Prevención de ataques por fuerza bruta y Man in the Middle

SEGURIDAD INFORMÁTICA

Los ataques de fuerza bruta así como los llamados «*Man In The Middle*», son dos de las violentas agresiones informáticas -tan temidas como frecuentes-, a las que todos aquellos que tenemos a cargo algún servidor, estamos expuestos pudiendo convertimos en posibles víctimas de las mismas. Tomar las medidas necesarias para prevenirlas, es la forma más segura de minimizar los riesgos a los cuáles nos enfrentamos.

Escrito por: **Eugenia Bahit** (Arquitecta GLAMP & Agile Coach)

Eugenia es **Arquitecta de Software**, **docente** instructora de tecnologías **GLAMP** (GNU/Linux, Apache, MySQL, Python y PHP) y **Agile coach** (UTN) especializada en Scrum y eXtreme Programming. Miembro de la **Free Software Foundation** e integrante del equipo de **Debian Hackers**.

Webs:
Cursos de programación a Distancia: www.cursosdeprogramacionadistancia.com
Agile Coaching: www.eugeniabahit.com

Redes sociales:
Twitter / Identi.ca: **@eugeniabahit**

Los ataques de fuerza bruta, no son los únicos de los cuáles nuestro servidor o aplicación, pueden ser víctimas (pensándolo bien, si cuando me roban la billetera, la víctima no es la billetera sino yo, no debería minimizar la situación, pues cuando mi servidor o mi aplicación es atacada y/o agredida, la víctima también soy yo).

Pero volvamos a los tipos de ataques de los cuáles nosotros (seres humanos) podemos ser víctimas. Si bien existe una gran cantidad de tipos de ataques informáticos (de la misma forma que existen infinidad de tipificaciones para los delitos no informáticos), en

este momento nos ocupan dos de los más temidos: los ataques de fuerza bruta y los ataques llamados "Man in the Middle". Pero, actuando desde ahora, podremos intentar prevenir a fin de minimizar los riesgos.

Man in the Middle

Mientras que los ataques de fuerza bruta podrían compararse al delincuente que intenta por todos los medios posibles, abrir una caja fuerte para robar su contenido, los ataques de tipo *«Man in the Middle»* (hombre en el medio -o como me gusta llamarlos a mi, *"mad" in the middle*), representan un modo de agresión mucho más "psicópata" que la anterior. En este caso, el agresor actúa como un psicópata que observa los movimientos de su víctima. Se posiciona en medio de nuestro ordenador y el servidor, "leyendo nuestra correspondencia" antes de enviarla. Podemos comparar un ataque *«Man in the Middle»* a la siguiente situación:

> *Cada vez que escribimos una carta, la depositamos en el buzón de correo, el psicópata nos la intercepta, la abre, la lee, le hace algunos cambios y la entrega él mismo al destinatario original. Luego, hace lo propio con la carta que nos retorne la otra persona.*

En esto mismo consiste el ataque *«Man in the Middle»*. Cuando intentamos conectar al servidor mediante el protocolo SSH, el agresor intercepta nuestros paquetes de datos, los acapara para sí y es él mismo, quien actúa como intermediario para enviar dicha información al verdadero servidor. Cuando el servidor envía una respuesta, el agresor vuelve a interceptarla y nos la entrega modificada, haciéndonos creer que ha sido el servidor quien nos la ha enviado, convirtiéndonos en rehenes de su ordenador.

Medidas preventivas

Contexto: En los ataques de este tipo, cuando el agresor intercepta nuestros "mensajes" al servidor, su ordenador (otro servidor) actúa de forma engañosa, haciéndose pasar por el verdadero servidor.

Cuando te conectas desde tu ordenador a un servidor por primera vez utilizando el protocolo SSH, el servidor te envía una huella digital única (*fingerprint*) que debes aceptar para poder *loguearte* o rechazar, negando así la posibilidad de acceso. Esta huella digital, el servidor la almacena en una clave pública y al ser aceptada por tu ordenador (cliente SSH), se guarda en el archivo know_hosts (hosts conocidos).

Las siguientes veces que te conectes, cuando el servidor envíe su huella digital, el cliente SSH buscará dicha clave en el archivo know_hosts que se encuentra en el directorio `.ssh` de tu home. Al encontrarla, verificará que ambas claves coincidan evitando así, volver a

pedirte su aceptación o rechazo.

