**MonitorPi**

**(A RaspberryPi based Network Monitoring Tool with Cloud Integration)**

**Course: ISCG 8052 (The Internet of Things)**

**Lecturer:** Guillermo Ramirez-Prado

Student Name: Sankha Subodha Palapotha Liyanage

Student ID: 1484052

Student Name: Jin Tae Kim

Student ID: 1485971

Student Name: Ovini Wasanthi Galbokka Hewage

Student ID: 1462372

# Due Date: 23rd June 2017

# Table of Contents

[Table of Contents 1](#_Toc485897650)

[Table of Figures 4](#_Toc485897651)

[Abstract 6](#_Toc485897652)

[Introduction 7](#_Toc485897653)

[Problem Overview 8](#_Toc485897654)

[Project Outline 9](#_Toc485897655)

[3.1 Scope 9](#_Toc485897656)

[3.2 Assumptions 9](#_Toc485897657)

[3.3 Constraints 10](#_Toc485897658)

[3.4 Solution Concept 10](#_Toc485897659)

[Similar tools comparison 11](#_Toc485897660)

[4.1 Wireshark 11](#_Toc485897661)

[4.2 PiTap 12](#_Toc485897662)

[4.3 NetPi 12](#_Toc485897663)

[4.4 PwnPi 14](#_Toc485897664)

[Requirement Analysis 15](#_Toc485897665)

[5.1 Hardware Requirements 15](#_Toc485897666)

[5.1.1 Device Selection 15](#_Toc485897667)

[5.1.2 Other Required components 16](#_Toc485897668)

[5.2 Software Requirements 17](#_Toc485897669)

[Components 17](#_Toc485897670)

[6.1 Monitoring Tool to detect Internal Attacks 17](#_Toc485897671)

[6.1.1 Functionality 17](#_Toc485897672)

[6.1.2 Software Architecture Diagram 18](#_Toc485897673)

[6.2 Network Activity Monitoring 18](#_Toc485897674)

[6.2.1 Functionality 18](#_Toc485897675)

[6.3 Upload & Display the sensor data on a MQTT Broker 19](#_Toc485897676)

[6.3.1 Functionality 19](#_Toc485897677)

[Critical Evaluation 22](#_Toc485897678)

[Conclusion 23](#_Toc485897679)

[References 24](#_Toc485897680)

[Appendix 25](#_Toc485897681)

[Appendix A – Tool to detect internal attacks 25](#_Toc485897682)

[Appendix B - Testing 26](#_Toc485897683)

[6.1 Test Environment Setup 26](#_Toc485897684)

[6.2 Results Published 27](#_Toc485897685)

# Table of Figures

[Figure 1: Wireshark Interface 11](#_Toc485855665)

[Figure 2: PiTap Device 12](#_Toc485855666)

[Figure 3: NetPi Interface 13](#_Toc485855667)

[Figure 4: PwnPi Application 14](#_Toc485855668)

[Figure 5: Attack Detection Flow Diagram 18](file:///C:\Users\Sankha\Desktop\IoT%20Assignment2\IoT%20Group%20Assignment.docx#_Toc485855669)

[Figure 6: Variable Assignment to MQTT Broker Web Dashboard 21](#_Toc485855670)

[Figure 7: Broadcast Packets Display 27](#_Toc485855671)

[Figure 8: DHCP Packets Display 27](#_Toc485855672)

[Figure 9: DNS Packets Display 28](#_Toc485855673)

[Figure 10: ARP Packets Display 28](#_Toc485855674)

[Figure 11: ICMP Packets Display 29](#_Toc485855675)

[Figure 12: IGMP Packets Display 29](#_Toc485855676)

[Figure 13: SYN Packets Display 30](#_Toc485855677)

[Figure 14: HTTP Packets Display 30](#_Toc485855678)

[Figure 15: MAC Address Table Log 31](#_Toc485855679)

[Figure 16: Protocol packet Counter Log 31](#_Toc485855680)

# Table of Tables

[Table 1: Raspberry Pi Device Comparison 15](#_Toc485897712)

[Table 2: Device Setup Costing 16](#_Toc485897713)

# Abstract

With the increased use of internet, the necessity and importance for monitoring tools has become an essential part of the networking domain. As the Local Area Networks allow escalated privileges, identifying an attack has become increasingly challenging. Malicious users can overload networks and significantly impact on the network performance. The authors have proposed to implement a cost-effective, portable system to real-time monitor a network with ability to plug-and- play.

A Raspberry Pi is used for this purpose due to its compact size and convenience. The prototype reads the incoming packets on the Ethernet interface and categorize them based on its type and logs the source MAC address. The read packet values will be uploaded to a MQTT Broker and displayed on graphs; allowing the administrators to make rapid decisions during malicious activity; thus, maintaining the Quality of Service of the network.

