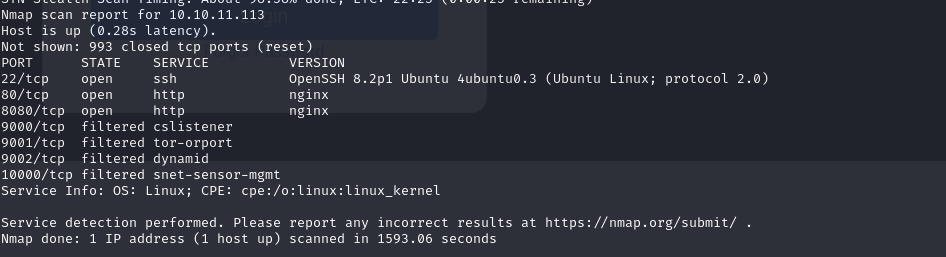
# GOBOX

# INFORMATION GATHERING

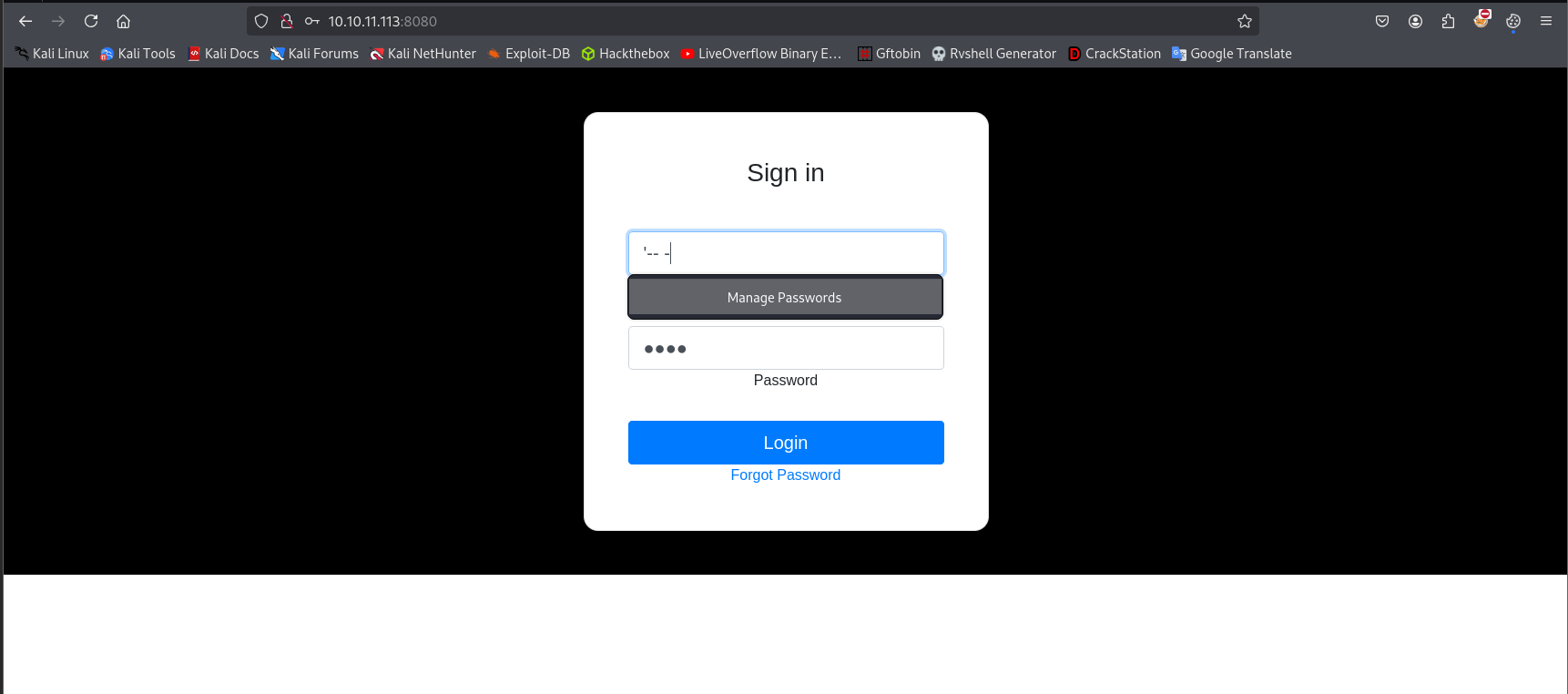
First Lets look at the Nmap result, Nmap reveals that OpenSSH(port 22) two Webserver(80,8080), and another ports 9000,9001,9002,10000 are running. Additionally, ports 9000-9002 are filtered which indicates there are iptable rules blocking access to these ports.



