Configuración de una DMZ

Sistemas Confiables
Máster Universitario en Investigación en Ciberseguridad

Se pretende construir una DMZ (utilizando contenedores Docker para simular las máquinas) siguiendo la topología de red de la Figura 1 y atendiendo los siguientes criterios:

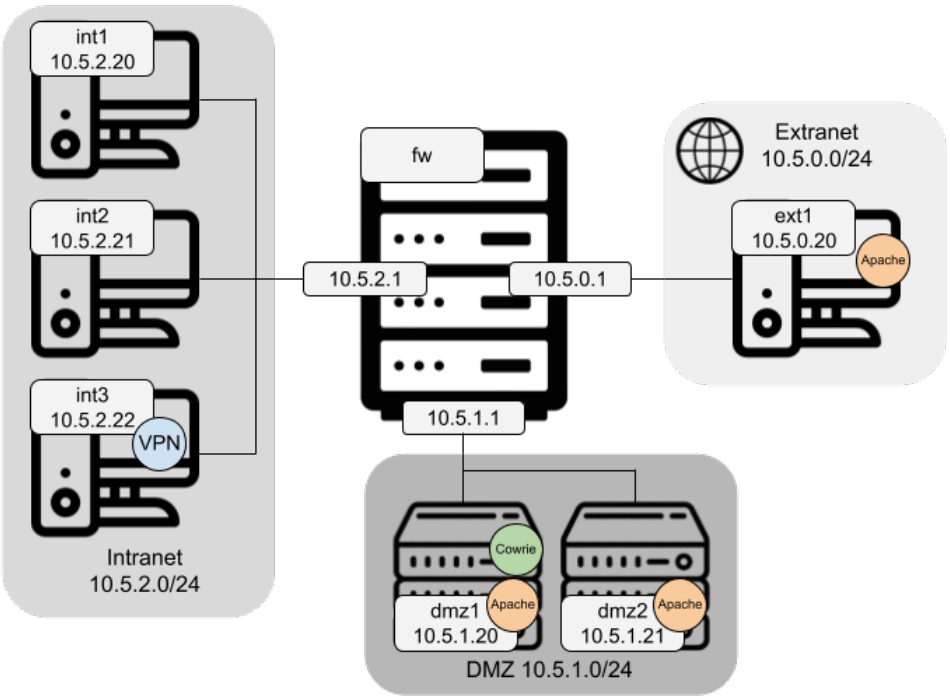

Figura 1: Arquitectura propuesta.

Entregables

Con respecto a la entrega, han de tenerse en cuenta las siguientes indicaciones generales:

- Es **obligatorio** grabar uno o varios vídeos que permitan visualizar el funcionamiento global del sistema en los términos descritos en cada ejercicio. Las pruebas a realizar están descritas en el guión (etiquetas desde T_1 a T_{23}). La duración acumulada de los vídeos no debe exceder los **10 minutos (una mayor duración puede conllevar penalizaciones)**. Si el tamaño del vídeo excede los 50MB será necesario compartirlo con el profesor indicando en el fichero de manifiesto explicado más adelante la url para acceder al mismo.
- La entrega debe contener todos los scripts, ficheros Dockerfile y docker-compose.yml, otros ficheros de configuración, etc. editados o creados en cada ejercicio debidamente etiquetados.
- También se incluirá un fichero de manifiesto (README) indicando vuestro nombre y apellidos, los ficheros que conforman la entrega y su contenido. También de debe incluir en este archivo la salida de los comandos `ip address` e `ip route` en todas las máquinas. Si es necesario, también pueden incluirse indicaciones para probar la práctica.En cualquier caso, estas indicaciones deben de ser breves y concisas, se trata de un fichero de manifiesto, **no de una memoria**.

