Taller de Seguridad Informática

Sistemas Operativos - DC.UBA.AR

1er. Cuatrimestre - 2023

Introducción

Este taller es un CTF. Los CTFs (Captura de bandera o Capture The Flag por su sigla en inglés) son competencias de seguridad informática que posibilitan el aprendizaje de distintas cuestiones vinculadas con la seguridad de manera lúdica. El objetivo de estas competencias es descubrir "flags". Las llamadas "flags" (banderas) son piezas de información que se ocultan, y que deben ser encontradas por los participantes.

Se incluyen una variedad de debilidades y vulnerabilidades comunes en Linux. Entre otros, encontrarás problemas de seguridad relacionados con:

- Permisos de archivos.
- Archivos con flag SetUID.
- Mal uso de variables de ambiente.
- Condiciones de carrera.

La máquina virtual provista presenta diferentes desafíos o niveles de seguridad. Para intentar resolver cada uno tenés que loguearte con el usuario "nivelXX" con clave "nivelXX" (sin comillas), donde XX es el número de nivel.

En todos los niveles, debes poder ejecutar el comando /bin/checkflag con los privilegios del usuario flagXX y te debe devolver: "Felicitaciones, nivelXX resuelto!" y una frase de una película (esa es la bandera). Esta es la forma de comprobar que resolviste el desafío de dicho nivel.

Si después de probar mucho rato no te sale un nivel, podés ejecutar el comando /bin/damepista, y obtener una pequeña ayuda. También podés saltear algún nivel, son todos independientes entre sí.

Descarga e instalación de la VM

En las PCs de los labos

- Ejecutar lo siguiente:

  ```
  wget https://bit.ly/tallerseginf -O - | bash
  ```

- Abrir el programa VirtualBox y encender la VM llamada so-labo-seginf

En una Notebook/PC Personal

- Es necesario tener instalado VirtualBox.
- Descargar el archivo OVA
- Importar el archivo descargado en VirtualBox (Menú Archivo ¿Importar servicio virtualizado...).
- Encender la VM importada.

Algunas ayudas para manejarte con la VM

En la máquina virtual ya están instalados todos los comandos y aplicativos necesarios para realizar los ataques. Si bien tenés un compilador gcc y podés usarlo, los niveles que requieren un poquito de programación se pueden resolver con scripts en bash.

Con Alt+tecla de función (F1,F2,F3, etc) podés abrir más de una terminal, y pasar de una a otra. Tambien lo podes hacer con Alt+Flecha Izquierda ó Flecha Derecha.

Si la configuración de teclado de la VM difiere de la de tu teclado, desde cualquier usuario podes cambiar la distribución usando según corresponda:

```
sudo loadkeys us
sudo loadkeys latam
sudo loadkeys es
```

Nivel00

Este nivel requiere que encuentres y ejecutes en el disco de la máquina virtual un programa con el flag SetUID que corra con los privilegios de la cuenta “flag00”.

Si necesitas ayuda, puedes consultar “man find”.

Para acceder a este nivel, logueate con usuario y clave “nivel00”.

Al finalizar, recuerda ejecutar /bin/checkflag y verificar que te devuelve “Felicitaciones, nivel 00 resuelto!” y una frase de una película.

Nivel01

Hay una vulnerabilidad en el código fuente que figura a continuación, y que permite que se ejecuten comandos arbitrarios con los privilegios de flag01, ¿podés encontrarla?

Para hacer este nivel, logueate como nivel01:nivel01. El binario compilado y con los permisos necesarios está en /home/flag01

Código Fuente

```
#include <stdlib.h>
#include <unistd.h>
#include <string.h>
#include <sys/types.h>
#include <stdio.h>

int main(int argc, char **argv, char **envp)
{
  gid_t gid;
  uid_t uid;
  gid = getegid();
  uid = geteuid();

  setresgid(gid, gid, gid);
  setresuid(uid, uid, uid);

  system("/usr/bin/env echo y ahora qué?");
}
```

Nivel02

Hay una vulnerabilidad en el código fuente que figura a continuación, y que permite que se ejecuten comandos arbitrarios, ¿podés encontrarla?

Para hacer este nivel, logueate como nivel02:nivel02. El binario compilado y con los permisos necesarios está en /home/flag02.

Código Fuente

```
#include <stdlib.h>
#include <unistd.h>
#include <string.h>
#include <sys/types.h>
#include <stdio.h>

int main(int argc, char **argv, char **envp)
{
  char *buffer;

  gid_t gid;
  uid_t uid;

  gid = getegid();
  uid = geteuid();

  setresgid(gid, gid, gid);
  setresuid(uid, uid, uid);

  buffer = NULL;

  asprintf(&buffer, "/bin/echo %s es genial", getenv("USER"));
  printf("a punto de invocar system(\"%s\")\n", buffer);

  system(buffer);
}
```

Nivel03

Hay un cron (tarea programada) del usuario flag03 que se ejecuta cada algunos minutos y que ejecuta el writable.sh.