Keywords: **Raspberry Pi, MQTT Broker, Python, Packet Capture, Packet Analysis, Ethernet, Network Flood Attacks, LAN Monitoring**

# Introduction

Security is considered a critical component in Computing and Networking; where the packets transferred rely on the communication, data storage, exchange and security of the Network. Weaknesses and vulnerabilities can damage the performance and functionality of a network, reflecting the negative effects on the users. Identifying such weaknesses and implementing preventive measures is essential for the continuity and longevity of the network. Considering wired networks, most attacks are hard-to detect due to defined user-rights within the network. By implementing a tool to detect abnormalities and malicious activity within the network, it would be easier for the network administrators to troubleshoot different network packet flood-attacks and implement preventive measures. Other than network monitoring, SYN and ICMP packet types can be utilized to identify faulty devices; thus, bringing down the problem identification time and help solve the issues efficiently for the optimum network continuity.

# Problem Overview

Considering the network analyzing tools, there are limited options available to monitor internal network flood-attacks. (Legit host devices generating traffic to harm the LAN performance). During an event as such, the network management team must review individual device and disconnect them from the network to identify the faulty host. Following method of network inspection is not a viable solution in practical environment; as the efficiency of the company is directly affected by network performance; thus, causing financial and resource loss.

During an attack, the administrator prepares to detect the attack origin; which may be hindered by multiple attack origins, user activity and automated-attack reproduction. The above process takes significant time and leads in further vulnerability spread, device infection and productivity loss. To minimize the above damages, the diagnostic time has to be significantly reduced; which in return saves time and productivity. Considering the security of LAN Networks, most do not feature a high-end monitoring system (except a firewall). Internal attacks are not considered as the individual usage differs from user-to-user and the errors are troubleshooted on an action-triggered policy. Mostly, the attacks are compensated with port mirroring, which consumes a significant amount of time for setup. With the availability of portable devices, an ideal monitoring tool can be implemented using a low-powered, small-form easy-to setup device.

# Project Outline

## 3.1 Scope

The tool is specifically developed to identify packets, ICMP and SYN flood-attacks within the Local Area Network. The proposed tool will run on a Raspberry Pi and capture data frames, analyze the packet types and detect any abnormalities through a wired Ethernet interface running on (IEEE 802.3 standard). Apart, the tool has added functionality to check individual device status (by sending ping requests) and record delay and jitter of each device. All the recorded data will be uploaded to a MQTT Broker, where the network administrators are given the privilege to view the network activity over cloud. The proposed tool will be implemented for LAN usage and the created features will be focused on LAN specifications.

## 3.2 Assumptions

* The following assumptions were made by the author to keep the project within the defined scope.
* The implemented application functionality is defined strictly for IPv4 Local Area Networks
* The device can read/write to/from the network and is connected to a port with LAN port/packet-mirroring ability; where the network-end and the proposed system-end operates in promiscuous mode. (receive all the packets regardless of the sender and recipient).
* The output results are displayed in a GUI and on a MQTT Web-dashboard.

## 3.3 Constraints

Considering the limited testing resources available, constraints were imposed on the system which can be elevated for future development and added resources.

The application is designed to run on Debian Linux based platform with Python and PyQt support. The API supports multiple platforms including Android/Unix and Windows; which can be run using LibPcap library.

The application doesn’t support Python backward compatibility and requires the exact same resources and libraries for the functionality of the application.

The application is limited to IPv4 packet analysis; no IPv6 support is provided for the application at the defined level.

## 3.4 Solution Concept

Considering the Problem Overview and Similar tools available in the market, the primary concern was to implement a LAN monitoring tool with added portability. After considering the main contenders, the tool was planned to implement on Raspberry Pi as the device has to be prioritized only for packet analysis process. Linux provide open-source Operating System and all the required libraries for free. The above decision was made over Android mobile devices as they lack an Ethernet port; which matter when using a serial Ethernet bridge to strip the incoming packet for IP address and MAC address rendering process. The tool sniffs packet headers via a socket. Python can be used for packet capturing services as ‘Pycap’ libraries and scripting provides the full-functionality to implement the required services.

Apart from packet analysis, the proposed tool will have added functionality to identify active devices by sending periodic ping requests to the devices in the network. By calculating the Delay and Jitter, the network administrators would be able to further optimize the system for efficient functionality. All the data will be uploaded to a MQTT Broker and displayed on a web-dashboard; thus, enabling the administrators to view the network activity without remote login to the system. Implementing the proposed system would enable the administrator to remotely monitor the network activity and easily compensate during a network attack with minimal damage to the company.

