de información de la que se estaba acostumbrado, pero el problema radicaba en que si se necesitaba procesar esa información tenía que ser llevaba a otro departamento; y se dieron cuenta que no había una manera eficaz de compartir datos entre varios

computadores. Si se modificaba un archivo había que generar un documento nuevo y compartirlo entre los usuarios. Se creaban copias innecesarias de los archivos. O si dos personas modificaban un archivo y lo compartían se perdían una de las modificaciones que habían hecho. Estos problemas llevo al uso de la tecnología llamada "networking", y con ello aumentaban la productividad y ahorraban en gastos; gracias a esto las redes se expandían cada vez más con la misma rapidez con la que lanzaban nuevas mejoras en tecnologías y productos de red.

A medida que pasaba el tiempo los computadores siguieron evolucionando, salieron los microcomputadores. Pero a medida que crecía el número de interesados, el sistema no podía soportar la demanda. A su vez las compañías informáticas encargadas del hardware y software para redes, creaban sus propios equipos, con sus propios protocolos y estándares de comunicación. Lo que hacía que muchas de las tecnologías no fueran compatibles entre sí. Se buscó soluciones y una de ellas fue la creación de estándares en una red de área local (LAN). Con esto se lograba compatibilizar los equipos provenientes de diferentes compañías, y daba estabilidad en la implementación de las LAN´s. Como las redes LAN´s iban creciendo se necesitó crear redes muchos más grandes dando

paso a las redes de área metropolitana (MAN) y redes de área amplia (WAN).

2.2. IPv4

2.2.1. Características de IPv4

Cada vez que necesitamos diseñar una arquitectura de red, nos basamos en el modelo internacional OSI (Open System Interconnection), este modelo está conformado por capas, las cuales son las siguientes:

Figura # 2.1. Capas del Modelo OSI

El protocolo de control de transmisión, es la base y da inicio a los protocolos de internet; a su vez, el protocolo de internet (IP) es un protocolo de capa de red, es decir es un protocolo de Capa 3 del

modelo OSI que fue desarrollado en el año de 1981 en redes interconectadas por conmutación de paquetes.

El objetivo del protocolo de internet es el de realizar la entrega de paquetes, aunque no todos los paquetes llegan en el mismo orden o ni siquiera llegan; y el de la fragmentación y armado de estos paquetes; y llevarlos de un nodo en una red hacia su nodo destino, para lograr esto, a cada paquete se lo identifica con una dirección IP.

La manera por la cual se designa una dirección IP, tiene un formato definido y componentes específicos.

En la actualidad tenemos dos estándares para las direcciones IP que son IPv4 e IPv6; aunque se trata de mudarnos completamente a IPv6, siempre se halla la manera de alargar la vida de IPv4, tanto así que es muy común usar aun redes basadas en IPv4.

Como lo habíamos mencionado, cada paquete que viaja a través de la red, dentro de si hay un lugar en donde se encuentra la dirección IP de origen y la dirección IP de destino. Y cabe recalcar que la dirección IP no es más que una dirección lógica que posee 32 bits

divididos en 4 grupos de 8 bits llamados octetos, con esa cantidad de bits, podemos sacar la dirección IP que identifica a la red, así como la dirección del host; cada grupo de 8 bits se encuentra separado por un punto; y al ser de 8 bits, significa que en formato decimal, cada casilla puede tomar el valor entre 0 y 255, es decir hay una posibilidad de 256 combinaciones.

Figura # 2.2. Ejemplo de una dirección de red IPv4

Una forma de clasificar las direcciones IPv4 son por sus diferentes clases; es decir, las direcciones pueden ser de tipo: A, B, C, D y E. Siendo una red de tipo A, generalmente está destinada a los diferentes gobiernos, sin embargo se la ha usado también en compañías transnacionales. El rango de direcciones de tipo B, en cambio, va destinado para medianas empresas. Siendo con ese

orden descendente tenemos que los usuarios o clientes normales, tienen asignado el rango de direcciones de tipo C. Las de tipo D son utilizadas para lo que es multicast. Y por último las direcciones tipo E se reservan solamente para fines investigativos.

Clase	Rango	N° de Redes	N° de Host	Máscara de Red	Broadcast ID
A	0.0.0.0 - 127.255.255.255	128	16.777.214	255.0.0.0	x.255.255.255
B	128.0.0.0 - 191.255.255.255	16.384	65.534	255.255.0.0	x.x.255.255
C	192.0.0.0 - 223.255.255.255	2.097.152	254	255.255.255.0	x.x.x.255
D	224.0.0.0 - 239.255.255.255				
E	240.0.0.0 - 255.255.255.255				

Figura # 2.3. Clases IPv4

2.2.2. Diferentes problemas con IPv4

Aunque IPv4 ha sido muy utilizada a nivel mundial, presenta ciertos problemas, ciertas desventajas que hace que poco a poco pensemos definitivamente en cambiarnos a IPv6.

Uno de los problemas que presenta las direcciones IP en su cuarta versión son:

Al tener una combinación de 4 octetos, 32 bits en total, nos da un número limitado de direcciones lógicas, que cuando llegue a ese

límite, como está sucediendo ahora; no se podrá asignarles direcciones IP a más equipos.

2.2.3. Transición del protocolo de IPv4 a IPv6

Cuando a inicios del año 2011, la IANA (Internet Assigned Numbers Authority, que es la entidad que se encarga de asignar las direcciones IP a nivel mundial) entregó el ultimo lote de direcciones IPv4 a APNIC (Asia-Pacific Network Information Centre, organización sin fines de lucro que se encarga del registro regional de Internet, específicamente abarca la región de Asia Pacific), se quedaron sin direcciones IPv4 disponibles. Así como APNIC que se encarga de una región completan el grupo 4 organizaciones más, lo que hace de esto 5 entidades que al ver este problema, necesitaban recurrir a las direcciones de reserva que cada una debería tener, para así poder solucionar este problema de agotamiento de direcciones.

Pero como pudimos llegar a esto, pues la respuesta es muy sencilla, la rapidez con la que aumenta el desarrollo de nueva tecnología, así como la cantidad de dispositivos es impresionante, cada vez son más los dispositivos que necesitan o permiten una conexión a internet para su mayor provecho, esto ligado a que se había

distribuido mal el rango de direcciones, como por ejemplo se había destinado un amplio rango de hosts innecesarios a entidades gubernamentales (como lo hablamos anteriormente con las direcciones de IPv4 de clase A), nos llevaba a un inminente colapso.

Pero la verdad a todo esto, es que este ocaso de las direcciones IPv4 a nivel mundial ya se venía venir, es decir no cayó de sorpresa; y durante varios años se han venido desarrollando varias soluciones a la escasez de direcciones en IPv4.

Una de estas soluciones es la NAT, con la cual todos los dispositivos conectados dentro de una red LAN, utilizaban una sola dirección pública para por medio del enrutador, comunicarse a través de Internet, y de la misma manera al momento de recibir paquetes, todos llegan al enrutador con la dirección pública, y este debe ser capaz de identificar para que dispositivo con dirección IP privada está destinado la recepción del paquete.

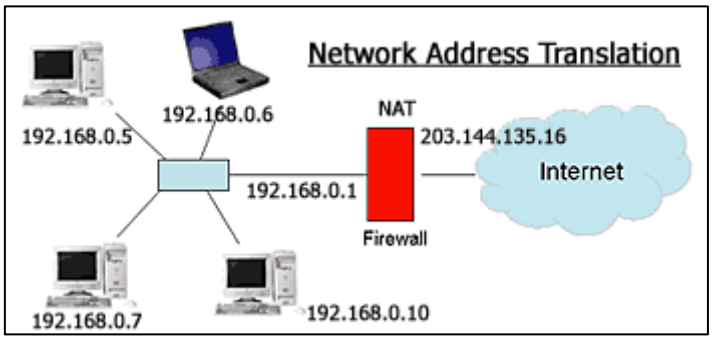


Figura # 2.4. Tecnología NAT

Otra de las soluciones al problema de la escasez de direcciones IPv4 son las famosas subredes, es decir todas estas herramientas ya las veníamos aplicando desde mucho antes que se agoten las direcciones públicas a nivel mundial.

Sin embargo una de las soluciones más acertadas, y hacia dónde vamos, es a la evolución del protocolo de Internet, de su versión 4 a la versión 6. IPv6 nos trae muchísimos beneficios, y aunque de cierta manera no se ha tenido una acogida total, con respecto a su implementación; es un destino al que poco a poco nos estamos mudando.

2.3. IPv6

2.3.1. Características de IPv6

Cuando hablamos de protocolo IPv6, ya no debemos preguntarnos si sería una opción acertada hacerlo; la pregunta que nos debemos hacer es, cuando nos vamos a cambiar y como lo haremos.

Debido al inevitable fin del protocolo IPv4, y aunque de muchas maneras se ha tratado de continuar con su uso, el protocolo IPv6 cada vez comienza a tomar mayor protagonismo dentro del mundo del “networking”; aunque no es algo nuevo, y se lo viene incorporando de a poco, es algo a lo que deberíamos ponerle más énfasis, comprender cada vez más este protocolo; y disfrutar de todas las bondades que nos brinda.

Una de sus principales bondades y características es que pone fin al problema de direccionamiento que sufría IPv4, ya que posee un esquema de direcciones mucho mayor. Al mismo un espacio mayor para las direcciones IP, permitirá a los servidores de internet una mejor organización y distribución. Cabe recalcar que una red basada en IPv6, trabaja exclusivamente con ICMPv6.

Otras de sus características es que en IPv4 se utilizaba mucho la difusión del protocolo de Resolución de Direcciones (ARP), que por cierto ha sido motivo de vulnerabilidades dentro de la red para posibles ataques, y esto afectaba a la eficiencia de la red. Ahora el método que se utiliza es el de multicasting. Que crea grupos de interfaces de red; disminuyendo los ciclos de procesamiento en el CPU al no procesar paquetes de difusión que no van dirigido a ellos.

Un gran cambio también se lo puede visualizar en su encabezado, este nos brinda muchas bondades a lo que estábamos acostumbrados anteriormente, es mucho más eficiente; por ejemplo entre sus cualidades esta que tiene menos campos, lo que provoca que se elimine la tarea de suma de verificación que se acostumbraba a realizar en el encabezado; entre los campos que ha eliminado están: el de la longitud del encabezado, identificación, banderas, desplazamiento por fragmentación, la suma que ya la mencionamos, opciones y relleno.

Además existe un campo llamado Etiqueta de Flujo, este los maneja el nodo fuente para poder realizar una diferenciación especial de las secuencias específicas de los paquetes; esta etiqueta va dirigida para el procesamiento en el dispositivo de destino, no para los

enrutadores que hayan en el camino, ayuda a realizar una diferenciación del tráfico de la red, siendo de gran ayuda para aplicaciones de VoIP o videoconferencias, cada una con el respectivo tratamiento en la calidad de servicio en cada uno de los enrutadores que atraviesa.

2.3.2. Estructura de la dirección IPv6

La estructura del protocolo IPv6 está formada por el encabezado, extensiones y el direccionamiento.

El encabezado se encuentra formado por ocho campos, lo que da lugar a un total de 40 octetos. Entre los campos tenemos a:

- La versión, está formada por 4 bits, aquí detalla la versión del paquete, sabemos que viene el número 6.
- Clase de tráfico, este ocupa 8 bits, aquí se etiqueta al paquete con un Punto de Código de Servicios Diferenciados (DSCP), en el que especifica cómo debe ser manejado.
- Etiqueta de Flujo, son 20 bits, marca la secuencia de los paquetes a través de la red, además de diferenciar a los paquetes que necesiten un tratamiento a lo largo de la trayectoria.

