“IDS, IPS”

Trabajo Preparatorio N°10

Laboratorio de Seguridad en Redes

Melanny Dávila
Ingeniería en Telecomunicaciones
Facultad de Eléctrica y Eléctronica
Quito, Ecuador
melanny.davila@epn.edu.ec

Alejandra Silva
Ingeniería en Telecomunicaciones
Facultad de Eléctrica y Eléctronica
Quito, Ecuador
alejandra.silva@epn.edu.ec

***Abstract*—En el siguiente documento se presenta el fundamento teórico acerca de los sistemas detección y prevención de intrusos en una red, con el fin de poder tomar contramedidas contra posibles acciones maliciosas.**

***Index Terms*—IDS, IPS, detección, HTTP, TCP.**

I. Introducción

Entre las diferentes aplicaciones de software que se utilizan para la detección de accesos no autorizados en dispositivos o en una red se tiene a los IDS, lo realiza mediante una alerta o log. mientras que los IPS a diferencia del anterior son sistemas de prevención de intrusos en un dispositivo monitorizando las actividades a nivel de capa red (3) y a nivel de capa de aplicación (7) con el objetivo de identificar compartimientos sospechosos o maliciosas permitiendo así una ejecución mucho mas rápida del administrador de la red mediante contingencia.

Esto se va a realizar mediante la implementación de detección de intrusos por medio de Snort, que es la prevención de intrusos de código y usa esas reglas para encontrar paquetes que coincidan con ellas y genera alertas para los usuarios.

II. Objetivos

- Reforzar los conocimientos relativos a mecanismos de defensa de redes y el uso de sistemas de detección de intrusos.
- Implementar un sistema de detección de intrusos por medio de Snort.

III. Cuestionario

A. Revisar el marco teórico para la realización de la práctica.

B. Consultar los operadores de control lógicos (||, &&, !) más utilizados en bash – Linux y describa la funcionalidad de estos (máximo media carilla).

- O lógico (||): Se utiliza para crear listas OR, permite ejecutar un comando solo si otro salió sin éxito [1].

```
command1 || command2
```

  En este ejemplo, command2solo se ejecutará si command1falla (si devolvió un estado de salida distinto de 0). Ambos comandos se ejecutan en primer plano. Este comando también se puede escribir como:

```
if command1
then true
else command2
fi
```

- Y lógico (&&): Se utiliza para crear listas AND, permite ejecutar un comando solo si otro salió con éxito.

```
command1 && command2
```

  En este ejemplo, command2 se ejecutará después de que command1 haya terminado y solo si command1 fue exitoso (si su código de salida fue 0). Ambos comandos se ejecutan en primer plano [1]. Este comando también se puede escribir como:

```
if command1
then command2
else false
fi
```

- No (!): Esta es una palabra reservada que actúa como el operador ”no” (pero debe tener un delimitador), utilizada para negar el estado de retorno de un comando [1].

C. Consulte como hacer peticiones HTTP utilizando netcat (máximo una carilla).

La mayoría del tráfico de Internet es tráfico HTTP. Netcat es la utilidad que se usa para casi cualquier secuencia de datos que involucre TCP o UDP. Puede abrir conexiones TCP, enviar paquetes UDP, escuchar en puertos TCP y UDP arbitrarios, escanear puertos y manejar tanto IPv4 como IPv6 [2].

La comunicación HTTP ocurre a través de una conexión TCP. Entonces al crear una conexión TCP con el servidor e intentar obtener una respuesta de él. Así, se creerá una conexión TCP y conectarse a ella usando netcat [2].

```
netcat localhost 'nro. del puerto'
```

El comando, junto con la creación de una conexión TCP, también abrirá un STDIN. Todo lo que pase en ese flujo de entrada llegará al servidor a través de la conexión.

A veces se desea/tiene que enviar solicitudes HTTP específicas, lo cual puede hacerse fácilmente con curl o simplemente escribir la solicitud uno mismo [3].

- Realizar una solicitud HTTP con NC

```
nc example.com 80
GET / HTTP/1.1
Host: example.com
```

- Realizar una solicitud HTTP con TELNET

```
telnet example.com 80
GET / HTTP/1.1
Host: example.com
```

- Realizar una solicitud HTTP con OpenSSL

```
openssl s_client -connect example.com:443
GET / HTTP/1.1
Host: example.com
```

- Realizar una solicitud HTTP especificando más cabeceras.

```
nc example.com 80
GET /foo.html HTTP/1.1
Host: example.com
Accept-Encoding: gzip
Accept: text/html
```

D. Consulte la sintaxis de reglas y acciones en Snort (máximo una carilla).

Una de las características mas importantes para el correcto funcionamiento de Snort, es sin lugar a dudas, las reglas de detección y alarmas, Snort cuenta con un pequeño lenguaje de definición de reglas que sigue una sintaxis definida para diseñar reglas robustas y útiles para un pentester y/o analista en seguridad informática. Las reglas en Snort se separan en dos secciones, en primera instancia esta el encabezado de la regla y luego se definen las opciones de la regla, el encabezado de la regla contiene información relacionada a la acción que se debe tomar (por ejemplo la acción de generar una alerta) al igual que direcciones IP, puertos y mascaras de red del origen y destino del paquete, por otro lado la sección de opciones de la regla determinan diversos filtros que le dan dinamismo a la regla, de esta forma solamente se aplicará cuando dichas opciones o filtros se cumplan, por lo tanto es de vital importancia diseñar correctamente esta sección de la regla para evitar falsos positivos. [4]

