Architectures for Securing Low-Cost IoT

A thesis submitted to the Centre for Intelligent Dynamic Communications,
Department of Electronic and Electrical Engineering,
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy

Ross McPherson
Mobile Communcations & Cybersecurity
University of Strathclyde
United Kingdom

17th May 2020

This thesis is the result of the author's original research. It has been composed by the author and has not been previously submitted for examination which has led to the award of a degree.

The copyright of this thesis belongs to the author under the terms of the United Kingdom Copyright Acts as qualified by University of Strathclyde Regulation 3.50. Due acknowledgement must always be made of the use of any material contained in, or derived from, this thesis.

Signed: R. McPherson

Acknowledgment

I would like to thank my supervisor Dr James Irvine for his continued support, guidance and overall insight which inspired me from the start and enabled me to complete my PhD. I would also like to thank my friends, my family and especially Tia. I could not have done this without you.

Abstract

This PhD study proposes various solutions to enable low-cost Internet of Things (IoT) devices to employ end-to-end encryption, with consideration to the financial and practical requirements of such devices. Manufacturers of IoT devices often sacrifice security in favour of features, user-friendliness, time to market or cost, in order to stay ahead of their competitors. However, numerous recent hacks on IoT devices have shown this is unwise. Extremely low-cost, microcontroller based devices, in particular struggle to create sufficient random data required for state of the art security features. This is in stark contrast to common internet connected devices such as smartphones, and personal computers, which with their high performance processors can generate sufficient randomness to implement state of the art security features. Through taking advantage of existing devices this PhD study enables these extremely low-cost devices implement state of the art security features. One such existing device is a smartphone which are abundant and have multiple sources of randomness for strong key generation. This abundance and power enables two discrete architectures to be proposed which offload key generation and transfer to the user's smartphone, removing the requirement for constrained IoT devices to participate in public key infrastructure.

The first proposed architecture takes advantage of the ubiquity of Wi-Fi in consumer devices, and employs it as a key transfer mechanism. The design only requires a $1 general-purpose microcontroller and a Wi-Fi module. The performance of such an architecture was modelled, and a threat analysis conducted. The design allows all communication to and from the device to be encrypted, without any additional manufacturing cost or sacrificing user experience. Since the architecture does not require any additional hardware, it can be retroactively applied and deployed to existing devices through a firmware update. This architecture has been integrated into a smart light switch product.

The second architecture targeted at increasingly popular low-power wide-area networks (LPWAN) technologies, such as LoRaWAN. Future utility networks will require much greater levels of machine-to-machine interaction to enable smart grid applications, and ultimately increase the efficiency of the network. In response a low-cost LPWAN module was developed to be retrofitted to existing monitoring device, an Fault Passage Indicators (FPI), creating an automatic remote detection and location system for medium voltage faults, reducing the time to find faults from hours to minutes. This prototype device was then tested on a physical 11kV overhead line network, where the device demonstrated it was robust and suitable for mass deployment on a live network. The finalised prototype module costs roughly 5% of a single FPI unit, making the solution cost effective. The system additionally features the ability to remotely reconfigure the FPI, making the installation and future maintenance more convenient.

Two high costs in a full-scale LPWAN deployment are the cost of deployment and the potential cost of re-keying the compromised network. Therefore, this thesis also explores methods of decentralising sensor device deployment using commonly available hardware. It details a procedure which uses a smartphone's camera flash to transfer the necessary credentials to such low-cost sensor devices. As with the domestic Wi-Fi-based solution, smartphones were chosen as a transfer mechanism since they are both abundant and suitably powerful to generate random keys. Using smartphone's LED flash removes the need for a wired connection, a laptop, and programming software, allowing devices to be provisioned out in the field without the need for specialised tools or knowledge. An implementation with a Long Range Wide Area Network (LoRaWAN) device was created as this is a prime candidate given its requirement for cryptographic keys and lack of ability to partake in public key infrastructure. The solution allows non-specialists to securely program and re-key devices without specialised tools. A security audit is then conducted on the system as a whole.

The final part of the thesis considers the dependency IoT devices currently have on centralised architecture. This reliance has made numerous devices non-functional when their infrastructure ceased to operate. Recent research into the security of internet-connected consumer equipment has showed the extent to which many of these devices remain highly vulnerable to remote compromise. The Mirai botnet highlighted this risk of having large numbers of vulnerable devices, all connected to high-speed internet connections, capable of performing Distributed Denial of Service (DDoS) attacks. A distributed hash table (DHT) based architecture is proposed, where an intermediary device provides access control and secure remote access of IoT devices located in the home. This design removes the need for a centrally operated server and improves longevity. Software running the intermediary separates vulnerable IoT devices from the rest of the home network, and coordinates with the DHT to publish and listen for messages. An implementation of this system was created, and demonstrated that the network overhead would be minimal, and the system would be able to integrate with existing IoT ecosystems.

Contents

Abbreviations

ABP activation by personalisation. 44

AC Alternating Current. 104

ADSL Asymmetric Digital Subscriber Line. 118

AES Advanced Encryption Standard. 14–19, 29, 30, 33, 36, 44, 48, 56, 71, 72, 76, 79, 80, 88, 131, 135, 168, 169

AES-128 AES 128 Bit Keys. 15, 136

AES-192 AES 192 Bit Keys. 15

AES-256 AES 256 Bit Keys. 14, 15, 136, 169

AES-CTR AES Counter Mode. 44, 80–82, 89

AES-GCM AES Galois/Counter Mode. 80, 81

API Application Programming Interface. 40, 42, 75, 90–92, 116, 134, 138, 139, 149, 150, 152, 155, 160, 161, 163

AppEUI Application Identifier. 44, 129, 135

AppKey Application Key. 44, 129, 135

AppSKey Application Session Key. 44

ASIC Application-Specific Integrated Circuit. 33

CA Certificate Authorities. 27

CBC Cipher Block Chaining. 17–19, 29, 80

CFB Cipher feedback. 80

CMAC Cipher-based Message Authentication Code. 36, 44, 80–82, 89

CMOS Complementary Metal Oxide Semiconductor. 33

COAP Constrained Application Protocol. 155

CPU Central Processing Unit. 135

CRC Cyclic Redundancy Check. 134, 135

DCMS Department for Digital, Culture, Media & Sport. 55

DDoS Distributed Denial of Service. 143, 144, 160

DES Data Encryption Standard. 13, 14, 29, 48

DevEUI Device Unique Identifier. 44, 135

DHCP Dynamic Host Configuration Protocol. 149, 150

DHT Distributed Hash Table. 3, 144–147, 150–163, 169

DNO Distribution Network Operator. 104, 115, 122, 125

DNS Domain Name Service. 146, 150

DTLS Datagram Transport Layer Security. 163, 169

ECB Electronic Codebook. 16, 17, 80

ECC Elliptic-curve cryptography. 25

EEA European Economic Area. 57, 65

ENA Energy Networks Association. 100

EU European Union. 53, 64–66

EVs Electric Vehicles. 100

FPI Fault Passage Indicator. 101, 104, 105, 109, 112, 113, 115, 118–122, 127

GCM Galois/Counter Mode. 19, 20

GDPR General Data Protection Regulation. 64, 65

GPRS General Packet Radio Services. 41, 46, 105, 107, 113

GSM Global System for Mobile Communications. 45, 46

HGU Home Gateway Unit. 151–155, 157–163

HMAC Hash-based Message Authentication Code. 35, 36, 80, 81

HSM Hardware Security Module. 15, 32

HTTP Hypertext Transfer Protocol. 90, 153, 155–157, 159, 161, 162

HV High Voltage. 114, 119, 120

ICE Interactive Connectivity Establishment. 76

ICO Information Commissioner's Office. 65

IETF Internet Engineering Task Force. 28

IFTTT If This Then That. 76, 88, 92, 159, 163

IIoT Industrial Internet of Things. 3, 93–96, 98, 99, 101, 126, 128, 165

IoT Internet of Things. 1–6, 8, 9, 22, 24, 29, 31–33, 37–39, 49–51, 53–65, 68–73, 75–81, 84, 87, 90, 92, 93, 106, 125, 140–156, 158–161, 163, 165–168

IP Internet Protocol. 41, 42, 70, 75, 88, 89, 122, 145, 148, 150, 155, 157, 158, 162

ISM Industrial, Scientific and Medical. 43

IT Information Technology. 103

IV Initialisation Vector. 18, 19, 80

JSON JavaScript Object Notation. 123, 155

JWT JSON Web Token. 161

LDR Light-Dependent Resistor. 130

LoRaWAN Long Range Wide Area Network. 32, 40–45, 47, 106, 108–116, 118–129, 134–140, 166

LPWAN Low Powered Wide Area Network. 4, 39–41, 45, 47, 95, 105–107, 110, 127, 166, 169

LTE Long-Term Evolution. 40, 41, 46, 47, 110, 111

LTE-M LTE Machine Type Communication. 46, 47

M2M Machine-to-Machine. 106

MAC Media Access Control. 31, 155, 157

MAC Message Authentication Code. 29, 35, 36, 74, 108

mDNS Multicast DNS. 75, 88, 89, 149, 150, 153, 155, 161

MIC Message Integrity Check. 44

MQTT Message Queuing Telemetry Transport. 42, 116, 155

NAT Network Address Translation. 76, 88, 141, 147–149

NB-IoT Narrowband Internet-of-Things. 40, 45–47, 106, 127

NFC Near Field Communication. 128

NIS Network and Information Systems. 66

NIST National Institute of Standards and Technology. 14, 16, 25, 26, 34

NSA National Security Agency. 25

NwkSKey Network Session Key. 44, 129

OFB Output Feedback. 18, 19, 80

Ofcom Office of Communications. 47, 110, 111

Ofgem Office of Gas and Electricity Markets. 100

OOK On-Off Keying. 129, 132

OSI Open Systems Interconnection. 155

OT Operational Technology. 103

OTAA over-the-air-activation. 44

PCB Printed Circuit Board. 86, 106, 122, 134, 137

PKI Public Key Infrastucture. 126, 127

PKP Public Key Partnership. 23

PNDC Power Networks Demonstration Centre. 118, 119, 122

POODLE Padding Oracle On Downgraded Legacy Encryption. 29

PPM Pulse Position Modulation. 129, 130, 132

PUF Physically Unclonable Function. 33, 34

PWM Pulse-width Modulation. 129, 130, 132, 137

REST Representational State Transfer. 42, 90, 116, 155

RSA Rivest, Shamir and Adleman. 23–26, 28

RSSI Received Signal Strength Indication. 124

SDK Software Development Kit. 133

SHA Secure Hashing Algorithms. 34

SIM Subscriber Identification Module. 48

SSH Secure Shell. 25

SSID Service Set IDentifier. 72

SSL Secure Socket Layer. 28, 29

STUN Session Traversal Utilities for NAT. 76

TCP Transmission Control Protocol. 155, 156, 158

TLS Transport Layer Security. 7, 9, 25, 28, 29, 32, 70, 72, 74, 88, 156, 158, 162, 163, 169

TURN Traversal Using Relays around NAT. 76

UART Universal Asynchronous Receiver-Transmitter. 112, 113, 120

UDP User Datagram Protocol. 42, 116, 157

UK United Kingdom. 47, 53, 55, 64–66, 72, 85, 99, 100, 104, 109, 110

UML Unified Modeling Language. 90

UPnP Universal Plug and Play. 93

VLAN Virtual LAN. 148, 149, 153, 158, 159

WAN Wide Area Network. 149, 159

XOR Exclusive OR. 12, 17–19

List of Publications

[A1] R. McPherson, C. Hay, and J. Irvine. Using LoRaWAN Technology to Enhance Remote Power Network Monitoring. In *2019 IEEE 89th Vehicular Technology Conference (VTC2019-Spring)*, pages 1–5, April 2019.

[A2] R. McPherson and J. Irvine. Secure Decentralised Deployment of LoRaWAN Sensors. *IEEE Sensors Journal*, 2020.

[A3] R. McPherson and J. Irvine. Using Smartphones to Enable Low-Cost Secure Consumer IoT Devices. *IEEE Access*, 8:28607–28613, 2020.

[A4] R. McPherson, G. Paul, and J. Irvine. Using Distributed Networks for a Server-Free Architecture for Secure and Private Internet of Things Applications. *IEEE Access*, 2020. Submitted.

[A5] G. Paul, K. Barlee, R. McPherson, D. Anderson, D. Crawford, J. Irvine, and R. Stewart. Rural Neutral Host Network Demonstration in Orkney, Scotland. *IEEE Tech Focus*, 2020. Wrote the financial and coverage sections.

[A6] D. Thomas, R. McPherson, and J. Irvine. Power analysis of local transmission technologies. In *2016 12th Conference on Ph.D. Research in Microelectronics and Electronics (PRIME)*, pages 1–4, June 2016. Designed, created and analysed the system, and wrote the Introduction, Experimentation and Conclusion sections.

[A7] D. Thomas, R. McPherson, G. Paul, and J. Irvine. Optimizing Power Consumption of Wi-Fi for IoT Devices: An MSP430 processor and an ESP-03 chip provide a power-efficient solution. *IEEE Consumer Electronics Magazine*, 5(4):92–100, Oct 2016. Developed the analysed system, wrote the Introduction, WiFi & 433 MHz Analysis, and Conclusion sections.

Chapter 1

Introduction

1.1 Motivation for Study

This work was motivated by the challenge of securing low-cost Internet of Things (IoT) devices. The idea came about as a response to numerous IoT devices being compromised [1–11], even though there has been considerable security research advancing the state of the art. Therefore, this study will investigate the current obstacles to implementing up to date security features in low-cost IoT devices and present solutions to these obstacles. This presents a technical, economic, and societal challenge. The technical challenge is ensuring devices have strong integrity, confidentiality and availability of data. The economic challenge is ensuring security does not present a monetary barrier, and the societal challenge is making sure that the resulting device remains usable. Although there has been research into securing low-cost devices, which are described in Chapter 2, these solutions tend not to be implemented for various non-technical reasons [12–14].

Bearing these considerations in mind, this thesis will focus on effectively deploying low-cost IoT security, specifically adding security features with little or no increase in the manufacturing cost of the device. Secondary goals of this thesis are minimising additional user setup to avoid detraction from the ease of use, as well as maintaining device responsiveness. Although this problem has been solved with high-performance systems, such as smartphones, it still presents a considerable challenge for computationally-constrained devices [15]. It is also important to note that the overall security of a device is not defined by a single connection, but by the weakest link. Therefore, each proposed solution to secure IoT devices will be examined as a whole.

The overarching theme of this work is to remove or reduce the barriers for manufacturers to correctly implement security features, given that the cost is a defining factor for consumers when deciding on which product to purchase [14]. This issue is compounded by manufacturers finding it difficult to differentiate themselves based on the inclusion of security, and therefore, struggle to charge a premium for their product [16]. Making this work as accessible and applicable to as many manufactures as possible will hopefully increase the overall cyber security of the industry.

1.2 Research Questions & Contributions

The contributions outlined in this study and its subsequent publications demonstrate that security can be provided with limited additional hardware. At the onset of this study, the primary research question was "Can the requirement for additional components for securing IoT devices be reduced?". The first contribution to the field, emerging from this question is detailed in Chapter 4. It is the inclusion of security features to domestic low-cost IoT devices at no additional hardware cost. As will be discussed, the addition of those security features has not detracted from any user experience. In fact, the modified architecture improved device usability, as it enabled the user to control their devices on the local network and configurate them without specialised knowledge. The proposed architecture resulted in the publication [A3].

The second research question, "Can the deployment of secure IoT devices be simplified?", is detailed in both Chapters 4 and 6. Automating the installation of keys in domestic IoT devices and industrial devices, using two separate architectures, simplifies deployment. Namely, it was noted key distribution was a common issue for low-cost wide-area Industrial IoT devices as well as Wi-Fi based domestic devices. Chapter 6 discussed the extent of this laborious process to install security credentials on each device without automation. The lack of automated key distribution also means that if the network got compromised, all deployed devices would have to be manually re-provisioned. Therefore, the second contribution of this PhD study is the creation of a Industrial Internet of Things (IIoT) provisioning architecture, using a low-cost component, that can be used with a smartphone to enable automatization of re-keying the deployed devices. This idea further led to the publication [A2].

The third research question, “Can critical national infrastructure IoT devices be secured with only passive sensing, ideally with unidirectional communications?”, is further discussed in the fifth chapter. In its current state, the electrical grid requires increased monitoring. As the utility networks are a target of attackers, any such communication must be secured. Therefore, an additional contribution to the field is the creation of a monitoring device and an evaluation of possible low-power wide-area network technologies for critical national infrastructure use cases, which was published in [A1].

The final research question was “Can the dependency of external components for IoT be reduced?”. IoT devices currently depend on servers to function, bringing ongoing maintenance costs of service provision. These servers have to be continually updated and patched to maintain security, and once decommissioned, IoT devices will lose functionality. Therefore, another contribution of this PhD study is enabling low-cost IoT devices to communicate without centralised hardware. This provides a number of additional security benefits, as there is no direct link between the IoT device and the manufacturers’ server. As will be discussed in Chapter 7, the proposed system uses a Distributed Hash Table (DHT) to decentralises communication, further minimising the potential for security breach. Finally, this change in architecture improves devices’ longevity and shields them from malicious attackers. This advancement in the field was presented as the [A4] paper and is currently submitted for publication.

Overall, this PhD improved the security of low-cost IoT devices by proposing alternative architectures for minimal additional hardware cost. The first system can be used in domestic IoT devices, as well as in industrial settings, removing the need for specialist knowledge in deployment and re-keying of devices. Creating a provisioning architecture brought automatization to the IoT security implementation, further improving current state of the art practice. Finally, a separate architecture isolates devices and reduces their dependency on centralised servers, increases their longevity and security. This PhD will now present the structure of the thesis, followed by a discussion on the IoT use cases examined in this PhD.

1.3 Structure of the Thesis

This thesis explores four different areas of IoT: domestic IoT devices, industrial IoT devices, provisioning IoT devices, and decentralised IoT devices. The first chapter introduces the reader to the field of IoT, and defines the difference between high performance devices which can natively operate state of the art security functions and low-cost devices which cannot implement these security features. The second chapter details the reasons why low-cost devices cannot support these security features without requiring additional hardware or incurring a poor user experience. It will go onto explain a strong candidate for the simplification of securing low-cost devices, low-power wide-area network sensor devices. Finally, an evaluation threat model is explained to measure the effectiveness of the proposed solutions. The third chapter details why reducing additional hardware cost and ease of use are important to the success of low-cost IoT devices. This will then be expanded upon explaining the reasons behind why some of these existing solutions have not been implemented, upcoming legislation focused on securing IoT devices.

The next four chapters detail proposed solutions for low-cost devices. Chapter 4, addresses domestic IoT devices where the author proposes an architecture to run state of the art security features on a computationally-constrained low-cost microcontrollers. The performance of the proposed architecture is then profiled and evaluated. Chapter 5, investigates low-power wide area networks, through the development of a secure sensor device, and does an evaluation of the security of various LPWAN, and their associated security. Chapter 6, Addresses the challenges of programming Low Powered Wide Area

Network (LPWAN) sensor devices with suitably random keys, and proposes a system for simplifying the provisioning procedure, allowing non-specialised individuals deploy and re-key sensor devices, reducing the time of (re)deployment. Chapter 7, looks at improving IoT device longevity by removing the dependency on a central server, This chapter details various issues concerning the current centralised architecture of IoT devices - longevity, security, and dependency, and proposes a solution to solve each of these issues by employing a decentralised architecture. Finally, Chapter 8, concludes the thesis and detailed an overview of the contributions. Chapter 9, proposes how this work could be expanded in the future.

1.4 IoT Use Cases

IoT devices have the potential to provide a wide range of benefits and can be deployed to assist in virtually every sector. In agriculture, smart farms and precision farming use Agritech to increase efficiency. By helping farmers know exactly when to harvest crops, which areas could benefit from increased fertilisation and where the seeds have mostly been eaten and require replanting. In livestock farms, sensors can provide: health information, medical alerts, eating habits and location, with all of these features improving the animal's quality of life. Health monitoring does not only apply to animals; increasingly internet-connected health equipment is becoming mainstream, from fitness tracking to internet-connected pacemakers. More sensors will allow doctors to have a better idea of the exact condition of the patient, and prescribe a personalised treatment. This is particularly beneficial in times of high demand or limited access to medical care. Additionally, sensors can also detect dangerous situations giving users peace of mind, and increased independence to otherwise at-risk individuals. For example, epileptics could drive safely if their car detected the onset of a fit and safely brought the car to a halt. A similar automotive system could be employed for elderly people to provide increased mobility. Accelerometers can also be employed while walking, for fall detection, which would automatically call for assistance. Similarly, non-invasive detection systems could be deployed in care homes, to see if an individual has left their bed in the middle of the night and has not returned for a significant time, as this may be an indication they have fallen on their way to the bathroom and need assistance.

In logistics, companies such as Allocco and Amazon have fleets of robots which move around distribution centres arranging, receiving, and storing products [17]. Each of these robots uses sensors to perceive their environment and communicates with a centralised control system to receive orders and report issues. Similarly, in manufacturing, sensors will check products for deficiencies, and as Industry 4.0 becomes commonplace, cyber-physical robotic systems will increasingly interact enabling more seamless order customisation. Previously this would have required a separate process, causing a halt to production. However, this alteration could be achieved through intelligent robots and sensors, previously even something as simple as a change in colour would have had to have direct human engagement but through automation and sensors this can be done seamlessly.

Utility companies maintain vast physical networks, and it can be a very time-intensive task to find exactly where the source of a problem is, and this is magnified in rural areas where an engineer might have to travel a significant distance to reach certain sites. Therefore, using monitoring sensors to detect a fault in the network, can dramatically decrease the time customers spend disconnected from the service. Additionally, as new threads, such as electric vehicles and distributed generation [1] are being introduced, more granular control over the network is required.

Finally, home automation devices advertise convenience, financial and energy savings, by selectively engaging appliances based on learned user preferences. This long list of potential IoT applications gives an impression of how large the field of IoT is. This can be seen through both the current market growth and the predicted market growth which according to Cisco will be 500 billion devices by 2030. As many of these systems are the evolution of existing computer networks, existing solutions have been applied to secure them.

[1] Distribution generation is the process of decentralised electricity generation commonly through renewable sources such as wind turbines

One of the most prominent solutions is public key infrastructure, the most common variant being Transport Layer Security (TLS) [18], which is used to generate and transfer random keys. One consideration with such an approach is TLS requires a high performance processor. Personal computers, smartphones, and home automation devices, such as Google Home, Alexa or Siri, all contain high-performance x86 or ARM processors as they are required for the devices primary purpose; therefore, there is no additional cost associated employing security on these devices.

In contrast, sensor devices only require limited functionality, possibly just a single trigger and report mechanism, such as a temperature sensor. Thus, there is no requirement for a x86 or ARM processor, which would increase the cost of manufacture for the device with marginal benefit, as well as its energy requirements, resulting in additional challenges of supplying power to a device, restricting the number of possible locations, and increasing the cost of installation. Therefore, it is beneficial to be able to operate devices from battery power and using low-powered processors significantly extends the battery life of these devices compared to using a high-performance device such as an x86 or ARM processor. These are two of the primary reasons that sensor devices often opt to use low-cost processors, incentivising manufactures to use a lower-cost alternative [14, 16, 19].

The downside of using low-cost processors is that modern state of the art cryptography requires random numbers [20–23] which standalone computationally-constrained devices struggle to generate [24]. Given this, two distinct categories of systems will be considered in this PhD thesis, high performance devices which are able to implement state of the art security features natively, and low-cost devices which are unable to implement these security features natively. Part of the motivation for this study concentrating on low-cost devices is that computer networks are envisaged to feature a range of low-cost, such as sensor devices, and high-performance devices, such as smartphones and personal computers. Therefore, compromising a low-cost device can be used as a steppingstone into an otherwise secure network. Determined attackers would use any avenue possible, as demonstrated through side-channel attacks and similar [25].

Some of the most recent publicised IoT hacks have been in relativity low-cost consumer products: Phillips Hue Smart Lightbulbs, Vtech Tablet, Amazon Ring Video Doorbell, a voice recording stuffed bear, and numerous video cameras [5, 8, 9, 26–30]. The commercial availability and minimal outlay required to investigate these types of products make them prominent targets for attackers. However, it is important to note without legislation or oversight it is likely the same principles used to design a low-cost consumer IoT devices would be mirrored within a more sensitive market, and both manufacturers would experience similar issues. A recent industry report found that 98% of IoT device traffic is unencrypted [31].

When designing an IoT product the manufacturer should incorporate threat model and a likely attack scenario of each individual device, as well as reflect on what can be inferred from its critical functionality. For example, a recent popular technology has been voice assistants, Google Home, Alexa and Siri. These voice assistants operate by permanently listening for a trigger word, and could reveal everything which was said in a home if they were broken into and set to transmit data to the attacker. Therefore, such a compromise would violate user's privacy. Whereas, if a domestic thermostat was compromised, an attacker could control the user's heating, but less personal data would be available. Consequently, each situation is different but improving the security for all devices would be beneficial.

With regards to critical functionality of the system, it is important to consider what can be inferred from it. For example, a smart lock can only have two states, locked and unlocked, and therefore it is likely there is also only two commands lock and unlock. Therefore, it would be important to both authenticate and encrypt these messages, so that an observer could not deduce which message was which. Alternately, there are domestic devices where the data does not have to be secret but should be checked for integrity, such as ambient temperature measurement. While this could be estimated from knowing the surrounding environmental temperature, an attacker influencing the message could cause a heating/cooling system to function, causing discomfort and unintended financial cost to the occupant.

A determined attacker can find multiple ways to attack computer systems, for instance, a remote temperature sensor could be physically moved to another location to alter the recorded measurements, or opened to modify the physical connection between the sensor and the processor. Even once encryption and authentication are employed, side-channel attacks are still a possibility. Therefore, there are numerous ways an IoT device can be influenced, but the overarching theme of this PhD study is to decrease the cost of implementing security features. This cost can be manifested as financial, energy or user experience. Each of these costs is detailed in Fig. 1.1. Hardware cost is cost of the raw materials required to create the IoT device. Maintenance cost refers to the ongoing cost of supporting software updates and maintaining servers. Ease of setup, is the complexity for a user to setup the device. User experience refers to how easy or otherwise it is for users to interact with their devices on an going bias. Finally, required performance determines what processing hardware is required and therefore what energy requirements the device will have.

An ideal secure IoT device should only require minimal resources, provide strong security features, and be simple to setup and use, with minimal maintenance and hardware costs. This is shown in the spider diagram Fig. 1.1 where the attributes on the left-hand side are negative and the attributes on the right are positive. Three example device classifications are detailed, a high performance IoT device, such as a smartphone. These devices can implement state of the art security features, such as TLS, while providing a seamless secure user experience [18]. However these devices are also battery intensive [32], and financially prohibitive for single purpose devices such as sensors [33]. A second classification is industrial devices which commonly lack usability, and state of the art security features [34] while incurring high hardware costs due to the increased reliability requirements, and increased maintenance costs given the envisaged life time [A1]. Finally, a low-cost IoT device is presented such as a temperature sensor, which is financially inexpensive, but also requires technical knowledge to setup, and lacks standalone security features. The aim of this thesis is to reduce the hardware, maintenance, and performance costs while increasing security features, user experience, and ease of setup. In terms of the spider diagram (Fig. 1.1) moving the points towards the right/green side.

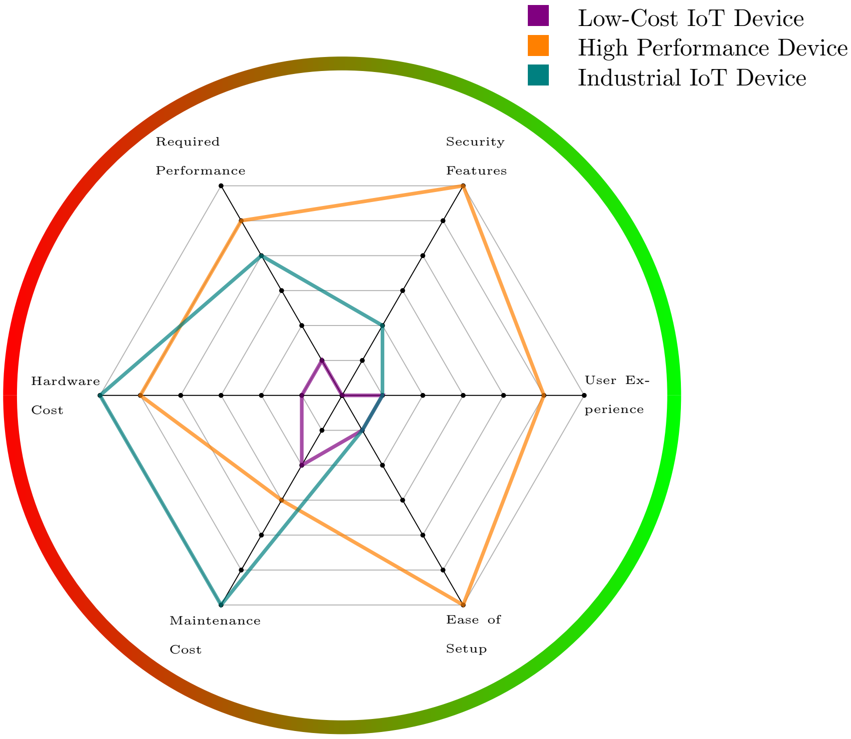


Figure 1.1: Spider Diagram of Current State of IoT Security

Chapter 2

Technical Background

2.1 Introduction

Cybersecurity aims to provide confidentiality, integrity and availability to systems. Confidentiality certifies only the intended recipient can read the message. Integrity verifies that the message has not been tampered with, and availability ensures the system functions as expected. These three names have been defined relatively recently but security algorithms have been employed for thousands of years. One of the first algorithms was the Caesar cipher, named after Julius Caesar, whereby letters were shifted in the message to make it unintelligible to anyone intercepting it [35]. However, with knowledge of the shift the receiver could reverse the operation and read the message. This is a rudimental algorithm by modern standards, but it does share the same properties as modern encryption methods. Namely, the message sender modifies the data in a deterministic way based on a set of rules. The output of these rules makes the message unintelligible, however the receiver can reverse this process if they know the exact same parameters used encrypt the message. In modern encryption algorithms there are so many possibilities it is statistically extremely unlikely that an eavesdropper could guess the exact parameters, therefore the message is encrypted. This shared parameter is commonly known as a key and should be unpredictable. All subsequent algorithms share these properties, two inputs used to encrypt the data, a key and the plaintext, and a single output, the ciphertext.

Decryption employs a similar process, two inputs — a key, and ciphertext, and one output — the plaintext.

2.2 Symmetric Cryptography

A specific type of encryption is symmetric encryption this is where the decryption process is the inverse of the encryption process, therefore the key is a shared resource. This can be further divided into block ciphers and stream ciphers. Block ciphers take a collection of bits (a block) as input data and apply an algorithm to shift and modify each of the input bits, based on a series of rules and the specific key. The enigma code employed by the German Military in World War 2, was cracked through allied cryptanalysts noticing links between the input and the output [35]. Therefore, advanced security algorithms are designed so that a slight change in the input will result in a substantially different output. This idea is called diffusion, where a single change in the plaintext of a message, should change several characters in the ciphertext of that message [36]. Likewise, on a decryption operation, a single character in ciphertext should dramatically change the generated plaintext.

The disadvantage of block ciphers is that the input data has to be a multiple of the block size. For instance, a common modern block size is 16 bytes. A message of exactly 16 bytes would encrypt exactly, but if the message was 17 bytes long, two blocks would be required to encrypt the message, where the majority of the second block would be padding, creating unnecessary transmission overhead. Stream ciphers, on the other hand, do not have any fixed-size requirement and can encrypt an arbitrary length of data. This is done through stream ciphers encrypting a series of blocks which are of equal or greater length to the plaintext. The Exclusive OR (XOR) of the encrypted stream and the plaintext is then taken with any additional bits in the encrypted stream discarded, creating an exact length ciphertext. The disadvantage of this scheme is that the input bit directly corresponds to the output bit, as an XOR is a single bit operation, resulting in low diffusion, which will be expanded on in Section 2.2.2.

2.2.1 Block Ciphers

IBM developed the Data Encryption Standard (DES) in the early 1970s, which went on to become the first standardized symmetric encryption algorithm in modern computing [37, 38]. The algorithm used 64-bit keys, with eight parity bits, providing 2^{56} or 72 quadrillion possible key values, sufficient security at the time, however, as processing power increased and computers became increasingly interconnected, the prior security assumptions were no longer valid. RSA Security, a security organisation, proposed a challenge of breaking DES in January of 1997, with a prize of $10,000. A group of researchers took up this challenge, and named themselves DESCHALL, a combination of DES and challenge. The group employed a distributed approach designed to use spare capacity on various machines, and collaborate through the internet. The group designed an algorithm to check every single possible key systematically, in a process known as brute forcing [39]. A single computer at the time would have taken an estimated 2,285 years, at roughly 1 million guesses per second to find the correct key. Instead, the algorithm was designed to distribute the workload between various computers, increasing the key guess rate. As the project progressed the number of computers involved increased, at one stage reaching 78,000; with so many computers guessing keys, any 64-Bit DES key would be able to be discovered in 119 days. As the number of computers joining the network was still increasing, the key was discovered in only 39 days, a quarter of the way through the key space[1]. The owner of the computer which successfully guessed the correct key was awarded $4000, with the rest going to the creators of the project. A year later the Electronic Frontier Foundation (EFF) developed a dedicated DES cracking machine which was able to crack any DES key in 56 hours [40]. This machine cost less than $250,000, comfortably within the budget of nation-states or dominant private companies. Finally, a collaboration between the EFF and Distributed.net, another organisation which distributed the load of cracking keys, combined both of these approaches to break any DES key in 22 hours and 15 minutes, proving this encryption scheme was no longer suitable. In response, Triple DES was created. This new

[1] The key space is the number of possible keys in a system

scheme used a series of encrypt, decrypt and encrypt operations with three separate keys to enhance security [41]. Nevertheless, this would not be sustainable. Therefore, work started on creating a new encryption standard.

In early January 1997, the American National Institute of Standards and Technology (NIST) announced that it wished to choose a successor to DES [42]. The announcement was met with great interest from the cryptographic community, and soon NIST amended its call to request specific algorithms to replace DES. Each algorithm was expected to support key sizes of 128, 192 and 256 bits, dramatically increasing the number of possible keys over DES by least 2^{72} times. Fifteen different algorithms were submitted for review. The grading criteria would measure the algorithms on its security features and performance on both powerful computers and low-performance devices such as smart cards. Through the review process some algorithms presented vulnerabilities, and others were infeasible in resource-constrained environments. Therefore, the number of possible candidates was decreased to five with an algorithm designed by Daemen & Rijmen, called Rijndael [23] receiving the most votes. Consequently, on the 2nd October 2000, NIST started the process of turning Rijndael into the new Advanced Encryption Standard (AES), where it became the official United States encryption standard the subsequent year [43]. As a note to the reader, NIST is currently organising a similar consultation for developing quantum-safe algorithms [44].

After its standardisation AES has been widely deployed with both software and hardware implementations becoming commonplace. As of the time of writing, there is no publicly released viable methods to full crack AES encryption. AES was not an evolution over DES, rather a revolution, where distributed attacks worked on DES, a similar attack on a 128-bit AES key, even if undertaken by all of the supercomputers in the world would still take 10^{25} years, while the age of the universe is only 10^{10} years [35]. Instead, other common AES attacks were considered.