Pautas generales

Ten en cuenta las siguientes pautas generales antes de empezar a trabajar:

- **Todas** las máquina deben tener el servicio SSH instalado con usuario y contraseña `root:root`.
- Para poder ejecutar reglas iptables en el contenedor es necesario instalar los paquetes `iptables` y `pkg-config`. Además, es necesario iniciar el contenedor con permisos especiales (`--privileged`, o `--cap-add=NET_ADMIN` y `--cap-add=NET_RAW`).
- Para poder ejecutar ping y otros comandos para chequear las comunicaciones es necesario instalar los siguientes paquetes en el contenedor: `net-tools`, `iputils-ping`, `traceroute`, `iproute2`, `tcpdump` y `nmap`.
- La ejecución de las pruebas es secuencial de la T_1 a la T_{23}. A medida que se ejecutan, hay que ir introduciendo reglas iptables. Para simplificar la ejecución de las pruebas, lo más sencillo es escribir las reglas iptables en scripts y guardarlos en los contenedores que correspondan. Así podrán ser ejecutados en el orden adecuado (desde el propio contenedor con `docker-compose exec`) sin tener que parar, re-build e iniciar de nuevo el contenedor.

1. Configuración de *fw*

fw contará con tres interfaces de red. Una, con conexión a la red externa (extranet); otra, con conexión a la DMZ; y la última, con conexión a la red interna (intranet).

- La dirección de la toma de red externa será 10.5.0.1. La red externa, que simulará una red WAN, será la subred 10.5.0.0/24.
- La dirección de la toma de red hacia la DMZ será 10.5.1.1. La DMZ será la subred 10.5.1.0/24.
- La dirección de la toma de red interna será 10.5.2.1. La Intranet será la subred 10.5.2.0/24.

Configuración inicial de las comunicaciones a través de *fw*

fw tiene que encaminar todo el tráfico procedente de la Intranet e impedir que cualquier otro tráfico penetre en ella. A su vez, deber permitir conexiones hacia *dmz1* y *dmz2* desde cualquier sitio, incluida la intranet. Configura las comunicaciones en *fw* siguiendo las siguientes consideraciones:

- *fw* debe ser capaz de encaminar tráfico, por tanto, es necesario activar el bit de forwarding.
- Cambiar la política por defecto de las cadenas INPUT y FORWARD para que descarten los paquetes.
- Cambiar la política de OUTPUT para que permita pasar todos los paquetes.
- Permitir el tráfico entrante a través de la interfaz de loopback.
- Permitir el tráfico entrante correspondiente a conexiones establecidas y/o relacionadas.
- Permitir consultas entrantes de tipo ICMP ECHO REQUEST.
- Asegúrate que los cambios anteriores persisten en caso de parar y arrancar *fw*. Puedes guardar las reglas y la configuración del bit de forwarding en un script que se ejecute al inicial el contenedor. [1].

Para probar la configuración anterior realiza las siguientes pruebas:

T_1 Usando nmap en *ext1*, *dmz1* e *int1*, comprueba que *fw* no ofrece ningún acceso desde ninguna de las tres redes. (nmap -sS fw).

```
$ nmap -sS fw
```

T_2 Comprobar que *fw* responde a las peticiones ICMP ECHO REQUEST desde las tres redes (ping fw).

```
$ ping 10.5.0.1    # desde external network
$ ping 10.5.1.1    # desde DMZ
$ ping 10.5.2.1    # desde int1
```

[1] En realidad, en Docker el bit está activo por defecto, pero para tomar conciencia se pide activarlo explícitamente.

2. Configuración de *ext1*

ext1 contará con una única interfaz de red conectada a la Extranet. Debe tener instalado un servidor web (apache2) accesible desde la Intranet. Crea un sitio web de prueba que contenga un fichero HTML estático con un mensaje (e.j. "Practica 1 Sistemas Confiables - Este servidor está en Extranet"). La dirección de la interfaz de red será fija; en concreto, 10.5.0.20.

3. Configuración de *int1* e *int2*

int1 e *int2* contarán con una única interfaz de red conectada a la Intranet. La dirección de las interfaces de red será fija; en concreto, 10.5.2.20 para *int1*, y 10.5.2.21 para *int2*.

