Published on hacktimes.com (http://www.hacktimes.com)

MYSQLBFTOOLS - SACANDO PROVECHO A INYECCIONES "CIEGAS" DE SQL.

By MeTalSluG
Creado 17 Mar 2006 - 11:34

MySqlbf es una herramienta de pentest que nos permite extraer información de una base de datos MySQL realizando un ataque de fuerza bruta sobre aplicaciones web vulnerables a una inyección “ciega” de SQL (Blind SQL Injection). Inicialmente desarrollada en C por **ilo** y publicada en www.reversing.org [1], la última versión disponible (v1.2) se acompaña además con otra serie de utilidades nada despreciables.
Pese a que el autor ha decidido abandonar el proyecto original, éste ha sido retomado por A.Ramos, quien ha portado la herramienta al lenguaje perl. El código fuente podeis encontrarlo en:
bsqlbf.pl [2] junto con un video explicativo accesible aquí [3].

La inyección de código SQL, permite al atacante interactuar con el gestor de base de datos utilizado por la aplicación, pudiéndose en ocasiones no sólo obtener, modificar, añadir o borrar el contenido de la base de datos, sino también ir más allá, ejecutando comandos propios del sistema operativo, con las implicaciones de seguridad que obviamente esto supone. Los ataques son posibles gracias a que la aplicación no realiza un correcto saneamiento de los datos de entrada recibidos por el usuario. El programa acepta y procesa los datos recibidos considerándolos como inocuos, permitiendo al atacante alterar la sentencia SQL original.

Para verificar, sin realizar una auditoría del código fuente, si una aplicación es vulnerable a un ataque de este tipo, se envían determinados datos de entrada para generar errores en la sintaxis SQL. Basándonos en estos errores que nos muestra el servidor, es posible hacer ingenieria inversa de la sentencia SQL original que se está ejecutando, para de este modo, forzar a la aplicación a ejecutar otra/s sentencias. En ocasiones, no se nos muestra ningún mensaje de error.
Esto puede ser debido a que el programador ha modificado la lógica de la aplicación con el fin de ocultar el problema subyacente, quizá redirigiendo al usuario a una página de error personalizada en la que no se muestra ninguna información útil, dando por supuesto que esta contramedida es lo suficientemente eficaz para disuadir a un posible atacante. Craso error. Nos encontramos entonces, ante una situación de inyección “ciega” de SQL, donde la única forma de proceder es formular preguntas a la aplicación, que nos serán contestadas mostrándonos un comportamiento diferente. Nos toca a nosotros interpretar esas respuestas como valores verdaderos ó falsos para extraer

información útil, ya sea simplemente averiguar si el usuario con el que se efectúa la conexión a la base de datos es el administrador, o bien preguntas mucho más complejas.

Las herramientas desarrolladas por ilo, nos facilitan esta labor. A destacar también el programa **mysqlget** que nos permitirá incluso descargar archivos interactuando con el servidor MySQL !! (Para que funcione, el usuario de mysql system_user() necesita tener permisos de lectura del archivo a descargar, y el usuario con el que se ejecuta la select donde se realiza la inyección tiene que tener privilegios FILE).

A continuación os incluimos la traducción del artículo original [4] escrito por **ilo**.
No os perdais el "otro" video [5]. Cuidadito con pestañear ;)

INYECCIÓN "CIEGA" EN MySQL Y ESTRESS DE LA BASE DE DATOS

Os sugiero completar la información proporcionada por este artículo con el resto de documentos que tratan la inyección ciega en MySql disponibles en la web.

Actualmente, ya existen algunas herramientas que permiten obtener información de las tablas de una base de datos (Microsoft SQL Server) cuando existe inyección ciega de código sql, tal es el caso de los programas Datathief o Absinthe. El problema en Mysql es la dificultad para obtener la estructura de la base de datos. En versiones antiguas de Mysql no hay objetos METADATA que definan la estructura de la base de datos o similares, por lo que no se puede crear un procedimiento almacenado para manipular y/o conocer esta información, como hacen estos programas con otros gestores de bases de datos.

