Práctica 2

Pentesting

1. Inyección SQL para obtener información privilegiada

Vamos a beneficiarnos de una mala gestión de los errores en una aplicación web determinada para obtener información privilegiada.

En ocasiones, algunas páginas web gestionan incorrectamente los errores que se producen, devolviendo incluso código SQL como posible origen del error. Esto puede ser aprovechado para obtener información importante, como podría ser la versión del gestor de bases de datos utilizado, el nombre de las tablas de una base de datos en cuestión e incluso, podríamos incrustar código SQL para obtener el contenido de dichas tablas.

Vamos a probar está técnica utilizando la página web de prueba Mutillidae. Para ello debemos tener arrancada Metasploitable (objetivo de nuestro ataque) y Kali (desde donde vamos a atacar):

1. Accedemos a la página web de Mutillidae (http://ipMetaisplotable/mutillidae) desde Kali y vamos a hacer una consulta que pueda generar un error SQL. Por ejemplo, accedemos a la sección View Blogs en menú OWASP top 10/A1-Injection/SQLMAP practice target/View someone's blog. En dicha sección aparecerá un combobox con los diferentes usuarios.

2. En SQL sabemos que en una consulta a una tabla el modificador "like" se suele utilizar para acceder a los registros. De esta manera una posible consulta a una hipotética tabla de usuarios del blog podría ser la siguiente: "SELECT * FROM blogs WHERE nombre_autor like 'nombre_usuario'". Podemos intentar modificar esa consulta para que genere un error de sintaxis poniendo, por ejemplo, una comilla simple.
   En la práctica anterior aprendimos a cambiar la información de los POST a través de BURP. Haz una intercepción con BURP, solicita ver el blog de un usuario cualquiera y cambia el nombre de usuario para que genere un error en la consulta SQL.

3. En la respuesta del servidor aparece el error en la consulta SQL, mostrando el nombre de la tabla “blogs_table” y el campo “blogger_name”. Esto implica que podemos utilizar inyección sql para obtener información relevante. Para ello vamos a utilizar una conocida herramienta: sqlmap. Entra en la línea de comandos y estudia la ayuda de sqlmap (man sqlmap), sobre todo las opciones principales (son las que llevan un solo guion).

4. En la ayuda, a través del modificador -r, hemos podido comprobar que podemos cargar una solicitud http desde un fichero. La solicitud http la podemos sacar a través de la intercepción con BURP. Intercepta una solicitud para ver el blog de cualquier usuario, copia la petición y guárdala en un fichero.

5. Lo primero que vamos a hacer es averiguar cuántas bases de datos hay en el sistema y para ello utilizamos el modificador *--dbs*, para acelerar el proceso podemos añadir dos hilos con el modificador *--threads=2*:

   $*sqlmap -r ficheroHttpInterceptado --threads=2 --dbs*

   Le decimos que no siga probando otros gestores de bbdd cuando haya averiguado cuál es y que incluya todos los test disponibles para el gestor de bbdd especifico.

6. Como resultado nos aparecerán las bbdd disponibles. La que nos interesa a nosotros es OWASP10 que es la que maneja la página objetivo. A través del modificador *-D* le especificamos la bbdd y con *--tables* sacamos la información de las tablas:

   $*sqlmap -r ficheroHttpInterceptado --threads=2 -D owasp10 --tables*

7. Una vez tenemos el nombre de las tablas, podemos tener acceso a los valores de los registros de cualquiera de las tablas a través del modificador *-T*. Vemos que hay una tabla con un nombre muy sugerente “credit_cards”. Con el modificador *--dump* podemos volcar el resultado a un fichero ().