- Longitud de Carga Útil, de 16 bits, es la parte que continua al encabezado en IPv6.
- Siguiente Encabezado, formado por 8 bits, se define el tipo de información que tomara el encabezado, puede ser un protocolo como TCP o UDP, o a las extensiones de encabezado.
- Límite de saltos; 8 bits, define la cantidad máxima de saltos que puede atravesar un paquete. Este tiene un pequeño contador, y cada salto que da va disminuyendo su valor en 1, y cuando ese valor llega a 0, el paquete será destruido, enviando un mensaje al nodo del cual salió el paquete, un mensaje indicado que el tiempo ha sido excedido.
- Dirección Fuente, 128 bits, especifica la dirección IPv6 de la cual sale el paquete.
- Dirección Destino, 128 bits, especifica la dirección IPv6 a la cual debe llegar el paquete.

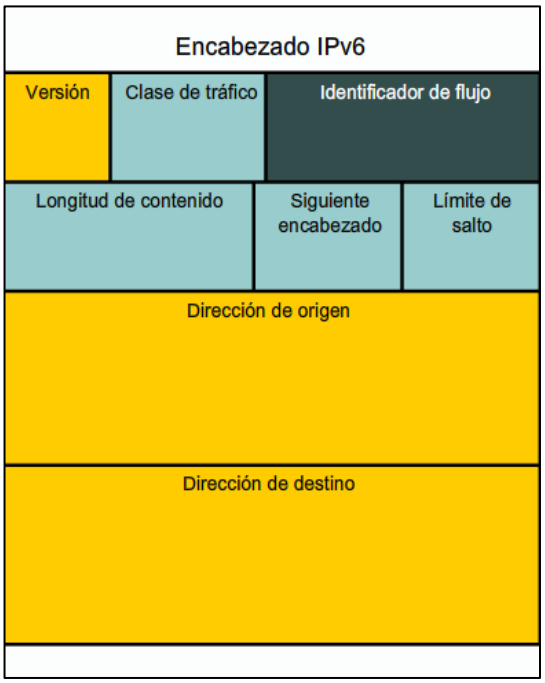


Figura # 2.5. Encabezado IPv6

En lo que respecta al direccionamiento, este consiste en 8 sets, de 16 bits hexadecimales, cada set está separado por ":", y todos juntos forman los 128 bits, que es la longitud de una dirección de protocolo IPv6.

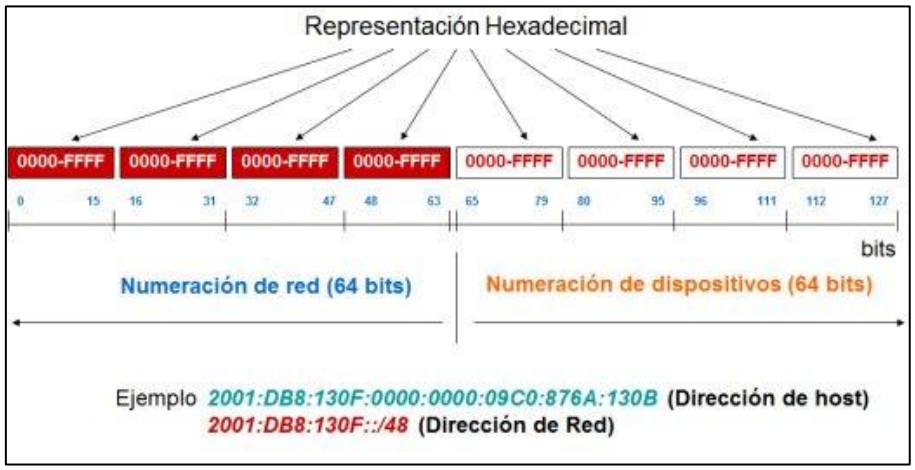


Figura # 2.6. Direccionamiento IPv6

Existen varios tipos de dirección, entre ellas están el unicast, multicast y anycast.

Cuando hablamos acerca de una dirección unicast, nos referimos a una interface de un nodo de la red IPv6. Cada paquete es entregado a la dirección de la interface que indique.

Si nos referimos a direcciones multicast, estamos asociando varias interfaces de red, y cuando enviemos un paquete a esa dirección multicast, el paquete será enviado a todos los miembros de interfaces que conforman el grupo multicast. Y por último, anycast, se refiere también a múltiples interfaces, pero estas por lo general se encuentran en diferentes nodos.

Una dirección unicast, está formada por 128 bits, los 64 primeros corresponden la parte de identificación de la red, cabe indicar que esta parte es asignada administrativamente, y los 64 bits restantes corresponden a la parte de identificación del host, y aquí lo configuramos de manera manual o se lo auto-configura los varios métodos, como DHCPv6, generando un numero aleatorio, o usando EUI (Extended Unique Identifier).

2.3.3. Requerimientos de software y hardware

Como ya lo hemos mencionado, el cambio del protocolo IPv4 a IPv6 es innegable, no es una utopía, es una realidad. Un cambio a IPv6 influye en toda nuestra estructura de la red y protocolos que esta envuelve; y aunque para la mayoría de los usuarios, este cambio será imperceptible, para muchos envueltos en la materia, tienen que estar preparados para esta nueva faceta.

Cada cambio de tecnología que atravesamos, nos afecta tanto en software como en hardware, y este protocolo no es la excepción. Si tenemos nuestros equipos funcionando con IPv4, al momento de comunicarse con dispositivos IPv6 vamos a tener problemas de comunicación.

Dado a que la adopción de este nuevo protocolo no se ha dado de manera repentina, varias empresas desarrolladoras lo que han hecho es adaptar estos equipo para que además que soporten IPv4, también trabajen en IPv6; sin embargo cabe recalcar que esta modificación es un costo adicional para la inversión inicial del equipo, lo que a la larga lo hace que su costo sea mucho más elevado. Dado que de alguna manera nos rehusamos a dejar a un lado IPv4, esta modalidad se la ha venido implementado, y ha funcionado muy bien.

Los métodos para la implementación de estos dos protocolos se la ha venido manejando con “dual stack” y “tunnelling”. La primera se encarga de proveer implementaciones completas de ambos protocolos de red, mientras que “tunnelling” lleva paquetes de IPv6 dentro de estructuras de paquetes no modificados de IPv4.

Para la transición es importante que los equipos sean capaces de soportar IPv6, con memoria suficiente para dar servicio de IPv6, que en el software apliquemos los diferentes protocolos que deseemos utilizar, cumplir los requisitos de diferentes RFCs que han sido publicadas; y también es muy importante que nuestro persona de IT este correctamente capacitado, ante cualquier eventualidad.

2.4. Tipos de ataques en redes de datos

2.4.1. Man In The Middle en IPv6 con Neighbor Advertisement Spoofing

En la actualidad, el uso del internet y conexiones a otras redes nos resultan de gran ayuda para la interacción con el resto del mundo, pero a la vez podemos ser atacados por personas con malas

intenciones que capturan nuestra información y la manipulan para perjudicarnos. Uno de los ataques más conocidos y utilizados en la actualidad en las redes de datos IPv4/IPv6 es el llamado “Man In The Middle” u “Hombre en el Medio”.

Este ataque tiene como objetivo leer, insertar y modificar a voluntad la información que se intercambia entre dos equipos sin que éstos no se den cuenta de que el enlace entre ellos ha sido violado. Este ataque es usado en ambos protocolos IPv4/IPv6 usando diferentes técnicas.

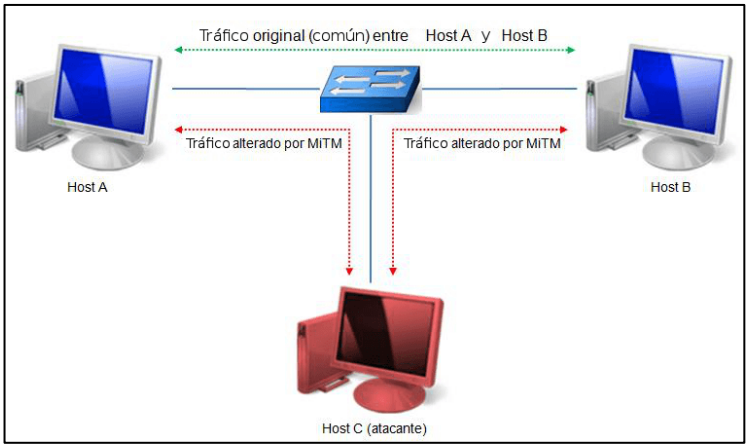


Figura # 2.7. Ataque Man In The Middle

En este caso el ataque será para IPv6 usando Neighbor Advertisement Spoofing el cual es similar al ataque ARP Spoofing

que existe para IPv4 usando el protocolo ARP, pero en IPv6 ese protocolo ya no se utiliza lo cual ahora se utiliza ICMPv6 que cumple funciones como las de ARP pero utilizando Neighbor Discovery Protocol(NDP).

El NDP consiste en un mecanismo con el cual un nodo que se incorpora a una red descubre la presencia de otros nodos en el mismo enlace y también puede ver sus direcciones IP. El funcionamiento normal de la comunicación en IPv6 debe ser que el nodo envía un mensaje Neighbor Solicitation(Solicitud de Vecino) como multicast para localizar su destino y luego de encontrarlo el destino envía un mensaje Neighbor Advertisement como unicast al origen.

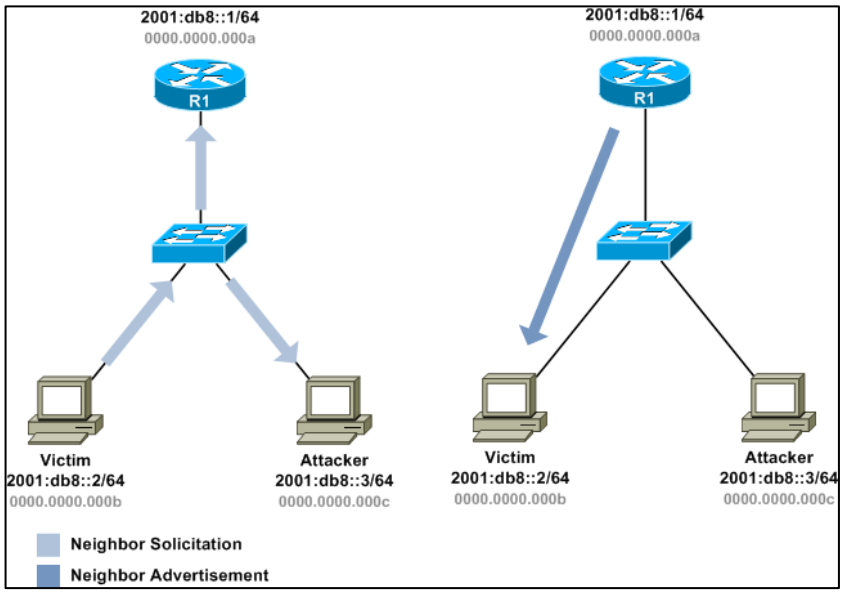


Figura # 2.8. Envío de paquetes para detección de Nodos

El ataque se realiza spoofeando la dirección IPv6 de origen del paquete, para simular ser un mensaje que viene del otro equipo víctima, pero en ambos casos se pone la dirección MAC del atacante, para conseguir que el conmutador de comunicaciones haga llegar todos los mensajes a la máquina del hombre en medio.

