

To start the database:

```
sudo docker compose -f 'docker-compose.yml' up -d --build
```

How to achieve the web server now:

```
go build -o ./bin/main ./cmd/ && sudo setcap 'cap_net_bind_service=+ep' ./bin/main  
&& ./bin/main
```

To setup the environment:

```
sudo apt update  
sudo apt install snapd -y  
  
sudo apt install git -y  
  
sudo snap install go --classic  
sudo snap install code --classic  
sudo snap install zaproxy --classic  
sudo snap install docker  
sudo apt install graphviz -y  
go install github.com/google/pprof@latest
```

To stop the web server:

```
Ctrl + c
```

Set up the web server:

You must manually register with the account name: tempuser, other part is not that matter, but consider the password and recover are: comp4334password / comp4334recover

You must try login with the account, if the login page show error: Fail to generate the token

Stop the web server

Go to the :

```
📦 cmd
  |- 📂 api
  .....
  L  📄 main.go <--
```

Manually set the drop_flag to true

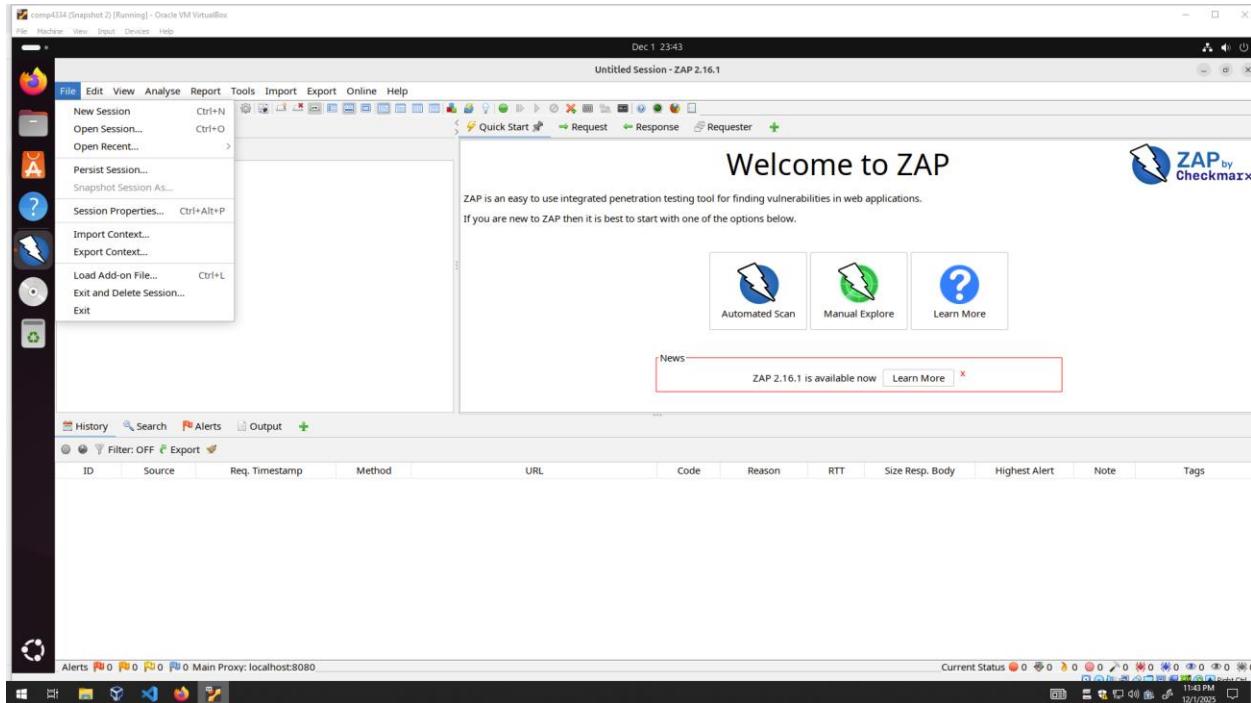
Restart the web server, due to the database do not having the key that insert into database.

Such that drop the database and restart the initialization database process.

