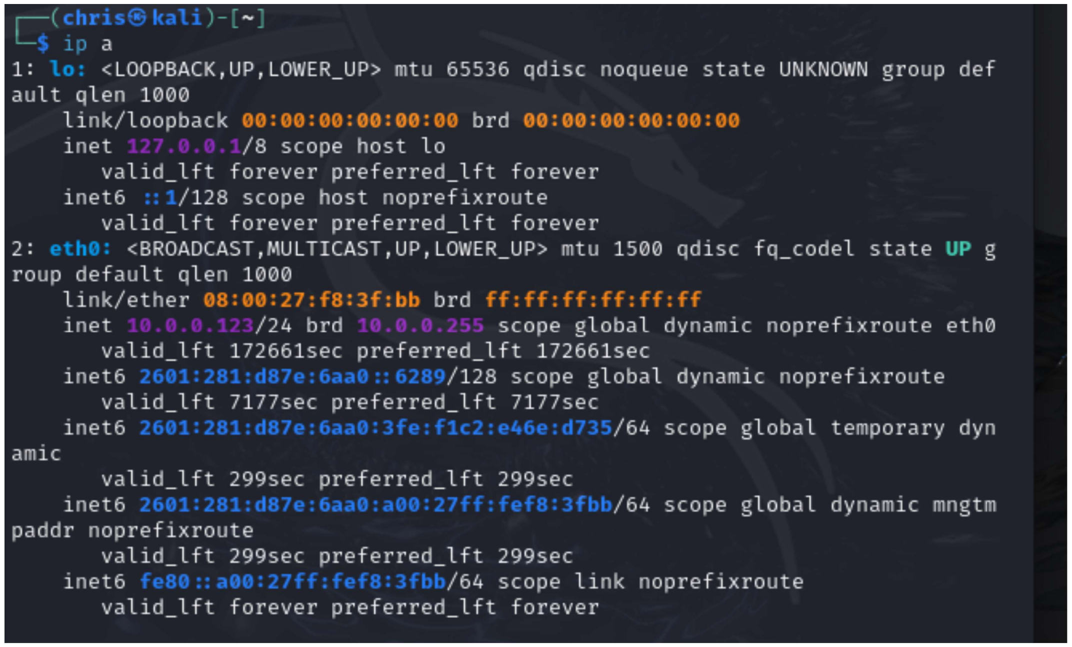
Beginner Bash

Before we begin lets check our IP because we will want to know this for the script we are about to write.

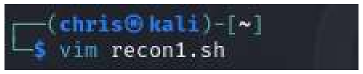


```
┌──(chris㉿kali)-[~]
└─$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group def
ault qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP g
roup default qlen 1000
    link/ether 08:00:27:f8:3f:bb brd ff:ff:ff:ff:ff:ff
    inet 10.0.0.123/24 brd 10.0.0.255 scope global dynamic noprefixroute eth0
       valid_lft 172661sec preferred_lft 172661sec
    inet6 2601:281:d87e:6aa0::6289/128 scope global dynamic noprefixroute
       valid_lft 7177sec preferred_lft 7177sec
    inet6 2601:281:d87e:6aa0:3fe:f1c2:e46e:d735/64 scope global temporary dyn
amic
       valid_lft 299sec preferred_lft 299sec
    inet6 2601:281:d87e:6aa0:a00:27ff:fef8:3fbb/64 scope global dynamic mngtm
paddr noprefixroute
       valid_lft 299sec preferred_lft 299sec
    inet6 fe80::a00:27ff:fef8:3fbb/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
```

