Centro de Procesamiento de Datos

Práctica 8 Mecanismos de Seguridad para acceso a un servidor

Objetivos

Saber configurar elementos adicionales de seguridad para reducir el riesgo de ataques por accesos no autorizados mediante un sistema Host IDS y añadir autenticación de segundo factor (2F).

Entrega:

Presentar un documento pdf en SWAD→Actividades→ Práctica 1 con la siguiente información:

-De los ejercicios 1,2 capturas personalizadas.
-(Opcional), del ejercicio 3 capturas personalizadas
-(Opcional), del apartado IV (TOR):
  -url de la dirección *.onion donde está activa la página de entrada
  -captura personalizada de la página web desde el TBB (Tor Browser Bundle)
  -captura personalizada del acceso por ssh al servidor utilizando la red TOR

Rúbrica:

Capturas de pantalla	25%
Descripción del trabajo desarrollado	15%
Entrega del ejercicio 3 (apartado opcional)	10%
Entrega del apartado 4 (apartado opcional)	25%
Entrega en plazo	25%

Descripción

Para poder garantizar la seguridad en un servidor es importante contar con herramientas que detecten de forma automática intentos no autorizados de acceso. Resulta conveniente que un servidor de esté conectado directamente a Internet tenga activo mecanismos de detección de intrusos.

Adicionalmente, añadiendo autenticación 2F basada en sistemas TOTP (Time One Time Password) se eleva la seguridad para evitar accesos no autorizados.

Desarrollo:

Todas los apartados se realizan dentro de una máquina virtual con Ubuntu, aunque también se dan indicaciones para CentOS/Red Hat/Fedora. Esta máquina la podemos crear con Vagrant.

I) Fail2Ban

Existen programas más sencillos como DenyHosts para la detección de accesos SSH no válidos y bloqueo de la IP de acceso. Sin embargo, Fail2Ban permite controlar mas servicios que DenyHosts. Además de SSH puede controlar: Apache, Nginx, Lighthttp, Openwebmail, Horde, Mysql, Drupal, Squid, Proftpd, Vsftpd, Postfix, Sendmail, Squirrelmail, Asterix, …

Fail2ban establece "jaulas" (jails) para controlar cada aplicación, de forma que si descubre un posible ataque en el fichero establece una acción. Por ejemplo, en el caso de SSH, añade una regla al cortafuegos bloqueando la entrada de paquetes de la dirección IP durante un tiempo (bantime).

Para instalar fail2ban en Ubuntu:

```
sudo apt-get install fail2ban
```

En CentOS/RPM debemos tener instalado el repositorio EPEL.

```
yum install fail2ban fail2ban-systemd
systemclt enable fail2ban
systemctl start fail2ban
```

```
sudo fail2ban-client status
```

Por defecto se activa la jaula de ssh.

Podemos comprobar las reglas en el cortafuegos

```
sudo iptables -L -n
```

O bien:

```
sudo iptables -S
```

Si queremos desbloquear una IP que ha sido filtrada:

```
fail2ban-client set sshd unbanip <DIRECCION_IP>
```

O bien eliminamos la entrada de las iptables.

```
sudo iptables -D <regla a eliminar>
```

En CentOS/RetHat:

Copiamos el fichero */etc/fail2ban/jail.conf* en */etc/fail2ban/jail.local*

```
cp /etc/fail2ban/jail.conf /etc/fail2ban/jail.local
```

Y editamos sólo la copia para modificar la configuración.

Podríamos cambiar:

bantime: Número de segundos que el nodo sospechoso quedará bloqueado.

maxretry: Número máximo de intentos fallidos.

findtime: tiempo en el que se consideran los intentos erróneos.

ignoreip: lista de direcciones IP que son excluidas del control . Nunca que bloquearán. Pueden ser direcciones IP, máscara CIDR o un nombre separadas por espacios. Ej: ignoreip = 192.168.10.0/24

Los distintos servicios los podemos encontrar en */etc/fail2ban/filter.d*

Para activar servicios debemos asignar *enabled = true* en el fichero de configuración.

Ejercicio 1. Instalamos fail2ban en la máquina virtual Ubuntu que levantamos con Vagrant. (Podemos utilizar cualquiera de las utilizadas en prácticas anteriores). Comprobamos el control de acceso SSH como en el ejercicio. (La parte de CentOS no es necesaria realizarla, sólo está puesta a nivel informativo).