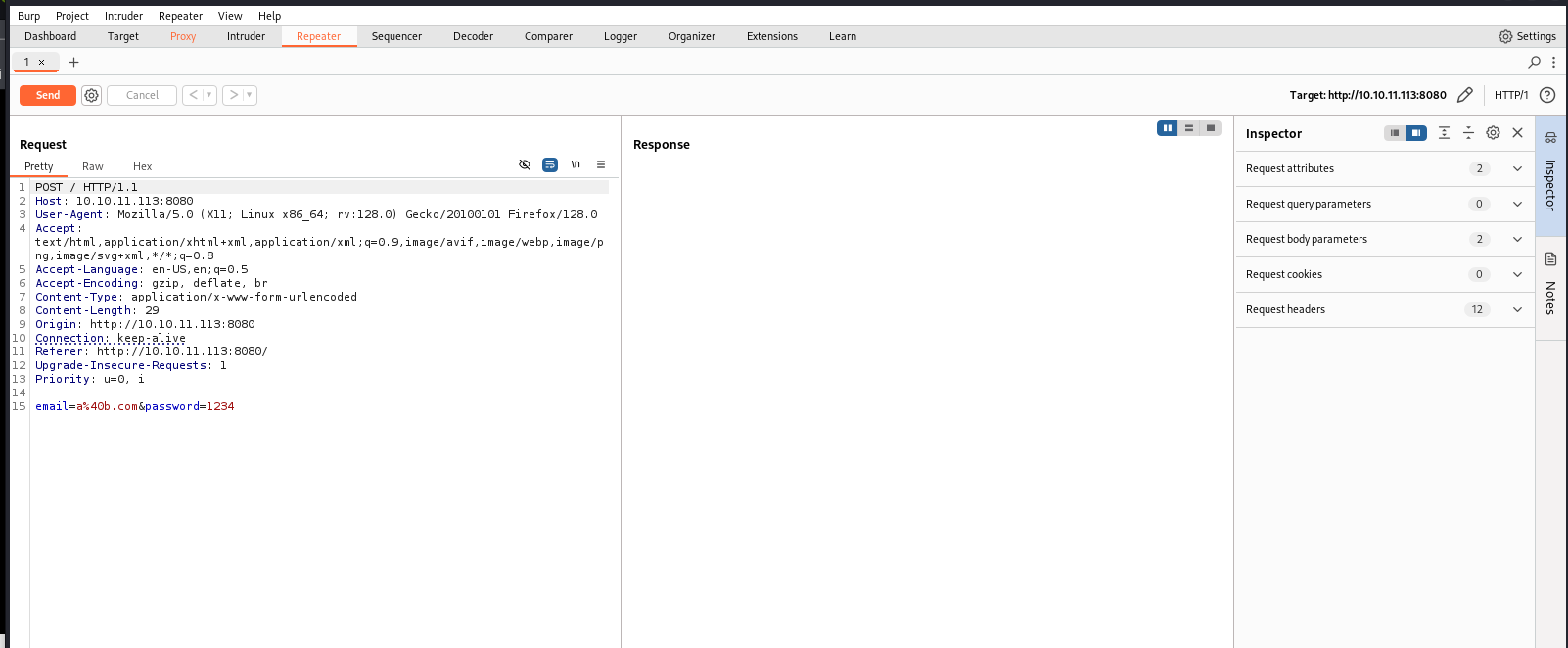
# EXPLOIT

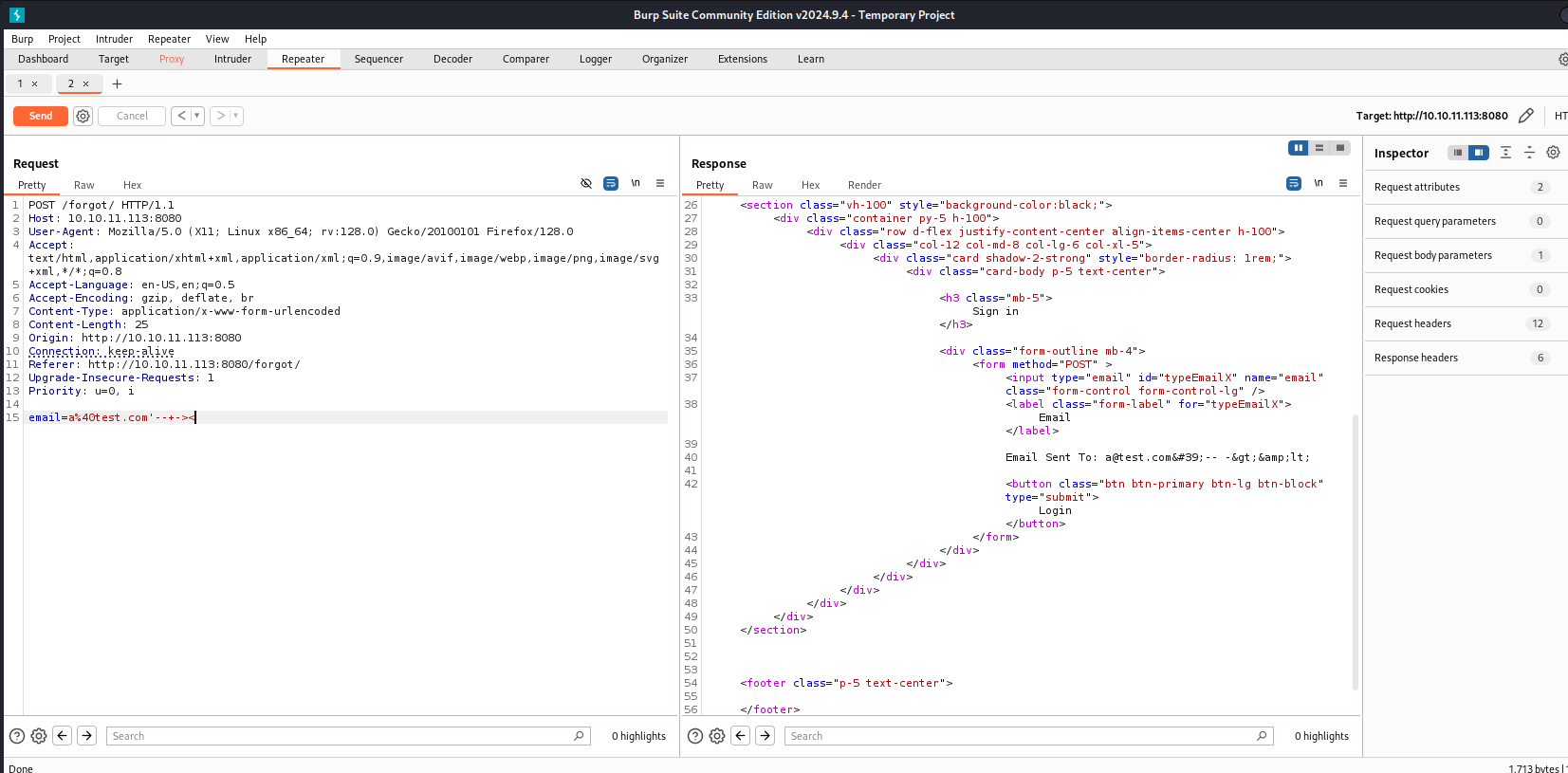
## FLAG 1

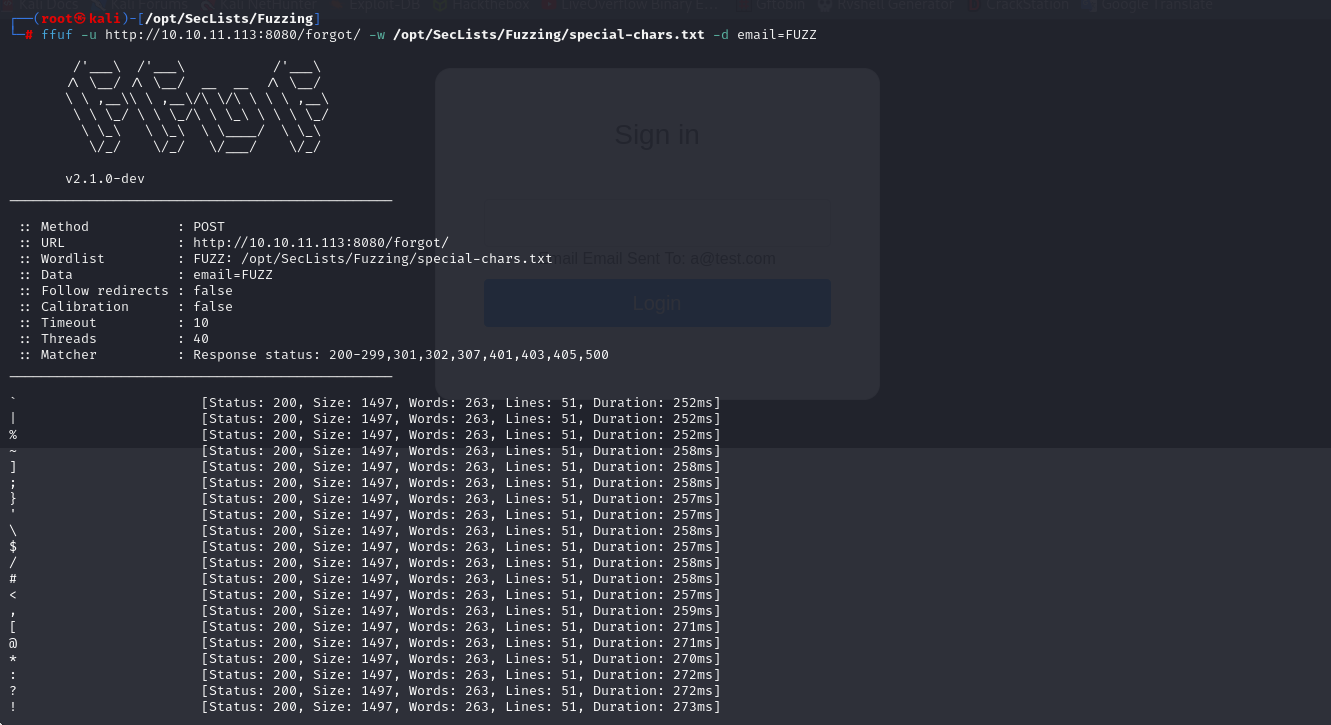
Access Webserver via port 8080 and try some basic sql injection and nothing happened



