27/04/2025

Student: Adam Fitzpatrick - C00273365  
Course: Software Development

Project Tutor: Joseph Kehoe

MUL-T Final Report

Contents

[Introduction 3](#_Toc196678419)

[Description of Project 4](#_Toc196678420)

[Technical Description 5](#_Toc196678421)

[Hardware 5](#_Toc196678422)

[Raspberry Pi Zero 2 W 5](#_Toc196678423)

[Waveshare 3.2 Inch Touchscreen 5](#_Toc196678424)

[Anavi Infrared 5](#_Toc196678425)

[Software 6](#_Toc196678426)

[Raspberry Pi OS 6](#_Toc196678427)

[Go 6](#_Toc196678428)

[Bash Shell Scripts 6](#_Toc196678429)

[Tview 6](#_Toc196678430)

[Ettercap 6](#_Toc196678431)

[HCITools 6](#_Toc196678432)

[Version Control 7](#_Toc196678433)

[Git 7](#_Toc196678434)

[GitHub 7](#_Toc196678435)

[Feature Fulfilment 7](#_Toc196678436)

[Achieved 7](#_Toc196678437)

[Wi-Fi 7](#_Toc196678438)

[Bluetooth 7](#_Toc196678439)

[Infrared 8](#_Toc196678440)

[Partially Achieved 8](#_Toc196678441)

[Rubber Ducky 8](#_Toc196678442)

[Not Achieved 8](#_Toc196678443)

[NFC / RFID 8](#_Toc196678444)

[Challenges 8](#_Toc196678445)

[Sourcing Hardware 8](#_Toc196678446)

[Device Performance 9](#_Toc196678447)

[Out-dated Libraries & Documentation 9](#_Toc196678448)

[Learning Outcomes 9](#_Toc196678449)

[Future Steps 10](#_Toc196678450)

[Conclusion 10](#_Toc196678451)

[References 11](#_Toc196678452)

# Introduction

This report provides a comprehensive and complete analysis of the MUL-T project, discussing the development process, technical objectives, and outcomes. This document details the project as it currently stands and offers insight into the different challenges that came with its creation.

# Description of Project

In the modern digital era as software grows and becomes more complex, cybersecurity threats are getting smarter, faster and more prominent than ever before. This means that now more than ever, penetration testers and red teamers need tools that are versatile, portable, easy to use, and quick to deploy.

**MUL-T** was built to meet these needs. MUL-T is an open-source, modular penetration testing tool that combines key wireless attack methods into one compact, pocket-sized device. Designed for cybersecurity professionals and ethical hackers, MUL-T streamlines the process of gathering information and testing defences, making it faster and easier to work in ever-changing environments.

MUL-T is designed as an aid for pen-testers and red teamers, allowing users to rely on one streamlined device that’s easy to carry and customize. Being built as an open-source project allows MUL-T to give users to modify workflows, upgrade components, and stay ahead of new security challenges as they see fit.

# Technical Description

## Hardware

### Raspberry Pi Zero 2 W

The Raspberry Pi Zero 2 W serves as the core of MUL-T. This compact microcontroller is powered by a quad-core 64-bit ARM CPU and 512MB of RAM, providing a strong foundation for the device’s performance. It features a 40-pin GPIO (General Purpose Input/Output) header, allowing for connectivity between the various hardware modules used in the project. The board also includes three ports: one mini-HDMI port for video output and two micro-USB ports for power and data connections (Raspberry Pi, 2021).

### Waveshare 3.2 Inch Touchscreen

The 3.2-inch touchscreen IPS display from Waveshare serves as the main display for MUL-T, acting as the interface to interact with the terminal user interface (TUI). The display offers a 480x320 resolution whilst using a 26-pin header, as opposed to the 40-pin header (Waveshare.com, 2025).

### Anavi Infrared

The Anavi Infrared pHAT allows MUL-T to utilise infrared technologies (IR). Using the boards transmitter, MUL-T can capture and save raw infrared signals. While the pHAT includes additional sensors for temperature, humidity, and light, these are reserved for future use cases. The board has a 40 pin GPIO header, allowing it to fit to MUL-T’s modular design (Anavi Infrared pHAT, 2025).