II) Instalación de Google Authenticator

Instalamos el módulo PAM siguiendo las indicaciones de:

```
sudo apt-get install libpam-google-authenticator
```

En el caso de CentOS, seguimos las indicaciones de:

https://github.com/google/google-authenticator-libpam

Editamos /etc/pam.d/sshd y añadimos al comienzo del fichero

```
auth required pam_google_authenticator.so nullok
```

Editamos /etc/ssh/sshd_config

```
ChallengeResponseAuthentication yes
```

Reiniciamos el servicio

```
sudo systemctl restart ssh.service
```

Instalamos qrencode

```
sudo apt install qrencode
```

Finalmente en cada cuenta ejecutamos

```
google-authenticator
```

En los clientes (móvil:Android, iOS) instalamos el Gauth o equivalente.

También podemos añadir el plugin de Authenticator para Chrome o Firefox.

Ejercicio 2. Instalar Google Authenticator. Realizar algunas captura del proceso de acceso mediante dicha autenticación activada.

III) (Opcional) Supervision de NGINX mediante Fail2ban.

Supongamos que queremos supervisar un servidor NGINX. Para controlar intentos fallidos de autenticación, editamos /etc/fail2ban/jail.conf:

```
[nginx-http-auth]
enabled  = true
port     = http,https
logpath  = %(ngninx_error_log)s
```

Reiniciamos:

```
systemctl restart fail2ban
```

Ejercicio 3 Instalamos NGINX en la máquina ubuntu (*sudo apt install nginx*). Creamos una página con acceso mediante autenticación. Configurar fail2ban para bloquear el nodo que intenta acceder. Comprobamos entre nodos el bloqueo en los accesos a la página.

Creamos el fichero /etc/nginx/.httpasswd con las claves

```
sudo sh -c "echo -n 'antonio:' >> /etc/nginx/.htpasswd"
sudo sh -c "openssl passwd -apr1 >> /etc/nginx/.htpasswd"
```

Fichero /etc/nginx/sites-enabled/default

```
server {
   listen 80 default_server;
   listen [::]:80 default_server ipv6only=on;

   root /var/www//html;
   index index.html index.htm;
```

```
    server_name _;

    location / {
        try_files $uri $uri/ =404;
        auth_basic "Restricted Content";
        auth_basic_user_file /etc/nginx/.htpasswd;
    }
}
```

Reiniciamos el servicio

```
sudo systemctl restart nginx
```

VI) (Opcional) Utilizar la red TOR para acceder directamente a la máquina virtual

La red TOR permite crear servicios ocultos y acceder de forma transparente a dicho servidor, ya sean nodos, máquinas virtuales o contenedores.
Configurando el servidor

instalamos tor en el servidor:

```
apt install tor
```

Editamos /etc/tor/torrc

```
RunAsDaemon 1
HiddenServiceDir /root/servicio1
HiddenServicePort 22 127.0.0.1:22
HiddenServicePort 80 127.0.0.1:80
```

Iniciamos con:

```
tor
```

Esta orden inicia el acceso a la red TOR y permite el acceso a los puertos definidos.
El directorio que utilizamos para definir el servicio debe tener permisos de grupo y otros desactivado:

```
chmod go-rwx /root/servicio1
```

En el directorio /root/servicio1 se crean dos ficheros:
hostname: Contiene el nombre del servidor para serutilizado por los clientes: Ej. i549xykbkyeupam4.onion
private_key: Clave privada que permite identificar al servidor

Configurando el cliente

Instalamos el TBB (Tor Browser Bundle)
Iniciamos el TBB

```
./start-tor-browser
```

Se puede acceder directamente utilizando la dirección que aparecen el fichero *hostname* del directorio definido en el servidor.

Para acceder mediante ssh, añadimos las siguientes líneas el el fichero .ssh/config que editamos anteriormente en nuestro equipo (ojo: estas líneas NO se incluyen dentro del contenedor).

```
Host *.onion
  Port 22
  ProxyCommand nc -X 5 -x localhost:9150 %h %p
```

Para Fedora/CentOS poner:
ProxyCommand connect-proxy -S localhost:9150 %h %p

En el acaso de arch-linux hay que instalar openbsd-netcat

Para acceder remotamente en la máquina vagrant:
Modificar */etc/ssh/sshd_config*

```
PasswordAuthentication yes
```

Reiniciar sshd

```
sudo systemctl restart ssh.service
```