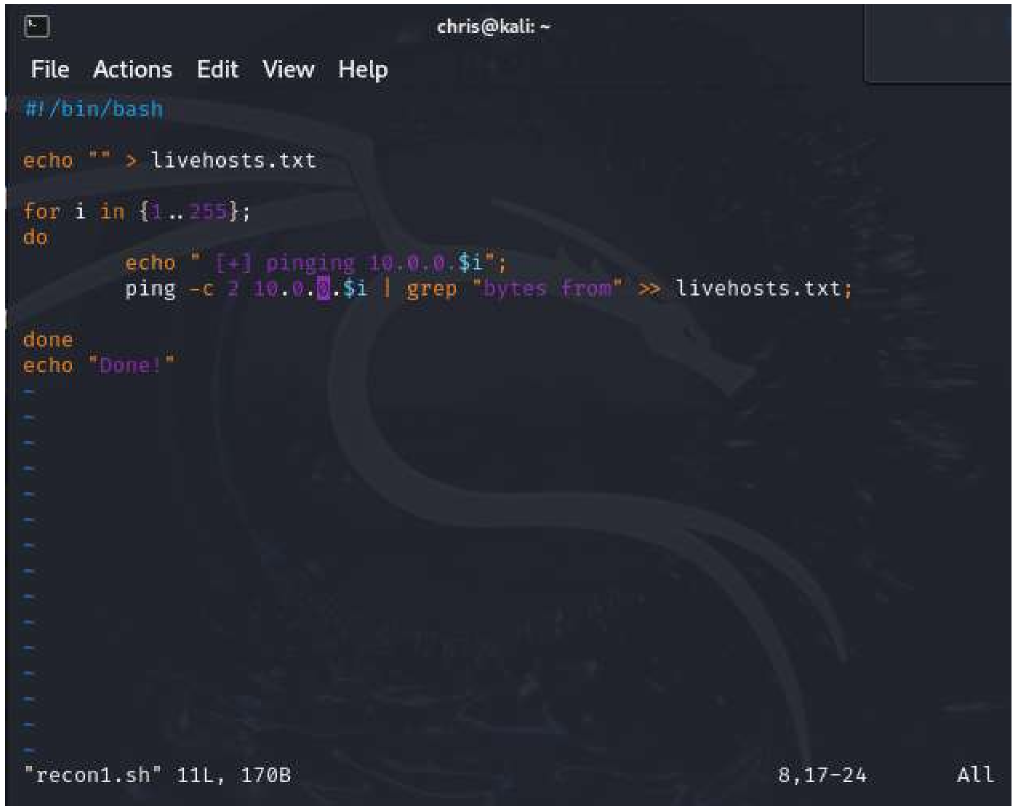
Now lets enter VIM to draft the bash script and call it recon1.sh

Lets setup a simple ping script for our host and see what we find.

Please note all bash scripts must start with #!/bin/bash



```
┌──(chris㉿kali)-[~]
└─$ vim recon1.sh
```



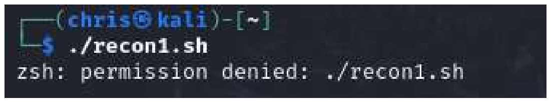
```
#!/bin/bash

echo "" > livehosts.txt

for i in {1..255};
do
        echo " [+] pinging 10.0.0.$i";
        ping -c 2 10.0.0.0.$i | grep "bytes from" >> livehosts.txt;

done
echo "Done!"
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
"recon1.sh" 11L, 170B                                  8,17-24       All
```

Lets run the script: "./recon1.sh"

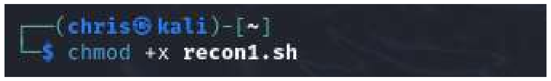
What did we forget…

Yes, privileges need to be set to executable. Lets do that fast using "chmod"

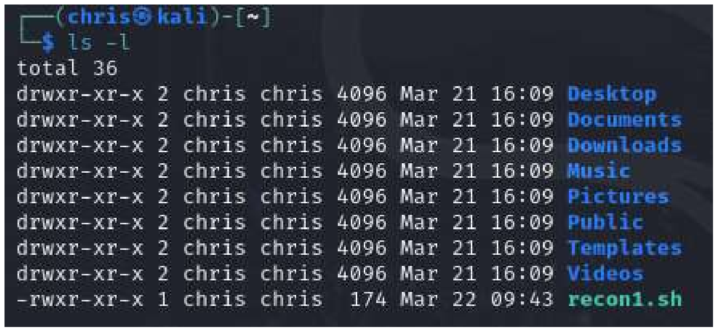
Run it again and we should see it start running.



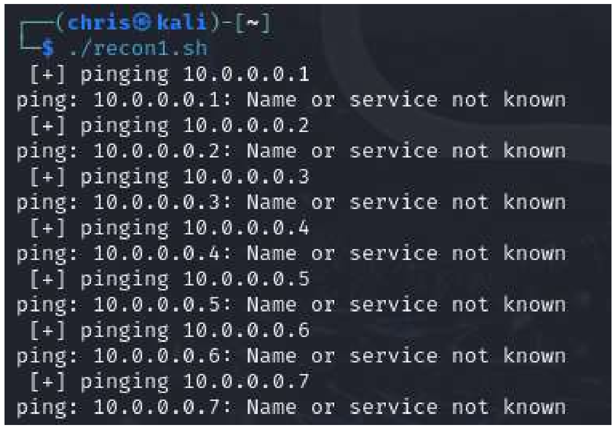
```
┌──(chris㉿kali)-[~]
└─$ ./recon1.sh
zsh: permission denied: ./recon1.sh
```



```
┌──(chris㉿kali)-[~]
└─$ chmod +x recon1.sh
```



```
┌──(chris㉿kali)-[~]
└─$ ls -l
total 36
drwxr-xr-x 2 chris chris 4096 Mar 21 16:09 Desktop
drwxr-xr-x 2 chris chris 4096 Mar 21 16:09 Documents
drwxr-xr-x 2 chris chris 4096 Mar 21 16:09 Downloads
drwxr-xr-x 2 chris chris 4096 Mar 21 16:09 Music
drwxr-xr-x 2 chris chris 4096 Mar 21 16:09 Pictures
drwxr-xr-x 2 chris chris 4096 Mar 21 16:09 Public
drwxr-xr-x 2 chris chris 4096 Mar 21 16:09 Templates
drwxr-xr-x 2 chris chris 4096 Mar 21 16:09 Videos
-rwxr-xr-x 1 chris chris  174 Mar 22 09:43 recon1.sh
```