Use Burpsuite to easily try multiple logins







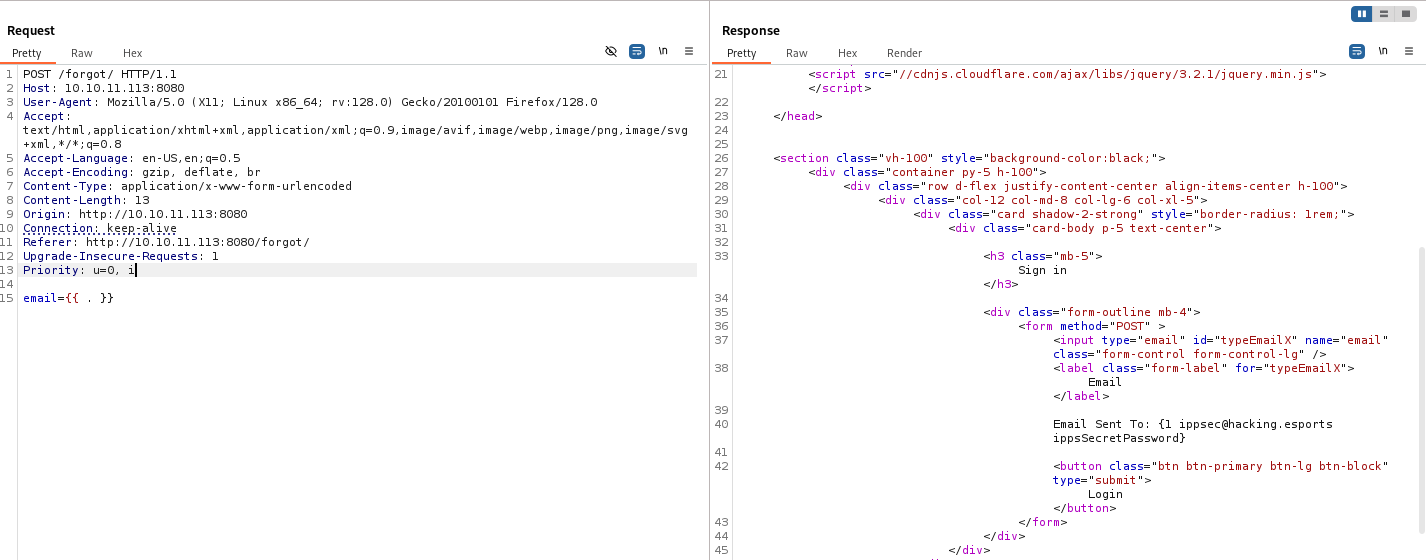
Affer digging around for a while, I found some special here. We can try with brackets