Comunicaciones desde la red interna

El primer paso es configurar el acceso desde la red interna a la red externa y al resto del mundo. Suponiendo que no se desea restringir demasiado este acceso, se debe permitir el siguiente tráfico a través de de la cadena FORWARD:

- Todo el tráfico de conexiones establecidas o relacionadas para los protocolos TCP, UDP e ICMP.
- Todo el tráfico que entre en *fw* a través de la interfaz con la red interna, con protocolo TCP, UDP o ICMP, con dirección IP origen en el rango 10.5.2.0/24, y que vaya a salir a través de la interfaz con la red externa.

Para comprobar la configuración anterior realiza las siguientes pruebas:

T_3 Comprueba la conectividad de int1 con ext1 mediante ping.

T_4 Usando wireshark, tshark, tcpdump o similar; identifica los paquetes intercambiados.

T_5 Comprueba la conectividad de int1 con ext1 usando SSH.

A continuación, utilizaremos SNAT para ocultar la estructura interna de la red interna. La configuración de SNAT debe cumplir los siguientes requisitos:

- Todos los paquetes que abandonen *fw* por la interfaz externa y que provengan de la red interna (10.5.2.0/24) deben cambiar su IP de origen para que sea la de *fw* en esa interfaz (10.5.0.1).
- Puesto que la IP 10.5.0.1 es fija, es conveniente usar SNAT y no MASQUERADE. El motivo es que así es más eficiente, ya que MASQUERADE debe consultar la IP de la interfaz cada vez que procesa un paquete.

Después de realizar los cambios anteriores, realiza las siguientes pruebas:

T_6 Comprueba la conectividad de *int1* con *ext1* usando ping.

T_7 Usando wireshark, tshark, tcpdump o similar; identifica los paquetes intercambiados, comprobando que la IP de origen del paquete que llega a *ext1* es la de la interfaz externa de *fw*.

T_8 Comprueba la conectividad de *int1* con *ext1* usando SSH.

T_9 Comprueba la conectividad HTTP de *int1* con *ext1*:

```
$ wget 10.5.0.20
```

4. Configuración de *dmz1* y *dmz2*

dmz1 y *dmz2* contarán con una única interfaz de red conectada a la DMZ. Ambas máquinas deben tener instalado un servidor web (apache2) accesible tanto desde la DMZ, como desde la Intranet y la red externa. La dirección de la interfaz de red será fija; en concreto, 10.5.1.20 (*dmz1*) y 10.5.1.21 (*dmz2*).

En cada máquina crea un sitio web de prueba que contenga un fichero HTML estático con un mensaje (e.j. "Practica 1 Sistemas Confiables - Este servidor está en la máquina dmz1 (o dmz2, respectivamente)").

Comunicaciones en la DMZ

Es necesario agregar reglas para permitir el acceso desde y hacia la DMZ. En particular, es necesario tener en cuenta las siguientes indicaciones generales:

- Los nodos de esta red actúan como nodos bastión y debe limitarse el acceso a ellos tanto desde Internet como desde la red interna. Es necesario permitir el acceso a los servicios que ofrecen y, en su caso, permitir el acceso desde las máquinas de la red interna que tengan tareas administrativas.
- Estos nodos están necesariamente expuestos a ataques desde Internet (pues tienen que ofrecer servicios), por tanto, es necesario proteger a los nodos de la red interna de cualquier tipo de acceso no deseado desde los nodos bastión. De esta forma, en caso de que un nodo bastión sea comprometido, se podrá detectar y tratar el incidente antes de que afecte a la red corporativa.

En nuestro caso, hay que modificar la configuración de *fw* para que permita los siguientes tipos de acceso:

- Acceso TCP desde cualquier máquina de la red interna y externa a las máquinas *dmz1* y *dmz2*, exclusivamente al servicio HTTP (puerto 80).
- Acceso SSH *int1* a las máquinas *dmz1* y *dmz2* para llevar a cabo labores de administración.
- Prevenir un ataque de DoS a la máquina *fw* limitando el numero de conexiones por minuto de tipo ICMP. *Establece un limite bajo para poder realizar las pruebas.* Esta regla de iptables sustituye a la definida en la Sección 1 en la cual permitíamos las consultas entrantes de tipo ICMP ECHO REQUEST. Además, es necesario especificar los protocolos TCP y UDP en las conexiones entrantes establecidas y/o relacionadas. (*Estos cambios no afectan al resto de Tareas que hayais realizado anteriormente*).