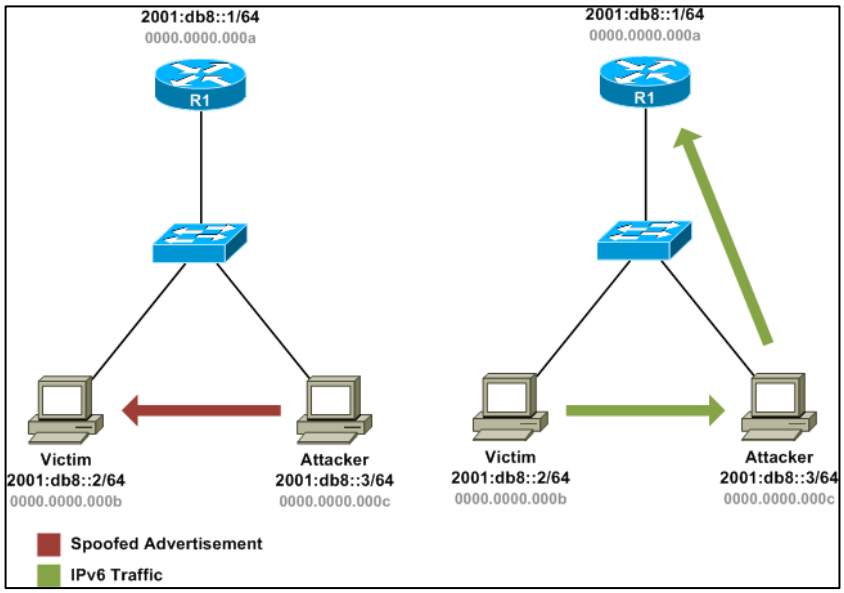


Figura # 2.9. Descripción del Ataque Man In The Middle en IPv6 con Neighbor Advertisement Spoofing

2.4.2. Man In The Middle IPv4 con ARP Spoofing

Una de la forma más antigua de lograr un ataque Man-in-the-middle es utilizar ARP Spoofing. Con ARP Spoofing el objetivo del atacante es la capa dos del protocolo de direcciones MAC.

En una red de área local, un host se comunica con otro host, incluyendo la puerta de enlace (gateway), mediante la entrega de paquetes a una dirección MAC. Primero el protocolo ARP tiene que resolver la dirección MAC de la dirección IP de un host. No existe un

procedimiento de comprobación o de autenticación del protocolo ARP. Cuando un equipo necesita enviar información a otro host en la red el ordenador genera un ARP broadcast. El ARP broadcast es enviado a cada ordenador de la red solicitando que una dirección IP específica responda con la dirección MAC correspondiente. Cuando un equipo responde con una dirección MAC, los datos pueden ser entregados a esa dirección MAC, el problema es que la respuesta se acepta sin verificación.

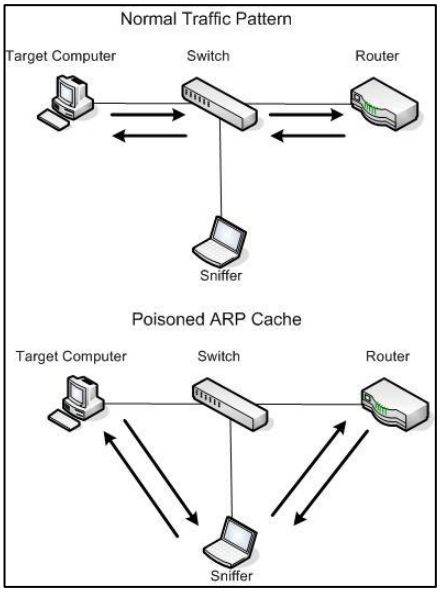


Figura # 2.10. Interceptar Comunicación con ARP Spoofing

ARP Spoofing implica el envío de información fraudulenta a los hosts atacados por lo que incorrectamente mapa la dirección MAC del atacante como pertenecientes a la dirección IP.

2.4.3. Man In The Middle en IPv6 con SLAAC

En IPv6 existe otra forma de realizar Man In The Middle y es utilizando el SLAAC (Stateless Address Autoconfiguration) o Autoconfiguración de direcciones libres de estado. Para realizar este ataque, el atacante debe introducir un router (o algún dispositivo que actúe como tal, puede ser su propio ordenador) en la red interna con dos interfaces (virtuales o no): una de cara a la red interna, que soporte solamente IPv6 y otra con la conexión a Internet (solamente IPv4). En esos momentos existirá una red adicional IPv6, pero el atacante no controlará el tráfico. El intruso comenzará a enviar RA (Router Advertisements, anuncios de rutas), que es una especie de DHCP para IPv6. El objetivo es que el tráfico pase a través de la interfaz IPv6 sin que los clientes noten nada y esto se consigue gracias a una especificación obsoleta.

NAT-PT es un mecanismo que permite traducir de IPv4 a IPv6 y viceversa para que dispositivos que soporten una u otra versión puedan comunicarse. Un protocolo ideado para facilitar la migración entre redes que fue abandonado en 2007 porque resultaba

demasiado complejo, contenía demasiados errores. El método es definir en el router un prefijo IPv6 e incrustar en los últimos 32 bits una dirección IP versión 4, que según el ataque previsto, debe coincidir con un servidor DNS del propio atacante, situado en la interfaz IPv4 del router (en Internet). Si se configura adecuadamente ese router del atacante para que se encargue de traducir (a través de NAT-PT) las direcciones IPv6 de las víctimas a IPv4, se consuma el ataque, engañando al usuario para que crea que su servidor DNS es el del atacante.

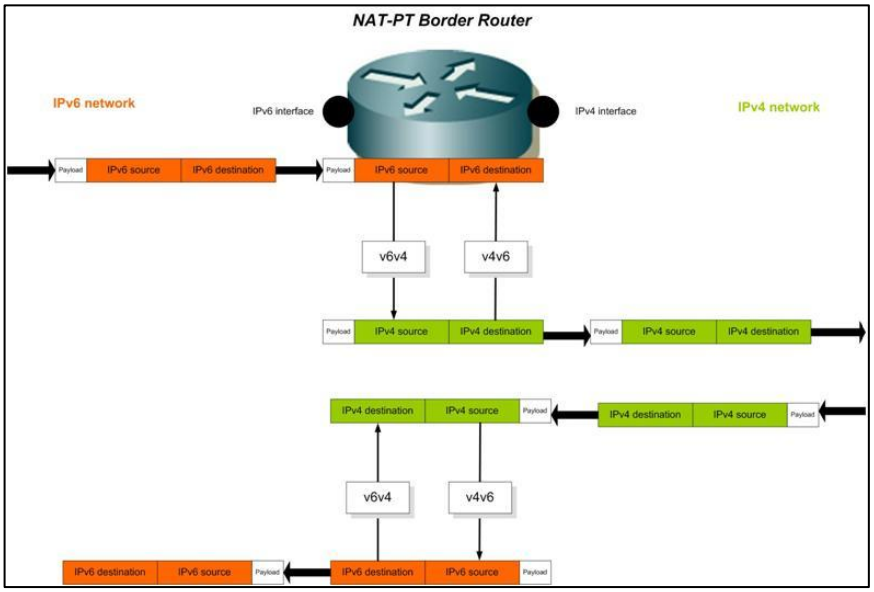


Figura # 2.11. Detalle de Ataque MITM con SLAAC

El siguiente paso es hacer que los sistemas operativos usen la red IPv6 (y sus DNS) creada paralelamente y que lo hagan instantáneamente (si no responde a tiempo, se usaría el DNS legítimo). Esto se consigue de forma muy sencilla por dos razones: La primera es el uso de Application Layer Gateways (ALGs), que es necesario en NAT-PT para hacer NAT en protocolos "especiales" como FTP. La segunda es que los sistemas operativos modernos prefieren siempre utilizar IPv6 (se han diseñado así para, facilitar la migración).

En resumen, la víctima utiliza sin darse cuenta el DNS del atacante para resolver direcciones y, por tanto, puede ser redirigido a cualquier página (que no use certificados) de forma transparente.

2.4.4. DoS en IPv4 con Invalid MAC Spoofing

Uno de los ataques más famosos, del que siempre estamos escuchando en los medios de comunicación es el ataque de denegación de servicio (DoS).

El ataque lo realizamos al servidor, saturándolo sus puertos con flujos de información y peticiones, llega un momento en que el

servidor no puede procesar todas las peticiones, lo que hace que colapse, haciendo que no pueda proveer el servicio; a este fenómeno se lo denomina denegación.

El periférico que nos facilita la conectividad entre demás dispositivos es la interfaz de red; cada interfaz de red viene asignada un número de identificación, a este lo denominamos MAC. La dirección MAC está conformada por 48 bits hexadecimales, en los 24 primeros se identifica el proveedor de la interfaz; y estas direcciones son únicas, y son asignadas por la IEEE.

El Invalid MAC Spoofing es una técnica que consiste en cambiar la dirección MAC en un dispositivo de red, por otro que no es el que le corresponde. Para que todas las peticiones que le deberían llegar al servidor, se pierdan en la red. Haciendo que tanto el servidor no sea capaz de proporcionar su servicio, ya que las peticiones correctas no llegan a este, así como que los paquetes que logren llegar podrían no ser legítimos, sino paquetes alterados.

2.4.5. DNS Hijacking

Para poder acceder a diferentes páginas en internet, acostumbramos a poner el nombre del dominio en la barra de

direcciones de nuestro navegador preferido. Pues este nombre de dominio es traducido a una dirección IP publica, que hace referencia al sitio web, que deseamos acceder, esta traducción de nombre de dominio a IP lo hace nuestro servidor DNS (Domain Name System).

Muchas veces este servidor se encuentra dentro de nuestra propia red Lan y es un blanco utilizado por los hackers, con las intenciones de redirigir o "secuestrar" las direcciones DNS, a los servidores DNS falsos; con el fin de inyectar malware en el PC; prueba de ello son las promociones de estafas de phishing, la publicidad en los sitios web de alto tráfico, y cualquier otra forma relacionada de la actividad criminal.

Una vez que el atacante logra secuestrar nuestro servidor, se transforma en un DNS falso, ya que en vez de traducir el dominio en una dirección legítima, este nos envía direcciones IP falsas, re direccionándonos a sitios web maliciosos. DNS Hijacking puede ocurrir con cualquier sitio web grande o pequeño y convertir esos sitios web en los sitios web maliciosos sin el conocimiento del internauta.

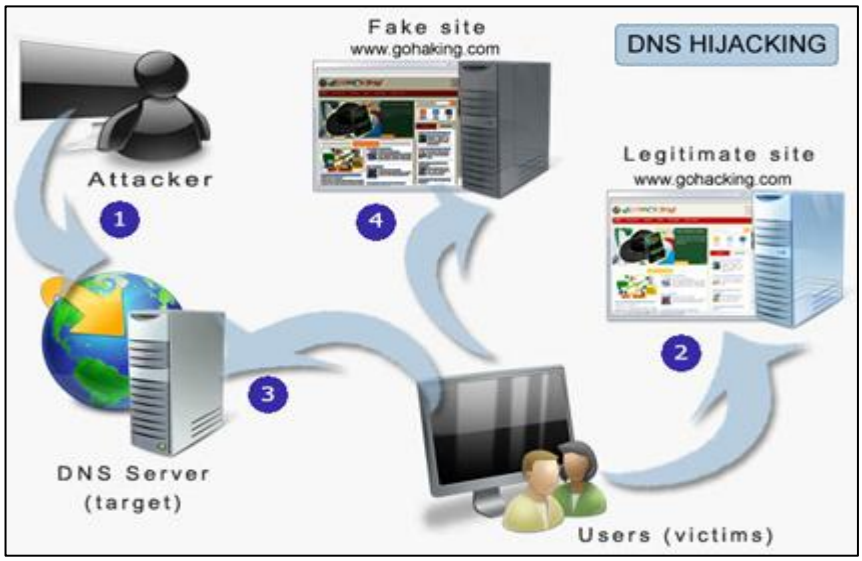


Figura # 2.12. Funcionamiento del Ataque DNS Hijacking

Dado no solo los usuarios sino que los propietarios de sitios web, dependen de servidor DNS legítimo, que se emitió por sus proveedores de servicios de Internet (ISP), secuestradores DNS utilizan el malware en forma de un troyano para intercambiar la asignación del servidor DNS legítimos por el ISP con una asignación de servidor DNS manual desde un falso servidor DNS.