   $*sqlmap -r ficheroHttpInterceptado --threads=2 -D owasp10 -T credit_cards –dump*

8. Con la información que ya tenemos vamos a volver a Brup. Interceptamos de nuevo la petición de los blogs. Ahora vamos a insertar código sql para obtener información de las tarjetas de crédito. Como ya sabes cómo es la consulta (la que generó el rror) vamos a concatenarle el código necesario para que la consulta resultante sea:

   select *
   from blogs_table
   where blogger_name like ‘admin’
   union select 100, ccnumber, ccv ,expiration
   from credit_cards
   where 'hola' like 'hola'
   ORDER BY date DESC LIMIT 0,100

   También podríamos haber obtenido nosotros los nombres de las tablas inyectando la cadena:

   'union SELECT 1,TABLE_NAME,'hola1','hola2' FROM INFORMATION_SCHEMA.TABLES WHERE TABLE_TYPE = 'BASE TABLE' AND TABLE_SCHEMA='owasp10

2. Vulnerabilidad en Network File System (NFS)

Network File System es un protocolo de sistema de ficheros distribuido para acceder a ficheros remotos como si fuese parte del sistema de ficheros local. NFS es un estándar abierto (RFC 3530), que cualquiera puede implementar y que está construido, como muchos otros protocolos de acceso remoto, bajo el Open Network Computer Remote Procedure Call (ONC RPC).

Vamos a aprovechar una incorrecta configuración del NFS para obtener acceso de superusuario. Para ello, primero localizaremos el puerto sobre el que está montado el NFS, luego averiguaremos sobre qué directorio esté montado el mismo, y si nos lo permite, añadiremos nuestra clave ssh al fichero authorized_keys para poder acceder al sistema objetivo en modo root.

Antes de empezar tenemos que asegurarnos de tener encendidas las máquinas Kali y Metasploitable:

1. Para escanear los puertos de nuestra máquina objetivo, vamos a utilizar un comando muy útil para explorar redes: *nmap*. Abre la consola de Kali y examina la ayuda de la orden nmap. Nos va a venir bien aquellos modificadores que nos ayuden a saber los puertos, el sistema operativo, el número de versión. Además queremos hacer un escaneo agresivo.

   Antes de pasar al siguiente punto intenta sacar los modificadores de la orden nmap para realizar todo lo descrito anteriormente. Consejo: para buscar en cualquier manual en Linux: /cadenaAbuscar

2. Una vez hayamos examinado la ayuda convenientemente, nos daremos cuenta de que la orden que necesitamos utilizar es la siguiente:

   $nmap -p 1-65535 -T4 -A -v ipMetasploitable 2>&1 | tee /var/tmp/scan.txt

   Donde -p 1-65535 nos va a permitir escanear todos los puertos, -A -v nos va a permitir averiguar el sistema operativos y su número de versión, con -T4 vamos a realizar un escaneo agresivo, con la opción 2>&1 vamos a sacar los errores (2 o stderr) por la salida estándar (1 o stdout), y finalmente el resultado lo coge como entrada la orden *tee* que va a crear un fichero scan.txt con dicha información: es decir cogemos la salida de nmap y la guardamos en un fichero.

3. Hemos realizado un escaneo intenso, de todos los puertos existentes de la máquina objetivo y el resultado lo hemos guardado en el fichero scan.txt. Ahora vamos a buscar qué puerto esta asociado al servicio potencialmente vulnerable, NFS. Para ello vamos a buscar mediante la orden grep la cadena ‘nfs’:

   $ grep -i ‘nfs’ /var/tmp/scan.txt

Como ejercicio opcional intenta sacar el sistema operativo, su versión y si hay puertos asociados a servicios como ssh o rpcbind.

4. Ya que NFS esta implementado sobre RPC, nos vamos a beneficiar la información que nos pueda dar la utilidad *rpcbind*. Rpcbind es una utilidad que asocia los servicios que utilizan RPC con los puertos en los que están escuchando. De esta manera el cliente accede al servidor mediante rpcbind con un número de puerto RPC determinado, y rpcbind lo redirige para que el cliente pueda establecer conexión con el servicio solicitado.

   Por defecto Kali no tiene instalado las utilidad rpcbind, ni la utilidad NFS, así que necesitamos instalarlas:

   $apt install rpcbind
   $apt install nfs-common

5. Vamos a averiguar qué programas están utilizando el servicio NFS. Para ello vamos a utilizar la orden rpcinfo. Explora la ayuda de rpcinfo y averigua cuántos programas están usando NFS y cuáles son (identificador).