# Similar tools comparison

## 4.1 Wireshark

Wireshark is considered the most famous tool for packet capturing over the years with its open-source license and multi-platform support. Wireshark has the advantages of live packet capture, offline analysis, flexibility in analysis while providing predefined packet filters to be applied on the captured session. Apart, Wireshark can review, generate and export statistical data with clarity and allows the user to create new filters according to their requirement. As drawbacks, Wireshark doesn’t provide real-time packet analysis and attack detection; but provides rich functionality for network packet dump analysis. (Wireshark.org, 2017)

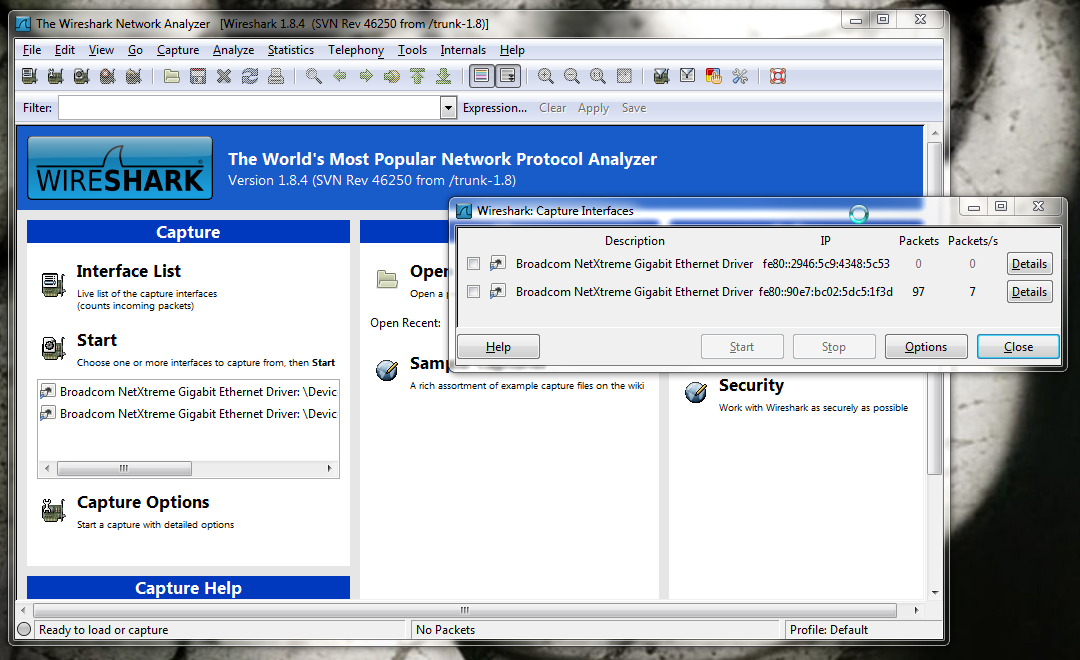


Figure 1: Wireshark Interface

## 4.2 PiTap

PiTap is a Debian based hardware packet dumping tool developed for network packet capturing by acting as a network bridge. The device uses two or more Ethernet interfaces (using USB to Ethernet adapters) and the captured packets are dumped to the SD card attached. The packet capture happens real-time, but the system lacks real-time packet analysis. The user must manually perform the analysis. The tool is ideal for network audits, but fails at real-time packet analysis. (Knowles, 2014)



Figure 2: PiTap Device

## 4.3 NetPi

NetPi is a penetration testing system based on Raspberry Pi. The testing tools and software are built into an SD card image. The portability is obtained by combining a touchscreen to the system. The features provided by the tool include

* CDP/LLDP – Neighbor Details Discovery through the Ethernet port of the NetPi
* DIA Diagram Software to build network topologies
* Reports on Ping/ Trace/ Speed Test are saved during every scan for post-analysis
* Built in Penetration Testing Toolkit (Wireshark/Wireless Scanner/Zen map)
* SYSLOG Server – Send Syslog messages from network devices for post analysis.

Considering the proposed system’s functionality, NetPi is almost similar. Unfortunately, the project is still available at beta level and the development has been paused. The touchscreen features give errors with the released beta version and is considered unstable. NetPi provides a platform for other penetration testing software to run on and includes features such as one-click launch, automatic launch panel and save to dump file. (Errors, 2017)