## Software

### Raspberry Pi OS

MUL-T is built upon Raspberry Pi OS (formerly Raspian), the officially supported operating system for the Raspberry Pi. This operating system is based on Debian and was chosen for MUL-T due to its pre-installed libraries, GUI tools, and broader hardware compatibility (Raspberry Pi, 2025).

### Go

Go is a high-level programming language originally designed at Google and known for its simplicity, efficiency and syntactic similarity to C. MUL-T utilizes Go for its backend logic, leveraging the built-in concurrency to manage both the terminal-based interface and tasks seamlessly (go.dev, 2025).

### Bash Shell Scripts

MUL-T utilizes Bash scripting to bridge the gap between Go and Linux tools. Bash scripts simplify invoking these tools without rebuilding their functionality from scratch, while also ensuring that the performance of the device does not suffer.

### Tview

Tview is an open-source TUI framework for Go. Tview comes with a variety of “commonly used components for terminal-based interfaces” (rivo, 2017) and serves as the basis for interaction with MUL-T.

### Ettercap

Ettercap is an open-source network analysis tool used for network sniffing and man-in-the-middle attacks. It has a variety of filters that can be applied when scanning networks for traffic. MUL-T utilises Ettercap to sniff network packets off Wi-Fi networks and to before Address Resolution Protocol (ARP) poisoning.

### HCITools

HCITools facilitates Bluetooth reconnaissance and attacks for MUL-T, allowing the device to discover nearby devices, log MAC addresses and device names and flood other Bluetooth devices with connection requests, forcing disconnections.

## Version Control

### Git

Git is a version control system used to manage file versions and source control. For this project Git was used to track changes to scripts and the source code of MUL-T.

### GitHub

GitHub is a centralized platform for git repositories. The source code for MUL-T is stored on GitHub to allow for easier access and so that the project may be downloaded by anyone.

# Feature Fulfilment

## Achieved

### Wi-Fi

#### Scanning

MUL-T allows pen-testers to scan for networks in the surrounding area using Linux’s built in `iwlist` command. They can filter for networks by SSID, MAC address and network frequency. After a short duration of hitting the scan button, MUL-T displays the surrounding networks with the user’s specified filters.

#### Sniffing

MUL-T can sniff network packets off a network it is connected to using the text-only version of Ettercap. The user can set a duration, after which MUL-T will save a pcap file containing the details of the sniff which can be viewed using a network tool such as Wireshark.

#### Man in the Middle

MUL-T is also able to use Ettercap to perform an ARP poisoning on a network that it is connected to. The user can toggle the attack on or off as they see fit. After the attack is turned off a log is created containing the details of the attack.

### Bluetooth

#### Scanning

MUL-T is capable of scanning for Bluetooth devices using the `hcitool` command. The user can set a duration after which all Bluetooth devices around the device are displayed to the user.

#### De-authentication / Connection Flooding

MUL-T can flood a Bluetooth device using the `l2ping` command. The user first can scan surrounding devices, after which they are able to select the device they wish to flood. Once the user presses the “De-auth” button, the selected device is flooded with connection requests.

### Infrared

#### Scanning

MUL-T can receive raw infrared transmissions using its infrared sensor and the `mode2` command. The user selects a duration to scan infrared for and, after pressing the scan button, is prompted to hold down the button they wish to record for the duration.

## Partially Achieved

### Rubber Ducky

Although not officially supported by MUL-T, the device can act as a rubber ducky due to the device being built on top of the Raspberry Pi and having USB ports that could be connected to another device to run payloads.

## Not Achieved

### NFC / RFID

MUL-T could not implement NFC or RFID due to a lack of native support on the Raspberry Pi and locks on external hardware modules. The [NFC pHAT](https://www.waveshare.com/wiki/PN532_NFC_HAT) used in this project could only read or write to NFC cards where the password was known. This meant that it was not possible to read NFC cards that were protected or to copy the data on them.