La información de este documento está enfocada desde el punto de vista de un servicio web vulnerable a una inyección de código sql.

MÉTODOS ACTUALES DE INYECCIÓN CIEGA DE SQL

A pesar del título, este artículo no trata en profundidad el complejo mundo de la inyección de sentencias sql en bases de datos Mysql. Son solo unas notas!

La mayoria de las herramientas disponibles utilizan procedimientos almacenados para extraer los datos. La configuración de la herramienta incorpora algunas variables para crear los procedimientos almacenados en el servidor remoto y la herramienta interactua posteriormente con ellos.
El servidor Microsoft SQL Server mantiene un catalogo (igual que lo hacen otros) sobre como están formadas las bases de datos, sus tablas y sus campos. Los procedimientos almacenados usan este catalogo para mostrar la estructura de la base de datos al usuario, de forma que se pueda interactuar con ella.

Otra de las funcionalidades no disponible en Mysql es la posibilidad de ejecutar más de una sentencia en una consulta a la base de datos desde el servicio web. Es decir, las sentencias se ejecutarán separadas con el caracter ";", mientras que Mysql solo permite la ejecución de una sentencia a la vez. En este sentido, Mysql es la menos desarrollada (o la más segura) interfaz, ya que Mysql v3 no permite el operador UNION en las sentencias.

TIPOS DE DATOS

Una de las caracteristicas que ofrece Mysql a la hora de realizar una inyección de código sql ciega son los tipos de datos de las variables.
Casi todos los valores pueden ser gestionados como una variable, por lo que no hay necesidad de identificar el tipo de datos en la petición, lo cual es realmente útil.

Olvídate de los formatos de los campos como VARCHAR, DATE ya que son establecidos por defecto como "variant" por el servidor web o por el motor de mysql. Por lo que por ejemplo, la sentencia "SELECT 1 FROM
users WHERE 1=1" será aceptada de igual forma que "SELECT '1' FROM users WHERE 1=1". Lo que significa que el gestor de base de datos dará el mismo resultado en la dos sentencias, sin embargo si el servicio web espera determinado tipo de datos la aplicación fallará.

EXPLOTANDO INYECCIÓN CIEGA DE SQL EN MySQL

Se asume que el lector conoce lo que es una inyección de sentencias SQL y cuando se dice que se trata de una inyección “ciega” (Blind sql injection). Comencemos por un script web que nos permita inyectar código SQL pero sin mostrarnos ningún error que se pudiera producir. Podemos esperar dos tipos de respuestas ante una inyección SQL correcta, una se produciría cuando la sentencia devuelve el mismo resultado esperado sin una inyección, y la otra sería una respuesta sin resultados, una página de error o una redirección, o quizá una página “por defecto”. Los resultados pueden ser diferentes si se utiliza el operador AND o el operador OR en la sentencia inyectada. Me explico:

- Operador OR: podría utilizarse para mostrar múltiples resultados procedentes de una inyección SQL.
- Operador AND: podríamos adivinar valores, con la finalidad de “concretar” las peticiones.

Sería útil determinar los diferentes vectores de ataque en una inyección SQL (MySQL) que nos permitan estresar la base de datos y obtener así su estructura:

- El uso de INTO OUTFILE, o LOADFILE. Hacer uso de funciones que nos permitan interactuar con el sistema de archivos nos será de gran ayuda en multitud de posibles ataques:

La posibilidad de crear una interfaz de la base de datos utilizando volcados a un archivo nos permitirá crear una sencilla versión de una página export.php o export.xxx que haga uso de una conexión a la base de datos. Esto nos permitirá ejecutar comandos como "SHOW DATABASES" o "SHOW TABLES" con el siguiente código: $result = mysql_query("SHOW tables", $db); . Posteriormente, el uso del comando "DESCRIBE nombretabla" con cada una de las tablas nos mostrará más información, con el objetivo de automatizar por completo la exportación de modo similar a la función de exportación de aplicaciones como PhpMyAdmin.
Utilizar el sistema de archivos para obtener el control de la base de datos es un método un poco intrusivo, pero a efectos prácticos viene a ser lo mismo que utilizar los procedimientos almacenados.

