EELE465 NCUR REPORT:
Creating a Simple Wireless Tor Access Point

Ryan French
ryanfrench3@montana.edu
Graduate Student, Department of Physics
Montana State Physics University - April 8, 2020

Introduction

In today's ever-increasing surveillance of internet activity, people are rightly becoming more vigilant of their internet footprint. The two most popular solutions sold to consumers are:

1. Proxies
    - The least secure method of anonymity. The client's requests are passed through an intermediary server, which masks the client's IP address.
2. Virtual Private Networks (VPN)
    - Very secure. A client's requests are sent to a dedicated server through an encrypted tunnel.

There are issues with both of these methods, however. Most notably is that personal information is being stored on the servers, which is something that cannot be tolerated in some contexts.

For those who require more anonymity, Tor is the go-to service. Tor is an decentralized network which encrypts a client's requests as it bounces between a number (usually 3) of volutary relays.[1] This means that all of the data sent between relays is inaccessible by means of man-in-the-middle attacks or other malicious attempts to intercept data. This makes Tor the solution that is used to access the "Dark Web," the largest part of the internet, which contain sites not indexed by any ordinary web browsers. Tor is not a perfect solution, however, and additional measures must be implemented to confidently lock down the client's identity.

Though Tor is usually thought of as a browser, it is a more fundamental anonymizing software. Because of this, it is possible to build a wireless access point (WAP) that filters the client's requests through the Tor network. Constructing one of these access points only requires a wireless chipset and a microcontroller as its main components. An excellent example of this setup is the Arduino Yún.[2] From here on, all explanations will reference this board, though it's possible to use any microcontroller and wireless chipset.

Implementation

Microcontroller Side: Access Point

Wireless chipsets usually contain a specialized version of the Linux operating system called OpenWrt. It is this OS that will control the wireless connections. However, this chipset is only designed to handle its own connections, which means that it is necessary to obtain another wireless chipset. A popular chipset is the ESP8266, which can easily be obtained in modular form. This connection will be handled explicitly by the microcontroller.

Setting up the access point is relatively easy. The process works as follows:

1. The ESP8266 scans for incoming traffic. Once detected, it passes the information to the MCU.

2. The MCU interprets the data. It is at this point that the MCU can spoof information in the incoming packets if additional anonymity is wanted.
3. The MCU sends this information to the on-board wireless chipset through a bridge.
4. This chipset runs the packets through the Tor process to forward the request.
5. The request is processed and sent back through this chain, and the ESP8266 returns the requested data to the client.

Of course, this explanation is very simplified, as configuring the wireless communication and bridges involves modifying plenty of configuration files. It's also important to test the Tor connection to make sure that the client's information (IP address, MAC address) isn't being leaked. See the "Client Side" subsection for more information on this.

The MCU on the WAP can handle plenty of things in addition to providing a bridge between the wireless chipsets. An excellent example of this is using the MCU to control several LEDs indicating the status of the connections. For example, one set of LEDs can confirm successful connection to the Tor network, another set can confirm that the client is sending requests to secure servers (i.e., servers with HTTPS).

Client Side

The client side has several responsibilities to maintain anonymity. Above, it was mentioned that it is important to confirm that the client's information isn't being leaked somehow. There are three major things that the client must do to protect against this:

1. MAC spoofing. A MAC address is an identifier that is tied to each individual network interface controller. Though Tor usually handles this, it is an additional layer of security to spoof it before handing off information to the access point.
2. DNS setup. A DNS is a system that maps domain names to IP addresses. For example, the DNS will translate the request "www.google.com" to its corresponding IP address before forwarding the request. Without configuring a custom DNS, requests are sent to the client's Internet Service Provider's DNS, which stores information about the client and its requests.
3. IP address spoofing. This is handled by Tor, but it is the client's repsonsibility to confirm that this is happening.

There exist many ways for a client to confirm that all of the above are satisfied. The easiest is to request data from specialized web servers that return as much information about the client as possible. In fact, in this case, the MCU can handle this through its bridge to the on-board chipset, which simplifies the responsibilities of the client.

Conclusion

Though many consumers are decently anonymized through commercialized solutions like proxies and VPNs, these methods can fall short. The Tor network, when used properly, is a much better solution for clients who require more than what these other solutions provide. Because the setup of Tor is relatively easy, it is possible to create a small wireless access point that filters all traffic through Tor. This is a great solution for avoiding most nefarious attempts to gather information about a client.

Note

Information security has been a hobby of mine on-and-off for about 7 months. I have familiarity with many of the topics I've presented here, however, I am by no means an authority. There may be some minor mistakes in interpretation. I must also note that I only advise using Tor for informational purposes, and by no means do I encourage using it with dangerous or illicit intent.

References

[1] "Tor Project." https://www.torproject.org/.

[2] "Getting Started with the Arduino Yún Rev. 2." https://www.arduino.cc/en/Guide/ArduinoYunRev2.