6. Con la orden showmount podemos averiguar quién puede montar qué determinado directorio. Explora las opciones de showmount y averigua las opciones de exportación del servidor NFS de nuestra máquina objetivo. NOTA: El * significa que el directorio puede ser montado por cualquier usuario.

7. Una vez que sabemos lo que podemos hacer con el NFS de nuestra máquina objetivo, procedemos a aprovecharnos de esta información montando el sistema de ficheros remoto en nuestra máquina. Para ello utilizaremos la orden *mount* de la siguiente manera:

   $mount -t nfs ipMetasploitable:dirNFS /mnt -o nolock

   El modificador *-t nfs* nos permite especificar el sistema de ficheros a montar (NFS), *ipMetasploitable* será la ip de nuestra máquina objetivo y *dirNFS* el directorio remoto que averiguamos en el punto anterior, /mnt el punto de montaje local y *-o nolock* deshabilita el bloqueo de ficheros.

8. Ahora ya tenemos montado en /mnt el directorio remoto de Metasploitable y podremos modificar lo que queramos. Dependiendo de cual sea el directorio remoto que hayamos montado podremos hacer más o menos cosas. Pero esto sólo nos permitirá modificar ficheros y directorios, pero no tenemos el control de la máquina sólo el acceso a una parte del sistema de ficheros. ¿Se te ocurre alguna idea de cómo podemos hacer para obtener acceso de superusuario?

   Si tuviésemos acceso al directorio root podríamos guardar nuestra clave rsa_pub en su directorio .ssh y meterla en su fichero de 'authorized_keys'. De esta manera podríamos hacer una conexión ssh sin que nos pida una contraseña.

3. Autenticación ssh a través de Metasploit

Metasploit es un framework para realizar pruebas de penetración. Dispone de multitud de exploits y la capacidad para desarrollar fácilmente nuestros propios exploits.

En este caso vamos a utilizar un exploit para obtener una sesión ssh, a través de un ataque con fuerza bruta. Intentaremos obtener acceso a la cuenta root de nuestra máquina Metasploitable 2 a través de un diccionario disponible en Kali.

1. Lo primero que tenemos que hacer es arrancar la consola de Metasploit:

   $msfconsole

2. Una vez dentro de la consola de Metasploit deberemos buscar el exploit correspondiente o si sabemos el nombre y la localización acceder a él. El exploit a utilizar es el *ssh_login*. Para buscarlo:

   msf> search ssh_login

   Una vez que sabemos dónde está, accedemos a él:

   msf> use auxiliary/scanner/ssh/ssh_login

3. Una vez dentro del exploit vemos las opciones que tiene:

```
msf auxiliary(scanner/ssh/ssh_login) > show options
```

4. Queremos obtener el acceso a la cuenta de superusuario de nuestra máquina objetivo, por lo tanto, deberemos cambiar los parámetros RHOSTS, donde le indicaremos la ip de la máquina objetivo y USERPASS_FILE, donde le diremos el diccionario para el ataque, que esta en /usr/share/metasploit-framework/data/wordlists/root_userpass.txt:

   ```
   msf auxiliary(scanner/ssh/ssh_login) > set RHOSTS ipMetasploitable

   msf auxiliary(scanner/ssh/ssh_login) > set USERPASS_FILE /usr/share/metasploit-framework/data/wordlists/root_userpass.txt
   ```

   En este caso el diccionario es un fichero de texto con los posibles nombres de de cuentas de superusuario y las contraseñas más comunes. Podéis echarle un vistazo al diccionario. Ahora asegúrate de modificar la password de root de la máquina objetivo para generar una coincidencia con el diccionario.

5. Una vez configurado los parámetros, ejecutamos el exploit:

   ```
   msf auxiliary(ssh_login) > run
   ```

6. El exploit emepezará a probar usuarios y claves hasta que uno de ellos funcione. Si tenemos éxito, nos mostrará información de la cuenta: nombre, clave, uid, gid... Además nos dejará abierta una sesión con una conexión ssh abierta. Para acceder a la sesión:

   ```
   msf auxiliary(ssh_login) > sessions -i 1
   ```

   Donde 1 sería el número de sesión si es la primera sesión abierta. Una vez obtenido el acceso de mediante ssh podremos hacer todo aquello que nos permita el usuario.