RASPBERRY HOUSE: AN INTRUSION DETECTION AND PREVENTION SYSTEM FOR INTERNET OF THINGS (IOT)

MCS Thesis Defence

Wen Fei

March 29, 2022

DALHOUSIE UNIVERSITY

MYTECHLAB
Emerging Wireless Technologies
Research Lab

DALHOUSIE UNIVERSITY

OUTLINE

1. Introduction
2. Literature Review
3. Proposed Raspberry House Design
4. Experimental Implementation
5. Experimental Results & Evaluation
6. Conclusion & Future Work

01

Introduction

Introduction to IoT

DoS Attacks on IoT

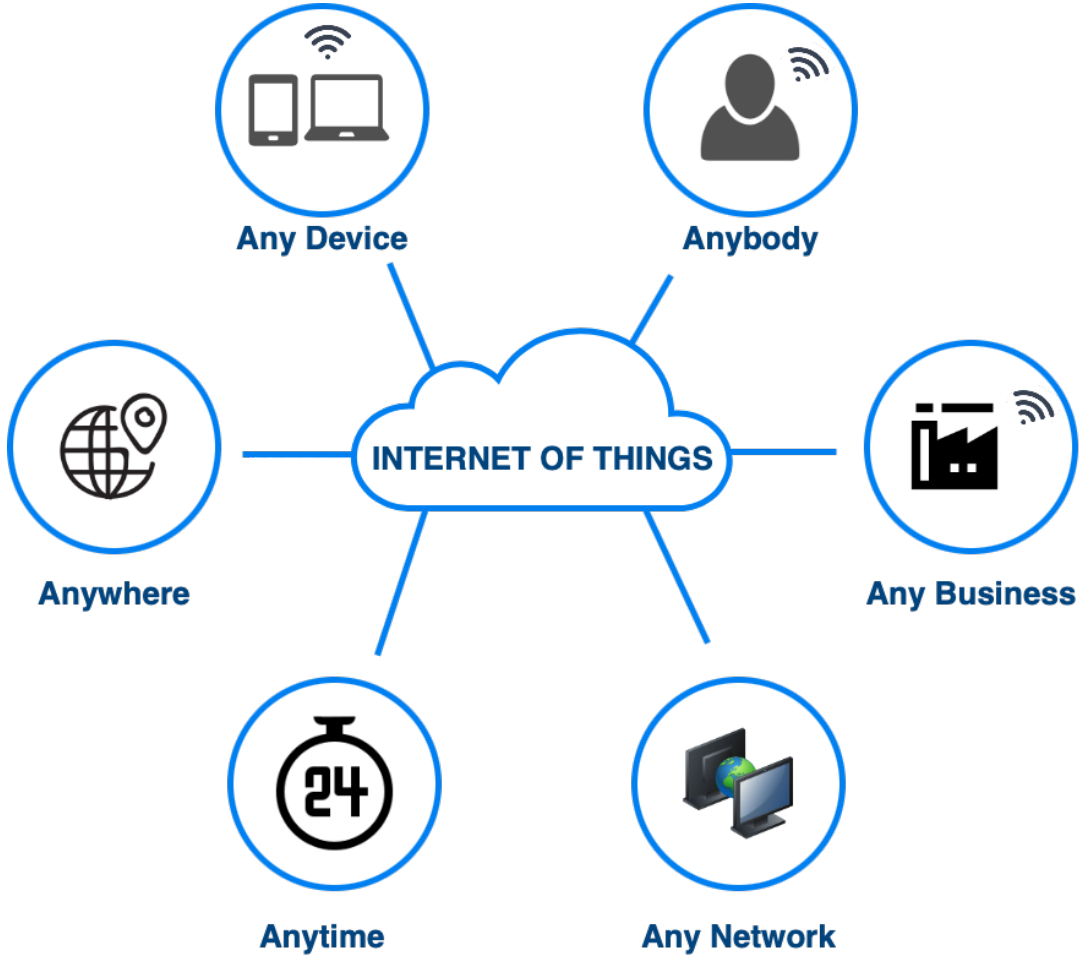
Research Objectives

Introduction to IoT

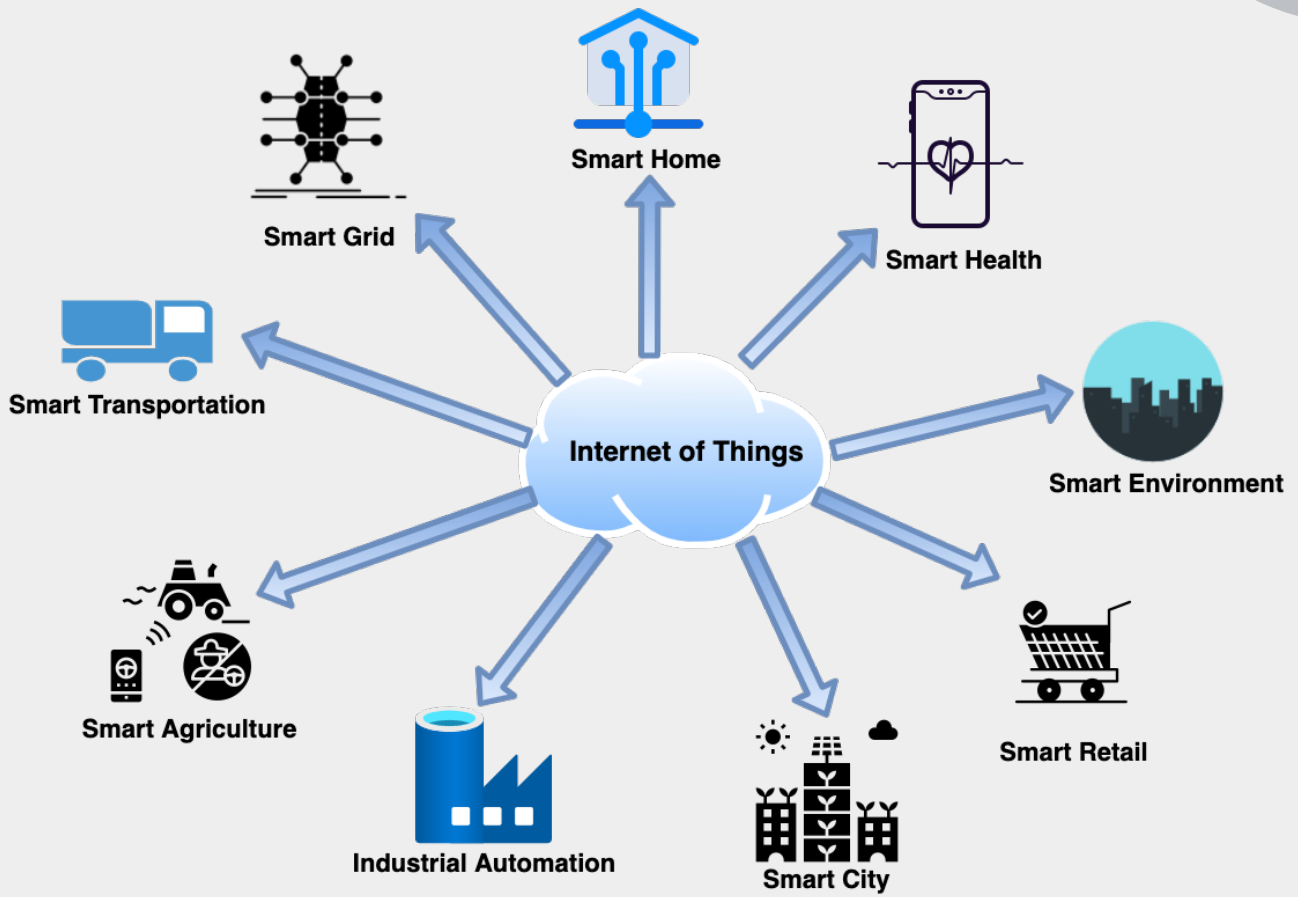
IoT is a network that connects various physical objects to the Internet.

IoT devices can:

- Share sensor data
- Communicate with other IoT devices
- Automate different tasks



Introduction to IoT (cont'd.)

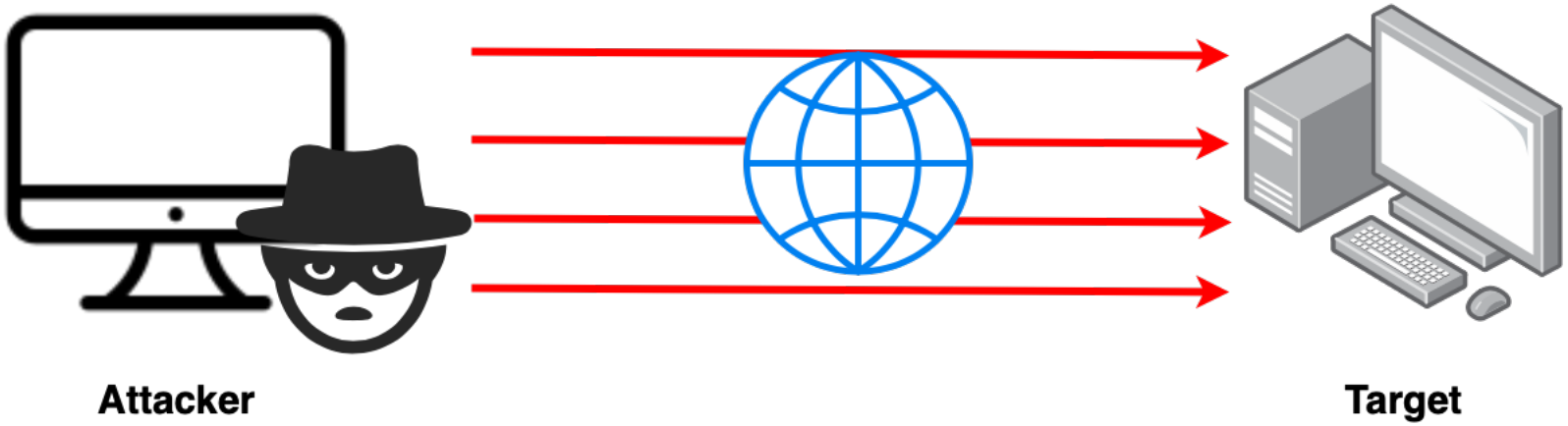


IoT Applications

Denial of Service (DoS) Attacks on IoT

- IoT devices have **low power**, and **limited computing and storage capabilities**, thus making them vulnerable to several attacks.

- **Denial of Service** (DoS) is one of the most common IoT attacks.



DoS Attack Process

Research Objectives

1. The main objective is to design, implement, and test a gateway called **Raspberry House**, a security gateway for detection and prevention of intrusions on IoT devices.
2. The proposed Raspberry House targets one of the major attacks on IoT devices, namely, **Denial of Service (DoS)**, at the data link, network, transport layers and the system security level.
3. The proposed Raspberry House can detect, alert, and prevent DoS attacks **in real time**, particularly applicable to small IoT devices with resource constraints.
    - **Deauthentication attack, SYN flood attack, ICMP flood attack, and the bash fork bomb attack.**
4. The proposed Raspberry House can **accurate detect and prevent DoS attacks**.

02

Literature Review

Comparative Study of Current Security Schemes for DoS Attacks

IoT Gateway Design

Research Gaps

Contributions

Comparative Study of Current Security Schemes for DoS Attacks

Table below shows the contribution of this paper compared with other research papers in this area.

