whoami

hostname

ss -tlnp | grep -E '2200|2222'

uptime

df -h /

free -h

swapon -- show

```
root@HoneypotVM:~# whoami
hostname
ss -tlnp | grep -E '2200|2222'
uptime
df -h /
free -h
swapon --show
root
HoneypotVM
LISTEN 0      128          0.0.0.0:2200       0.0.0.0:*    users:(("sshd",pi
d=2741251,fd=3))
LISTEN 0      50           0.0.0.0:2222       0.0.0.0:*    users:(("twistd",
pid=1479,fd=11))
 20:59:31 up 249 days, 16:45,  6 users,  load average: 0.01, 0.06, 0.09
Filesystem      Size  Used Avail Use% Mounted on
/dev/root        30G   23G  7.6G  75% /
               total        used        free      shared  buff/cache   avail
able
Mem:           892Mi       579Mi        86Mi       3.1Mi       377Mi       3
12Mi
Swap:          2.0Gi       177Mi       1.8Gi
NAME      TYPE SIZE   USED PRIO
/swapfile file   2G 177.7M   -2
root@HoneypotVM:~#
```

Separation between real SSH and honeypot SSH

VM stability (uptime, load)

Swap configured and active

Disk under control

cd /home/azureuser/cowrie

ps aux | grep cowrie | grep -v grep

```
root@HoneypotVM:~# cd /home/azureuser/cowrie
ps aux | grep cowrie | grep -v grep
azureus+    1479  0.1  4.7 105708 43856 ?        S     2025 497:09 /home/azu
reuser/cowrie/cowrie-env/bin/python3.12 /home/azureuser/cowrie/cowrie-env/bi
n/twistd --umask=0022 --pidfile=var/run/cowrie.pid --logger cowrie.python.lo
gfile.logger cowrie
root@HoneypotVM:/home/azureuser/cowrie# ||
```

ls -lh var/log/cowrie/cowrie.json

tail -n 5 var/log/cowrie/cowrie.json

```
root@HoneypotVM:/home/azureuser/cowrie# |ls -lh var/log/cowrie/cowrie.json
tail -n 5 var/log/cowrie/cowrie.json
-bash: syntax error near unexpected token `|'
{"eventid":"cowrie.login.failed","username":"hadi","password":"123456","mess
age":"login attempt [hadi/123456] failed","sensor":"HoneypotVM","timestamp":
"2026-01-09T21:04:00.741178Z","src_ip":"194.59.31.74","session":"1509244b5da
8"}
{"eventid":"cowrie.session.closed","duration":"5.9","message":"Connection lo
st after 5.9 seconds","sensor":"HoneypotVM","timestamp":"2026-01-09T21:04:02
.617813Z","src_ip":"194.59.31.74","session":"1509244b5da8"}
{"eventid":"cowrie.session.connect","src_ip":"120.48.60.44","src_port":50772
,"dst_ip":"10.0.0.4","dst_port":2222,"session":"c6670ff788a6","protocol":"ss
h","message":"New connection: 120.48.60.44:50772 (10.0.0.4:2222) [session: c
6670ff788a6]","sensor":"HoneypotVM","timestamp":"2026-01-09T21:04:07.415878Z
"}
{"eventid":"cowrie.client.version","version":"SSH-2.0-libssh_0.11.1","messag
e":"Remote SSH version: SSH-2.0-libssh_0.11.1","sensor":"HoneypotVM","timest
amp":"2026-01-09T21:04:07.416826Z","src_ip":"120.48.60.44","session":"c6670f
f788a6"}
{"eventid":"cowrie.client.kex","hassh":"03a80b21afa810682a776a7d42e5e6fb","h
asshAlgorithms":"curve25519-sha256,curve25519-sha256@libssh.org,ecdh-sha2-ni
stp256,ecdh-sha2-nistp384,ecdh-sha2-nistp521,diffie-hellman-group18-sha512,d
iffie-hellman-group16-sha512,diffie-hellman-group-exchange-sha256,diffie-hel
lman-group14-sha256,ext-info-c,kex-strict-c-v00@openssh.com;chacha20-poly130
5@openssh.com,aes256-gcm@openssh.com,aes128-gcm@openssh.com,aes256-ctr,aes19
```

grep -a "login attempt" cowrie.log | wc -l

The honeypot isolates attacker interaction on port 2222, while administrative access is restricted to port 2200.

grep -a '"eventid": "cowrie.login.failed"' \

/home/azureuser/cowrie/var/log/cowrie/cowrie.json | wc -l

```
root@HoneypotVM:/home/azureuser/cowrie# grep -a '"eventid": "cowrie.login.fa
iled"' \
/home/azureuser/cowrie/var/log/cowrie/cowrie.json | wc -l
0
root@HoneypotVM:/home/azureuser/cowrie#
```

grep -a '"src_ip"' /home/azureuser/cowrie/var/log/cowrie/cowrie.json \

| jq -r '.src_ip' | sort | uniq | wc -l

```
root@HoneypotVM:/home/azureuser/cowrie# grep -a '"src_ip"' /home/azureuser/c
owrie/var/log/cowrie/cowrie.json \
| jq -r '.src_ip' | sort | uniq | wc -l
198
root@HoneypotVM:/home/azureuser/cowrie#
```

Cowrie was configured in JSON logging mode, enabling structured ingestion into Loki and Grafana.