One such attack by Biryukov et al. found that a nine round implementation of AES 256 Bit Keys (AES-256) could be exploited to reveal the encryption key. This process required two related keys, and took taking 2^{39} computational time. A subsequent attack broke 10 round AES-256 in 2^{45} time [45]. Both attacks offered a significant increase in the state of the art where previously this attack would have required 2^{176} and $2^{99.5}$ time for AES-192 and AES-256 respectively [46]. The decrease in required time makes the attacks practical. However, the attack had three stipulations: it only works on AES-256, the cryptanalyst required access to multiple plaintexts encrypted with related keys, and the attack is only practical for ten rounds, where full AES-256 encryption is 14 rounds.

Another attack was later developed on full AES uses a biclique attack to reduce the key space of possible keys to $2^{126.1}$, $2^{189.7}$ and $2^{254.4}$ for AES 128 Bit Keys (AES-128), AES 192 Bit Keys (AES-192) and AES-256 respectively [47]. This attack uniquely did not require related keys or a reduced number of rounds. The attack also had small memory requirements where commonly other attacks required a considerable amount of memory. However, this attack is also not considered practical as the key space is still considerable. Instead, side-channel analysis was considered. These attacks rely on the physical properties of a system such as the time it takes for an encryption operation to occur, or the amount of energy consumed while preforming an encryption operation. Minute differences in time or energy consumption can then be recorded and the AES key can be inferred. One example of this type of attack was demonstrated by Daniel J. Bernstein, where he found that AES was vulnerable to remote cache timing attacks [21]. Through issuing various commands to a local webserver he was able to populate the cache with specific data and then based on the data access times infer the AES encryption key.

Although this was successfully achieved, it has only been demonstrated on a closed network and would be considerably more challenging on a live system. Moreover, four separate ideas have since been developed, which would compound the difficulty. Firstly, cache timing attack countermeasures have been deployed [48]. Secondly, x86 AES-NI instructions decrease the execution time. Thirdly, processing power has increased, meaning the network latency would represent a higher proportion of execution time. Finally, Hardware Security Module (HSM) (Section 2.5.2) are increasingly being deployed in high-security environments achieve tasks in constant time. In response to his findings, Bernstein created the ChaCha20 algorithm which preventing such a caching attack. Despite Bernstein's theoretical attack, this thesis will examine AES given its popularity, extensive cryptoanalysis and widespread support, and AES has been standardised by NIST.

2.2.2 Block Cipher Modes

AES can operate in various different modes of operation, but as with all block ciphers, the input data has to be an exact multiple of the block size. Any extra data which is longer than this, will be rounded up to the nearest block multiple. This is commonly done by padding the message with bytes, extending the overall length of the message. There are various schemes employed for this task, but one of the most popular is PKCS#7, which involves adding bytes of the value of the number of bytes required to pad the message. For example, if the message was 25 bytes long, and therefore seven bytes short of two 16-byte-blocks, the message would be padded with seven bytes of 0x07. This scheme is used so the receiver can determine where the message ends and the padding begins. For example, if the message ends with two blocks of 0x1, the receiver can determine that the last block is padding, while the penultimate block is the last block of the message.

Electronic Codebook

The simplest AES mode of operation is Electronic Codebook (ECB). The message is broken into sections 16 bytes long — a block — to match the state size and each of these blocks is passed through the cipher to create a ciphertext, as shown in Fig. 2.1. Unencrypted text, known as plaintext is passed into the encryption alogirthm one block at a time, as denoted by p_n, and a key k is used to encrypt that block and output a ciphertext C_n. The main disadvantage of this mode of operation is each identical plaintext block will map to the same ciphertext block, so that information can still be deduced from the ciphertexts. Consequently, ECB is not recommended for messages longer than a single block long [49,50]. This was famously demonstrated with an image of Tux the Linux penguin encrypted using ECB, as shown in Fig. 2.2. This occurs because the plaintext value of the colour white is #FFFFFF, so once the value is encrypted with ECB, the ciphertext may be #ABABAB, which would map to a light grey, and therefore still clearly show the outline of the picture.

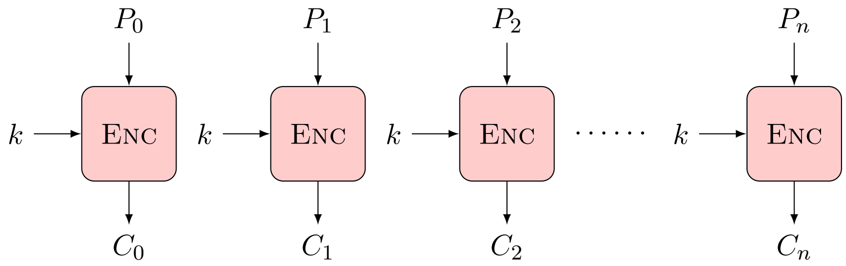
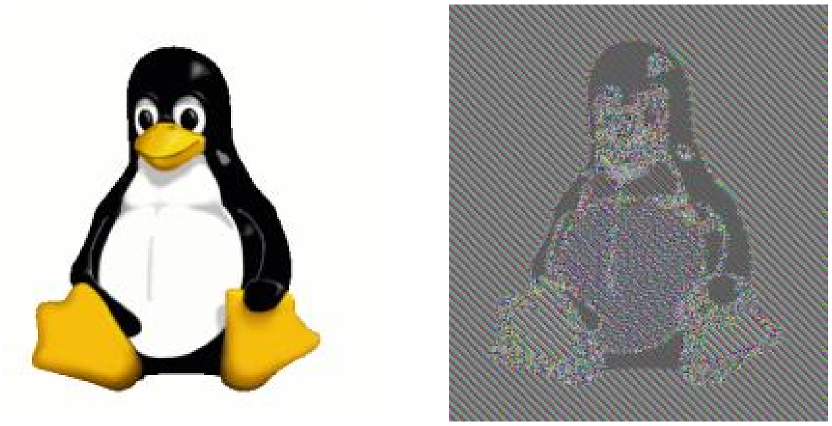


Figure 2.1: Block Diagram of Electronic Codebook Encryption



(a) Original Image

(b) Image encrypted with ECB

Figure 2.2: Tux Penguin Encrypted with Electronic Codebook
Attribution: lewing@isc.tamu.edu Larry Ewing and The GIMP.

Cipher Block Chaining

Cipher Block Chaining (CBC) is the second AES mode of operation which will be discussed. As highlighted in Section 2.2.2, it is important that each block encrypts to a discrete value. CBC resolves this problem by linking or chaining all of the blocks together, as shown in Fig. 2.3 To link blocks together without extending the overall length of the message an XOR operation is used. An additional advantage of XOR operations is that they are very efficient resulting in a low computational time. The output from the previous block is XORed with the plaintext of the current block. The result is that two identical plaintext blocks generate two distinct ciphertexts. For the initial block, an Initialisation Vector (IV) is used, which is simply a random number which ensures that the first block in the sequence translates to a unique ciphertext. The IV does not need to be secret but does have to be suitably random [51]. If the IV was deterministic, an attacker could use a series of chosen plaintext operations to determine information about previous encryptions without knowledge of the key.

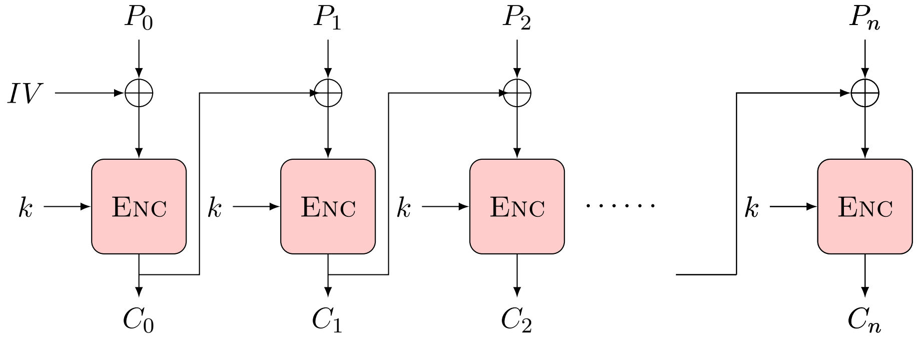


Figure 2.3: Block Diagram of Cipher Block Chaining

Output Feedback Mode

Another mode which was detailed within the original AES proposed suite was Output Feedback (OFB). Similarly, to CBC mode this mode requires a random IV, but unlike CBC, OFB initially encrypts the IV and then applies an XOR operation on the encrypted IV and the plaintext creating a ciphertext. The encrypted IV is then used for the input of the subsequent block and is encrypted to create the input for the next block, creating a string of encrypted values, as detailed in Fig. 2.4. The advantage of XORing the plaintext with the ciphertext is that the string of encrypted values can be pre-determined, and any number of plaintext bits can be encrypted without the need for padding. Another benefit of this system is that bit errors during transmission do not propagate for the whole encryption. Both of these are potentially advantageous for constrained devices, as an encryption stream could be pre-calculated to reduce latency, and message retransmissions would be reduced.

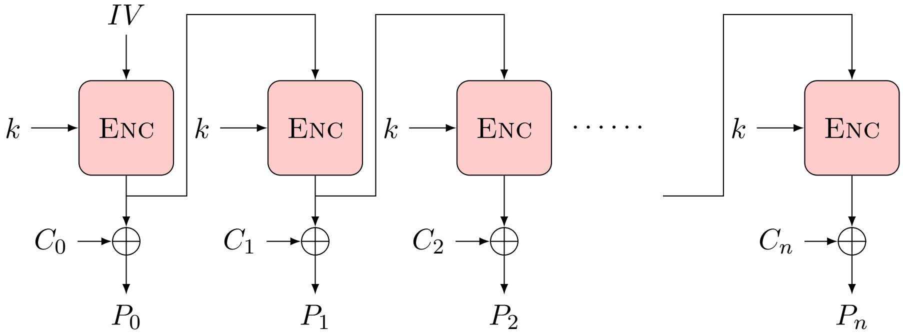


Figure 2.4: Block Diagram of Output Feedback Encryption

Counter Mode

AES counter mode is slightly different from the previous mode, while both CBC and OFB requires a random non-deterministic IV, counter mode's IV does not need to be random, instead, as the name suggests the IV is a counter. A nonce and a counter are encrypted using the AES key, and then the plaintext is XORed with the encrypted IV creating a ciphertext, as shown in Fig. 2.5. As each block is independent the process can be parallelised increasing the speed of execution. The counter value will increment with each message therefore, even if two identical plaintext blocks are XORed with the encrypted IV two discrete values will be created. The risk of using counter mode is if the same counter-value/IV is used to encrypt two messages, the XOR of the two ciphertexts is equal to the XOR of the two plaintext messages. $p_1 \oplus p_2 \equiv c_1 \oplus c_2$. Therefore, once the attacker XORs two publicly available ciphertexts together, the difference between the messages would be the same as the difference between the plaintexts. Therefore, information could be inferred about the plaintexts. Using a different IV every time would solve this issue [52].

Galois/Counter Mode

Unlike all of the previously introduced modes of operation Galois/Counter Mode (GCM) provides authentication and encryption in a single cipher providing a performance benefit over calculating both operations separately [53]. Also, unlike the previous modes the speed of GCM is variable on the size of memory able to build

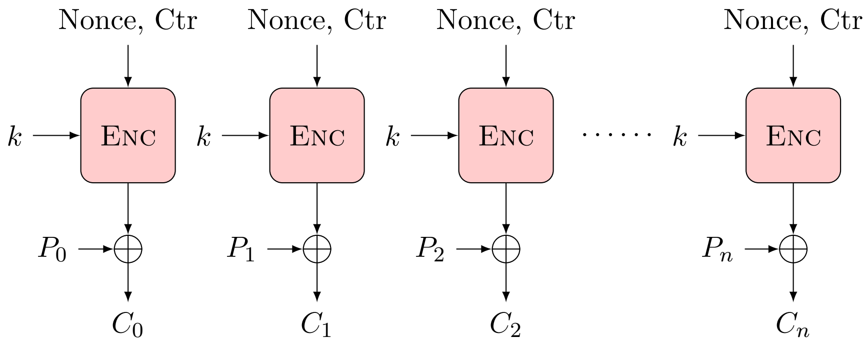


Figure 2.5: Block Diagram of Counter Mode

lookup tables. The fastest mode requires 64 kB for each key, although this comparatively small memory requirement for modern day devices, it is considerable for constrained devices [54]. Authentication is achieved through creating a tag value and a unique feature of GCM is that the length of this tag can be adjusted depending on the implementation. A shorter tag of perhaps 32 bits would require less computation to calculate however an attacker would also only require a maximum of 2^{32} attempts to forge a new tag [55], although common tag values are 96, 104, 112, 120 and 128 bits [53]. GCM also has the unique property that if only a few bytes of the message are changed then a new tag can be created authenticating the whole message at a computation cost proportional to the number of bits changed. Fig. 2.6 shows that a counter value is input to the algorithm and duplicated. The first copy is encrypted and used as an input to the authentication tag. The second copy of the counter is incremented and encrypted, outputting a cipher, this is then XORed with the plaintext to create a ciphertext. This process is repeated for each of the blocks in the message. Any additionally authentication data is XORed with each of the created ciphertexts, and a modified through a series of This second counter value is then encrypted creating a random output, which is XORed with the first block of plaintext to create a ciphertext. This process is repeated for each of the blocks to be encrypted, and the results are taken as the output ciphertext but also multiplied together. This intermediate result is then XORed with the length of the authentication concatenated with the length of the ciphertext which is XORed, and finally XORed against the original encrypted counter value to give an authentication tag. A disadvantage of this mode is that if the same nonce is used twice, ciphertexts could be easily forged by an adversary, and there is a maximum message length of 68 gigabytes, a maximum number of 2^{64} blocks, and a maximum number of messages 2^{32} which can be encrypted before the probability of a repeated nonce is sufficiently high to become inadvisable. This is an on-going area of research through AES-GCM-SIV [56].

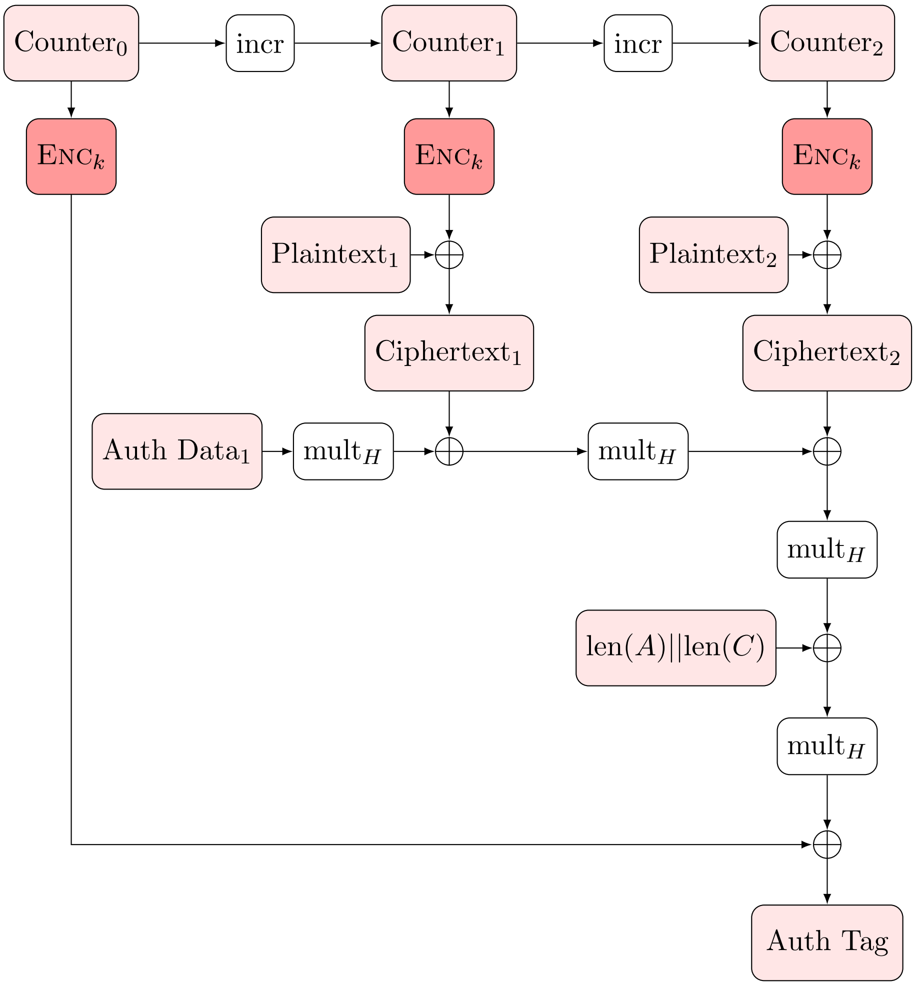


Figure 2.6: Block Diagram Galois/Counter Mode Encryption

2.3 The Key Distribution Problem

A well-documented issue with symmetric encryption is that both the sender and receiver are required to possess a copy of the key before secure communication can occur. This means that the keys must be given in advance using another means of communication as, if the keys were transmitted over the same connection, any attacker would be able to simply read the key and then read all future communication. This is an issue for the initial setup of a device, and if the device was later compromised, as the device and server would have to be re-keyed and another secure connection would have to be created. One proposal researchers suggested for low-performance wireless sensor networks was using shared keys, where each device has the same key; however, if one key is compromised, the whole network would be compromised [57]. Therefore, for very constrained IoT devices this is an ongoing research problem. High performance devices, on the other hand, use asymmetric cryptography.

2.4 Asymmetric Cryptography

2.4.1 Diffie Hellman Key Exchange

Whitfield Diffie and Martin Hellman first came up with a solution to securely transfer keys in 1976. By using mathematical properties, they were able to devise a method for creating a secure communication channel without any prior knowledge. The downside of this method is that it cannot be used to encrypt messages directly. Instead, a shared key is generated between the communicating parties, which is then used to encrypt future communications.

The algorithm relies on a trap door function, which is a property of mathematics where it is simple to compute the result, but very difficult to work out the inputs from the output. For example, Alice and Bob, wish to securely communicate without first meeting up to share a key. Therefore, they start an insecure communication channel and first agree on a prime modulus p and a base g where g is a primate root of the p. Next, Alice randomly generates a private value a. Alice then calculates A, where

$A = g^a \bmod p$ Similarly, Bob generates his random private value of b, and calculates his version of B using the same formula. The two parties then swap their calculated versions A and B. Each person then takes the value they received and use it as the base, and their own private number as the exponent, after applying the modulus of the p, each party will have the same number $g^{ab} \bmod p$ which can act as a shared secret this is detailed in Fig. 2.7. But an eavesdropper would not be able to calculate the key due to the mathematics of trap door functions.

This concept is now known as public-key or asymmetric cryptography, due to the use of a public key — a value which can be made known to the public — with an asymmetric algorithm. The private key, a and b, should be kept secret. The asymmetry in the algorithm demands the use of a one-way or trap door function which is easy to compute in one direction but difficult to work back from.

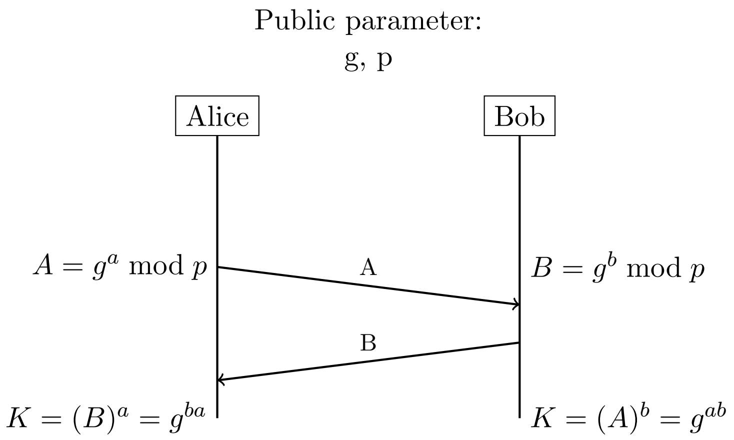


Figure 2.7: Diffie-Hellmen Key Exchange Enabling Secure Communication over an Insecure Channel

2.4.2 RSA

Rivest, Shamir and Adleman published the first public-key cipher algorithm in 1978 [58]. Although newer algorithms such as Rabin and McEliece were developed [59, 60], Rivest, Shamir and Adleman (RSA) was the simplest to understand and therefore, most straightforward to implement. Public Key Partnership (PKP), a collaboration between RSA Data Security, Inc and Caro-Kahn, Inc patented RSA along with a collection of other public-key algorithms [61–65]. Once the algorithm entered the public domain in 2000, the popularity built on top of its simplicity was cemented as licenses would no longer be required. An RSA public key requires two prime numbers, which are multiplied together to create a significantly larger number. RSA relies on the mathematical property that multiplying numbers together is easy while factoring them is hard; this is then combined with the discrete logarithm problem. An RSA keypair consists of a public and a private key, so-called because it should never be shared with anymore. RSA has two main modes using these keys. The first is enabling encrypted communication. This is done by the public key being made available to anyone who wants to send the owner of the keypair a message. The sender then encrypts their message with the public key, and sends it to the receiver who has the matching private key, the private key can then decrypt the message, so there is one-way encrypted communication. If the receiver wanted to reply to the sender securely, they could use the senders public key and encrypt their reply. The second use of RSA is as a digital signature which will be described in Section 2.4.5.

2.4.3 ElGamal

ElGamal published his eponymous public-key cipher ten years after Diffle & Hellman had published their proposal, and seven years after Rivest, Shamir, and Adleman published RSA. Both RSA and Diffie Hellman had patented their ideas, where ElGamal was not patented; although it arguably infringed on Diffle-Hellman's patent, so did not become popular that patent expired. This briefly made it a popular choice with organisations such as OpenPGP [66] which did not or could not deploy RSA until the RSA patent expired. This is an example of where non-technical barriers prevent technical advancement. As with modern implementations of RSA, ElGamal requires random number generation to operate with the additional stipulation that the random number would have to be relatively prime. ElGamal's use of finite fields makes it particularly useful to multiple recipients. Although ElGamal is rarely used in IoT devices, it serves as a powerful example of how the technology is not the only concern for implementing the state of the art as open source projects would not have the resources to license patents.

2.4.4 Elliptic Curve Cryptography

Elliptic-curve cryptography (ECC) was first proposed in 1985 [67, 68], but it took roughly 20 years before it was seriously considered for deployment. RSA had a strong foothold in the market, and it had been rigorously tested and no defects were found. However, as the recommended key lengths for RSA were ever-increasing to outpace attackers, and these longer keys took longer to compute and required more storage, therefore, in 2006 the TLS working group created a draft to add elliptic curves to TLS [69]. Later in 2009 an RFC was created for adding elliptic curve algorithms to Secure Shell (SSH) [70].

OpenSSL, was one of the most popular SSL libraries at the time, which was later forked[2] to create BoringSSL and LibreSSL in 2014, in light of the Heartbleed vulnerability [71]. BoringSSL is used in Google products such as Google Chrome, and LibreSSL used in various BSD projects. At the time, OpenSSL was estimated to be installed on between 24% and 55% of webservers [71]. OpenSSL added support for elliptic curves in 2010 [72], and OpenSSH added support in 2011 [73]. But the adoption of ECC was further limited in the early 2010s with by the leak of confidential National Security Agency (NSA) documents by Edward Snowden. One leaked document seemed to suggest the NSA had deliberately weakened a popular elliptic-curve cryptosystem to allow for a backdoor, although this was never definitely proved, researchers recalled an earlier conference presentation in 2007 which detailed the possibility of a backdoor in one of the NIST recommended elliptic curve algorithms, SP800-90 Dual EC Prng [74]. This possibility of a backdoor created doubt that the whole family of algorithms were compromised. Therefore, alternative curves were investigated, and Daniel J Bernstein's X25519 created in 2006 [20] was considered, this algorithm would only require 256-bit keys for the same level of protection as 2048-Bit RSA keys. Moreover, this algorithm runs in constant time,

[2] forked: a clone of a code repository

protecting it from timing-based side-channel attacks, it also advertises high-speed processing, and only requires a small amount of storage. All these features make it beneficial for computationally-constrained devices. As such a number of implementations has been created for constrained microcontrollers however as X25519 relies on a random number component for a strong implementation a random number source would still be required [20, 75]

2.4.5 Digital Signatures

Digital signatures are used to confirm the identity of a message sender [76]. The same public and private key pair used in asymmetric encryption can be used but the messages are not encrypted. Instead, a copy of the message is hashed, and the private key of the message sender is used to create a fingerprint for the message, this fingerprint is sent along with the message. The receiver of the message then hashes the message they received, creating their own fingerprint using the senders public key. If this value is the same as the included signature in the message, then the message receiver can be sure that the owner of the presented public key has sent the message. A digital signature should be easy to compute by the signing party and also easy to verify without requiring private information. In RSA signatures are created through the exponential multiplication of the key, therefore even constrained devices can verify them. It is also important for the signature to have an appropriate lifespan, as technology advances the signing keys could be broken and therefore signatures forged. This means that a certain number of bits is required to ensure validity, currently the NIST recommendation is at least 2048-bits for RSA signatures, but only 512-bits for ED25519 [77]. Digital signatures are commonly used to sign firmware updates so that a user can be sure that that they are updating their software to a legitimate version.

2.4.6 Public Key Infrastructure

Asymmetric encryption enables setting up a secure communication channel; however, this is only part of implementing a secure system. A client should also verify they are communicating with the intended system. Even if a user has attempted to connect to the correct site, a man-in-the-middle attack could see this request, and modify the responses from the legitimate server, including swapping the initial encryption key. This would then allow the attacker to decrypt any of the information the client is sending to the server, while still giving the client the illusion of security, as detailed in Fig. 2.8.

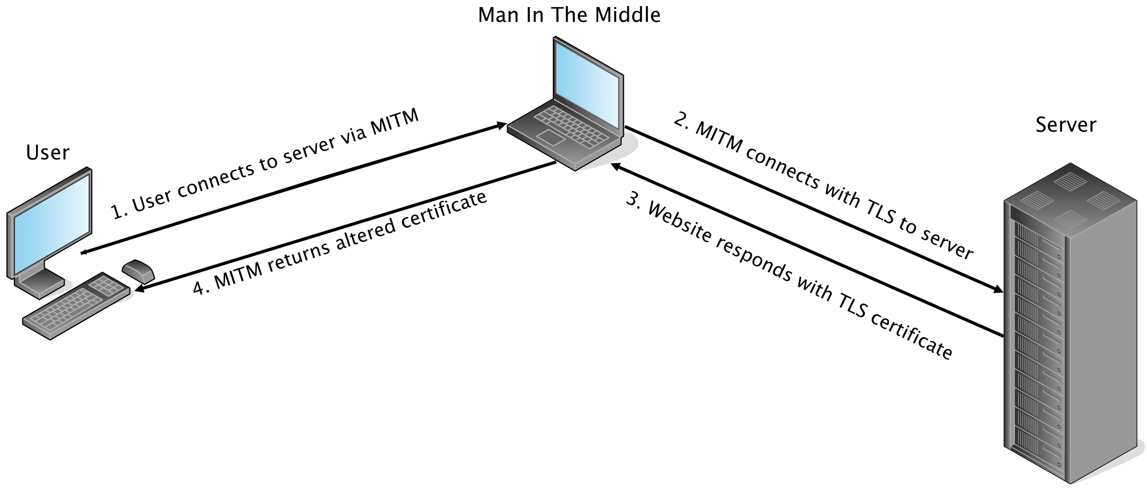

Man In The Middle
User
Server
1. User connects to server via MITM
2. MITM connects with TLS to server
3. Website responds with TLS certificate
4. MITM returns altered certificate

Figure 2.8: Man-in-the-middle Attack on Public Key Infrastructure

Therefore, it is critical to ensure that the public key came from the correct organisation. To resolve this issue, a trusted third party verifies the server is who it claims to be. These trusted third parties are known as Certificate Authorities (CA) which generate keypairs on behalf of the server, and digitally signs these credentials using the CA own private key, meaning the signature can be verified using the CA's public key. The CA itself may have its private key signed by a higher-level CA, until it reaches the top level of CA known as a root certificate, which is generally installed into systems through browsers or operating systems, so they can be verified locally. Certificates can also be revoked if a private key is compromised, and include a validity period, to ensure that an up-to-date certificate is being used. DigiNotar, a root CA was noticed to be compromised in early September 2011. Therefore all 531 certificates it issued, and all certificates issued from CAs using DigiNotar as a parent CA, had to be revoked. Finally, as this was a root-level CA, browser patches had to be released.

2.4.7 Transport Security Layer

Netscape created Secure Socket Layer (SSL) to improve security and integrity on the internet. The newly emerging field of e-commerce also required buyers and sellers to be able to securely communicate, but it was also important this method would be user friendly. The first version of SSL was never publicly released due to numerous security vulnerabilities, however as it represented an advancement in the field of computer security. The design was improved with version 2.0, and released in 1995. After release vulnerabilities were also discovered in this version, and finally SSL version 3.0 was released in 1996. Shortly after release ownership was gifted to the Internet Engineering Task Force (IETF), which renamed the protocol to the TLS 1.0 and publicly released the specification as RFC 2246 [78]. The code base remained mostly the same, however some slight differences were introduced meaning the two versions were incompatible. This created the distinction between SSL 3.0 and TLS 1.0, hence why both versions are still operational as of time of writing. Although, the earlier version of this protocol is very much outdated, a recent report found that of the 150k top sites, 1.2% still support SSL v2, and 5.8% support SSL v3. The majority of websites also still support TLS 1.0, at 60.1%. TLS 1.2 is widely supported at 97.1% [79].

In a high-level overview, SSL/TLS operates by using the previous outlined systems: certificates, digital signatures, public-key cryptography, and symmetric encryption. A certificate is used to ensure the user or device is speaking to the correct system, then public-key cryptography is used to create a shared symmetric key, and symmetric encryption is then used for all subsequent communication as symmetric encryption is significantly more efficient than asymmetric. As the TLS specification has advanced, the selection of available ciphers has changed, primarily adding new proposals and removing schemes with proven vulnerabilities. One of the largest differences between the schemes is that while SSL only supported RSA, its successor TLS supports many more protocols. TLS 1.2 added support for elliptic curve cryptography, allowing for more efficient key exchange (Section 2.4.4). The most recent version, TLS 1.3 released in August 2018, took ten years and 28 drafts to complete. The primary aim was to improve the simplicity of the protocol and improve performance. It has also removed several unsafe technologies such as SHA-1, MD5, RC4, DES and 3DES.

A TLS cipher suite is comprised of a key exchange protocol, an encryption protocol, a signing protocol, an authentication protocol, and both parties will agree on a cipher suite which is suitable for both of them so that they can communicate. One of the most common would be `TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256`. Where TLS is the protocol being used, ECDHE is Elliptic Curve Diffie-Hellman Exchange the method used to create a symmetric key. ECDSA is an elliptic curve signing algorithm to ensure the communicating parties are who they claim to be. `AES_128_GCM` is the symmetric encryption scheme used to secure all future messages, using 128-bit keys and using Galois/counter mode, and finally, SHA256 is the Message Authentication Code (MAC) algorithm. Although AES-GCM provides message authentication data, SHA256 is used to authenticate the TLS setup messages.

There have been various attacks employed against TLS over the years, Padding Oracle On Downgraded Legacy Encryption (POODLE), CRIME, BREACH and Heartbleed [71, 80, 81] although the majority of these attacks were the best suited to computer browsers, certain attacks would be applicable to IoT devices, for example, the POODLE vulnerability took advantage of a feature in TLS to downgrade the version employed. This was achieved by a man-in-the-middle attack blocking all requests until SSL 3.0 was automatically employed. The communication between the client and the server is still encrypted but as SSL 3.0 uses AES CBC mode and uses Mac-then-Encrypt (Section 4.1.2). Therefore, the attacker can modify the ciphertext and systematically vary the input to the decryption function to decipher the message one byte at a time [80].

2.5 Entropy Generation in Computing Devices

Entropy is the amount of randomness collected by the system for the generation cryptographic keys to encrypt and sign data; therefore sufficient is required to make these keys statistically unguessable and ideally unique per device [82, 83]. A message encrypted with a key of all zeros would appear to be suitably encrypted, however systematically trying every possible key, in a process known as brute-forcing, would quickly find such a key. All types of symmetric encryption are vulnerable to brute force attacks, but on average, a brute-forcing attack would require the attacker to check half the key space to get the correct key. As the AES key space is larger than the number of atoms in the universe, statistically, it would take a very long time to discover a key, if a suitably random number was selected. Another consideration when generating cryptographic keys is related key attacks. If two separate devices are loaded with the same key and then send an identical message, it would be trivial to discern these devices had the same key through only analysing the ciphertexts. If a malicious user had access to such as device, they could then systematically modify the messages their devices sends, until it matches ciphertexts, and then be able to infer what data other devices are sending. Moreover, if a batch of devices were all populated with the same key or keys which follow a determinable sequence, then a determined attacker could extract a couple of keys from devices and work out the pattern of all the keys and predict the keys in other devices. This could be done either by physically breaking into devices, or examining the memory of the hardware.

How the system generates the key should also be considered. If a key is seeded through an eight-character password, it would be much simpler to brute force the password rather than the AES key. During the NotPetya virus [84] outbreak several years ago, which crippled Maersk the shipping giant, Maersk asked Microsoft for assistance in attempting to restore Maersk's crippled systems, which had been encrypted with ransomware. Microsoft were able to decrypt one of the encryption keys within 23,000 compute hours [85]. They were likely able to achieve this due to a bug in the implementation or brute forcing a BitLocker password. This was the equivalent complexity cost of about 200 dollars at the time of writing, which demonstrates that if a key is seeded from a weak password, the encryption can be brute forced through guessing the seed.

This shows that suitable entropy is required in order to generate suitably random keys, normally strong cryptographic keys are generated from random data sources, such as the inherently random nature of human interaction with computers, or other nondeterministic events. One such popularised example is pointing a camera at a wall of lava lamps. This results in an entirely random image [86]. As simple IoT devices are generally single purpose, they have very little variety in their scheduled tasks and therefore do not have sufficient randomness to create strong keys.

2.5.1 Existing Solutions for Entropy Generation

Desktop computers have many possible sources of entropy. The Linux kernel, for instance, uses Media Access Control (MAC) address, mouse tracking, keyboard timings and disk times [87, 88]. Smartphones build on top of Linux's entropy generation, but it has also been found that sensory inputs, such as accelerometers, gyroscopes and microphones can all be used to expand the entropy [89, 90]. Furthermore, smartphones run multiple variable processes making the whole system less predictable. Combining these sources allows for the generation of robust unpredictable keys. However, employing the same technique in low-cost IoT devices is more challenging; these devices generally run a single program and have minimal user interaction, i.e., no keyboard or mouse [83].

2.5.2 Proposed Solutions for Entropy Generation in IoT Devices

Noting this issue, research was conducted into which existing solutions which had similarities. One such similarity was found during the boot process of embedded devices, this is a sequential task, therefore, little randomness is introduced [82], similar to how basic IoT devices operate. Therefore, solutions for entropy generation during the boot process were investigated, and research was discovered which looked at less deterministic hardware components such as DRAM decay, and phase-locked loop latency, both of these are commonly found in MIPS, ARM and AVR32 processors, and they successfully generated sufficient random data during the boot process [91]. Nonetheless, the focus of this thesis' work has been on small, low-cost low-power single-purpose devices. The MIPS, ARM and AVR32 processors, with their ability to run full operating systems, are overpowered for these applications and are significantly more expensive than alternatives such as the MSP430 which are typically used in these small IoT applications.

