File Descriptors and Redirections

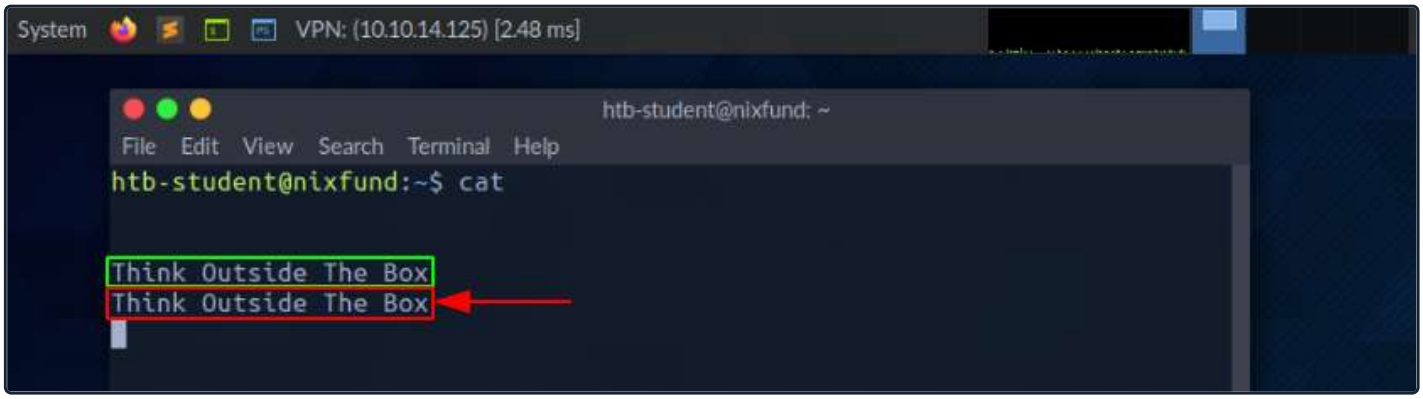
File Descriptors

A file descriptor (FD) in Unix/Linux operating systems is an indicator of connection maintained by the kernel to perform Input/Output (I/O) operations. In Windows-based operating systems, it is called filehandle. It is the connection (generally to a file) from the Operating system to perform I/O operations (Input/Output of Bytes). By default, the first three file descriptors in Linux are:

1. Data Stream for Input
    - `STDIN - 0`
2. Data Stream for Output
    - `STDOUT - 1`
3. Data Stream for Output that relates to an error occurring.
    - `STDERR - 2`

STDIN and STDOUT

Let us see an example with `cat`. When running `cat`, we give the running program our standard input (`STDIN - FD 0`), marked `green`, wherein this case "SOME INPUT" is. As soon as we have confirmed our input with `[ENTER]`, it is returned to the terminal as standard output (`STDOUT - FD 1`), marked **red**.

