

**Development of a Secure Raspberry Pi Based Home Surveillance System**

A dissertation submitted in partial fulfilment of the requirements of Glasgow Caledonian University for the degree of Master of Science in Big Data Technologies

Christopher Dillon

S1514278

**“Except where explicitly stated, all work in this report, including the appendices, is my own original work and has not been submitted elsewhere in fulfilment of the requirement of this or any other award”**

**Signed: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Date: \_\_\_\_\_\_\_\_**

# Abstract

# Table of Figures

[Figure 1: Smart Home - revenue forecast for the segment Security worldwide\* from 2017 to 2023 (in million U.S. dollars) (source: www.statista.com) 7](#_Toc11682043)

[Figure 2: The basic three layered architecture of IoT Devices (Burhan et al., 2018) 10](#_Toc11682044)

[Figure 3: Diagram detailing the SSH Protocol for Machine-To-Machine Communication (ssh.com, 2019) 14](#_Toc11682045)

# Table of Tables

[Table 1: Comparison table of common IoT Network Communication Protocols 12](#_Toc11682082)

[Table 2: Table highlighting features of most common IoT Application Communication Protocols 14](#_Toc11682083)

Table of Contents

[Abstract 2](#_Toc11682009)

[1.0 Introduction 5](#_Toc11682010)

[1.1 Background 5](#_Toc11682011)

[1.2 Problem Description 6](#_Toc11682012)

[1.3 Project objectives 6](#_Toc11682013)

[2.0 Literature Review 8](#_Toc11682014)

[2.1 IoT Architecture 8](#_Toc11682015)

[2.1.1 Perception Layer 8](#_Toc11682016)

[2.1.2 Network Layer 8](#_Toc11682017)

[2.1.3 Application Layer 8](#_Toc11682018)

[2.2 Network Communication Protocols 9](#_Toc11682019)

[2.2.1 Bluetooth 9](#_Toc11682020)

[2.2.2 Wi-Fi 10](#_Toc11682021)

[2.2.3 ZigBee 10](#_Toc11682022)

[2.3 Application Communication Protocols 11](#_Toc11682023)

[2.3.1 Message Queue Telemetry Transport 11](#_Toc11682024)

[2.3.2 Advance Message Queueing Protocol 11](#_Toc11682025)

[2.3.3 Constrained Application Protocol 12](#_Toc11682026)

[2.3.4 Redis (google) 12](#_Toc11682027)

[2.4 Security Protocols 13](#_Toc11682028)

[2.4.1 SSH Protocol 13](#_Toc11682029)

[2.4.2 WPA2 Protocol 13](#_Toc11682030)

[2.5 Security Threats 14](#_Toc11682031)

[2.5.1 DOS Attack 14](#_Toc11682032)

[2.5.2 Man in The Middle Attack 14](#_Toc11682033)

[2.5.3 Malicious Code Attack 14](#_Toc11682034)

[2.6 Cryptography Options 14](#_Toc11682035)

[2.6.1 RSA Algorithm 14](#_Toc11682036)

[2.6.2 AES Algorithm 14](#_Toc11682037)

[2.7 Proposed Software 14](#_Toc11682038)

[2.7.1 Face Recognition Technology 14](#_Toc11682039)

[2.7.2 Application Development 15](#_Toc11682040)

[2.8 Simulating Cyber Attacks 15](#_Toc11682041)

[5.0 References 17](#_Toc11682042)

# 1.0 Introduction

## 1.1 Background

Smart security is growing at an exponential rate with an expected revenue forecast of 28 million by the end of 2023 as shown in Figure 1. Although such technology poses many advantages over traditional non-internet based security, cybersecurity has been a rising concern with small IoT based devices. A recent report by HP highlighted that out of the ten home security systems analysed, 100% of them had security vulnerabilities such as not requiring strong passwords and only one offered two-factor authentication (HP, 2015).

This is concerning enough that the UK Government poses to introduce new legislation based on Internet of Things (IoT) security, requiring products to pass security requirements before being sold and introducing mandatory labelling schemes for each product, showing how secure they are (Ashford, 2019). This is further highlighted with recent media scares such as baby monitors being hacked with the hacker being able to speak through the baby monitor and threatening the parents (Wang, 2018).

With mainstream IoT devices being consistently hosted on the cloud, this offers individuals with malicious intent another method in taking down or accessing IoT devices. Recently, Google Cloud, one of the cloud vendor giants was taken down in a possible cyber-attack (Merriman, 2019), highlighting the potentially unreliability of Cloud hosted services. Organisations that depended on Googles Cloud, including security company Nest had their services taken down for over 4 hours. This is problematic in an area such as face-recognition based CCTV, meaning if data cannot be accessed from the Cloud, the whole system could break.