El problema: La única posibilidad de que la huella digital del servidor cambie, es que se genere una nueva clave -de forma *ex profesa*- en el servidor, ya sea porque se efectuó una reinstalación completa o porque se eliminó la clave actual y se creó una nueva. Es decir, que si ya te has conectado al servidor anteriormente y aceptado su huella digital, si te vuelves a conectar y te pide aceptar una nueva clave sin que la huella original se haya modificado, deberás rechazarla.

Tomar las precauciones necesarias para evitar el problema: Para estar seguro de cuál es la verdadera huella digital de tu servidor, deberás verificarla en el directorio `/etc/ssh/` del servidor:

```
$ cd /etc/ssh
$ ls -lh
total 164K
-rw-r--r-- 1 root root 123K Apr  2  2012 moduli
-rw-r--r-- 1 root root 1.7K Apr  2  2012 ssh_config
-rw-r--r-- 1 root root 2.6K Dec 10 19:47 sshd_config
-rw------- 1 root root  668 Dec  8 16:44 ssh_host_dsa_key
-rw-r--r-- 1 root root  604 Dec  8 16:44 ssh_host_dsa_key.pub
-rw------- 1 root root  227 Dec  8 16:44 ssh_host_ecdsa_key
-rw-r--r-- 1 root root  176 Dec  8 16:44 ssh_host_ecdsa_key.pub
-rw------- 1 root root 1.7K Dec  8 16:44 ssh_host_rsa_key
-rw-r--r-- 1 root root  396 Dec  8 16:44 ssh_host_rsa_key.pub
-rw-r--r-- 1 root root  302 Jan 10  2011 ssh_import_id
```

Los archivos `.pub` son las llaves públicas generadas por SSH para tu servidor.

Los archivos `ssh_host_`*`formatoclave`*`_key.pub` son aquellos que contienen las huellas digitales públicas que nos interesan.

Para saber cuál de los archivos `.pub` debes mirar, tendrás que verificar el formato de clave especificado en la huella digital que el servidor te está ofreciendo (se resalta el formato en negritas):

```
$ ssh usuariocomun@123.456.78.90
The authenticity of host '123.456.78.90 (123.456.78.90)' can't be established.
ECDSA key fingerprint is 91:bf:9d:7b:b9:9f:15:54:07:fb:d4:23:c6:fa:fd:b0.
Are you sure you want to continue connecting (yes/no)?
```

En el caso anterior deberás mirar la clave de tu archivo `ssh_host_`**`ecdsa`**`_key.pub` . Para visualizarla en el mismo formato -hexadecimal legible- en el que te es ofrecida por el servidor al intentar conectarte, puedes utilizar el comando `ssh-keygen`:

```
$ ssh-keygen -lf ssh_host_ecdsa_key.pub
```

El mismo, arrojará una salida similar a la sguiente:

```
2048 91:bf:9d:7b:b9:9f:15:54:07:fb:d4:23:c6:fa:fd:b0  root@localhost (ECDSA)
```

> ***Antes de aceptar la clave**, verifica que coincida con la que se te está ofreciendo aceptar o rechazar.*

Para verificar que ambas claves son idénticas, puedes escribirlas en dos archivos y utilizar el comando `diff` para que sean evaluadas con precisión. Si éste no arroja nada, significará que no existen diferencias entre ambas claves (es decir, ambas claves son idénticas). Ergo, puedes aceptar la huella con absoluta confianza:

```
$ echo "91:bf:9d:7b:b9:9f:15:54:07:fb:d4:23:c6:fa:fd:b0" > a
$ echo "91:bf:9d:7b:b9:9f:15:54:07:fb:d4:23:c6:fa:fd:b0" > b; diff a b
```

Claro que también podrías compararlas manualmente y confiar en tu buen ojo =)

Ataques de fuerza bruta

Un ataque de fuerza bruta, es aquel mediante el cual, se intenta descubrir una contraseña de forma agresiva, ya sea de modo sencillo (probando diferentes combinaciones de caracteres) o mediante el uso de un diccionario (probando combinaciones de nombres de usuarios y contraseñas de uso frecuente).

> *Utilizar contraseñas que incluyan letras en minúsculas, mayúsculas, números y otros caracteres con una longitud mayor a 16 caracteres, **ayuda a minimizar los riesgos** frente a intentos de robo por ataques de fuerza bruta.*

Medidas preventivas para conexiones por SSH

Lo primero que debemos hacer, es configurar ciertas medidas de seguridad en el demonio de SSH (en el servidor). El archivo de configuración se encuentra en

```
/etc/ssh/sshd_config
```

Las **medidas que nos ayudarán a prevenir ataques de fuerza bruta**, son las siguientes:

1) No permitir el login remoto del usuario root

```
PermitRootLogin no
```

Esta medida evitará que en caso de una agresión con resultado favorable para el atacante, puedan acceder al servidor con permisos absolutos (es decir, como súper usuario).