Researchers at Stanford University successfully generated random numbers using basic hardware components costing \$1.25, the use of basic components also makes it easy to understand and therefore auditable [92]. Furthermore, researchers investigated employing an accelerometer to generate random data [93]. The accelerometer was suitably resistant to external influence and could generate 128 bits of random data in 1.5 seconds even when perceived to be stationary. However, even this low hardware cost matches or exceeds the cost of the of low-cost microcontrollers, effectively doubling the overall cost of the unit.

Long Range Wide Area Network (LoRaWAN) wireless sensor devices experienced comparable issues of single-purpose devices not being suitably complex to generate entropy, but rather than adding additional hardware; the governing body decided to use the antenna as a sensor. Using the antenna, the received signal strength would be measured and used to generate a random nonce [94]. However, researchers have discovered that by jamming or otherwise influencing the signal strength, the nonces become more predictable as demonstrated by researchers at the University of Padova [95]. This indicates that using influenceable sensors, solely as input for random number generation would generate insecure keys. Therefore, using influenceable sensors was discounted as a solution.

Hardware Security Module

High utilisation servers are tasked with terminating thousands of TLS connections per second, and to assist this, HSM are used. These devices feature custom hardware specifically designed to execute cryptographic instructions. Therefore, their speed and energy consumption is a substantial improvement over software implementations. Accordingly, these devices are commonly employed for high traffic websites as they decrease the time required to compute cryptographic numbers, and setup secure connections. Moreover, they offer a secure environment to store cryptographic keys without fear of an attacker receiving the key. This technology has been scaled down for IoT devices where custom Application-Specific Integrated Circuit (ASIC)s have been made with circuits for enabling asymmetric cryptography, secure key storage and signature generation and verification, making it possible for microcontrollers to engage in various cryptographic systems which would otherwise be computationally expensive. However, adding this type of chip into the design of an IoT device would increase the price, thereby making it unsuitable for the design criteria.

Physically Unclonable Function

Fingerprint sensors have become increasingly common as a convenient way to open smartphones without having to enter a long passcode while keeping the device secure. This security paradigm relies on the property that no two human fingerprints are exactly identical. Even identical twins who share the same DNA do not have matching fingerprints, as fingerprints are developed in the womb [96]. Likewise, a Physically Unclonable Function (PUF) is a piece of hardware which is manufactured in the same way but due, to microscopic differences in how atoms settle a device, will always have unique behaviour. Using this property, a sequence of bits can be issued to the chip, which applies its function to the input bits and outputs a different sequence of bits known as a response. No two chips, when presented with the same challenge, will generate the same response. This process is known as challenge-response pair. PUF circuits are commonly implemented using different Complementary Metal Oxide Semiconductor (CMOS) technologies, most prominently as an ASIC and less commonly on field-programmable gate array. This type of circuit is commonly used for either authentication or secure key generation. Authentication can be achieved through the manufacturer creating the device and then issuing thousands of challenges to it while recording all of the responses. The device can then be deployed, and once the server wishes to authenticate the device, it would send one of the pre-recorded challenges and check if the correct response is returned. However, each challenge can only be used once as an attacker could monitor the response of a previous challenge, and issue the correct response. The PUF could also be used as random number generator to seed an AES key, or a plaintext challenge can be issued to the PUF, which will generate a fixed response, known to both the client and the server through the predetermined tables, therefore allowing both devices to communicate using the response as a secret key.

2.6 Message Authentication

Authentication of a message is important for two reasons. Firstly, to confirm who the message is from, and secondly to ensure the message has not been tampered with before reaching its intended recipient. Although encryption may seem like the ultimate goal of security, authentication also has a considerable role to play as, without authentication an encrypted message can be modified without the recipient's knowledge. This type of attack is particularly concerning if the attacker has knowledge of the message structure, as they could craft malicious messages without requiring the decryption key. To prevent this type of attack messages include a deterministic value, using a cryptographic hash functions, which the communicating parties can verify.

2.6.1 Cryptographic Hash Functions

Cryptographic hash functions input variable length data and output an ideally unique fixed-length response, known as a hash. The standard hashing algorithm recommended by NIST is the Secure Hashing Algorithms (SHA). There have been multiple variants of SHA. The first iteration has since retroactively been called SHA-0, and was found to be flawed by Chaband and Joux in 1997 [97], whereby multiple input values would map to the same hash value. SHA-1 the subsequent family of hashing algorithms was also found to be vulnerable to such an attack [98]. Moreover, in 2017 researchers at Google & CWI were able to craft two distinct PDFs which hashed to the same value, a full targeted collision [99]. Allowing specific hashes to be created to invalidate any authentication provided to the message. Subsequently, additional algorithms have been created within the Secure Hash Algorithm family, SHA-2, and SHA-3. In order to be cryptographically secure, hashes must comply with four properties.

1. Any alternation to the input of the function must create a significant change on

the output of the function.

2. It must be infeasible to determine any characteristic of the input from the output.
3. Two non-identical inputs should map to two distinct outputs.
4. Any indeterminate length input must be converted into a fixed-length output in a computationally efficient manner.

Each of these conditions must hold true to provide message authentication without leaking message data. If an attacker knows the hash of a message, it must be challenging to discover the message itself. It must also be sufficiently difficult to two find two distinct inputs which map to the same output, otherwise, a message using a cryptographic hash to preserve its integrity, could be altered and a new hash generated which would match the expected hash. In a similar way if any character from the input can be determined from the output then systematically the input could be modified until it matches the expected output.

When cryptographic hash functions are applied to messages, they are known as a MAC. The MAC is encoded within the transmitted message, and the receiver using the same hashing algorithm calculates their own MAC using the received message as an input. The received MAC and calculated MAC are then compared. If the values match then the message has not been tampered with. Otherwise, the message will be discarded.

2.6.2 Hash-based Message Authentication Code

As MAC algorithms are publicly available, attackers must be prevented from capturing a message, modifying it, and then creating a new hash with the modified data and attaching it to the message. This can be prevented through including secret data in the function that only the transmitter and the receiver know. This is commonly done using cryptographic keys, one such example is Hash-based Message Authentication Code (HMAC), as shown in Fig. 2.9, where a secret key is exclusive OR'd with repeating bytes of 0x36, and the hash is taken of the result. The result of this is then concatenated with the secret key XORed with 0x5c, and the result is hashed again. Therefore, it is required to have both the secret key and the message to calculate the HMAC correctly.

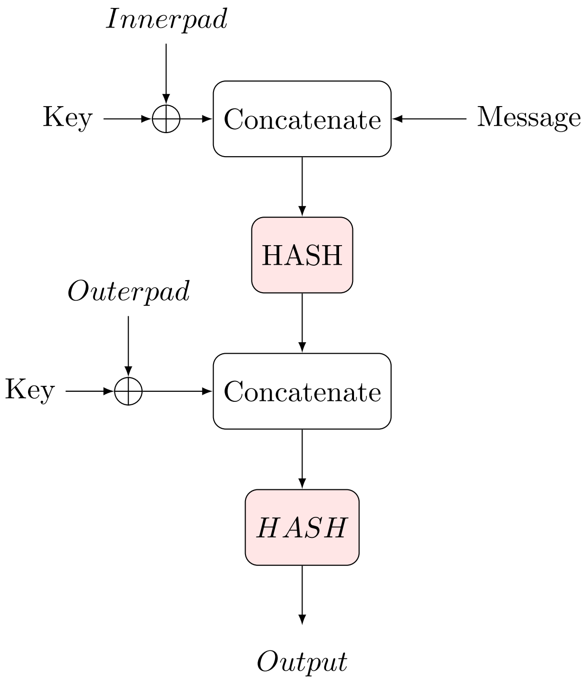


Figure 2.9: HASH Based Message Authentication Code (HMAC)

2.6.3 Cipher-based Message Authentication Code

Cipher-based Message Authentication Code (CMAC) as the name implies, uses a cipher, specifically AES as a building block to create a MAC. The inputs are an AES key, a message of variable length and its output is a fixed-length string which can be used to verify the integrity of the message as the MAC can only be calculated with access to the key. As shown in Fig. 2.10 the message is split into blocks and each encrypted, where the message does not match a multiple of the block size padding is added to the end of the message, and the output of each block encryption is passed into the next encryption operation as a parameter, and a subset of the final encryption output is then returned [100].

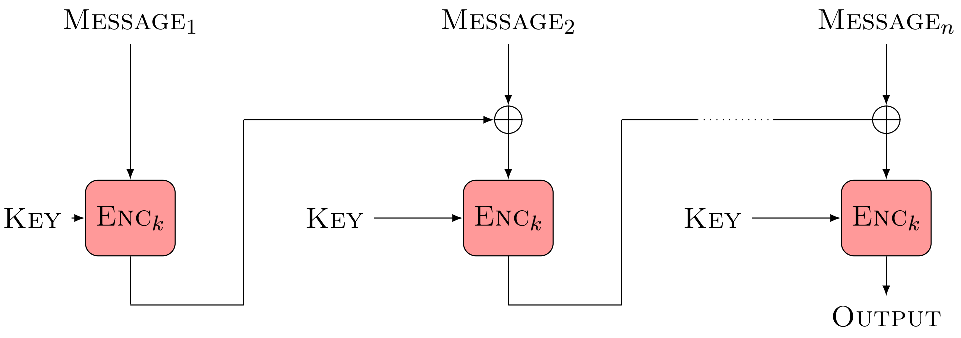


Figure 2.10: Cipher-based Message Authentication Code (CMAC)

2.7 Securing IoT Devices with Other Means

2.7.1 Blockchain

Blockchain has been widely regarded as a solution to IoT communication issues; it enables smart contracts, decentralises infrastructure and enables append-only data structures [101–104]. A blockchain message consists of a payload and a header, where the header contains a hash of the previous block, and the Merkle root[3] of all the transactions within that block. As each new block contains the hash of the previous block, this creates a verifiable trail back to the initial block. This makes it impossible to reasonably tamper with any data which has been committed to the chain. An attacker attempting to alter a previous block would have to also alter all subsequent blocks, which means that the longer ago a value was committed, the harder it would be to alter it. Moreover, the system is constructed of a distributed network of computers, all of which continually advance the chain through executing some action. The most common action is proof-of-work, this is the system that Bitcoin uses, and requires computers to hash values to find a specific result, as this result is statistically unlikely, on average, many operations are required to find the result [105]. Moreover, this difficultly is dynamically scaled to make each result take roughly the same time. The advantage of this is, if there is a sudden increase in members of the network the challenge increases. Therefore, even throwing vast computing power at the blockchain

[3] Merkle root is the hash of all of the nodes associated with a node, so that if any node is modified the Merkle root would be modified

would not allow an attacker to easily change previous blocks. This means provided a majority of the network are honest, an attacker would not be able to change values.

One of the reasons that blockchain is considered for IoT devices is that there is predicted to be to be billions of IoT devices, and all of these devices would have to communicate with a series of central servers [105]. This would require a significant amount of server power and therefore cost. Instead, blockchain can be used, and each device just has to connect to its peers and data will be shared amongst each other without having to rely on a centralised server, potentially flooding it with requests, or having to maintain vast architecture to cope with the demand. Blockchain is also useful for where data should be kept forever. As once the data is added to a chain it will survive as long as the blockchain does, and with a distributed network of interested parties this may last significantly longer than a company would maintain its architecture, for example Bitcoin is currently 11 years old [106]. On the other hand, as any data stored on the blockchain must be stored on each node, and as the data in the chain increases as does the cost to store all this data for each participant [107].

However, for the computationally-constrained devices referred to in this thesis makes blockchain an unsuitable technology [105]. Proof-of-work systems are too processor intensive, while proof-of-stake systems, another Blockchain design, is both too processor and memory intensive [105, 108]. A recent paper by Deepak et.al, suggested a proof-of-authentication scheme specifically for IoT devices which employs a model of using trusted nodes to verify other nodes in the network and increasing their trust using message signatures [109]. However, Deepak et.al, found that Raspberry Pis which have a 62 times more clock speed, and 62500 times the memory available [110] than constrained IoT based microcontroller devices [111], took 3 seconds to process a block on a closed system. Therefore, making such a solution unsuitable computationally-constrained IoT devices as even a 3-second latency for home automation devices would be considered unresponsive [112, 113]. Moreover, Blockchain is primarily a verifiable decentralised storage mechanism, where the data itself would still need to be encrypted before being stored in the chain, otherwise the data would be publicly visable [108]. Other solutions have been proposed through offloading the required calculations to a local third party [101], meaning that low powered IoT devices would not have to deal with the transactions themselves. However, this would require the use of a third-party device, making this unsuitable for this use case.

2.7.2 Software Defined Networking

Another solution which was considered for securing low-cost devices was analysing traffic at network-level. Vilalta et al. suggested using intelligent network monitoring to detect traffic anomalies and then block such traffic [114]. This type of system would be particularly useful to prevent IoT devices joining botnets, or executing denial of service attacks [115–118]. Sivanathan et al. continued this idea to investigate flow analysis rather than packet inspection and found that it was achieved by setting rules on the domestic router to mirror certain parts of traffic to an offsite analysis engine which would then be able to detect any unusual or suspicious activity [119]. By placing the intelligence engine offsite, users would not have to have additional hardware in their house, although it would require a subscription to an external service, thereby removing the additional manufacturing cost, but adding a service cost.

2.8 Low-Powered Wide Area Network Technologies

A field of technology which has also considered the issue of enabling secure communication in constrained devices is LPWAN. LPWAN is a wireless technology best suited for applications that require long-range communication in power-constrained environments, with potentially thousands of connected devices. End-devices in an LPWAN are typically battery-powered, which makes deployment notably easier, as there is no requirement to find an external power source. These devices are typically low-cost and are characterised as low throughput, mainly idle devices that can receive limited downstream traffic. All of these characteristics make LPWAN devices suitable for remote sensing and monitoring as they are low-cost, and can operate independently.

This thesis will examine three LPWAN technologies, Sigfox, LoRaWAN and Narrowband Internet-of-Things (NB-IoT). Each of the technologies has its own advantages and disadvantages. Sigfox and LoRaWAN operate in the unlicensed industrial, scientific and medical (ISM) bands of 868 MHz in Europe, and 915 MHz in the United States. NB-IoT, on the other hand, operates in licensed spectrum, in pre-existing Long-Term Evolution (LTE) mobile bands. All of these technologies rely on a similar architecture of devices connecting to a base station which covers a particular geographical region and passing data back to a central point to be issued to the data sensor owner.

2.8.1 Sigfox

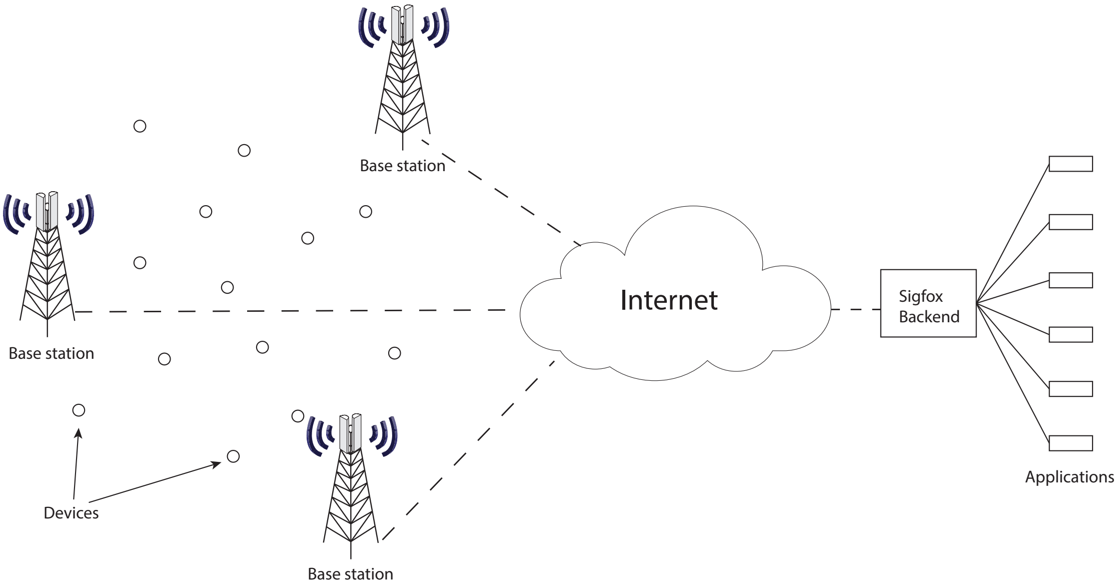


Figure 2.11: Sigfox Network Architecture

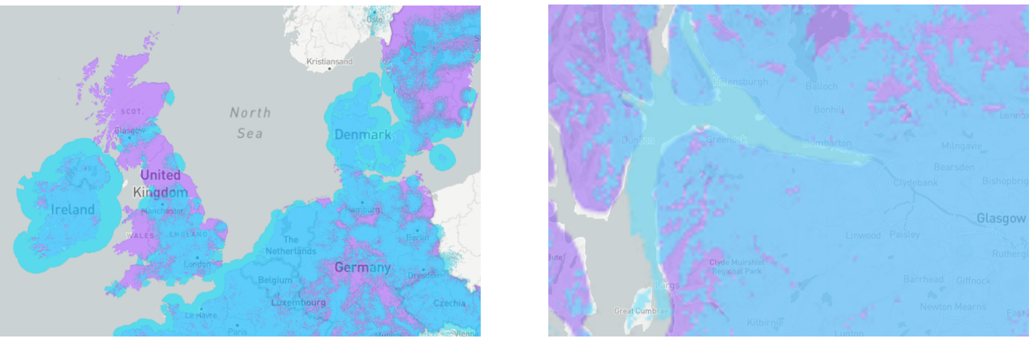
Sigfox has been designed to follow the traditional mobile subscriber payment model. Sigfox, as a company, supplies all of the required backend infrastructure (Fig. 2.11) for the network to operate and allows devices to be attached to its network for a monthly fee. Once a device is registered, received data from the device is made available to the user through Application Programming Interface (API) endpoints or existing integrations. One significant difference Sigfox network provides over a

General Packet Radio Services (GPRS) network is that is costs less. Radio spectrum licenses are prohibitively expensive, at ≈£20 million per MHz [120]. Sigfox transmits in unlicensed spectrum therefore avoiding any spectrum license fee. Sigfox further manages to reduce deployment costs as it uses very low data rates, meaning that a single site can provide comparatively more coverage than a comparable GPRS before becoming saturated, which in turn reduces the required number of sites. The operational costs of running a Sigfox network are also less than a GPRS or LTE network as the architecture is simpler. These cost-saving measures result in significantly lower monthly costs for consumers compared to a mobile subscription. This is particularly important for industrial use cases as there may be thousands of devices. To connect a device to the Sigfox network a user must program the required credentials into the module, and then register the device on the online portal. Sigfox uses a custom lightweight protocol to communicate messages, only requiring 14 bytes of overhead, for message authentication and addressing per message. LTE a comparable wide area network communication technology requires 40 bytes for an Internet Protocol (IP) header, and must first initiate communication requiring an attach request which is roughly 270 bytes [121]. As each additional byte requires a little bit more energy to be transmitted, reducing the number of bytes as much as possible increases the battery life of devices.

One of the downsides with Sigfox is the lack of coverage. Figure 2.12 shows the current Sigfox coverage in the UK. The lighter/blue areas of the map denote areas of coverage, whereas the darker areas indicate a lack of coverage. As shown in the figure, the overall coverage is sparse, and there is a total absence of coverage outside of major population centres. The gaps in coverage become even more apparent upon focusing on a particular area. This makes Sigfox unsuitable for a nationwide deployment in most countries, however it would operate well for urban use cases.

2.8.2 LoRaWAN

LoRaWAN is another LPWAN communications protocol which developed a straightforward environment for devices to communicate back to a control centre.



(a) UK Coverage

(b) Around Glasgow Coverage

Figure 2.12: Sigfox Coverage Map - [122]
The Lighter/Blue Areas of the Map Denote Areas of Coverage

This is achieved by an end-device transmitting on the 868 MHz frequency to a LoRaWAN gateway, which then forwards messages to a remote server, either via 4G mobile communications or via a wired IP network. Multiple gateways can be deployed to pass messages back to the same server in a star-of-stars network topology, creating a robust network where several gateways can receive and pass along the message. The primary distinction between LoRaWAN and Sigfox is that organisations can deploy their own private LoRaWAN networks while Sigfox is orchestrated by a central company per country. In LoRaWAN commercial providers exist, who provide coverage and infrastructure for a monthly subscription fee, but individual organisations can also create private networks or hybrid networks using public infrastructure where available and private networks to fill in coverage gaps.

The architecture requires a packet forwarder, a network server and an application server, although the last two can be co-located on the same machine as shown in Fig. 2.13. The packet forwarder is installed on the LoRaWAN gateway which receives LoRaWAN packets on the air interface, and encapsulates them into User Datagram Protocol (UDP) packets and sends the message along to the network server. The network server authenticates the message and if it is valid passes the message to the application server, which decrypts the packet Finally, the application server publishes the message to Representational State Transfer (REST) API endpoints and on Message Queuing Telemetry Transport (MQTT) for users to integrate their own systems.

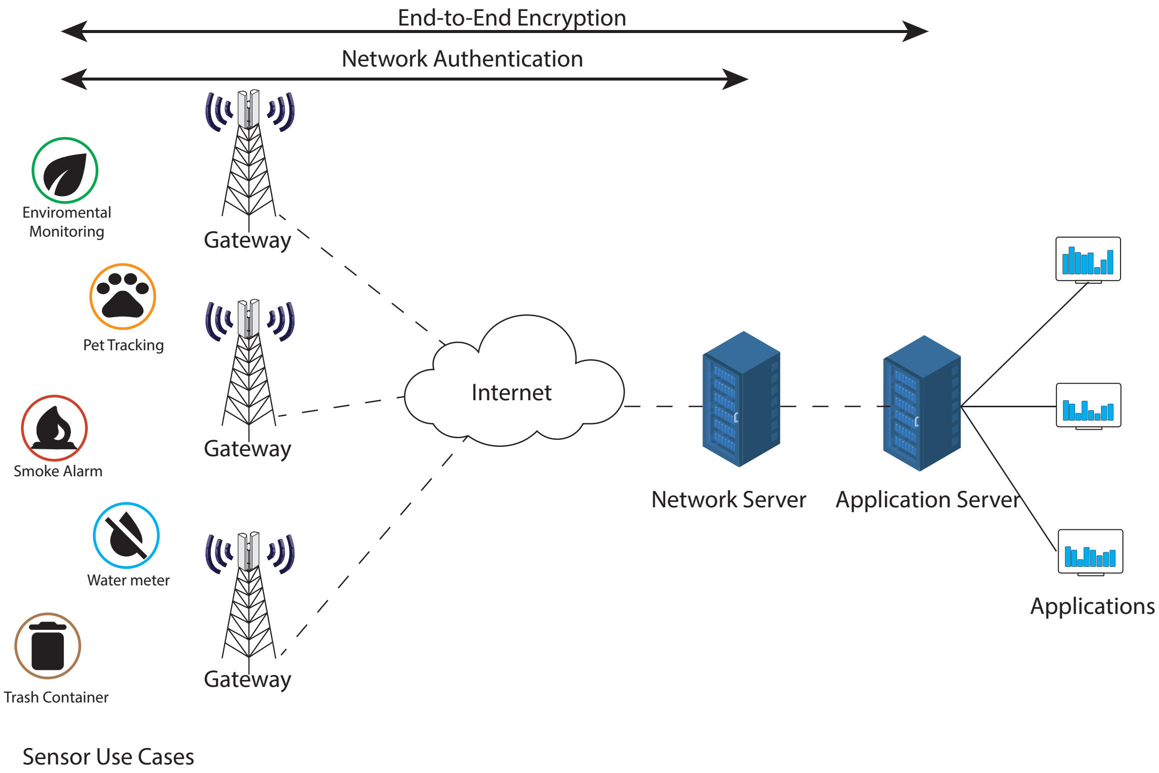


Figure 2.13: LoRaWAN Architecture

As LoRaWAN operates in license free Industrial, Scientific and Medical (ISM) bands, there is the potential for interference as this band used by various short-range devices [123]. To account for this LoRaWAN employs a technology called chirp spread spectrum which alters the frequency while transmitting, making the system robust to accidental interference. Additionally, an end-devices also change their broadcast channel in a pseudo-random fashion for every transmission [94], further improving the robustness to noise. LoRaWAN's built-in ability to dynamically change the broadcasting spreading factor means that if there is channel interference, the per-symbol bandwidth is increased to ensure reliance. These features result in a calculated link budget of 154 dB [124]. However, like many radio technologies, LoRaWAN can still be jammed by a local attacker transmitting on high power across the entire frequency band. [125]

In the LoRaWAN security model, there are two security layers: the network layer and the application layer. The network layer is responsible for end-device message authentication using a CMAC and a 128-Bit AES secret key shared between the end-device and the Network Server to prevent message tampering. The application layer is responsible for ensuring data privacy, using a separate 128-bit AES secret key — called the Application Session Key (AppSKey) — which is shared between the end device and user application. The payload is encrypted using AES Counter Mode (AES-CTR) mode. There are currently several versions of LoRaWAN [94, 126, 127], but the most common versions are 1.0.2 & 1.0.3 [128], therefore these versions discussed in this thesis.

LoRaWAN devices are activated in two ways, activation by personalisation (ABP) and over-the-air-activation (OTAA). ABP programs two keys directly into the device for the application and network layers, where OTAA programs a master key, Application Key (AppKey), into the device which is then used to derive both the Network Session Key (NwkSKey) and the AppSKey. This establishes new keys on each network join request, increasing the robustness of the communication. As OTAA is both more secure and more common than ABP [95, 129], therefore this mode discussed in this thesis.

The network layer is responsible for authenticating messages, by including a Message Integrity Check (MIC) field generated using the CMAC algorithm within every message. The MIC sums together all of the bits in the message and applies a hash function to create an ideally unique but deterministic fingerprint. If a single bit in the message is changed, it would dramatically alter the generated fingerprint. Therefore, once the recipient receives the message, they can check the validity of the message. The application layer, on the other hand, is responsible for ensuring data privacy, where the AppKey encrypts the payload of the message. To successfully join a LoRaWAN network both of the session keys are required, together with an Application Identifier (AppEUI), which tells the device where to route messages, and a Device Unique Identifier (DevEUI) so that the receiver knows which device the message has come from [130]. The AppEUI is commonly pre-programmed into the device by the manufacturer, however, the JoinEUI and AppKey would need to be programmed into the device prior to deployment.

2.8.3 Narrowband-IoT

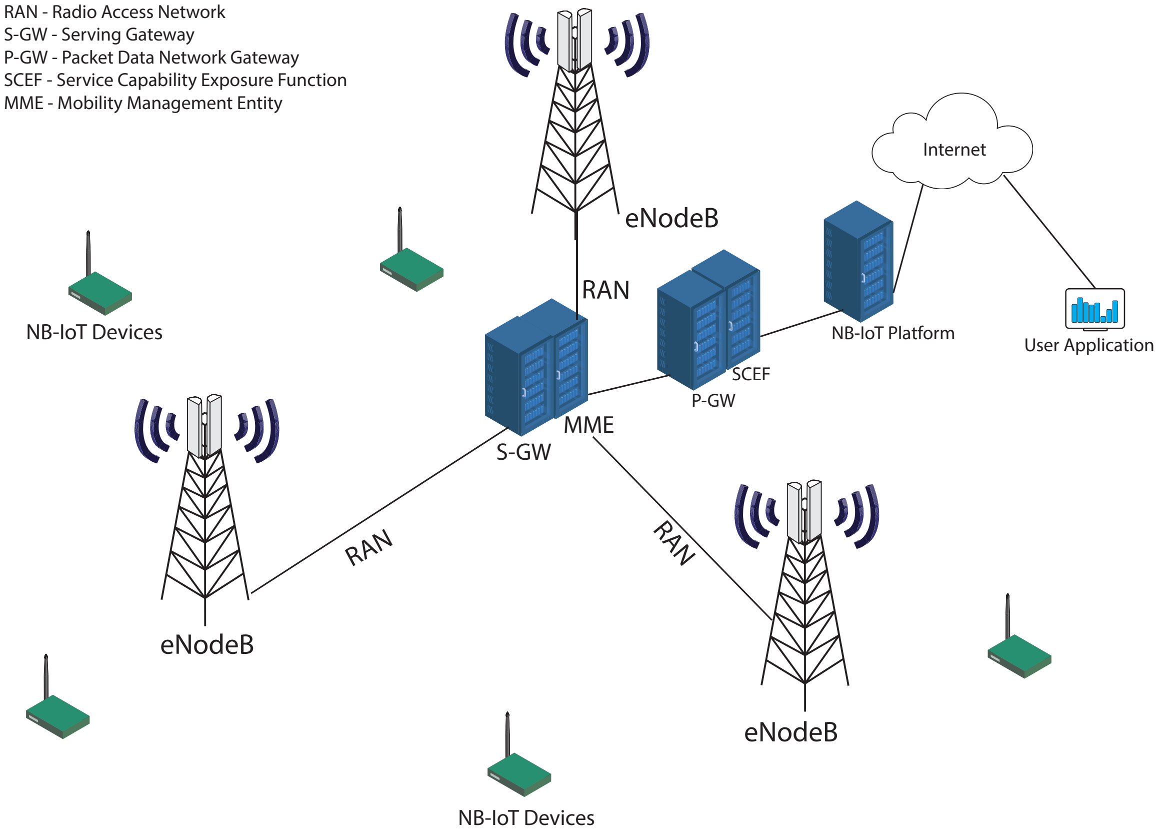

RAN - Radio Access Network
S-GW - Serving Gateway
P-GW - Packet Data Network Gateway
SCEF - Service Capability Exposure Function
MME - Mobility Management Entity
eNodeB
RAN
Internet
NB-IoT Devices
SCEF
NB-IoT Platform
User Application
P-GW
MME
S-GW
RAN
RAN
eNodeB
eNodeB
NB-IoT Devices

Figure 2.14: Narrowband IoT Network Architecture

NB-IoT is a further LPWAN technology, which differs from both LoRaWAN and Sigfox in that it utilises existing mobile network infrastructure and transmits within licensed frequency bands. It also retains functional similarities with both LoRaWAN and Sigfox, where both public and private networks can be used, however, given the complexity of acquiring licensed spectrum required for NB-IoT, it is likely that most deployments would prefer the subscription model. Cellular based system such as NB-IoT employ the star topology as LoRaWAN and Sigfox, but requires more network components as shown in Fig. 2.14.

NB-IoT has a channel width of 180 kHz and 10 kHz of guard bands. Therefore, a single Global System for Mobile Communications (GSM) carrier is a like for like replacement with a NB-IoT cell without requiring reframing[4], and can be done on a cell by cell basis. To this end, NB-IoT can be deployed in three distinct configurations standalone, guard band, and in-band, as shown in Fig. 2.15. Standalone is where the system has its own decided spectrum, commonly either previously used or created through reframing. Guard band is where NB-IoT is deployed within the gaps between existing LTE spectrum allocation, and in-band with an existing LTE carrier. NB-IoT also provides several advantages over a GSM deployment: it increases the receiver sensitivity by 20dB, allowing it to travel further and penetrate hard to reach areas. It also uses significantly less energy than LTE or GPRS, giving a significant battery life of an estimated ten years using AA batteries [131]. A single NB-IoT channel can also support at least 52,000 devices [132], given the low data rates and short bursts of traffic common in NB-IoT systems.

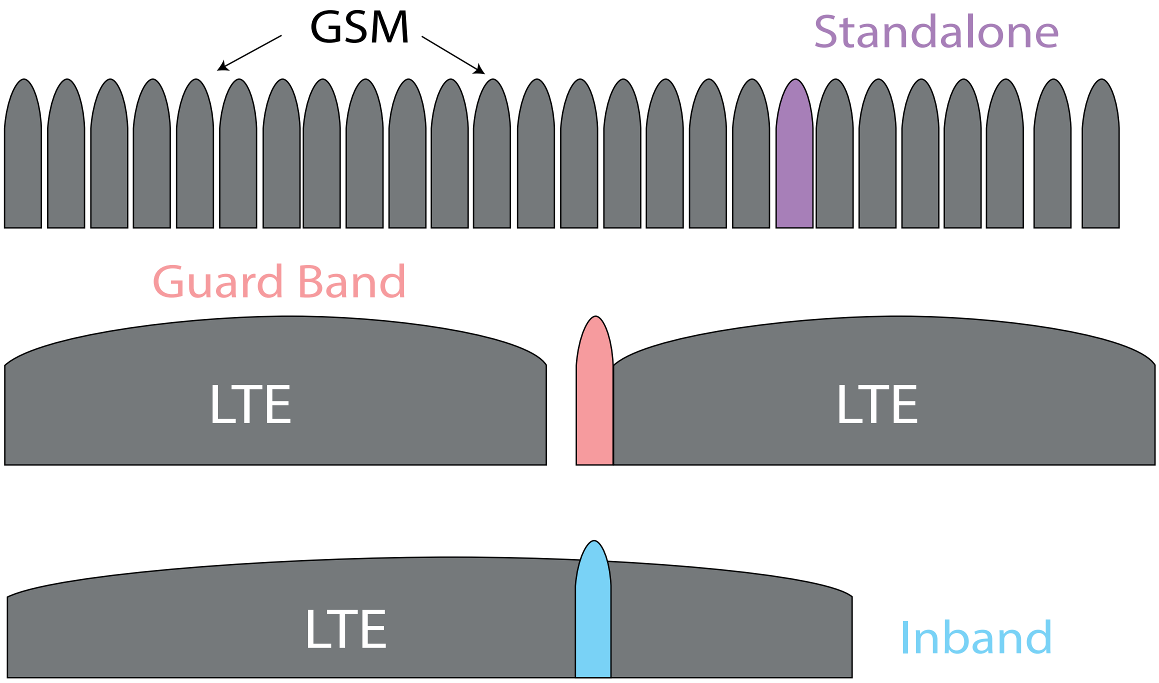


Figure 2.15: Narrowband IoT Configurations

Another cellular-based technology is LTE Machine Type Communication (LTE-M), which is specifically designed to allow machine-to-machine communication. NB-IoT is

[4] The process of reallocating spectrum

designed to have a throughput of kbps, while LTE-M using a significantly larger channel width of 1.4 MHz can achieve 1 Mbps, although this wider bandwidth requirement means the design is less flexible. Similarly, to NB-IoT, LTE-M also supports a receiver sensitivity of 156 dB, allowing it to get into otherwise challenging areas [133].

The propagation of both LTE technologies, NB-IoT and LTE-M depend on the frequency band the system is deployed in. Whereas LoRaWAN and Sigfox are deployed at a fixed frequency, LTE can be deployed in various frequencies, with lower frequency bands propagating further. However, only certain bands are feasible as it also requires an existing ecosystem of devices as custom making devices would be prohibitively expensive for deploying connectivity solutions. The most common cellular bands in the UK are band 3, band 7, and band 20 which operate at 1800 MHz, 2600 MHz and 800 MHz respectively. These bands have significant commercial value to mobile network operators, with each allocation being auctioned on a 10-year deal. When this cost is added to the cost of infrastructure, equipment and installation, mobile operators struggle to make a suitable return on investment installing mobile infrastructure in rural locations, therefore coverage is lacking in rural areas.