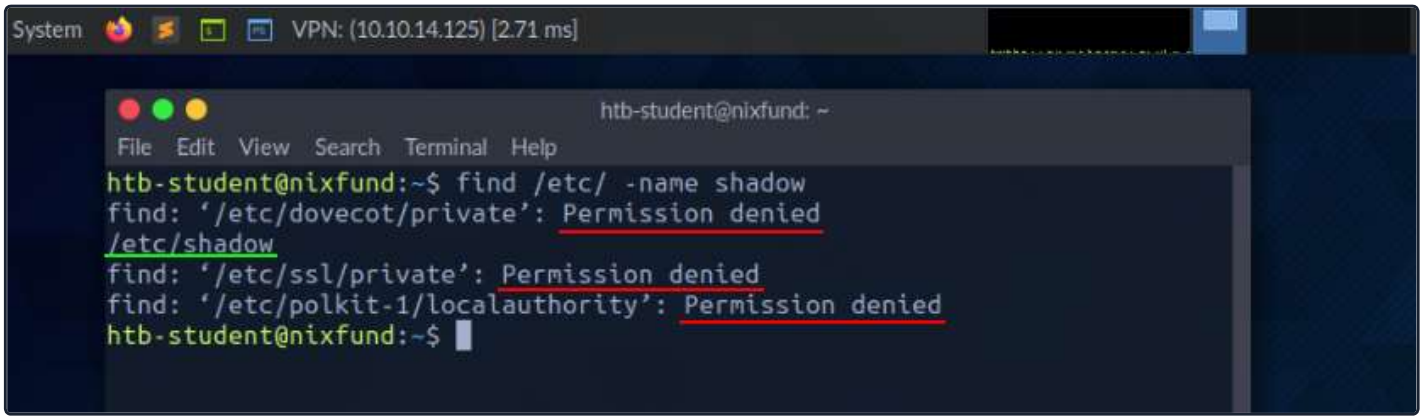


STDOUT and STDERR

In the next example, by using the `find` command, we will see the standard output (`STDOUT - FD 1`) marked in `green` and standard error (`STDERR - FD 2`) marked in red.

STDOUT and STDERR

```
amit8986@htb[/htb]$ find /etc/ -name shadow
```



In this case, the error is marked and displayed with "`Permission denied`". We can check this by redirecting the file descriptor for the errors (`FD 2 - STDERR`) to "`/dev/null`." This way, we redirect the resulting errors to the "null device," which discards all data.

STDOUT and STDERR

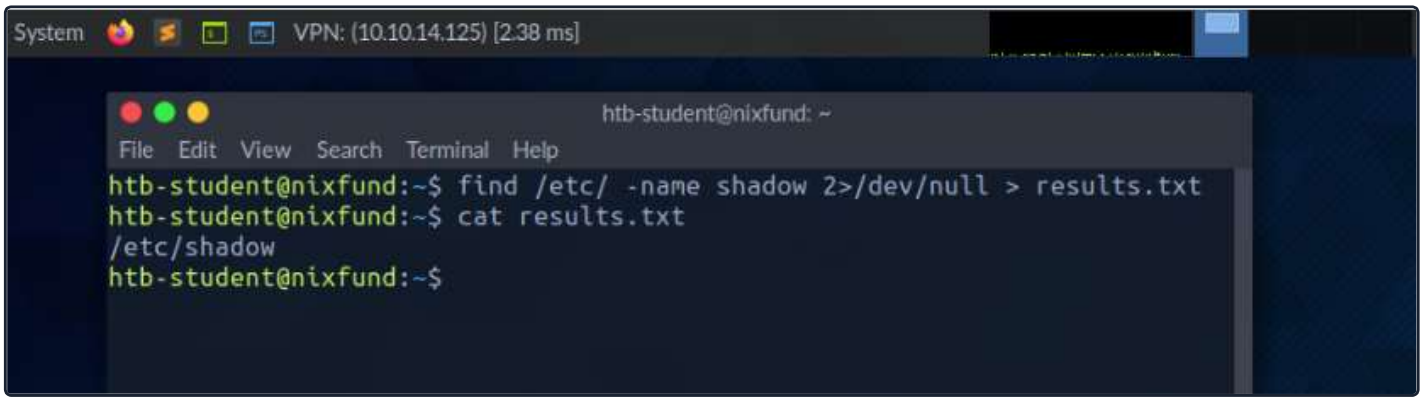
```
amit8986@htb[/htb]$ find /etc/ -name shadow 2>/dev/null
```

Redirect STDOUT to a File

Now we can see that all errors (`STDERR`) previously presented with "`Permission denied`" are no longer displayed. The only result we see now is the standard output (`STDOUT`), which we can also redirect to a file with the name `results.txt` that will only contain standard output without the standard errors.

Redirect STDOUT to a File

```
amit8986@htb[/htb]$ find /etc/ -name shadow 2>/dev/null > results.txt
```