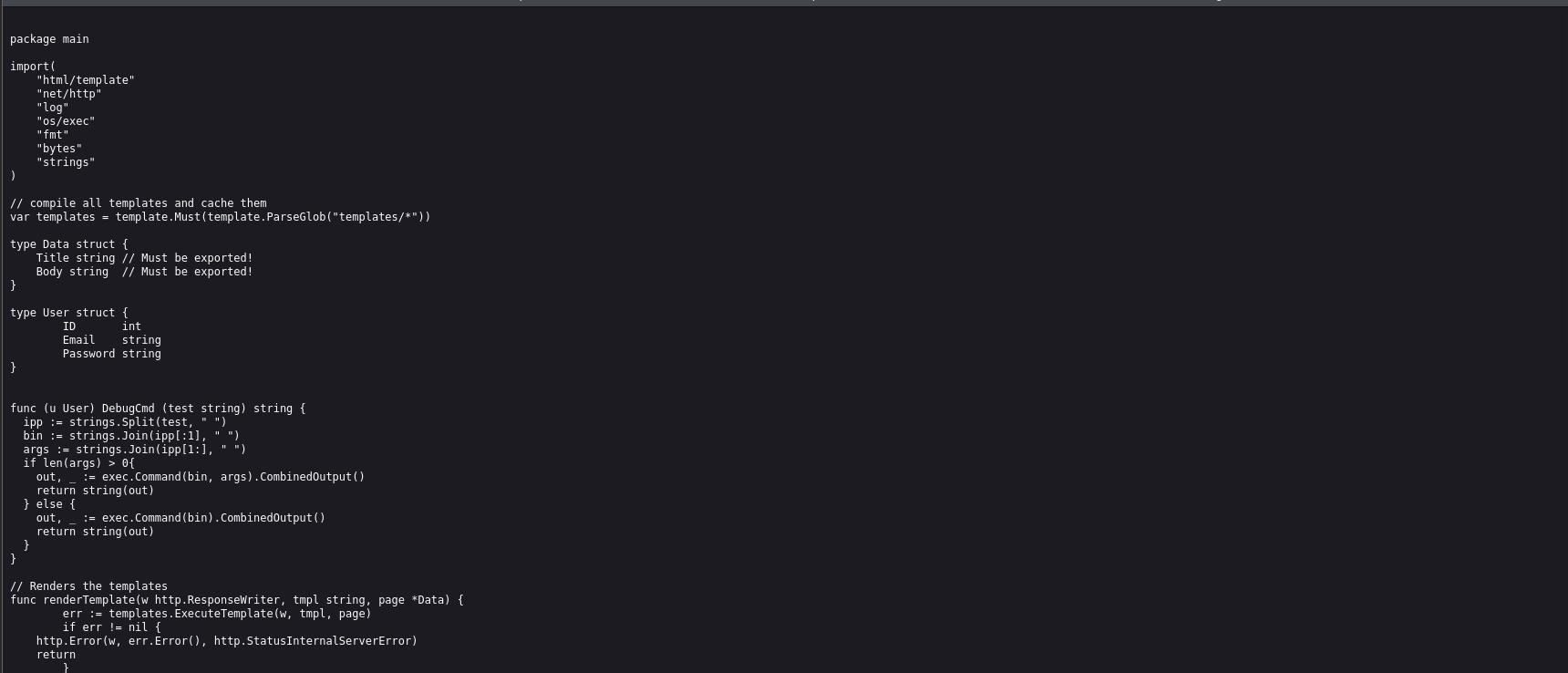


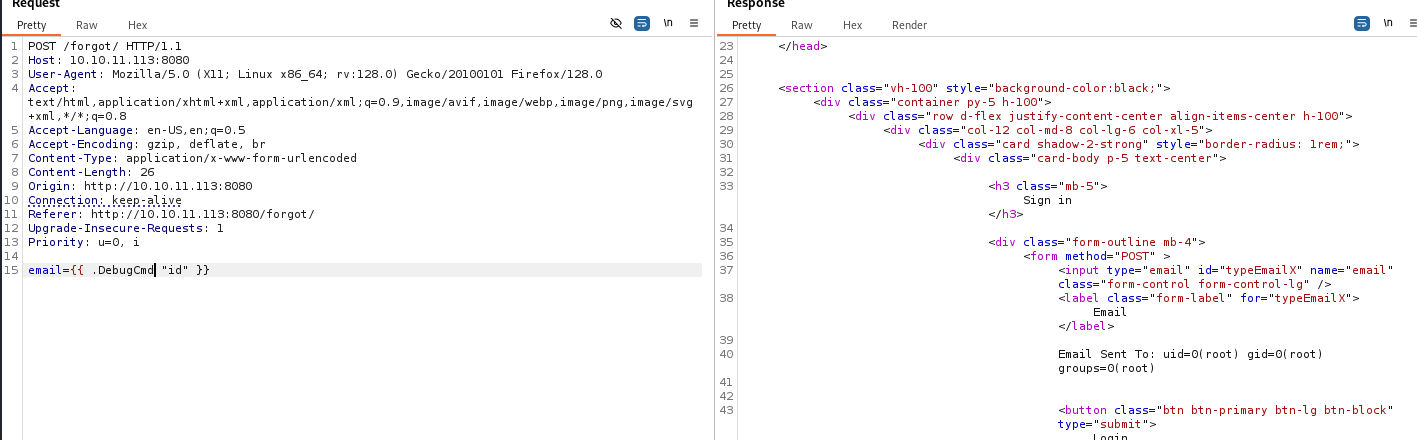
*ffuf -u http://10.10.11.113:8080/forgot/ -w /opt/SecLists/Fuzzing/special-chars.txt -d email=FUZZFUZZ -mc all -H 'Content-Type: application/x-www-form-urlencoded' -fs 149*

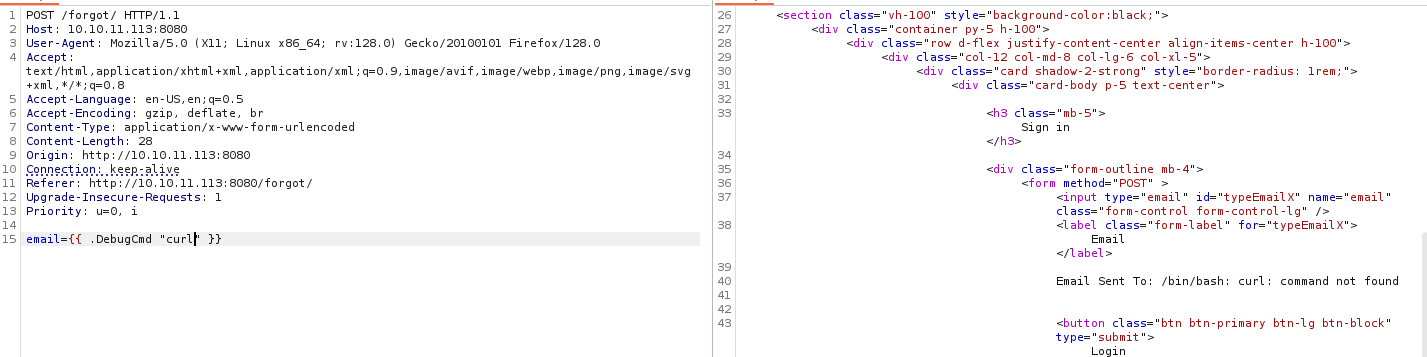
After searching google with golang stti, we know some method about using SSTI in Golang to bypass XSS Filters. However, the key piece of information is that it is possible to dump variables with {{ . }}, doing this will dump the credentials