- El uso del operador UNION para expandir el área de ataque a otras tablas y objetos por petición:

Cuando está disponible, el operador UNION permite realizar peticiones sql que afecten a otras bases de datos o tablas mediante el formato *** UNION SELECT * FROM mysql.user WHERE **** , esto nos servirá de gran ayuda para extraer datos de la base de datos.

- Manejo de errores, es importante resaltar que una inyección de sentencias sql puede ser “ciega” incluso si el manejo de errores de la aplicación está establecido de tal forma que muestre algun tipo de información al usuario sobre el error que se ha producido.

Lo importante aquí, es la posibilidad de que se nos muestre un error ocasionado por el código que será ejecutado posteriormente al que nosotros hemos insertado. El error mostrado no sera de mucha utilidad en el proceso de extracción de información, pero puede ayudarnos a determinar cómo explotar la inyección, es decir, cuando está bien formada la inyección de prueba.

OBTENIENDO LA ESTRUCTURA DE UNA BASE DE DATOS MySQL

En mi opinión, la primera tarea a realizar es la obtención de la estructura de la base de datos, pero esto, obviamente dependerá de los privilegios que tenga el usuario utilizado por el script o página en cuestión para conectarse a la base de datos. Esta situación podría limitar nuestro ámbito de actuación en la base de datos utilizada por el usuario en la consulta actual o en aquellas a las que el usuario tiene permisos de lectura.

NOTA: De ahora en adelante, con la expresión "resultado esperado" me refiero a la página original que es devuelta por el servidor web en el caso de que no se haya insertado ninguna inyección en la petición, es decir, la página esperada sin ningún tipo de intervención por nuestra parte.

LA TABLA mysql.db

MySql hace uso de una tabla llamada db. Esta tabla, contiene la relación de permisos del gestor de base de datos, y puede ser utilizada para forzar una sentencia select y extraer los nombres de las bases de datos existentes. Pero, sólo de las bases de datos con permisos previamente asignados (lo que se ejecuta es un "GRANT PRIVILEGES" para un usuario) y almacenados aquí.

UTILIZANDO OBJETOS EN UNA INYECCIÓN "CIEGA"

En las notas que acompañan al proyecto sqlbftools están documentados algunos de los objetos disponibles en una inyección “ciega” de sql. Esta prueba de concepto utiliza algunos métodos de fuerza bruta explicados en este documento:
adaptive dictionaries for bruteforce cracking [6].

Los objetos disponibles en una inyección de este tipo que nos resultarán más interesantes son aquellos que exportan información importante del gestor de base de datos, como funciones o valores importantes:

* version(): version actual de mysql
* database(): base de datos actual a la que esta conectado el usuario
* user(): conectado a la base de datos
* system_user(): su significado es obvio :)
* session_user()
* current_user()
* last_insert_id()
* connection_id()
* .....
* y por supuesto, todos los datos disponibles en la petición

Aunque no es posible ejecutar mas de una sentencia select por petición, existen varias funciones como las mostradas anteriormente que sí pueden ser ejecutadas en la propia sentencia.

FUERZA BRUTA DE OBJETOS

Cuando no existe ninguna otra forma de obtener la estructura de la base de datos, un ataque por fuerza bruta puede permitirnos adivinar algunos valores. Uno de los principales problemas que nos encontraremos aquí son los privilegios del usuario. Si el usuario no tiene permisos para leer la base de datos, se mostrará un error. Si la inyección es “ciega”, no se muestra información (ni errores), y por tanto no podemos saber si la base de datos no existe o si el usuario no tiene permisos de lectura sobre ella. El primer problema que se nos presenta es obtener el numero de argumentos en la sentencia SELECT.