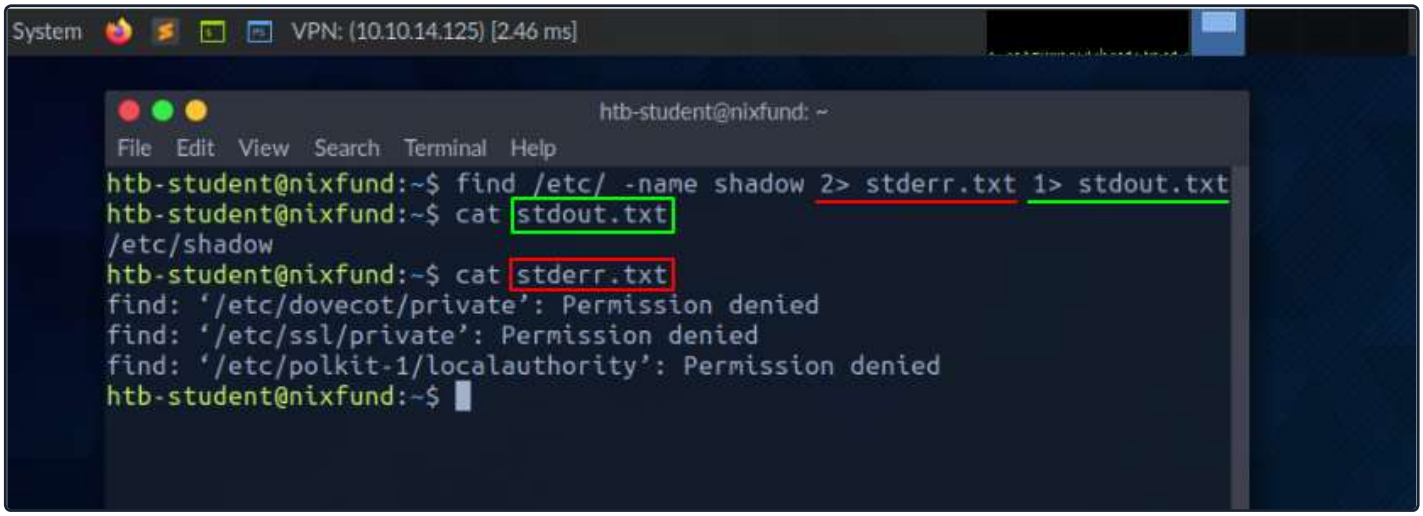


Redirect STDOUT and STDERR to Separate Files

We should have noticed that we did not use a number before the greater-than sign (`>`) in the last example. That is because we redirected all the standard errors to the "`null device`" before, and the only output we get is the standard output (`FD 1 - STDOUT`). To make this more precise, we will redirect standard error (`FD 2 - STDERR`) and standard output (`FD 1 - STDOUT`) to different files.

Redirect STDOUT and STDERR to Separate Files

```
amit8986@htb[/htb]$ find /etc/ -name shadow 2> stderr.txt 1> stdout.txt
```

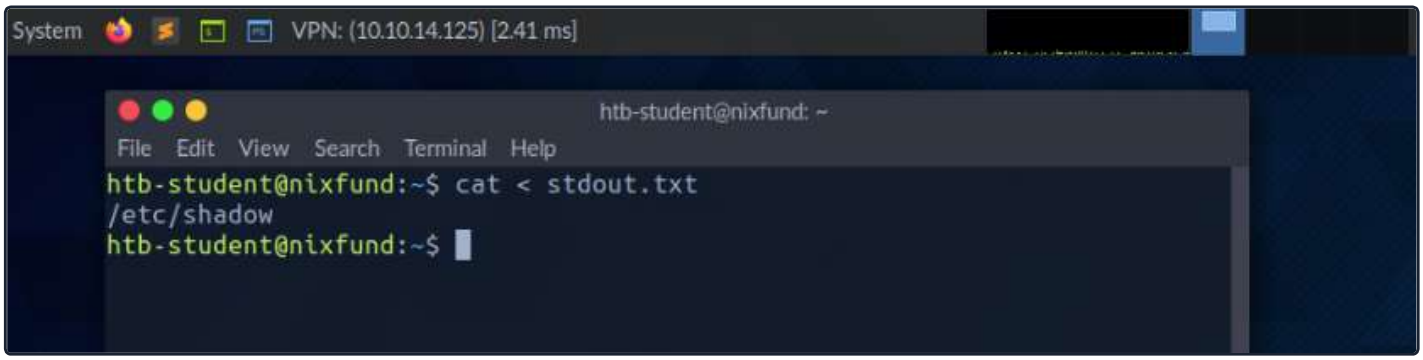


Redirect STDIN

As we have already seen, in combination with the file descriptors, we can redirect errors and output with greater-than character (`>`). This also works with the lower-than sign (`<`). However, the lower-than sign serves as standard input (`FD 0 - STDIN`). These characters can be seen as "`direction`" in the form of an arrow that tells us "`from where`" and "`where to`" the data should be redirected. We use the `cat` command to use the contents of the file "`stdout.txt`" as `STDIN`.