Cuando los internautas visitan los sitios web de confianza con los nombres de dominio legítimos, son secuestrados automáticamente a un sitio web malicioso que se disfraza como el legítimo. El enrutador

del servidor DNS legítimo al servidor DNS falso pasa desapercibido tanto por el internauta y el propietario legítimo sitio web. Esto abre el sitio web malicioso para realizar cualquier acto criminal que el hacker desea porque el usuario piensa que son en el sitio web real.

2.5. Python

2.5.1. Descripción

Python es un lenguaje de scripting independiente de plataforma y orientado a objetos, es un lenguaje de programación muy poderoso que nos permite realizar cualquier tipo de programa, desde aplicaciones web, o aplicaciones en Windows hasta servidores de red. Se desarrolla como un proyecto de código abierto. Es un lenguaje de programación interactivo, e interpretado, esto significa que no necesita compilar el código para poder ejecutarlo, lo que lo hace con rapidez de desarrollo y con inconvenientes a menor velocidad.

Ya hace varios años, un individuo motivado por su experiencia en otro lenguaje de programación ABC, diseña Python. Su nombre es Guido Van Rossum. El buscaba reducir la dificultad al momento de utilizar un lenguaje de programación orientado a objetos, para

realizar diversas tareas que por lo general se llevaban a cabo usando C. Sin embargo este lenguaje no surgió de la noche a la mañana, su creación duro varios años.

Python aunque es muy fácil de usar, es un lenguaje de programación muy poderoso, nos brinda estructuras de datos muy eficientes, de alto nivel y siempre sin perder de vista la meta que es de hacerlo de una manera sencilla. Su fácil sintaxis, su modo de interpretar los comandos y su extensa biblioteca hacen de este un lenguaje perfecto para el desarrollo de aplicaciones en diversos campos, y cabe indicar que también en diferentes plataformas.

2.5.2. Ventajas y Desventajas de utilizar Python

Como ya lo hemos mencionado anteriormente, Python lleva una serie de ventajas con respecto a otros lenguajes de programación.

Una de ellas es su fácil manejo del programa, ya que nos permite un rápido desarrollo de las aplicaciones que queremos realizar.

Su manera de trabajar hace que todo lo que se realice en Python sea de fácil comprensión, su simplicidad deja entrever que aunque es un lenguaje muy potente no deja de ser sencillo.

Su inmensa biblioteca, no olvidemos que Python es de código abierto, esto ayuda muchísimo a los desarrolladores a la hora de programar, ya que tienen a su disposición una infinidad de biblioteca para dejar volar su imaginación.

Otro atractivo es su buen manejo de las bases de datos, ya que soporta diversos tipos y de una manera muy eficiente.

Sin embargo no todo es perfecto, y es que aunque al ser un lenguaje interpretado, es decir que no necesita una compilación para ejecutarse; hace que las aplicaciones desarrolladas en lenguajes interpretados son más lentas de los compilados; aunque las aplicaciones en Python son más cortas lo que hace que esa lentitud sea inapreciable.

2.5.3. Ejemplo de aplicaciones basadas en Python

En el mercado existe un sinnúmero de aplicaciones desarrolladas en Python, entre sus más conocidas tenemos a:

Twisted Matrix Labs

Figura # 2.13. Logo de la empresa desarrolladora de Twisted

Twisted es un marco de programación de la red dirigida por eventos escrito en Python y licenciado bajo la licencia MIT. Los proyectos Twisted soportan TCP, UDP, SSL/TLS, multicast IP, sockets de dominio Unix, un gran número de protocolos (incluyendo HTTP, XMPP, NNTP, IMAP, SSH, IRC, FTP y otros), y mucho más. Twisted se basa en el paradigma de la programación orientada a eventos, lo que significa que los usuarios de Twisted escriben devoluciones de llamada cortos que son llamadas por el marco¨.

Fluendo

Figura # 2.14. Logo de Fluendo

Fluendo tiene como objetivo mejorar la experiencia multimedia global en el mundo del software libre por la financiación, el desarrollo y mantenimiento del marco medios GStreamer y ofrecer una amplia gama de, productos comerciales y libres, en la parte superior de la misma.

Mailman

Figura # 2.15. Logo de Mailman

GNU Mailman es una aplicación de software del proyecto GNU, que maneja listas de correo electrónico o simplemente listas de correo. Mailman se compone principalmente de código en lenguaje Python actualmente es mantenido por Barry Warsaw. Mailman es software libre, y se distribuye bajo la licencia GNU General Public License

BitTorrent

Figura # 2.16. Logo de BiTorrent

BitTorrent Inc. es una compañía de tecnología de Internet con sede en San Francisco. Diseñamos tecnologías distribuidas que escalan de manera eficiente, mantenemos la inteligencia en el borde, y mantenemos los creadores y los consumidores en el control de su contenido y datos. Más de 170 millones de personas utilizan nuestros productos cada mes. Nuestros protocolos se mueven tanto como el 40% del tráfico mundial de Internet sobre una base diaria.

CAPÍTULO 3

ANÁLISIS Y DISEÑO DEL CONJUNTO DE APLICACIONES

Para el desarrollo del conjunto de aplicaciones para la detección de ataques en redes IPv4 e IPv6, nos hemos propuesto hacerlo de una manera sencilla y de fácil manejo para el usuario.

El lenguaje de programación que hemos utilizado, como lo hemos venido comentando es Python, para este tipo de aplicaciones. Python nos provee una gran variedad de herramientas para la elaboración de esta aplicación. Cabe recalcar que para el diseño de la aplicación y la elaboración del programa,

utilizamos la versión de Python 2.7; y además lo desarrollamos dentro del sistema operativo Ubuntu.

La arquitectura en la cual se basa este conjunto de aplicaciones es el siguiente:

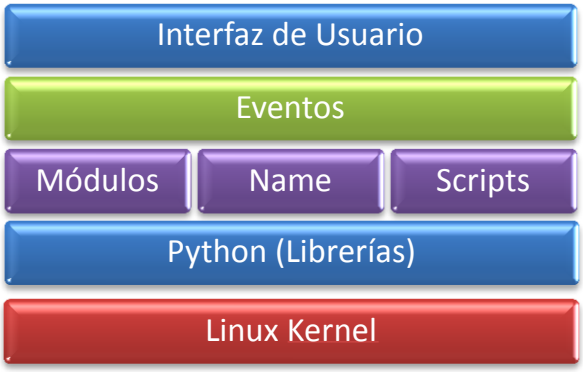


Figura # 3.1. Arquitectura de la aplicación general

Como ya lo mencionamos, la aplicación fue desarrollada en Ubuntu, por lo que esta va a correr sobre el kernel de Linux, es aquí donde Python hará su trabajo, ejecutaremos librerías muy conocidas, que darán al usuario una interfaz gráfica muy amigable y fácil de manejar

Para su interfaz gráfica hemos tratada de resumirla en una sola ventana de presentación inicial. Esta será nuestra carta de presentación. Como veremos en la ventana, tendremos los iconos correspondientes a cada ataque que la aplicación está preparada para detectar.

Figura # 3.2. Interfaz Gráfica de la ventana principal

Es decir, dentro de esta ventana, si necesitamos comenzar a detectar un ataque específico, lo que tendremos que hacer, simplemente será de dar un clic con nuestro cursor, sobre el ícono del ataque que deseamos detectar. Esto hará, que el programa ejecute un nuevo script, haciendo que aparezca una nueva ventana indicando que está preparado y atento para la detección de un ataque; las ventanas para los diferentes ataques, son muy parecidas y gráficamente funcionan de la misma manera. Esto hace que cumplamos con uno de nuestros objetivos en la realización del software, que era que nuestra aplicación sea muy sencilla para el usuario.

Además de los íconos referentes a cada ataque, es importante señalar, que para detectar un posible ataque, la suite de aplicaciones debe estar corriendo, es decir, el script de la ventana principal debe estar activo, y a su vez,

seleccionar el ataque que deseamos capturar. Si cerramos estas ventanas, el programa no se ejecutará en segundo plano.

3.1 Desarrollo de la aplicación para el ataque MITM en IPv6 con Neighbor Advertisement Spoofing.

3.1.1. Análisis de detección del ataque.

Para poder realizar la detección se debe saber cómo funciona el ataque, la victima envía un mensaje Neighbor Solicitation (NS) a una dirección multicast para buscar los nodos que se encuentran en la red y encontrar el dispositivo con el que se desea comunicar, una vez que recibe el mensaje el nodo receptor este envía un mensaje unicast de Neighbor Advertisement (NA) y almacenara en la tabla de vecinos la dirección MAC asociada con la dirección IPv6.

El atacante aprovecha esos envíos de mensajes para interceptarlos y enviar mensajes Neighbor Advertisement a ambos equipos poniendo la dirección IPv6 del otro y la dirección del atacante.

Por lo que para realizar la detección nosotros realizamos captura de paquetes con un “sniffer” de la librería scapy, pero la efectividad se encuentra en que debemos realizar filtros para obtener los paquetes.

El primer filtro es obtener todos los paquetes IPv6, luego hay que capturar todos los nodos IPv6 que incluyen enrutadores y Neighbor Advertisement con el siguiente código que nos permite realizarlo:

Filter= "ip6 and not tcp and not udp"

Luego obtenemos los paquetes que son de Neighbor Advertisement ICMPv6 NA los cuales corresponden a la detección de este ataque, vemos si dicho paquete no tiene como destino la dirección multicast ff02::1 entonces estamos siendo atacado. Obtendremos la siguiente información :

- **Atacante:** pkt[IPv6].src
- **Dirección MAC del Atacante:** pkt[Ether].src
- **Objetivo:** pkt[ICMPv6ND_NA].tgt

3.1.2. Arquitectura de desarrollo.

La arquitectura de la aplicación para la detección de este ataque es la siguiente:

Figura # 3.3. Arquitectura de aplicación de detección de ataques IPv6

Aquí se puede observar las distintas capas que permiten realizar la aplicación de detección empezando por la capa de Linux Kernel que fue utilizado para empezar el desarrollo debido a que presta facilidades al acceso de la siguiente capa que son las librerías de

Python principalmente la de scapy que es la que permite realizar las captura de paquetes.

En la siguiente capa se encuentran los módulos y scripts que presenta la aplicación, esta es la que contiene los procesos y funciones que permiten la detección y generación de eventos. En los eventos se visualizaran los mensajes de alerta que aparecerán una vez que ocurra alguna detección de intrusos.

Finalmente la última capa es la interfaz de usuario la cual nos permite visualizar que es lo que ocurre de una manea amigable para el usuario.