To limit the scope of potential security vulnerabilities, a locally stored Raspberry Pi based security system has been proposed. Presently, there are many people opting for Raspberry Pi smart systems at home due to the flexibility and cost of the Raspberry Pi. This, as with all Internet of Things devices, has many security flaws which are easily exploitable by those with malicious intent. Offering set guidelines and covering every layer of IoT architecture can present a model to base future systems on and prevent these cyber-attacks.

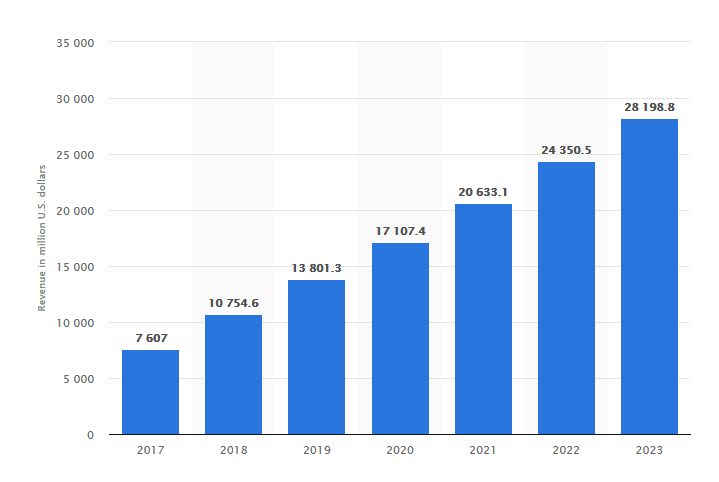


Figure 1: Smart Home - revenue forecast for the segment Security worldwide\* from 2017 to 2023 (in million U.S. dollars) (source: www.statista.com)

## 1.2 Problem Description

Modern IoT based security systems pose various security risks to the individual. As the majority of these services use cloud technology, the risk of security or information leaks become much more probable and concerning, with malicious access to the cloud service leading to potential massive data breaches and privacy concerns. The individual that uses such services has no control over how their data is collected, stored and sold, highlighting the concern of privacy.

The development of a home security system using a Raspberry Pi model is proposed to deal with the aforementioned challenges. By storing information locally, the amount of potential backdoors/security breaches is limited. This means security can be focused down to device level (Raspberry Pi), Micro-services and video security. Data can be captured with a non-intrusive mindset, limiting the amount of user-identifying data being produced.

With more individuals opting to create their own smart home devices using technology such as the Raspberry Pi, this projects aims to give set guidelines and establish best security practices to ensure the security and safety of those devices. To ensure that the model is successful, a testbed will be set up to simulate various attacks on multiple architectural levels.

## 1.3 Project objectives

This project aims to achieve the following objectives:

* Identify, from the literature available, state of the art guidelines for Internet of Things based device security. It is important to understand commonly used attacks or threats to these devices to prevent attacks against them. Furthermore, identifying common security weaknesses in IoT can help develop an understanding on why these attacks occur in the first place.
* There are multiple security flaws inherent with IoT devices present at multiple layers of the basic IoT architecture, giving individuals with malicious intent different angles of attack at the perception, network and application layers. Developing best practices to ensure security at all levels can bring forth a model to base similar IoT Projects on.
* Establish best practices for handling a user’s data. In the case of an unlikely breach, the data held on an individual must be non-intrusive. This is to minimize the damage caused by security attacks and ensure the safety of the individuals. Best data handling practices must be established and carried out throughout the project.

# 2.0 Literature Review

The proposed system is a home security system using the Raspberry Pi as the core component. The system would be able to detect individuals faces entering the premises and inform the family members if a face was not recognised via app notification. As with all IoT based devices, the Raspberry Pi poses multiple security risks at the Perception, Network and Application levels.

The proposed hardware for this system would be a Raspberry Pi 3 and one or multiple cameras placed near the vicinity of the home entrance. Although simplistic, a similar system was used for border surveillance and shown successful (Abdalla and Veeramanikandasamy, 2017).

## 2.1 IoT Architecture

IoT security faces three levels of architecture that can be attacked with malicious intent. As detailed in a recent 2018 paper, the most basic agreed upon architecture consists of three layers: Perception Layer, Network Layer and Application Layer (Burhan et al., 2018). For cloud based systems, an additional Processing layer is also considered (Aziz and Haq, 2018).

### 2.1.1 Perception Layer

This layer consists of the physical sensors which collect information and identify objects. Commonly used sensors include RFID, barcode & cameras. In this experiment the security camera is the main focus of this layer and can pose multiple potential security issues. As highlighted in the article mentioned previously, eavesdropping is definitely the most concerning security issue in this layer. Having unauthorised access to video footage is a serious breach of privacy. This is accomplished by taking advantage of insecure modes of transmission, granting access to a third party without authorisation.