	Raspberry House	Ullah et al. 2021	Simadiputra & Surantha 2021	Aung & Thant 2017	Zhang & Sampalli 2010
IoT Architecture	●	●	●	●	●
Real Time IDS	●	●	●	●	●
IDS against data link, network, transport layers and the system security level	●	Transport layer only	Network layer Transport layer	Data link layer	●
Real Time IPS	●	●	●	●	●
IPS against data link, network, transport layers and the system security level	●	Transport layer only	●	●	Data link layer
Alert user approach	●	●	●	●	●

● : Covered ● : Not Covered

Comparison of Raspberry House with Other Related Research

Comparative Study of Current Security Schemes for DoS Attacks (cont'd.)

	Nguyen et al. 2022	Javanmardi et al. 2021	Binu et al. 2021	Mariam & Negash 2021
IoT Architecture	●	●	●	●
Real Time IDS	◐	◐	◐	●
IDS against data link, network, transport layers and the system security level	●	Transport layer	Except system Level DoS	Transport layer
Real Time IPS	●	◐	◐	●
IPS against data link, network, transport layers and the system security level	●	Transport layer	Except system Level DoS	●
Alert user approach	●	●	●	●
	◐ : Covered		● : Not Covered	

Comparison of Raspberry House with Other Related Research (cont'd.)

IoT Gateway Design

- **Shang et al. 2013** propose a novel configurable smart IoT gateway. However, the proposed gateway does not support WiFi communication, and it does not include any security policy.
- **Gló´ria et al. 2017** propose an IoT gateway for creating a smart swimming pool dedicated to real-time monitoring and remote control of a swimming pool. However, the proposed gateway only serves a specific application and does not provide security functions .
- **Zachariah et al. 2021** propose an IoT gateway architecture that uses a generic IoT gateway as a software service on modern smartphones to provide generic and ubiquitous Internet access to Bluetooth Low Energy (BLE)-connected IoT devices.

Research Gaps

IoT Gateways

- Design for specific Applications.
- Lack of security functions.

IoT IDS and IPS Approaches

- The approaches proposed by most of the research are **limited to detecting DoS attacks** in IoT without providing prevention solutions.
- It is difficult to build and run the IDS and IPS on **small IoT devices** (i.e., resource-constrained devices).
    - limited processing speed, storage capacity, and communication bandwidth.
- Many IDS and IPS for IoT DoS attacks are based on **machine learning approaches**, which may result in high operating costs, time cost and resource usage.
- Many IDS and IPS approaches can **only be run once** .

Contributions

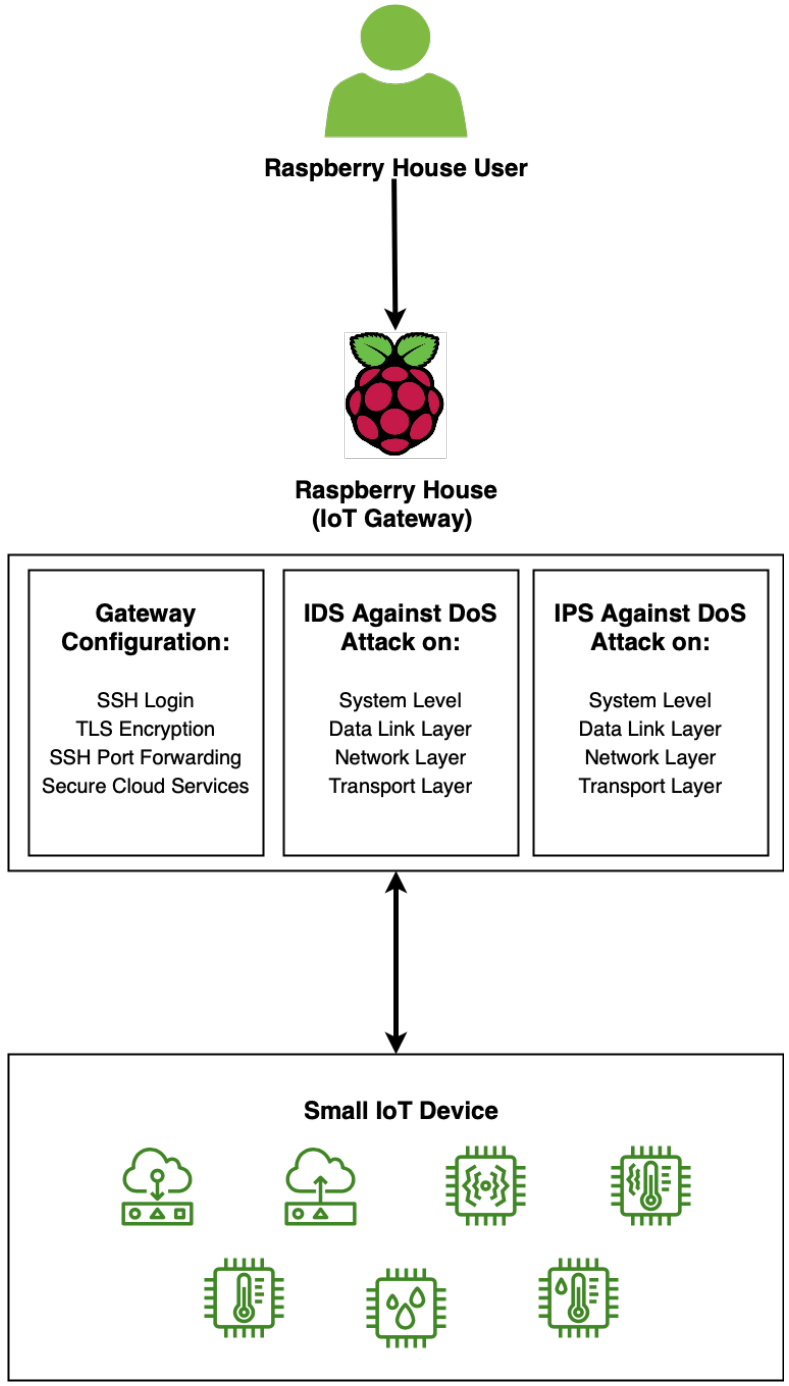
The main contribution of this thesis is to **propose a novel intrusion detection and prevention system against DoS attacks on IoT devices at all levels - data link layer, network layer, transport layer and system security level**.

- Add security functions to the proposed **Raspberry House**.
- The proposed IDS & IPS methods are **compatible with resource-constrained small IoT devices**.
- The proposed IDS and IPS methods are mainly based on **shell scripts** and **systemd services.**
- **Tested the average delay** of the proposed IDS and IPS methods at a given time for reference by other researchers.
- Provide researchers with suggested **optimal IPS solutions** of different DoS attacks for their further study.

03 Proposed Raspberry House Design

Raspberry House Architecture Design
Raspberry House IDS Design
Raspberry House IPS Design
Advantages of The Designed Architecture

Raspberry House Architecture Design



Raspberry House IDS Design

Data Link layer

- Algorithm 1 shows the algorithm of the proposed **Python script** to detect deauthentication packets.

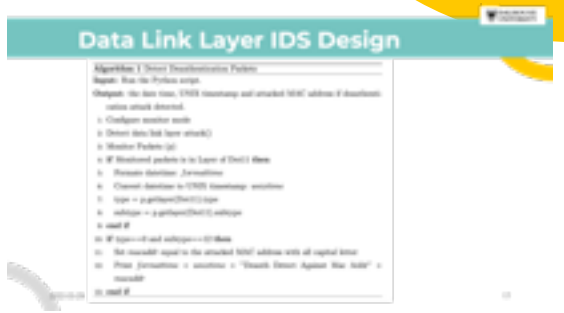
Network Layer & Transport Layer

- Default **Snort community rules** and **custom Snort rules** are used, which can reduce processing speed and RAM usage load.

System Security Level

- **Built in watchdog timer (WDT)** and **external WDT** (gentle WDT and delayed WDT) are used.

Note: We use publish/subscribe mode of **Message Queuing Telemetry Transport (MQTT)** broker to transmit message. Once the DoS attacks have been detected, the administrator will receive an alert through **email** and **Twitter message**. In addition, different color of **LEDs** connected to Raspberry House will blink according to the type of DoS attacks to alert the user.



Data Link Layer IDS Design

Algorithm 1 Detect Deauthentication Packets

Input: Run the Python script.

Output: the date time, UNIX timestamp and attacked MAC address if deauthentication attack detected.

1: Configure monitor mode
2: Detect data link layer attack()
3: Monitor Packets (p)
4: **if** Monitored packets is in Layer of Dot11 **then**
5: Formate datetime: *formattime*
6: Convert datetime to UNIX timestamp: *unixtime*
7: type = p.getlayer(Dot11).type
8: subtype = p.getlayer(Dot11).subtype
9: **end if**
10: **if** type==0 and subtype==12 **then**
11: Set *macaddr* equal to the attacked MAC address with all capital letter
12: Print *formattime* + *unixtime* + "Deauth Detect Against Mac Addr" + *macaddr*
13: **end if**

System Security Level IDS Design

- The Raspberry House communicates with the watchdog timer at a set interval to indicate that it is still working normally.
- If Raspberry House **does not output a signal**, **output too many signals**, or **output a different signal than the pre-set pattern**, the watchdog timer will detect the system anomaly.
- The **external WDT** is aiming to add an extra layer of security to the Raspberry House.
    - Since there is no guarantee that a failing Raspberry House will be able to monitor itself and detect failures when it is only monitored by the built-in WDT.

Raspberry House IPS Design

Data Link Layer

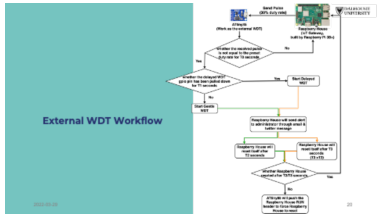
- When a deauthentication attack is detected, the IPS in the Raspberry House will be triggered if the deauthentication packets received by the Raspberry House **exceed a defined threshold**.
- The proposed IPS will **block the WiFi interface** and then re-enable the WiFi interface after a predefined interval.