Está muy claro que necesitamos contar con un usuario que cuente con dichos permisos. Solo bastará con acceder mediante un usuario común y *loguearnos* como *root* una vez dentro del servidor.

La alternativa, es **crear un usuario común y agregarlo al grupo de administradores**:

```
# adduser usuariocomun
# usermod -a -G sudo usuariocomun
```

2) Limitar la cantidad de intentos de logueo fallidos

```
MaxAuthTries 3
```

En este caso, indicamos que el máximo número de veces que podemos equivocarnos al intentar *loguearnos*, es de tres. **Esta directiva no bloqueará al usuario ni su IP**. Simplemente le cerrará la conexión, haciendo más "tedioso" el trabajo del agresor.

Esto ayuda a prevenir los ataques de fuerza bruta que utilizan diccionarios, ya que el proceso de agresión se hará más lento al tener que reiniciar la conexión en cada oportunidad que el servidor desconecte al atacante.

3) Limitar la cantidad de conexiones simultáneas

```
MaxStartups 2
```

Mediante esta directiva, estamos indicando que no puede haber más de dos conexiones simultáneas desde una misma IP. Esto es muy útil -al igual que el caso anterior-, para ralentizar el trabajo del agresor, puesto que la cantidad de intentos simultáneos se verán limitados, generando una una menor tasa de acierto.

4) Limitar la cantidad máxima de tiempo durante la cual se mostrará la pantalla de logueo

```
LoginGraceTime 45
```

45 segundos, es tiempo más que suficiente para introducir tu contraseña. Al igual que en los dos casos anteriores, esta medida hará que el trabajo del agresor se haga cada vez más lento y tedioso. Menos tiempo para los intentos, generará una tasa de acierto cada vez más baja.

5) Limitar el acceso SSH a un usuario determinado

```
AllowUsers usuariocomun
```

Si solo habilitamos el acceso por SSH a un usuario, los intentos del agresor por ingresar al servidor, serán prácticamente imposibles, siempre y cuándo éste, desconozca por

completo el nombre del usuario que cuenta con dicho permiso.

Si se desea habilitar dicho acceso a más de un usuario, se colocarán sus nombres en la misma línea, separados por un espacio en blanco:

```
AllowUsers usuariocomun usuario2 usuario3
```

Si se cuenta con una IP fija en el cliente, se puede restringir aún más el acceso de este usuario, solo desde la IP del mismo. En este caso, a no ser que el agresor logre acceder al ordenador del usuario (ya sea de forma virtual o física), **será imposible que logre acceder por SSH**:

```
AllowUsers usuariocomun@123.456.78.90
```

Se puede necesitar ser un poco menos radical con las autorizaciones y, limitar el acceso a un grupo determinado de usuarios:

```
AllowGroup grupohabilitado
```

6) Cambiar el puerto por defecto

```
Port 372
```

Una medida menos efectiva pero algo astuta, es modificar el puerto por el cual se accede mediante el protocolo SSH. Por defecto, este puerto es el 22.

Una gran cantidad de "armas" (perdón, debí decir "herramientas"), se dedican a intentar realizar los ataques de forma predeterminada (y automatizada) directamente a través de este puerto. Cambiar el número de puerto, no garantiza absolutamente nada. Simplemente retrasará unos pocos minutos el ataque, dado que existen formas -también agresivas- de escanear los puertos remotamente y ver cuál es el que está a la escucha de SSH.

Vale aclarar que el puerto, debería ser inferior a 1024.

7) La alternativa más segura: no permitir el acceso mediante contraseña

A esta altura, no debes estar entendiendo nada y hasta me debes haber catalogado de demente irreversible. Y créeme: lo entiendo. Pero déjame que te explique de que se trata.

Existe una alternativa para iniciar sesión por SSH, mediante la cual, en vez de utilizar una contraseña, el servidor verifica tu identidad de la misma forma que tu verificas la suya: a través de una huella digital única.

Esta técnica, consiste en generar una clave RSA en el ordenador desde el cual se le permitirá el acceso a un determinado usuario y luego, enviar una copia de la clave pública generada al servidor.

De todas las medidas, ésta es quizás, la menos vulnerable de todas.