### 2.1.2 Network Layer

The network layer is responsible for the transmission of data, acting as a sort of bridge between the perception and application layer. This involves carrying and transmitting information using through a wireless network which poses a set of security challenges. The layer acts as a sort of Central Nervous System for the whole network. In this scenario, the network layer consists of a standard wireless home network.

### 2.1.3 Application Layer

This layer is used to define all applications that use IoT technology. Common examples of this layer include smart homes, smart cities, smart health etc. This layer utilises the data gained from the previous layers and allows the user to use the application and enjoy its benefits. In this experiment the application layer would consist of the smart phone app used to operate the door and receive images/information from the security camera.

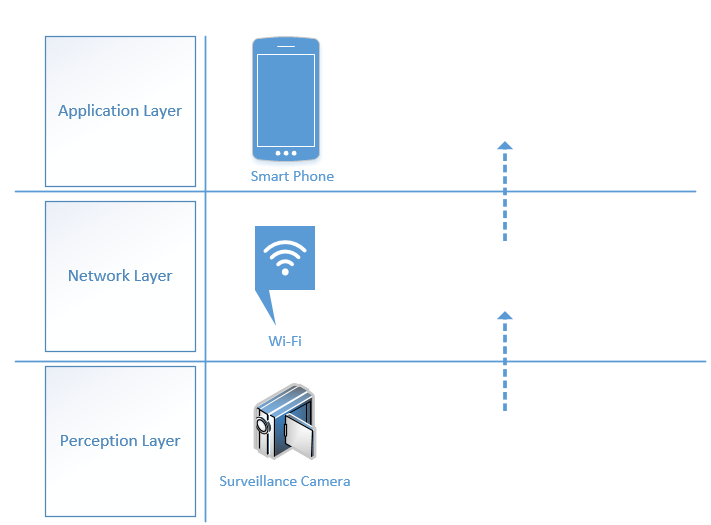


Figure : IoT 3 layered architecture in relation to a home surveillance system

## 2.2 Network Communication Protocols

Internet of Things devices differ to traditional computing devices in terms of network communication, utilising additional communication technologies such as ZigBee and Bluetooth, compared to traditional computers solely relying on Wi-Fi based communication.

The network communication protocol to be used will have the following attributes:

* An adequate range to reach the Raspberry Pi carrying out the surveillance
* Commonplace in home, to increase scalability
* High levels of security capability (encryption & authentication)
* Capable of fast data transmission (50mb/s+)

### 2.2.1 Bluetooth

Bluetooth is used to communicate between two devices within a short distance. It provides a layer of encryption by converting a message or data into cipher text before sending it to the receiver device. This message, cannot be understood by other devices except those which have the rights to see the message. The sender must always get permission rights from the receiving device before the message can be sent. This is done through the sender device requesting permission to send data to the received device, once agreed to, the devices can then communicate within the short distance. Bluetooth has a max signal rate of 1mb/s and an average range of 10 meters (Lee, Su and Shen, 2007).

### 2.2.2 Wi-Fi

Wi-Fi is a wireless communication network that transmits communication in the form of radio signalling. This is the most common type of communication used in modern homes which is problematic as it provides multiple security vulnerabilities. The main issue being, by default, there is no encryption mechanism. This leaves the network prone to MITM attacks, sniffing etc. It is well documented that Wi-Fi is one of the most commonly used network protocols in smart homes (Alam, Reaz and Ali, 2012). Wi-Fi has the added benefit of being the fastest of all common communication technologies with an average speed of 54mb/s (Lee, Su and Shen, 2007) with the ability to scale up speed if necessary. The average range of a Wi-Fi connection is around 100 meters.

### 2.2.3 ZigBee

ZigBee is a communication protocol, which utilises small, low-power and cost radios. It is frequently utilised in the smart homes in conjunction with hubs such as the Amazon Echo (Zigbee Alliance, 2019). The technology is very limited in terms of signal rate with a rate of 250Kb/s (Lee, Su and Shen, 2007), making it unsuitable for video transmission.

2.2.4 Heart

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Requirement | Bluetooth | Wi-Fi | ZigBee | Heart |
| High range |  |  |  |  |
| Commonplace in home |  |  |  |  |
| Security |  |  |  |  |
| Fast Data Transmission |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

Table 1: Comparison table of common IoT Network Communication Protocols

## 2.3 Application Communication Protocols

### 2.3.1 Message Queue Telemetry Transport

MQTT is the most commonly used application layer protocol, being light weight and using a publish-subscribe model. The protocol is mainly used where a small code footprint is required i.e. sensor data and where bandwidth is limited. With the maximum amount of data being transferred 256MB (Rastovich, 2015); speed is the priority with this protocol. It is based on the client/server architecture, the server being responsible for handling the clients request of receiving or sending data between each other. This means when a device sends data to the broker (client) it is ‘publishing’ data for any ‘subscribers’ listening to that particular topic.