Network Layer & Transport Layer

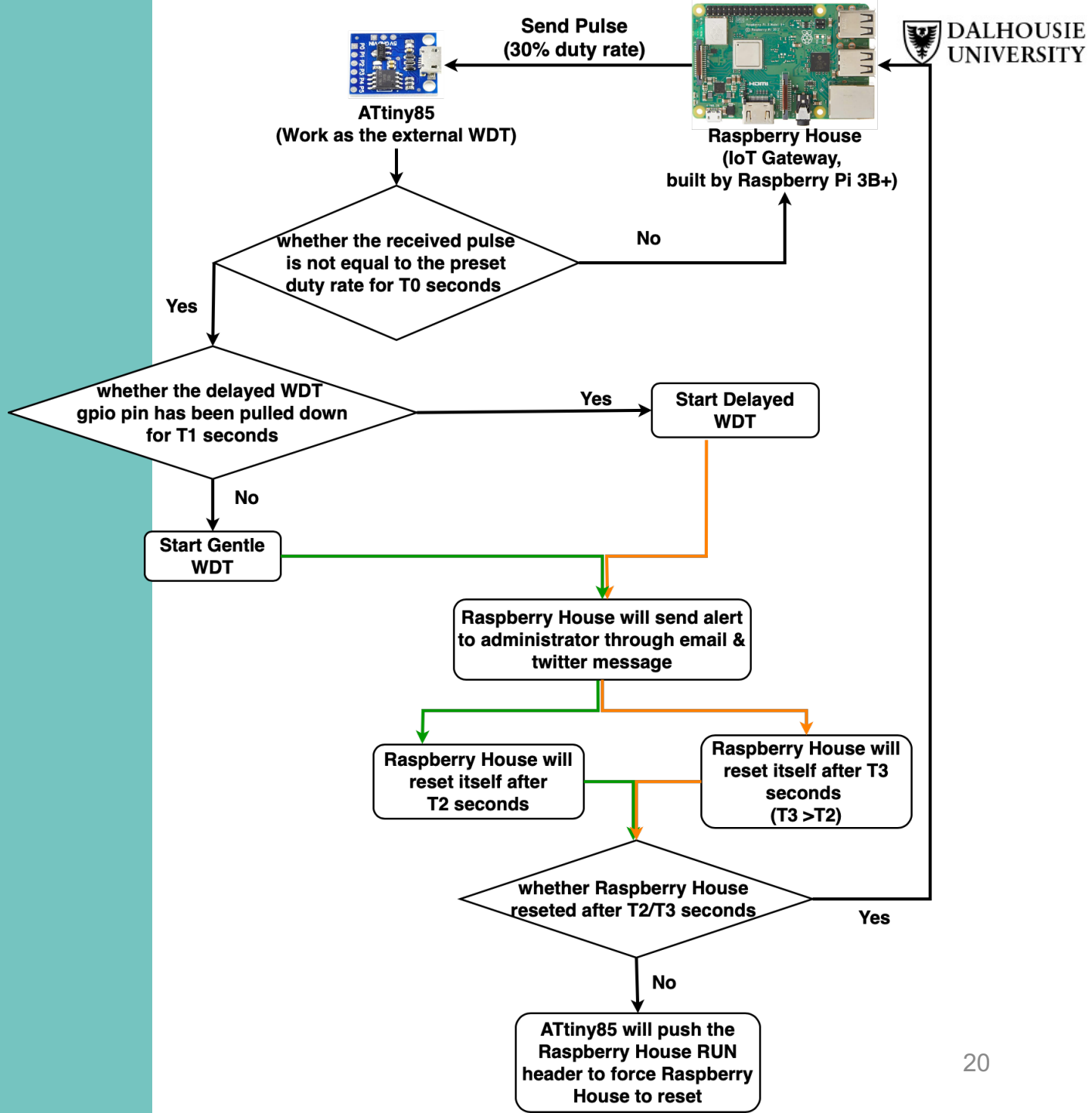
- **Snort's inline mode** & **custom ruleset**.

System Security Level

- Built in WDT & external WDT (gentle WDT and delayed WDT).
- Next slide shows the workflow of the external watchdog timer (WDT).



External WDT Workflow



Advantages of Designed Architecture

- The proposed Raspberry House can **provide a secure environment for IoT devices**.
- The proposed Raspberry House **considers IDS and IPS against DoS attacks in the data link layer, network layer, transport layer** and **system security level**.
- The proposed Raspberry House can **detect and prevent DoS attacks in real time** to enhance the security of small IoT devices which connect to it.
- The proposed Raspberry House **considers the severity of DoS attacks** and **decides on preventive measures based on severity**.
- The proposed Raspberry House can **send alerts to users** through email and Twitter when a DoS attack is detected.
- The proposed Raspberry House is **low cost and easy to carry**, to ensure that most IoT researchers can afford it and use it to work anywhere.

04

Experimental Implementation

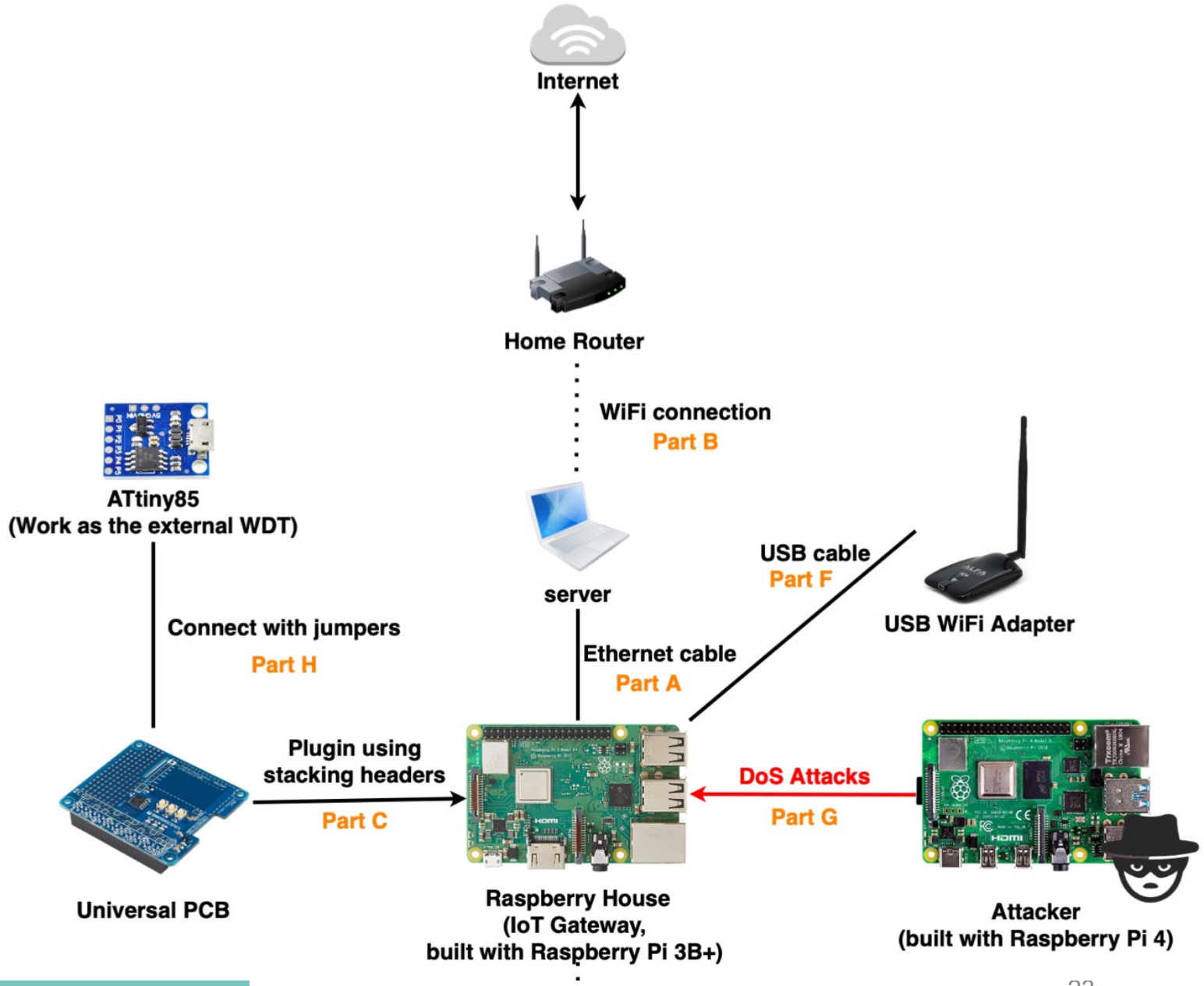
Raspberry House Architecture Implementation

Raspberry House IDS and IPS Implementation

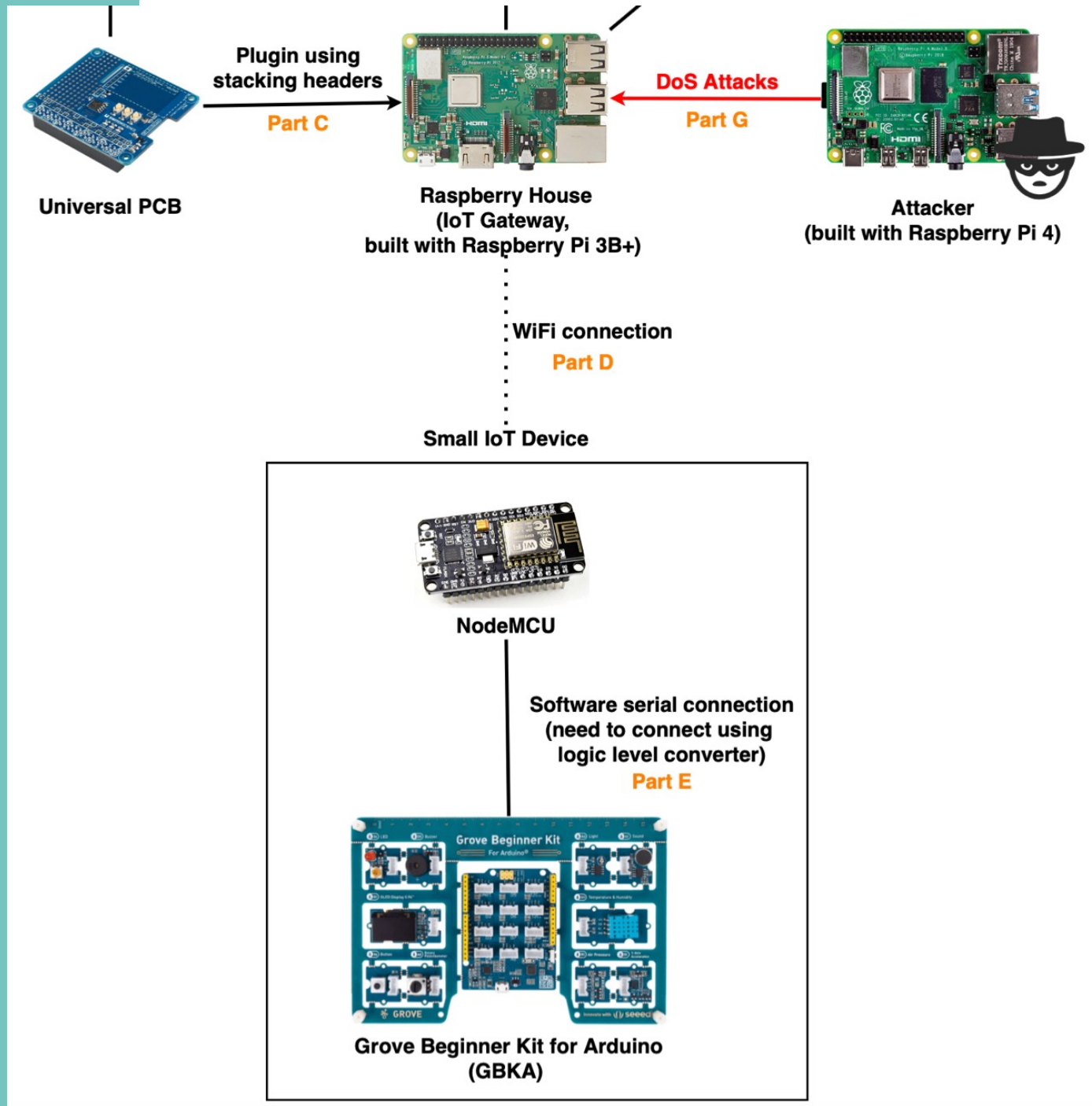
Evaluation Scenario Implementation

Cost of The Proposed Raspberry House

Raspberry House Architecture Implementation



Raspberry House Architecture Implementation (cont'd.)



Raspberry House IDS and IPS Implementation

- Raspberry House's IDS and IPS are based on **shell scripts** and every shell script's running information will be stored in logs for future analysis.
- To enable these shell scripts to run automatically when the Raspberry House is powered, we use the **systemd service**.
- Table below shows an example of the systemd services created in our thesis and the shell scripts used by the services.