Redirect STDIN

```
amit8986@htb[/htb]$ cat < stdout.txt
```

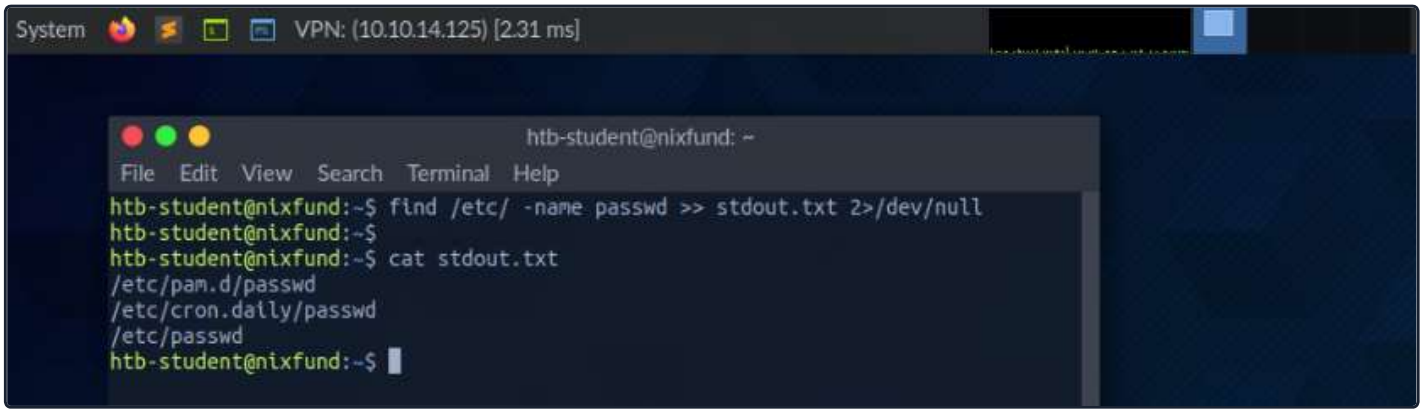


Redirect STDOUT and Append to a File

When we use the greater-than sign (`>`) to redirect our `STDOUT`, a new file is automatically created if it does not already exist. If this file exists, it will be overwritten without asking for confirmation. If we want to append `STDOUT` to our existing file, we can use the double greater-than sign (`>>`).

Redirect STDOUT and Append to a File

```
amit8986@htb[/htb]$ find /etc/ -name passwd >> stdout.txt 2>/dev/null
```