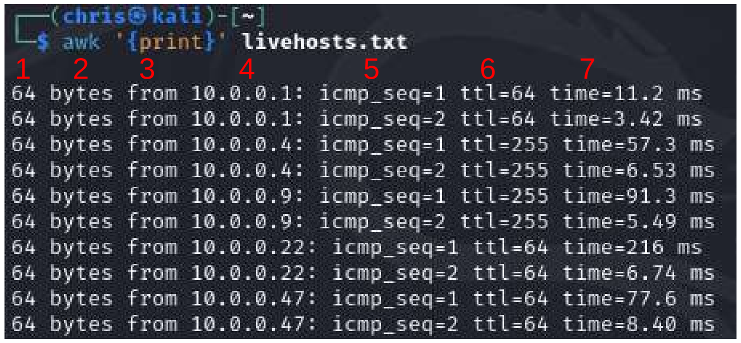


```
┌──(chris㉿kali)-[~]
└─$ ./recon1.sh
 [+] pinging 10.0.0.0.1
ping: 10.0.0.0.1: Name or service not known
 [+] pinging 10.0.0.0.2
ping: 10.0.0.0.2: Name or service not known
 [+] pinging 10.0.0.0.3
ping: 10.0.0.0.3: Name or service not known
 [+] pinging 10.0.0.0.4
ping: 10.0.0.0.4: Name or service not known
 [+] pinging 10.0.0.0.5
ping: 10.0.0.0.5: Name or service not known
 [+] pinging 10.0.0.0.6
ping: 10.0.0.0.6: Name or service not known
 [+] pinging 10.0.0.0.7
ping: 10.0.0.0.7: Name or service not known
```

Lets use awk command to see what our script found.

Awk is a good command for pattern data processing and more.

“awk ‘{print}’ will show us the file. Notice the spacing and think of those as columns.

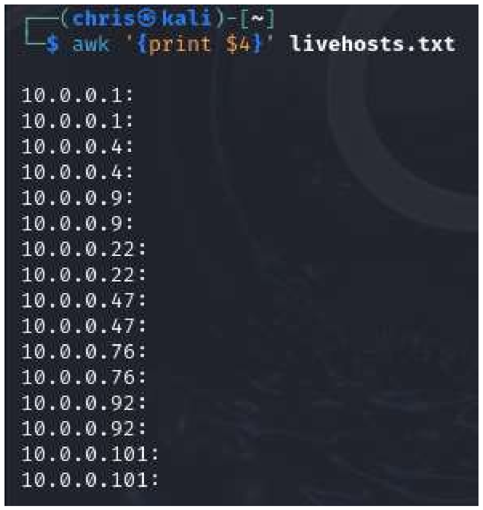


```
┌──(chris㉿kali)-[~]
└─$ awk '{print}' livehosts.txt
1   2     3        4          5          6        7
64 bytes from 10.0.0.1: icmp_seq=1 ttl=64 time=11.2 ms
64 bytes from 10.0.0.1: icmp_seq=2 ttl=64 time=3.42 ms
64 bytes from 10.0.0.4: icmp_seq=1 ttl=255 time=57.3 ms
64 bytes from 10.0.0.4: icmp_seq=2 ttl=255 time=6.53 ms
64 bytes from 10.0.0.9: icmp_seq=1 ttl=255 time=91.3 ms
64 bytes from 10.0.0.9: icmp_seq=2 ttl=255 time=5.49 ms
64 bytes from 10.0.0.22: icmp_seq=1 ttl=64 time=216 ms
64 bytes from 10.0.0.22: icmp_seq=2 ttl=64 time=6.74 ms
64 bytes from 10.0.0.47: icmp_seq=1 ttl=64 time=77.6 ms
64 bytes from 10.0.0.47: icmp_seq=2 ttl=64 time=8.40 ms
```