To attempt to alleviate this problem there have been various United Kingdom (UK) government interventions to expand rural mobile coverage [134–136]. Additionally, the UK mobile regulator Office of Communications (Ofcom) recently announced a scheme to allow organisations to apply for low-power transmission licenses for 3.3 MHz in band 3 for a nominal fee for a limited geographical area. The aim of the scheme was to reduce the cost of deploying a mobile network so that communities and organisations could setup their own private mobile networks without having to acquire spectrum. This 3.3 MHz could be used to deploy a 1.4 MHz regular LTE cell for voice calls and traditional mobile communication, alongside a 1.4 MHz LTE-M cell to support machine-to-machine communication, and an NB-IoT cell for many low rate communication sensor devices as detailed in Fig. 2.16. This type of architecture is more flexible than solely LPWAN technology as this would be able to have higher data throughputs for devices which require it. The trade-off for these benefits is that there is an increased number of required network components. In addition to that custom made Subscriber Identification Module (SIM) cards, and spectrum licences would be required.

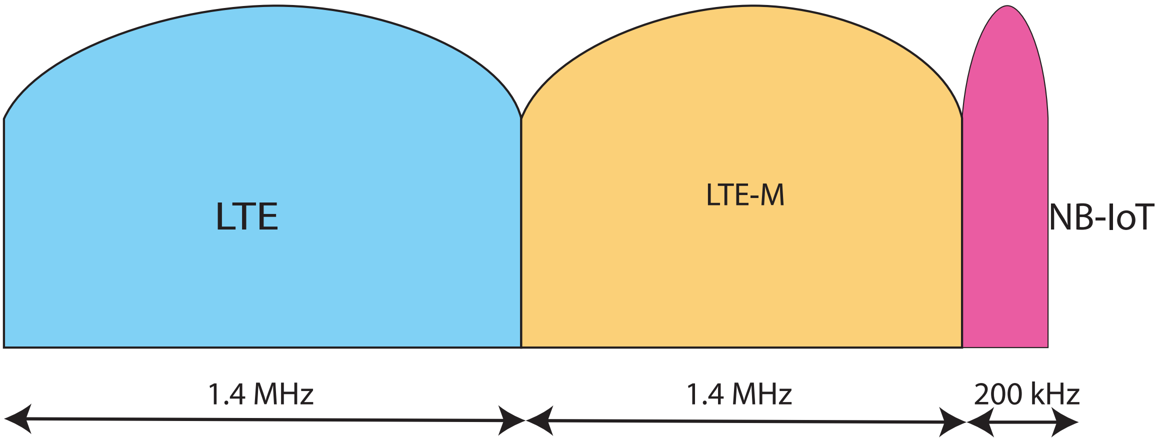


Figure 2.16: Channel Spacing NB-IoT Configurations

2.9 Threat Modelling

Commonly there is a trade-off between security and convenience, increasing security features often makes it harder to interact with a system for users. Gene Spafford, famously said "The only truly secure system is one that is powered off, cast in a block of concrete and sealed in a lead-lined room with armed guards - and even then I have my doubts." [137]. A system must then sacrifice security to be usable. Therefore, the predicted aggressor determines the level of security required. Different attackers have different resources available to them, some attackers can employ physical attacks, while others are limited to remote attacks. An attacker only has to find one mistake, while defenders must protect all potential possible attack avenues. This has been demonstrated with attempts to break AES. Once brute forcing attacks were no longer suitable due to the increased key size over DES. The implementation itself through using a reduced number of rounds was attacked so that if an defender had implemented a mistake this would be caught. Next, the execution time and energy consumption was considered through side-channel attacks. Security attacks never get worse, they only get better.

Various models have been developed for identifying potential threats. Microsoft created one of the most popular threat models called STRIDE [138], which stands for Spoofing, Tampering, Repudiation, Information Disclosure, Denial of Service and Elevation of privilege. Each of these potential attack vectors is described below. This model is often combined with another model, DREAD [138] — Damage, Reproducibility, Exploitability, Affected Users and Discoverability. This latter model attempts to work out how likely an attack is, and what would be the resulting damage if it was to occur. Some incidents may not be very likely however if they were to happen it would be disastrous, on the other hand, there may be minor inconveniences, but if they happen repeatedly, it could result in a cumulative result, causing a larger issue.

2.9.1 Device Tampering

IoT devices are designed to operate unattended for the majority of their lives; therefore, physically tampering with devices should be considered. Depending on the use case physical security measures may also be considered, but for devices where this is not practical the overall IoT system should be tolerant to individual devices which may have been tampered with. It is also important to consider that this type of attack is less scalable, provided a link between devices cannot be found such as using the same key on every device.

2.9.2 Spoofing

The act of spoofing is using another user or devices credential's in order to gain access to systems which are otherwise inaccessible. The credentials can be extracted from devices through physical inspection/device tampering, or taken passively through recording the devices communications. A common attack is through phishing emails which trick users into providing their credentials [139]. Once the credentials have been captured the attacker can pretend to be that user and insert false information into the system through programmatically changing values. The attacker could also use the credentials to gaining access to privileged information.

2.9.3 Information Disclosure

Devices should be careful not to reveal information to users or devices not entitled to see that information, both through a remote attack and through physical inspection of the device. This can be prevented by validating you are communicating with the correct party.

2.9.4 Privacy Breach

Systems can inadvertently leak private information through metadata or the frequency of communication. The messages themselves may be encrypted, but the presence of the message would still be visible. Therefore, the attacker could infer the message through knowledge of the device classification.

2.9.5 Denial of Service

Systems can be made inoperative through bombarding the device with requests, manipulating its software, theft of the device, or disrupting the communication channel. Therefore, systems must be made resilient to this type of attack to be able to continue operating when a malicious actor is performing undesired actions.

2.9.6 Elevation of Privileges

Users are often separated into having different levels of different privileges. Certain users might be able to control a whole range of devices while other users can only control devices in specific locations. This has the effect of limiting those who can influence specific device functions resulting in increased security. An example might be IoT lights within a hotel, where the cleaning staff should be able to control the lights to clean all the rooms in the hotel when required. However, a hotel would not want certain guests controlling other guest's lights. The elevation of privileges models allows the system designer to consider ways which a low-privileged user could increase their privileges to commit unintended functionality.

2.9.7 Signal Injection

Signal injection is the process of introducing false data into a system. An attacker could do this without affecting the IoT device itself at all. For instance, if the device is a temperature sensor the attacker could affect the physical environment surrounding the sensor to make it report artificially high or low readings. The device itself could also be moved away from the device it has been tasked with measuring, defeating its purpose.

2.9.8 Side Channel Attacks

An additional attack avenue is side-channel attacks which target the physical characteristics of a device, such as the timing, power consumption, and traffic, which can all reveal information, such as the encryption key used. This was demonstrated in a cache timing attack [140].

This threat model helps system designers consider the various attack vectors a device could be exploited with. The possible entities which could target a device are the user, the manufacturer and an external actor. A user may wish to exploit unintended functionality in the device, and use it as a means to attack over devices in the system, or use the device as a connection into the manufacturer's infrastructure. Likewise, the manufacturer may benefit from a compromised device by implementing unadvertised features of the device to secretly gain additional data about their consumers, either for advertising purposes or to build corpora for machine learning algorithms. Finally, some external actors may wish to compromise devices to gain personal notoriety, wealth or to leverage equipment to attack another organisation.

2.10 Summary of Technical Background

As Leverett et al noted, the digital and physical worlds are converging with the increase of IoT devices. As these devices increase it is critical to secure them [13]. However, not all devices are capable of running security features [15], or security has not been implemented through choice [29, 141]. This has resulted in various devices being compromised [1–11]. Prior research undertaken to address this issue have suggested adding additional hardware [91–93]. However, given the aims of this thesis this approach was not suitable. Sensor devices in particular are low-cost, computationally and battery constrained, but challenges in securely provisioning them has led to the use of insecure credentials [128]. Therefore, this study will investigate how to implement security features on low-cost devices and examine the reasons why security may not be implemented through choice in the following chapter.

Chapter 3

Economic & Policy Background

As the market for IoT devices has been steadily growing, a recent report by the UK Parliamentary Office of Science and Technology [16,142], listed economic drivers as one of the main reasons for poor cybersecurity in consumer devices [143]. The European Union (EU) Agency for Cybersecurity, academics, and industry have all expressed concerns that manufacturers prioritise cost, user experience and time to market over security [144–148]. Part of the reasoning for this concern is that a 2017 survey found that 42% of manufacturers' consumers were unwilling to pay any extra for security, while 28% of consumers were only willing to pay a 1-10% premium [149], which was recently validated by an additional survey in 2019 [14]. This creates an economic constraint for securing IoT devices. This chapter will discuss the impact of economic driven policies on consumers' security.

3.1 A Market for Lemons

The lack of security in IoT devices can be partly attributed to the fact the industry has created a market for lemons. A market for lemons is a thought experiment by the economist George Akerlof [150], which proposed there were two types of cars on the market, *peaches* and *lemons*. *Lemons* refers to the American slang for a defective car, as it would leave the buyer with a sour taste in their mouth. A peach, on the other hand, is considered sweet and therefore a good car. In a market of multiple buyers and sellers there will be a mix of lemons and peaches available. However, car salespeople are not going to outwardly advertise their *lemons* as inferior products, but lower the price of the car to make it a more attractive proposal. This results in the value of a *lemon* being $1000, while a peach may be $3000. A consumer who is unaware of the deficiency of the *lemon*, and therefore unable to differentiate between the two cars, would rationally only pay the average value for all the available cars. In this example, this would lower the price the consumer is willing to pay to $2000, lower than the price of any *peach*. Subsequently, *peaches* would become unsellable as they have been priced out of the market. In order to stay in business, car dealerships which previously sold *peaches*, would resort to selling only *lemons* as the market cannot bear the higher cost of the *peaches*, further perpetuating the situation. Soon there would be only *lemons* available. However, for a market for lemons to occur, certain conditions must be met [150]:

1. Asymmetry of information, as buyers cannot accurately access the value of a product before the sale is made, while all sellers can accurately assess the value of a product prior to sale.

2. An incentive for the seller to pass off a low-quality product as a higher-quality one.

3. Lack of credible disclosure technology for sellers to advertise a potentially superior product.

4. All available products are of similar quantity or the average quantity of products is sufficiently low.

5. Lack of effective public quality assurance and relevant legislation.

3.1.1 Asymmetry of Information

The situation described is remarkably similar to the current state of IoT security [151], where only the manufacturers know what security features they included in the device. Moreover, the manufacturer may believe their product is secure when in reality, it is not. This can occur for numerous reasons. Firstly, a company may include backdoors to their systems, with secret passwords, only known internally, but could be extracted from the devices themselves, and redeployed against other devices. Secondly, companies may believe that obfuscation provides security and unknowingly introduce security vulnerabilities. Thirdly, companies may argue that introducing security features reduces the performance of their system, therefore the company may introduce short cuts to prevent degrading device response time [152]. Finally, the manufacturer may simply not understand what is required to secure a product or the significance of security, and therefore avoid it.

3.1.2 Advertising Lower Quality Products as Higher Quality

There is an incentive for the seller to advertise a low-quality product as a higher-quality one. Similarly to [149], a recent study by the UK's Department for Digital, Culture, Media & Sport (DCMS), found that although consumers listed security as their third most important concern when considering an IoT product, only 59% were willing to pay an extra 5% for the product if it listed security features, falling to only 40% when security would add an additional 10% compared to a competitor's price, Fig. 3.1. This further strengthens the argument for manufacturers having a financial incentive to limit security expenditure.

One popular IoT product is a Google Home voice assistant, which costs $44.03 dollars in raw materials and manufacturing costs [153]. The retail value of the product is $149 representing a 320% mark-up on the manufacturing cost. Part of this overhead would be spent on research & development, advertising and consumer support, but with high-volume low-cost devices such as sensors and switches much of this fixed cost is spread across the high volume of devices, therefore the auxiliary services have a small per device cost. Instead, a significant part will be due to distribution and retail mark-ups. Assuming a conservative 100% mark-up on the raw materials, a 50¢ component costs $1 by the time it gets to a consumer, which for a simple low-cost IoT device would make the product 50% more than a consumer would be willing to spend. This creates an incentive for manufacturers to save that revenue and invest in alternative areas, such as advertisement, features, or paying their

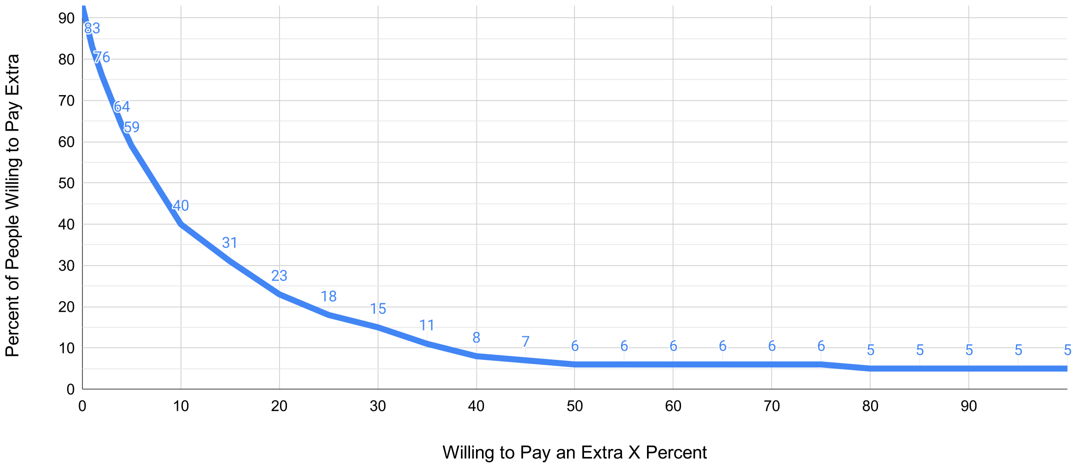


Figure 3.1: Additional Expenditure Consumers Are Willing to Pay for Security Features [14]

shareholders. Moreover, adequately securing devices can reduce the user-friendliness of the device. Even in the best-case scenario of a high-performance device with a simple, easy-to-use interface, usability often gets priority over security. One such case is the iPhone, where Apple implemented TouchID allowing consumers unlock their phones using biometrics, although this method can be circumvented [154], the public perception is that TouchID is more secure [155]

3.1.3 Lack of Credible Disclosure

The third similarity between the market for lemons and the current state of security in low-cost IoT devices is that there is a lack of credible ways for sellers to advertise *secure* devices to potential buyers. If a company believed they made an *unhackable* product, clearly stating and advertising it would make them potentially liable for any potential damages, if their claim was later discovered to be false. Instead, they often use comparisons and familiar phrases to advertise particular security credentials, such as ‘military-grade encryption’, which simply refers to employing AES-256. It should be noted that any encryption algorithm is only as secure as its implementation. For example, even if military grade encryption is used, if the password used to encrypt the device is weak, the security would not hold up. Consequently, a consumer trying to distinguish between two products advertising ‘military grade security’ would reasonably assume both devices had equal quality, when in reality one may have a defective implementation. The outcome of this situation is that manufacturers have no credible way to advertise their devices security credentials.

3.1.4 All Products are of Equal Quality

A fourth parallel is that consumers have a general belief that all products are of equal quality. This is likely because the majority of modern day products consumers can buy are safe. Regulations such as CE marking, which every product sold within the European Economic Area (EEA) must feature, signifies the product has passed health and safety standards [156]. Considering this along with the Semmelweis reflex [157], means that consumers are likely to believe all products are of equal quality, and therefore safe. This argument was reinforced by a recent survey which found that 72% of respondents believe that security features are already built into the devices if they are on the market [14].

3.1.5 Lack of Effective Quality Assurance & Legislation

The final similarity is a lack of effective public quality assurance. Although there is existing legislation on security, as detailed in Section 3.6, it is too recent to have any evaluations published, at time of writing. Furthermore, a general trust in a market tends to lower the importance consumers put in security. This can be seen in numerous situations, as presented in [26]. Although there tends to be an initial outrage, this rarely resulted in long term negative effects for the offending company, particularly when the company holds monopoly or a significant majority in the market [158, 159].

To the author’s knowledge, no IoT manufacturer has offered any kind of consequential damages protection, and this is unlikely to happen as if the system was compromised the potential damage and disruption incurred could be extensive and difficult to attribute to a particular organisation. For example, if an IoT smart lock failed and enabled a burglar to enter a property, potentially incurring thousands of pounds worth of damage, the IoT company would not want to be held liable for the possibly limitless undocumented expense. To counteract this, they would have to become an insurance provider, requiring an itemised list of all high-value items the consumer would want to *insure*. However, the attribution of liability would still be challenging. Was the burglar and the smart lock hacker connected, or was the property previously targeted, and therefore likely the burglary would have happened anyway, therefore removing legal liability from the IoT manufacturer. This is an ongoing area of research for devices which interact with the physical world [160]. However, designing systems to be secure and fail-safe would reduce the likelihood of such a negative occurrence.

Insurance coverage for computerised crime first appeared in 1970s, and software bundled with insurance started in the 1990s, and was subsequently expanded into providing cyber security insurance. This comes in the format of either personal insurance, where an individual is insured or businesses insurance protecting against professional mistakes. The top concern for individuals is an attacker getting access to their banking information, which could lead to identity theft, data loss, and illicit publication of personal data. However, an insurance company would only be able to provide financial relief for money stolen or loss of income, and provide the financial support to rectify identity fraud. Companies can also take out cyber security insurance which primarily protects against the costs of data breach, but most policies will also expect companies to follow certain government guidance for security features, such as not employing default passwords. An example of this is Cyber Essentials [142], which would also reduce a company's potential vulnerability. It should be noted though that after data breaches it is challenging to recapture stolen consumer data, so instead in the case of TalkTalk and Equifax credit monitoring was offered [161, 162].

Marotta et al. found that data breaches were both common and expensive, therefore likely to have high insurance premiums [163]. Paying for insurance would also be a recurring cost for IoT vendors, which under existing payment models would likely be time bound depending on if the company had an continuing source of income, such as later versions of the products. However, it should be considered that as 40% of new IoT products are created by start-ups [164], many of these companies may not remain in the market. As of writing there is no consumer protection legislation for IoT vendors going bankrupt as is common with financial institutions [165, 166]. Therefore, if a company is no longer operational it would leave the consumer on their own with little recourse.

3.2 Challenges of Software Updates

Software updates are a blessing and a curse. As internet connected devices became more prevalent, it has become common for device updates to be propagated through the internet, after a bug has been discovered and fixed, or a new feature is required. This created a methodology of *Move Fast and Break Things*, a term coined by Mark Zuckerberg [167], where software would be created quickly, to make a minimum viable product, and see what works, fixing its potential mistakes later. However, this creates numerous security issues, as *hacking* code together can lead to unintended consequences. This problem is exacerbated by the fact that security is mainly an afterthought [34, 168]. Researchers have developed various schemes to distribute and install IoT updates [169], however the updates must first be created. Updates are commonly made for desktop computers, the number of commercial sellers is small, and their updates apply to millions of devices. IoT devices on the other hand have multiple discrete manufacturers and multiple different software stacks, each with different software versions, this means that such a software update system is not economically viable for IoT devices [4].

Another consideration is that IoT devices are commonly sold for a fixed price. Therefore, no recurring revenue is allocated to long term support. This means that there is usually a fixed duration of support, after which the effective cost for continued support and updating increases or becomes untenable, as the number of new consumers decreases and the cost of support increases. This is particularly concerning for industrial IoT devices which are expected to operate for 10 years. Smartphones, for example, are commonly updated for only 2 years [13]. Security conscious companies like Google and Apple extended 3 and 4 years support on their handsets respectively, but this is shorter than the expected 10 year lifespan for industrial devices [170]. Redhat is an example of a company offering long term support for their operating system with ≈5.5 years of full support and ≈3.5 further years of maintenance support. This provides roughly 10 years of support, which can be further extended through their extended life phase, which receives critical security updates. However, there are two considerations which should be made when applying this methodology to IoT devices. Firstly, Redhat does not support the latest software features. For example, the latest release available supports version 4.18.0-147 of the Linux kernel, while, at the time of writing, the latest version of the kernel was 5.5.10. This position is mirrored in user space packages as well. The advantage of this is that the releases are very stable and reducing the number of possible test cases makes it possible to ensure that there are no unintended consequences.

IoT devices on the other hand are commonly made with the latest and greatest features, and lack standardisation between devices, given the large number of manufacturers, means that currently a Redhat model could not be cost effectively employed for IoT devices. The second consideration against IoT devices employing the Redhat model is that their continued support would be prohibitively expensive for most IoT devices, particularly as these are designed to be low-margin high volume devices. Finally, even if support packages are available, encouraging users and organisations to actually update their devices remains a challenge. A technically oriented survey found that 40% of consumers have never knowingly updated their devices [171], and some users actively avoid updates after negative experiences [172]. These issues lead to poor longevity and IoT devices, not receiving security updates [141].

Academic researchers and governments have proposed a scheme where devices could be issued with labels [14, 173], similar to food labels which detail the items best before date and its ingredients, IoT devices would show the predicted duration of support updates for the product — provided the company remains in business. However, there has been considerable debate about ways to accurately measure security features and as such this is an on-going area of research.

3.3 Device Longevity

IoT devices precarious longevity should also be considered. Although new legislation (Section 3.6) is likely to address product longevity and repairability [174], the dependency of a physical product on the availability of a vendor-provided server is becoming increasingly untenable. A high profile example of this problem was the acquisition of Revolv, which was purchased by Nest to expand Nest's product line, but once Google acquired Nest, they no longer wished to maintain the servers for Revolv, leaving the product inoperable after the switch-off date [175, 176]. Such situations highlight a key conflict between customers' expectations and those of service providers: customers expect a product capable of performing as advertised with a reasonable life-span, and service providers aim towards flexibility to cease service when its maintenance becomes burdensome. While this approach is common approach for software companies [177–180], actions like this have led to customer dissatisfaction when hardware purchased has been rendered non-functional as a result of a discontinued service [175,176].

Alongside long term longevity issues, short-term connectivity issues can affect the operation of hardware devices, if a service provider suffers from an outage — electrical, network, or software — the availability of the server, and therefore control of the device may be affected. One such documented failure was of an automated and IoT-enabled pet-feeder, where failure of the backend service led to the device not functioning, preventing users issuing new instructions. Additionally, as all device logic was stored in the on vendors server, pre-recorded instructions also did not operate, failing to dispense previously agreed portions of food [11]. The ease with which physical devices can lose key functionality when their server-side companion is no longer available highlights the fragility of services wholly dependent upon a centralised external service, as well as the risks of relying on such solutions. Situations like these underscore the high dependency of current IoT devices upon internet connectivity for their core purpose: when equipment is unable to fulfil its primary purpose, due to a server or network outage, users lose the convenience that the connected equipment offered them.

3.4 The Trade-Offs of Complex IoT Devices

Running a full operating system has numerous advantages such as being able to take advantage of other existing libraries and solutions easily, making development and therefore, time to market faster. Additionally, developing in higher-level programming languages is generally easier, and there are more experienced engineers to do this [181]. The disadvantage of using a full operating system is that more resources are required to maintain it, and therefore larger batteries and more powerful processors. Moreover, many of the software packages are so large it would not be feasible for a single development team to ensure their security. Additionally, when a vulnerability is discovered, the device would have to be updated, which can be challenging for low-powered IoT devices. In response researchers have made simpler lightweight operating systems for devices [182, 183], which create a smaller system that is easier to maintain and ensure there is no unintended consequences. However, they will not suit every use case. Therefore, where possible, it is better to create simple auditable systems, both from a cyber security perspective but also from a longevity perspective. Qualified software developers are required to update and maintain systems, and as technology advances, older technologies are replaced with newer systems. However, this leads to situations of organisations not being able to maintain their old systems, as the knowledge is lost or forgotten when developers move on or retire. This issue is compounded by programming design methodologies changing, and software libraries altering functionality. The outcome of all these changes mean that organisations will not update their applications out of fear of breaking a component and not being able to fix it.

3.5 Personal Data & Privacy

There has a been a lot of controversy recently around the use of personal data without the individual realising which personal data is being collected, for how long, and how it can be used to influence and understand individuals [184]. This data has been mined behind the scenes through each individual's particular reaction to a presented situation; what they responded positively too, what caused them to engage further with the platform, and what discouraged further browsing. Using these data points enabled grouping users into separate categories for each individual situation predicting their future reactions to similar situations. Famously, in the Cambridge Analytica scandal, Facebook was manipulated into selectively presenting information to its users [185]. Separately journalists discovered that Facebook held 29,000 data points on its average user [186]. This shows the level of the data which can be extracted from seemingly innocuous interactions. Therefore, as the number as IoT devices increases so does the potential for individuals to be monitored, despite the newly implemented legislation that regulates data mining. As this does not restrict the inferences that companies can make using that data, it is critically important to ensure that unnecessary data is not held on consumers, and that data is only going to the intended recipient, as it could be intercepted and exploited.

Prior work in the area of security and privacy of IoT devices typically focused on the challenges around security and privacy of personal data within connected devices, without practical considerations [187]. Research also discussed unifying and enhancing connectivity to improve home control and facilitate greater convenience in the use of connected equipment [188]. Notra *et al.* highlighted potential security benefits of separating IoT devices from other devices on the network, particularly to restrict the opportunity for IoT devices to access the internet [189]. This concept has been proposed as a potential solution to privacy violations, such as surveillance cameras transmitting user images to hard-coded email addresses without notifying the user, or trivial privilege escalation through insecure hard-coded root credentials [190–192]. On the other hand, Notra *et al.* also suggested that restricting internet access could be challenging due to limitations in consumer equipment [189].

3.6 Legislation to Improve IoT Security

To attempt to combat these issues, numerous governments drafted legislation and guidelines. For example, at the start of 2020, the California state legislature was the first organisation to ban the sale of IoT devices which use default passwords [193]. Similar legislation is being proposed by the United Kingdom [194], the European Union [195], and the federal government of the United States of America [196]. However, the issue with such legislation is that they only tackle clearly measurable and definable tasks. For example, although default passwords would no longer be used, this would not prevent a user employing a weak password, potentially due to the complexity of entering (and memorizing) a more complex one. The UK's code of practice for IoT devices [197], along with similar schemes elsewhere in the world, has created a series of guidelines aiming to improve IoT security, such as encouraging companies to start a vulnerability disclosure policy, ensuring timely software updates, minimising exposed attack surfaces, and communicating security developments. However, all these guidelines are general and subject to interpretation. The five eyes countries (i.e. New Zealand, Canada, Australia, United States of America, and the UK), which have had an intelligence-sharing program for a number of years, agreed to collaborate and work with industries and standards-bodies to provide better protection to users, specifically advocating that devices should be *secure by design* [194].

3.6.1 GDPR

The EU introduced the General Data Protection Regulation (GDPR) on the 25th of May 2018 [198]; legislation designed to improve consumer data protection and accessibility. GDPR requires companies to acquire users consent to process their data, mandates safe storage of data for a prescribed duration, and alerts consumers to any data breach. Moreover, companies must give consumers the ability to withdraw their personal data and either move it to another company or download it for themselves, after which the company must delete all the users data.

Data privacy is becoming more of an issue for consumers being increasingly aware of the potential data being stored on them, and companies using that data for marketing purposes. IoT devices, in particular, have a unique insight into an individual's private lives. For example, wearable devices can reveal one's daily activities [199], smart TVs can report the programmes watched [200], and connected children's toys have been used as microphones into homes [201]. From a personal security aspect, thermostats and smart locks can further disclose when no one is not at home [2], making the individuals vulnerable for potential breaking and entering (e.g. when one is on holiday). The penalties for not complying with GDPR rules can be up to 4% of the company's global annual revenue or up to €20 million, whichever is higher. The fine is dependent on the exact violation [198, 202]. For large tech companies, with global revenues in the billions of dollars, this can be significant. For example, Google was fined €50 million for forcing users to create a google account in order to operate an Android phone [203]. The UK's Information Commissioner's Office (ICO) recently fined British Airways £183 million when personal data of approximately 500,000 customers was leaked, including their login details, payment card information, and other personal and travel specifics [204]. Similarly, the ICO intends to fine Marriott International Hotels £99 million, as a variety of personal data of 339 million guests staying at Marriott properties were compromised from 2014 until the breach was discovered in 2018. At the time, around 30 million of these guests were residents of countries in the EEA [205]. Marriott has since suffered a second data breach of up to 5.2 million consumers in March 2020 [206]

The powerful nature of the EU, means that any organisation operating within the EU must comply with the legislation. In the year and a half GDPR has been operating it has issued 124 fines to various organisations, most commonly for insufficient legal basis for data processing and insufficient technical and organisational measures to ensure information security [207]. This resulted in a world-wide change, as any company wishing to process the data of EU citizens regardless of location has to comply [208].

3.6.2 NIS directive

The European Parliament adopted the directive on security of Network and Information Systems (NIS) on 6th of July 2016 [209], and member states had to implement the directive into their nation's laws by 9th of May 2018. The aim of the directive is to improve the general cybersecurity within the EU and it has three main foci: (1) member states should be appropriately prepared by having computer security incident response teams and a competent national NIS authority (2) member states should share information about specific risks, incidents and prevention for specific cybersecurity incidents, and (3) detailing that a culture of security should be created across all sectors which are vital to the economy and society, and rely heavily on information and communication technology. Such sectors include energy, transport, water, banking, healthcare, and digital infrastructure, such as cloud computing services, search engines, and online marketplaces.

One of the issues with the economics of utilities in the UK is that they are financed through government-mandated funding, which has to be allocated before the start of each seven-year-long financing period [210]. This makes it challenging to rapidly respond to a changing situation and allocate budget to objectives which are difficult to quantify, such as securing a system. As a consequence, utility companies have been asked to implement security solutions, without being attributed any budget to achieve the requested work [13]. The National Audit Office further reported that several government departments struggled to detail a return on investment for security projects, partly due to the lack of metrics used to measure what has been achieved as it is [211]. Although using standardized metrics to measure outcomes is a prominent business strategy, it is not possible to state that a system is 5% more secure, as it might be to say that an advertisement campaign had 5% more engagement from the target audience. This makes it difficult to justify cyber security expenditure, which is mirrored in all areas of business, industrial, commercial, and domestic markets.

3.7 Summary of Economic & Policy Background

This chapter showed that despite the demand for secure devices, there is little appetite to pay extra for it. This is partly due to an assumption that security is already present in products and partly given the difficulty in measuring tangible outcomes of securing devices [211]. Legislation might assist in making devices and their data more secure, for example through enforcing no default passwords, but this would not prevent individuals or organisations from setting their own easily guessable passwords out of convenience. In addition, it is also important to provide a tangible benefit to manufacturers, to get them to engage. One benefit of securing devices and limiting the amount of personal data obtained from users is the reduction of manufacturers' potential legal and financial exposure. Although that would reduce their insurance premiums, due to the high cost of current security solutions, there is insufficient financial incentive to implement them. Hence, technical solutions to providing security at little additional cost that is easy to use are still required. As the financial constraints unite industrial devices and domestic appliances, potential solutions will be examined for both, starting with the domestic environment.

Chapter 4

Domestic IoT

Recent developments in consumer behaviour have driven an increase in the demand for Home Automation devices, which promise to make our lives more convenient and automated. Lights can be turned on and off using a voice assistant, a smartphone, or through automatic triggers such as when approaching the home. Although automated systems have been available for many years, as technology has improved, the cost of these devices and the difficulty of installation has decreased, and as a result consumer IoT devices have seen a rapid increase in demand, particularly in a domestic setting [212]. However, this convenience and affordability comes with security and privacy risks. There have been numerous recent instances of domestic internet-connected devices being compromised [27, 213–215]. Low-cost domestic IoT vendors have often struggled with correctly implementing security protocols and creating a secure system. One of the reasons for this is the lack of consumer willingness to pay a significant premium for security, as discussed in Chapter 3, which in turn incentivises manufacturers to concentrate on more promotable aspects of their product [16]. Two of the main financial costs of implementing security features on low-cost devices is the engineering time required to develop the system and the additional hardware cost required to run the system. The former of these — like all software development — is a fixed cost. Regardless of how many units are sold, the same software can be replicated for a single payment. Moreover, various cost-saving measures can also be employed to reduce the time spent on software development, such as using recognised software libraries. Instead this chapter will focus on hardware, as the cost will increase with every additional unit. This is known as a variable cost, as it varies with the number of units created [216]. Therefore, the hardware cost for the required security features cannot be distributed over all the products sold. Instead, this cost is passed directly onto the consumer, who, as highlighted in Chapter 3 is unwilling to pay extra for security features. Thereby creating an incentive for manufactures not to include security features. The solution proposed in this chapter will suggest using pre-existing hardware to enable security features in low-cost devices without an additional hardware cost.

4.1 Proposal to Improve Domestic IoT Security

4.1.1 Architecture Design

Considerable research has been conducted into securing IoT devices using various methods. One strand of research involves making significant changes to the IoT ecosystem [217], and therefore, a significant amount of momentum would be required to overcome the existing inertia and ecosystems consequently, none of these techniques have taken hold (Section 2.7). Another strand involves employing various different hardware, from sensors to more powerful processors, to enable security features [93, 218]. However, all of these solutions require additional hardware, and therefore additional cost. One way to overcome this cost of additional hardware is to use pre-existing hardware, of course, this would limit the potential audience to only people with the required hardware, but by using a ubiquitous product, this requirement would be minimised. One such device are smartphones which 95% of the target market for smart home devices have at least one of [219]. Moreover, users are familiar with smartphones and therefore, this will make it easier for users to setup and control their device.

There are a wide variety of domestic IoT device protocols and communications systems; Zigbee, Z-Wave, WirelessHART, Wi-Fi and Bluetooth, however, these can be grouped into two main categories, devices which can directly communicate with their intended recipient and devices which use an intermediary. Wi-Fi-based IoT devices can directly communicate with other IP devices, most domestic IoT protocols, on the other hand, rely on sending messages through a separate hardware module, which is connected to the Wi-Fi router. This intermediary is often called a hub, and any IP commands it receives are converted into its local wireless technology such as Zigbee, before reaching the end device. The advantage of this system is that Zigbee or similar technologies require considerably less energy than Wi-Fi devices. Requiring less power means that devices can last longer on battery power and make a battery install viable. However, Zigbee based devices require separate hardware to connect to the IP network [220]. Bluetooth devices also offer lower energy requirements, and 3.7 billion Bluetooth enabled devices were shipped in 2018 [221]. However this technology is mainly used for devices which connect directly to a smartphone, such as headphones or wearables, as it follows a master-slave topology [222]. This design also means that multiple devices could not control a single device at the same time. Meaning as with Zigbee, Bluetooth devices require an intermediary to gain access to the IP network. This is one of the reasons that Wi-Fi has become the de-facto standard for standalone domestic IoT devices, and as such, was selected for the proposed architecture.

An advantage of devices using Wi-Fi is that devices can either operate as a Wi-Fi client or as a Wi-Fi access point. While in access point mode, other Wi-Fi devices, such as a smartphone, can connect to the IoT device broadcasting a Wi-Fi network, and transfer data as shown in Fig. 4.1. This data transfer can be used to transfer security credentials, as described in Section 4.1.2. This is especially beneficial for devices which have very limited user input, where it would be a challenge to enter credentials otherwise.