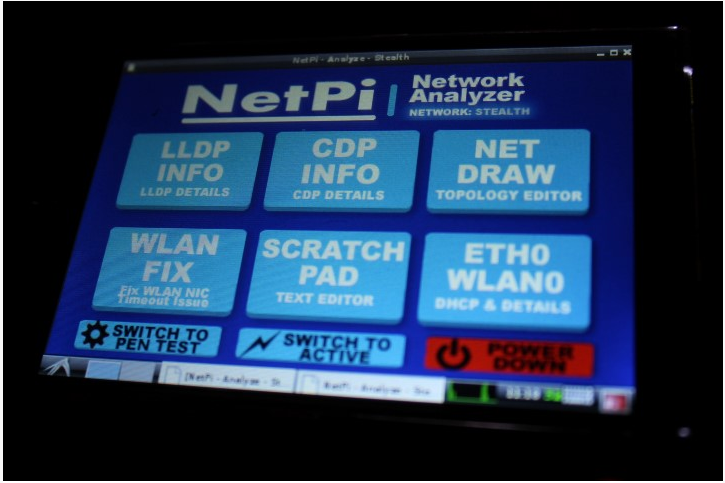


Figure 3: NetPi Interface

## 4.4 PwnPi

PwnPi is a stripped down Linux Debian Wheezy based penetration testing dropbox distribution which provide over 200 pre-installed network security tools in aid of penetration testing. Openbox is used as the window manager and has the functionality to send reverse connections from inside of a target network by editing a simple configuration file. The project’s last development was carried out in 2012 and the project has been abandoned without further development. The options for penetration testing has been provided, but the packet capturing and analysis has to be done manually; the application provides no direct way to analyze a SYN/ICMP flood attack. (Pwnpi.sourceforge.net, 2017)

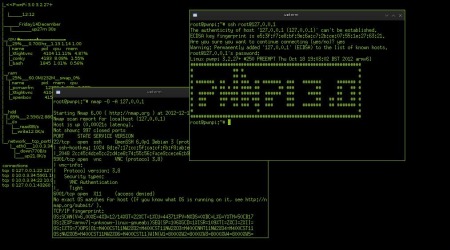


Figure 4: PwnPi Application

# Requirement Analysis

## 5.1 Hardware Requirements

Considering the scope of the project, the hardware required should be highly portable with significant processing power. The 4 factors ‘Hardware Device Type’, ‘Supported Operating Systems’, ‘supported programming languages’ and ‘access to packet capture’ were considered when identifying the best solution.

### 5.1.1 Device Selection

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Model | Model B | Model B+ | 2, Model B | Raspberry Pi3 |
| Price | $35 - $40 | $25 - $30 | $35 - $40 | $ 69 |
| Summary | Original Raspberry Pi Model | Original Model with multiple USB ports | Raspberry Pi2 Model with high processing power | Latest Raspberry Pi Model |
| Chip | Broadcom BCM2835  ARMv6 single core | | Broadcom BCM2836  ARMv7 quad-core | Broadcom BCM2837 ARM Cortex A53 quad-core | |
| Processor Speed | 700MHz | | 900MHz | 1.2GHz | |
| Power consumption | 600mA, 5V | | 650mA, 5V | 700mA, 5V |
| Memory | 512MB SDRAM, 400MHz | | 1GB SDRAM, 400MHz | 1GB LPDDR2 RAM | |
| USB interfaces | 2 | 4 | | | |
| Ethernet | 10/100MB Ethernet RJ45 Port | | | | |

Table 1: Raspberry Pi Device Comparison

References: <https://www.pbtech.co.nz/product/SEVRBP0045/Raspberry-Pi-3-Model-B-with-12GHz-64-bit-Quad-Core>

<https://www.makershed.com/pages/raspberry-pi-comparison-chart>

After analyzing the available solutions, Raspberry Pi3 is selected as it provides a significant processing power at an affordable cost. Apart, the device consumes slightly higher current (700mA vs 650mA); but provides a quad-core processor running at 1.2GHz compared to its predecessor with a single-core 900MHz processor. The device has 4 USB 2.0 interfaces which can be used to plug-in multiple USB to Ethernet interface adapters. Raspberry Pi3 provides significant high processing power, but comes with a hefty price tag of around $70 – $80; but packs high power (1.2GHz Broadcom BCM2837 chipset) with a built-in wi-fi and Bluetooth module; compared to every previous Raspberry Pi model. The authors have decided to implement the product on Raspberry Pi2 Model B as it is identified as the most commercially viable solution at the time of implementation. As the technology advances with time, the device can be upgraded according to requirement.

### 5.1.2 Other Required components

|  |  |  |
| --- | --- | --- |
| Component | Usability | Price |
| Power Adapter | Power the Raspberry Pi device | $ 20 |
| Heat Sink | Minimize the heat produced during operation | $ 5.75 |
| Case with cooling fan | Ventilate the device minimizing the heat generated | $ 21.85 |
| Ethernet cable CAT6 3m | Connect the Raspberry Pi with the network | $ 5.85 |
| Ethernet to USB adapter | Add more Ethernet interfaces to the device to enable packet capturing | $ 25.30 |
| 16GB Micro SD card | Provide storage space and installation for Operating System | $ 16.10 |
|  | **Total** | **$ 94.85** |

Table 2: Device Setup Costing

References: [https://pishop.nz/](https://pishop.nz/RPI-16GB-NOOBS/)

https://www.pbtech.co.nz/brand/Raspberry%20Pi

The above identified equipment are required to setup the device for its basic function; the rates are obtained from New Zealand authorized dealers ‘pbtech’ and ‘pishop’. As the device involves in packet capturing and analysis, higher level of heat is to be expected. The device is setup with heat sinks and a cooling fan to keep the device working at optimum temperature.