Después de realizar los cambios anteriores, realiza las siguientes pruebas:

T_{10} Comprueba que es posible acceder al servicio HTTP desde la red interna:

```
$ wget 10.5.1.20
```

T_{11} Comprueba, con nmap, que no es posible acceder a ningún puerto más, tanto desde la red interna como desde la externa. En la red interna, prueba desde *int1* e *int2*, ya que desde *int1* sí está permitido el acceso SSH:

```
$ nmap -sS 10.5.1.20
```

T_{12} Comprueba que es posible conectarse por SSH desde *int1* a *dmz1* y *dmz2*.

T_{13} Comprueba que desde *dmz1* y *dmz2* no es posible acceder a ningún puerto ni de *int1*, ni de *int2*.

T_{14} Comprueba desde *ext* que se limitan el numero de conexiones ICMP a la máquina *fw*, para ello haz uso de hping3 y envía 1000 paquetes por minuto.

```
$ hping3 --icmp 10.5.0.1 -i u1000
```

5. Nuevos servicios

Se pide introducir cuatro nuevos servicios en la DMZ: primero, habrá que bastionar las conexiones SSH con los equipos de la DMZ; después, habrá que adaptar *dmz1* y *dmz2* para que además de servir HTTP, sirvan HTTPS; además, se instalara un servidor VPN en una nueva máquina para establecer conexiones con la red interna. Y se realizará la instalación de una honeypot.

Se recomienda trabajar en dos fases:

1. Para cada uno de los servicios crea una imagen Docker basada en Ubuntu o similar que permita desplegar un contenedor para dar el servicio correspondiente. Puedes probarlo desde el host.
2. Una vez tengas el servicio funcionando, intégralo en el ecosistema creado con Docker Compose.

5.1. Hardening OpenSSH server

Los servidores OpenSSH de la DMZ tienen que soportar las siguientes funcionalidades:

- El servidor debe escuchar peticiones en el puerto 2222.
- No debe permitirse la conexión con el usuario root. Habrá que crear al menos usuario sin privilegios en el contenedor para establecer las conexiones.
- Se permitirá la autentificación mediante contraseña siempre y cuando esta no sea vacía.
- También se permitirá el login sin contraseña mediante clave pública. Habrá que cargar las claves del usuario/s en el propio contenedor.
- El número máximo de intentos de autentificación será 2.
- Debe mostrarse un *banner* personalizado con los datos del alumno cuando se establezca una conexión.
- Es necesario implementar las siguientes medidas de seguridad adicionales.
  - Usar fail2ban para detectar ataques de fuerza bruta y bloquear direcciones IP.
  - Instalar un sistema de autentificación multifactor.

T_{15} Realiza las pruebas que consideres necesarias para poder visualizar la nueva funcionalidad en el/los vídeo/s.

5.2. Servicio HTTPS

- Modifica la configuración de Apache para poder servir HTTPS. Y configura nuevas reglas iptables que permitan el acceso al puerto 443.

T_{16} Realiza las pruebas que consideres necesarias para poder visualizar la nueva funcionalidad en el/los vídeo/s.

5.3. Servidor VPN

El nuevo servidor lo instalaremos en *int3* y dará servicio de conexión a una VPN. Para ello instalaremos un servidor VPN (a elección por el alumno) que permita a los nodos de la red externa establecer conexiones con *int3* para conectarse a la misma red que los nodos de la interna y poder realizar el mismo tipo de conexiones que las máquina de la red interna (*p.e. Acesso SSH a las máquinas de la DMZ)*).

T_{17} Realiza las pruebas que consideres necesarias para poder visualizar la nueva funcionalidad en el/los vídeo/s.