Service Name (.service)	Description	Run After (.service)	Shell Script Used By This Service (.sh)	Description
SYN_ twitterAlert	Alert administrator through twitter message when get SYN flood attack.	- network (.target) - mosquitto	- SYNtwitter -SYN_twitterAlert	- SYNtwitter: send a twitter alert to the administrator with the type of the DoS attack (SYN flood attack) and the timestamp. - SYN_twitterAlert: use mosquitto_sub to subscribe Topic B. If receive any SYN flood DoS attack, then run SYNtwitter.sh

Example of systemd service and its shell scripts

Snort Custom Rules

Figure below shows the snort custom rules used in this thesis to drop the malicious packets of SYN flood attack and ICMP flood attack.

```
drop icmp any any -> $HOME_NET any (msg:"ICMP Flood Attack Rejected"; detection_filter\
:track by_dst, count 30, seconds 10; sid:1000001; rev:1;classtype:icmp-event;)
drop tcp any any -> $HOME_NET 80 (msg:"SYN Flood Attack Rejected"; detection_filter:tr\
ack by_dst, count 30, seconds 10; sid:1000002;rev:2;classtype:attempted-dos;)
```

Raspberry House Custom Snort Rules

Evaluation Scenario Implementation

- The DoS attacks in this thesis for **penetration testing** using the Kali Linux is **deauthentication attack**, **ICMP flood attack** and **SYN flood attack**.
- **Data Link Layer**: use **aircrack-ng suite** to perform the deauthentication attack.
- **Network & Transport Layers**: use **hping3** to perform ICMP/SYN flood attacks.
- **System Security Level**: we perform the **bash fork bomb** on Raspberry House to test the performance of WDTs.

Raspberry House Implementation Cost

Since this research aims to design a portable and **low-cost IoT gateway**, we need to ensure that the **proposed Raspberry House is affordable for most IoT researchers/engineers**. The total cost to build the system is **$188CA**, including Raspberry Pi 3B+ Extreme Kit $99CA, universal PCB $26CA, ATtiny85 board $3CA, and USB WiFi adapter $60CA. Note that all hardware we used in this thesis was bought from Amazon Canada.

05

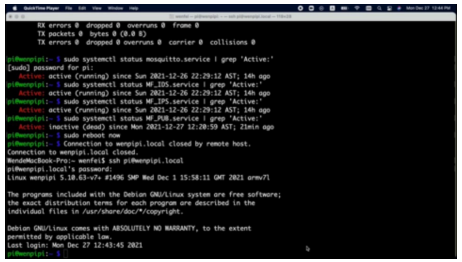
Experimental Results & Evaluation

Data Link Layer
Network Layer and Transport Layer
System Security Level
Summary

Data Link Layer

Data Link Layer – Deauthentication Attack

- The deauthentication attack overloaded the Raspberry House wireless interface performance approximately 1 minute after execution.
- The average delay for Raspberry House to detect deauthentication packets is about **5.3 milliseconds**.
- Demo Video 1.



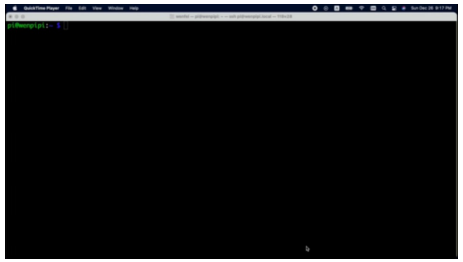
```
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

pi@wenpipi:~ $ sudo systemctl status mosquitto.service | grep 'Active:'
[sudo] password for pi:
   Active: active (running) since Sun 2021-12-26 22:29:12 AST; 14h ago
pi@wenpipi:~ $ sudo systemctl status MF_IDS.service | grep 'Active:'
   Active: active (running) since Sun 2021-12-26 22:29:12 AST; 14h ago
pi@wenpipi:~ $ sudo systemctl status MF_IPS.service | grep 'Active:'
   Active: active (running) since Sun 2021-12-26 22:29:12 AST; 14h ago
pi@wenpipi:~ $ sudo systemctl status MF_PUB.service | grep 'Active:'
   Active: inactive (dead) since Mon 2021-12-27 12:20:59 AST; 21min ago
pi@wenpipi:~ $ sudo reboot now
pi@wenpipi:~ $ Connection to wenpipi.local closed by remote host.
Connection to wenpipi.local closed.
WendeMacBook-Pro:~ wenfei$ ssh pi@wenpipi.local
pi@wenpipi.local's password:
Linux wenpipi 5.10.63-v7+ #1496 SMP Wed Dec 1 15:58:11 GMT 2021 armv7l

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Mon Dec 27 12:43:45 2021
pi@wenpipi:~ $
```

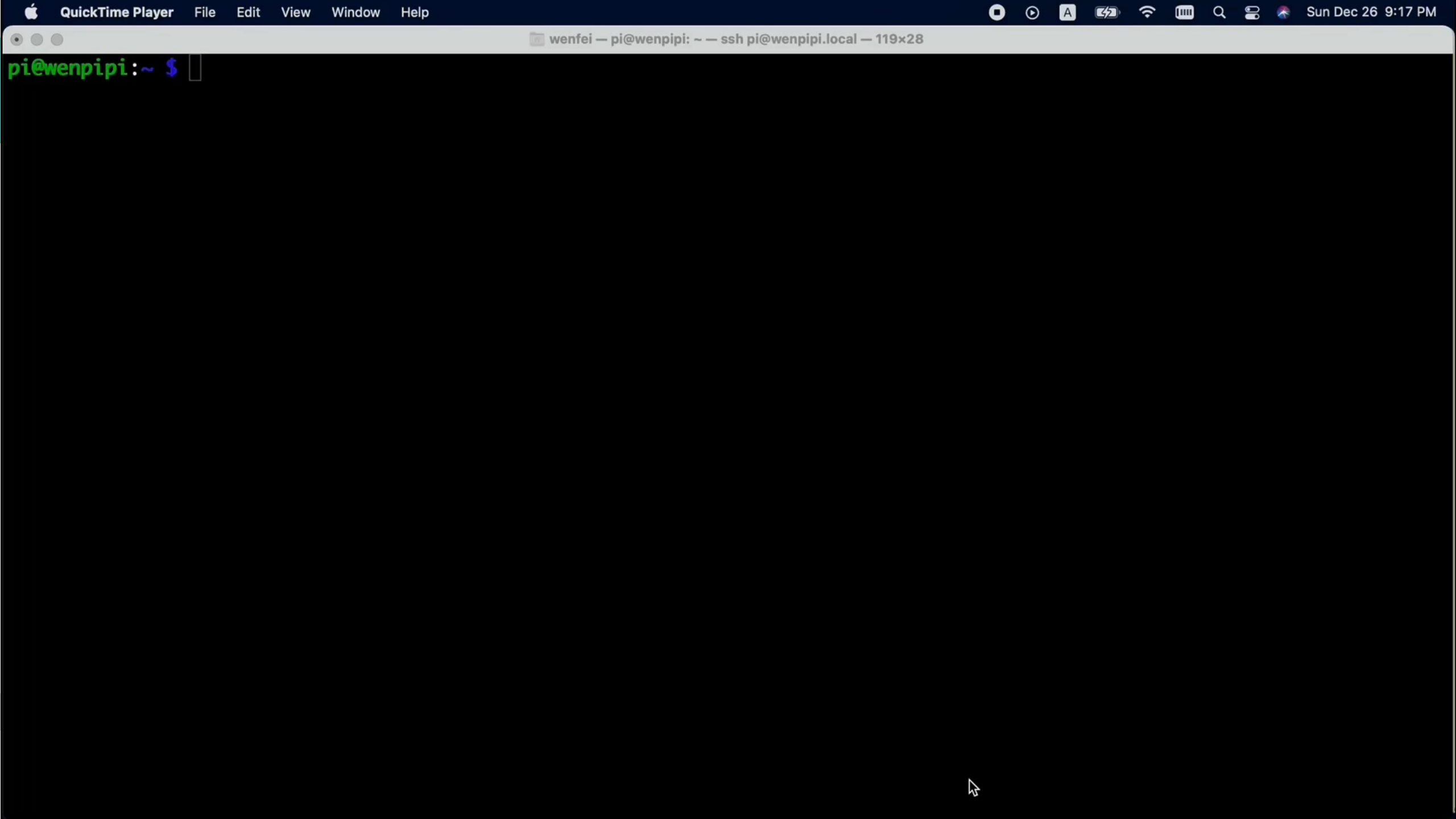
Network Layer & Transport Layer

Network Layer – ICMP Flood Attack

Transport Layer – SYN Flood Attack

- The results show that in the **5-minute test**, the Raspberry House hardware performance is not affected by the ICMP flood attack or SYN flood attack.
- The **delay of each ICMP flood attack** to trigger the drop rule is about **9 milliseconds** for another ICMP flood attack.
- The **delay of each SYN flood attack** to trigger the drop rule is about **13 milliseconds** for another SYN flood attack.
- Demo Video 2.




QuickTime Player File Edit View Window Help
Sun Dec 26 9:17 PM
wenfei — pi@wenpipi: ~ — ssh pi@wenpipi.local — 119×28
pi@wenpipi:~ $

System Security Level

System Security Level – Bash Fork Bomb Attack

- As shown in Figure A, compared to the other two WDTs, the **hardware watchdog can immediately detect** the system anomaly of the Raspberry House and restart the Raspberry House after 15 seconds.
- To **test the performance of the other two WDTs**, we did a GPIO test on the Raspberry House.
- The result is shown in Figure B and Figure C. If the **gentle WDT** is enabled, the Raspberry House will be restarted after **60 seconds**; if the **delayed WDT** is enabled, the Raspberry House will be restarted after **120 seconds**.

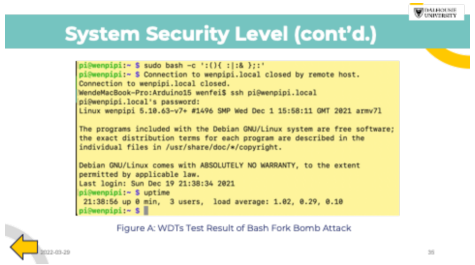


Figure A: WDTs Test Result of Bash Fork Bomb Attack

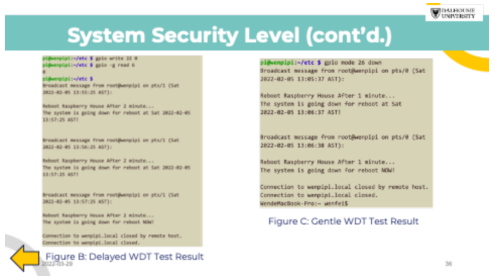


Figure B: Delayed WDT Test Result

Figure C: Gentle WDT Test Result

System Security Level (cont'd.)

```
pi@wenpipi:~ $ sudo bash -c ':(){ :|:& };:'
pi@wenpipi:~ $ Connection to wenpipi.local closed by remote host.
Connection to wenpipi.local closed.
WendeMacBook-Pro:Arduino15 wenfei$ ssh pi@wenpipi.local
pi@wenpipi.local's password:
Linux wenpipi 5.10.63-v7+ #1496 SMP Wed Dec 1 15:58:11 GMT 2021 armv7l

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Sun Dec 19 21:38:34 2021
pi@wenpipi:~ $ uptime
 21:38:56 up 0 min,  3 users,  load average: 1.02, 0.29, 0.10
pi@wenpipi:~ $
```

Figure A: WDTs Test Result of Bash Fork Bomb Attack

System Security Level (cont'd.)