En Snort, todas las reglas deben ser escritas en una sola linea, ya que el procesador de reglas no entiende los saltos de linea. [4]

Un ejemplo de regla:

```
alert tcp any any -> 192.168.1.0/24 111
( content:``|00 01 86 a5|''; msg ``mountd access'
  '; )
```

- Elementos antes paréntesis comprenden el encabezado de la regla.
- Elementos dentro paréntesis son las opciones de la regla:
  -Palabras antes : son keywords
  -Esta sección no es forzosa para todas las reglas

Encabezado de la regla [5] Contiene información que define:

- El quien, donde y que de un paquete.
- La accion a tomar si un paquete cumple con la regla

Formato:

```
accion protocolo IP puerto -> IP puerto
```

El primer elemento en una regla es la acción de la regla. Se tiene definidas acciones por default.

Acciones por default: [5]

- **alert:** Genera una alerta usando el método de alerta seleccionado y envía a una bitácora el paquete.
- **log:** Envía a una bitácora el paquete
- **pass:** Ignora el paquete.
- **active:** Alerta y activa otra regla dinámica.
- **dynamic:** Permanecer inactiva hasta activarse por otra regla, entonces actuar como una regla tipo log.

Protocolos: [5] Existen cuatro protocolos que snort analiza buscando un comportamiento sospechoso que son:

- TCP
- UDP
- ICMP
- IP

Direcciones IP: [5]

- Información dirección IP entrada y salida.
- Palabra any puede definir cualquier dirección.
- Operador negación !, cualquier dirección excepto la especificada.
- Formadas por dirección numérica IP y bloque CIDR
  - Bloque CIDR indica el netmask que debe aplicarse a las direcciones de las reglas
  - Cada paquete que entra es probado contra la regla
  - /24: red clase C, /16 red clase B, /32 dirección específica.

Ejemplo dirección con bloque CIDR [5]

```
192.168.1.0/24
```

Bloque de direcciones 192.168.1.0 a 192.168.1.255.

Ejemplo negación

```
alert tcp !192.168.1.0/24 any -> 192.168.1.0/24 111
    \
( content: ``|00 01 86 a5|''; msg:``peligro''; )
```

Operador de dirección: [5]

- Indica la orientación o dirección del trafico de la regla que se esta aplicando.
- Las direcciones y puertos del lado izquierdo corresponden a tráfico proveniente del host origen y del lazo derecho al host destino
- Existe un operador bidireccional ¡¿

Opciones de reglas: [5]

- Forman el corazón de la máquina de detección de intrusión.
- Todas las opciones están separadas por el carácter ;
- Las keywords de las opciones están separadas por el carácter :
- Cuenta con 35 keywords.

Aclaración de algunos ficheros: [5]

- **msg:**
  - Indican a la maquina de alerta y bitácoras el mensaje a imprimir con el paquete.

Keyword	Significado
msg	imprime un mensaje en las alertas y bitácoras
logto	envía paquete a archivo usuario en lugar archivo usual
ttl, tos, id	prueba los campos encabezado IP por un valor en específico
flags	prueba los valores de las banderas TCP
rpc	prueba
priority	identificador de severidad de la regla
sameip	verifica de la dirección fuente es igual a la destino
content	busca por un patrón dentro del payload
seq,ack	verifica en el campo TCP por un valor específico

Fig. 1. Keywords.

– Carácter “ ” para indicar carácter que pueda coincidir con las reglas de snort.
– Formato: msg: “¡texto mensaje¿”
- Ejemplo:

```
alert tcp any any -> any 21 ( msg:``ataque sobre
    servidor ftp''
```

- **reference:**
  - Referencia para sistemas externos de identificación de ataques.
- **sid:**
  - Identificación única de reglas de snort.
- **rev:**
  - Identificación de forma única de revisiones
- **classtype:**
  - Clasifica alertas para formar clases de ataques.
- **priority:**
  - Asigna niveles de severidad a las reglas

REFERENCES

[1] “¿Cuáles son los operadores de control y redirección de shell?” https://qastack.mx/unix/159513/what-are-the-shells-control-and-redirection-operators (accedido ago. 17, 2021).

[2] A. Bhayani, “HTTP Requests - The Hard Way with Netcat — Codementor”. https://www.codementor.io/@arpitbhayani/http-requests-the-hard-way-with-netcat-5v0b1p5hg (accedido ago. 17, 2021).

[3] “How to make http/https requests yourself with nc/telnet/openssl”. https://makandracards.com/makandra/45025-how-to-make-http-https-requests-yourself-with-nc-telnet-openssl (accedido ago. 17, 2021).

[4] Adastra (2011). ”Utilizando RuleSets en Snort para Detección de Amenazas y Generación de Alarmas – Parte I”. Recuperado de: https://thehackerway.com/2011/07/20/utilizando-rulesets-en-snort-para-deteccion-de-amenazas-y-generacion-de-alarmas-parte-i/. accedido (16,Agos,2021).

[5] Roberto. G. (1998)”Snort”. Recuperado de: http://cryptomex.org/SlidesSeguridad/Herra3-snort.pdf. accedido (16,Agos,2021).