

# Set-UID Lab

Erik Olsen - Infosec Club

# What is a UID?

- User Identifier
- Every user in linux has a unique UID
- When a user executes a program that program is given the UID of the user
- Used for access control

UIDs can be found in the `/etc/passwd` file

```
(kali@kali)-[~]  
$ cat /etc/passwd | grep -v "chroot" | grep -e  
root:x:0:0:root:/root:/usr/bin/zsh  
kali:x:1000:1000:kali,,,:/home/kali:/usr/bin/zsh  
      ^  
      UID
```

The UID for the `root` user is always 0.

The UID for my `kali` user happens to be 1000.

# The /etc/shadow file

Contains the hashed+salted passwords for all of the users on the system

```
kali:$y$j9T$HlGqULZv2BS4c1t9LSd8f0$.s8R4LbyUjkADBNmIQIKETz68lWbRLdVWTXE  
QbWvQB7:18693:0:99999:7:::
```

```
(kali@kali)-[~]  
$ ls -l /etc/shadow  
-rw-r--r-- 1 root shadow 1577 Mar  7 14:17 /etc/shadow
```

```
(kali@kali)-[~]  
$ cat /etc/shadow  
cat: /etc/shadow: Permission denied
```

```
(root@kali)-[/home/kali]  
# cat /etc/shadow  
root:!:18693:0:99999:7:::  
daemon:!:18693:0:99999:7:::  
bin:!:18693:0:99999:7:::
```

# What is a Set-UID program?

- A program that will temporarily change its UID, regardless of the UID of the user who ran it, to perform a privileged action.
- Example: `/bin/passwd`
  - If a user changes their password, the shadow file must be modified.
  - No one, except root, should be allowed to modify the shadow file.
  - `/bin/passwd` temporarily sets its UID to 0 to modify the shadow file when a user changes their password.

```
(kali㉿kali)-[~/Documents/SetUIDLab]  
$ ls -l /bin/passwd  
-rwsr-xr-x 1 root root 63960 Feb  7  2020 /bin/passwd
```

# Making a Set-UID program

```
(kali㉿kali)-[~/Documents/SetUIDLab]  
$ ls -l myprogram  
-rwxr-xr-x 1 kali kali 16656 Mar  7 15:39 myprogram  
  
(kali㉿kali)-[~/Documents/SetUIDLab]  
$ sudo chown root myprogram  
  
(kali㉿kali)-[~/Documents/SetUIDLab]  
$ sudo chmod 4755 myprogram  
  
(kali㉿kali)-[~/Documents/SetUIDLab]  
$ ls -l myprogram  
-rwsr-xr-x 1 root kali 16656 Mar  7 15:39 myprogram
```

# Lab: Setup

5 min

1. Clone the lab repository

```
(kali@kali)-[~/Documents]  
$ git clone https://github.com/eriknj99/SetUIDLab.git
```

2. Execute enable-vulnerability.sh

```
(kali@kali)-[~/Documents/SetUIDLab]  
$ ./enable-vulnerability.sh  
Making the system vulnerable to Set-UID attacks ...  
sudo ln -sf /bin/zsh /bin/sh  
Done.  
Make sure you run disable-vulnerability.sh when you are done with the lab
```

# Lab: Part 1

```
(kali㉿kali)
$ whoami
kali
```

```
(kali㉿kali)-[
$ id -u kali
1000
```

```
(kali㉿kali)-[~/l
$ id -u $(whoami)
1000
```

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 //PrintMyUID.c
5
6 int main(){
7     printf("Hello! My UID is:\n");
8     system("id -u $(whoami)");
9
10    printf("\nwhich corresponds to the user: \n");
11    system("whoami");
12
13    return 0;
14
15 }
```



# Lab: Part 1

10 min

Task 1: Compile PrintMyUID.c into the program PrintMyUID. (Hint: Use gcc)

Task 2: Run the PrintMyUID program.

Task 3: Change PrintMyUID to a root Set-UID program.

Task 4: Run the PrintMyUID program again and observe the changes.

```
(kali㉿kali)-[~/Documents/SetUIDLab]  
$ sudo chown root myprogram  
  
(kali㉿kali)-[~/Documents/SetUIDLab]  
$ sudo chmod 4755 myprogram
```

# Lab: Part 1 Demonstration

# Lab: Part 2

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <string.h>
4 //-----Exploitable.c-----
5
6
7 char *v[3];
8 char *command;
9
10 int main(int argc, char *argv[]){
11     // argc : The number of command line arguments
12     // argv : An array of string (size argc) containing all command line arguments.
13
14     if(argc > 1){
15
16         printf("The contents of the file %s are:\n", argv[1]);
17
18         // Concatenate /bin/cat with argv[1] and set command equal to the result.
19         // Equivalent to command = "/bin/cat" + argv[1] in python
20         v[0] = "/bin/cat";
21         v[1] = argv[1];
22         command = malloc(strlen(v[0]) + strlen(v[1]) + 2);
23         sprintf(command, "%s %s", v[0], v[1]);
24
25
26         //Execute the command
27         system(command);
28
29     }else{
30         printf("Please enter a filename as a command line argument.\n");
31         printf("Example: $ Exploitable /file/to/print \n");
32     }
33
34
35
36     return 0;
37 }
```

## Lab: Part 2

15 min

Task 1: Compile `Exploitable.c` and make it a Set-UID program with `compile-exploitable.sh`

```
(kali@kali) - [~/Documents/SetUIDLab]  
$ ./compile-exploitable.sh  
Compiling Exploitable.c...  
Making Exploitable a Set-UID program...  
done.
```

Task 2: Make the `Exploitable` program execute arbitrary commands as root using only command line arguments.

# Lab: Part 2 Demonstration

# Lab: Clean Up

1. Execute `disable-vulnerability.sh` to patch the vulnerability we enabled earlier.

```
(kali@kali) - [~/Documents/SetUIDLab]  
$ ./disable-vulnerability.sh  
Patching the Set-UID vulnerability...  
sudo ln -sf /bin/dash /bin/sh  
Done.  
Have a good night :)
```

# Thank You!

```
.....  
      ..,;;ccc,.  
      .....''';lx0.  
.....''''.....:ld;  
      .';;;;;;,,.x,  
      ..'''.  
      .....0Xxoc:,, ...  
      ....,ONkc;;;cok0dc',.  
      .OMo' :ddo.  
      dMc :00;  
      0M. :.o.  
      ;Wd  
      ;X0,  
      ,d00dlc;,...  
      ..',;;cd00d:,,.  
      .:d;.'';.  
      'd, .'  
      ;l ..  
      .o  
      c  
      .  
      .
```