3.1.3. Diseño de la interfaz gráfica.

Su interfaz gráfica se basa en dos simples ventanas; cuando seleccionamos el ataque Man In The Middle en IPv6 con Neighbor Advertisement Spoofing, se ejecutará el script y aparecerá la siguiente ventana.

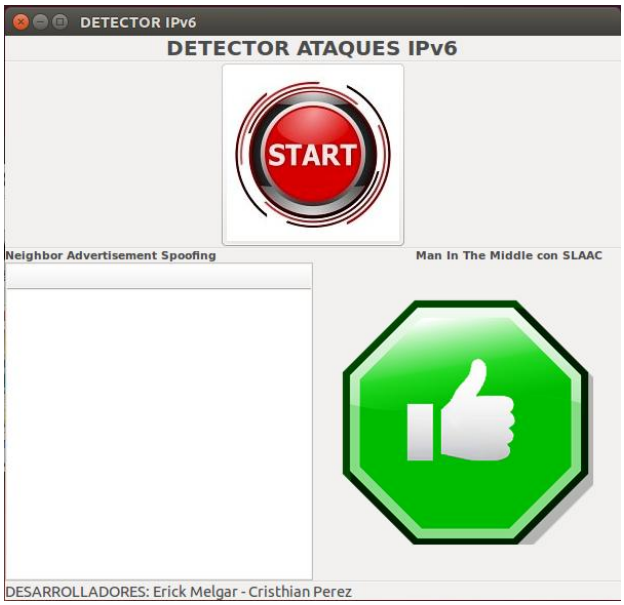


Figura # 3.4. Interfaz gráfica de Detección de IPv6 – Estado Inicial

Como podemos ver aquí nos indica, que está listo para detectar un posible ataque de este tipo, y que hasta el momento no lo ha detectado. La aplicación de detección iniciara una vez que se presione el botón de “START” que se encuentra en la parte superior. Mientras no se encuentre actividad anormal el aplicativo mostrara esa imagen de color verde, pero una vez que lo detecta, su apariencia automáticamente cambia y se cambiara a un símbolo de “ADVERTENCIA” que significa que está siendo atacado, en el costado izquierdo se mostrara información sobre el ataque en proceso.

Figura # 3.5. Interfaz gráfica de Detección de IPv6 – Estado Advertencia

Este aplicativo funciona para los dos ataques de IPv6 solo que cuando se detecte un ataque en particular te permitirá saber cuál es el nombre del mismo.

3.2 Desarrollo de la aplicación para el ataque MITM en IPv4 con ARP Spoofing.

3.2.1. Análisis de detección del ataque.

Este ataque es muy común en las redes IPv4 lo cual realiza un envenenamiento a la tabla ARP simulando a las víctimas que tienen una comunicación directa entre ellos sin saber que existe un intermediario que es el atacante.

El método de detección es realizar una captura de paquetes IPv4 realizando un filtro a todo el tráfico “ARP”, ahí obtenemos los datos del enrutador por lo cual si nuestra IP se encuentra en el registrado entonces lo almacenamos en una variable igual que la MAC.

Si en un futuro recibimos una IP del enrutador que no estaba almacenada en la variable anterior y a la vez no se encuentra la MAC registrada anteriormente almacenada es porque estamos siendo atacados. Por lo cual mostraremos una alerta para decir al usuario que está siendo atacado.

3.2.2. Arquitectura de desarrollo.

La arquitectura para esta aplicación es la siguiente:

Figura # 3.6. Arquitectura de aplicación de detección de ARP Spoofing

Aquí la arquitectura es similar a las anteriores solo con la diferencia en los scripts desarrollados en Python son para IPv4 y para el ataque especifico que es el ARP Spoofing.

3.2.3. Diseño de la interfaz gráfica.

En esta interfaz se muestran la ventana para la detección única de ARP Spoofing lo cual iniciara al momento de presionar el botón “START” que se encuentra en la parte superior y si aún no detecta actividad sospechosa se mostrara de la siguiente manera.

Figura # 3.7. Interfaz gráfica de Detección de ARP Spoofing – Estado Inicial

Cuando llegara a encontrar una anomalía por motivo del ataque realizado entonces advertirá al usuario mostrándole la información necesaria y cambiando el icono de verde al de advertencia como se lo muestra a continuación.

Figura # 3.8. Interfaz gráfica de Detección de ARP Spoofing – Estado Advertencia

3.3 Desarrollo de la aplicación para el ataque MITM en IPv6 con SLAAC.

3.3.1. Análisis de detección del ataque.

Dentro de las funcionalidades en la implementación de IPv6, una de ellas consiste en la posibilidad de realizar una configuración rápida de un adaptador de red, donde los parámetros son proporcionados por un enrutador.

La capacidad de SLAAC viene definida nuevamente a través de la RFC4861 y puede ser utilizada por un atacante a través de la red IPv6 y acceder a todas las comunicaciones.

El objetivo del ataque es poder hacer un “Man in The Middle” cuando un usuario se conecta a Internet a un servidor que no tiene soporte para IPv6 y que por lo tanto es necesario conectarse usando IPv4.

La metodología para la detección es realizar la captura de paquetes con un “sniffer” de la librería scapy, pero la efectividad se encuentra en que debemos realizar filtros para obtener los paquetes.

El primer filtro es obtener todos los paquetes IPv6, luego hay que capturar todos los nodos IPv6 que incluyen enrutadores y Neighbor Advertisement con el siguiente código que nos permite realizarlo:

Filter= “ip6 and (dst host ff02::1)”

Usamos la dirección multicast ff02::1 que representa todos los nodos que se encuentran en el segmento de red, también se lo puede filtrar por la dirección multicast ff02::1:2 que es para los

servidores DHCP y los agentes de retransmisión de la red. De ahí almacenamos en una variable los mensajes RA(Router Advertisement) que son del enrutador que inicialmente las envio, luego con el “sniffer” realizado en el paso anterior filtramos los nuevos mensajes RA para compararlos y si no pertenecen al enrutador original entonces advertiremos al usuario que está siendo atacado indicando la mac del atacante con el comando:

pkt[ICMPv6NDOptSrcLLAddr].lladdr

3.3.2. Arquitectura de desarrollo.

La arquitectura de esta aplicación es la misma que del primer ataque debido a que ambos son para IPv6 y tienen similitudes en la captura de paquetes, por lo cual su arquitectura es la siguiente:

Figura # 3.9. Arquitectura de aplicación de detección de ataques IPv6

Cumpliendo las mismas funciones cada capa de la arquitectura ya mencionada.

3.3.3. Diseño de la interfaz gráfica.

La interfaz gráfica de este ataque es similar a las demás con un botón de “START” al inicio de la pantalla para poder inicializar la aplicación, a la vez si no existe ningún ataque desde que inició el proceso entonces permanecerá con el estado normal y el icono verde.

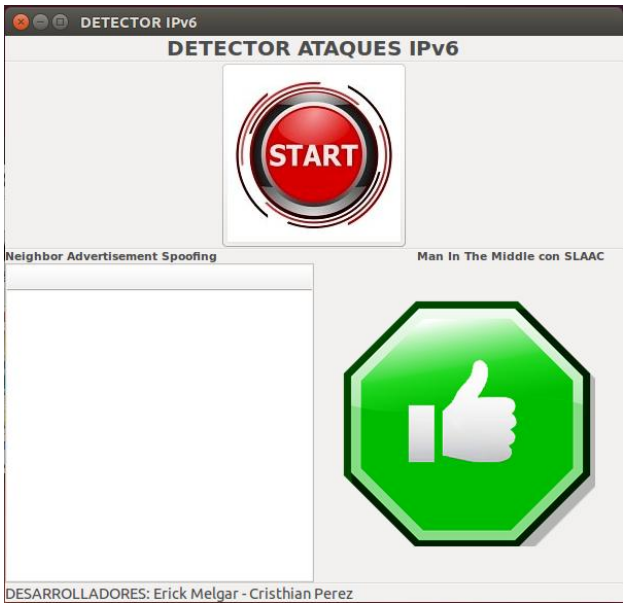


Figura # 3.10. Interfaz gráfica de Detección de IPv6 – Estado Inicial

Y cuando llegara a detectar un ataque se mostrara la siguiente ventana con el símbolo de advertencia e ilustrando la información producida.

Figura # 3.11. Interfaz gráfica de Detección de IPv6 – Estado Advertencia

3.4 Desarrollo de la aplicación para el ataque DoS en IPv4 con Invalid MAC Spoofing.

3.4.1. Análisis de la detección del ataque.

Este ataque es similar al ataque “Man In The Middle” con ARP Spoofing solo que en este caso lo que realiza es una denegación de

servicio por lo que se detecta con el método para ARP Spoofing pero además se necesita saber si hay respuesta con el otro nodo por lo que se realiza un tiempo de espera y si no hay respuesta entonces será por negación de servicio.

3.4.2. Arquitectura de desarrollo.

La arquitectura es la misma del ataque en IPv4 ARP Spoofing debido a que poseen funcionalidades similares al momento de ejecutarse, por lo tanto su diseño de arquitectura será el siguiente:

Figura # 3.12. Arquitectura de aplicación de detección de DoS Invalid MAC Spoofing

3.4.3. Diseño de la interfaz gráfica.

El interfaz gráfico de este aplicativo es igual a los demás que permiten iniciar el proceso y si no existe amenaza se muestra de la siguiente manera:

Figura # 3.13. Interfaz gráfica de Detección de DoS MAC Invalid – Estado Inicial

El cambio de una ventana a otra, nos indica una alerta, de que dentro de nuestra red se está produciendo un ataque. En esta ventana nos muestra una advertencia, y nos da detalles, para que de manera

instantánea notifiquemos esta anomalía a nuestro administrador de la red.

Figura # 3.14. Interfaz gráfica de Detección de DoS MAC Invalid – Estado Advertencia.

3.5 Desarrollo de la aplicación para el ataque DNS Hijacking.

3.5.1. Análisis de detección del ataque.

Para la detección de este ataque se necesita primero configurar dos archivos, los cuales son “dns_ip.conf” donde se almacenaran las direcciones ip que se desean monitorizar, y el otro archivo llamado “nombres_dns.conf” donde irán el nombre de los DNS para poder notificar. Estos son almacenados para poder verificar que no exista cambio de ip’s al momento de interactuar y saber si el sitio ha sido falsificado.

3.5.2. Arquitectura de desarrollo.

La arquitectura de este ataque si es diferente a la de los demás, la cual es la siguiente:

Figura # 3.15. Arquitectura de aplicación de detección de DNS Hijacking

En este caso lo que se realiza es una monitorización de las DNS existentes para ver si son alteradas por lo que se necesitará la librería Python-dnspython. El resto de la arquitectura es similar a las anteriores que permiten tener una captura de tráfico e ilustración de los eventos.

3.5.3. Diseño de la interfaz gráfica.

El diseño de esta interfaz es similar a las anteriores ya que tienen el mismo objetivo lo cual es determinar si se encuentra en estado normal o está siendo atacado, debido a esto la ventana en estado normal es la siguiente:

Figura # 3.16. Interfaz gráfica de Detección de DNS Hijacking – Estado Inicial

Cuando existe la detección de un DNS falso entonces saltara el mensaje en el recuadro junto con el icono de advertencia, esto

debido a mantener el mismo esquema en todo el conjunto de aplicaciones.