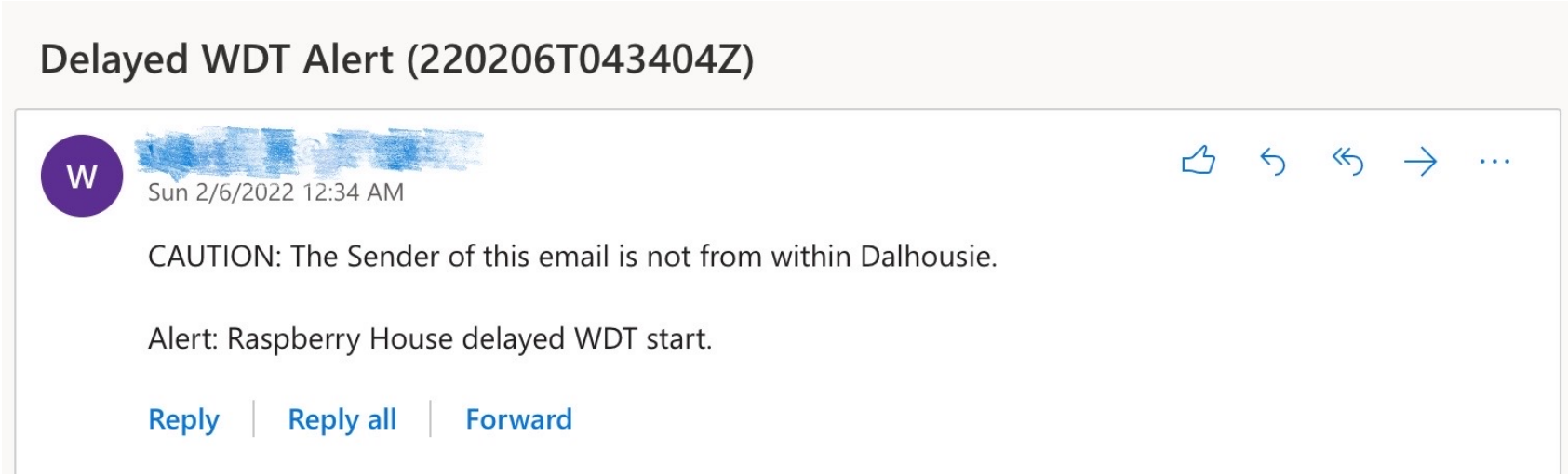
```
pi@wenpipi:~/etc $ gpio write 22 0
pi@wenpipi:~/etc $ gpio -g read 6
0
pi@wenpipi:~/etc $
Broadcast message from root@wenpipi on pts/1 (Sat
2022-02-05 13:55:25 AST):

Reboot Raspberry House After 2 minute...
The system is going down for reboot at Sat 2022-02-05
13:57:25 AST!


Broadcast message from root@wenpipi on pts/1 (Sat
2022-02-05 13:56:25 AST):

Reboot Raspberry House After 2 minute...
The system is going down for reboot at Sat 2022-02-05
13:57:25 AST!


Broadcast message from root@wenpipi on pts/1 (Sat
2022-02-05 13:57:25 AST):

Reboot Raspberry House After 2 minute...
The system is going down for reboot NOW!

Connection to wenpipi.local closed by remote host.
Connection to wenpipi.local closed.
```

Figure B: Delayed WDT Test Result

```
pi@wenpipi:~/etc $ gpio mode 26 down
Broadcast message from root@wenpipi on pts/0 (Sat
2022-02-05 13:05:37 AST):

Reboot Raspberry House After 1 minute...
The system is going down for reboot at Sat
2022-02-05 13:06:37 AST!


Broadcast message from root@wenpipi on pts/0 (Sat
2022-02-05 13:06:38 AST):

Reboot Raspberry House After 1 minute...
The system is going down for reboot NOW!

Connection to wenpipi.local closed by remote host.
Connection to wenpipi.local closed.
WendeMacBook-Pro:~ wenfei$
```

Figure C: Gentle WDT Test Result

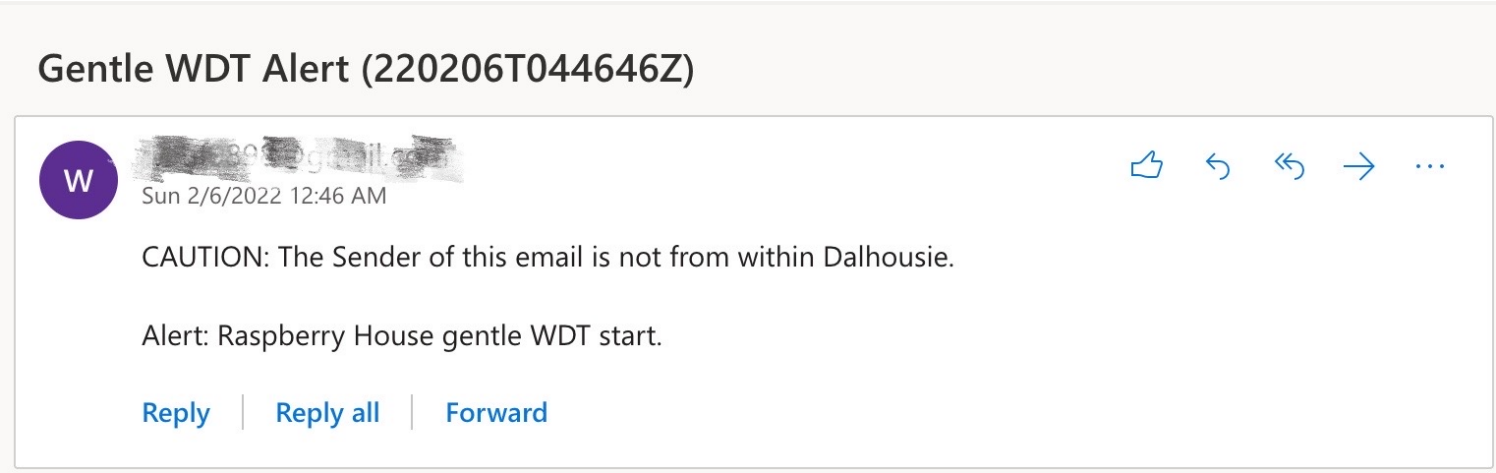
System Security Level (cont'd.)



Delayed WDT Email Alert

Delayed WDT Twitter Message Alert

System Security Level (cont'd.)



Gentle WDT Email Alert

Gentle WDT Twitter Message Alert

Summary

We have proposed **a novel and comprehensive IDS & IPS approach which covered detection and prevention methods in various DoS attacks** at the:

- **Data Link Layer – Deauthentication Attack**
- **Network Layer – ICMP Flood Attack**
- **Transport Layer – SYN Flood Attack**
- **System Security Level – Bash Fork Bomb Attack**

Summary (cont'd.)

Also, **many related studies only consider IoT IDS and ignore IPS**. Therefore, we summarize the proposed IPS covered in this thesis and show our recommended IPS approach for different DoS attacks as shown.

	Snort custom rules & inline mode	Block/unblock wlan interface of Raspberry House	Raspberry House built in (hardware) WDT	Raspberry House gentle WDT	Raspberry House delayed WDT
Deauthentication Attack (Management Frame Attack)		C best choice	C		
ICMP Flood DoS Attack	C best choice		C		
SYN Flood DoS Attack	C best choice		C		
Bash Fork Bomb Attack			C best choice	C	C
Other continuous flood type DoS attacks			C	C	C best choice
Other 'gentle' DoS attack such as inode problem				C best choice	C
	C: covered				

06

Conclusion & Future Work

Conclusion

Future Work

Conclusion

- The results show that Raspberry House IDS and IPS can **detect, alert and prevent DoS attacks** in real time, including
  - **Data Link Layer**
  - **Network Layer**
  - **Transport Layer**
  - **System Security Level**

- Summarize the DoS attacks proposed in this thesis and the feasible IPS. Provide users with our **suggested optimal solutions** for different DoS attacks.

- Raspberry House is **cost effective** since it can be built using off-the-shelf devices.

Future Work

DALHOUSIE UNIVERSITY

- Duplicate Raspberry House in **various research environments**.

- Plan to implement **more lightweight IoT gateway applications** dedicated to low-performance IoT devices to further improve the performance of Raspberry House.

- Plan to **design other IDS and IPS** for Raspberry House to prevent **other common types of IoT attacks** such as Man-in-the-Middle attacks, making our Raspberry House more robust.

Publications

Conference paper

- **W. Fei, H. Ohno, and S. Sampalli, "Design and implementation of raspberry house: An IoT security framework," in *2020 IEEE International Conference on Internet of Things and Intelligence System (IoTaIS)*, 2021, pp. 1–7.**

Journal paper

- **W. Fei, H. Ohno, and S. Sampalli, "A Systematic Review of IoT Security: Research Potential, Challenges and Future Directives" – manuscript under preparation, to be submitted to Computers and Security Journal.**

Acknowledgement

I would like to express my deepest and sincere gratitude to my supervisor, Prof. Srinivas Sampalli, for his guidance, support, encouragement and patience during my study and for providing all the necessary opportunities to make this work possible.

Very special thanks to Prof. Hiroyuki Ohno (Kanazawa University, Japan) for his excellent support, guidance and knowledge in the implementation and testing of Raspberry House.

DALHOUSIE UNIVERSITY

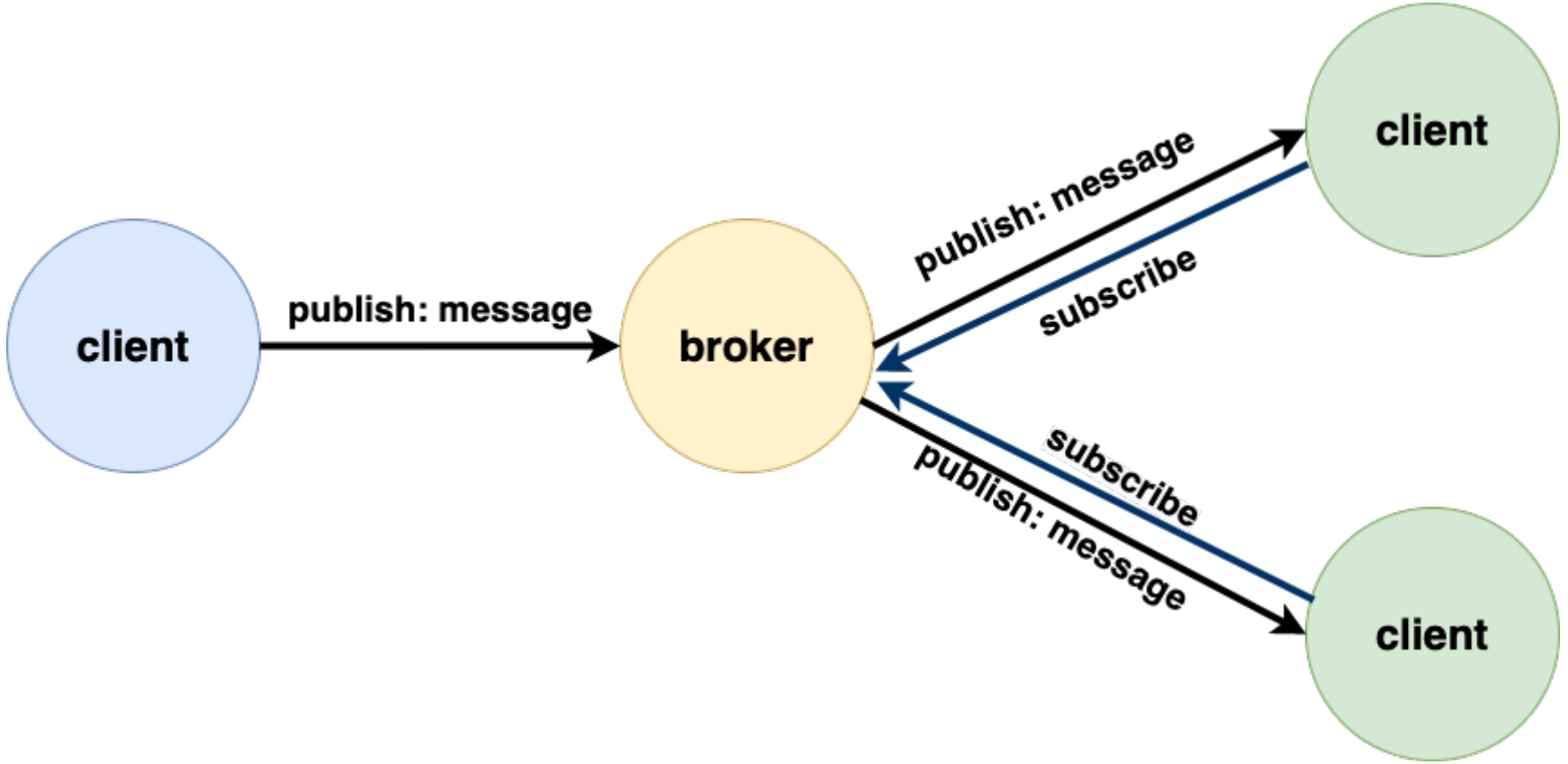
THANKS

ADDITIONAL SLIDES

Message Queuing Telemetry Transport (MQTT)

- MQTT, Message Queuing Telemetry Transport, is a lightweight IoT messaging protocol used to collect the measurement data from remote sensors and send it to the server via TCP.
- It uses a publish/subscribe (pub/sub) technology and allows simple data flow between different devices.
- Clients can publish a message on a topic or subscribe to a particular topic through the broker.
    - Topics represent the destination address for a message sent by the MQTT broker.
- The client who subscribes to a topic will receive all the messages published to that topic.
- Figure 25 shows the MQTT mechanism.