It supports three levels of Quality of Service (QoS), with each having different volumes of packets exchanged. QoS Level 0 sends a message once and does not check if the message arrived to its given destination. Level 1 improves upon this by sending the message at least once and checks the delivery status using PUBACK, a status check message. If the PUBACK is lost, this can result in the same message being sent twice, due to no confirmation of the original message. Finally, Level 2 sends the message exactly once by utilising a 4-way handshake between the client and broker. This has the benefit of there being no possibility of the message being lost, but can cause additional delays. In a correlation analysis measuring lost and delay of MQTT QoS levels, it was found that for Levels 0 and 1, the average packet loss was around 0.90% to 1.40% whereas packet loss only occurred at maximum, 0.18% with QoS level 2 (Lee et al., 2013). Although QoS level 2 did have the highest end-to-end delay, this was at maximum one second, suggesting QoS Level 2 would be most suitable for the proposed system. Sharing the same attributes as MQTT, SMQTT introduces an encryption/decryption protocol to enhance security (Singh et al., 2015).

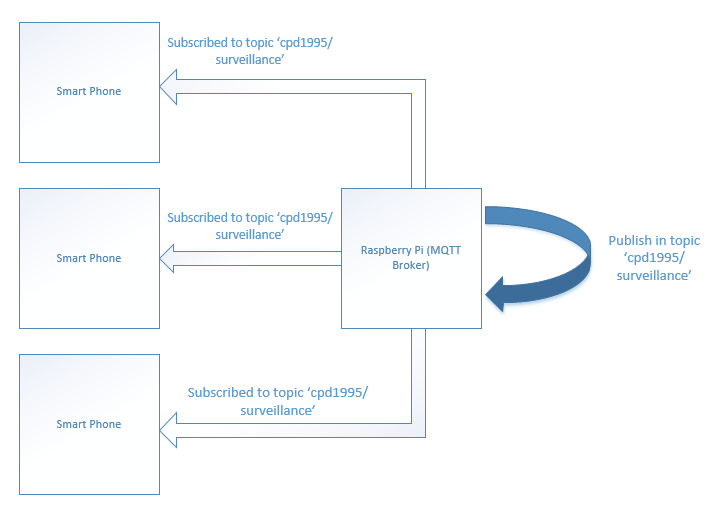


Figure : MQTT Architecture using a Raspberry Pi as both a Publisher and Broker

### 2.3.2 Advanced Message Queueing Protocol

AMQP is an open standard application layer protocol for middleware messaging. It has additional features such as message orientation, switching, reliability and queueing. Both request/response and publisher subscriber models are available with this protocol. Communication via this protocol is done through either the publisher or consumer creating an “exchange” with a given name, that name can then be broadcasted for both publishers and consumers to discover each other. Along with this exchange, the consumer creates a “queue”, attaching it to the exchange. Messages that are received by the exchange are matched to the queue via a binding process. AMQP offers the most flexibility when it comes to message exchange, offering multiple methods such as message by topic (similar to MQTT), directly, based on headers or in a fan-out form. In terms of Quality of Service, AMQP offers two types of delivery of messages: Unsettle Format and Settle Format, being similar to MQTT’s Level 0 and 1.

### 2.3.3 Constrained Application Protocol

CoAP is a stateless client/server application protocol based on the REST framework. This is based on a request/response model using the request types ‘GET’, ‘POST’, ‘PUT’ and ‘DELETE’, being similar to the standard HTTP protocol. CoAP uses Universal Resource Identifier (URI) instead of topics seen in protocols such as MQTT. This works using a similar publish/subscribe method. The publisher sends new data to the URI, and all the subscribers are notified about the new value indicated by the URI, similar to MQTT. CoAP utilises UDP as a transport protocol and DTLS for security (Ludovici, Moreno and Calveras, 2013). Using UDP instead of TCP does create less reliability as there is no guaranteed delivery of a packet. To combat this; there is the option of “confirmable” and “non-confirmable” messages to provide a Quality of Service architecture. Confirmable messages work by being acknowledged by the receiver via an ACK packet. Non-confirmable messages has no such system in place.

