

CS790 Assignment 3

Onions: Are they always tasty? Report

Kavya Gupta (22B1053)

Department of Computer Science,
Indian Institute of Technology Bombay

Task 1

- Coded in Python using the stem library in `make_4-hop.circuit.py`.
- Accessed tor easily (with suitable configurations) with `stem.process`. Used `pycurl` to send a web request (I did for `https://check.torproject.org`).
- Randomly selected 1 entry, 1 exit and 2 middle nodes and built the circuit with the library.
- From the above url, I got a response that we are indeed connecting via Tor but not via Tor Browser, showing success of the connection.
- Used the `nyx` program to confirm the circuit ID with the destination IP address, both of which matched.
 1. Circuit and its ID (which is 9 as shown in Figure 1) printed by my Python script matched with those in `nyx`.
 2. The destination IP (which is `204.137.14.106` as shown in Figure 2) shown in `nyx` for that circuit ID matched with the IP returned by the above url (which claimed it to be *my* IP from its perspective).

```
[*] Connected to Tor ControlPort.
[*] Selecting 4 relays for custom circuit...
[+] Built 4-hop circuit 9
Hop 1: 000D11C0BF30C48FD81644ED2AA3D3733A5149D1
Hop 2: 000A10D43011EA4928A35F610405F92B4433B4DC
Hop 3: 0017A4738B6FFC5E3D0FC7EE380C10FC2160328A
Hop 4: 000F3EB75342BE371F1D8D3FAE90890AEB5664EE
[*] Circuit built successfully.

[*] Waiting for request/stream to go through circuit... (Press Ctrl+C to quit)

[+] Attaching stream 7 to circuit 9
Time taken: 2.2678685188293457
[+] Tor connection recognized.
[+] But not via Tor Browser.
[+] Your IP address appears to be: <strong>204.137.14.106</strong></p>
[*] Press Ctrl+C to quit.
```

Figure 1: Output of the Python Script

```

185.242.225.25:18710 (gb) 93E49E39CB66D0EB4E57F5ADBDFAC0D74DAC3B8 hyberion 2 / Middle
192.42.116.178:9007 (nl) 28338766EA7AD4C9CAE9429C8BC8CA815A53CD4C NTH26R8 3 / End
127.0.0.1 --> 204.137.14.106:443 (us) Purpose: General, Circuit ID: 9 1.0m (CIRCUIT)
85.167.77.39:9001 (no) 000D11C08F30C48FD81644ED2AA3D3733A5149D1 hubbabubbaABC 1 / Guard
104.53.221.159:9001 (us) 000A10D43011EA4928A35F610405F92B4433B4DC seele 2 / Middle
84.16.234.150:9030 (de) 0017A4738B6FFC5E3D0FC7EE380C10FC2160328A MORDEKAISER 3 / Middle
204.137.14.106:443 (us) 000F3EB75342BE371F1D8D3FAE90890AEB5664EE SENDN00SEplz 4 / End
127.0.0.1:38948 (??) --> 127.0.0.1:9051 python3 (67981) + 5.8s (CONTROL)
127.0.0.1:49926 (??) --> 127.0.0.1:9051 nyx (68291) + 5.8s (CONTROL)

```

Figure 2: Output of the Nyx program

Task 2

- Coded in `circuit_selector.py` but used as `python3 make_4-hop_circuit.py --task 2`.
- Made a function `get_path(num_hops)` that returns a valid “num_hops”-hops TOR relay path.
- Fetches the latest consensus and server descriptor files and parses them using `parse_file()`.
- Followed the rules mentioned in section 2.2 of <https://github.com/torproject/torspec/blob/main/path-spec.txt>:
 1. Unique nodes
 2. Picked in the order of first Exit, then Guard and finally middle node(s)
 3. Not two nodes in the same family (found in FAMILY attribute of the server descriptors)
 4. Not two nodes in the same /16 network
 5. Among the viable candidates, the winner was picked randomly with probability proportional to their BANDWIDTH attribute
- As can be seen in Figure 3, the circuit ID and destination IP matches for the Python script and nyx, hence proving the correctness.

```

make_4-hop_circuit.py M X
make_4-hop_circuit.py > connect
42 def connect(task num):
77     if len(relays) < 4:
78         raise KeyboardInterrupt
79     else:
80         # Task 2
81         relays = get_path(4)
82         print()

[*] Connected to Tor ControlPort.
[*] Selecting 4 relays for custom circuit...

--- Tor Path Selected ---
Guard: GroverWashingt17755 (23.92.34.113)
Middle 1: Pouet (146.59.197.114)
Middle 2: prsv (145.249.169.38)
Exit: Quetzalcoat1 (107.189.8.56)

[*] Built 4-hop circuit 9
Hop 1: 01C19C69D2B1EA4897D06DDE5A1896AE673D44A2
Hop 2: 553652FBACFDB28098FDFB020F8F4998FF47D0E9E
Hop 3: 4A860F79A1508F1D9417F8357E77390F61C364FE
Hop 4: B541581CB0599A36269196EFC6E23F67B75285C
[*] Circuit built successfully.

[*] Waiting for request/stream to go through circuit... (Press Ctrl+C to quit)

[*] Attaching stream 7 to circuit 9
Time taken: 2.0498385429382324
[*] Tor connection recognized.
[*] But not via Tor Browser.
[*] Your IP address appears to be: <strong>107.189.8.56</strong></p>
[*] Press Ctrl+C to quit.

```

```

nyx - ubuntu-linux-22-04-desktop Tor 0.4.9.2-alpha-dev (new)
Relaying Disabled, Control Port (open): 9051
cpu: 0.2% tor, 1.3% nyx mem: 47 MB (2.4%) pid: 24042 uptime: 01:49

page 2 / 5 - m: menu, p: pause, h: page help, q: quit
Connections (3 outbound, 12 circuit, 2 control):
40.211.55.5:43550 --> 23.92.34.113:443 (us) + 1.5m (OUTBOUND)
10.211.55.5:38946 --> 45.83.105.223:443 (de) + 1.5m (OUTBOUND)
40.211.55.5:60714 --> 51.159.186.85:9001 (fr) + 1.5m (OUTBOUND)
127.0.0.1 --> 45.138.16.42:9000 (pl) Purpose: General, Circuit ID: 7 1.7m (C
51.159.186.85:9001 (fr) 1 / Guard
51.75.17.236:443 (fr) 2 / Middle
45.138.16.42:9000 (pl) 3 / End
127.0.0.1 --> 81.169.186.16:29002 (de) 1.7m (CIRCUIT)
45.83.105.223:443 (de) 1 / Guard
188.68.33.200:443 (de) 2 / Middle
81.169.186.16:29002 (de) 3 / End
127.0.0.1 --> 107.189.8.56:9000 (lu) Purpose: General, Circuit ID: 9 1.7m (C
23.92.34.113:443 (us) 1 / Guard
146.59.197.114:9001 (fr) 2 / Middle
145.249.169.38:9300 (es) 3 / Middle
107.189.8.56:9000 (lu) 4 / End
127.0.0.1 --> 158.179.267.229:9001 (us) 1.7m (CIRCUIT)
51.159.186.85:9001 (fr) 1 / Guard
81.169.200.33:9001 (de) 2 / Middle
158.179.267.229:9001 (us) 3 / End

```

Figure 3: Outputs of the Python script and Nyx program side-by-side