<https://www.onsecurity.io/blog/go-ssti-method-research/>

**

**

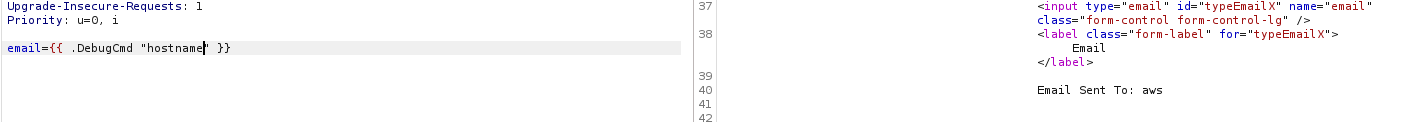
**

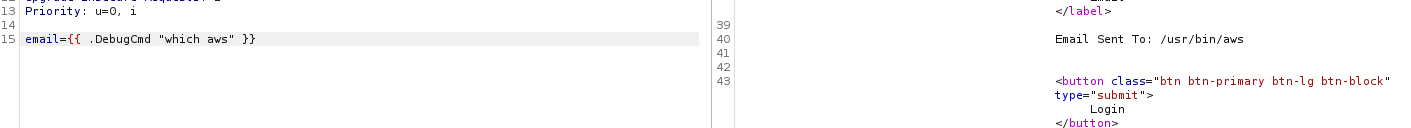
**

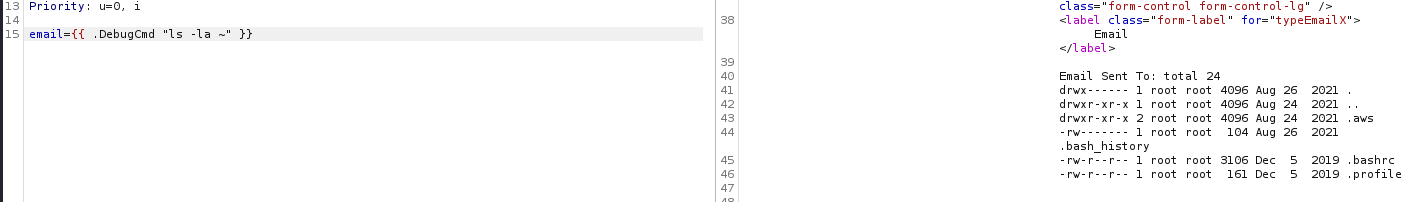
**

**

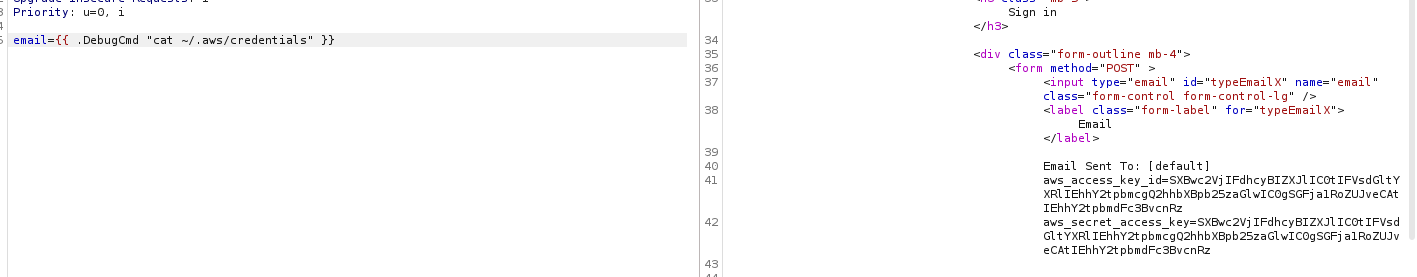
**

**

**

**

AWS (Amazon Web Services) is a cloud computing platform that provides a variety of services such as storage, servers, and databases. S3 (Simple Storage Service) is AWS's object storage service, which allows for highly scalable storage and retrieval of data over the internet.

**

**List all buckets:**

*aws s3 ls*

Displays a list of existing buckets.

**Create a new bucket:**

*aws s3 mb s3://bucket-name*

Creates a new bucket with the specified name.

**Delete a bucket**:

*aws s3 rb s3://bucket-name --force*

Delete the bucket and all data inside (use --force to force deletion).

**Upload a file to a bucket:**

*aws s3 cp file.txt s3://bucket-name/*

Upload the file.txt to the specified bucket.

**Download a file from the bucket to your computer:**

*aws s3 cp s3://bucket-name/file.txt .*

Download the file from the bucket to the current directory.

**Sync local directory to bucket:**

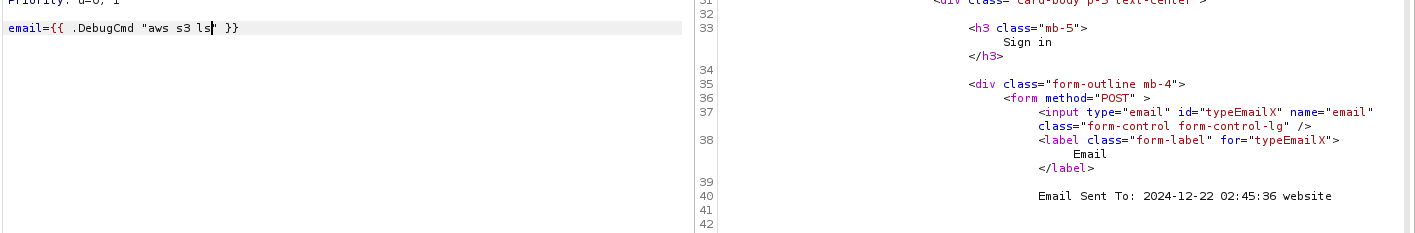
*aws s3 sync ./local-dir s3://bucket-name/*

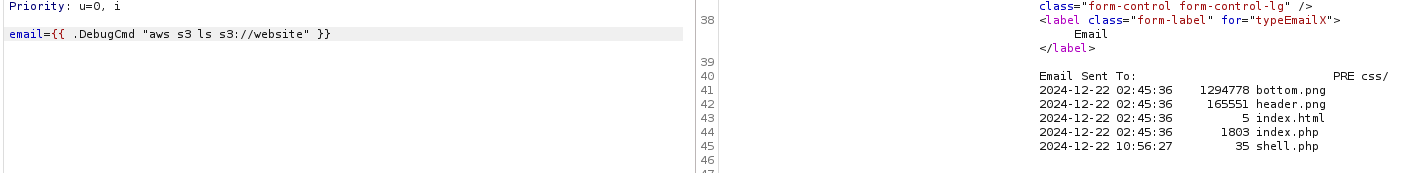
Sync content from local directory to bucket.