MQTT Cont'd



MQTT Mechanism

MOTT Topic for Systemd Services

We create two MQTT topics for these services. Table below classifies the services based on the MQTT topic name.

MQTT Topic	
wen/data1	**wen/data2**
MF_IDS.service	snort_PUB.service
MF_IPS.service	ICMP_blink.service
MF_PUB.service	SYN_blink.service
MF_kotoriotokoAlert.service	ICMP_emailAlert.service
MF_emailAlert.service	SYN_emailAlert.service
	ICMP_twitterAlert.service
	SYN_twitterAlert.service
mosquitto.service (used in both topics)	
*Notice: the service to setup GPIO pins of Raspberry House (WDTgpioSetup.service) and enable the USB WiFi adapter to monitor the network traffic (enableUSBadapter.service) do not need MQTT service to transfer data. In addition, the services for WDT (watchdog.service, WDTnotification.service, and kickWDT.service) also do not need to use MQTT service.	

Classify Services Based on MQTT Topic

IEEE802.11 Management Frame

- In order to be able to connect to an access point (AP), the client must have established a connected state before it can start exchanging data messages. The following are the three 802.11 connection states:
    - Unauthenticated and Unassociated
    - Authenticated but Unassociated
    - Authenticated and Associated
- When clients wish to disconnect from the AP, they need to send a deauthentication frame to the AP.
- However, according to the 802.11 networking standard, deauthentication or disassociation frames are unencrypted and do not require authentication.
- Therefore, an attacker can use this to easily spoof the MAC address of a client or AP in order to generate deauthentication packets on behalf of the client or AP.

IEEE802.11 Management Frame (Cont'd)

- Table below shows the subtype of management frames.

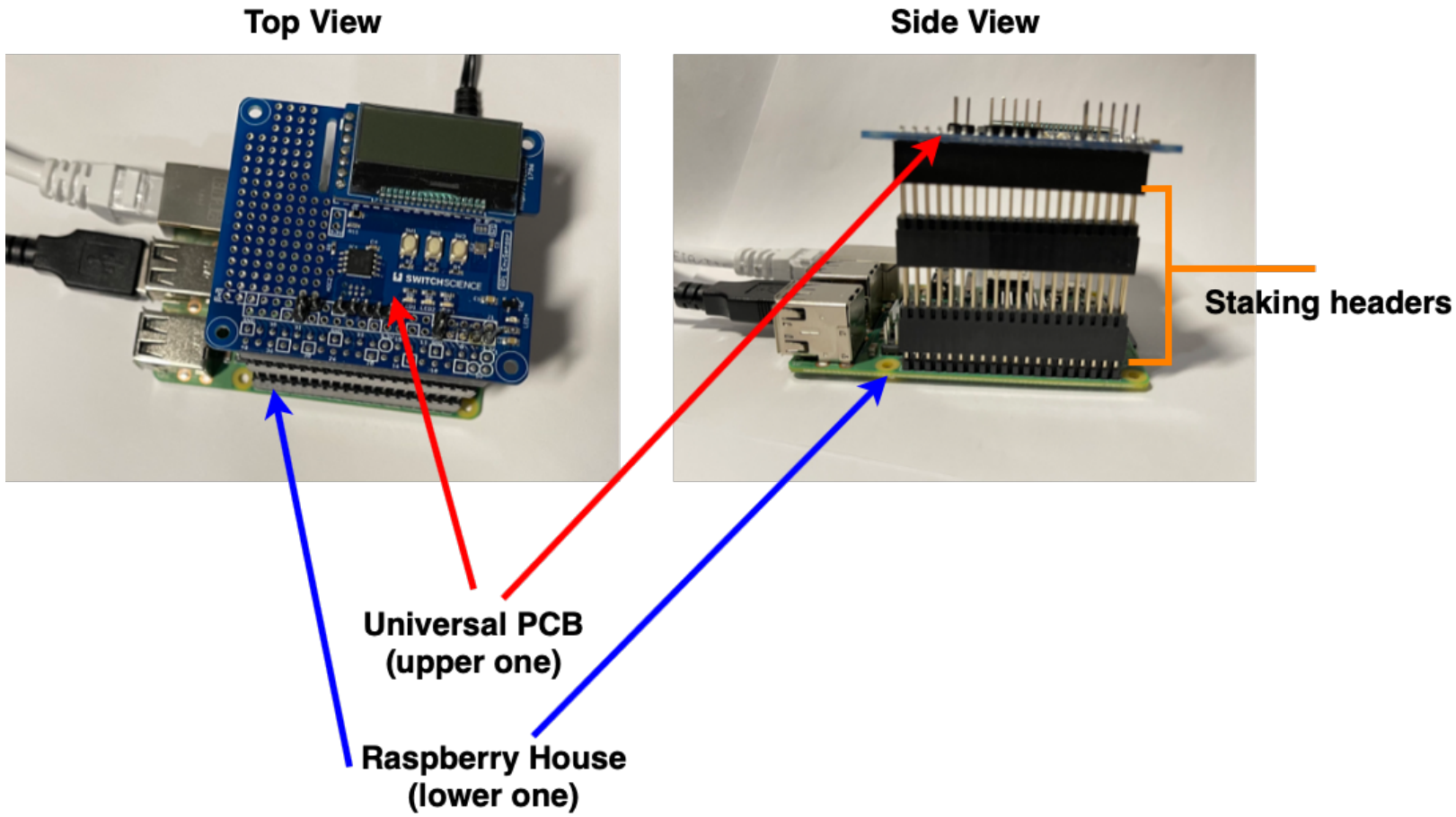
Type Value	Type Description	Subtype Value	Subtype Description
00	Management	0000	Association Request
		0001	Association Response
		0010	Re-association Request
		0011	Re-association Response
		0100	Probe Request
		0101	Probe Response
		1000	Beacon
		1001	Announcement Traffic Indication Massage (ATIM)
		1010	Disassociation
		1011	Authentication
		1100	Deauthentication
		1111	Reserved

Sub-types of Management Frames

Inode problem

- Linux file systems use inodes to store information about files, directories, and devices. Inodes are the basis of the Linux file system.
- Inodes manage metadata about files and are an important part of the inner workings of Linux.
- Metadata includes file type, permissions, owner ID, group ID, file size, last access time, last modification time, soft/hard links, and access control lists.
- Therefore, an attacker can consume all the free inodes on the disk by creating a large number of temporary empty folders.

Connection Between Universal PCB and Raspberry House



MSMTP & Kotoriotoko

- The msmtp is an SMTP client that can be used to send mail from MUA (mail user agent).
- Kotoriotoko is a command set for operating Twitter, makes the users possible to operate Twitter on the command- line interface (CLI). Table below shows the functionality provided by the Kotoriotoko command.

Function	Detail
Posting	- Tweet - Retweet - Cancel a tweet - Like - Unlike
Tweets Viewing	- View Somebody's timeline - Search tweets by keywords
User Controlling	- Follow somebody - Unfollow somebody - List users who you following - List users who you are followed - View Somebody's info
Direct Message Managing	- Send - Receive - Delete - List
Other functions	- View trend list - Gather tweets in bulk continuously

The Functionality Provided By The Kotoriotoko [1]

"kotoriotoko: KOTORIOTOKO (little bird man) – extremely compatible and sustainable twitter application written in shell script," https://github.com/ ShellShoccar-jpn/kotoriotoko.

DALHOUSIE UNIVERSITY

Raspberry House IDS & IPS Design

(summary & WDTs Description)

Description of the Proposed Watchdog Timer (WDT) Schemes

WDT Type	Description
Built in WDT	When the DoS attacks make Raspberry House freeze and are easily detected, the hardware watchdog will restart the Raspberry House automatically after a predefined interval.
Gentle WDT	When DoS attacks which make Raspberry House freezes, but difficult to be detected immediately, the gentle WDT will restart Raspberry House automatically after a predefined interval.
Delayed WDT	The delayed WDT will run when the DoS attack is massive and sustained. It will restart the Raspberry House automatically after a predefined interval (the time interval will be relatively long compare to the other two WDT since it will restart the Raspberry House after the large amount DoS attacks).

Proposed Raspberry House's IDS and IPS Design

DoS Attacks on...	IDS Design	IPS Design
Data Link Layer	- Develop a Python script to detect the DoS attacks against IEEE802.11 network. The malicious packets can be detected based on the type and subtype of their frames. - Blink LED to notify the user. - Send alert through email and twitter to the user.	Block the WiFi interface for a predefined interval and then unblock the WiFi interface. Repeat the above operation until the Raspberry House no longer receives any malicious packets.
Network Layer & Transport Layer	- Use community ruleset and custom rules to detect the attacks. - Blink LED to notify the user. - Send alert through email and twitter to user.	Use inline mode and custom rules to drop the malicious packets.
System Security Level	- Design different kinds of WDT (built-in WDT, gentle WDT, and delayed WDT) to detect the DoS attacks based on the serious of the attacks. - Send alert through email and twitter to the user.	The WDTs will reboot Raspberry House with different predefined intervals to prevent DoS attacks.

DALHOUSIE UNIVERSITY

Raspberry House IDS and IPS Implementation

(systemd services & corresponding shell script)

Summary of The Systemd Services and Shell Scripts

Service Name (.service)	Description	Run After (.service)	Shell Script Used By This Service (.sh)	Description
mosquitto	Run mosquitto (MQTT broker) in the background.	- network (.target)	mosquitto-wrapper	Run mosquitto based on its configuration file.
enable-USBadapter	Enable USB WiFi Adapter as monitor mode to monitor the network traffic.	- network (.target)	enableAdapter	Enable the wireless interface of USB WiFi adapter as the monitor mode to monitor the network traffic.
MF_PUB	Run Python script which can detect the deauthentication (deauth) packets and publish to Topic A with local MQTT publisher (mosquitto_pub).	- network (.target) - mosquitto - MF_IDS - MF_IPS - MF_kotori-otokoAlert - MF_emailAlert	- printDeauth.py - MF_Python	- printDeauth.py: Python script uses to detect the deauth packets. - MF_Python: a shell script to publish the result of printDeauth.py to Topic A using local MQTT publisher (mosquitto_pub)
MF_IDS	IDS for deauth attacks, which will blink the red LED on universal PCB after receive deauth packets.	- network (.target) - mosquitto	- MF_Blink	Use local MQTT subscriber (mosquitto_sub) to subscribe Topic A. If receive the deauth packets, then blink the red LED on universal PCB.