Para implementarla, el primer paso consistirá en generar una clave RSA para tu usuario, en tu ordenador local. Para ello, deberás ejecutar el comando `ssh-keygen` y seguir los pasos conforme vayan apareciendo en pantalla:

```
$ ssh-keygen
```

Una vez generada la clave, **en el directorio `.ssh` de tu home**, encontrarás dos archivos:

```
id_rsa       esta es tu clave privada. No debes compartirla con nadie
id_rsa.pub   es tu clave pública y solo ésta, deberás enviar a tu servidor
```

Enviarás dicha clave al servidor, mediante el comando `scp`, reemplazando -obviamente- tu usuario e IP:

```
scp ~/.ssh/id_rsa.pub usuariocomun@123.456.78.90:
```

Una vez enviada al servidor, ingresas en éste y realizas los siguientes cambios:

Creas un directorio (dentro de la home de tu usuario en el servidor) para almacenar la clave que has enviado anteriormente:

```
mkdir .ssh
```

Mueves la clave pública dentro del directorio creado:

```
mv id_rsa.pub .ssh/authorized_keys
```

Modificas los permisos del directorio y del archivo, de a uno a la vez:

```
chown -R usuariocomun:usuariocomun .ssh
chmod 700 .ssh
chmod 600 .ssh/authorized_keys
```

Y finalmente, desactivas el acceso SSH mediante contraseña, desde el archivo `/etc/ssh/sshd_confg`:

```
PasswordAuthentication no
```

> *Recuerda que cada vez que realices cambios en el archivo* `/etc/ssh/sshd_config` *deberás reiniciar el servicio SSH:* `service ssh restart`

8) Bloquear las IP tras varios intentos fallidos

Puedes instalar una herramienta que se encargue de detectar los intentos de acceso fallidos (verificando los *logs* de autenticación) y automáticamente, cree reglas que

bloqueen temporalmente, las IP que hayan realizado dichos intentos. Una de estas herramientas, es **fail2ban**. Puedes encontrarla en los repositorios oficiales de tu distribución GNU/Linux con suma facilidad o descargarla ingresando en:

http://www.fail2ban.org/wiki/index.php/Downloads

Una excelente guía en español sobre **cómo configurar correctamente fail2ban**, puede encontrarse en el sitio Web oficial, de la mano de Manuel Aróstegui Ramírez, ingresando en: http://www.fail2ban.org/wiki/index.php/**HOWTO_fail2ban_spanish**

Medidas preventivas en aplicaciones Web

Es posible que en tus aplicaciones Web, se realicen ataques de fuerza bruta. Las medidas de seguridad a implementar, dependen solo y exclusivamente de cada aplicación en particular. Sin embargo, tener en cuenta las siguientes pautas, ayudará a un desarrollo mucho más seguro:

- **Utiliza una política de contraseñas estricta:** obliga a tus usuarios a utilizar contraseñas de no menos de 12 caracteres, que incorporen tanto números, como letras en mayúsculas y minúsculas y al menos 1 o 2 caracteres que no sean alfanuméricos;
- **Limita la cantidad de intentos de acceso:** procura permitir un máximo de 4 intentos. Genera *logs* de aplicación propios, con cada uno de los intentos de acceso a la aplicación. Estos *logs* te servirán para que tu aplicación pueda analizar de forma rápida y confiable, aquellos usuarios e IP que hayan fallado varias veces sucesivas en su intento de acceso;
- **Bloquea los usuarios que hayan fallado al menos 4 veces seguidas en su intento de acceso:** procura avisar previamente que serán bloqueados e inhabilítales el reingreso por al menos 60 minutos;
- **Bloquea aquellas IP desde las cuáles, se haya intentando acceder de forma errónea:** ten presente que desde una misma IP, el atacante podría intentar probar con diferentes combinaciones de usuario y contraseña. Es muy importante que verifiques esto en los *logs* de tu aplicación y bloquees por intentos fallidos por IP y no solo por usuario. En estos casos, no será al usuario al que debas inhabilitar, sino a la IP;
- **Minimiza la posibilidad de que los agresores lleguen a tu aplicación a través de los buscadores:** procura no utilizar nombres comunes para los sistemas de administración. Evita URIs que contengan palabras como *admin*, *panel*, *login* y otros términos que puedan asociarse fácilmente, a formularios de *logueo*.

Tip by Commander in Chief:

Una forma de utilizar contraseñas fáciles de recordar y complejas de descubrir, es *hashear* el nombre de tu canción favorita, grupo de música o deportista preferido y utilizar ese *hash* como *password*. Si tienes **PHP-CLI** instalado, desde la terminal ejecutas:

```
php -r 'print md5("creedence clearwater revival");'
```

o con **Python**:

```
python -c "import hashlib; print hashlib.md5('creedence
clearwater revival').hexdigest()"
```