NÚMERO DE ARGUMENTOS EN LA SENTENCIA SELECT

El numero de argumentos (1 hasta n) se desconoce inicialmente, pero es fácil adivinarlo de modo incremental mediante una inyección del tipo:
"UNION SELECT 1", "UNION SELECT 1,2" y así sucesivamente hasta obtener los resultados esperados.

OBJETOS DE LA BASE DE DATOS Y OBJETOS DE LAS TABLAS

Una vez que sabemos el numero de argumentos, la inyección podría ser algo tal que así:
UNION SELECT 1,2,...,n FROM basededatos.tabla WHERE 1=0

La sentencia "WHERE 1=0" no añadirá ningún resultado procedente del operador union, de este modo no estaremos modificando la sentencia select correcta, pero si la página devuelta es la esperada, entonces los nombres de la base de datos y de la tabla son válidos. Casi el 90% de las bases de datos tienen una tabla usuarios o quizá otros nombres fácilmente adivinables dependiendo del contexto, de este modo, "a.usuarios" "a.categorias" or "a.articulos" pueden ser una buena opción de cara a efectuar un ataque de fuerza bruta en la base de datos.

Una vez que conocemos la base de datos sobre la que podemos actuar, un ataque de fuerza bruta nos ayudará a determinar los nombres de sus tablas.

Posteriormente, conociendo el nombre de la base de datos y el nombre de una de sus tablas, podemos comenzar el proceso para averiguar los valores de los campos, modificando 1 o 2 o n con el nombre del campo a adivinar. Hazlo uno por uno, o no sabras cuando varios de ellos son incorrectos o uno es correcto, recuerda, es una inyección “ciega”.

OBTENIENDO LOS NOMBRES DE LAS BASES DE DATOS MEDIANTE FUERZA BRUTA

Cuando el operador UNION no esta disponible (MySql v3) , adivinar los nombres de las bases de datos no es tan sencillo como los nombres de las tablas (en la misma base de datos) o los campos.

OBTENIENDO LOS NOMBRES DE LAS TABLAS MEDIANTE FUERZA BRUTA

El uso de inyecciones sql sencillas como "AND tabla.campo = 2" no constituyen una sentencia válida si la tabla no forma parte de la SELECT original. Sólo es posible adivinar más nombres de otras tablas de la base de datos en cuestión si estas son referenciadas en la sentencia SELECT. De todas formas, en una inyección “ciega” no es posible determinar cuando la tabla o el campo son inválidos si no se devuelve el resultado esperado: Podría fallar si la tabla o el campo no son correctos, y ejecutarse sólo de forma correcta sólo si los dos, tabla y campo de la tabla existen.

OBTENIENDO LOS NOMBRES DE LOS CAMPOS MEDIANTE FUERZA BRUTA

La inyección de sentencias sql sencillas como pueda ser "AND field_name = 2" ajustarán la select con un nombre de campo válido. El problema aquí es el selector utilizado para la consulta (AND o OR), porque incluso aunque el nombre del campo sea correcto, no es posible esperar que la base de datos lleve a cabo una selección válida, por lo que no se devolverán resultados. Tras varias pruebas, podemos considerar que la sentencia "OR nombre_campo = 0" constituye una buena aproximación, ya que el resultado devolverá al menos el mismo registro que devolvería sin la inyección, pero es posible que devuelva tambien toneladas de registros (todos aquellos en los que el campo nombre_campo sea 0). Por esto, para asegurarnos de que estamos ante una coincidencia válida, debemos esperar que algunos de los resultados obtenidos esten entre los considerados como válidos. Si la página soporta más de una coincidencia en la petición, y se muestra un elevado número de resultados, podemos dar por concluido el proceso, el campo es válido. Puede suceder, que no se muestren resultados, por dos factores: no hemos dado con un nombre de campo correcto, o bien se ha producido un error cuando el operador OR obtiene más de un resultado. Entonces, el uso del operador AND nos Asegurará que la consulta no ha obviado el nombre del campo (supuestamente válido) debido a errores al mostrar la página.