### 2.3.4 Redis (google)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Feature | MQTT | CoAP | AMQP | XMPP |
| Header Size | 2 Byte | 4 Byte | 8 Byte |  |
| Architecture | Client/Broker | Client/Server or Client/Broker | Client/Server or Client/Broker |  |
| Abstraction | Publish/Subscribe | Request/Response or Publish/Subscribe | Request/Response or Publish/Subscribe |  |
| Message Size | 256mb Max | Small & Undefined | Negotiable & Undefined |  |
| QoS / Reliability | QoS 0 – At most once  QoS 1 – At least once,  QoS 2 – Exactly once | Confirmable or Non-Confirmable Message | Settle Format or Unsettle Format |  |
| Transport Protocol | TCP | UDP | TCP, SCTP |  |
| Default Port | 1883/8883 (TSL/SSL) | 5683 (UDP) | 5671/5672 (TLS/SSL) |  |

Table 2: Table highlighting features of most common IoT Application Communication Protocols

As shown in table 2, MQTT seems to offer the most in terms of reliability of message delivery. Utilising QoS Level 2 has the guarantee the message is not lost, whereas the other protocols offer no such service, with CoAP and AMQP only offering the equivalent to a Level 1 QoS service. Transport protocol is also a consideration when selecting a communication protocol, as TCP is much more reliable than its UDP counterpart. It was also shown when it comes to sending small bytes of data similar to the payloads of most IoT devices, MQTT appears to perform better with overall less latency compared to its AMQP counterpart (Sreeraj and Kumar, 2018). In contrast, a recent comparison study found that out of all the common IoT communication protocols, MQTT was scored on the lowest in terms of security (Naik, 2017). This suggests that that SMQTT is the preferred choice for a home security system, as this protocol would meet all the given requirements.

## 2.4 Security Protocols

### 2.4.1 SSH Protocol

Secure Shell protocol or SSH as its commonly referred to is a method to remotely access one computing from another securely (Ylonen, 1996). Based on client-server model, a connection is established by the SSH client by connecting to the SSH server. The client then uses public key cryptography to verify the identity of the SSH server. For data exchange, the protocol uses symmetric encryption and hashing algorithms, ensuring the privacy and integrity of the data sent between the two devices.

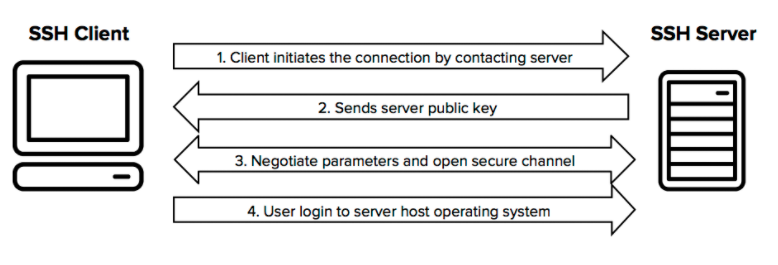


Figure 4: Diagram detailing the SSH Protocol for Machine-To-Machine Communication (ssh.com, 2019)

### 2.4.2 WPA2 Protocol

WPA2 is the most widely spread security protocol for Wi-Fi based communication. WPA2 accomplishes Authentication and Key Agreement via two different modes. For most personal networks, this involves using a Pre-Shared Key (PSK). AES is typically used for encryption, which is a symmetric-key algorithm that uses the one key both encryption and decryption. The initial cryptographic keys for both authentication and data encryption is initiated by both the client and access point knowing the shared passphrase, allowing access to the network. This is then used with additional variables such as Service Set Identifier (SSID) and SSID length to produce a Pairwise Master Key (PMK). A recent paper included that this protocol is especially vulnerable to dictionary based attacks and key reinstallation attacks (Abo-Soliman and Azer, 2017).

## 2.5 Security Threats

This section highlights potential security threats relating to IoT devices. The aim of this section is to uncover the most frequent security vulnerabilities present within the IoT architecture.

### 2.5.1 DOS Attack

Denial of Service attacks are used to prevent access to devices or network resources, essentially taking the network offline. It is accomplished by flooding a network with packets and redundant requests to the point of the network being unable to be accessed by authentic users (Prabhakar, 2017).

### 2.5.2 Man in The Middle Attack

The man-in-the-middle has the ability to capture all messages between a server and IoT device via network spoofing. Using this spoofing strategy, it can assume identification of both the IoT device and server and trick both devices into believing they are still communicating with each other (Conti, Dragoni and Lesyk, 2016).

### 2.5.3 Malicious Code Attack

This is code in any part of the software intended to cause undesirable effects and potentially causing the system to malfunction. This cannot be detected by security tools such as anti-viruses as it not detected as a virus or malware.

### 2.5.4 Physical Tampering

Physical manipulation of a device could occur. This is when the device is modified to act in unexpected ways by someone with malicious intent. It is safe to assume that if this happens the intruder is already in the individual’s home and there is not much that can be done. A way to protect the user’s data if this occurred would be full encryption of the storage and 2-factor authentication to prevent users from accessing the micro SD that is present in the device.