Summary of The Systemd Services and Shell Scripts (cont'd.)

Service Name (.service)	Description	Run After (.service)	Shell Script Used By This Service (.sh)	Description
MF_IPS	IPS for deauth attacks, which will block the WiFi interface of Raspberry House.	- network (.target) - mosquitto	- disableWiFi - MF_IPS	- disableWiFi: block WiFi interface of Raspberry House, and unblock it after 15 seconds. - MF_IPS: use mosquitto_sub to subscribe Topic A. If the deauth packets received is greater than 5 (threshold), then start to run disableWiFi.sh.

Summary of The Systemd Services and Shell Scripts (cont'd.)

Service Name (.service)	Description	Run After (.service)	Shell Script Used By This Service (.sh)	Description
MF_kotori-otokoAlert	Alert administrator through twitter message when get deauth attack.	- network (.target) - mosquitto	- kotoriotokoAlert - MF_kotoriotoko Alert	- kotoriotokoAlert: send a twitter alert to the administrator with the type of the DoS attack (deauth attack) and the timestamp. - MF_kotoriotokoAlert: use mosquitto_sub to subscribe Topic A. If receive any deauth packets, then run kotoriotokoAlert.sh.
MF_ emailAlert	Alert administrator through email when get deauth attack.	- network (.target) - mosquitto	- emailAlert - MF_emailAlert	- emailAlert: send an email alert to the administrator with the type of the attack (deauth attack) and the timestamp. - MF_emailAlert: use mosquitto_sub to subscribe Topic A. If receive any deauth packets, then run emailAlert.sh.

Summary of The Systemd Services and Shell Scripts (cont'd.)

Service Name (.service)	Description	Run After (.service)	Shell Script Used By This Service (.sh)	Description
snort_PUB	Run Snort to catch the network layer attack (ICMP flood attack as an example in this thesis), and transport layer attack (SYN flood attack as an example in this thesis). Then publish the results to Topic B using local MQTT broker (mosquitto_pub).	- network (.target) - mosuiqtto - SYN_twitterAlert - SYN_emailAlert - SYN_blink - ICMP_twitterAlert - ICMP_emailAlert - ICMP_blink	snort_pub	Run Snort with its configuration in inline mode to capture any malicious packets defined in community ruleset and custom rules (SYN flood attack and ICMP flood attack). Then drop the packets defined in custom rules. Publish the results to Topic B using local MQTT publisher (mosquitto_pub).
SYN_blink	If receive any SYN flood DoS attack, then blink the blue LED on universal PCB.	- network (.target) - mosquitto	SYN_Blink	Use local MQTT subscriber (mosquitto_sub) to subscribe Topic B. If receive the SYN flood attack, then blink the blue LED on universal PCB.

Summary of The Systemd Services and Shell Scripts (cont'd.)

Service Name (.service)	Description	Run After (.service)	Shell Script Used By This Service (.sh)	Description
SYN_ twitterAlert	Alert administrator through twitter message when get SYN flood attack.	- network (.target) - mosquitto	- SYNtwitter -SYN_twitterAlert	- SYNtwitter: send a twitter alert to the administrator with the type of the DoS attack (SYN flood attack) and the timestamp. - SYN_twitterAlert: use mosquitto_sub to subscribe Topic B. If receive any SYN flood DoS attack, then run SYNtwitter.sh
SYN_ emailAlert	Alert administrator through email when get SYN flood attack.	- network (.target) - mosquitto	- SYNemail - SYN_emailAlert	- SYNemail: send an email alert to the administrator with the type of the DoS attack (SYN flood attack) and the timestamp. - SYN_emailAlert: use mosquitto_sub to subscribe Topic B. If receive any SYN flood DoS attack, then run SYNemail.sh
ICMP_blink	If receive any ICMP flood DoS attack, then blink the green LED on universal PCB.	- network (.target) - mosquitto	- ICMP_Blink	Use local MQTT subscriber (mosquitto_sub) to subscribe Topic B. If receive the ICMP flood attack, then blink the green LED on universal PCB.

Summary of The Systemd Services and Shell Scripts (cont'd.)

Service Name (.service)	Description	Run After (.service)	Shell Script Used By This Service (.sh)	Description
ICMP_ twitterAlert	Alert administrator through twitter message when get ICMP flood attack.	- network (.target) - mosquitto	- ICMPtwitter - ICMP_twitter Alert	- ICMPtwitter: send a twitter alert to the administrator with the type of the DoS attack (ICMP flood attack) and the timestamp. - ICMP_twitterAlert: use mosquitto_sub to subscribe Topic B. If receive any ICMP flood DoS attack, then run ICMPtwitter.sh
ICMP_ emailAlert	Alert administrator through email when get ICMP flood attack.	- network (.target) - mosquitto	- ICMPemail -ICMP_emailAlert	- ICMPemail: send an email alert to the administrator with the type of the DoS attack (ICMP flood attack) and the timestamp. - ICMP_emailAlert: use mosquitto_sub to subscribe Topic B. If receive any ICMP flood DoS attack, then run ICMPemail.sh

Summary of The Systemd Services and Shell Scripts (cont'd.)

Service Name (.service)	Description	Run After (.service)	Shell Script Used By This Service (.sh)	Description
WDTgpio-Setup	Service to setup all GPIO pins on Raspberry House at startup.	N/A	WDTgpioSetup	- Set initial mode of all GPIO pins for attack notification to OUT mode. (i.e., red LED pin which indicate deauth attack, green LED pin which indicate ICMP flood attack, and blue LED pin which indicate SYN flood attack). - Set the initial mode of GPIO pin A which work to generate the heart beat pulse on Raspberry House and send to WDT built by ATtiny85 to OUTPUT HIGH. - Set the initial mode of the GPIO pin B which work as a switch between gentle WDT when it set to HIGH) and delayed WDT (when it set to LOW) to OUTPUT HIGH. - Set the initial mode of the GPIO pin C which will receive the signal from ATtiny85 WDT (i.e. if it goes LOW level, then reboot Raspberry House) to INPUT_PULLUP.

Summary of The Systemd Services and Shell Scripts (cont'd.)

Service Name (.service)	Description	Run After (.service)	Shell Script Used By This Service (.sh)	Description
kickWDT	Use GPIO pin A on Raspberry House to generate heart beat pulse and send to WDT built by ATtiny85.	- network (.target) - mosquitto	kickWDT	Start the heart beat pulse on Raspberry House. The purpose of this is that if the pulse is different than the defined one, which means the system does not run as normal, then the external WDT built by ATTiny85 will start to run based on the state of GPIO pin B and GPIO pin C.
watchdog	The built in (hardware) WDT on Raspberry House.	multi-user (.target)	N/A	N/A
WDT-notification	Run the external WDT built by ATtiny85, and send alert to administrator through twitter message and email.	- network (.target) - WDTgpio-Setup	WDTdelayed	Check the WDT GPIO pin C, if its state has been turned to LOW level (which means the gentle WDT start to run) for 6 seconds, then start the gentle WDT. Meanwhile, check the GPIO pin which control the delayed WDT (GPIO pin B), if its state has been turned to LOW also for 6 seconds, then start the delayed WDT, otherwise remain run gentle WDT. Before the external WDT start reboot Raspberry House, it will send an alert to the administrator through email and twitter message with the type of the external WDT (gentle or delayed WDT) and the timestamp.

DALHOUSIE UNIVERSITY

Experimental Results & Evaluation

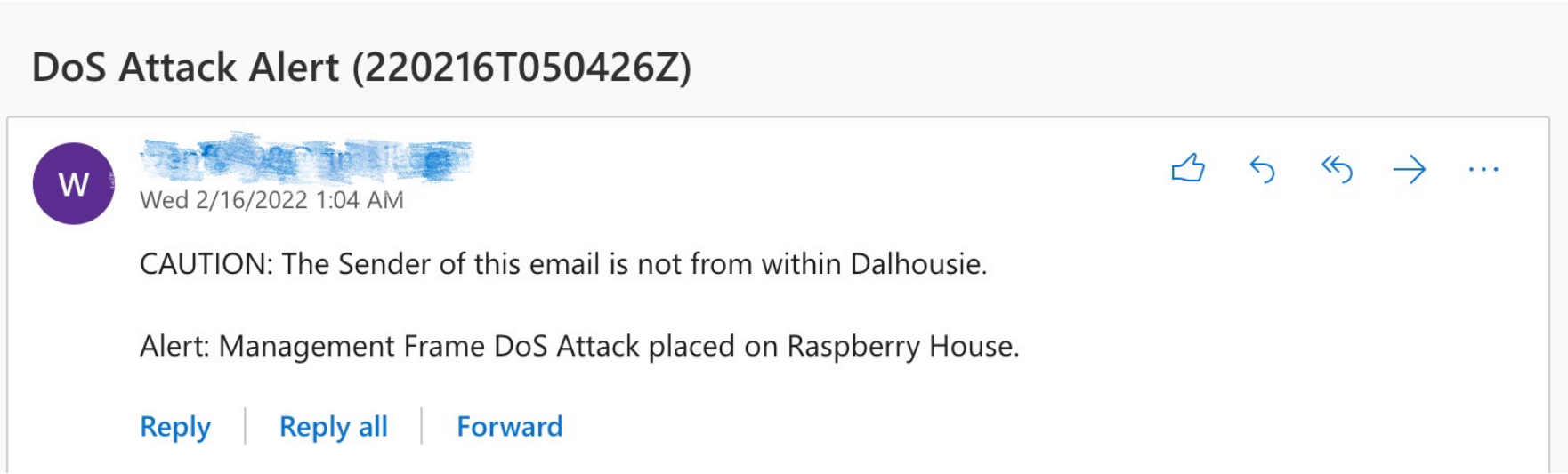
Data Link, Network, Transport Layers

Data Link Layer

```
^Cpi@wenpipi:~/Desktop sudo python printDeauth.py
2021-11-16 23:05:36 1637118336227 Deauth_Detect_Against_Mac_Addr B8:27:EB:79:20:82
2021-11-16 23:05:36 1637118336239 Deauth_Detect_Against_Mac_Addr B8:27:EB:79:20:82
2021-11-16 23:05:36 1637118336292 Deauth_Detect_Against_Mac_Addr B8:27:EB:79:20:82
2021-11-16 23:05:36 1637118336295 Deauth_Detect_Against_Mac_Addr B8:27:EB:79:20:82
2021-11-16 23:05:36 1637118336342 Deauth_Detect_Against_Mac_Addr B8:27:EB:79:20:82
2021-11-16 23:05:36 1637118336363 Deauth_Detect_Against_Mac_Addr B8:27:EB:79:20:82
2021-11-16 23:05:36 1637118336632 Deauth_Detect_Against_Mac_Addr B8:27:EB:79:20:82
2021-11-16 23:05:36 1637118336667 Deauth_Detect_Against_Mac_Addr B8:27:EB:79:20:82
2021-11-16 23:05:36 1637118336745 Deauth_Detect_Against_Mac_Addr B8:27:EB:79:20:82
2021-11-16 23:05:36 1637118336793 Deauth_Detect_Against_Mac_Addr B8:27:EB:79:20:82
2021-11-16 23:05:36 1637118336797 Deauth_Detect_Against_Mac_Addr B8:27:EB:79:20:82
2021-11-16 23:05:36 1637118336815 Deauth_Detect_Against_Mac_Addr B8:27:EB:79:20:82
```

Deauthentication Attack Detection Results

Data Link Layer (cont'd.)