FUERZA BRUTA DE VALORES

Lo último que nos queda por hacer con una inyección sql “ciega” es llevar a cabo un ataque de fuerza bruta sobre algunos valores de la consulta.(o bien de los objetos mysql descritos con anterioridad)

La inyección sería algo similar a la siguiente:

AND MID($$$VAL$$$,1,1) LIKE CHAR(37)

Esto nos permitirá comprobar si el primer carácter de $$$VAL$$$ es 37.
La subcadena puede ser tan larga como se quiera, y se pueden utilizar otras cadenas:

AND MID($$$VAL$$$,1,4) LIKE 'root'

Para comprobar si el valor de la variable es root. Esto debería retornar la página esperada del mismo modo que si no hubieramos realizado ninguna inyección cuando la comparación LIKE es correcta, es decir, se ha adivinado de forma correcta.

Utilizando un ataque de fuerza bruta contra un juego de caracteres es posible adivinar cada uno de los caracteres del valor. $$$VAL$$$ es el valor, podría ser de igual modo cualquiera de los objetos mysql mencionados anteriormente, como por ejemplo VERSION() , o cualquiera de los valores que formen parte de la sentencia select cuando se realiza la inyección

Apreciaciones a tener en cuenta:

No todo es tan fácil como pudiera parecer..
La versión de Mysql es importante.

Versiones diferentes de Mysql tienen diferentes funcionalidades. De forma que en MySQL 3 y 4 no hay modo de conocer la estructura de las bases de datos si no es utilizando SHOW, en la versión 5 de MySQL existe una nueva tabla, INFORMATION_SCHEMA, que contiene todo acerca de las bases de datos, tablas, columnas y demás. Algunas consultas recursivas en esta tabla nos proporcionarán toda la información concerniente a la base de datos.

http://dev.mysql.com/doc/refman/4.1/en/
http://dev.mysql.com/doc/refman/5.0/en/

Como podeis ver, no sólo están disponibles diferentes funciones, sino tambien diferentes tablas en las diferentes versiones de MySQL. Es importante por lo tanto que nuestra primera tarea sea adivinar la versión de MySQL que queremos explotar, a toda costa. MySQL v3 no soporta sentencias UNION, MySQL v5 incorpora la tabla INFORMATION_SCHEMA… no se pueden ejecutar las mismas operaciones en todas las versiones de MySQL.

LA IMPORTANCIA DEL SISTEMA OPERATIVO

En el manual de MySQL podemos encontrar el siguiente enlace:
http://dev.mysql.com/doc/refman/4.1/en/operating-system-specific-notes.html.

Es aconsejable leerlo cuidadosamente para comprender las diferencias existentes en el comportamiento de MySQL en los diferentes sistemas operativos soportados.
Estas diferencias, dificultarán la creación de herramientas automatizadas. Estándares en

las rutas de acceso al sistema de archivos, tipos de datos y tamaño, operaciones prohibidas y demás son ejemplos de particularidades según el sistema operativo.

Algunas funciones pueden no estar disponibles en todos los sistemas operativos, y otras podrían devolver diferentes valores. En el proceso de automatización de una herramienta para capturar la información hay que tener en cuenta las matizaciones debidas a la ejecución de MySQL en un sistema operativo o en otro, o al menos implementar las operaciones ajustandolas a determinado sistema.

LA IMPORTANCIA DEL SERVICIO WEB

La conexion establecida entre el servidor de aplicaciones y el servidor web puede llevar a cabo diferentes operaciones con los datos hasta que éstos son solicitados o enviados al gestor de bases de datos. La Canonicación de rutas, cadenas o conversiones de formato, juegos de caracteres y adaptación de los mismos … Son factores a tener en cuenta.

La mayor parte de las veces, el servidor web será Apache, y el servidor de aplicaciones será PHP, pero no siempre. Si automatizamos este entorno (LAMP) entonces nuestra herramienta sólo será eficaz aquí.