Promtail forwards Cowrie JSON logs to Loki for centralized analysis.

System & isolation

ss -tuln

uptime

df -h /

free -h

swapon –show

Cowrie health

ls -lh /home/azureuser/cowrie/var/log/cowrie/cowrie.json

tail -n 5 /home/azureuser/cowrie/var/log/cowrie/cowrie.json

Attack evidence (JSON)

grep -a '"cowrie.login.failed"' cowrie/var/log/cowrie/cowrie.json | wc -l

Total login attempts

jq 'select(.username != null)' /home/azureuser/cowrie/var/log/cowrie/cowrie.json | wc -l\

```
root@HoneypotVM:/home/azureuser# jq 'select(.username != null)' /home/azureu
ser/cowrie/var/log/cowrie/cowrie.json | wc -l
121226
root@HoneypotVM:/home/azureuser#
```

Unique attacker IPs

jq -r '.src_ip' /home/azureuser/cowrie/var/log/cowrie/cowrie.json \

| sort | uniq | wc -l

```
root@HoneypotVM:/home/azureuser# jq -r '.src_ip' /home/azureuser/cowrie/var/
log/cowrie/cowrie.json \
| sort | uniq | wc -l
198
root@HoneypotVM:/home/azureuser#
```

Top usernames

```
root@HoneypotVM:/home/azureuser# jq -r '.username' /home/azureuser/cowrie/va
r/log/cowrie/cowrie.json \
| grep -v null | sort | uniq -c | sort -nr | head
   6616 root
    292 ubuntu
    285 at
    285 ansible
    213 admin
    159 user
    158 user1
    154 test
    124 git
    123 deploy
root@HoneypotVM:/home/azureuser#
```

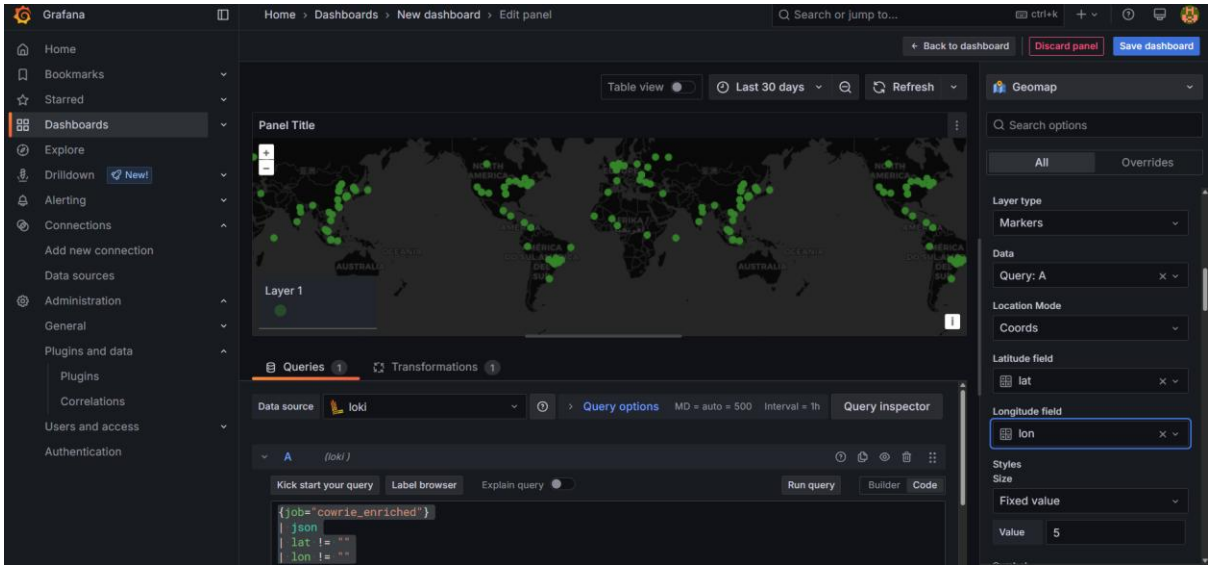
Count credential attempts

jq 'select(.username != null)' /home/azureuser/cowrie/var/log/cowrie/cowrie.json | wc -l

```
root@HoneypotVM:/home/azureuser# jq 'select(.username != null)' /home/azureu
ser/cowrie/var/log/cowrie/cowrie.json | wc -l
121436
root@HoneypotVM:/home/azureuser#
```

Cowrie was configured to log events in JSON format. Due to version-specific event identifiers, login attempts were identified by the presence of credential fields (username, password) rather than fixed event names.

Structured JSON logs were parsed using jq to extract attacker behavior, credentials, and source IPs. This approach avoids brittle string matching and supports scalable ingestion into Loki.



These points show the geographic origin of SSH login attempts captured by my Cowrie honeypot.
IPs are enriched with GeoIP data and visualized in Grafana using Loki as the log backend.