## 2.6 Cryptography Options

### 2.6.1 RSA Algorithm

RSA works on a public-private key basis which is a form of asymmetric encryption. The public key is available to everyone, i.e. placed on the server (Raspberry Pi) whereas the private key is used to ‘unlock’ that specific public key. This gives an added layer of security as the data can only be decrypted using the individuals private key. The private key is usually stored on a client device e.g. computer or mobile phone (Goshwe, 2013). This form of encryption was used successfully in the creation of a payment terminal using a Raspberry Pi, TSL and AES based encryption, highlighting its efficiency (Kakar, 2016).

### 2.6.2 AES Algorithm

Advance Encryption Standard is symmetric form of encryption, supporting key lengths of 127, 192 and 256 bits. In a comparison study against 4 different algorithms, it was shown that AES has the highest avalanche effect, which is recorded as how much change there is in an output based on the change of an input. Scoring high in the avalanche effect reflects a high level of cryptographic strength. This suggests when confidentially and integrity are the highest priority, AES should be used (Patil et al., 2016).

## 2.7 Password Handling Best Practices

With there being various different methods of retrieving an individual’s password such as keylogging, insecure password storage and brute-force attacks, this section aims to establish the correct methodology in dealing with those threats. The UK government has provided a manual for password handling to establish best practices (ukgov, 2015):

1. Change all default passwords. When it comes to small devices like the Raspberry Pi the default username is always ‘pi’ with the password being ‘raspberry’. These should both be changed, as an unauthorised user using a remote connection such as SSH will have to guess the username as well.
2. Prioritise administrator and remote user accounts. Good practices involve not using administrator accounts for day-to-day activities with a separate account being established for this purpose. If remote access to the Raspberry Pi were granted, limits should be set on the account along with two-factor authentication to verify who is accessing the device remotely.
3. Implement a form of account lockout. As mentioned previously, brute force attacks are a large security threat to IoT devices. The UK government recommends a maximum of 10 login attempts before locking out the account.
4. Do not store passwords as plan text. If a password was stored within a database in plain text, that database being compromised leaves the users vulnerable to their details being stolen. It is recommended that passwords be stored in a form of hashing and salt combination.

<https://www.pyimagesearch.com/2018/06/25/raspberry-pi-face-recognition/>

https://developer.ibm.com/recipes/tutorials/sending-and-receiving-pictures-from-a-raspberry-pi-via-mqtt/

# 3.0 Technology Review

## 3.1 Face Recognition Technology

OpenCV is an open source that offers a range of feature detecting and feature matching algorithms. It is shown that the majority of the algorithms used, detect thousands of features with seconds (Noble, 2016), making it suitable for a face detection system. To highlight the accuracy of this technology, a recent paper showed success in detecting eye fatigue in drivers using OpenCV (Manoharan and Chandrakala, 2015).

## 3.2 Application Development

Due to time constraints on this project, hybrid app development is proposed.

To allow a family member receive an image and notification of who is at the door, a smart phone application is proposed. A recent 2018 paper showed a similar system using image notifications from a Raspberry Pi Camera sent via email when a potential intruder was located at their door (Pawar and Umale, 2018).

## 3.3 Remote Access Security

Enabling ports such as SSH for remote access comes with the risk of unauthorised access (ref).

### 3.3.1 Fail2Ban

Fail2Ban is a Python based open source software that interacts with the Raspberry Pi’s operating system’s firewall (Fail2ban.org, n.d.). Its main purpose is to ban IPs that show malicious intent. This includes too many password attempts, seeking for exploits, etc.

## 3.4 MQTT Broker

## 3.3 Simulating Cyber Attacks

Currently, the standard operating system used in cyber security and digital forensics is Kali Linux, derived from the Linux distribution Debian (ref). The operating system contains over 300 programs and tools used in ethnical hacking, making it the gold standard for carrying out simulated attacks. In the 2016 paper (Liang et al., 2016), three different types of DoS attacks were carried out, resulting in all three being successful against a small IoT device.