Google employs a superficially similar setup procedure with its popular Wi-Fi-based Chromecast devices, where a smartphone is used in the setup procedure to connect the IoT device to the domestic Wi-Fi network. This approach has also been patented for household appliances [223]. However, for the actual provisioning of the device, and secure communication, Chromecast initiates a TLS connection to Google's servers. The Marvell DE3005-A1 system-on-chip [224] contains a hardware

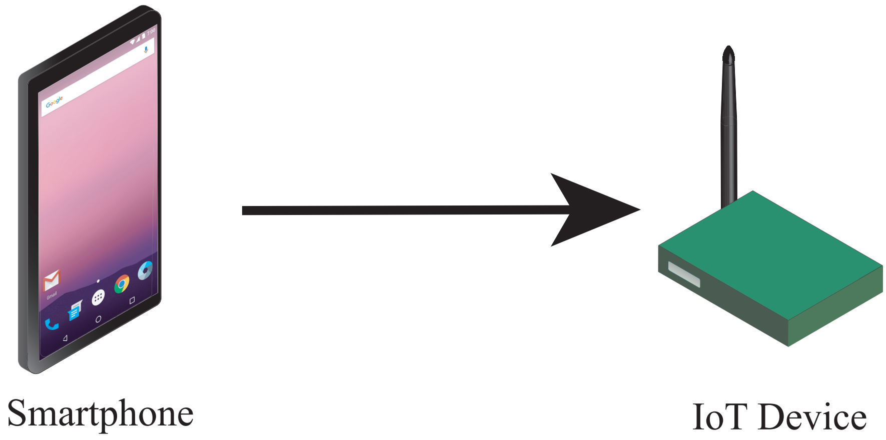


Figure 4.1: Using a Smartphone to Transfer Credentials to an IoT Device

random number generator and AES hardware accelerator, enable this connection setup. Low-cost, low-powered devices do not feature any of this additional hardware, therefore, cannot implement this existing solution [15]. Instead, seven specific steps will be discribed to securly transfer credentials using an existing hardware.

4.1.2 Key Transfer Protocol Details

The secure initialisation protocol is shown in Fig. 4.2 and operates as follows:

1. The device is factory provisioned with an (ideally unique) Wi-Fi credential, made available to the user in the packaging

2. On initial activation, the device sets up a private network using this credential, and the user connects to it using their smartphone

3. The smartphone seeds random data and generates an AES key

4. The smartphone shares this key with the IoT device, along with the credentials for the network the device will use to connect to the Internet

5. The smartphone disconnects from the private IoT network and reconnects to the domestic network

6. The smartphone and IoT device establishes a secure connection between each other over the local network using the key the smartphone generated and shared

7. The smartphone shares the IoT device's key securely with the remote server.

On start-up, the device will broadcast its Wi-Fi access point, using a default Service Set IDentifier (SSID) and device-specific password, in line with the UK's Code of Practice for Consumer IoT Security [197]. The passwords will be provisioned into the devices at the factory and will be optionally supplied inside the product box in the form of a QR code. An additional advantage of this approach would be the automated usage of 63 character long Wi-Fi passwords while removing the potential for human error during input.

An accompanying app will be downloaded from an application marketplace, e.g. the Google Play Store, which generates an AES key, and automates the connection to the private network. Initiating this process, the user would scan the QR code, the app would extract the credentials and switch the smartphone's Wi-Fi connection from the home network to the IoT device's generated network. The smartphone would then generate an AES key and transfer it to the IoT device. The user would then be prompted for any specific information, e.g. a personalised device identifier and the domestic Wi-Fi credentials for the IoT device to join the network.

After the key is shared, both devices (re)join the domestic Wi-Fi network and once the IoT device and the smartphone can see each other on the network, the joining process will be considered successful. Finally, the smartphone will take the user through an optional stage of linking the device to an external server, for out of the house control. The smartphone will transfer the shared key by setting up a TLS connection to the server to securely transfer the AES key. As performing TLS requires a significant amount of mathematical operations and random numbers constrained devices would not be able to create a secure connection natively. Smartphones on the other hand, would be able to make this connection trivially. The phone, the IoT device and server should now all be able to contact each other and encrypt and decrypt all messages. As illustrated in Fig. 4.2, the proposed architecture is based on local interaction, with keys then being sent to a remote location, rather than trying to provision a unique key from a remote location. This architecture provides the following features:

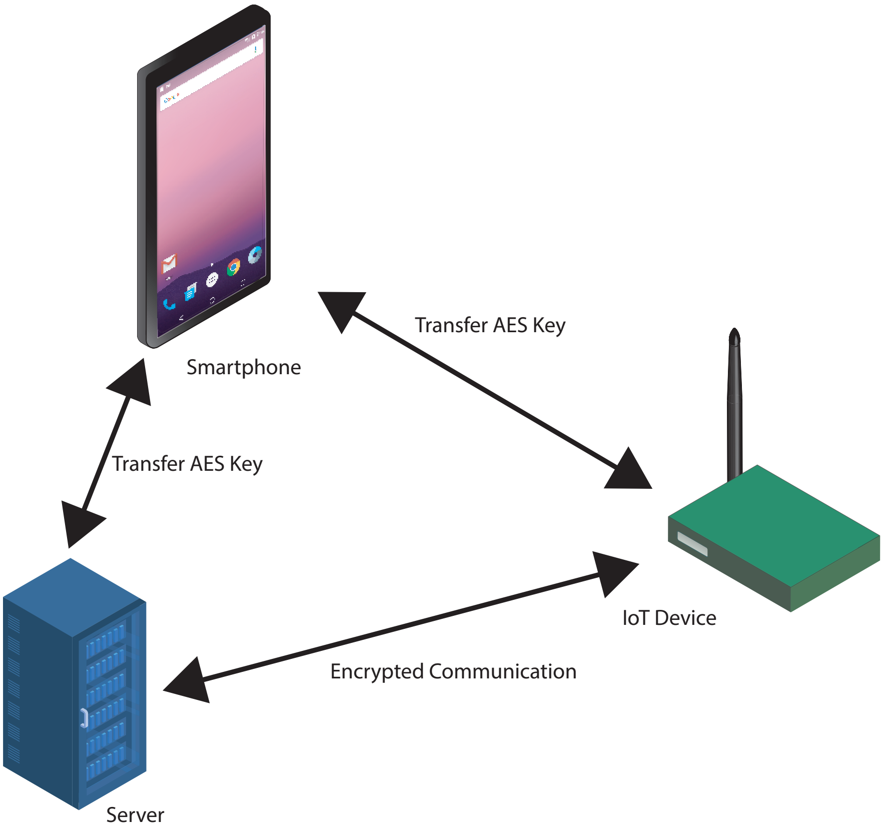


Figure 4.2: Transferring Security Credentials to an IoT Device & Remote Server

- Removes the requirement for any additional hardware to implement state of the art security features.
- An architecture which allows IoT devices to be controlled without relying on an external server decreasing latency [113].
- If the manufacturer's servers were down or decommissioned, the product would continue to operate, whereas similar devices have previously lost functionality in these situations Section 3.3. In all these instances the device could default to running locally and still be locally functional.
- An automated key generation and transfer process, which uses well-established cryptographic libraries, removing the risk of an attacker modifying the manufacturers key generation algorithm.

- Transferring the key over a short term private network, means the device is not compromised even if the domestic Wi-Fi network is.

Encryption & Authentication

When implementing cryptographic functions another practical consideration is the order of operations when encrypting and authenticating messages. Messages should be encrypted and then authenticated, shown in Fig. 4.3. This sequence of operations is important as, running creating the MAC over the encrypted message would ensure the ciphertext could not be tempered with.

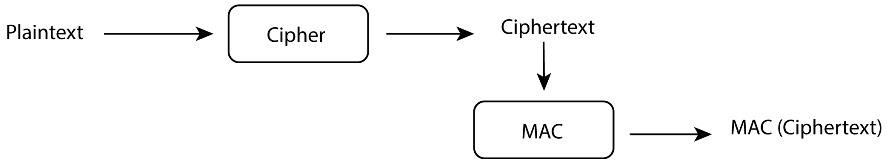


Figure 4.3: Encrypt-then-MAC

Taking the MAC of the plaintext and then encrypting result (Fig. 4.4) means that the receiver of the message has to decrypt the message before authenticating that it is valid, thereby opening an opportunity for an attacker to send arbitrary messages to the receiver which they will decrypt before validating. However, without the attacker knowing the session key, they will likely create unintelligible garbage after decryption causing the MAC check to fail. As block ciphers require a fixed length the plaintexts must be a multiple of the block size, and padding is added to resolve this issue. The padding is checked during decryption and invalid padding causes an error to occur, therefore through systematically editing the ciphertext, an attacker can modify the padding and identify changes in behaviour and timing, for example, a webserver returning a specific error code. This is known as a padding oracle attack [225], and can result in discovery of the plaintext message. This was deployed in TLS 1.0 and previous versions and resulted in attacks such as BREST, and Lucky13 [81, 226].

Encrypt-and-MAC (Fig. 4.5) would allow both tasks to be parallelised, and therefore faster in hyperthreaded processors, however as with MAC-then-Encrypt this would provide no integrity on the ciphertext as the MAC is calculated on the

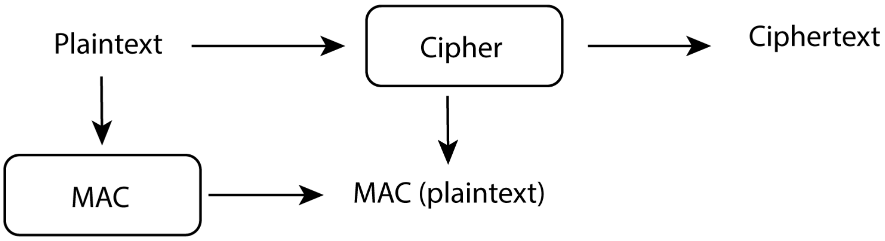


Figure 4.4: MAC-then-Encrypt

plaintext. Creating the possibility of chosen-ciphertext attacks on the cipher as researchers showed in [227]. Therefore, making the Encrypt-Then-MAC the preferred option [228].

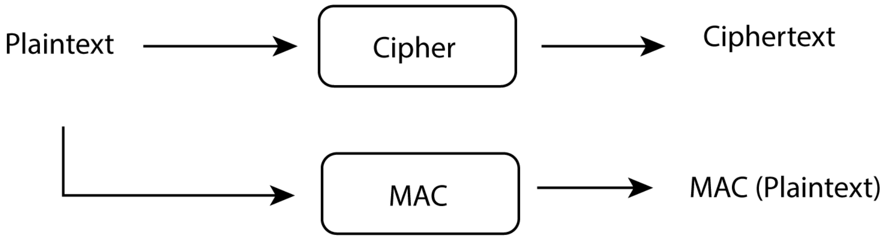


Figure 4.5: Encrypt-and-MAC

Internal Network Communication

The mobile application achieves internal communication without a centralised hub by monitoring the network. If any of the IoT devices lost their previously assigned address, periodic Multicast DNS (mDNS) searches would find and update any record any change in IP address. The downside to this approach is that any malicious device on the network could detect the IoT device's presence through its fingerprint. However, considering the device could be detected through several other techniques [229–231], it was reasoned that this would make a minimal difference, considering all communication would be encrypted.

External Network Communication

Communication with the IoT device when on a separate network is managed via a central server, as shown in Fig. 4.6. The server presents an API endpoint and an authorised trigger such as, the user's phone, a voice assistant, or one of a range of automation apps, such as If This Then That (IFTTT), Tasker, Zapier, issue a request to the same server, which updates the status of the device on the server. The IoT device connects to the central server and update accordingly, to enable Network Address Translation (NAT) traversal solutions such as Traversal Using Relays around NAT (TURN) [232], Interactive Connectivity Establishment (ICE), [233], and Session Traversal Utilities for NAT (STUN) [234].

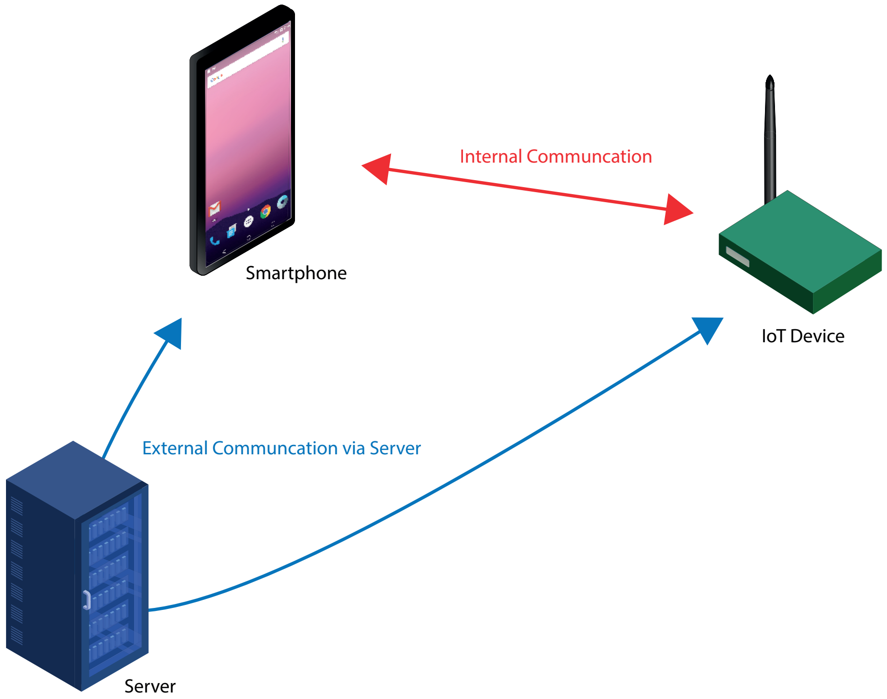


Figure 4.6: IoT Smart Switch Direct Local Communication and External Communication via Server

4.2 Evaluation of Proposed Architecture

The proposed architecture alters the classical model of generation and transfer of AES keys. Traditionally the key would be generated by the device manufacturer and then transferred to the device in the factory. Under this arrangement, the manufacturer is responsible for implementing a suitably random key generation scheme. This means that sequential keys, identical keys or otherwise insecure keys could be inserted without the user's knowledge. On the other hand, in the proposed architecture, the user automatically generates their own keys through well-established and maintained cryptographic libraries as part of the setup process [235]. The possible actors in the system are the owner/user of the IoT device, the manufacturer of the device, and a third party who is targeting the IoT device. The device owner would have the same level of access as the traditonal architecture, in both instances, the owner/user would have physical possession of the device. However, through installing their own keys, the user could modify their key generation and installation program to extract the generated key or replace the generated key with one of their own. This would allow them to perform a 'man-in-the-middle' attack to view the packets the IoT device is sending, but this would only reveal the packets of their own device, without compromising other devices, whereas employing the same attack against a manufacturers generated key has resulted in all those devices being compromised [3, 128]. Other types of attack, such as tampering and denial of service attacks, would also be possible, but this would be the case with all devices of this classification.

A similar consideration with the established architecture, is if an external attacker compromised the manufacturer's provisioning server, and altered the key generation algorithm, this could result in deterministic keys being generated. Whereas, in the proposed architecture it as key generation is achieved through the smartphone application and application store providers put additional safeguards in place to prevent malicious code entering their stores as part of the deployment process [236]. Once a developer initially submits a new application for review, the application store provider will then generate a new private key and sign the application for validity [237]. The developer will then be required to upload their own private key which will be used to encrypt the signing key. Therefore, once the developer wishes to update the app, they must use their stored key to allow the provider to decrypt the signing key, which in turn signs the app, as part of the deployment procedure. There are several advantages to this model. Firstly, if the developer's key is compromised then that key can be immediately revoked, so an attacker could not deploy any new versions of the app. Secondly, the involvement of a third party organisation also decreases the likelihood of compromise, as the legitimate developer would be alerted to the fact a new version of their application is attempting to be deployed. This would prevent an attacker from attempting to publish an application without the developer's knowledge. Another consideration is if the attacker, or malicious member of staff, modified the source code and then waited for the next software release. In this instance, automated testing suites both on the developer's side and the publisher's side would catch any changes to the source code and alert the developer of failing tests.

A consideration which the proposed architecture shares with traditional architecture is that in order to interact with remote devices it is required to store encryption keys on the server, although it would be best practice to store keys in a hardware security module, and these can be rented along with cloud compute power from the four largest cloud providers [238–241], there is no guarantee that a manufacturer would take advantage of such a device, therefore, an attacker could target the vendor's key store and steal the symmetric key pair. With the existing architecture, a user would have little recourse but to wait for the manufacturer to implement a solution. Instead, using the proposed architecture, the user could disconnect the device from the remote server while maintaining use locally, and later re-join the device and in the process generate a new keypair. A futher advantage of using a smartphones, to generate and store the required credentials takes advantage of modern smartphone security properties such a sandboxing, an encrypted file system, and memory management strategies [242], to prevent other applications extracting the cryptographic keys.

Using the traditional architecture, a malicious vendor may use their IoT device to extract information or exploit access unknown to the buyer [200]. Instead, the proposed architecture allows consumers to cut all connections to the vendor, making it untenable for manufactures to construct a business model based on exploiting consumer data in order to finance the overall device and therefore would be unlikely to go ahead [243]. In the existing architecture if a vendor wished to create a secure device they may select to use a higher-performance device, such as an ARM device, which would support full operating system functionality and therefore present a larger attack surface for attackers [3]. Additionally, being able to run a full operating system would make the device more attractive to an attacker as the device can relatively easily be reprogrammed to join a botnet or join a denial of service attack [215]. Extremely low-cost microcontrollers cannot support such operation, given their limited functionality. Moreover, as detailed in Section 3.4 complex system rely on many dependant systems to seamlessly operate while constrained devices have a smaller attack surface.

Information discloser, privacy breach, signal injection, elevation of privileges, device tampering, and spoofing would all represent an equal chance of attack with the proposed architecture, as in the traditional architecture. Therefore, there would be no change in the threat model of these attack vectors.

The final instance of where the device differs to the traditional architecture is the possible interception of the package en-route to the user. A determined attacker could interception an IoT en-route to the user, extract the cryptographic keys, and then intercept, decode and modify messages unknown to the user. Whereas, with the proposed architecture, as the cryptographic keys are not on the device while in transit, they cannot be intercepted. A package interceptor could open the package and record the details on the supplied QR code to intercept the transmission of the AES key during device setup. However, several additional steps can be taken to mitigate these risks.

1. The Wi-Fi transmit power of the IoT device is turned down to its lowest power, 0.25 dBm in the case of the ESP8266, a common IoT device Wi-Fi module so that the signal disappears into the noise floor when attempting to detect at a distance greater than 30 cm away. The Wi-Fi adaptive power control of Android phones will also cause the corresponding smartphone to transmit power to decrease accordingly [244]

2. Configuring the Wi-Fi module to support a single connection; therefore, only a malicious attacker or a legitimate user can be on the temporary Wi-Fi at any one

time.

3. Sealing the QR code in a tamper-evident envelope

Additionally, the temporary Wi-Fi network would only be configured to broadcast for approximately one minute, less than the typical time a Wi-Fi Protected Setup (WPS) device allows devices to connect to the Wi-Fi network [245].

4.3 Testing Proposed Domestic Key Transfer Architecture

As detailed in Chapter 2 there are various AES modes and authentication schemes. ECB would not be suitable for messages longer than the size of the block. CBC, Cipher feedback (CFB) and OFB would also not be suitable given the requirement for a random nonce which the proposed system cannot rely on. AES-CTR, on the other hand, does not require a random IV, therefore it would be suitable. AES Galois/Counter Mode (AES-GCM) also does not require a random IV, therefore both of these modes were considered. The advantage of AES-GCM would be the built-in authentication, whereas AES-CTR would have to be combined with an authentication cipher, such as HMAC or CMAC. To compare the effectiveness of each scheme on a constrained device encryption and authentication modes were developed and tested on the MSP430G2553. It was initially compiled with various different optimisation levels to show the proformance increases possible through optimisation. The number of clock cycles required for each mode is shown in Table 4.1. The number of cycles for decrypting and verify messages, were also profiled as IoT devices would commonly receive messages from the user or server as well, this is shown in Table 4.2.

Constrained devices are often limited memory space as well as processor, for example, the MSP430G2553 only has 512 Bytes of RAM, and 16KB of flash memory [111]. therefore, the memory and stack requirements for each mode of operation at different optimisation preferences was profiled to discover the overhead for adding security features. Depending on the devices use case, it may require more

Optimisation	AES-CTR	AES-CTR+HMAC	AES-CTR+CMAC	AES-GCM
none	65487	265100	131726	713810
low	60178	247317	119220	420503
medium	46122	203565	91902	364626
high speed	30604	169880	63042	126474
high size	50277	238890	100213	210400
high balanced	37902	225437	75929	187548

Table 4.1: Number of Cycles Required for Encryption & Message Authentication Using Different Optimization Profiles

Optimisation	AES-CTR	AES-CTR+HMAC	AES-CTR+CMAC	AES-GCM
high speed	32721	1744136	75410	155976
high size	52573	245760	124557	236464
high balanced	39435	234473	77492	201786

Table 4.2: Number of Cycles Required for Decryption & Verification Using Different Optimization Profiles

speed, or require more memory for the primary application of the device, or both, therefore each mode was profiled. The number of bytes required for both storage and stack are recorded in Table 4.3.

These results show that computation computational cost of HMAC authentication was consistently higher than a CMAC operation, however given the sequential nature of CMAC it was reasoned that as the message length increased the number of cycles for CMAC would also increase, while the number of cycles for a HMAC operation would increase at a lower rate given that only a single encryption is required and two hashing operations. Therefore, the number of cycles for different message lengths was recorded as shown in Fig. 4.7. As hypothesised it showed that for small messages CMAC was more efficient, while as the message length increased the difference decreased between the two modes decreased, however it should also be noted that the majority of consumer IoT devices would send short messages. Therefore, CMAC is the most efficient.

As AES-GCM required more cycles and memory, and CMAC was more efficent than HMAC, it was decided to use employ AES-CTR & CMAC, as would result in lowest processing time, and therefore faster response times and less energy consumption. It was noted that this decision would benefit the IoT device, but they would also benefit

	Speed		Balance		Size	
Mode	Memory	Stack	Memory	Stack	Memory	Stack
AES-CTR	1848	60	1886	60	1518	210
CTR & HMAC	4734	672	3688	766	3688	766
CTR & CMAC	2280	164	2050	284	1818	244
AES GCM	2863	216	2765	216	2771	216

Table 4.3: Memory Usage in Bytes

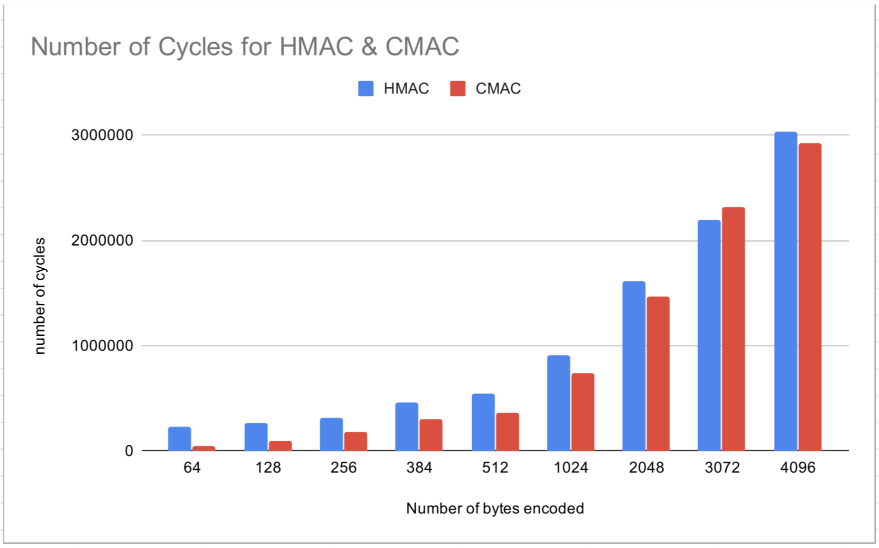


Figure 4.7: Comparing CMAC vs HMAC for Different Message Lengths

the smartphone application, and the backend server as these more powerful devices can take advantage of the parallel properties of AES-CTR, to operate on separate blocks concurrently [246], decreasing the effective overhead to run the backend system. Another advantage which was discovered during testing was that encryption operation using various different keys required the same amount of time each time. This means that due to the single threaded software implementation of the MSP430, this was not vulnerable to a timing side-channel attack as outlined in Section 2.9.8, resulting in improved security.

Power Consumption

Next the energy consumption and execution time were measured for transmitting and receiving a 64 byte message using AES-CTR & CMAC broken down into encrypton

	Encryption				Decryption			
	500 Operations		1 Operation		500 Operations		1 Operation	
Clock Speed	Time	Energy	Time	Energy	Time	Energy	Time	Energy
(MHz)	(seconds)	mW	(ms)	mW	(seconds)	mW	(ms)	mW
1	37.859	512.979	77.798	1.268	38.899	634.411	77.798	1.268
8	3.955	370.639	9.8254	1.416	4.913	458.553	9.825	0.917
12	2.5797	358.963	10.077	0.882	3.205	441.318	6.411	0.882
16	1.969	354.847	4.894	0.877	2.447	2.4471	4.894	0.877

Table 4.4: Power Consumption & Response Time of AES-CTR + CMAC at Different Clock Speeds Optimised for Speed

	Encryption				Decryption			
	500 Operations		1 Operation		500 Operations		1 Operation	
Clock Speed	Time	Energy	Time	Energy	Time	Energy	Time	Energy
(MHz)	(seconds)	mW	(ms)	mW	(seconds)	mW	(ms)	mW
1	50.176	800.293	121.873	1.914	60.936	957.291	121.873	1.914
8	6.328	573.973	15.421	1.154	7.710	708.184	15.421	1.416
12	4.133	556.412	10.077	0.882	5.038	679.547	10.077	1.359
16	3.147	547.909	7.679	1.339	3.839	669.932	3.839	1.339

Table 4.5: Power Consumption & Response Time of AES-CTR + CMAC at Different Clock Speeds Optimised for Size

which would encrypt the message, and calculate the message authentication code, and an decryption operation which would validate the message authentication code, and then decrypt the message. This was conducted for all three highest compiler optimisation modes: increased speed, reduce code size, and a balance between the two. Tables 4.4 to 4.6. From this it was discovered that each mode required less time than the average human response time of 250 ms [112]. It was also noted that although the MSP430G2553 uses less power at lower clock speeds [247]. The increased time it required to execute the operation meant the overall consumption was increased. Therefore, it would be more efficient to selected 16 MHz to execute the operation as fast as possible, and then enable one of the MSP430's built-in low power modes to converse energy while idling. The faster response time than human reaction time would make the operation appear instant therefore removing any potential a barrier due to latency when users are interacting with their device.

	Encryption				Decryption			
	500 Operations		1 Operation		500 Operations		1 Operation	
Clock Speed	Time	Energy	Time	Energy	Time	Energy	Time	Energy
(MHz)	(seconds)	mW	(ms)	mW	(seconds)	mW	(ms)	mW
1	37.859	613.258	99.315	1.602	49.657	801.237	99.315	1.602
8	4.781	442.197	12.5512	1.154	6.275	577.394	12.551	1.155
12	3.124	429.199	8.207	1.120	4.103	560.319	8.207	1.120
16	2.380	6.249	6.249	1.104	3.125	552.146	6.249	1.104

Table 4.6: Power Consumption & Response Time of AES-CTR at Different Clock Speeds Optimised for Balance

4.4 Creation of Smart Switch Using the Proposed Architecture

4.4.1 Market Requirements

Two of the prominent issues with IoT device security is that the device has to be low-cost and simple to use. Both of these factors are driven by market competition [248]. The low-cost requirement means that low-power processors, such as the Texas Instruments MSP430 family, would be used where possible. An internet-connected light switch, would only have to respond to web requests and actuate a relay to toggle the light, therefore a MSP430 would be suitably powerful to achieve this task. The downside of using a low-cost, low-power processor is the time taken to execute computationally intensive tasks, such as generating a suitably large random prime numbers to initiate secure public-key cryptography [249]. This creates an incentive for manufacturers to not include security features. However, employing the proposed architecture would resolve this issue.

4.4.2 Design

An IoT smart light switch was considered a good candidate for implementing the proposed architecture as this is simple enough that it would not normally require a high-performance processor, and a product which is currently being internet connected. This device was constructed using an MSP430G2553 microcontroller and an ESP8266 Wi-Fi chip. Integrating the concept into an operational product did require a number of other considerations to be made, for example, market requirements and the practicalities of such a system. One such practicality was although there are existing smart light bulbs, such as Phillips Hue and LIFX, both of these products store their intelligence in the bulb itself, meaning that once a user switches off the traditional light switch, the bulbs and their built-in intelligence no longer receive power, as shown by the circuit diagram in Fig. 4.8, once the external switch is opened, the Phillips Hue unit cannot be powered. Therefore, the bulb cannot be turned on again until the physical switch has been switched back on. The current solution to this issue is installing an additional battery-operated light switch to issue radio commands to the light, leaving the traditional physical light switch permanently on [250]. Another consideration when placing the intelligence in the bulb is that it ties the consumer into a specific system and increases the cost per bulb. Furthermore, not of all types of bulbs are available with the required build in intelligence, meaning that anyone who has uncommon bulbs would not be able to use the system.

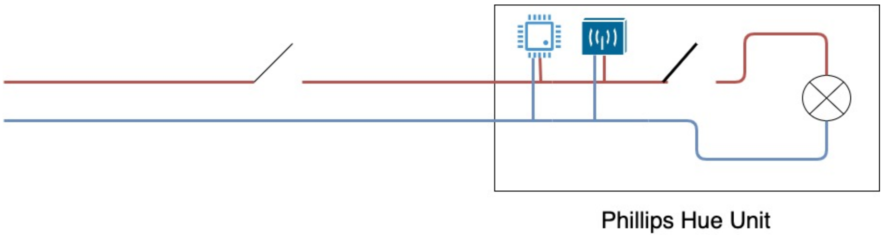


Figure 4.8: Phillips Hue Wiring External Switch Ceases All Power to the Bulb and Electronics

Instead, it was decided to store the intelligence of the system in the light switch itself, which would allow all existing bulbs to work using the traditional physical switch, and through a remote trigger. However, this decision created new issues. Due to the design of UK lighting circuits, there is commonly no neutral wire, instead using a single live wire as detailed in Fig. 4.9. Running current through the light switch also causes current to flow through the light bulb, therefore powering the switch permanently would result in the bulb being permanently powered as well. Although this is not an issue for incandescent bulbs as they require a higher at current of ≈260 mA at 230 V to illuminate. LEDs lights on the other hand, are more efficient, therefore, it was discovered that a only very small current could be leaked through LED bulbs without illuminating them. Moreover, with increasingly efficient designs the exact limit was found to be inconsistent between bulb manufacturers. Resulting in uncertainty if the system would work in a user's practical configuration. This value was found to be between 2 - 6 mA at mains voltage. Therefore, this added a requirement for the device to operate using a small amount of energy to make the overall consumption less than 2 mA at 230 V AC. This could be partly achieved through using low powered components such as the MSP430G2553 [111], and the ESP8266 Wi-Fi chip [251], this was further reduced through reducing the clock rate on the ESP8266, and turning of all unnecessary components on both modules. The system was also designed so that transmissions were limited as this requires more energy than listening [A6, A7]. The device listens on the local Wi-Fi network for commands from the smartphone, and opens a web socket connection to the remote server, and monitors for changes on the server side. Under these constrained conditions the system was recorded to require 0.08 A at 3.432 V therefore requiring an average of 274.5 mW, less than the required 460 mW. A supercapacitors was added to provide a smooth power supply, and allow for a momentary increase in power consumption. The standalone Printed Circuit Board (PCB) design and created board as shown Figs. 4.10 and 4.11.

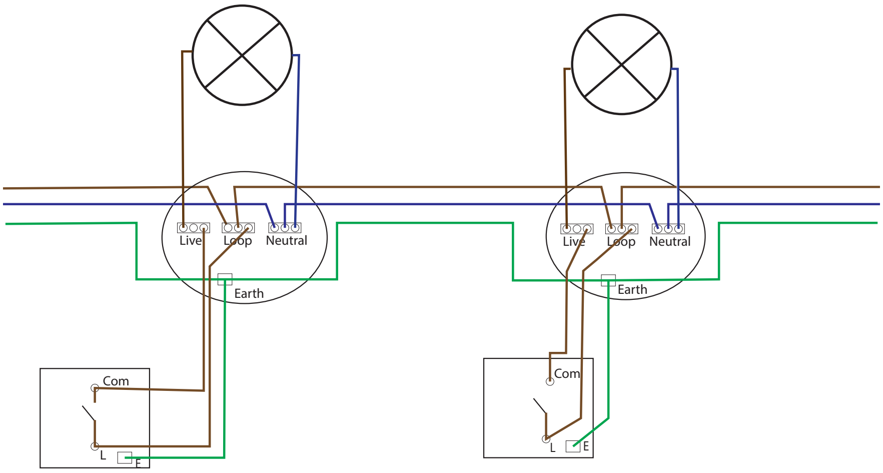
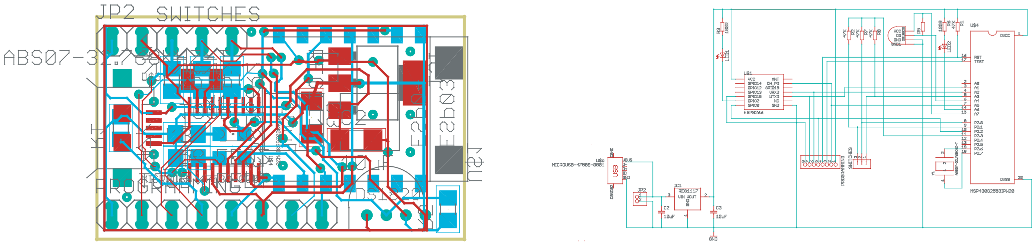


Figure 4.9: UK Lighting Circuit Showing There Is No Flow of Electricity Without Illuminating the Bulb.



(a) Board Layout

(b) Schematic Layout

Figure 4.10: SmartSwitch Printed Circuit Board

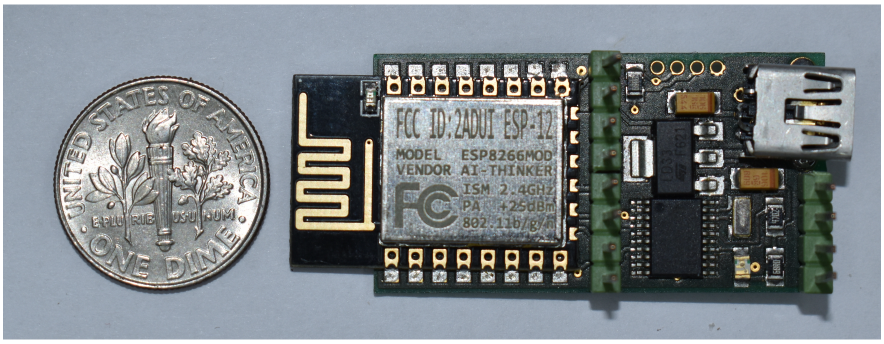


Figure 4.11: Smart Switch PCB Board

This implementation of proposed architecture, could be implemented for a generic IoT device, therefore it was compared against other devices currently on the market and proposed in academic research. A Google Chromecast, was selected as a comparison as it includes start of the art security features, but requires a high performance processor [224]. A generic IoT sensor device was also proposed representing the state of the art in Wi-Fi sensor devices. The Chromecast required 1650 mW, while Guegan et.al, found that the static power consumption for a generic Wi-Fi sensor was 900 mW while idle, and 1254 mW while transmitting [252]. The created smart switch using the proposed architecture requires notably less than both of these solutions while offering the same level of security the Chromecast offer, and the same functionality as Guegan et.al generic sensor.