APLICACIONES DE SEGURIDAD

Casi siempre hay que contar con la existencia de una capa de seguridad añadida a cada una de las anteriores. Las opciones de configuración de seguridad de PHP, mod_security o cualquier intérprete de uri (parser), un firewall de capa 7, listas de control de acceso del sistema operativo … etc. Cualquier ataque podría ser detectado y evitado en cualquiera de ellas, de modo que no siempre es posible detectar de forma automatizada cuando el resultado es bueno o cuando el ataque ha sido detectado.
No siempre se pueden sortear los diferentes mecanismos de seguridad, pero en ocasiones es posible hacerlo, en función de la forma que tenga mysql de interpretar los datos …
http://dev.mysql.com/doc/refman/5.0/en/string-functions.html

Pon todo esto junto …

.. y tendrás unos cuantos problemas para extraer datos de un servidor MySQL según las diferentes opciones/configuracion/entornos que puedas encontrar. De todos modos, cuanto más antigua sea la versión que intentes explotar, más operaciones podrás realizar en versiones posteriores, incluso si hay algun modo más facil en la última versión.

Bien, una vez visto esto, consideremos un entorno compuesto por un servidor MySQL v3, con PHP y magic_quotes activado en un sistema windows
(algunas cosas son más dificiles) y un caso de inyección sql “ciega” donde no se reporta al usuario de la existencia de ningún error. Imaginate las posibilidades … finalmente, como ejemplo de lo que es posible hacer puedes ver algunos videos aquí:

beyond mysql injection video [7]

HERRAMIENTAS

Para probar y juguetear con todo esto, hemos desarrollado unas estupidas herramientas para estresar la base de datos en el peor de los casos:
inyección “ciega” de sql en MySQL v3 (no se aceptan UNIONS) con php magic_quotes y con safe_mode activado …

Las herramientas forman parte del proyecto sqlbftools . Descárgalo aquí. [8]

SQLBF

Estas herramientas hacen un ataque de fuerza bruta a los valores de mysql. El siguiente video http://www.unsec.net/ muestra como usar la herramienta y como extraer información de la base de datos. La mejor opción para un ataque de fuerza bruta son las funciones informativas:

http://dev.mysql.com/doc/refman/5.0/en/information-functions.html

Estas herramientas ahora las mantiene **dab** en lenguaje perl msqlbf.pl.
(http://www.unsec.net)

SQLGET

Te permite descargar archivos de un sitio web vulnerable a inyecciones “ciegas” de sql … incluyendo archivos binarios !!

SQLST

sqlst es un parser para MYD, FRM y archivos MYI .. , estas herramientas fueron diseñadas para leer y parsear tablas, formatos y contenidos de campos para, sin la necesidad de bajarse toda la base de datos ser capaz de leer cualquier valor de la base de datos. Es muy dificil parsear archivos mysql por los cambios del formato MyISAM en las diferentes versiones, pero puedes llegar a hacer algo si te esfuerzas un poco..
Estas herramientas necesitan más pruebas antes de poder publicarlas .. lo siento!! (las herramientas comerciales … son tan caras !! jeje)

NOTAS FINALES

He implementado la fuerza bruta de valores en el proyecto sqlbftools [9] para estresar algunos de los valores que se pueden obtener en una inyección “ciega”. En la web de reversing.org encontrareis algunos videos para que podais ver cómo funciona. De todos modos, el programa es muy fácil de usar, y está tambien documentado para que puedas compilarlo y ejecutarlo tú mismo. Se incluye un archivo leeme con un ejemplo ilustrativo.

Links:

[1] http://www.reversing.org
[2] http://www.unsec.net/download/bsqlbf.pl
[3] http://www.unsec.net/download/bsqlbf.avi
[4] http://www.reversing.org/node/view/13
[5] http://www.reversing.org/files/beyond_mysql_injection.avi
[6] ”http://www.reversing.org/node/view/12”
[7] http://www.reversing.org/files/beyond_mysql_injection.avi
[8] http://www.reversing.org/node/view/11
[9] http://www.reversing.org/node/view/11