## 5.2 Software Requirements

The default Raspbian Jessie was replaced with Raspbian Jessie Lite operating system as the implemented system is highly resource-intensive. Compared to Raspbian Jessie, Raspbian Jessie Lite lacks X-Server and its respective components. A very light Graphical User Interface is used to ease user involvement with the device. The program is designed to implement in Python as it is a very-powerful, versatile programming language running on devices with minimal resources.

# Components

## 6.1 Monitoring Tool to detect Internal Attacks

### 6.1.1 Functionality

The application involves in identifying network packets at OSI Layer 2 and 3. The process involves in socket implementation and packet header disassembly for packet-type identification. The counted packets are displayed on a Graphical User Interface and a counter involves in identifying sudden surges of a specific packet-type; helping to identify unusual network activity in return. The read packets are categorized by the source MAC address over IP addresses as IP addresses can be spoofed easily compared to MAC address. However, the application only checks for the source MAC address. To gain the full functionality, the source and destination MAC addresses tracing can be added to the application; which would enable the network administrators to easily identify the rogue device.

### 6.1.2 Software Architecture Diagram

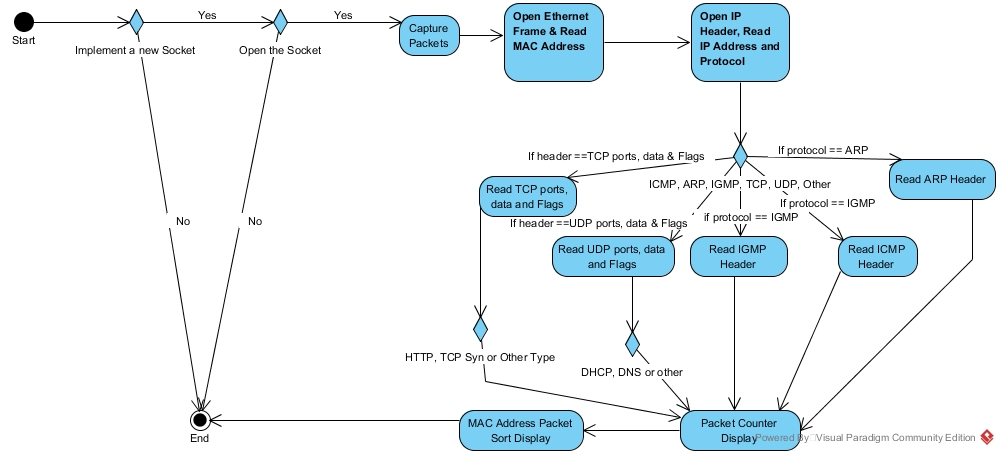


Figure 5: Attack Detection Flow Diagram

## 6.2 Network Activity Monitoring

### 6.2.1 Functionality

The second component of the application involves in applying the monitoring tool to a fully operational industrial scale network; where the network activity is monitored using the Raspberry Pi. The proposed system will monitor the ping, delay, jitter and load of each networking device. The functionality and application of each component is explained below.

Ping checks the reachability status of a network component by sending ICMP request -reply packets. By monitoring Ping requests, the network administrators gain the ability to identify inactive/faulty devices and would make it easier to troubleshoot.

Delay defines the time taken for 1-bit to travel from the source to the destination. The delay is measured in milliseconds and significantly affects QoS on VoIP networks. By monitoring the delay, network engineers can analyze the devices with highest traffic (higher delay compared to average network delay) and opens the opportunity to increase the efficiency of the network by implementing new routes and algorithms.

Jitter is the variation in delivery of packet-order during a network packet transmission. The packets take multiple paths (where paths vary from packet to packet depending on the best path available) due to queuing, configuration errors and network congestion. The packets are sent from the source port with defined time intervals; but the packets may receive at the destination with variable time gaps due to multiple path selection. Jitter significantly affects video and data streaming networks; the analysis can increase the efficiency and overall productivity of the network.

## 6.3 Upload & Display the sensor data on a MQTT Broker

### 6.3.1 Functionality

The packet capturing tool is provided with the functionality to upload the sensor data to a MQTT Broker (ThingSpeak in this scenario). The packet values are uploaded to the cloud service and displayed under multiple graphs (one graph to display one packet type). The different packet types are assigned to variables by the following commands.