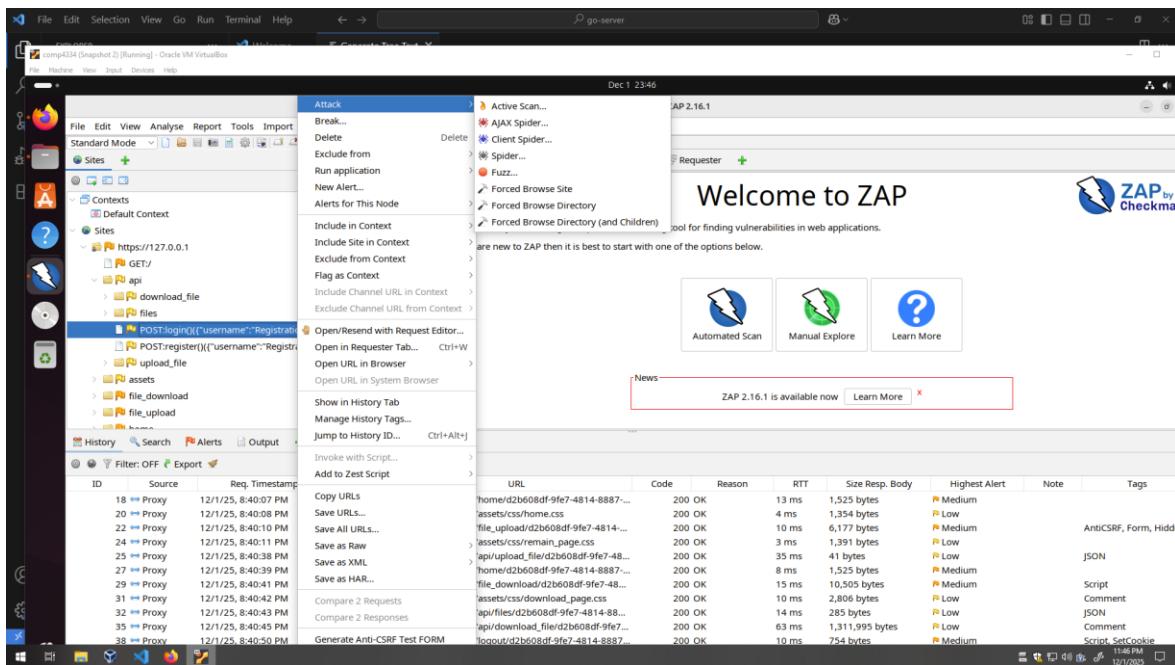
Open the zap:

Import the session:

File -> open the session , choose the session



Choose the fuzz:

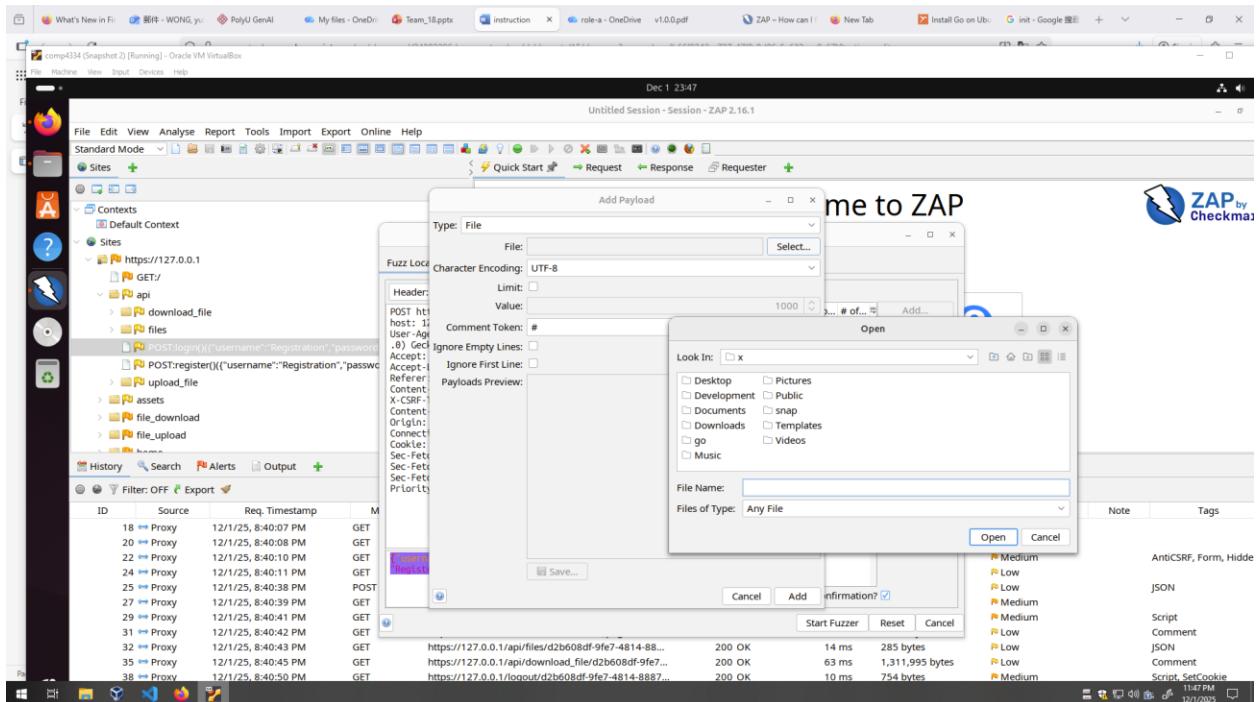


The screenshot shows the ZAP interface with a session titled "Untitled Session - Session - ZAP 2.16.1". In the center, a "Fuzzer" dialog box is open, displaying a payload for a POST request to `https://127.0.0.1/api/login`. The payload is set to `{"username": "Registration", "password": "Registration", "recover": "Registration"}`. The "Fuzz Locations" dropdown is set to "Body: Text". The "Fuzz Locations" table lists various fields with their current values and processor counts. At the bottom of the dialog, there is a "Start Fuzzer" button.