Using this configuration, the light was able to be controlled from either the app on the local network, remotely via a centralised server, or using the physical switch. Linking with a centralised server also allows the switch to join a more extensive system.

For example, an IFTTT recipe was also set up in conjunction with a Google Home assistant, so that a user could say, *turn on the lights*, and a message would be sent to the remote server, and the light switch would detect the change and alter the light itself.

4.4.3 Device Setup Procedure

The device itself is setup similar to existing products on the market, such as the Nest smart thermostat. Once wires are connected, the MSP430 powers up, and starts the initialisation sequence. The Wi-Fi module enables access point mode, creating a temporary wireless network. At the same time the device owner is expected to use the smartphone app to scan the supplied QR code. This extracts the temporary Wi-Fi password, and connects the smartphone to the temporary network, enabling a two way communication system between the smartphone and the smart switch. The user is then asked to enter the domestic Wi-Fi network credentials, and a personal identifier for the switch. A AES key is then generated, and all the credentials are transferred to the switch. Once this is complete, the switch shuts down its private Wi-Fi network and attempts to join the domestic Wi-Fi network. Simultaneously, once the private Wi-Fi network is no longer available, the smartphone will automatically start searching for it's default Wi-Fi network. Once the smart switch has successfully connected to the domestic network, it enables periodic mDNS broadcasts. Receiving the mDNS messages the smartphone uses the IP address encoded within the mDNS broadcast, to send a confirmation message to the smart switch, confirming the device is correctly setup. However, due to NAT on common domestic internet connections, once the user leaves their internal network they would not be able to control the device anymore. Therefore, as the final stage in the setup process they are asked if they wish to use their device outwith their internal network, and if they choice this, the smartphone application makes a TLS connection to the central server, and securely transfers the credentials to the remote server. This enables the user to issue commands via the server to control the device as detailed in Fig. 4.12. Now, once any change on the device on the local network either through physical interaction or a command on the local network, is reported to the server via a websocket connection.

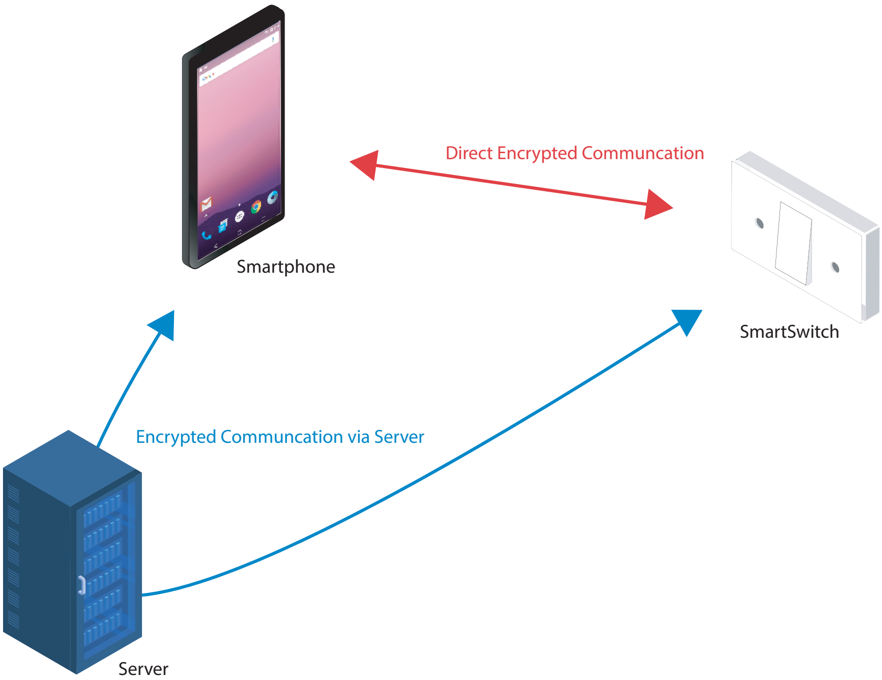

Direct Encrypted Communcation
Smartphone
SmartSwitch
Encrypted Communcation via Server
Server

Figure 4.12: IoT Smart Switch Direct Local Communication and External Communication via Server

If the user did not wish to enable external connections, or the manufacturer ceased providing such a service. The service would still operate locally using mDNS. Similar to the way mDNS messages were employed in the setup process these messages are used to enable communication for any device capable of running a mDNSResponder on the local network. Removing the need for a static IP address, which would require specialised knowledge to configure on the user's router. Instead, if the device's IP address changes it issues an updated mDNS broadcast containing a unique trackable identity, to update the stored IP address. All messages in the system are encrypted using AES-CTR and authenticated using encrypt-then-MAC with CMAC as demonstrated this required the least resources on the microcontroller.

4.4.4 Application Programming Interface

The creation of an API will allow other developers to work with the product, a common design criterion for new IoT products. Additionally, the overall system has been designed with extensibility in mind so as to be able to support multiple types of switch. The system has also been designed to create a group *HomeGroup*, which owns all of the switches. This means that users can be added and removed from the group to control who had access to the lights, rather than each individual user having to add the lights themselves, once a single user adds a new light it will be available to everyone within that group, this would the lights to be accessible by multiple users, as this is a common requirement. It was also observed that a single light switch could contain multiple switches for different areas of the room. These individual sectional switches are referred to as gangs. Therefore, within the system, a switch can have multiple gangs. A Unified Modeling Language (UML) diagram detailing this principle is shown in figure 4.13

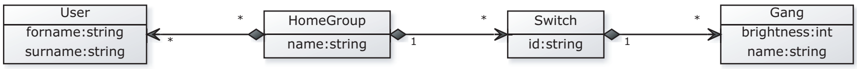


Figure 4.13: UML Diagram of System Model Architecture

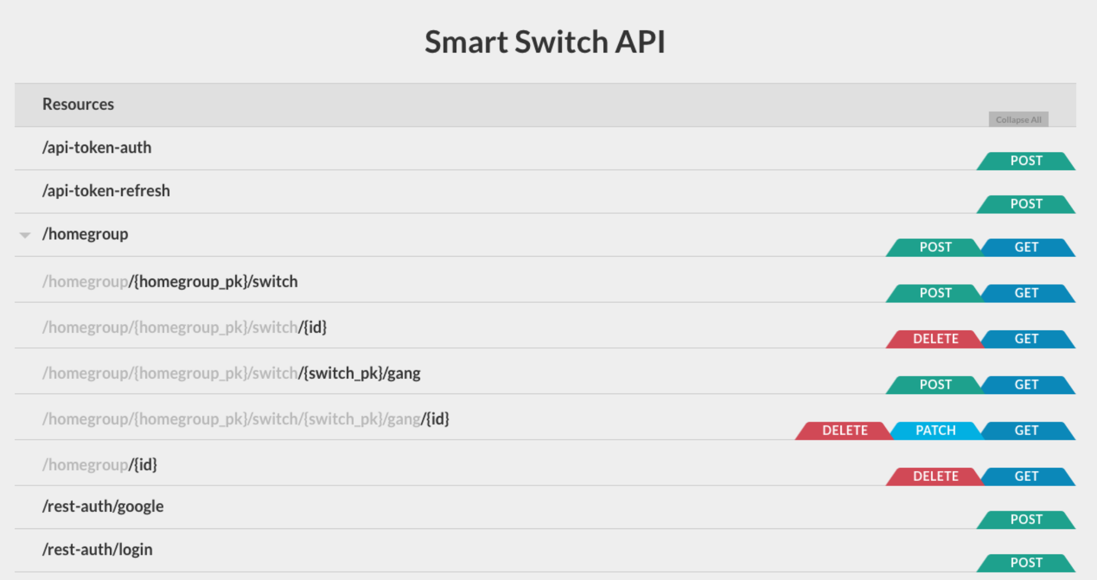
Following this UML structure, a REST API was designed with the path structure of `/homegroup/{homegroup_id}/switch/{switch_id}/gang/{gang_id}` for example, the first section on the first light would be found at `/homegroup/1/switch/1/gang/1`. The follow API is shown in Table 4.7 and Fig. 4.14

The server also supports displaying the API in an interactive format, so that a developer can test the API methods. This format also details what Hypertext Transfer Protocol (HTTP) methods can be used on each endpoint, e.g. the POST and GET are available on `/homegroup`, while DELETE, GET and PATCH are available on `/homegroup/1/switch/1/gang/1`, since this will allow the user to delete a gang which they no longer want, check the status of the gang, and finally modify the brightness and/or name of the gang using a JSON packet, such as

HTTP Method + End Point	Result
GET /homegroup/	Return JSON array of all homegroups
POST /homegroup/	Creates a new homegroup
GET /homegroup/	Return JSON object of homegroup
GET /homegroup/1/switch/	Returns JSON array of switches
POST /homegroup/1/switch/	Creates a new switch in homegroup
GET /homegroup/1/switch/1/	Returns JSON object of switch
GET /homegroup/1/switch/1/gang	Returns JSON array of gangs
POST /homegroup/1/switch/1/gang/	Creates new gang on switch
GET /homegroup/1/switch/1/gang/1/	Returns JSON object of gang

Table 4.7: Server Side API

`brightness:100`, would set the brightness of the light to 100%. A screenshot of this interactive API is shown in Fig. 4.14. The API also supports JSON Web Tokens (JWT), so that commands issued to the devices can be validated without requiring to transmit the user's username and password.



Smart Switch API

Resources

/api-token-auth

/api-token-refresh

/homegroup

/homegroup/{homegroup_pk}/switch

/homegroup/{homegroup_pk}/switch/{id}

/homegroup/{homegroup_pk}/switch/{switch_pk}/gang

/homegroup/{homegroup_pk}/switch/{switch_pk}/gang/{id}

/homegroup/{id}

/rest-auth/google

/rest-auth/login

Figure 4.14: Interactive API Screenshot

4.4.5 Integration with Existing IoT Ecosystem

Another common design choice is allowing users to sign up for accounts using their existing credentials, such as their Google account. Therefore, the backend systems were also configured to allow for social platforms, which meant that Android users would have one-click login when downloading the app, giving a smooth user experience. The API could also be used by other online services such as IFTTT, a popular integration platform allowing two seemingly independent devices to communicate. This meant that users could create their own specific sequence of events, such as a door sensor detecting the user is home and turning on the hall lights. Through this configuration, users could also use their voice assistants to create custom voice commands such as *Turn on Cinema Lights*, which would be programmed to alert the smart switch server and turn on the lights at the rear of the room at a low light level while leaving the front of the room lights off.

4.4.6 Manufacturer Interest

Part of this work was sponsored by a local business who were contracted with a major multi-national lighting company. They wanted a design which was cheap, secure, and did not require a hub. The remote control design was accepted as meeting their requirements, but further development producing a commercial product suitable for working with a wide range of LEDs was suspended in the face of the difficulty in making a cost-effective power supply fit within a standard light back-box.

4.5 Conclusion

The problems with computationally-constrained devices generating entropy are well documented. As discussed, offloading key generation and key transfer to a smartphone removes this entropy requirement allowing standalone devices to implement state of the art security features, while also removing barriers for not including security. The proposed architecture was implemented on a general purpose $1 microcontroller and profiled to determine the overhead required. This showed that the additional processing overhead of security would not adversely effect the latency of domestic IoT devices from the users' perspective, as the time required was shorter than human response time. The energy and timing requirements of such a system were also investigated, by creating a live-wire only compatible IoT smart switch. This found no noticeable delays in responsiveness, nor a large increase in energy requirements. Furthermore, a threat analysis of the architecture found that the proposal would not introduce additional security risks to domestic consumers, instead reducing possible attack vectors, thereby validating the model. As the design only requires a general purpose microcontroller and a Wi-Fi chip, which are both abundant on consumer IoT devices, the design can be retrofitted to existing devices. Moreover, the use of known, existing hardware simplified the setup of a device featuring the new architecture.

Overall, the created system enables end-to-end encryption for low-cost IoT devices, protecting them from 'man in the middle' attacks, Universal Plug and Play (UPnP) vulnerabilities, and malicious devices on the same network. All these features directly benefit consumers by improving the security of their IoT devices, while keeping the ease of use. Consumers will also indirectly benefit from a competitive market, as the barrier for providing security is kept low, protecting against the increase in price. Therefore, the proposed architecture would bring low-cost security with no additional hardware to IoT devices that could not have secure features before.

Consumer IoT devices are mainly targeted at individual customers for home or office use and they mostly allow users to interact with devices and appliances in their environment. IIoT, on the other hand, are primarily developed for use in factories and infrastructure.

Chapter 5

Industrial IoT

Sensor devices such as, temperature, pressure, magnetism and light are all common IIoT devices. Measuring these values allows for increased automation, as systems can detect irregularities and automatically respond if an anomaly is detected. Depending on measured data this response may involve aggregating all the available data and presenting it to a human in an easily understandable format or automatically adjusting the system to prevent any disruption. Both of these options reduce the amount of human intervention required for menial tasks increasing the efficiency of the overall system. Subsequently, increased preventative monitoring can reduce unexpected downtime, labour, and costs. One trade-off for these IIoT devices is they demand high reliability and interoperability, as they should operate with little or no human oversight.

A distinction between consumer and industrial devices is the quantity at which these devices would be purchased. A domestic user may buy tens of devices; single devices built into appliances such as toasters or washing machines, and groups of devices such as lightbulbs or sensors, whereas industries would buy thousands of sensing devices. Therefore, any small increase in the cost of an individual unit is replicated many times over. Another difference between domestic and industrial devices is that domestic devices would normally attach to a user's Wi-Fi network for internet connectivity. IIoT devices, on the other hand, cannot rely on this connectivity for a number of reasons. Firstly, Wi-Fi networks generally only provide coverage for small geographical areas such as homes or offices, whereas an industrial deployment would be over a wide geographical area. Consequently, numerous Wi-Fi access points would be required to deploy such a network. Moreover, the sheer number of devices which are currently envisaged in industrial settings would overload conventional Wi-Fi networks, as these are designed for comparatively few devices. Finally, attaching a large number of devices to a Wi-Fi network is a labour-intensive process, particularly if different devices are connected to discrete Wi-Fi networks, without a federated system such as eduroam [253]. Instead, IIoT devices are commonly deployed on wide area networks designed to operate with thousands of devices.

One of the main challenges with deploying sensor networks to monitor physical environments is creating the infrastructure to power and communicate with devices. IIoT devices can either be battery operated or connected an external power supply. Battery operated devices are significantly easier to install, as no specialist hardware or knowledge is required to provide a power source, but they have a limited lifespan. However, the difficulty and cost of getting a power source to certain areas, particularly rural or difficult to reach areas, makes it impractical to rely on external power, such as the mains. Energy harvesting has been considered as a way of extracting energy from the environment [254]. However, currently this provides little energy or is environmentally dependent working in conjunction with batteries. Therefore, devices must be energy efficient. The second issue when deploying sensor networks is ensuring device connectivity. These networks must be capable of supporting a large number of devices over a large geographical area, and be able to convert the received data into a useable format, to allow integration into existing systems.

This chapter will examine several possible use cases for IIoT devices, and examine three of the most established LPWAN technologies, including their security, and practicality of their deployment. The chapter will then go onto create a simple LPWAN device to examine the security of the solution using the most demanding use case, as any solution would be applicable to other use cases as well.

5.1 Industrial IoT Use Cases

There are many examples where IIoT devices would be beneficial for society in both reducing costs and increasing the quality of service. This section will explore three such use cases and detail their benefits and technical challenges. The common characteristic of each of these use cases is a necessity for low cost security features in industry deployments, while maintaining reliability and robustness [13]. To investigate adding security features to IIoT devices a typical device was proposed where it would be an example of a IIoT device.

5.1.1 Care Homes

Through conversations with various stakeholders including Age Scotland, the writer was made aware of a recurring issue with care home residents leaving their bed during the night to use the bathroom, but struggling to return, due to their mobility issues. While on-site nurses are available to assist, without routinely checking on all of the residents during the night, which would be a labour intensive and invasive procedure for the residents, any accidents would not be noticed until the next morning. Therefore, it would be beneficial to monitor if an individual got up in the middle of the night and has not returned to their bed after a defined period of time. Although there are various devices available on the market which would be able to monitor this, such as video cameras, these were defined as unsuitable, as they are both invasive and financially restrictive [255, 256]. There is also a variety of other sensors on the market, such as bed occupancy sensors, or wearable devices. However, each of these have their own constraints, such as cost, deployment and personal convenience [256, 257]. The author of this PhD study was focused on adding security features to devices for a low-cost, so the exact sensor device can be variable. The low-cost is important as healthcare costs are spiralling to £1000 a week [258], and there is very little large-scale investment [259]. Therefore, for any potential solution, it would be required to keep capital costs as low as possible [259, 260].

It was noted that light could be used as a non-invasive proxy for occupancy and movement. When a resident got up during the night, for any reason, they would turn on a light, and once they returned, turn off the light. Measuring the time from the light being switched on to it being switched back off again would provide a non-invasive way to ensure the resident had safely returned to their bed. If the light is not switched off after a fixed time, an alert would be issued to the on-site care assistant to check on that particular resident.

A difficulty with employing such a system would be the installation within the care home itself; devices would have to be powered and have a way of communicating alerts to a central server which would then alert the on-duty nurse. A Wi-Fi-based device would be able to communicate using existing infrastructure. However, each sensor device would have to be connected to a Wi-Fi network, requiring a user interface. This would be a labour-intensive process as each room would have to be equipped with such a device. Providing a user interface would increase the cost of the unit, invalidating one of the main aims of this study – deploying a low-cost device. Another consideration in deploying a proposed device to this use case, was the energy consumption of Wi-Fi-based devices, which is compared to other communication devices [A6]. A battery-operated Wi-Fi device, which would be more straightforward to install, would routinely have to have its batteries replaced [A6]. Connecting the device to mains power would remove these concerns, but also limit the placement opportunities. Additionally, wall sockets may be in short supply within the resident's room, or the device may be unplugged unintentionally by visitors wishing to use the socket, inadvertently rendering the device non-functional. Although a separate electrical network could be created, this would require an electrician and further increase the cost of employing such a system. Therefore, for a practical deployment, all of these issues must be considered. The work reported here will investigate a viable low-cost secure communication strategy which could be applied to such a sensor device.

5.1.2 Smart Street Lamps

Another use case, where monitoring the light levels would be beneficial, is in street lighting. Street lighting has been proved effective in both increasing road safety and decreasing crime [261]. To this end, city councils and municipalities have requirements to provide sufficient light to an area. However, councils have little oversight of the day to day operation of their lights, resulting in a lack of service for citizens. IIoT light sensors could measure light level output from each streetlamp to monitor the health of each light. Through knowing the health of streetlamps, they can be intelligently replaced prior to or soon after malfunctioning, whereas the current system relies on members of the public reporting any malfunctioning lights. Moreover, the current practice is to replace the whole batch of lights once they approach the end of their predicted service life, leading to waste and unnecessary expenditure.

Measuring the individual luminescence could determine the health of a light and allow for a targeted replacement program, maximising the efficiency of the streetlights and provide diagnostic information. For example, if the light level abruptly decreased, this would indicate a fault with that particular light, while if a series of lights went off, it is likely that a power cut has occurred instead. This increased diagnostic information can enable lights to be repaired at the earliest practical opportunity, enhancing public safety and providing a better service. An alternative solution to timely replacement of faulty lights may be to measure the energy consumption of the lamppost to estimate how much light it is outputting. However, this would be challenging as the effectiveness of the light decreases towards the end of its life. It would also require a probe to be installed on the electrical supply of the light. Instead, using a light sensor and a communication module, similar to the homecare example discussed before, the light level could be automatically reported to the local council. Through restricting the sensor to a battery-operated device, the device could be retrofitted into the head of the lamppost without having to install additional electrical circuitry, reducing the installation and hardware costs. Any created device would have to be low-cost as it would be deployed in thousands of lampposts over a city or county. It is also important to consider the security of such a system, as although the presence of the light itself could be visibly seen, an attacker could introduce fictitious errors to cause a drain on council resources, detracting from the benefits of such a solution. Overcoming these constrains make a proposed device a good use case for adding security to low-cost devices.

5.1.3 Smart Grid Monitoring

A third potential user of IIoT devices is utility companies, these organisations maintain vast physical networks and would benefit from increased visualisation. Electrical companies, in particular, are facing a number of challenges. The traditional paradigm of a large power plant generating and distributing power throughout the country is changing. Previously, the transmission network would transport power onto the smaller distribution network, then to the low voltage layer of 400 V, and finally down to consumer level of single-phase 230 V. Yet, distributed generation, such as wind farms and renewables, challenges this existing architecture. Drawing a parallel between the electrical network and the communication network, from the early years of the internet to the 2010s data originated from centralised servers which were distributed to many discrete users. Therefore, the internet was structured asynchronously favouring downloads to the users. However, as user-generated content is becoming increasingly popular, this paradigm is becoming outdated. Consequently, internet connections are moving towards synchronous links which are better suited to cope with upload bandwidth. In the same way, the electrical grid needs to change its physical infrastructure to be able to cope with increased remote electrical generation. The Orkney islands are a good example of this phenomenon, where wind turbines generate more power than the sparsely populated islands require, but the islands cannot export that energy back into the rest of the UK as the existing connection becomes saturated. As a result, the renewable electrical generation has to be turned off and potential energy is lost. A new connection system is currently being built in the Orkney islands, but will not be ready until 2023 [262].

The electrical supply also has to meet the demand of keeping the frequency stable, which requires communications at every generation site to signal to other grid components if they are increasing or decreasing production. Previously, large plants could monitor the change in demand and alter their supply accordingly, but with considerably more parties generating, all electrical generation and distribution will have to be coordinated. Moreover, Electric Vehicles (EVs) require a large amount of electricity in a short period of time. This is manifested in two problems. Firstly, once people arrive home, they will likely connect their cars to charge, creating a large spike in demand for electricity. Secondly, there will be a sudden drop in demand as users disconnect their cars; even more so if this happens at the same time for many users (e.g. to drive children to school). Without communications, the electrical grid would not be able to tell whether this drop appeared because the EVs are disconnecting, electrical generation has stopped, or a fault has occurred, rendering part of the network no longer functional. Previously, and somewhat incidentally, monitoring users behaviour through daily TV programming schedules, the grid could accurately predict sudden spikes in demand, such as when millions of kettles would be turned on during the intermission of popular TV shows or sporting events. However, increased demand in power resulting from individuals arriving home and connecting their EVs to charge, and decreased due to disconnecting once their vehicle is full, is much less deterministic, potentially causing a change in frequency and damage to electrical components.

Therefore, energy companies must meet these upcoming energy demands, along with additional objectives of increasing the energy efficiency of the grid, reducing carbon emissions, increasing the capacity for renewables' generation, and decreasing the number of consumer minutes lost — one of the key performance indicators UK utilities are measured on and fined accordingly by the regulator Office of Gas and Electricity Markets (Ofgem) [263]. This is a significant challenge and creates a financial, societal and environmental justification for increasing automatisation and monitoring of power distribution. However, to increase grid monitoring there is a need to increase the number of communication devices in the system. For example, the Energy Networks Association (ENA) plans to have nine hundred thousand devices connected to the distribution network by 2021 [264]. Six hundred fifty thousand of these devices will be required for the 11 kV distribution network to gain increased visibility of the network. As this PhD study was concerned with enabling secure low-cost communication rather than the sensing device itself a simple device was selected, an Fault Passage Indicator (FPI) to represent various utility monitoring devices. The FPI in question was designed to operate underneath an 11 kV overhead power line and measure the magnetic field generated by the line. Once there is a fault on the line, the magnetic field changes and the FPI would trigger a visual alert in the form of a flashing LED on the device which a line engineer would use to help dialogise the fault on the electrical line.

Faults on the 11 kV network account for a significant number of customer minutes lost per year. The two types of faults which can occur are a transient fault and a permanent fault; the former only records a brief disturbance in the line, while the latter identifies a persistent issue. Traditional systems rely on a visible indicators of detection of a fault current on the line, either a mechanical flag or a light [265]. Therefore, in the event of fault, the first step is to go along the length of the line and examine which FPIs are indicating the recent passage of fault current. Depending on the topology of the line and the geography of the area it traverses, this method typically takes hours to complete.

It was noted that each of these three use cases could be achieved using a light sensor: (1) care homes would use light as a proxy for occupancy, (2) streetlights would use it as a health indicator, and (3) electrical networks would use light to signal a fault. Since a light sensor connected to a communication module also represents a general IIoT device, the communication and security requirements of each of these use cases were investigated next.

5.2 IIoT Sensor Requirements

Communication Infrastructure

Although care homes may have a local network, such as a Wi-Fi network, this would not be provisioned for resilience. Hence, it would require a local network administrator to maintain and setup the network. Moreover, the energy consumption of Wi-Fi devices would make it challenging to deploy a sensor network [A7]. Therefore, a Wi-Fi network would be unsuitable, and another network would have to be created. Secondly, councils would require a region wide network to connect each of the lights to. Although some municipalities already have such networks, at the time of writing, this was an exception rather than the rule. Subsequently, the prevalence of this network would have to increase, in order to support such a smart streetlight use case. Finally, a nationwide network would be required for utilities. Particularly in rural areas as increasingly electrical generation is positioned away from population centres. Moreover, both cities and small villages both require electricity therefore monitoring.

Message Security

The message that a care home resident has not returned to their bed, is a simple message, with the intention to confirm lack of bed-occupancy. Therefore, the primary concern is if messages were not delivered residents being left unattended for the duration of the night. A secondary concern was that additional messages could be forged to request unnecessary visits from on-site nurses. This would be inefficient and over time would result in nurses being less attentive due to false alarms. The streetlights have a similar threat model, where fake messages could be introduced into the system to organise a unnecessary repair, reducing the effectiveness of the system. Messages could also be blocked to prevent lights being repaired. However, this would have limited effect, as the potentially the traditional system of members of the public reporting the outage would return.

The electricity distribution network, on the other hand, is a known target for malicious attackers [266, 267], and many known threats could bring the whole system down [266, 267]. However, the most dangerous attacks are those which scale to many devices and can be triggered remotely, as they can affect many devices simultaneously, without requiring any physical presence. Therefore, it is of critical importance to ensure that any future communication method used for utilities is secure. There are three distinct reasons why security in utility networks is of particular concern: (1) the perimeter of the network, (2) its interlinked nature, and (3) longevity of communication devices. Additionally, utility networks are also a target for attackers [13, 268]. Each of these will be briefly discussed with indication how low-cost devices can comply with these security requirements.

The legacy design of utility networks means that the primary security focus is on the perimeter of the network, as traditionally only the control room and the high voltage network had remote connections. These endpoints also have physical defences to deter against unauthorised access. However, as more distributed smart assets join the electrical network, these devices must be highly scrutinized, so that the new devices do not introduce a backdoor past existing defences and provide a launchpad for an attacker to reach the Operational Technology (OT) networks. OT networks are used to control the cyber-physical systems on the grid, such as breakers and switches. This network is specifically air-gapped from the Information Technology (IT) network where normal business operations would reside, such as consumer service, billing, payroll and HR. Separating these networks avoids attacks whereby compromising members of staffs' regular computer through a phishing attack would affect the operational network. However, if a device on the OT network can be compromised, the rest of the operational components on that network are at risk. This shows that if smart devices are going to be added to the electrical network they must have security features.

A second concern with critical national infrastructure networks, such as electrical grids, is the intrinsically linked nature of the network. Once sufficient electrical generation is prevented in a particular region, either through destroying equipment, as shown in the Aurora Generator Test [269], or by preventing generation equipment from operating as originally intended, possibly through false readings being fed into the system to cause the system to alter the output correspondingly, the generation will decrease. If this decrease in generation continues, the overall grid demand can surpass supply and the frequency will rise. Electrical equipment is designed to operate at 50 Hz within a strict tolerance; in fact, this frequency is so consistent that certain clocks use the grid frequency to keep time [270]. Therefore, all of the grid components must be synchronised, and a sudden change to that frequency can damage their mechanical components. Additionally, due to the interlinked nature of the electrical grid, it is possible a chain reaction would occur, damaging other parts of the network and potentially bringing down the whole network. Restoring the network after such an occurrence would be a very costly operation, both in terms of human resources and financial penalties, and require reliable communication between generating parties.

Thirdly, organising a restoration operation would require parties to be able to reliably communicate, however, most mobile base stations only have one or two-hour battery backups [271–273]. Soon after mains power is no longer being generated mobile phones would not be able to communicate. Additionally, the majority of landline phones require Alternating Current (AC) power, which would therefore in operational. Certain phones can operate from the power received from the telephone exchange using the copper phone line, however, after 2025, the traditional circuit switched telephone network is going to be switched off to be replaced with fibre optic connections. Meaning that landline communications would not be available either [274]. The National Grid UK Black Start Strategy aims to restore 60% of national demand in 24 hours, but there is currently no plan for how to achieve this [275]. Instead, most estimates believe it would take seven days to restore the grid. It is, therefore, essential to consider the threat model when adding new connected devices.

These requirements demonstrate that the utility use case is the most demanding both in terms security and coverage demands therefore it was decided to continue with this use case, as if the case could be proved for this use case it would also be applicable to the others.

Design Requirements

With these requirements in mind, a consortium of Distribution Network Operators (DNOs) and the author of this PhD study collaborated to select a suitable use case for a system which would provide secure communication to a monitoring device in a rural area. One monitoring device which increases the overview of the electrical grid is an FPI. The author decided to use an existing FPI for base of the design, Upgrading FPIs to use wireless communications which report the passage of fault current can dramatically reduce the time taken to determine the location of a fault to a few minutes, as personnel are no longer required to traverse the length of the line. Instead, the FPI itself would report the location of the recorded fault automatically. Although GPRS communications solutions already exist, this would require coverage in rural areas which is commonly lacking [276]. Furthermore, such a solution would require connecting thousands of devices to a GPRS network, which would be prohibitively expensive. Therefore, there is a need to consider other technologies. By adding automatic communication technology server-side processing would be able to overlay the existing power network, predict exactly where the fault occurred, and issue this information to local linesmen to resolve the issue more quickly. However, the disadvantage of wirelessly reporting electrical faults over the air is a natural trade-off in systems between cost and security. This is often a financial cost, but implementing security features also requires additional energy, therefore an energy cost as well. Hence, energy consumption should be considered too, so that the created device would be able to operate for years unattended. Finally, if these devices are commonly deployed within rural areas where there is little to no oversight, physical attacks become a feasible possibility. Therefore, physical attackers should be examined within the threat model.

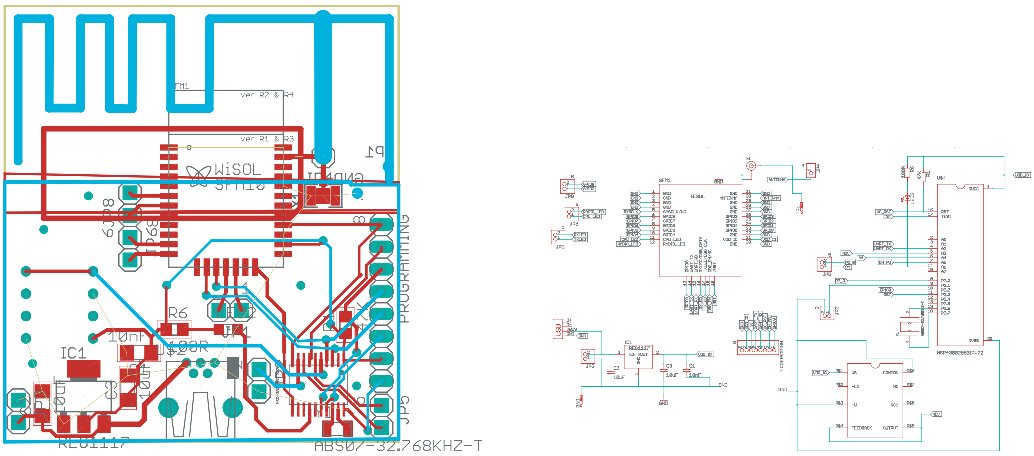
5.3 Evaluation of Low-Power Wide Area Network Technologies

LPWAN devices are designed to operate for years using very little energy consumption and without requiring human interaction. These devices run on batteries, making them easy to install and removing the necessity to rely on an external permanent power source. One of the ways this technology achieves such a long battery-life is by enabling the transceiver equipment only when the device has a message to send, reducing the amount of time the device has to be powered for. A similar idea can be seen when putting a mobile phone into flight mode and only enabling the radio to send messages, which would increase the phone's battery life. As one of the largest costs in deploying a wireless sensor network is acquiring licensed spectrum suitable for existing devices, it is important to consider what use cases require the exclusivity licensed spectrum provides. The three most well-known LPWAN technologies are NB-IoT, Sigfox and LoRaWAN, with NB-IoT operating in licensed spectrum while the latter two operate in unlicensed spectrum [277].

LPWAN devices provide a wide area coverage and a high link budget which allows them to be placed over a large geographical area and in rural or hard to reach places. Thereby, they remove many of the constraints with deploying sensor networks. These benefits have been recognised by the market and a recent report estimated that there would be 2.7 billion LPWAN IoT connections by 2029; much of this growth will be Machine-to-Machine (M2M) [278]. As the aim of this study focuses on low-cost devices, and LPWAN devices are designed to be low cost they are a good candidate to be examined further.

5.3.1 Sigfox

At the time of the investigation, Sigfox was the only wildly deployed LPWAN network therefore it was the first technology to be investigated. However, there was no published research pertaining to the security of Sigfox. Therefore, a custom-made PCB was created (Figs. 5.1 and 5.2) to test the security features and the overall suitability of Sigfox for a low-cost device. This was created with the previously presented use cases in mind; however, it would be applicable to other use cases if the system proved to be suitable. The PCB featured a general-purpose low-cost microcontroller, a Sigfox radio module, a voltage regulator and a light sensor. To keep the cost of the module low, an antenna was designed directly onto the PCB. Therefore, no external antenna would be required, reducing the overall device cost. This low upfront capital cost is what differentiated Sigfox from other LPWAN technologies available at the time. Sigfox subsidises the cost of their modules and supplies all of the capital intensive infrastructure, in exchange for charging their user a monthly fee — similar to how mobile phone contracts are charged. However, the Sigfox monthly fees are much lower than a comparable GPRS package, as Sigfox uses reduced data rates and operates in unlicensed spectrum.



(a) Board Layout

(b) Schematic Layout

Figure 5.1: Sigfox Printed Circuit Board

The primary disadvantage of Sigfox is that the coverage was only available in built-up areas. However, as this would overlap with most common location of care homes and streetlights, it would still operate well in these use cases. Whereas, the coverage would have to be expanded to provide connectivity to rural utility assets. The microcontroller was programmed to detect a significant change of light, by routinely sampling the light sensor. For the first use case, the care home, the light over the course of a day was recorded. As the ambient light levels would be higher for longer during the summer months rather than the winter months, it was important to use differential processing to detect a change, where a light would be turned on. This processing would have to occur locally on the device, because requiring server-side processing to calculate the change in light, would necessitate the device to transmit many messages per day, which LPWAN technology is not designed for. Each significant change in light detected by the sensor should be reported to the server. This may cause a false alarm where the sensor was temporarily blocked. Server-side processing could be used to assist in determining if this is likely a case for a nurse to be alerted.