**List files in bucket:**

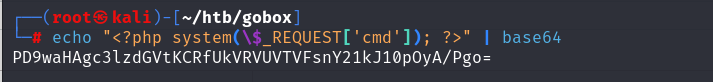
*aws s3 ls s3://bucket-name/*

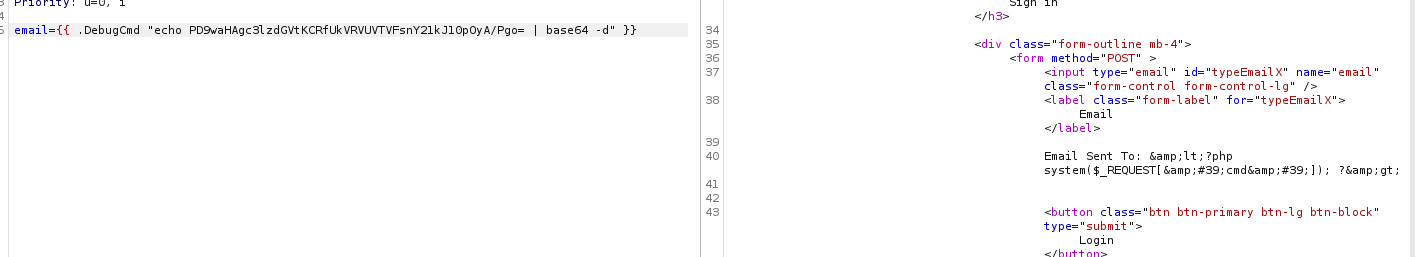
Display list of files in bucket.

**

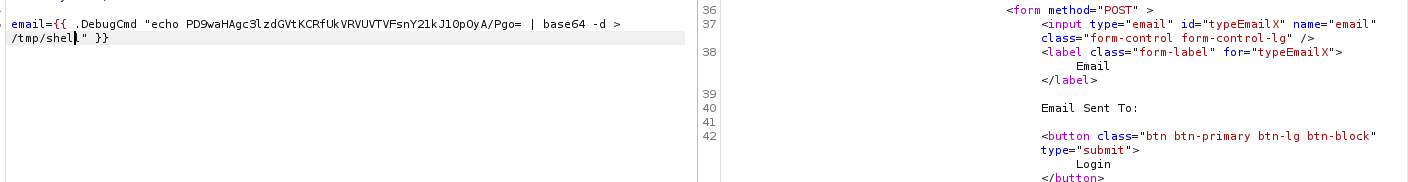
**

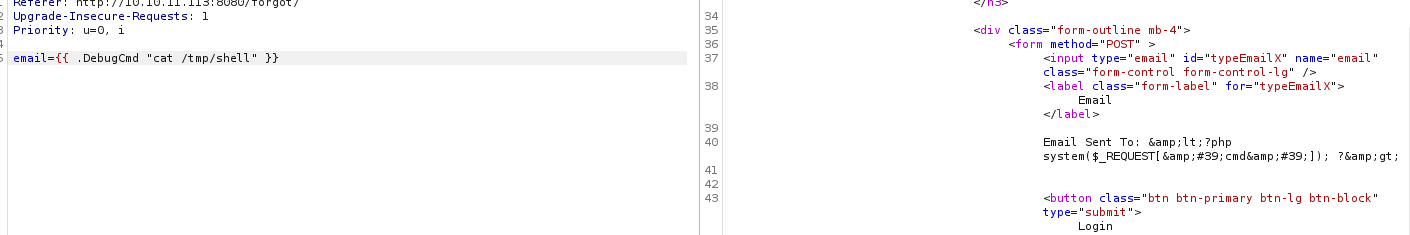
We generate a php code, this PHP code executes a system command from a user-supplied cmd parameter via URL or form. It can lead to RCE (Remote Code Execution) vulnerability if left unchecked. For security reasons, it is not recommended to use user input directly in the system() function. We encode it with base64.