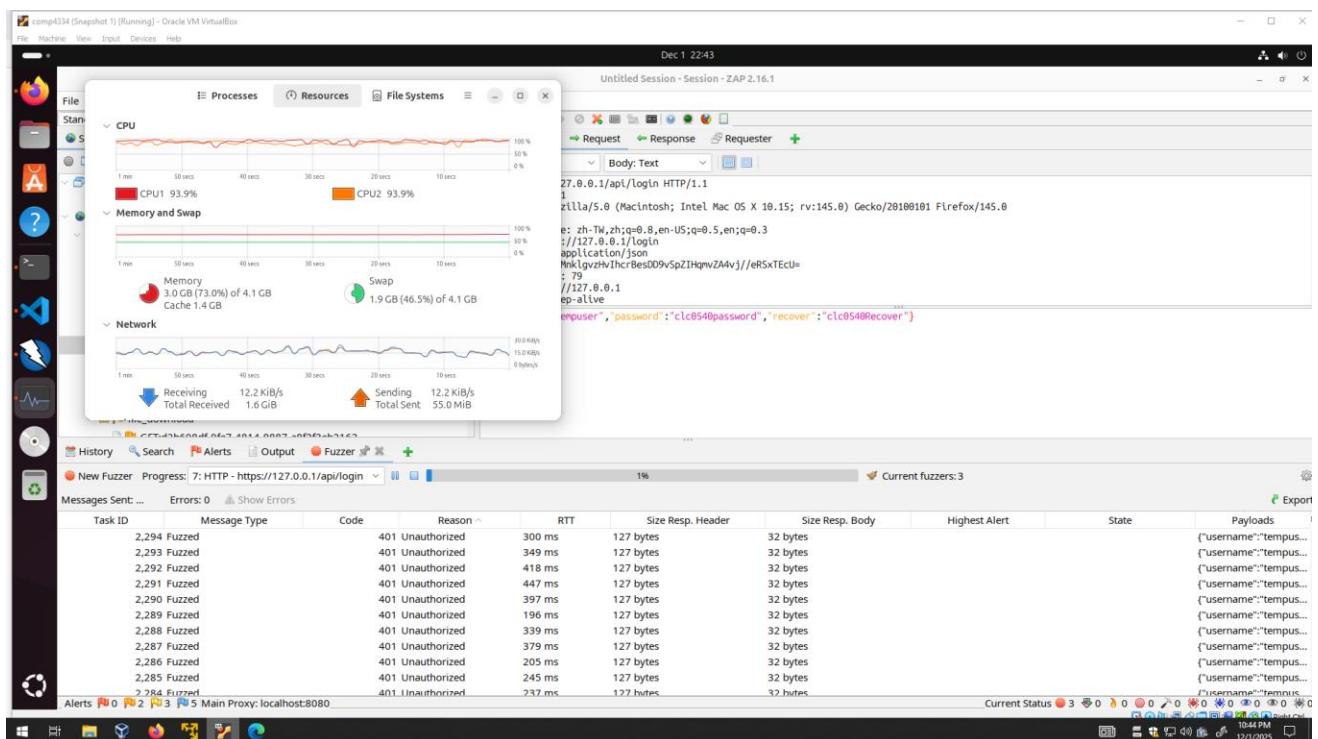
, add

The screenshot shows the ZAP interface with a session titled "Untitled Session - Session - ZAP 2.16.1". In the center, a "Payloads" dialog box is open, displaying a payload for a POST request to `https://127.0.0.1/api/login`. The payload is set to `{"username": "Registration", "password": "Registration", "recover": "Registration"}`. The "Payloads" dialog has tabs for "Header", "Body", and "Footer". The "Body" tab is selected, showing the value: "Value: {"username": "Registration", "password": "Registration", "recover": "Registration"}". The "Payloads" table lists various fields with their current values and processor counts. At the bottom of the dialog, there are "Cancel" and "OK" buttons.

, import the txt payload file



, start the fuzzer, Result



```

Dec 1 22:45
File Edit Selection View Go Run Terminal Help
logs > system.log
logs > security_events.csv U Extension: Rainbow CSV
main.go M utils.go M method_helper.go M system.log X security_events.csv M json_router.go M
GO-SERVER
cmd
router
json_handler
web_server
internal
certs
db
store
logs
system.log
security_events.csv
outline
timeline
go
package outline
Ln 4075, Col 97 10:46 PM 12/1/2025

Dec 1 22:45
File Edit Selection View Go Run Terminal Help
logs > security_events.csv > data
logs > system.log
logs > security_events.csv U Extension: Rainbow CSV
main.go M utils.go M method_helper.go M system.log X security_events.csv M json_router.go M
GO-SERVER
cmd
router
json_handler
web_server
internal
certs
db
store
logs
system.log
security_events.csv
outline
timeline
go
package outline
Ln 4075, Col 97 10:46 PM 12/1/2025

```

The terminal windows show log files for a Go application. The top window displays the 'system.log' file, which contains numerous entries of failed login attempts. The bottom window displays the 'security_events.csv' file, which provides a structured log of these events. Both logs indicate that most failed logins are due to invalid credentials.

analyze the data:

sudo apt install graphviz -y

go install github.com/google/pprof@latest

go tool pprof -http=:6060 cpu/heap_* .pprof