When further examining the security of the system, it was discovered the Sigfox

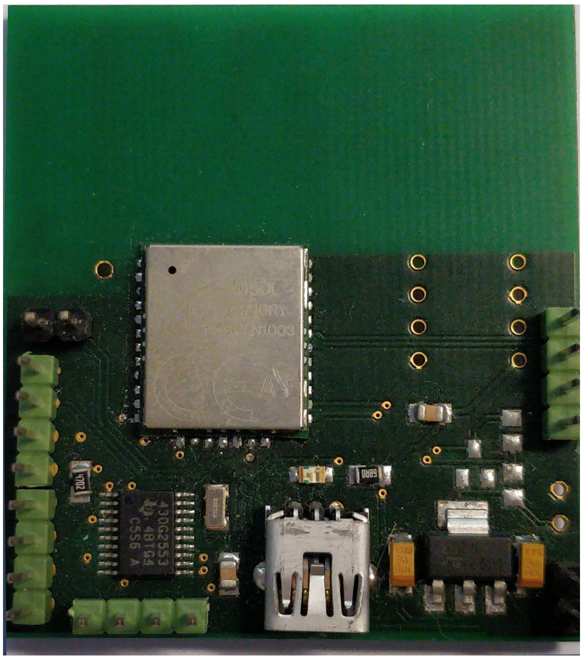


Figure 5.2: Sigfox Created PCB

payload of the message was not encrypted; it was only encoded and authenticated [279]. Sigfox subsequently published a white paper, which explained that payload was not encrypted and only encoded using a propriety scheme [280]. It is important to note that, depending on the threat model of the use case, encryption for the payload may not be required. Not encrypting the payload and only authenticating the message would save battery life and prolong the lifetime of the end product [280]. However, a study later published a number of observations which found that only a 16-bit MAC was used for authentication, and a 12-bit field was used for a sequence number [279]. Having these values so short meant that the sequence numbers could rollover and the MAC could be brute-forced, making the messages unsuitable for authentication. The same study also found that the message authentication keys were stored in plaintext in the regular device memory, so they could be easily accessed through physical inspection, enabling the creation of forged messages. This resulted in fundamental flaws in the security of the network, which were not addressable by this PhD study due to the closed nature of the system. Therefore, LoRaWAN was considered instead.

5.3.2 LoRaWAN

Given the PhD's study requirement for a secure low-cost communications device, it was decided to advance the utility use case for secure monitoring in rural environments. Therefore, it was decided, in conjunction with UK energy companies, to create a connected FPI using LoRaWAN to explore the suitability of such a system. LoRaWAN is available in three separate configurations: (1) class C; permanent bidirectional communications, (2) class B; periodic checking for downlink messages, and finally (3) class A; listening for messages only briefly after the device transmits the message itself. Using class A configuration increases the battery life, as the radio only has to be engaged when the device is transmitting, working under the same principle as what was discussed in Section 5.1.3. Given the possibly of limited mains power availability a battery-operated device would be best suited for reasons already discussed, consequently class A was selected. The high latency of the downlink messages makes control operations of utility assets a non-ideal candidate. Monitoring applications, on the other hand, only require predominantly unidirectional communications, which also matched the operating pattern of FPIs.

One of the major advantages of LoRaWAN over Sigfox is that organisations can configure their devices to join publicly broadcast networks, but they can also create private networks or hybrid networks - a combination of public and private. This is particularly beneficial when there is no public coverage available; private networks can be built to expand the coverage. One practical example of this phenomena is The Things Network, which is a shared worldwide public network which provides backend services for free. Individuals who wish to expand the network can register a gateway and increase their coverage. Subsequently, everyone in the range of that gateway can join the network and that device's packets are routed to the user who added the device. Using this system The Things Network have provided nationwide coverage in the Netherlands.

A feature of LoRaWAN is its ability to propagate long distances, getting communications into hard to reach areas. LoRaWAN uses a low frequency and chirp spread spectrum, which is resistant to multipath fading, giving LoRaWAN a high link budget. In fact, LoRaWAN managed to reach 766 km in one world breaking record [281].

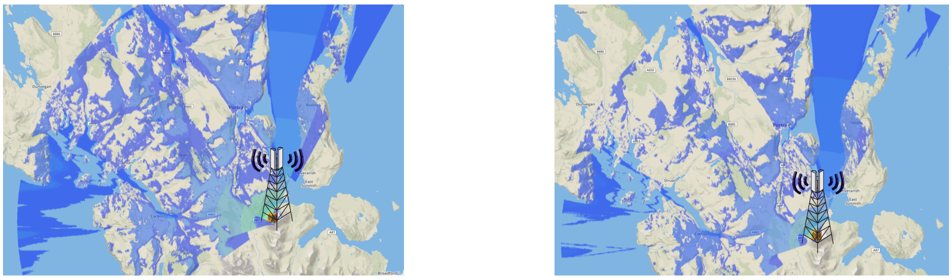
LoRaWAN is currently by far the most popularLPWAN technology utilising unlicensed spectrum [128, 282]. LoRaWAN has successfully built an ecosystem using unlicensed spectrum, thereby removing this cost. Combining this feature with the high community support, openness and low installation costs makes LoRaWAN a suitable candidate for industrial deployment. The downsides of LoRaWAN is that either a specialist is required to program the modules with suitably random keys, or, alternatively, devices can be purchased with pre-programmed credentials. However, the user would be reliant on the third-party organisation to have generated suitably random keys and responsibly dispose of the credentials once they have been securely transferred to the buyer. These problems are amplified for organisations which wish to monitor many assets, such as utility companies, where it would be a large undertaking to program all the required devices, even for a team of specialists.

5.3.3 NB-IoT

LTE based technologies may seem like a viable communication technology for utilities given their prevalence in modern society, and security features [283–285]. However, the Connected Nations report by Ofcom found that, currently, 5% of the UK did not have the most basic voice and text coverage, increasing to 12% in Scotland [276]. There have been various government interventions to try and reduce this lack of coverage by providing infrastructure to mobile operators. However, due to the lack of commercial incentive, this has so far been unsuccessful [135, 136]. This highlights the importance of ensuring connectivity in rural areas. Moreover, public LTE systems do not have the resilience and uptime required by energy regulators. Energy regulators demand 99.999% uptime [286], the equivalent to six minutes downtime every year, while last year public mobile network operators only provided between 99.50% (2 days downtime per year) and 99.92% (7 hours downtime per year) [276]. Therefore, there are presently no nationwide coverage communication technology available to utilities. Therefore, before monitoring on the grid can be increased, secure communication infrastructure will have to be created.

At the time of writing, private LTE networks for utility control operations are being considered [A5]. Consequently, they will be briefly mentioned here. One of the important characteristics of a 4G/5G system is which band the system it operates in, where a LoRaWAN or Sigfox device will always be able to connect to a complaint base station, 4G/5G devices do not as easily seamlessly connect. This is because LoRaWAN and Sigfox operate at one frequency. Therefore, any LoRaWAN/Sigfox device only has to support this single frequency. Cellular connections, on the other hand, operate in a wide range of possible frequencies, and as each additional supported frequency requires specific hardware there is a financial cost associated with supporting each additional band. This creates a situation where handset manufacturers will optimise for the most commonly deployed bands, such as Band 3 (1800 MHz), as this gives them the largest possible market. Consequently, the value of mobile spectrum at these common frequencies increases as there is an increased demand. This issue is compounded by handset manufacturers will not support additional bands to their handsets, until there is a proven demand. The outcome of this state of affairs is that unless the 4G/5G system can be deployed in a pre-existing hardware supported band, it would be challenging to find compatible handsets. Therefore, acquiring spectrum in a common band would be prohibitively expensive for a regulated industrial such as utilities, particularly when they are competing against a highly popular and lucrative industry.

In order to enable innovation and creation of smaller private networks Ofcom recently announced a shared access licence, where uses can apply for a geographically restricted license of licensed spectrum. This license applies to several frequencies, but 1800 MHz offers the best propagation characteristics. Therefore, the propagation of an LTE system at 1800 MHz is compared to LoRaWAN. Figure 5.3 shows that LoRaWAN manages to reach more remote areas compared to the LTE system. Private LTE solutions at lower frequencies would provide better propagation characteristics, but at time of research these were not commonplace, with negotiations ongoing between stakeholders and regulators for the use of 450 MHz, LTE Band 31, for private LTE deployments [287, 288]. Given these limitations of LTE based systems LoRaWAN was selected to provide the communication network for a simple IIoT device, and the utility network monitoring FPI was selected as it presented the most challenging situation.



(a) LoRaWAN Rural Propagation Model

(b) LTE 1800 MHz Rural Propagation Model

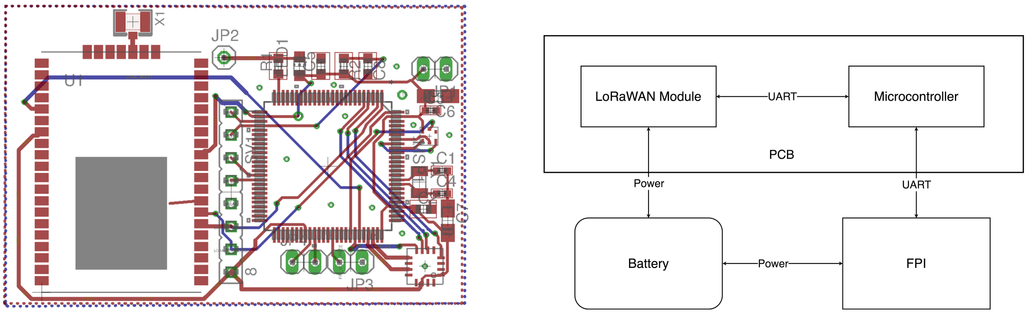
Figure 5.3: These Diagrams Show That LoRaWAN is Able to Reach More Areas than LTE at 1800 MHz.

5.4 Creating a LoRaWAN Smart Grid Sensor Device

The FPI was connecting to a PC via a USB-serial converter. A magnet was then used to induce a change in the magnetic field and thereby trigger a fault. The FPI reported that a fault had occurred over the serial interface. Taking advantage of this property, a microcontroller featuring a Universal Asynchronous Receiver-Transmitter (UART) was employed to receive this message. Through similar testing, it was discovered that the FPI could be configured through a Hayes AT command set via the UART to modify properties such as the sensitivity, the flash duration, and the threshold for a permanent fault. The next stage was to design a circuit board, which would feature a LoRaWAN radio module, to send and receive messages and a microcontroller to act as an intermediary between the FPI and LoRaWAN module as shown in Figs. 5.4 and 5.5

One area of concern, which was considered while developing the device, was the energy efficiency, as the final device would have to operate on battery power unattended for years. The design of the system took this into account and used dedicated low energy components to initially reduce energy consumption. Secondly, the microcontroller was programmed to enter low power sleep mode until triggered by the FPI, further diminishing the energy-use and prolonging the battery life. Another concern was keeping the add-on module's costs low, so that it would be practical to produce and deploy on a large scale. Considering that the cost of producing add-on module would be approximately £15 at a scale of 2000 units, compared to an FPI unit cost of almost £500, this was regarded as successful [A1].

It was decided to power the LoRaWAN module using the battery included in the FPI, since it was originally designed to last for 10 years, while powering the FPI circuitry, a GPRS communications module, and a powerful LED light. LoRaWAN consumes notably less power than GPRS. Hence, the expectation was that using LoRaWAN would significantly reduce the time to find the fault on the network, and therefore reduce the time the LED would be flashing for, making it viable to use the existing battery. The microcontroller software converted long, verbose FPI messages into short LoRaWAN reporting messages, provided heartbeat messages, and formed the verbose serial commands required to reconfigure the FPI, as shown in Fig. 5.4. Additionally, as the solution utilises the existing UART connections on the FPI and a 3.7V DC supply from the FPI's long life lithium thionyl chloride battery, the communication board could be retrofitted into existing devices. The created module is shown in Fig. 5.6.



(a) LoRaWAN-FPI Board Layout

(b) LoRaWAN-FPI Block Diagram

Figure 5.4: LoRaWAN FPI Board Design

5.4.1 Programming Device to Join Network

Another design decision was to program the device to automatically attempt to connect to the LoRaWAN network once powered up. If it failed to connect or

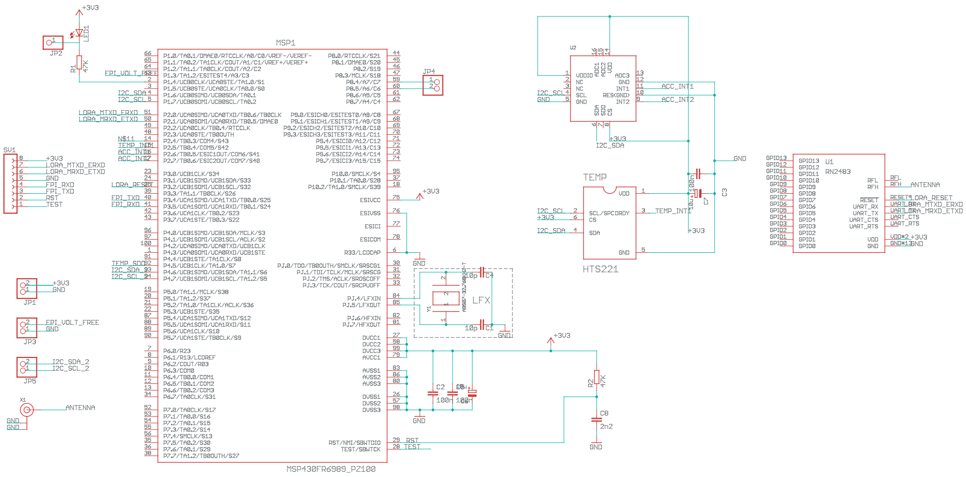


Figure 5.5: LoRaWAN-FPI Design Layout

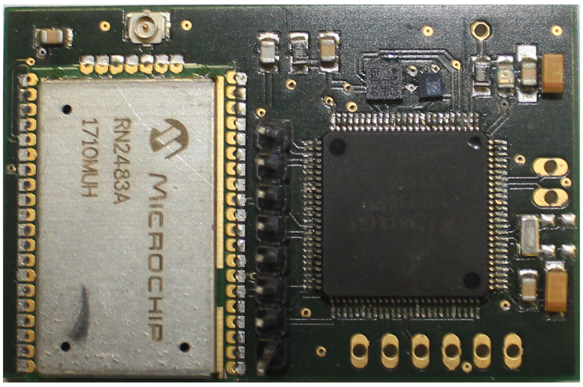


Figure 5.6: LoRaWAN Board

attempted to transmit a message while not being connected to the network, the device would periodically attempt to re-join the network, and would transmit any backlogged messages upon successful connection. Hence, if the LoRaWAN network was non-operational, the physical device would not need to be reset to become functional again. This is beneficial, as resetting the device would involve unmounting each device from its pole and power cycling it individually. Given their rural location and fixed position below High Voltage (HV) power lines, this would be an laborious task.

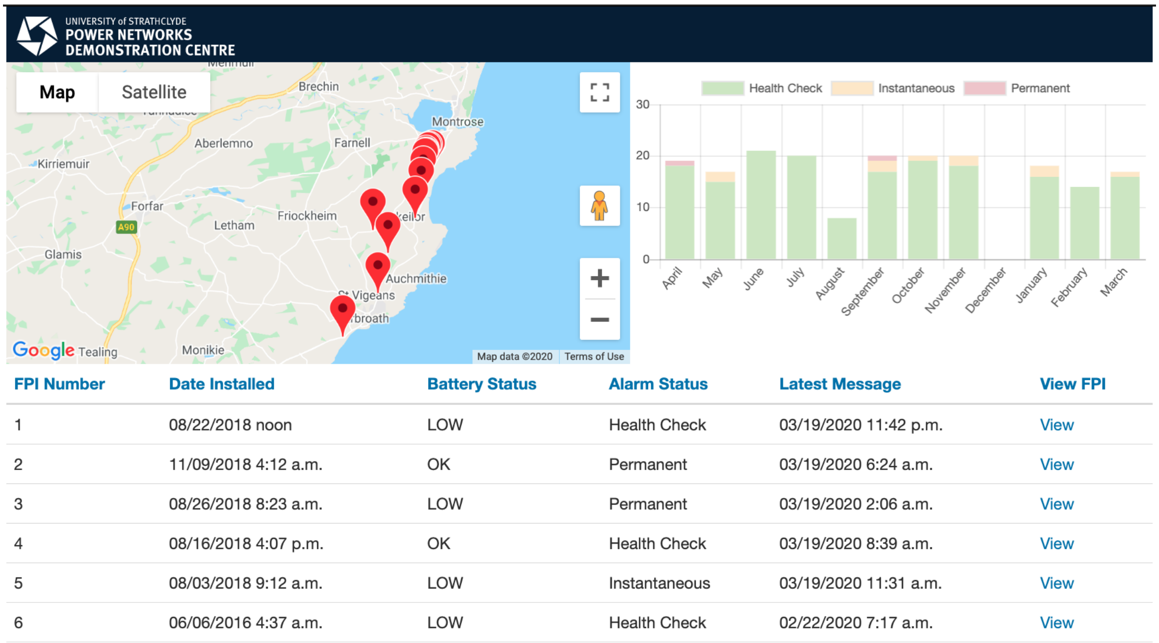
5.4.2 Programming Device for Dynamic Reconfiguration

The embedded software on the add-on module was then expanded to include remotely re-configuring the FPI over LoRaWAN using downlink messages. This was achieved by using a feature of LoRaWAN class A devices, where upon successful completion of sending an uplink message the device will listen for a short window of time for a response from the gateway. This response can either be an acknowledgement of the successful receipt of the message, or a message stored on the central server waiting to be delivered to the LoRaWAN device. Although this procedure is high latency, as messages can only be delivered after the FPI transmits, it greatly decreases the battery consumption of the end-device, as the device does not have to listen continuously for messages. Moreover, for this particular use case of LoRaWAN enabled FPIs, a reconfiguration delay is not a significant inconvenience. As misconfigured devices would be triggering prematurely, therefore transmitting and able to listen for a response. On the other hand, if the value was set to low, a daily heartbeat message would still cause the device to check-in every 24 hours, which, in most cases, would be faster than assigning an engineer to actually go out an inspect the physical device.

5.4.3 User Interacting with Sensor Devices

As part of the work, a web interface was developed which integrates with the LoRaWAN Application Server to display any faults on an interactive map, pinpointing the location of the issue quickly, reducing the number of customer minutes lost when a fault occurs. This web interface was designed to integrate with DNOs existing and future computer systems, for example, if the utility wanted to utilise an existing dashboard or create an accompanying mobile application, so that field technicians could be alerted and routed to the issue. The FPI line dashboard (Fig. 5.7) shows the general health of the electrical line and shows the status of each of the FPIs on the line. Fig. 5.8 shows the specific location of the fault and allows the user to remotely reconfigure the device.

Through the web application, a network operator's control engineer would be able to remotely reconfigure FPIs, to changing the trigger current, the trip delay, and the duration of LED flashing, to preserve energy when not required. The website would also



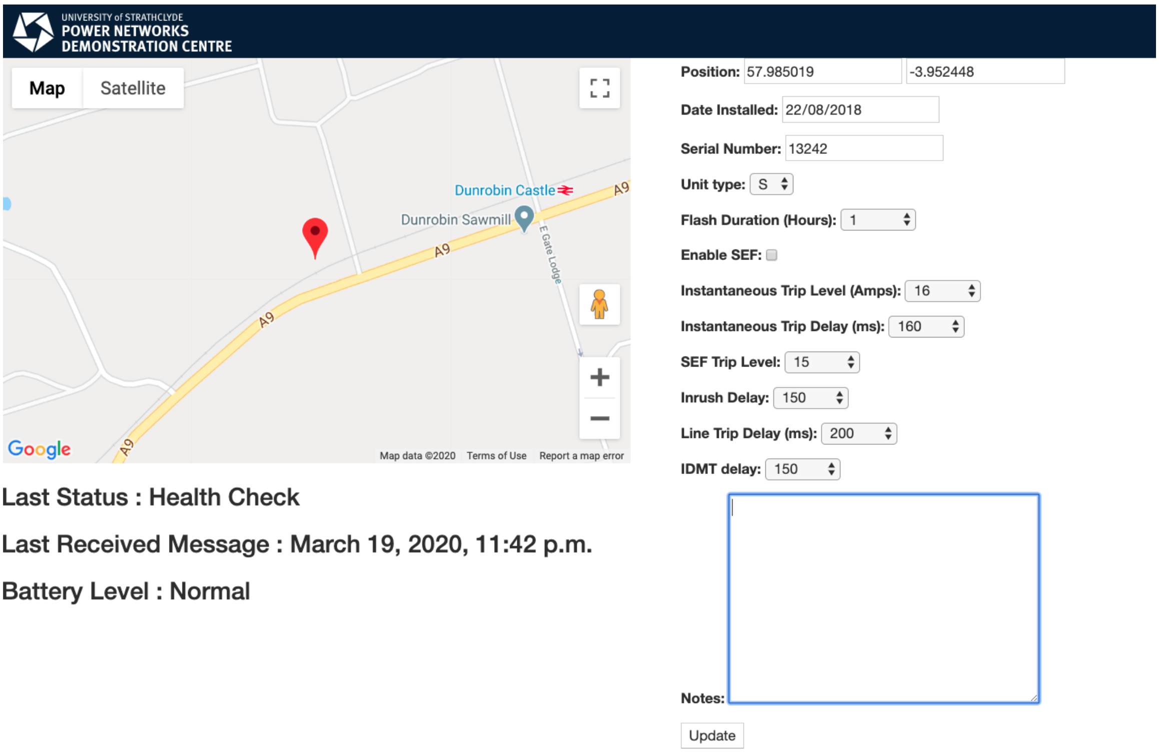
FPI Number	Date Installed	Battery Status	Alarm Status	Latest Message	View FPI
1	08/22/2018 noon	LOW	Health Check	03/19/2020 11:42 p.m.	View
2	11/09/2018 4:12 a.m.	OK	Permanent	03/19/2020 6:24 a.m.	View
3	08/26/2018 8:23 a.m.	LOW	Permanent	03/19/2020 2:06 a.m.	View
4	08/16/2018 4:07 p.m.	OK	Health Check	03/19/2020 8:39 a.m.	View
5	08/03/2018 9:12 a.m.	LOW	Instantaneous	03/19/2020 11:31 a.m.	View
6	06/06/2016 4:37 a.m.	LOW	Health Check	02/22/2020 7:17 a.m.	View

Figure 5.7: FPI Line Dashboard

detail the last message and the time of its receipt, which could be useful in situations of faults. For example, the type of fault occurred would inform the decision whether or not to dispatch an engineer. The web application also displays electrical lines in geographic segments, so the performance of these sections could be monitored over time assisting in anomaly detection.

LoRaWAN Server

Once a message is received at a LoRaWAN gateway it is encoded into UDP packets and then sent to a central LoRaWAN server. Device owners can then subscribe to receive notifications once any of their devices transmits a message. This is available on a series of MQTT channels and as a REST API, so that web or mobile applications could be developed to alert mobile workers to the site of the fault and provide them with any notes about the device in general, or station in particular.



Last Status : Health Check

Last Received Message : March 19, 2020, 11:42 p.m.

Battery Level : Normal

Position:	57.985019 -3.952448
Date Installed:	22/08/2018
Serial Number:	13242
Unit type:	S
Flash Duration (Hours):	1
Enable SEF:	□
Instantaneous Trip Level (Amps):	16
Instantaneous Trip Delay (ms):	160
SEF Trip Level:	15
Inrush Delay:	150
Line Trip Delay (ms):	200
IDMT delay:	150
Notes:	

Update

Figure 5.8: Individual FPI Page

5.5 Testing Smart Grid Sensor Device

To complement the lab testing, the LoRaWAN-FPI device was also tested using an isolated, 11 kV network; a subsection of the University of Strathclyde Power Networks Demonstration Centre (PNDC)'s 11 kV network. The test aimed to prove the constructed device would correctly operate on a real power network. The 11 kV network testing involved introducing faults onto the electrical network, while observing and verifying both the traditional visual response of the FPI and the LoRaWAN remote reporting. Initial tests were designed to affirm the standalone FPI operated correctly, followed by confirming that faults thrown on the network would be accurately reported back to the central control centre over LoRaWAN. Finally, the most crucial test was to certify that the reconfiguration of the LoRaWAN-FPI had correctly altered the magnitude of fault current required to trigger the device. This test had to be undertaken on a live 11 kV network, as lab tests could not accurately confirm the successful reconfiguration of the FPI, since the reconfigurable parameters of the device were so slight it was challenging to emulate the change required using only permanent magnets and human movement.

As mentioned in Section 5.4.1, the FPI device was configured to automatically join a LoRaWAN network upon receiving power. If this network was momentarily not available or non-functional, the device would periodically check and try to connect again; once the device was powered it would connect without external involvement. For the purpose of this test, the LoRaWAN network in question was provided interchangeably by one of the PNDC LoRaWAN gateways. The first was connected via a 3rd party Asymmetric Digital Subscriber Line (ADSL) line and the second was connected via a 4G cellular network, as these are the most common connections which would be provided when deploying a LoRaWAN network. Both of these gateways were configured to forward LoRaWAN messages to a LoRaWAN server, which would then be received by the web interface described in Section 5.4.3. That would increase visibility and clarity of incoming messages, showing simply if it was a heartbeat message, an instantaneous fault, or a permanent fault.

5.5.1 Experiemental Setup

A subsection of the PNDC 11 kV network was used to test the LoRaWAN-FPI. The testing utilised the PNDC HV network powered by the motor-generator set and the HV fault thrower. The circuit was loaded using load banks, and the fault thrower first threw a fault generating a fault current of 15A. In the event of an unsuccessful attempt at detecting the fault using the FPI, the fault was to be thrown again until successfully detected. A successful detection would be confirmed by observing the flashing white LED on the FPI. The LoRaWAN-FPI itself was physically mounted upon a standard 11 kV utility pole, between a mock impedance unit and a pole mounted voltage regulator, shown in Fig. 5.9. Substations A, D and G were used to apply maximum loading to the network. Initially, only a simple fault was tested to confirm the system was operating correctly, and then the FPI was reset to be ready for the next test.

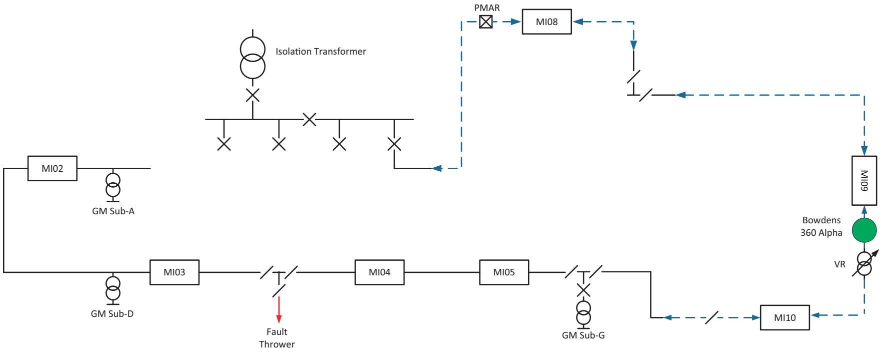


Figure 5.9: Transmission Powerline Network at PNDC

5.5.2 Test Results

For all tests shortly after a fault was thrown the device would illuminate, demonstrating the FPI had acknowledged the fault. The web interface would then detail the type of fault experienced, along with the time of the fault, and the location of the device in question, demonstrating the successful transmission of faults via the LoRaWAN network. The latency of the message was measured by recording the time faults were thrown and the time messages were received by the LoRaWAN Application Server.

This was found to be roughly 30 seconds, due to the fault detection logic of the FPI, where the FPI waits for potential upstream autorecloser[1] operations to complete before signalling a permanent fault. Subsequently, there is a delay before passing the message along to the microcontroller via the UART, before the microcontroller can action it and send it over LoRaWAN. The latency from sending a message from the LoRaWAN module to the web interface was found to be within a couple of seconds.

To test the reconfiguration procedure, a permanent fault was thrown at 15A and the FPI successfully triggered on the first attempt. The FPI was then reconfigured over the LoRaWAN network to trigger at its highest fault level of 45A. The HV network was reset and the previous fault of 15A was thrown again. The FPI did not trigger, as indicated by no illumination and no message delivered through the LoRaWAN network. The HV fault thrower was reconfigured again to throw a fault of 50A and the FPI triggered successfully. Between each fault thrown, the FPI and the network were reset to ensure results were not affected by any residual elements. Once all the tests had been achieved the system was reset once more and left to run in normal operation to confirm the FPI would periodically check-in using heartbeat messages while deployed. This sequence of tests proved that the FPI was successfully reconfigured over the LoRaWAN network, and that it would operate correctly on a live power network.

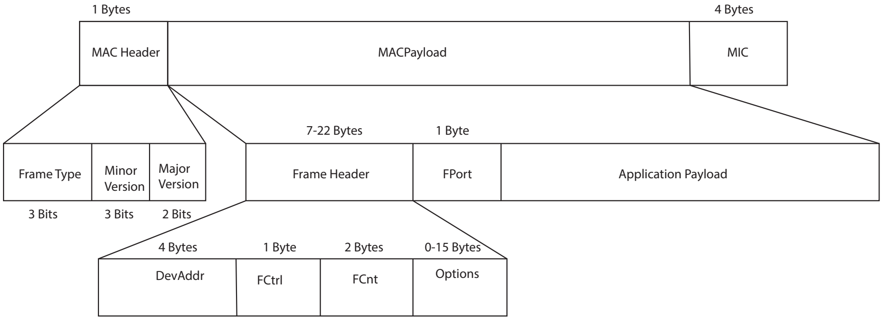


Figure 5.10: LoRaWAN Packet Structure

The size of the LoRaWAN packets used within this test from the FPI were a consistent 16 bytes; 14 bytes of LoRaWAN overhead and 2 bytes of payload (Fig. 5.10). The downlink packets to reconfigure the FPI were 22 bytes, as 8 bytes

[1] Electrical equipment to detect and interrupt momentary faults

were required to encode the FPI reconfiguration parameters. Therefore, in a real-world application, the most significant factor in data usage would be the frequency which the operator would configure their FPIs to check-in with a heartbeat message, as fault messages should not be common and barring currently unforeseen circumstances the devices should not require frequent reconfigurations.

5.6 Evaluation of LoRaWAN Smart Grid Sensor

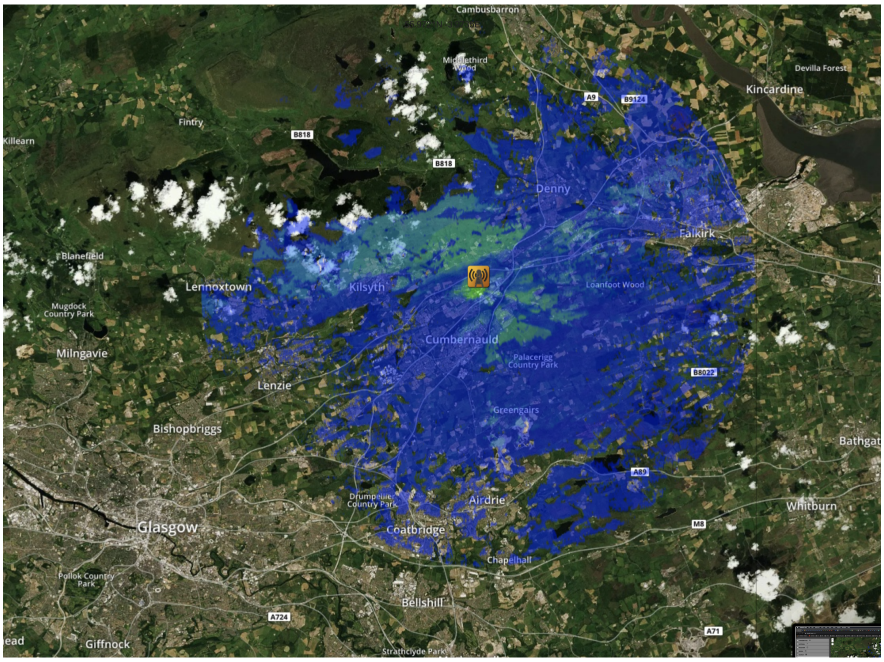


Figure 5.11: Predicted Coverage at PNDC

The created LoRaWAN enabled FPI has combined two technologies high on the technology readiness level scale by creating a custom add-on module. This module was tested and confirmed to operate as expected, enabling the remote detection of fault currents on 11 kV lines, through LoRaWAN. However, the practicalities of a deployment should be considered. Firstly, widescale deployment would require a considerable amount of labour to install all the devices. Secondly, physical location of LoRaWAN gateways should be considered as the geography surrounding the gateway will dictate its propagation and therefore its range. Consequently, it is best to position gateways on the highest point of the surroundings, where possible. On the other hand gateways require a permanent electrical connection and either a wired IP or a cellular connection. As these connections can be challenging and costly to install on hilltops, there is often a trade-off between optimum geographical placement and ease of installation, usually favouring the latter. Positioning gateways in less optimal positions was briefly investigated. Prior research examined this area and discovered that rural propagation would allow for tens of kilometres with line of sight, and LoRaWAN had high diffraction properties [289], (see Section 2.8.2). This was independently confirmed using simulation software as well, which predicted a 15km radius of coverage would be possible from PNDC in less than optimal conditions (Fig. 5.11). Considering the range and capacity of LoRaWAN, each gateway could support thousands of LoRaWAN devices. Therefore, a large proportion of the deployment cost would be for the PCB LoRaWAN add-on boards. However, this cost is small compared to the price of an FPI unit itself. Furthermore DNOs own a lot of existing network equipment that will not be replaced for many years, due to their regulated spending profiles. Therefore, the ability to retrofit part of the device would be a large business advantage, as the whole device would not need to be replaced. All of these properties makes LoRaWAN a viable technology to monitor utility assets in rural areas, and such a deployment would enhance the service provided to the end consumer, therefore, the created device was considered a success by the utility consortium. One aspect the project did not take into account was the labour required for a large scale deployment. This would be considerable as security keys would have to be provisioned into each of the devices.

5.6.1 Server Implementation

As LoRaWAN is a specification, companies can implement the details as they see fit. Consequently, it is critical to ensure that each implementation is secure before blindly trusting the system. Stream Technologies, acquired by ARM [290], operate a LoRaWAN dashboard where users can add their physical LoRaWAN infrastructure and then utilise Stream's user interface to monitor their LoRaWAN gateways and end-devices.

While studying this dashboard, it was discovered that the device up and down counters could be modified on the end device, and Stream's Application Server would accept subsequent messages with invalid counters, therefore allowing replay attacks making this specific implementation of LoRaWAN vulnerable. Figs. 5.12 and 5.13 shows both JavaScript Object Notation (JSON) packets with the same data and counters, but received time at the gateway (gw_time) differs, showing this LoRaWAN server implementation is vulnerable to replay attacks.

Figure 5.12: Stream's Output for Device 1

Figure 5.13: Stream's Output for Device 2

5.6.2 Metadata Collection

As LoRaWAN is an over the air technology, attackers can listen to LoRaWAN metadata from devices communicating. Inspecting both the metadata and the encrypted payload exposes trackable parameters as shown in Fig. 5.14.