5.4. Honeypot Cowrie

En este ejercicio instalaremos y configuraremos la honeypot Cowrie. La instalación se realizará en la maquina *dmz1*. Ya que simula un servicio que una organización podría tener expuesto.

Cowrie se trata de un honeypot de media interacción que simula el servicio SSH. Para instalar Cowrie, sigue los pasos indicados en la documentación oficial [2]. Cabe destacar que Cowrie necesita ser ejecutado por un **usuario sin privilegios**.

Cowrie, por defecto, dispone de un sistema de ficheros debian 5.0 simulado, con el que el atacante puede interactuar. Es posible crear nuestro propio sistema de ficheros, configurando los usuarios y los archivos que nos interesen. Para ello sería necesario crear un nuevo sistema de ficheros base en el que crear el sistema que realizar la copia que se va a utilizar en el honeypot. En nuestro caso trabajaremos con la configuración por defecto de Cowrie, pero llevaremos a cabo algunas modificaciones que hagan el sistema más real. En el archivo de configuración de Cowrie será necesario:

T_{18} Modificar el hostname de la máquina con vuestro nombre.

T_{19} Modificar el archivo userdb.txt de cowrie, en el introducimos un usuario con vuestro nombre y una password con la que podréis entrar posteriormente a la honeypot.

La ejecución de Cowrie realmente no expone el puerto 22, si no que expone el puerto 2222 que es un puerto que no está reservado. Por ello, es necesario modificar la configuración de las máquinas *fw* y *dmz1*, para que permitan:

[2] https://cowrie.readthedocs.io/en/latest/

- Máquina *fw*:
  - Permitir acceso al puerto 22 de la maquina *dmz1* desde la red externa e interna (la maquina *dmz1* por defecto tiene el servicio 22 para llevar a cabo tareas de gestión que se deben realizar desde la red interna).
- Máquina *dmz1*: Permitir las conexiones al puerto 22 procedentes de la red externa se deben encaminar al puerto 2222 en el que realmente escucha la honeypot.

Para probar que las configuraciones se han realizado correctamente, realiza las siguientes pruebas:

T_{20} Comprueba que puertos tiene expuestos la maquina *dmz1* con el comando nmap, desde la maquina *ext1*.

```
nmap -sS 10.5.1.20 -Pn
```

T_{21} Prueba a conectarte a la honeypot desde la maquina de la red externa (ext1) usando el servicio que hay expuesto en el puerto 22 con el usuario y contraseña que has introducido en el archivo userdb.txt.

T_{22} Comprueba que se generan los logs correspondientes a la conexión que has hecho en la honeypot de la máquina *dmz1*. (cowrie/var/log/cowrie/cowrie.json)

T_{23} Prueba a conectarte al servicio SSH real de la maquina *dmz1* desde la maquina de la red interna (int1).

6. Ayuda

Para implementar algunas de las funcionalidades anteriores puede que necesitéis echarle un ojo a la siguiente lista. Contiene comentarios, enlaces a la documentación oficial de Docker y a otros tutoriales que permiten solventar algunos de los problemas que os podéis encontrar.

1. Arrancar varios servicios en un mismo contenedor [1]. Entre las soluciones planteadas en [1], está el uso de *Supervisord*. En [2] tenéis un tutorial rápido sobre Supervisord.
2. Para *fail2ban* es necesario conceder al contenedor permisos especiales al arrancar que permitan definir las reglas iptables que utiliza la herramienta. Para ello basta incluir las siguientes opciones al iniciar el contenedor: `--privileged`[3], o bien `--cap-add=NET_ATMIN --cap-add=NET_RAW`.
3. Es necesario instalar el demonio *rsyslogd* en el contenedor para utilizar *fail2ban*, ya que el log por defecto que utiliza esta herramienta es creado y gestionado por rsyslogd.

Referencias

[1] `https://docs.docker.com/config/containers/multi-service_container/`

[2] `https://advancedweb.hu/supervisor-with-docker-lessons-learned/`

[3] ¡Ojo! Manejar con precaución. Tampoco se hace en un entorno real de producción.