Para hacer este nivel, logueate como nivel03:nivel03. Los archivos de este nivel están en /home/flag03.

Nivel04

Este nivel requiere que leas el archivo token, pero el código restringe los archivos que se pueden leer. Encontrá una manera de saltear el control :)

Para hacer este nivel, logueate como nivel04:nivel04. Los archivos de este nivel están en /home/flag04

Código Fuente

```
#include <stdlib.h>
#include <unistd.h>
#include <string.h>
#include <sys/types.h>
#include <stdio.h>
#include <fcntl.h>

int main(int argc, char **argv, char **envp)
{
  char buf[1024];
  int fd, rc;

  if(argc == 1) {
      printf("%s archivo_a_leer\n", argv[0]);
      exit(EXIT_FAILURE);
  }

  if(strstr(argv[1], "token") != NULL) {
      printf("No podés acceder a '%s'\n", argv[1]);
      exit(EXIT_FAILURE);
  }

  fd = open(argv[1], O_RDONLY);
  if(fd == -1) {
      err(EXIT_FAILURE, "No puedo abrir %s", argv[1]);
  }

  rc = read(fd, buf, sizeof(buf));

  if(rc == -1) {
      err(EXIT_FAILURE, "No puedo leer fd %d", fd);
  }

  write(1, buf, rc);
}
```

Nivel05

Verificá el contenido del home del usuario flag05. Ojo con los permisos excesivos.

Para hacer este nivel, logueate como nivel05:nivel05. Los archivos de este nivel están en /home/flag05.

Nivel06

Las credenciales de la cuenta flag06 provienen de un sistema unix legacy.

Para hacer este nivel, logueate como nivel06:nivel06.

Nivel07

El comando sudo permite ejecutar comandos con privilegios de otros, pero permitiendo una configuración más granular con respecto al uso del flag SetUID. Encuentre el error en la configuración de esta herramienta que permita al usuario de este nivel ejecutar cualquier comando con los privilegios de flag07.

Para hacer este nivel, logueate como nivel07:nivel07.

Nivel08

El binario que está en /home/flag08/flag08 hace un upload de un archivo, mientras que se cumpla la condición de la syscall access(). ¿Podés encontrar y explotar una condición de carrera?

Para hacer este nivel, logueate como nivel08:nivel08. Luego, es recomendable abrir dos terminales, en una levantá un servidor con el comando nc y los parámetros correspondientes. En la otra desarrolla un pequeño script que realice el ataque.

Código Fuente

```
#include <stdlib.h>
#include <unistd.h>
#include <sys/types.h>
#include <stdio.h>
#include <fcntl.h>
#include <errno.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <string.h>

int main(int argc, char **argv)
{
  char *file;
  char *host;

  if(argc < 3) {
      printf("%s archivo host\n\tenvía archivo, si tenés acceso al mismo, al host", argv[0]);
      exit(1);
  }
  file = argv[1];
  host = argv[2];
  if(access(argv[1], R_OK) == 0) {
      int fd;
      int ffd;
      int rc;
      struct sockaddr_in sin;
      char buffer[4096];
      printf("Conectando a %s:18211 .. ", host); fflush(stdout);
      fd = socket(AF_INET, SOCK_STREAM, 0);
      memset(&sin, 0, sizeof(struct sockaddr_in));
      sin.sin_family = AF_INET;
      sin.sin_addr.s_addr = inet_addr(host);
      sin.sin_port = htons(18211);
      if(connect(fd, (void *)&sin, sizeof(struct sockaddr_in)) == -1) {
          printf("No me pude conectar al host %s\n", host);
          exit(EXIT_FAILURE);
```

```
      }

#define HITHERE ".oO Oo.\n"
      if(write(fd, HITHERE, strlen(HITHERE)) == -1) {
          printf("No pude escribir el banner al host %s\n", host);
          exit(EXIT_FAILURE);
      }
#undef HITHERE

      printf("Conectado!\nEnviando archivo .. "); fflush(stdout);

      ffd = open(file, O_RDONLY);
      if(ffd == -1) {
          printf("Maldición. No puedo abrir el archivo\n");
          exit(EXIT_FAILURE);
      }

      rc = read(ffd, buffer, sizeof(buffer));
      if(rc == -1) {
          printf("No puedo leer el archivo: %s\n", strerror(errno));
          exit(EXIT_FAILURE);
      }

      write(fd, buffer, rc);

      printf("Archivo escrito!\n");

  } else {
      printf("No tenés acceso a: %s\n", file);
  }
}
```