Deauthentication Attack Alert Through Email

Deauthentication Attack Alert Through Twitter Message

Data Link Layer (cont'd.)

```
pi@wenpipi:~/etc $ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.3.2  netmask 255.255.255.0  broadcast 192.168.3.255
        inet6 fe80::c5ef:ae3e:e1b1:8699  prefixlen 64  scopeid 0x20<link>
        ether b8:27:eb:2c:75:d7  txqueuelen 1000  (Ethernet)
        RX packets 573  bytes 157912 (154.2 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 476  bytes 85825 (83.8 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 163  bytes 15399 (15.0 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 163  bytes 15399 (15.0 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

wlan0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.4.1  netmask 255.255.255.0  broadcast 192.168.4.255
        ether b8:27:eb:79:20:82  txqueuelen 1000  (Ethernet)
        RX packets 280  bytes 46577 (45.4 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 305  bytes 104924 (102.4 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

wlan1mon: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        unspec 00-C0-CA-99-0F-AB-00-00-00-00-00-00-00-00-00-00  txqueuelen 1000  (UNSPEC)
        RX packets 6972  bytes 1141584 (1.0 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Normal Interfaces on Raspberry House

```
pi@wenpipi:~/etc $ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.3.2  netmask 255.255.255.0  broadcast 192.168.3.255
        inet6 fe80::c5ef:ae3e:e1b1:8699  prefixlen 64  scopeid 0x20<link>
        ether b8:27:eb:2c:75:d7  txqueuelen 1000  (Ethernet)
        RX packets 546  bytes 155038 (151.4 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 459  bytes 82615 (80.6 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 157  bytes 14959 (14.6 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 157  bytes 14959 (14.6 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

wlan1mon: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        unspec 00-C0-CA-99-0F-AB-00-00-00-00-00-00-00-00-00-00  txqueuelen 1000  (UNSPEC)
        RX packets 6834  bytes 1109671 (1.0 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Raspberry House Interfaces Under Deauthentication Attack

Network Layer & Transport Layer

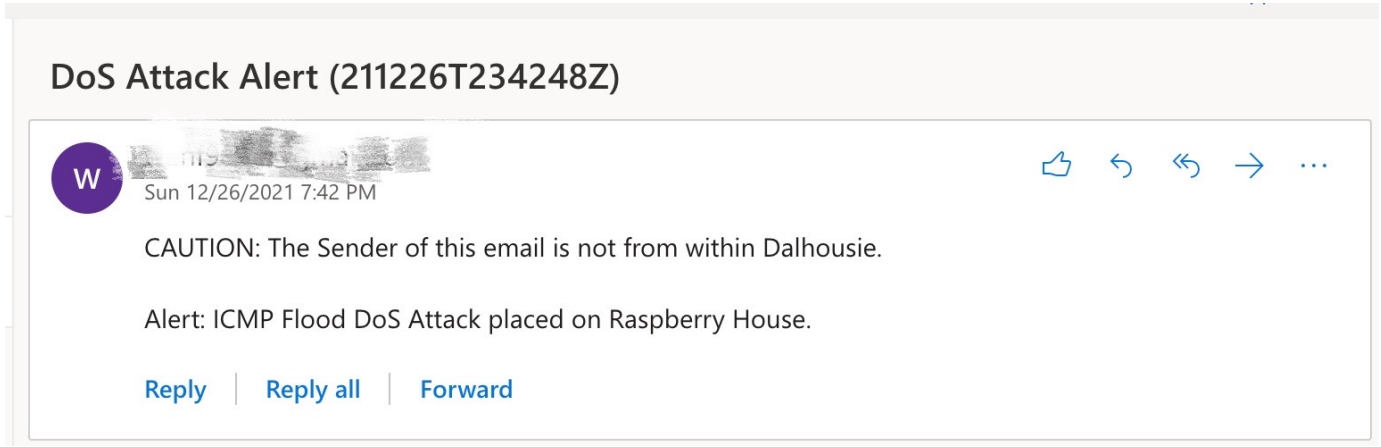
```
12/26-18:29:35.943822  [Drop] [**] [1:1000001:1] ICMP Flood Attack Rejected [**] [Classification: Generic ICMP event]
 [Priority: 3] {ICMP} 192.168.4.3 -> 192.168.4.1
12/26-18:29:35.943972  [Drop] [**] [1:1000001:1] ICMP Flood Attack Rejected [**] [Classification: Generic ICMP event]
 [Priority: 3] {ICMP} 192.168.4.3 -> 192.168.4.1
12/26-18:29:35.944047  [Drop] [**] [1:1000001:1] ICMP Flood Attack Rejected [**] [Classification: Generic ICMP event]
 [Priority: 3] {ICMP} 192.168.4.3 -> 192.168.4.1
12/26-18:29:35.944184  [Drop] [**] [1:1000001:1] ICMP Flood Attack Rejected [**] [Classification: Generic ICMP event]
 [Priority: 3] {ICMP} 192.168.4.3 -> 192.168.4.1
12/26-18:29:35.944251  [Drop] [**] [1:1000001:1] ICMP Flood Attack Rejected [**] [Classification: Generic ICMP event]
 [Priority: 3] {ICMP} 192.168.4.3 -> 192.168.4.1
```

ICMP Flood Attack Detection and Prevention Results

```
12/26-18:32:50.972400  [Drop] [**] [1:1000002:2] SYN Flood Attack Rejected [**] [Classification: Attempted Denial of
Service] [Priority: 2] {TCP} 192.168.4.3:20751 -> 192.168.4.1:80
12/26-18:32:50.972431  [Drop] [**] [1:1000002:2] SYN Flood Attack Rejected [**] [Classification: Attempted Denial of
Service] [Priority: 2] {TCP} 192.168.4.3:20752 -> 192.168.4.1:80
12/26-18:32:50.972725  [Drop] [**] [1:1000002:2] SYN Flood Attack Rejected [**] [Classification: Attempted Denial of
Service] [Priority: 2] {TCP} 192.168.4.3:20753 -> 192.168.4.1:80
12/26-18:32:50.972770  [Drop] [**] [1:1000002:2] SYN Flood Attack Rejected [**] [Classification: Attempted Denial of
Service] [Priority: 2] {TCP} 192.168.4.3:20754 -> 192.168.4.1:80
```

SYN Flood Attack Detection and Prevention Results

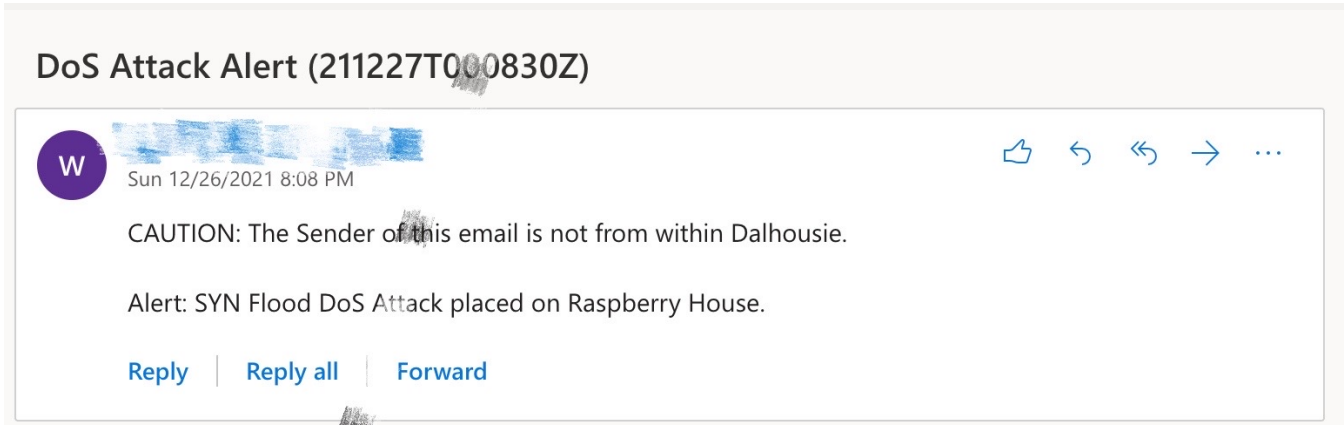
Network Layer & Transport Layer (cont'd.)



ICMP Flood Attack Email Alert

ICMP Flood Attack Twitter Message Alert

Network Layer & Transport Layer (cont'd.)

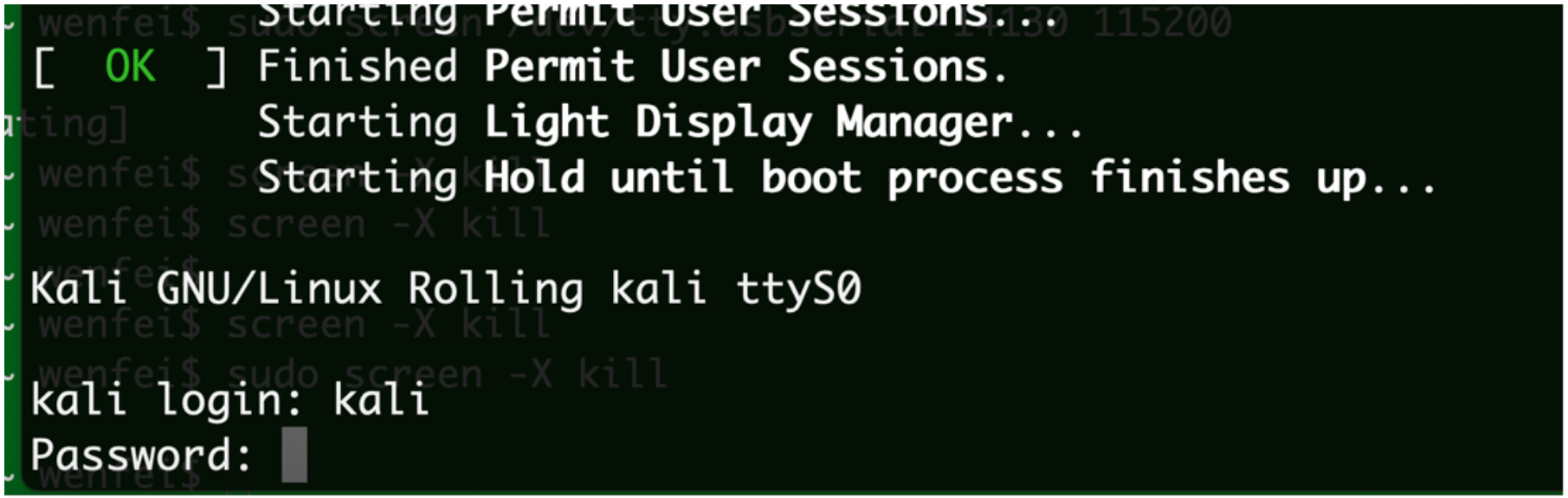


SYN Flood Attack Email Alert

SYN Flood Attack Twitter Message Alert

Enable USB to TTL Connection on Attacker PC

- In this research, we use a USB to Transistor-Transistor Logic (TTL) serial cable (UART connection) to operate Raspberry Pi 4 (malicious PC) on the MacBook Pro, which is **smaller and lighter than older serial cables**.
- As shown in Figure, we successfully connect to the Raspberry Pi 4 on our MacBook Pro terminal using UART connection.



UART Connection Result