<https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8589403>

<https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7354752> smart home security guidelines

<https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=7902207>

<https://blog.ipswitch.com/internet-of-things-101-iot-device-authentication-explained>

<https://www.sbprojects.net/projects/raspberrypi/ga.php> - 2 factor auth

<https://opensource.com/article/17/3/iot-security-raspberry-pi>

dictionary attacks

<https://www.raspberrypi.org/forums/viewtopic.php?t=196010>

<https://www.zymbit.com/securing-raspberry-pi/> - data and file encryption

<https://www.raspberrypi.org/forums/viewtopic.php?t=196010> app dev

<https://mntolia.com/mqtt-python-with-paho-mqtt-client/#0_Installing_Paho-MQTT> – mqtt tut

# 5.0 References

1. Merriman, C. (2019). *Google Cloud burst downs G Suite, Nest and Snapchat during 4-hour outage | TheINQUIRER*. [online] http://www.theinquirer.net. Available at: https://www.theinquirer.net/inquirer/news/3076657/google-cloud-outage [Accessed 7 Jun. 2019].
2. Wang, A. (2018). ‘I’m in your baby’s room’: A hacker took over a baby monitor and broadcast threats, parents say. [online] Washington Post. Available at: https://www.washingtonpost.com/technology/2018/12/20/nest-cam-baby-monitor-hacked-kidnap-threat-came-device-parents-say/?noredirect=on&utm\_term=.04321c476652 [Accessed 7 Jun. 2019].
3. Ashford, W. (2019). *UK gears up for new laws on IoT security*. [online] ComputerWeekly.com. Available at: https://www.computerweekly.com/news/252462505/UK-gears-up-for-new-laws-on-IoT-security [Accessed 7 Jun. 2019].
4. Aziz, T. and Haq, E. (2018). Security Challenges Facing IoT Layers and its Protective Measures. *International Journal of Computer Applications*, [online] 179(27), pp.31-35. Available at: https://www.researchgate.net/publication/323892938\_Security\_Challenges\_Facing\_IoT\_Layers\_and\_its\_Protective\_Measures [Accessed 9 Jun. 2019].
5. Prabhakar, S. (2017). NETWORK SECURITY IN DIGITALIZATION: ATTACKS AND DEFENCE. *INTERNATIONAL JOURNAL OF RESEARCH IN COMPUTER APPLICATIONS AND ROBOTICS*, [online] Vol. 5 Issue 5(ISSN 2320-7345). Available at: https://www.ijrcar.com/Volume\_5\_Issue\_5/v5i512.pdf [Accessed 9 Jun. 2019].
6. Conti, M., Dragoni, N. and Lesyk, V. (2016). A Survey of Man In The Middle Attacks. *IEEE Communications Surveys & Tutorials*, [online] 18(3), pp.2027-2051. Available at: https://ieeexplore.ieee.org/document/7442758 [Accessed 9 Jun. 2019].
7. Burhan, M., Rehman, R., Khan, B. and Kim, B. (2018). IoT Elements, Layered Architectures and Security Issues: A Comprehensive Survey. *Sensors*, [online] 18(9), p.2796. Available at: https://www.researchgate.net/publication/327272757\_IoT\_Elements\_Layered\_Architectures\_and\_Security\_Issues\_A\_Comprehensive\_Survey [Accessed 9 Jun. 2019].
8. Noble, F. (2016). Comparison of OpenCV's feature detectors and feature matchers. *2016 23rd International Conference on Mechatronics and Machine Vision in Practice (M2VIP)*. [online] Available at: https://ieeexplore.ieee.org/document/7827292 [Accessed 26 May 2019].
9. Diachok, R., Dunets, R. and Klym, H. (2018). System of detection and scanning bar codes from Raspberry Pi web camera. *2018 IEEE 9th International Conference on Dependable Systems, Services and Technologies (DESSERT)*. [online] Available at: https://ieeexplore.ieee.org/document/8409124 [Accessed 26 May 2019].
10. Abdalla, G. and Veeramanikandasamy, T. (2017). Implementation of spy robot for a surveillance system using Internet protocol of Raspberry Pi. *2017 2nd IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT)*. [online] Available at: https://ieeexplore.ieee.org/document/8256563 [Accessed 26 May 2019].
11. Alam, M., Reaz, M. and Ali, M. (2012). A Review of Smart Homes—Past, Present, and Future. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, [online] 42(6), pp.1190-1203. Available at: https://ieeexplore.ieee.org/document/6177682 [Accessed 9 Jun. 2019].
12. Pawar, A. and Umale, V. (2018). Internet of Things Based Home Security Using Raspberry Pi. *2018 Fourth International Conference on Computing Communication Control and Automation (ICCUBEA)*. [online] Available at: https://ieeexplore.ieee.org/document/8697345 [Accessed 9 Jun. 2019].
13. Manoharan, R. and Chandrakala, S. (2015). Android OpenCV based effective driver fatigue and distraction monitoring system. *2015 International Conference on Computing and Communications Technologies (ICCCT)*. [online] Available at: https://ieeexplore.ieee.org/document/7292757 [Accessed 10 Jun. 2019].
