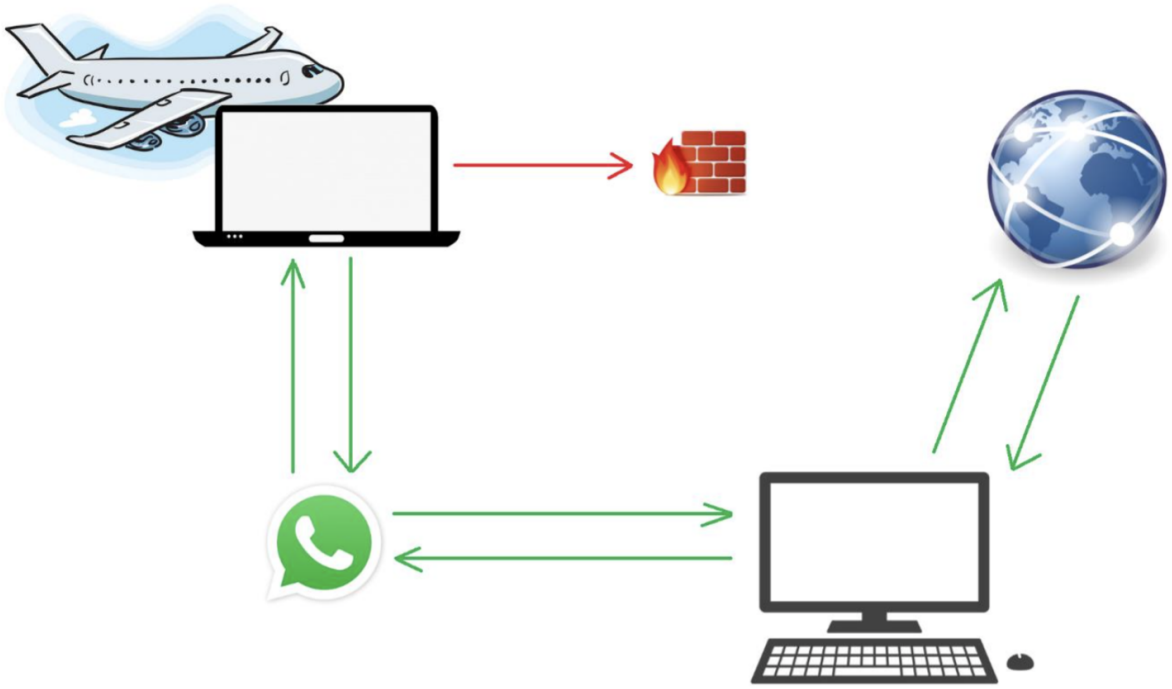
In-Flight WhatsApp Tunneller

Introduction

Most modern airlines offer in-flight wifi... for a price. Some airlines offer free in-flight messaging over WiFi, but restrict browsing traffic on the network. For this project, we hypothesized that we could bypass the in-flight firewall by creating a proxy tunnel through WhatsApp Messenger. By tunneling traffic through WhatsApp to an internet-connected computer on the ground, we are able to visit otherwise restricted websites while in the air. This project simulates a proof-of-concept rather than a live exploit, because we are mere graduate students, not professional hackers. The insight here, perhaps, is that like insects, attackers will find the tiniest openings in your walls and sneak through undetected. It also means that despite forcing the customer to pay for in-flight wifi, it is possible to still browse the internet using a free service and an unconventional method.

Setup



The system works as follows:

Host A requests a restricted website in-flight. The browser is unable to display the page, because the request does not make it past the firewall. We set up a **client-side proxy**, routing all browser traffic to port 44455.

A listener on port 44455 then sends request data as WhatsApp messages to **Host B** on the ground. This process is automated using the PyAutoGUI library and a **WhatsApp client** on Host A.

Host B reads the message and forwards the request through a Squid proxy server, then sends the response data as WhatsApp messages back to **Host A**. **Host A** is then able to render the HTML in a browser to view the page on the plane!

For full technical details, please refer to the presentation and source code.

Conclusion

While this proof of concept was executed with great success, we unfortunately lacked the opportunity to test in-flight. Due to time and resource constraints, we opted out of implementing several essential elements that could make this project a viable service.

We considered implementing multithreading in order to handle concurrency, but the added complication of perfecting polling, queuing, locking, and making it all work at scale made this aspiration far beyond our current scope. The program unfortunately also lacks testing, error handling, and thorough logging.

Overall, this proof of concept demonstrates that a fully functional proxy tunnel through WhatsApp messenger is certainly possible. With more time, perhaps we could all expect a stern letter from the TSA.

Appendix

1. Project repository: https://github.com/NCDiesel/WhatsAppTunneler
2. Virtual Machine Repository: NCDiesel/squid-ubuntu-2004 The repository will be kept public through the end of class(~8-16). Anyone wanting access longer than that please contact Spencer Yost
3. Software dependencies:
    - HashiCorp Vagrant
    - VirtualBox
    - Squid
    - WhatsApp
    - PyAutoGui
    - Python3
4. Reference materials:
    - https://mackgrenfell.medium.com/how-to-get-unlimited-free-wifi-on-delta-flights-e28678b924c9
    - https://dl.acm.org/doi/pdf/10.1145/3433210.3453080
    - https://upgradedpoints.com/travel/airlines/how-airplane-wi-fi-works/