# TaklBack Definition

# ThingsSpeak -> Apps -> TalkBack

TalkBackID = '15763'

TalkBackAPIKey = 'X0TF1DGNJFOY2G6W'

WKEY = '57U9CEIXT2WF00PC'

headers = {"Content-type": "application/x-www-form-urlencoded", "Accept": "text/plain"}

The values are passed to the Cloud service using the commands

#Upload packet counter to ThingsSpeak

params = urllib.parse.urlencode({"field1": broadcast\_packet\_total,"field2": dhcp\_packet\_total, "field3":dns\_packet\_total, "field4":arp\_packet\_total, "field5":icmp\_packet\_total, "field6":igmp\_packet\_total,"field7":syn\_packet\_total, "field8":http\_packet\_total, "key": WKEY})

conn = http.client.HTTPConnection("api.thingspeak.com:80")

print( "=====Debug=======" )

try:

conn.request("POST", "/update", params, headers)

response = conn.getresponse()

print ("Status :", response.status,"Reason:", response.reason)

data = response.read()

conn.close()

The Cloud display graphs are updated frequently as the ‘upload data’ function is called within the ‘packet capturing’ loop. This enables the upload tool and packet capturing tool run simultaneously; increasing the accuracy of the published data.

By analyzing the packet counter values, a network administrator can view the activity status of the network and helps to identify any abnormal network behavior. However, the management team has to define the threshold values for each packet type (the average number of packets transferred within the network in a given time) during peak hours; as this will trigger an alert if incorrectly configured.

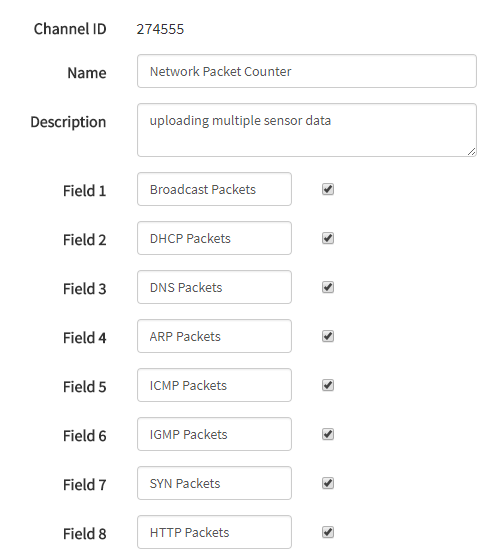
. 

Figure 6: Variable Assignment to MQTT Broker Web Dashboard

# Critical Evaluation

The implemented application was tested for its functionality and multiple test procedures were utilized. ‘Cat Karat builder’ application was used to generate packets for application functionality testing. The test environment had certain limitations on the community-edition; only one type of packet generation allowed for a given instance and the application lacks functionality to generate HTTP traffic. Therefore each component was tested separately by generating their respective packet type.

The Ethernet interface of the Raspberry Pi was changed to promiscuous mode. This enables the device to capture all the incoming packets without restriction. To enable promiscuous mode, the adapter parameters had to be changed; which disables the interface usage for other network-related purposes. Wireless adapter was configured for data publishing to the MQTT Broker. As Ethernet is preferred over wireless by default, the Ethernet connection is prioritized; disabling the wireless interface. Therefore, Wireless interface had to be manually prioritized over Ethernet connection.

Raspbian Lite version is used over the default Raspbian Operating System as the application is resource intensive. LXAppearance theme is used to simplify the Graphical User Interface. As the key-point of the implementation is portability, the device power consumption plays a significant role. By using less-resource intensive Operating System, this challenge can be avoided. As the device doesn’t involve in displaying the data on a display while capturing packets, the Graphical User Interface is considered a negligible factor. But, a Graphical User Interface is used at the prototype for demonstration purposes.

Considering the concept to check network-device activity, a separate Ethernet interface has to be used as the existing interface is dedicated for packet capturing. This challenge can be easily overcome by using plug-in USB to Ethernet interfaces. The application efficiency and functionality can be increased by using the above method; where the network traffic can be monitored by multiple mirror-ports.

# Conclusion

After identifying the gap in the network monitoring field, the authors have come up with a prototype monitoring tool. The partial-implantation was the result of the research carried out on the related field of study. Considering network security, the lack of real-time portable monitoring tool is identified as a main weakness of the networking domain. The packet analysis must be done manually and is considered a major drawback to the industry.

The authors have provided a prototype to minimize the above weakness and provide a reliable, cost effective, easy to setup device with ability to read the network activity and notify the administrators when necessary. Packet analysis aid in identifying malicious users and as the MAC address get recorded in the log file, the necessary precautions can be easily implemented. Apart, the same device is proposed to check device activity within the network by checking the delay, jitter and availability via ping requests.