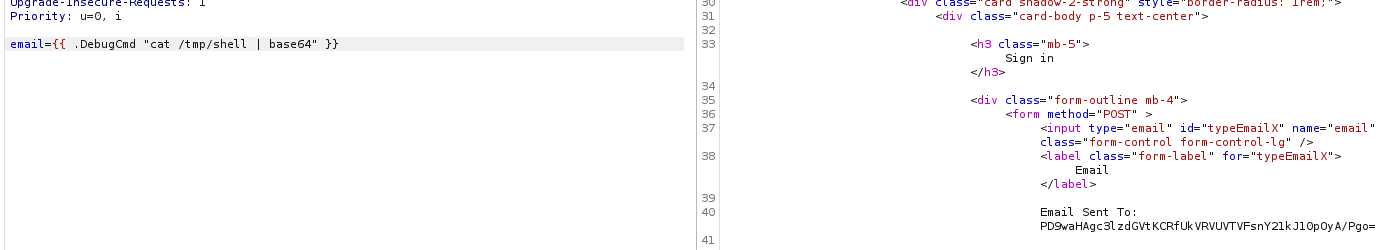
**

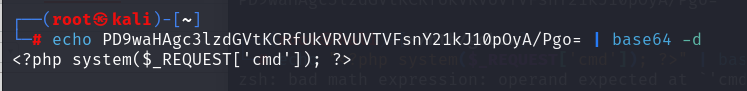
**

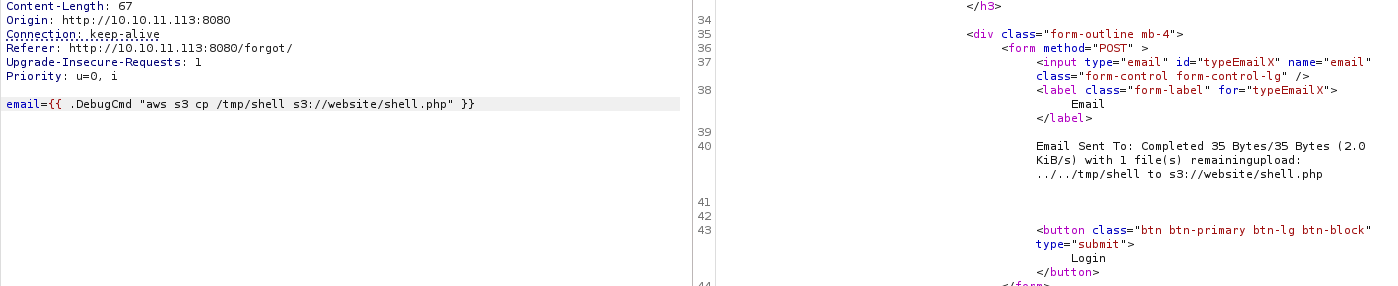
We save php code in a file named shell.php in buckets named website

**

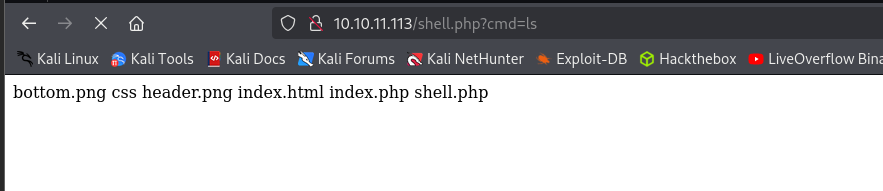
**

**

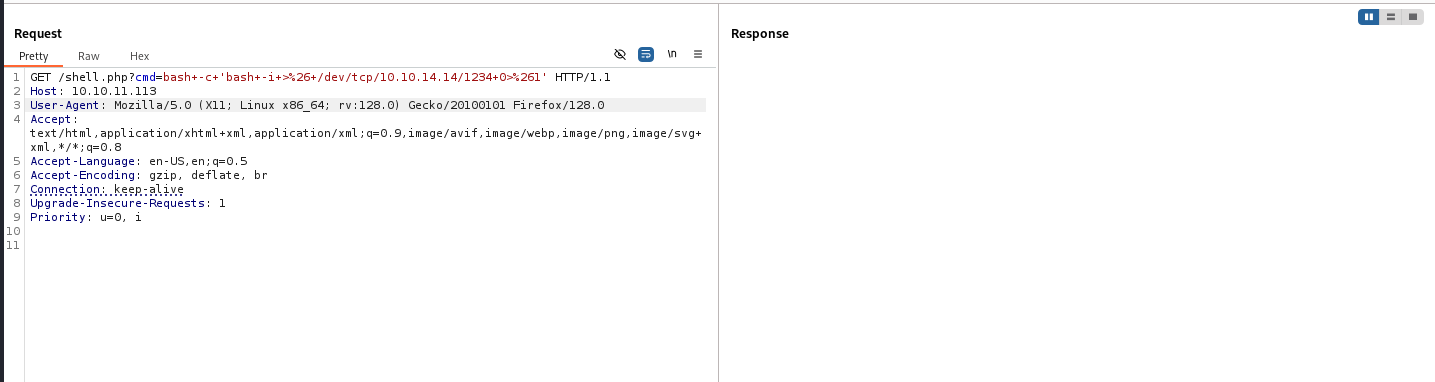
**

**

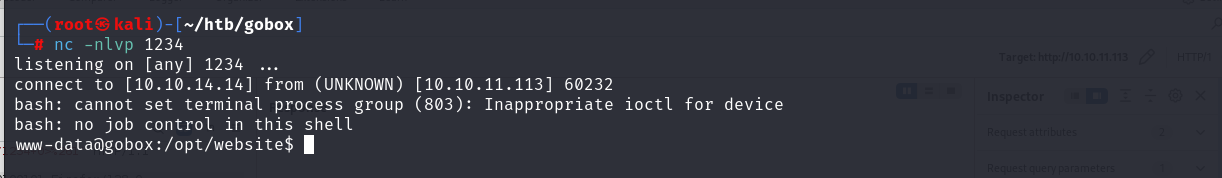
And we have a RCE

**

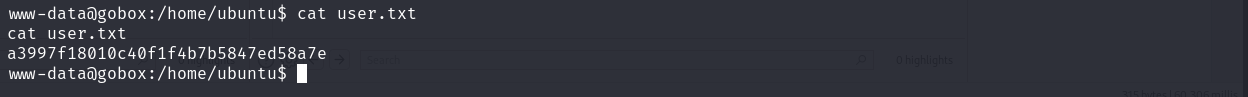
Next thing, just run a reverse shell and listen on the attacking machine