Figura # 3.17. Interfaz gráfica de Detección de DNS Hijacking – Estado Advertencia

CAPÍTULO 4

IMPLEMENTACIÓN Y RESULTADOS

4.1 Instalación y configuración del conjunto de aplicaciones.

Python 2.7.9

Para construir el conjunto de aplicaciones para la detección de ataques en redes IPv4/IPv6, fue necesario ayudarnos de diversas herramientas y aplicaciones; precederemos a detallar cuales fueron y su sencilla instalación y configuración de las mismas.

En primer lugar tenemos a Python, en su versión 2.7.9. Como ya describimos, esta fue nuestra principal aplicación para el desarrollo del software. Su instalación en el sistema operativo es muy sencillo, podemos usarlo en Windows, Mac OS, y en muchas distribuciones de Linux; inclusive para algunas ya viene instalado por defecto. Pero en nuestro caso que utilizamos la distribución de Ubuntu, solo necesitamos utilizar el siguiente comando en el terminal.

$ sudo apt-get install python2.7

Con esto, ya podemos comenzar a disfrutar de todas las bondades de Python, si necesitamos una versión de Python más actualizada solo en vez de 2.7, deberemos poner la 3. Sin embargo, muchas librerías aún no estaban migradas completamente a la nueva versión por lo que decidimos trabajar con la 2.7 que funciona muy bien.

Scapy

Otra de las herramientas que necesitaremos es Scapy. ¿Qué es y para qué nos sirve? Bueno pues Scapy es un script desarrollado en Python. Y nos sirve para poder generar paquetes, es inclusive capaz de realizar ataques. Trabaja con los principales protocolos de comunicación, si deseamos ver con detalles cuales son los protocolos con los que trabaja, solo debemos

poner el comando *ls().* Dentro de la línea de comandos de Scapy. Esto realmente nos es de gran ayuda ya que podemos analizar minuciosamente cada paquete y protocolo, lo que nos facilitará la tarea en nuestro objetivo de la detección de los diversos ataques. Para la instalación de Scapy lo único que necesitamos será descargarse desde la versión deseada, y ejecutar los siguientes comandos.

$ cd /tmp

$ wget scapy.net

$ unzip scapy-latest.zip

$ cd scapy-2.*

$ sudo python setup.py install

Tkinter

Tkinter es otra herramienta de la cual vamos a hacer mucho uso, no es más que el paquete GUI (Graphic User Interface) estándar de Python. No es la única librería de interfaz gráfica pero sin embargo es una de las más populares. Y para su instalación solo se necesita escribir.

$ sudo apt-get install python-tk

Inicio de scripts

Para ejecutar un script debemos ver primero si este tiene los permisos de ejecución, en Linux no se ejecutan los scripts de manera predeterminada; para fijarnos si un script tiene permisos de ejecución por medio de la línea de comandos, debemos escribir el comando.

$ ls -l nombre-del-script

Si este no posee, los permisos de ejecución procederemos a otorgárselos de la siguiente manera.

$ chmod +x nombre-del-script

Y ahora si con los permisos correspondientes, procederemos a ejecutarlo de la siguiente manera.

$ ./ nombre-del-script

4.2 Prueba de funcionalidad.

Una vez culminado el conjunto de aplicaciones, el siguiente paso es la prueba de funcionalidad con lo cual damos por cumplido nuestro objetivo,

para lo cual hubo la necesidad de usar una herramienta que pueda realizar los diversos ataques propuestos para nosotros poder detectarlos.

Para lograr estas pruebas se necesitó seleccionar el escenario para la prueba para lo cual usamos el que se muestra en la figura 34. Que consta de la máquina del atacante junto con el de la víctima, otra máquina que sirve como destino de conexión con la victima; además necesitamos un enrutador interconectado a un servidor DNS.

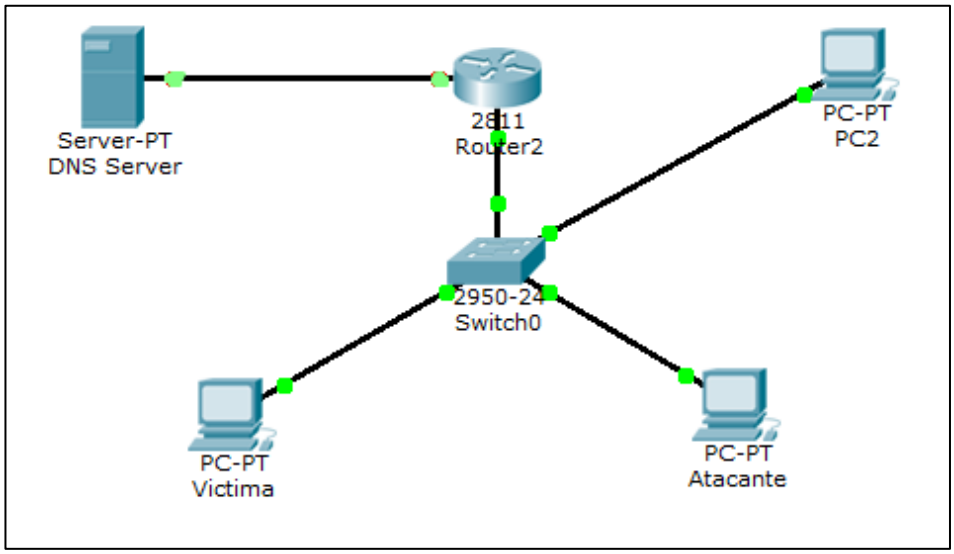


Figura # 4.1. Esquema para pruebas del aplicativo

La herramienta utilizada en nuestro caso se llama “Evil Foca” desarrollado por la empresa “Eleven Paths” la cual fue desarrollado para personas que

realizan test de penetración lo cual es un procedimiento metodológico y sistemático en el que se simula un ataque real a una red o sistema, con el fin de descubrir y reparar sus problemas de seguridad.

Esta herramienta es gratuita y puede realizar diversos ataques en redes IPv4 e IPv6 de manera fácil solo identificando los nodos que van a ser víctimas y ejecutar el ataque.

Figura # 4.2. Logo de la herramienta Evil Foca

Para el primer ataque llamado “Neighbor Advertisement Spoofing” se ejecuta la herramienta “Evil Foca”, nos colocamos en la pestaña con el nombre del ataque, luego seleccionamos a la víctima y a su puerta de enlace en esa red, y finalmente presionamos el botón “Start” con lo cual se iniciara el ataque.

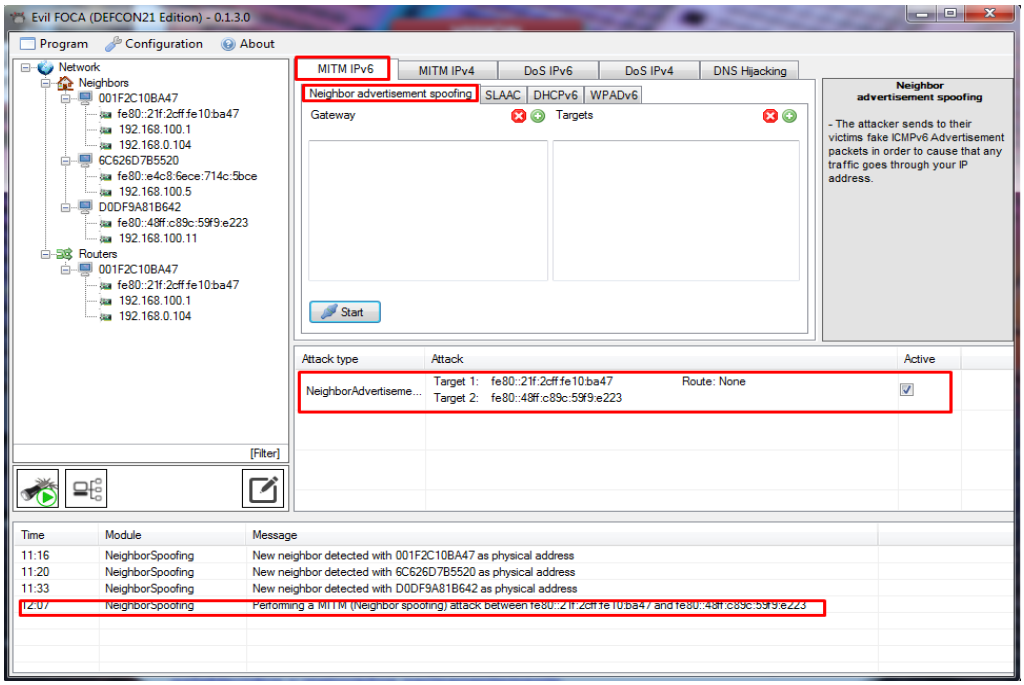


Figura # 4.3. Inicio del ataque “Neighbor Advertisement Spoofing”

Una vez iniciado el ataque ejecutamos nuestro aplicativo seleccionando el botón de detección de IPv6 y luego se abrirá una nueva ventana donde se presionara el botón “Start” para que se inicie la detección.

Figura # 4.4. Detección del ataque "Neighbor Advertisement Spoofing"

En la ventana una vez detectada la anomalía, este mostrara el símbolo de advertencia junto con un mensaje de indicando datos de la víctima y del atacante, junto al tipo de ataque que se realizó.

Para el segundo ataque es similar al primero, se selecciona n la pestaña de "MITM IPv4" junto con "ARP Spoofing", seleccionamos la víctima con su puerta de enlace, y finalmente presionamos el botón de "Start" para iniciar el ataque.

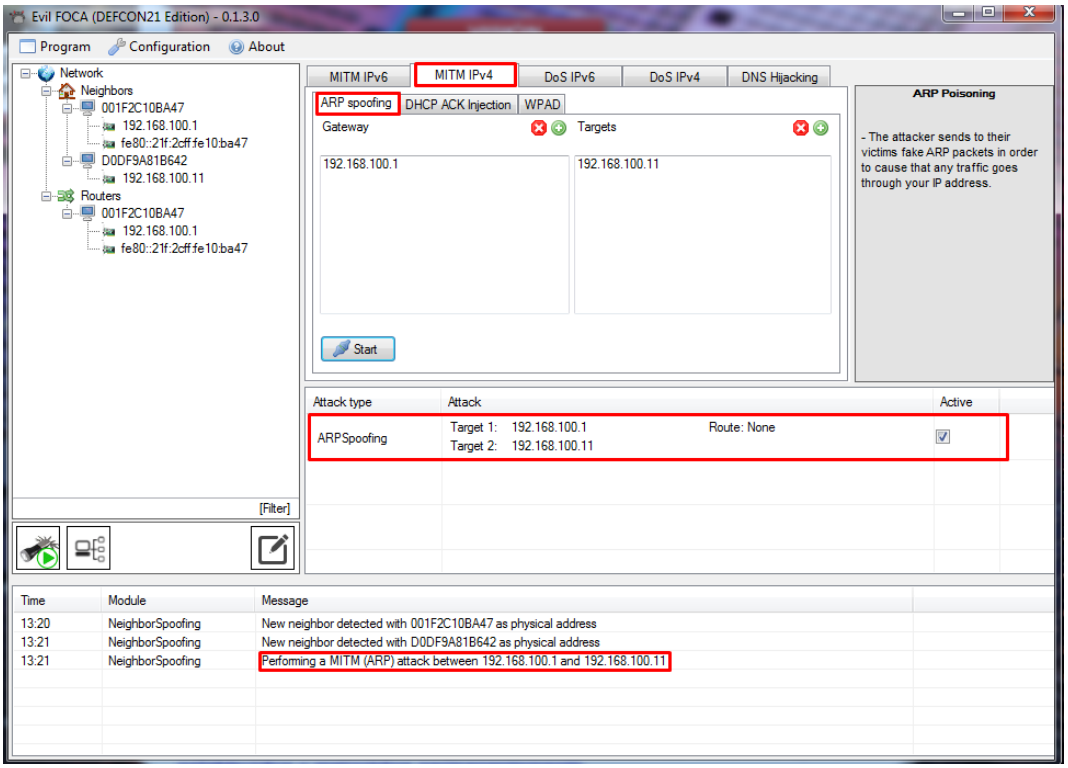


Figura # 4.5. Inicio de ataque “ARP Spoofing”

Ahora para realizar la detección, en nuestro aplicativo presionamos el botos de ARP Spoofing donde se abrirá una nueva ventana, la cual se deberá presionar el botón “Start” para iniciar el proceso de detección.