Domain research has provided the authors with beneficiary information related to the topic and they had a significant impact on the implementation process. As a result, a fully functional application covering weaknesses are proposed and the major component has been implemented. The implemented application is capable of handling multiple packet types and uploading the readings to the MQTT Broker; providing accurate details of the ongoing network activity.

The project will be continuously developed into a fully functional application as a project of the authors and will be available to the users with open-source license.

# References

Errors, S. (2017). *NetPi - Raspberry Pi Network Analyzer - Blame The Network*. Retrieved June 05, 2017, from Blame The Network: http://blamethenetwork.com/netpi-raspberry-pi-network-analyzer/

Knowles, W. (2014, May). *PiTap: Automated packet capture on a Raspberry Pi #piday #raspberrypi @Raspberry\_Pi*. Retrieved June 01, 2017, from Adafruit Industries - Makers, hackers, artists, designers and engineers!: https://blog.adafruit.com/2014/05/16/pitap-automated-packet-capture-on-a-raspberry-pi-piday-raspberrypi-raspberry\_pi/

Pwnpi.sourceforge.net. (2017). *PWNPI.NET | The Pen Test Drop Box Distro for the Raspberry Pi*. Retrieved June 03, 2017, from Pwnpi.sourceforge.net: http://pwnpi.sourceforge.net/index.html#

Wireshark.org. (2017). *Wireshark · Go Deep.* Retrieved June 03, 2017, from Wireshark.org: https://www.wireshark.org/

# Appendix

## Appendix A – Device Setup

Implementation

* Download Raspbian Lite Operating System.

https://downloads.raspberrypi.org/raspbian\_lite\_latest

* Guide to install the image on to the SD card.

https://www.raspberrypi.org/documentation/installation/installing-images/README.md

1. Run “sudo apt-get update” and “sudo apt-get upgrade” commands to update the current application list and system files.
2. Run “sudo apt-get dist-upgrade” to upgrade the distribution and run “sudo apt-get clean” command to clean the installation files.
3. Run “sudo apt-get install  xserver-xorg” to install the GUI host server and run “sudo apt-get install  xinit” to initialize the GUI at login.
4. Run “sudo apt-get install python3 python3-pip python3-pyqt5” to install the runtime environment.
5. Identify the Ethernet network interface with the command “netstat -I”
6. Edit the ‘/etc/network/interfaces’ file with the following configuration to enable Promiscuous mode on the Ethernet interface.

auto lo   
iface lo inet loopback   
auto eth0   
up ip link set $IFACE promisc on   
down ip link set $IFACE promisc off   
down ifconfig $IFACE down

## Appendix B - Testing

The following results were observed on ThingSpeak MQTT broker when the tests were carried out on the implementation.

### 6.1 Test Environment Setup

To setup the test environment, the following devices are required.

* A router with two or more Ethernet interfaces
* Two Ethernet cables
* Raspberry Pi
* Power adapter
* Wireless adapter (optional)
* Display to view network activity (MQTT Broker)
* PC running Packet generator (Cat Karat Builder in the considered scenario)

1. Switch on the router and plug-in the two Ethernet cables to the LAN ports. (Make sure not to plug-in the cable to the Broadband interface).
2. Power-up the Raspberry Pi and allow it to boot to desktop. Use the wireless adapter and connect to an existing wireless network (the wireless adapter is optional and depends on the Raspberry Pi module used.
3. Check the Raspberry Pi Ethernet port is running on Promiscuous mode (when promiscuous mode is enabled, the adapter will capture all the incoming packets regardless of its type.)
4. Plug one Ethernet cable from the router to the Ethernet interface on Raspberry Pi. Once plugged in, the green LED will light up, confirming the port is enabled.
5. Run Cat Karat builder with administrator privileges.
6. Plug the existing Ethernet cable to the PC with packet generator.
7. Select the Ethernet interface to send out packets and start packet generation.
8. Run the packet capturing code on Raspberry Pi and wait for incoming packet identification. If no packets are detected, disable the firewall of the packet builder PC and relaunch the packet generator.
9. The captured packets will be displayed on the CLI; the captured details will be uploaded to the MQTT Broker simultaneously.