And we have a shell

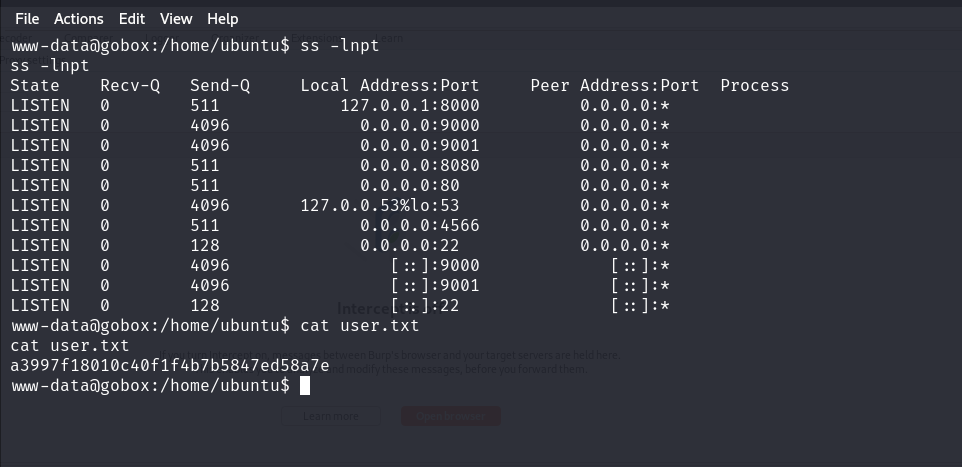


Take the first flag

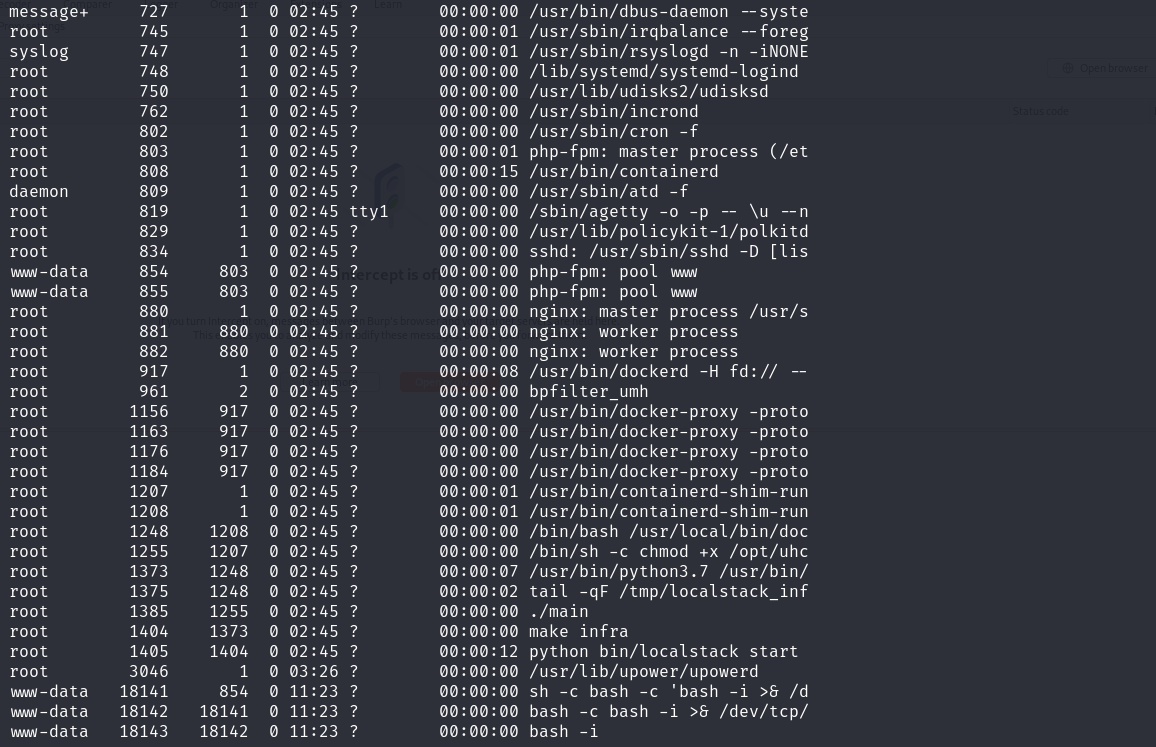


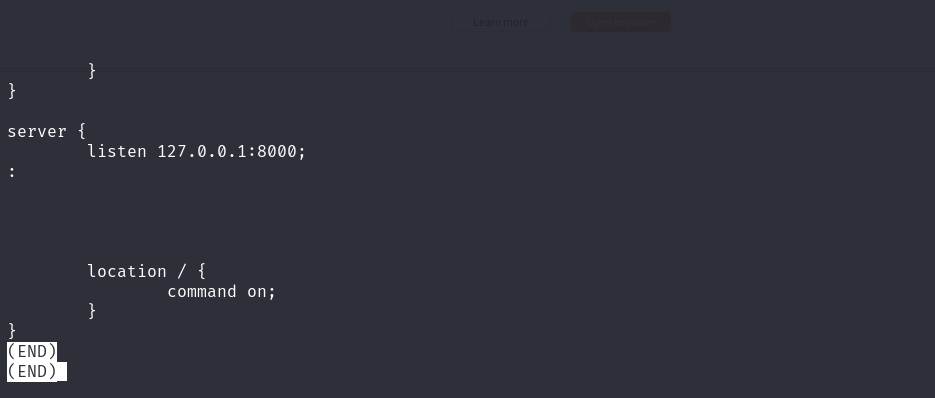
## FLAG 2

Running *ss –lnpt* reveals many ports are listening on the box, specially 127.0.0.1:8000 sticks out because it is the only port listening on localhost

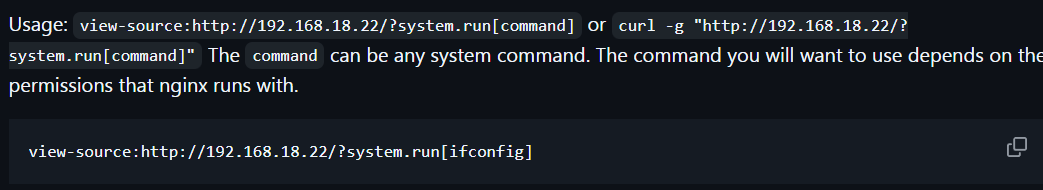


Making a web request to that port reveals that it is an HTTP Server and running ps -ef shows that nginx is running not apache. The NGINX Configuration has a weird option (Command: on) that is not used in any nginx documentation.

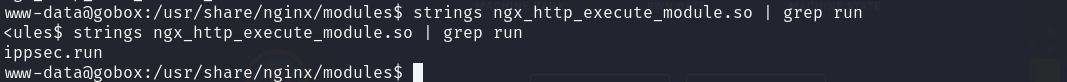




Searching Google for "Command: On" Nginx Github should bring up the NginxExecute module ( <https://github.com/limithit/NginxExecute>). Additionally, looking into /usr/share/nginx/modules/ reveals the name of the module.



This GitHub page states that curl -g "http://192.168.18.22/?system.run[command]" can be used to execute commands. However, when attempting to do it with our IP/Port, it does not work. Running strings against the nginx module and grepping for "run", reveals the name was changed from system to ippsec and updating our command will allow for escalation to root





Take the second Flag