Figura # 4.6. Detección del ataque "ARP Spoofing"

Al momento de detectar el ataque este cambia al símbolo de advertencia y muestra el tipo de ataque junto a la dirección MAC del atacante.

Similar para el tercer ataque, en la herramienta escogemos la pestaña de "MITM IPv6" junto a la otra pestaña "SLAAC" la cual esta vez solo escogemos la IP versión 6 de la víctima y presionamos el botón "Start" para iniciar el ataque.

Luego en nuestro aplicativo presionamos el mismo botón del primer ataque debido a que ese aplicativo detecta ambos ataques de IPv6 realizados mostrando las mismas alertas pero distinto tipo de ataque.

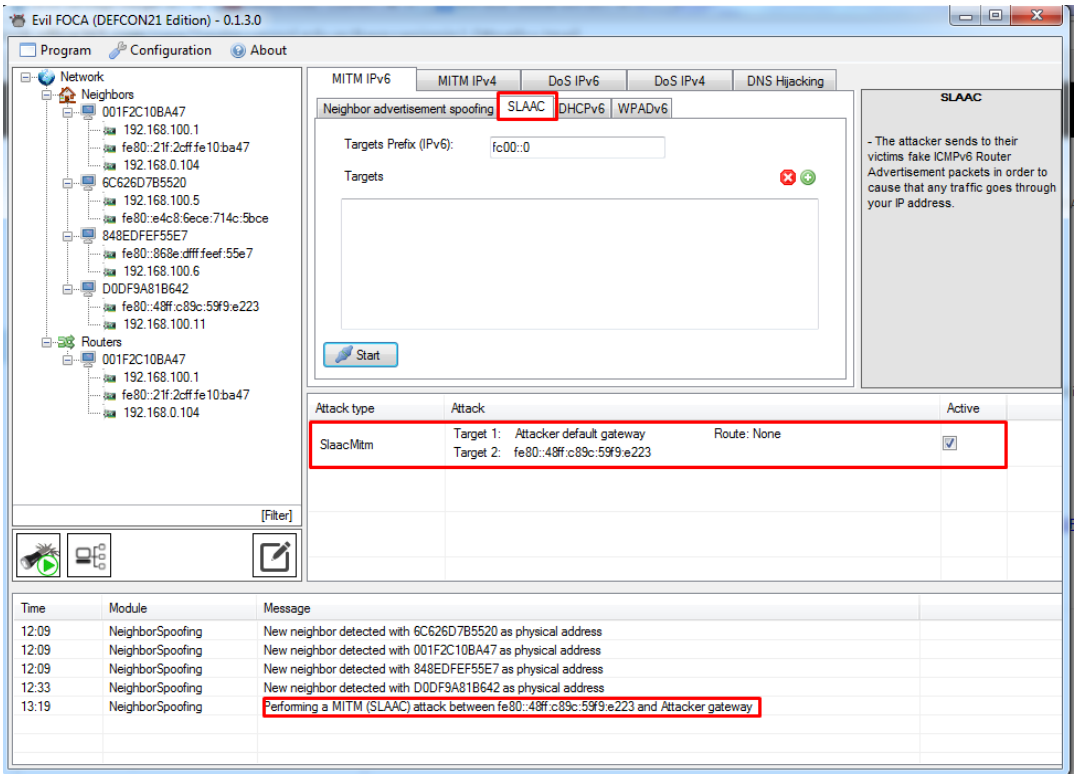


Figura # 4.7. Inicio de ataque "MITM con SLAAC"

Para el cuarto ataque seleccionaremos la pestaña de "DoS IPv4" junto con "Invalid MAC Spoofing" para lo cual seleccionaremos la IP de la víctima junto a la dirección IP con la que se va a perder comunicación, puede ser la del servidor o la de otro nodo.

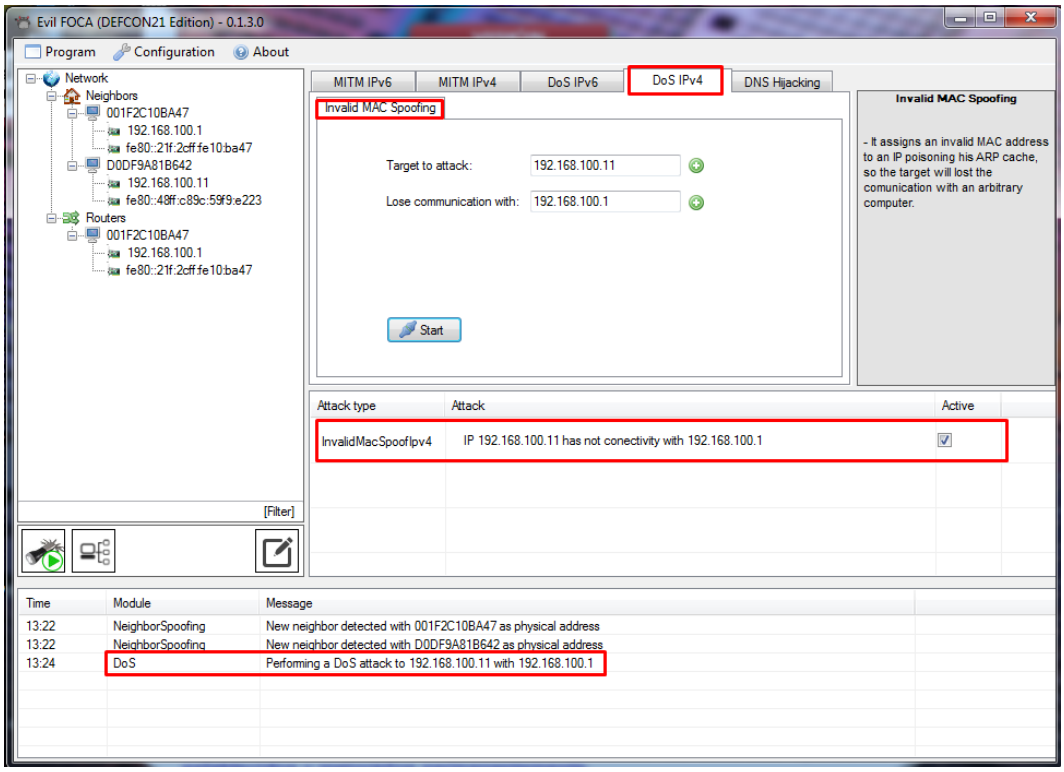


Figura # 4.8. Inicio de ataque "DoS con Invalid MAC Spoofing"

Una vez iniciado el ataque podemos ejecutar el aplicativo presionando el botón de "DoS Invalid MAC Spoofing" la cual abrirá una ventana que nos permitirá ejecutar la detección de dicho ataque al presionar el botón "Start" que se encuentra en la parte superior.

Figura # 4.9. Detección del ataque DoS Invalid MAC Spoofing

Cuando se encuentre el ataque realizado aparecerá en la tabla la información del ataque junto a la dirección MAC del atacante la cual es importante para saber de dónde se realizó.

Para el último ataque llamado “DNS Hijacking” el modo para realizar el ataque es diferente usando la herramienta. Primero se debe realizar un ataque MITM a la víctima, luego nos dirigimos a la pestaña con el mismo

nombre del ataque y ahí marcamos la opción “wildcard”, finalmente colocamos la dirección IP de cualquier página que deseemos utilizar para el ataque, en este caso se usó la de Google. Finalmente se presiona el botón “Start” y así inicia el ataque.

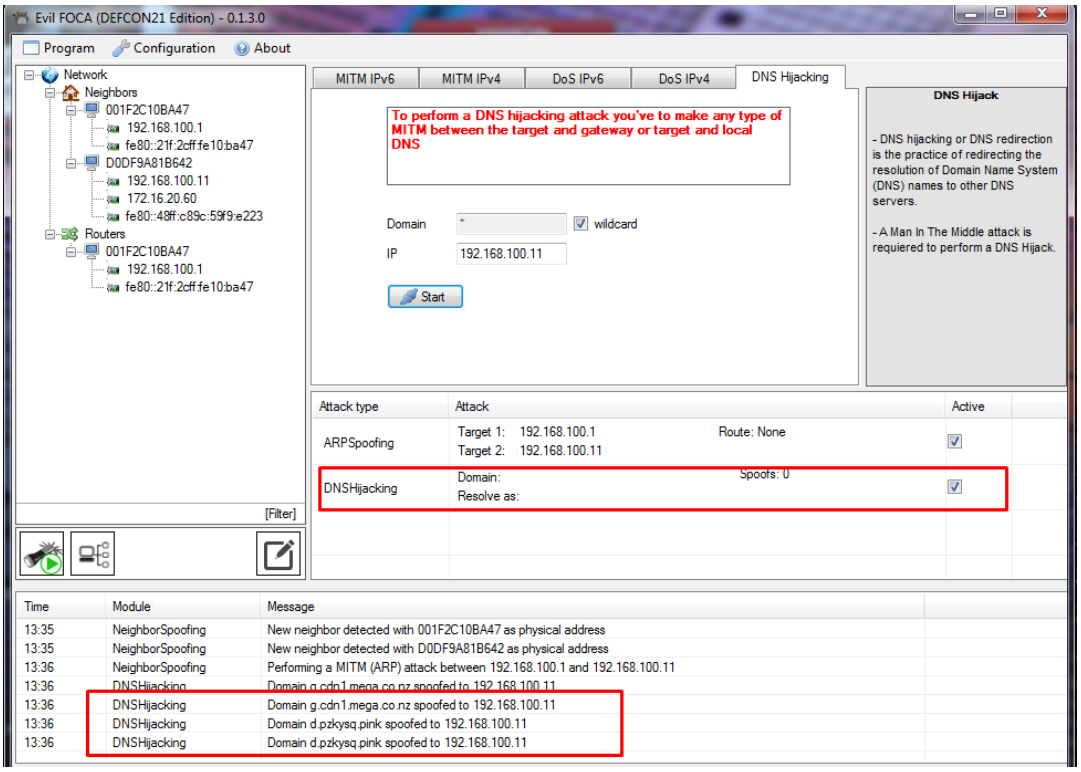


Figura # 4.10. Ejecución del ataque DNS Hijacking

Para la detección de este ataque necesitamos configurar un archivo con las direcciones IP que necesitamos monitorear para poder así detectarlas cuando exista algún cambio en ellas al estar navegándolas y saber si hemos sido atacados.

Figura # 4.11. Detección del ataque DNS Hijacking

Cuando se detecte el ataque el sistema mostrara el cambio de direcciones IP que tuvo la página por lo cual sabremos que fue atacada.

4.3 Análisis estadístico de resultados.

Para poder obtener una aprobación de nuestro proyecto se necesita realizar un estudio estadístico mediante los resultados que se dieron en las pruebas

para lo cual necesitaremos empezar calculando el número mínimo de observaciones.