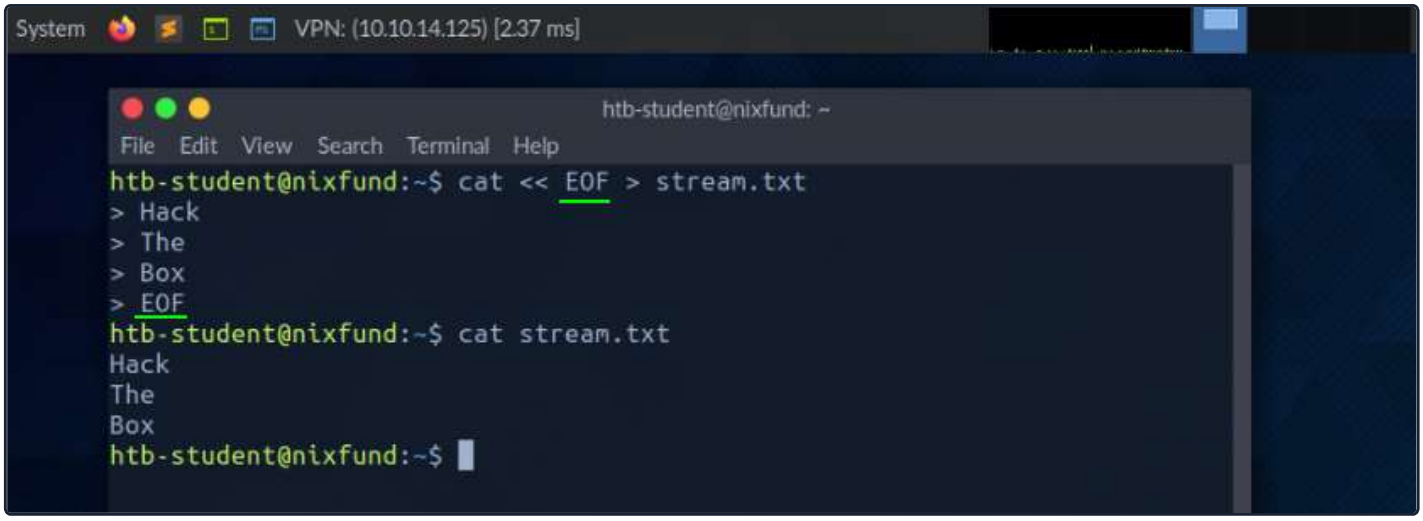


Redirect STDIN Stream to a File

We can also use the double lower-than characters (`<<`) to add our standard input through a stream. We can use the so-called `End-Of-File` (`EOF`) function of a Linux system file, which defines the input's end. In the next example, we will use the `cat` command to read our streaming input through the stream and direct it to a file called "`stream.txt`."

Redirect STDIN Stream to a File

```
amit8986@htb[/htb]$ cat << EOF > stream.txt
```



Pipes

Another way to redirect `STDOUT` is to use pipes (`|`). These are useful when we want to use the `STDOUT` from one program to be processed by another. One of the most commonly used tools is `grep`, which we will use in the next example. Grep is used to filter `STDOUT` according to the pattern we define. In the next example, we use the `find` command to search for all files in the "`/etc/`" directory with a "`.conf`" extension. Any errors are redirected to the "`null device`" (`/dev/null`). Using `grep`, we filter out the results and specify that only the lines containing the pattern "`systemd`" should be displayed.

Pipes

```
amit8986@htb[/htb]$ find /etc/ -name *.conf 2>/dev/null | grep systemd
```

```
htb-student@nixfund:~$ find /etc/ -name *.conf 2>/dev/null | grep systemd
/etc/systemd/system.conf
/etc/systemd/timesyncd.conf
/etc/systemd/journald.conf
/etc/systemd/user.conf
/etc/systemd/logind.conf
/etc/systemd/resolved.conf
htb-student@nixfund:~$
```

The redirections work, not only once. We can use the obtained results to redirect them to another program. For the next example, we will use the tool called `wc`, which should count the total number of obtained results.

Pipes

```
amit8986@htb[/htb]$ find /etc/ -name *.conf 2>/dev/null | grep systemd | wc -l
```

```
htb-student@nixfund:~$ find /etc/ -name *.conf 2>/dev/null | grep systemd | wc -l
6
htb-student@nixfund:~$
```

VPN Servers

⚠ Warning: Each time you "Switch", your connection keys are regenerated and you must re-download your VPN connection file.

All VM instances associated with the old VPN Server will be terminated when switching to a new VPN server.

Existing PwnBox instances will automatically switch to the new VPN server.

eu-academy-1

PROTOCOL

UDP 1337 TCP 443

DOWNLOAD VPN CONNECTION FILE

Start Instance

0 / 1 spawns left

Waiting to start...

Questions

Answer the question(s) below to complete this Section and earn cubes!

Cheat Sheet

Download VPN Connection File

Target: 10.129.55.110

Life Left: 55 minutes

SSH to 10.129.55.110 with user "htb-student" and password "HTB_@cademy_stdnt!"

+ 1 How many files exist on the system that have the ".log" file extension?

32

Submit

+ 0 How many total packages are installed on the target system?

856

Submit

Previous Next

Cheat Sheet

Go to Questions

Table of Contents