### 6.2 Results Published

1. Broadcast Packet Monitor

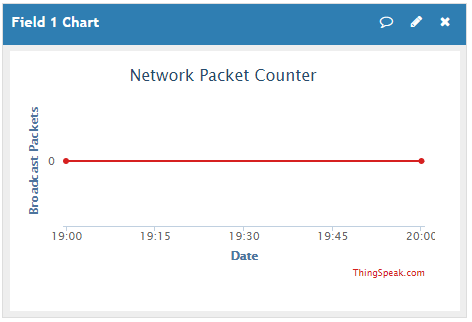


Figure 7: Broadcast Packets Display

1. DHCP Packet Monitor

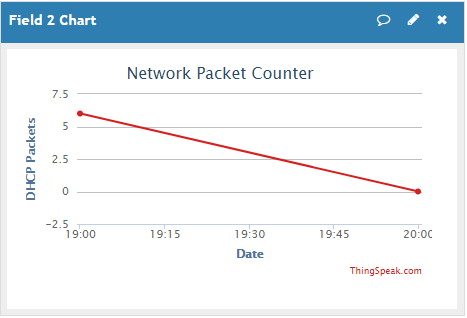


Figure 8: DHCP Packets Display

1. DNS Packet Monitor

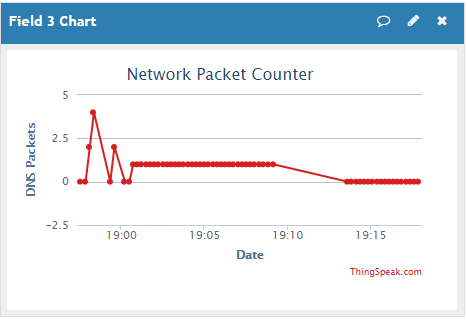


Figure 9: DNS Packets Display

1. ARP Packet Monitor

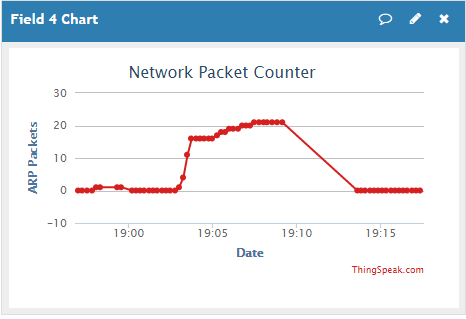


Figure 10: ARP Packets Display

1. ICMP Packet Monitor

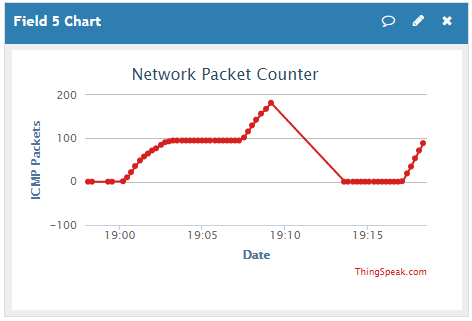


Figure 11: ICMP Packets Display

1. IGMP Packet Monitor

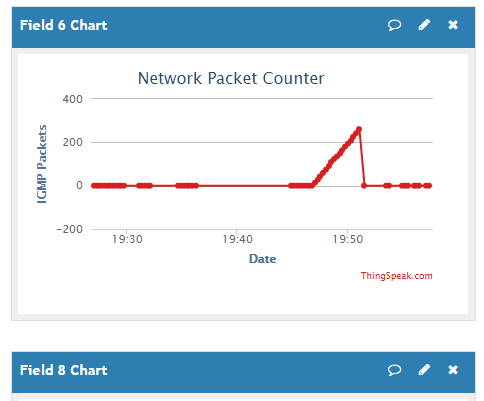


Figure 12: IGMP Packets Display

1. SYN Packet Monitor

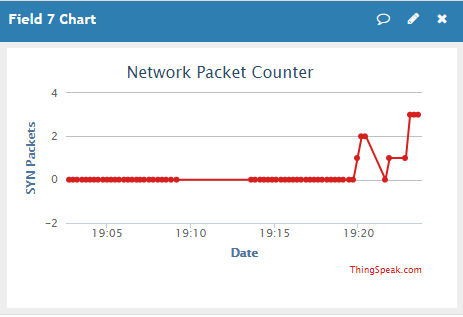


Figure 13: SYN Packets Display

1. HTTP Packet Monitor

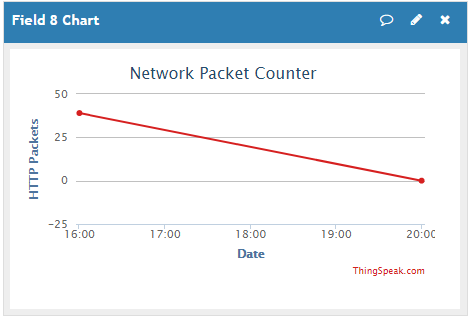


Figure 14: HTTP Packets Display

6.2 Saved Log files

MAC Address Table

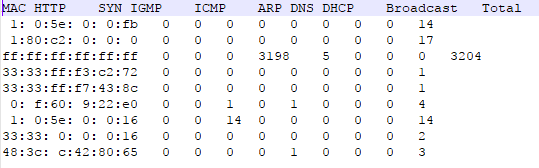


Figure 15: MAC Address Table Log

Protocol packet Counter

A screenshot of a cell phone

Description generated with very high confidence

Figure 16: Protocol packet Counter Log