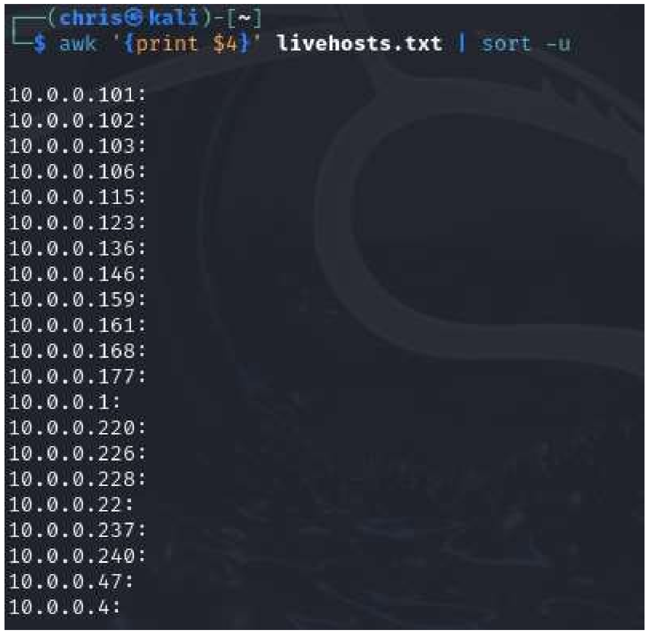
Lets now check and view the IP column (4)

Ok, but looks like we have duplicates so lets sort (hint) those out. We will do that by adding “|” sort -u.

Looks good!



```
┌──(chris㉿kali)-[~]
└─$ awk '{print $4}' livehosts.txt

10.0.0.1:
10.0.0.1:
10.0.0.4:
10.0.0.4:
10.0.0.9:
10.0.0.9:
10.0.0.22:
10.0.0.22:
10.0.0.47:
10.0.0.47:
10.0.0.76:
10.0.0.76:
10.0.0.92:
10.0.0.92:
10.0.0.101:
10.0.0.101:
```

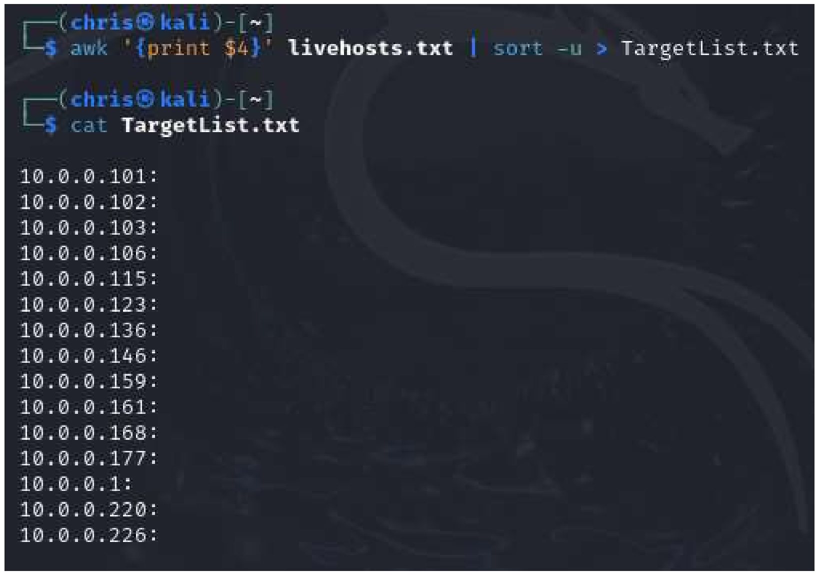


```
┌──(chris㉿kali)-[~]
└─$ awk '{print $4}' livehosts.txt | sort -u

10.0.0.101:
10.0.0.102:
10.0.0.103:
10.0.0.106:
10.0.0.115:
10.0.0.123:
10.0.0.136:
10.0.0.146:
10.0.0.159:
10.0.0.161:
10.0.0.168:
10.0.0.177:
10.0.0.1:
10.0.0.220:
10.0.0.226:
10.0.0.228:
10.0.0.22:
10.0.0.237:
10.0.0.240:
10.0.0.47:
10.0.0.4:
```

Lastly, lets put the results in a separate file so we can use them later on.

We are going to append “>” them to TargetList.txt

Quick check with “cat” and...success!



```
┌──(chris㉿kali)-[~]
└─$ awk '{print $4}' livehosts.txt | sort -u > TargetList.txt

┌──(chris㉿kali)-[~]
└─$ cat TargetList.txt

10.0.0.101:
10.0.0.102:
10.0.0.103:
10.0.0.106:
10.0.0.115:
10.0.0.123:
10.0.0.136:
10.0.0.146:
10.0.0.159:
10.0.0.161:
10.0.0.168:
10.0.0.177:
10.0.0.1:
10.0.0.220:
10.0.0.226:
```