# Challenges

## Sourcing Hardware

One of the main challenges while developing this project was sourcing the correct hardware for MUL-T. Due to the modular nature of MUL-T, it was important that parts were easily compatible with one another. Most of the parts for the device came from The Pi Hut, a retailer based in the United Kingdom. This meant that certain parts, due to international shipping laws, such as batteries, were unable to be shipped abroad.

## Device Performance

Device performance was a crucial aspect of this project. Due to the constraints of the hardware, namely the Raspberry Pi Zero 2 W, and its form factor MUL-T had to be able to be optimized to ensure smooth operation and maintain functionality, whilst not affecting the user’s experience.

Resources had to be carefully managed to ensure that they were not being used when unnecessary. This often meant only giving resources a certain amount of time to run, for example giving the user an option to allow a scan

Another side-effect of the device performance was the operating temperature of the device. When running resource intensive tasks, the device tended to heat up significantly. While not an issue for shorter uses of the device, this behaviour could be an issue for the device if it were to be used for an extended period.

## Out-dated Libraries & Documentation

Much of the libraries and documents associated with the hardware of MUL-T were outdated or simply incorrect. This meant that there was a lot of trial and error, and more time involved in trying to get the hardware to function correctly than strictly necessary.

# Learning Outcomes

Through this project I was able to deepen my understanding of terminal operation within Linux and their integration with the operating system. I learned about how commands interact with hardware drivers and various system resources.

I also gained insight into cybersecurity, learning more about pen-testing through hands on experience. I learned how pen-testers simulate real-world threats, from information gathering to exploitation, and got to learn more about the attacks themselves and the intricacies involved with them.

Finally, I learned more about hardware-software integration. I learned about GPIO pins, and the different signals carried by each, as well as how the different hardware modules interact with the Raspberry Pi.

# Future Steps

One future step I would like to take with the project is to implement the transmission of infrared signals from the device. MUL-T is currently able to take infrared transmissions in as raw code, however it is unable to send them due to the transmitter expecting the signals in a format other than raw code.

Another step I would like to take is to add a heatsink or cooling to the device. This would be to help combat the device’s high temperature after extended periods of use. Alternatively, giving the device more RAM or a better CPU would also help combat the issue slightly.

Finally, I would like to create concrete documentation for the device so that the project can be reproduced by others. Currently documentation for the device is rather spread out as each different hardware module has its own documents. I would like to compile these documents into a single, easy-to-read document so that the project can be re-created easily.

# Conclusion

In conclusion, I am very happy with how MUL-T has turned out. MUL-T successfully delivers a portable, modular penetration testing tool that consolidates Wi-Fi, Bluetooth, and infrared attack capabilities into a single compact platform. MUL-T not only addresses the growing demand for agile pen-testing tools but also shows how resourcefulness and iterative problem-solving can transform technical limitations into opportunities for growth.

# References

1. Ltd, R.P. (n.d.). *Raspberry Pi Zero 2 W*. [online] Raspberry Pi. Available at: <https://www.raspberrypi.com/products/raspberry-pi-zero-2-w/> [Accessed 24th April 2025].
2. Waveshare.com. (2025). *3.5inch RPi LCD (B) - Waveshare Wiki*. [online] Available at: [https://www.waveshare.com/wiki/3.5inch\_RPi\_LCD\_(B)](https://www.waveshare.com/wiki/3.5inch_RPi_LCD_(B)%20) [Accessed 24th April 2025].
3. ANAVI Infrared pHAT. (n.d.). [online] Available at: <https://anavi.technology/files/anavi-infrared-phat.pdf> [Accessed 24th April 2025].
4. Raspberry Pi (2025). *Raspberry Pi OS*. [online] Raspberry Pi. Available at: <https://www.raspberrypi.com/software/> [Accessed 24th April 2025].
5. go.dev. (n.d.). *go.dev*. [online] Available at: <https://go.dev/> [Accessed 25th April 2025].
6. rivo (2017). *GitHub - rivo/tview: Terminal UI library with rich, interactive widgets — written in Golang*. [online] GitHub. Available at: <https://github.com/rivo/tview> [Accessed 25th April 2025].