La ecuación (4.1) nos permite calcular el número mínimo de observaciones que deben efectuarse; dónde n es el número mínimo de observaciones, W el rendimiento mínimo esperado, Z_β poder estadístico y Z_α nivel de confianza asignado.

$$n = \frac{W - W^2\,[Z_\beta + 1.4\,(Z_\alpha)]^2}{W^2} \qquad (4.1)$$

Para este proyecto necesitamos seleccionar un valor de Z_α para lo cual lo se seleccionara de la **Tabla 1** que se describe a continuación:

Tabla 1. Valores de Zα para diferentes niveles de confianza

NIVEL DE CONFIANZA (1-α)		
α	**%**	**Z_α**
0,050	95,0	1,960
0,025	97,5	2,240
0,010	99,0	2,576

Luego obtengo el valor de Z_β que representa el poder estadístico para lo cual usamos los datos de la **Tabla 2** que se encuentra a continuación:

Tabla 2. Valores de Z_β para diferentes niveles de poder estadístico

NIVEL DE CONFIANZA (1-β)		
β	**%**	$\mathbf{Z_\beta}$
0,20	80,0	0,842
0,15	85,0	1,036
0,10	90,0	1,282

Para nuestro proyecto hemos decidido tomar los siguientes valores: nivel de confianza (1-α) 99%, diferencia mínimo observable (W) 70% y Poder estadístico (1-β) 80%. Usando estos valores, la ecuación 2 queda de la siguiente manera:

$\mathbf{Z_\alpha}$ = 2,576 $\mathbf{Z_\beta}$ = 0,842 **W** = 0,70

$$n = \frac{0{,}70 - 0{,}70^2\,[0{,}842 + 1.4\,(2{,}576)]^2}{0{,}70^2} = 9 \qquad (4.2)$$

Tenemos como resultado que el número mínimo de observaciones debe ser 9 por tanto este será aplicado para cada una de las pruebas y posterior tabulación de datos.

Al tener esta información se empezó a realizar las respectivas pruebas tomando como datos el resultado de la detección de cada ataque descritas en el subcapítulo 4.2 para esto se tomó como referencia una respuesta

positiva o negativa en la detección y estos fueron los resultados en los nueve intentos.

Tabla 3. Resultados de frecuencias en pruebas exitosas o fallidas

ATAQUES	RESULTADOS		
	EXITOSOS	FALLIDOS	TOTAL
ARP SPOOFING	9	0	9
NEIGHBOR ADVERTISEMENT SPOOFING	8	1	9
DoS INVALID MAC SPOOFING	9	0	9
MITM SLAAC	8	1	9
DNS HIJACKING	7	2	9

Una vez que hemos tomado el número de observaciones suficientes procedemos a la aplicación de estadística descriptiva. En este caso tomaremos los datos de resultados exitosos los cuales nos interesan para poder dar por valido al proyecto.

Tabla 4. Resultados de estadística descriptiva

Media	**8,2**
Moda	**9**
Desviación estándar	**0,75**
Varianza de la muestra	**0,56**
Mínimo	**7**
Máximo	**9**

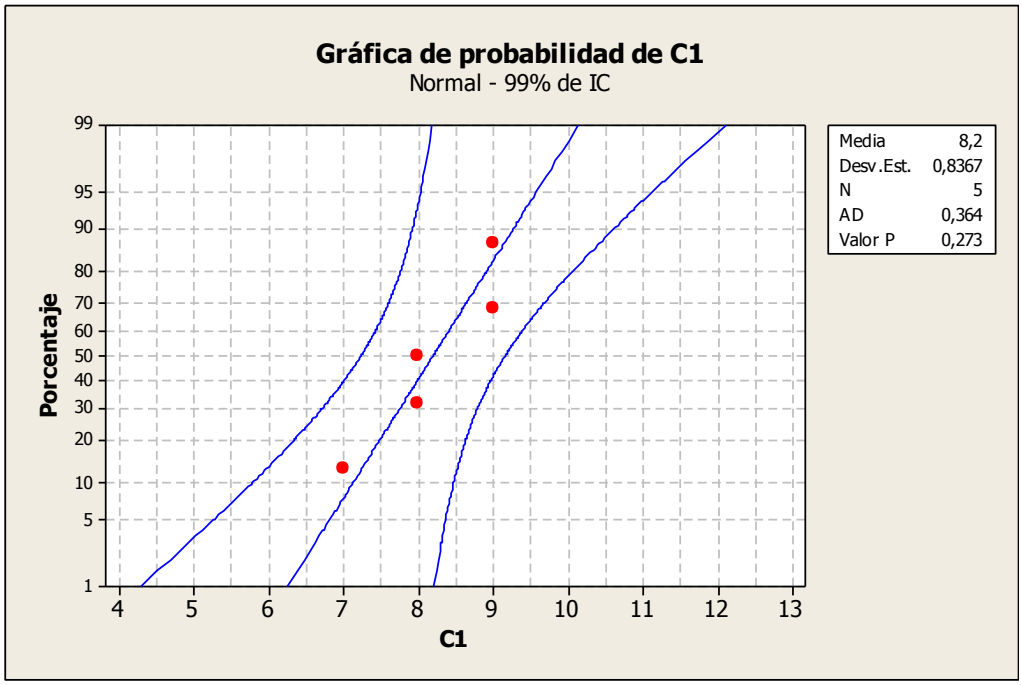


Figura # 4.12. Gráfica de datos de la estadística descriptiva

Con los datos obtenidos podemos darnos cuenta que la media es de 8,2 siendo esto un dato favorable para la aprobación del proyecto debido a que el resto de datos no sufren grandes variaciones lo cual nos permite concluir que las pruebas fueron exitosas.

4.4 Soluciones a los problemas presentados en las aplicaciones.

Al empezar a implementar el conjunto de aplicaciones primero realizamos el diseño de cómo se empezara a desarrollar, mientras se estaba

desarrollando empezamos a encontrar ciertos problemas los cuales fueron solucionados y los detallaremos a continuación.

a) **Problema:** Al momento de realizar este tipo de aplicaciones en Windows, utilizando el lenguaje de programación Python, algunas de las librerías nos dieron problemas en el momento de compilación por consiguiente algunos scripts que hemos detallado nos presenta ciertos problemas. Lo que nos dificulta el desarrollo de la aplicación, esto se debe a los permisos de varios de los scripts de desarrollo, así como el de la implementación de algunos comando de las librerías que usamos.

   Solución: Esto nos llevó a elegir el sistema operativo Ubuntu, además de ser de código abierto y de libre distribución, nos da la facilidad de manejar de una mejor manera todas las configuraciones que queramos.

b) **Problema:** Para la captura de los paquetes en la red, necesitamos los permisos correspondientes de “root”. Recordemos que el usuario “root” es el que administra el sistema, el que tiene todos los privilegios. Por este motivo al ejecutar el script sin estos permisos, no me permitía capturar los paquetes por lo cual el script dejaba de funcionar.

   Solución: La solución es iniciar sesión como “root”; en el sistema operativo Ubuntu, esta cuenta viene desactivada por defecto, por

motivos de seguridad y para activarla lo hacemos en la terminal, con el comando.

$ sudo su -

Luego nos pedirá la contraseña, la ponemos; y podremos disfrutar de los privilegios de ser “root”.

Otro método que nos permitía ejecutar los scripts es darle permisos de “root” a la librería “scapy” la cual es la que nos permite la captura de paquetes. Para realizar eso necesitamos seguir lo siguiente: primero debemos hacer una copia del “Python binary”.

$ sudo cp /usr/bin/python2.7 ~ / python_netraw

$ chmod -x, u + x ~ / python_netraw

$ sudo setcap CAP_NET_RAW = eip / usr / bin / python_netraw

CONCLUSIONES Y RECOMENDACIONES

CONCLUSIONES

1. Una vez que se ha implementado la aplicación, hemos quedado muy satisfechos, debido a que los resultados de las pruebas realizadas fueron exitosos. Hemos logrado implementar con éxito una aplicación desarrollada en Python, aprovechando sus librerías de código abierto y fácil manejo. Hemos sido capaces de detectar los cinco diferentes tipos de ataque, que anteriormente han sido especificados, entre estos estaban tres tipos diferentes de Man In The Middle, un DoS y un ataque al DNS,

dentro de una red LAN basada en IPv4 como en IPv6. Se ha trabajado y se ha conseguido con éxito darle una interfaz gráfica de fácil manejo y fácil entendimiento al usuario; a fin de que tanto la ejecución del programa, como la comprensión de la detección de ataques, sean de fácil entendimiento para todos aquellos que necesiten utilizar la aplicación.

RECOMENDACIONES

1. Se recomienda seguir con la actualización de este tipo de aplicaciones, debido a que a nivel global los ataques cada vez son más complejos y van mutando, cada día se detectan vulnerabilidades en nuestra red, y estas debilidades en nuestra seguridad informática son aprovechadas por otras personas para buscar un beneficio de ellas y salimos perjudicados nosotros. En este conjunto de aplicaciones solo hemos manejado el control de cinco diferentes tipos de ataques, pero sin embargo, se recomienda aumentar el número de detecciones de ataques, mantener actualizados los filtros que utilizamos para poder detectar los ataques, seguir aprovechando las funciones y librerías de Python, y compartirla para que cada vez más desarrolladores usen esta aplicación como base para otras más complejas.

BIBLIOGRAFÍA

[1] García Rambla, Juan Luis, Ataques en Redes de Datos en IPv4 e IPv6, España: 0xword 2nd Edition, 2013

[2] Verdejo Álvarez, Gabriel, Seguridades en Redes IP, 1st Edition España: Universidad Autónoma de Barcelona, 2003.

[3]CISCO SYSTEMS, Seguridad en Internet, : http://www.cisco.com/web/ES/solutions/es/internet_security/index.html, fecha de consulta octubre 2014

[4] 6DEPLOY, IPv6 Deployment and Support, http://www.6deploy.org/index.php?page=tutorials2, fecha de consulta septiembre 2014.

[5] CISCO SYSTEMS, IPv6 Basics, http://www.cisco.com/en/US/docs/voice_ip_comm/cucm/srnd/ipv6/basics.htm, fecha de consulta septiembre 2014.

[6] Lutz, Mark, Learning PYTHON, 4th Edition United States of America: Published by O'Reilly Media, Inc., 2009.

[7] CISCO SYSTEMS, IPv6 Headers, http://www.cisco.com/en/US/technologies/tk648/tk872/technologies_white_paper0900aecd80260042.pdf, fecha de consulta octubre 2014.

[8] ARIN, Arin IP Addressing Statistics, https://www.arin.net/knowledge/stats.pdf, fecha de consulta noviembre 2014.

[9] Seitz, Justin, Gray Hat Python, 1st Edition No Starch Press, 2009.

[10] O'Connor, TJ., Violent Python, 1st Edition United States of America: Elsevier, 2013.

[11] Van Rossum, Guido, El tutorial de Python, 1st Edition Argentina: Python Software Foundation, 2013.

[12] IPv6 MX, Fundamentos de IPv4, http://www.ipv6.mx/index.php/informacion/fundamentos/ipv4, fecha de consulta octubre 2014.

[13] Eleven Paths, Presentando Evil FOCA, http://blog.elevenpaths.com/2013/08/re-presentando-evil-foca-defcon-edition.html?q=Evil+Foca, fecha de consulta julio 2014.

[14] Bird, Steven, Natural Language Processing with Python, 1st Edition O'Reilly Media, 2009.