14. Rastovich, R. (2015). *Why the MQTT Protocol is Ideal for the Internet of Things – ThingLogix, Inc*. [online] Thinglogix.com. Available at: https://www.thinglogix.com/why-the-mqtt-protocol-is-ideal/ [Accessed 10 Jun. 2019].
15. Liang, L., Zheng, K., Sheng, Q. and Huang, X. (2016). A Denial of Service Attack Method for an IoT System. *2016 8th International Conference on Information Technology in Medicine and Education (ITME)*. [online] Available at: https://ieeexplore.ieee.org/document/7976501 [Accessed 12 Jun. 2019].
16. Ssh.com. (2019). *SSH Protocol – Secure Remote Login and File Transfer | SSH.COM*. [online] Available at: https://www.ssh.com/ssh/protocol/#sec-How-to-Cite-SSH [Accessed 13 Jun. 2019].
17. Ylonen, T. (1996). SSH - Secure Login Connections over the Internet. *Proceedings of the 6th USENIX Security Symposium*, pp.37-42.
18. Abo-Soliman, M. and Azer, M. (2017). A study in WPA2 enterprise recent attacks. *2017 13th International Computer Engineering Conference (ICENCO)*. [online] Available at: https://ieeexplore.ieee.org/document/8289808 [Accessed 13 Jun. 2019].
19. Sreeraj, S. and Kumar, G. (2018). Performance of IoT protocols under constrained network, a Use Case based approach. *2018 International Conference on Communication, Computing and Internet of Things (IC3IoT)*. [online] Available at: https://ieeexplore.ieee.org/document/8668105 [Accessed 13 Jun. 2019].
20. Singh, M., Rajan, M., Shivraj, V. and Balamuralidhar, P. (2015). Secure MQTT for Internet of Things (IoT). *2015 Fifth International Conference on Communication Systems and Network Technologies*. [online] Available at: https://ieeexplore.ieee.org/abstract/document/7280018 [Accessed 13 Jun. 2019].
21. How safe are home security systems?. (2015). [ebook] HP. Available at: http://go.saas.hpe.com/l/28912/2015-07-21/32bhy5/28912/69170/IoT\_Home\_Security\_Systems.pdf [Accessed 13 Jun. 2019].
22. Goshwe, N. (2013). Data Encryption and Decryption Using RSA Algorithmin a Network Environment. *IJCSNS International Journal of Computer Science and Network Security*, [online] VOL. 13(No. 7). Available at: http://paper.ijcsns.org/07\_book/201307/20130702.pdf [Accessed 16 Jun. 2019].
23. Kakar, V. (2016). Secure communication and authentication using Raspberry Pi. *2016 5th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO)*. [online] Available at: https://ieeexplore.ieee.org/document/7785011 [Accessed 16 Jun. 2019].
24. Patil, P., Narayankar, P., Narayan D.G. and Meena S.M. (2016). A Comprehensive Evaluation of Cryptographic Algorithms: DES, 3DES, AES, RSA and Blowfish. *Procedia Computer Science*, [online] 78, pp.617-624. Available at: https://www.sciencedirect.com/science/article/pii/S1877050916001101?via%3Dihub [Accessed 16 Jun. 2019].
25. Lee, S., Kim, H., Hong, D. and Ju, H. (2013). Correlation analysis of MQTT loss and delay according to QoS level. *The International Conference on Information Networking 2013 (ICOIN)*. [online] Available at: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7428475 [Accessed 17 Jun. 2019].
26. Ludovici, A., Moreno, P. and Calveras, A. (2013). TinyCoAP: A Novel Constrained Application Protocol (CoAP) Implementation for Embedding RESTful Web Services in Wireless Sensor Networks Based on TinyOS. *Journal of Sensor and Actuator Networks*, [online] 2(2), pp.288-315. Available at: https://www.researchgate.net/publication/276039057\_TinyCoAP\_A\_Novel\_Constrained\_Application\_Protocol\_CoAP\_Implementation\_for\_Embedding\_RESTful\_Web\_Services\_in\_Wireless\_Sensor\_Networks\_Based\_on\_TinyOS [Accessed 17 Jun. 2019].
27. Naik, N. (2017). Choice of effective messaging protocols for IoT systems: MQTT, CoAP, AMQP and HTTP. *2017 IEEE International Systems Engineering Symposium (ISSE)*. [online] Available at: https://ieeexplore.ieee.org/document/8088251 [Accessed 17 Jun. 2019].
28. UK Government advice on Password Guidance, Simplifying your approach. (2015). [ebook] CESG and CPNI. Available at: https://assets.publishing.service.gov.uk/government/uploads/system/uploads/attachment\_data/file/458857/Password\_guidance\_-\_simplifying\_your\_approach.pdf [Accessed 18 Jun. 2019].
29. Fail2ban.org. (n.d.). *Fail2ban*. [online] Available at: https://www.fail2ban.org/wiki/index.php/Main\_Page [Accessed 18 Jun. 2019].