Figure 5.14: LoRaWAN Packet Decoded

From this example, it is possible to track individual devices based on their DevAddr, and monitor the number of messages that the device has sent through the frame counter (FCnt). The port number (FPort) is user-programmable, designed to act as a simple way of differentiating different messages, possibly for different functions in a system. The type of message can also be detected, i.e. whether it is an uplink or a downlink message and if it is a confirmed or an unconfirmed message. Through analysing this an attacker could determine how many messages the recipient expects to receive confirmed and unconfirmed, and modify their subversion accordingly.

The encrypted payload, FRMPayload, shows how long the message is to the nearest 16 bytes, rounded up. If the message length is different for different messages, an observer can infer the type of messages the device is sending. For example, the message {"data":"on"} sent as a pseudo-random heartbeat message using 16 bytes would be easy to differentiate from the message {"data":"light_off"} using 20 bytes, as the latter one requires two blocks rather a single block for the heartbeat. Consequently, it is vital to pad all messages sent from a single device to be of equal length in order to minimise information leakage.

In LoRaWAN 1.0.2 join requests are transmitted in plaintext. Therefore, it is trivial to decode the join requests revealing a device-specific identifier. This identifier can be used to track devices. Moreover, the application ID the device is attempting to join can be seen, which can use segment devices based on their objective.

It is also possible to use three or more gateways, all within range of the target LoRaWAN devices, to approximately locate the position of the devices. This is possible because LoRaWAN gateways log the Received Signal Strength Indication (RSSI) of messages they receive and by using the device's address to recognise a single device from multiple gateways allows triangulation to give a rough indication of location. As more messages are recorded the group of gateways can also refine the predicted location. A more refined location is calculated by measuring the difference in time of arrival at each of the gateways in nanoseconds [291]. As the gateway measures these properties, any actor can estimate device position, regardless of whether that individual owns the end-devices or not.

5.7 Conclusion

With the increase of distributed electrical generation smart grids are required to ensure the efficient operation of the grid. One such smart grid technology is monitoring devices. Given the coverage and security requirements of electrical grids, LPWAN technologies are envisaged to be used to enable monitoring of smartgrids. Consequentially, this chapter evaluated popular LPWAN technologies. A custom made Sigfox module demonstrated that Sigfox's security and coverage were unsuitable for smart grid use case. Subsequently, a LoRaWAN device and network was constructed and analysed. This study found no vulnerabilities in the LoRaWAN specification itself, but highlighted several concerns with specific implementations. Accordingly, a LoRaWAN smart grid monitoring device, a FPI, was created along with user interface and tested on a live 11kv network. This device was then validated by UK DNOs, demostrating LoRaWAN is a suitable technology for smart grid use cases.

In evaluating the device one concern the DNOs rised was the provisioning of IoT devices. The large number of devices envisaged to be deployed on smart grids would be a large undertaking to manually program. Therefore, the next chapter will investigate alternative ways to provision IoT sensor devices.

Chapter 6

Provisioning Sensor Devices

Computationally-constrained sensor devices, struggle to take part in Public Key Infrastucture (PKI), and therefore given the large number of envisaged sensor devices automation would be required to provision these devices as scale. In fact, one of the discovered constrained for IIoT deployments is the secure key distribution [292]. Therefore, the chapter will describe a distributed deployment architecture for rapidly and securely deploying devices. Moreover, this was designed to use existing hardware to avoid increasing the cost of the devices, and keep the design simple to allow non-specialists to securely program devices.

As discussed in Section 5.3.2, the LoRaWAN ecosystem has built up community support, a thriving market, and became a de-facto standard. All these qualities benefit product longevity which is particularly important for industrial deployments. The costs of deploying a LoRaWAN system have also been discussed, yet two costs that remain for any generic LoRaWAN network are the costs of deployment and the potential cost of re-keying a compromised network. These costs are primarily for the labour required to configure and install many discrete devices over a wide geographical area.

Deployment costs can be minimised through purchasing pre-programmed modules. Yet, this would rely on a third party manufacturer, and if the third party was compromised or lacked suitable key generation, the deploying organisation would be left with a non-functional network and no ability to re-enable secure communication. They would have to remove all of the deployed devices and return them to the third party manufacturer, before deploying a new batch of devices, which is time-consuming and costly. This process is known as re-keying, and the costs originate from the cost of dispatching a specialist to the location of each device and removing a device from service or returning it to a central location. This is required because necessary credentials need to be programmed-in before any secure communication can take place, as these devices are not designed to cope with PKI.

As detailed in Chapter 2, deploying LoRaWAN devices requires both the device and the server to know a random shared key before any secure communication can commence. To this end, IO Active, a security company, released a report detailing that organisations should be careful when deploying LoRaWAN networks. They had discovered that many LoRaWAN implementers took the advertised built-in encryption to mean "secure by default". Therefore, implementers did not create random keys, but used default or easily guessable keys such as the device ID [128], thereby reducing the security of the system. Programming random shared keys into LoRaWAN devices is a challenge. While high-performance devices, such as smartphones or laptops, can automatically setup a secure connection without any prior data and remotely provide new devices with suitably random keys, low power LoRaWAN devices cannot generate keys with sufficient randomness to provide a strong level of security. Therefore, manual interaction from an installer is required.

The following section investigates a way to simplify installing shared keys into any generic low-cost LoRaWAN device, without relying on third parties. The proposed solution is applied to the existing LoRaWAN FPI developed in this chapter, although it would work equally well with other LPWAN devices which require credential input, such as Sigfox or NB-IoT using eSim.

6.1 Proposed Key Provisioning Architecture

The proposed system uses a smartphone as the basis for deployment. Smartphones are widely available, powerful, and understandable [293]. Using the smartphone all the required security credentials for the LoRaWAN device will be generated and transferred to the device. This makes the installation process similar to the one outlined in Chapter 4. However, as IIoT devices do not commonly need Wi-Fi, Bluetooth, or similar short-range over the air communication interface, they do not support these technologies. Instead IIoT sensor devices are commonly programmed through a wired interface and then deployed using the wide-area network technologies, which, by default, are not supported by smartphones. Therefore, other means of transferring the key were considered.

The features of smartphones outlined in Section 4.2 as a means of generation and transfer meant the that alternative methods of transfer were considered, bearing in mind the overarching aim of the architecture to use minimal additional hardware. Transferring the credentials to the LoRaWAN device could be undertaken using radio, as smartphones are fitted with an array of interfaces such as Bluetooth, Wi-Fi and Near Field Communication (NFC). However, this would require additional receiving hardware on the LoRaWAN device, as IIoT do not support these means of communication. In addition, new radio devices must be tested before they can be advertised and sold on the market [294]; a costly and time-intensive process. Physical methods of manually entering keys on to the LoRaWAN device would require a keypad, a relatively expensive addition unsuitable for many environments. In addition, such entry would be time-consuming and error-prone, given the length of the keys and the requirement to exactly match the server side. Instead, using light sensor is a simple and accessible solution for receiving credentials. Since most smartphones have LED flash, messages can be encoded using the flash and received by the light sensor on the LoRaWAN device.

One issue with this communication is that it is unidirectional, towards the LoRaWAN device. Hence, the device cannot respond once the key entry is complete. Although some sensor devices will have status LEDs or similar, which could be used to signal the receipt of the credentials, many do not. For this reason, the mobile phone monitors the LoRaWAN network for join accepts, signalled by the LoRaWAN server, and once the device has successfully joined the LoRaWAN network, the mobile application will be alerted, completing the process. The result is that the only additional hardware required for provisioning is a light sensor, a low-cost component both in terms in financial and energy consumption.

Consequently, the proposed solution is described as follows. The smartphone first receives the AppEUI from the deployment organisation's server using mobile internet. This can be done in advance of an actual deployment as these credentials are not device specific. Next, the smartphone generates the required credentials, namely the AppKey — and the NwkSKey, if the most recent implementation of LoRaWAN is being used — and transfers these credentials to the sensor device. The smartphone then sets up a secure session to the LoRaWAN application server over mobile internet, to transfer the matching credentials to the server. Finally, the LoRaWAN device repeatedly sends a join request to the LoRaWAN application server and, upon successful connection, the smartphone will be alerted that the process is complete over cellular internet.

This is detailed in Fig. 6.1 along with the following list of bullet-points:

1. The smartphone assembles required LoRaWAN credentials, such as AppEUI.
2. The smartphone transfers credentials to the LoRaWAN device via light
3. The smartphone transfers the credentials to the application server from the smartphone of a cellular or Wi-Fi connection.
4. The device initiates a LoRaWAN join request using the credentials.
5. The smartphone receives an acknowledgement that the device is successfully connected.

6.1.1 Testing Proposed Industrial Key Transfer

While most smartphones have an LED camera flash, it is not intended for the purpose of transmitting information, so some innovation in implementation is required. The LED can either be on or off. Three of the most prominent, yet straightforward transmission protocols, are On-Off Keying (OOK), Pulse Position Modulation (PPM), and Pulse-width Modulation (PWM) [295, 296]. OOK is the simplest transmission method, where the presence of a pulse represents one value (e.g., '1') and the absence

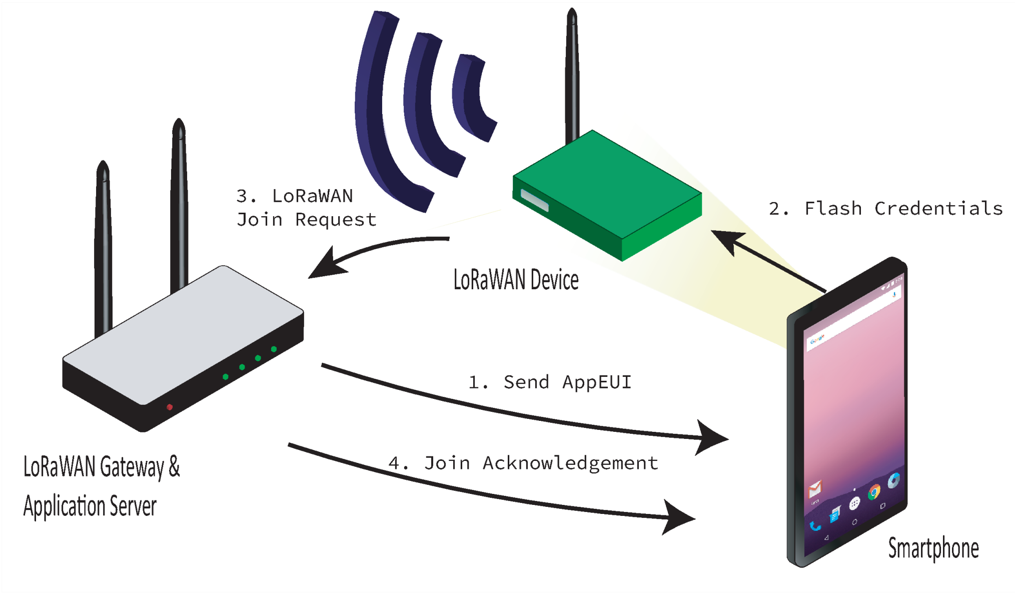


Figure 6.1: LoRaWAN Device Setup Procedure

of a pulse the other (e.g., '0'). PPM, on the other hand, uses the position within a particular time slot to determine the received value. The downside with this approach is that the transmitter and the receiver both have to be synchronised, so that the position within the time slot is known. Finally, PWM uses the duration of the pulse to determine the received value. As the very accurate timing of the pulse was impossible to determine, for reasons discussed below, the PWM method was selected.

Possible light sensors are a Light-Dependent Resistor (LDR) and a phototransistor. Phototransistors are generally cheaper than LDRs and have a faster response. To confirm these predicted characteristics, Android application was created to periodically toggle the light every 100 ms, and the phone was directed towards the sensor. The LDR took 40 ms to fully recognise the change in light level, while the phototransistor only required 10 µs. Given the faster response time and lower cost of the phototransistor, a phototransistor was used to create a prototype.

The operation of the LED flash on Android smartphones is controlled through the Android operating system and not directly visible to an installed app, which causes issues with accurate timing. For one test phone, if the app was configured to have the LED on for 10 ms, the LED would, in fact, stay illuminated for 16 ms. This

Phone Model	Set Light	Actual Light	Set Light	Actual Light	Set Light	Actual Light
OnePlus 5T	50	56	30	36	10	16
Google Pixel 3	50	55	30	35	10	14
Nexus 6P	50	54	30	34	10	14
Sony Xperia XZ2	50	54	30	34	10	14
Motorola Moto G4	50	55	30	35	10	15

Table 6.1: Phones Tested for LED Response Time

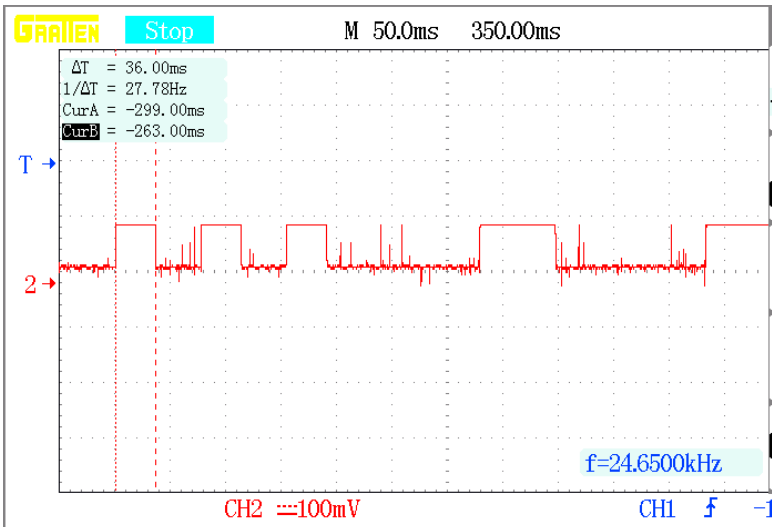


Figure 6.2: Phototransistor Light Detection from 30 ms LED flash

prolonged light was fairly constant, with similar results with the LED being set on for 20, 30 and 50 ms, each resulting in roughly an additional six ms of recorded light. The Android App was tested on various different phone models, measuring the light response (Table 6.1), but they all overran by four, five or six ms. Further tests were undertaken to confirm that this behaviour is predictable in all conditions. As part of the tests, all other applications on the phone were closed, assorted common applications were run, and the periodic sequence of the light being on and off was replaced with a randomly generated sequences of on and off pulses, mimicking randomly generated AES key. In each case, each programmed light period was followed by 4-6 ms of continued light. Fig. 6.2 demonstrates the case of the OnePlus 5T, flashing a key of 101010001100011, and showing that the width of the recorded pulse was 36 ms, while the programmed time for a pulse was 30 ms.

While relatively consistent at 6 ms, this delay was a combination of the response time of the smartphone's camera hardware abstraction layer and the Android scheduler, and how many tasks the phone was undertaking at the same time. Since the proposed smartphone key transfer solution was to be applicable to a range of smartphones and use existing hardware, it was not feasible to require the user to force close all other applications on their phone or require a specific handset. Therefore, very accurate timing will not be possible, which makes OOK and PPM unsuitable; consequently, low rate PWM was implemented. As discrete two symbols had to be encoded, each one with a different pulse length, one-third of the transmission time was devoted to representing a zero, and two-thirds of the transmission period representing a one. The total transmission time would have to be suitably larger than the largest overrun measured, so that the overrun would not change the measured result. A symbol period of 50 ms was chosen, allowing a transmission rate of 20 bit/s, making each third of a pulse 16.7 ms long. This is significantly larger than the expected overrun of 6 ms, so that a symbol could not overrun into the next threshold and be misinterpreted. Figure 6.3 shows the two-thirds pulse has a 17 ms buffer before it would run into the next period. Similarly, the one-third pulse has a 16 ms buffer before it could be considered a long pulse. Both buffers are more than double the highest recorded overrun. A PWM sequence of 1010 is shown in Fig. 6.4.

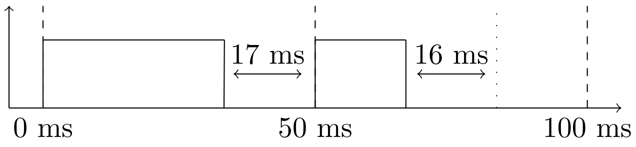


Figure 6.3: Pulse Width Buffer for Overruns

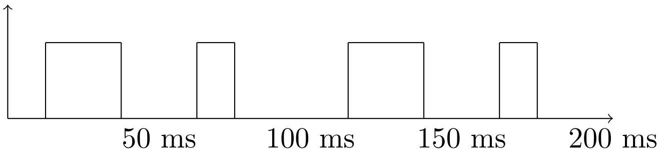


Figure 6.4: Pulse Width Modulation Sequence of 1010

The Android app was originally programmed to loop through the message one bit at a time, switch the light to the corresponding state and then sleep for the duration of the pulse. However, this resulted in the overrun time being added to each system call, (Fig. 6.5); instead of the first symbol ending at 50 ms, it would end at 56 ms. Therefore, by the completion of the tenth symbol the elapsed time was 560 ms. This meant that while the receiver expected 11 symbols to have been transmitted within this time, only 10 would have been sent, creating an inconsistency in the sent and received signals. This discrepancy would be variable for different hardware. A potential solution would be to extend transmission time of an individual message until the receiver had enough redundant data to discard any mismatch, but this would increase the overall setup time. The more effective solution was to modify the Android application to pre-schedule the toggle instructions, so that even if a small error was introduced it would not cascade into all of the symbols, rather it would be rectified on the next symbol. Employing such a revised system would make the first pulse end at $t = 50$, the second at $t = 100$, and the tenth at $t = 500$. Allowing for the receiving to decode the symbols without synchronization.

$$delay = (50 \times (1 + index)) - (start_{time} - loop_{time}) + 4 \tag{6.1}$$

Figure 6.5: Code for Basic Toggling Light On and Off.

Android has an Alarm Manager built into the Android Software Development Kit (SDK) specifically for periodic tasks, such as the proposed pulse system, however this library is unsuitable for sub-second precision, as Google opted to let the operating system modify the pre-scheduled trigger time to better optimise battery life [297]. Instead, the Android application was programmed to use an array of Runnable tasks staggered to set the corresponding light value based on the position in the message. This modified configuration resolves the issue with the first few samples triggering at the expected time, but a slight linear delay was introduced after the first flash. This was due to the time taken to execute the loop, causing each subsequent iteration to be set slightly after the correct time. To resolve this issue, the algorithm was modified to subtract the time taken to execute the loop as shown in Equation 6.1. The algorithm is shown in Figure 6.6.

Figure 6.6: Android Light Scheduler

6.1.2 Implementation of Proposed Key Transfer Architecture

To test the light key transfer solution, a PCB featuring a phototransistor, a MSP430FR6989 microcontroller and a LoRaWAN module. On start-up, the MSP430 was programmed to routinely sample the phototransistor converting the detected value to a digital representation through the built-in analogue to digital converter. These samples were stored and once the rolling average of the 20 previous symbols was deterministically higher than the average of the 50 prior symbols, this would signify the message preamble has been detected and initiate recording of all subsequent samples. Configuring the microcontroller to sample the phototransistor every millisecond meant that for each transmit period 50 samples would be taken. Through analysis of each of these groups of samples, a long or a short pulse would be determined. Once the user positions their smartphone camera flash over the phototransistor and initiates the process, the app creates a binary string the preamble, credentials, and Cyclic Redundancy Check (CRC) check. First, 1 second worth of preamble is added to the string, then an Appkey is securely generated using well-established crypto APIs [235]. The AppKey and previously received AppEUI are converted to a binary format, and a 16-bit CRC applied [298]. This result is then appended to the preamble string to create the overall message string. The app then schedules a series of delayed tasks, each staggered by 50 ms, to light the camera flash either for a long or short pulse, depending on the corresponding bit in the message. Concurrently, the Android app also initiates a TLS connection to register the device on the LoRaWAN application server using the phone's cellular connection to transfer the matching credentials.

The firmware on the sensor device would know what kind of data it is expecting from the flashing light. Therefore, after the initial preamble message, the sensor device will know the message length without requiring synchronisation. After this expected period has elapsed, the cyclic redundancy check of the message will be validated. If the message is successfully validated, the credentials will be extracted; otherwise, the sensor device will reset and continue waiting to be configured. Using the received credentials, a LoRaWAN join request will be attempted. If successful, the application server will issue an acknowledgement to the smartphone. If no such acknowledgement is received, potentially due to credentials mismatch in the join request, the app will reset and start flashing the same message again, until successful acknowledgement is received. At this stage, the companion app may also automatically detect additional information about the installation, and feed it back to the application server, for example, the GPS location of the installed site.

6.2 Evaluation of Light Key Transfer System

During a full system test, the created LoRaWAN sensor device was provisioned in 17.1 seconds, without the need for specialist tools or knowledge on the part of the phone operator. This was broken down into 1 second for the preamble, 12.8 seconds for transfer of AppKey, AppEUI, and DevEUI, 0.8 seconds for transfer of CRC check, and 2.4 seconds for saving the credentials to the microcontroller, checking the CRC, joining the LoRaWAN network, and issuing a confirmation message to the Android handset. LoRaWAN 1.0 was used for the implementation, but LoRaWAN 1.1 would be compatible with a slightly longer programming time due to the additional Network Server Key.

The smartphone transferring the light ran an Android application on an OnePlus 5T, which took minimal Central Processing Unit (CPU) resources (requiring approximately 1% of CPU), and the whole process took 956,416 us of processor time, with CRC generation and AES key generation taking 12,452, and 1,237 us respectively. It also only required 80mb of RAM to operate, with 8.2 MB of code, 31 MB of graphics, and 30.4 MB of native C/C++ code used to interact with the underlying hardware. This meant that the resource requirement was negligible for the device tested, and the majority of modern phones would be able to run the application [299]. Another area of consideration is the energy usage. Although the actual running of the application itself was minimal, it would require $\approx$ 1.7 mAh of energy to light the LED for a key transmission [300] depending on the exact number of ones and zeros in the message. The energy usage was low compared to the capacity of modern mobile phone batteries as this allowed for an estimated 1000 devices to be programmed before the phone would need to be charged [299].

Aside from allowing non-specialists to program devices, the proposed key transfer system transmitter only uses visible light, and the receiver only uses a passive light sensor and pre-existing hardware. Therefore, neither the receiver nor transmitter would be required to undergo RF Regulatory Compliance Testing; a legal requirement for any new device using radio frequency [294]. This would save considerable time and cost in getting the device manufactured.

The system was implemented using LoRaWAN 1.0, as this is the most common implementation at the time of writing [301]. However, as the proposal can transfer arbitrary data, the setup procedure can be modified with a software update to improve device longevity. This could involve using LoRaWAN 1.1 or adding additional security keys to the microcontroller itself. For example, although LoRaWAN only natively supports AES-128, the MSP430FR6989 contains an AES-256 hardware accelerator, which could encrypt the payload before passing it to the LoRaWAN module for transmission. The move to AES-256, although not required at time of writing, may become the norm with the increase of qubits in quantum computers, and the possibility of Grover's algorithm being successfully implemented [302]. Utility companies, in particular, would benefit as they require equipment to be operational for decades.

While an additional component (a phototransistor) is an additional cost it is a low-cost at less than 5% of the cost of the raw materials of the device, and additional software would be required to interact with the sensor device this is a minimal addition, and the cost of the alteration would be shared between all of the required devices. Therefore, the financial considerations are met. The provisioning software will only run immediately after the device was reset and be disabled afterwards. The phototransistor circuit can also be disabled to save power after provisioning; the ability to disable IO pins is inherent in the MSP430 microcontroller and requires no additional hardware. The expected impact on battery life is, therefore, negligible, meeting the low energy considerations.

To test the reliability of the proposed system in different conditions the PCB sensor device was programmed in a laboratory in a variety of conditions from brightly lit during the day to evening conditions and artificial light. In all cases, the system was easily able to distinguish between the LED flash and background light. A feature of the PWM implementation means that the proposed design is both hardware and software agnostic, any mobile phone featuring a flashlight can be reliably used to program devices without having to re-calibrate any equipment, as the light sensor is self-calibrated through measuring the ambient light and then recording any values consistently higher.

As the installation process for these devices is accessible and straightforward, it allows individuals without specialised tools or training to install them, reducing the cost and time associated with creating the network as a whole. Particularly for utility networks, blank LoRaWAN devices could be stored in maintenance vehicles or distributed to individuals, and as maintenance engineers would commonly visit the same areas as the sensors which would be deployed to undertake daily maintenance of the infrastructure, they could also be deploying sensing devices. Moreover, by distributing the provisioning workload through a larger proportion of people, devices can be more rapidly deployed, and if the central key server was compromised, devices could be re-keyed faster than the traditional design of sending each device to the manufacturer.

The design has an apparent weakness in the fact a mobile data connection would be required to program the devices. In fact, only the confirmation message requires an active mobile data connection for the smartphone. An installer can either provision the device in advance, while approaching the site if they are aware of the lack of cellular coverage, or as the sensor device is programmed to automatically attempt to join the LoRaWAN network. The installer could install the device a receive the confirmation once they return to cellular coverage or otherwise communicate with the control centre to confirm the join was successful.

6.2.1 Security Evaluation

The proposed key transfer system only alters the method for generating and transferring key into the LoRaWAN module. Therefore, this section will explore the benefits and limitations of that method, and the security features of LoRaWAN itself will not be considered. As the credentials are transferred using visible light, it would be possible for an attacker to record the deployment using a high speed video camera to decipher the key. Security in visible light communications is an evolving field [303]. However, due to the contained nature of the proposal, only using a single fixed intensity transmission LED and a receiving diode, this is an on-going research question [304]. However, this attack can be mitigated through the product casing, providing a circular housing for the receiving phototransistor to be protected from onlookers, or by having the installer position themselves anyway from any surrounding spectators.

Given the predominantly unattended nature of LoRaWAN devices, an attacker could also factory reset a device and attempt to program the device themselves. However, without a corresponding key held on the LoRaWAN server, the join request would fail and prevent the device from joining the network, meaning no malicious data would be injected into the network.

A physical attack could result in a malicious actor obtaining one of the development organisation's smartphones capable of programming devices, and therefore adding new devices to the system. However, this can be prevented by using a passcode or biometrics, a common protection scheme used to secure other systems [305]. Moreover, once such a device is reported missing or tracked as unaccounted for through *find my phone* or similar scheme, it can be blacklisted. Additionally, this would have minimal effect on the existing network as keys only remain on the smartphone until they can be transferred to the LoRaWAN server, and the API on the server only accepts adding keys, with no provision to extract keys.

Another potential attack would be malicious applications running alongside the

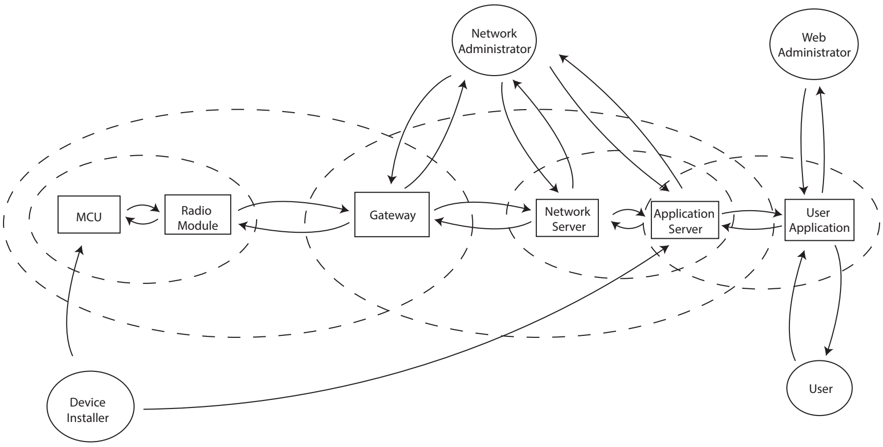


Figure 6.7: LoRaWAN Threat Model

provisioning application. However, modern mobile applications benefit from built-in security features such a sandboxing to prevent other applications on the installer's phone from extracting the keys while programming the device [242], and the installed keys only remain on the device until they have been securely transferred to the LoRaWAN server, at which point they are removed. Smartphones also have a well-established set of API for cryptographic operations to ensure sufficient entropy generation for strong key generation [235].

Applying the key transfer architecture to a LoRaWAN, the possible threat actors are the installer, the user, the network provider, the application provider and external actors. The device installer would have access to that specific device's credentials, but without valid user credentials the installer could not load the matching credentials into the application server to add a new device to the network. There is not capability to retrieve credentials from the Application server, therefore an impersonator would not be able to compromise existing keys through impersonating an installer. This is detailed in the LoRaWAN threat model diagram Fig. 6.7, where each dotted line is a threat boundary and the circles represent users interacting with the system, and the arrows represent data interaction. Sensor data recorded by the microcontroller is passed to the LoRaWAN module, where it is encrypted and authenticated, and secure until it reaches the LoRaWAN application server. This means that for a widescale compromise either the website or network administrator or would have to be compromised.

6.3 Conclusion

The proposed system allows non-specialised individuals to easily program low-cost, low-performance devices, which commonly rely on symmetric-key cryptography, such as LoRaWAN. As the security of these devices depends on the cryptographic keys installed, it should not be configured with default or easily guessable keys. For large scale deployments, particularly in critical national infrastructure such as utilities, the cost and security of a deployment is critical. The proposed system removes the complexity overhead associated with setting up devices securely, reducing the incentive to deploy devices using default or easily guessable credentials. Moreover, since the system does not require any credentials to be pre-loaded into the devices, the credentials cannot be targeted before installation. Hence, devices can be freely distributed without fear of the keys being intercepted. Simplifying the installation process would allow the configuration workload to be distributed across various parties, reducing the cost and time associated with it, and allowing the network to be re-keyed faster. The distributed nature of the installation process also removes a potential attack vector, as a malicious attacker would have to visit each sensor location or influence every installer. As the system uses a smartphone, which is very common and can generate suitably random keys, the additional cost of the hardware on the IoT device is minimal and requires few resources of the micro-controller. Therefore, the system alleviates some of the concerns for deploying sensor networks in real environments as the cost is reduced, while improving security, by enabling the system to be re-keyed, if compromised.

Chapter 7

Remote IoT

Many recent consumer products have heavily advertised their internet-based functionality as a key selling point and differentiator of their product. One prevalent design with such systems is using the user's existing Wi-Fi network to leverage connectivity within the home [306]. As Wi-Fi is now ubiquitous, products supporting Wi-Fi can be installed onto an existing network with minimal configuration, improving user experience. IoT devices leverage this existing connectivity to initiate an outbound connection to a vendor-operated server, often in the cloud, this is primarily done to bypass NAT, which by default blocks all incoming connections to devices. This design improves the security of devices within the home as by default no external devices can access private devices within the home. This architecture worked especially well as domestic computers commonly only downloaded information from organisation's servers thereby not requiring any direct inbound connections. IoT devices however are designed to be controlled from outside of the home network, therefore for users to control their devices externally NAT must be bypassed to enable inbound connections. This is achieved by the IoT device connecting to a central server and then all commands from client applications, such as smartphone apps, are issued via the server to the device Fig. 7.1. This advantage of this approach is its disregard for the physical location of the user; as long as the user has internet connectivity, the system will work. However, there are various downsides using this existing paradigm: (1) it creates a reliance on an external server to maintain use of

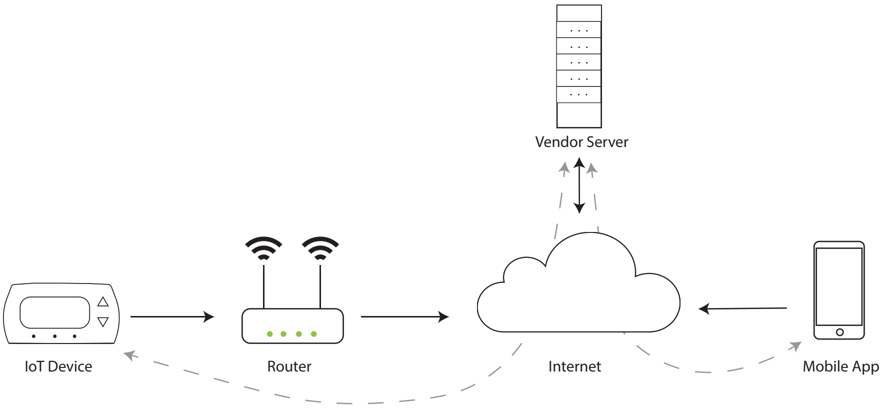


Figure 7.1: Domestic IoT and Wi-Fi Network Architecture.

devices; (2) Malicious actors outside the house can control IoT device within the house; (3) IoT devices can access both the internal and external networks.

Firstly, the reliance on an external service. This reliance can be severed in various ways: the external service can be compromised, rendered inaccessible through a fault, or no longer maintained by the provider [175, 307]. These can be summarised as longevity concerns, where the underlying service may not last as long as the physical product would otherwise. Secondly, several devices have neglected to feature security precautions [29, 30]. This includes carrying out ineffectual authentication or using hard-coded default authentication credentials [308]. One such example was seen with IoT security cameras which were advertised as secure, yet the FTC found inadequate security had been employed by the manufacturers [29, 30]. The first manufacturer used default passwords for its cameras, and made them globally accessible with knowledge of the cameras IP address, which was easily discoverable through services such as Shodan [309, 310]. The latter manufacturer “failed to take steps to address well-known and easily preventable security flaws”, including command injection and plaintext user credentials [30], while examples show that even products marketed as secure, may not be secure, and without auditing the devices themselves, users would be unaware of the potential security issues. Finally, there is a fundamental flaw with the current domestic security model. In the current setup, everything inside the network is trusted and freely able to communicate internally, yet everything outside the network is considered a threat, as shown in Fig. 7.2. The outcome from this is that unsolicited incoming connections to internal devices are blocked by the firewall, while outgoing connections from those same devices are left unmonitored. Creating an opportunity for devices to initiate unauthorised connections. This was demonstrated with the Mirai botnet [311], where over 145,000 compromised cameras and DVRs, each held on private networks, participated in a distributed denial of service (DDoS) attack generating in excess of 1.5 Tb/s [312]. Each device typically produced between 1 and 30 Mbps of traffic, as most devices were connected to high speed internet connections. When all this traffic is combined it was sufficient to bring down several high profile organisations such as Netflix, Twitter, Reddit and CNN [313].

IoT devices operate in this way, not because of malicious intent, but because of the status quo. For the device to function, it has to be connected to a Wi-Fi network, and with the current security model, that means it is trusted Fig. 7.2. However, where laptops and smartphones which previously were the primary devices connected to these networks commonly receive security patches, IoT devices commonly do not [4, 13, 172, 173], which brings into question their overall trustworthiness. Limiting both external internet and internal access of these devices would prevent devices from joining botnets, exposing unintended information, and spying on other devices [3, 314]. These ideas were shared by Notra *et al.* who highlighted potential security benefits of segmenting IoT devices from other devices on the network, and restricting IoT devices internet access [189]. To attempt to resolve some of these issues this paper describes a proposed solution to enable IoT devices to securely communicate, while preventing inter-device communication, compromised devices performing Distributed Denial of Service (DDoS) attacks, and reduce the reliance on an external centralised server.

IoT devices operate in this way, not because of malicious intent, but because of the status quo. For the device to function, the path of least resistance is connecting it to the Wi-Fi network, and with the current security model, that means it is trusted. However, where laptops and smartphones which previously were the primary devices connected to these networks commonly receive security patches, IoT devices commonly do not. Section 3.2 detailed the issues with IoT receiving software updates, which brings

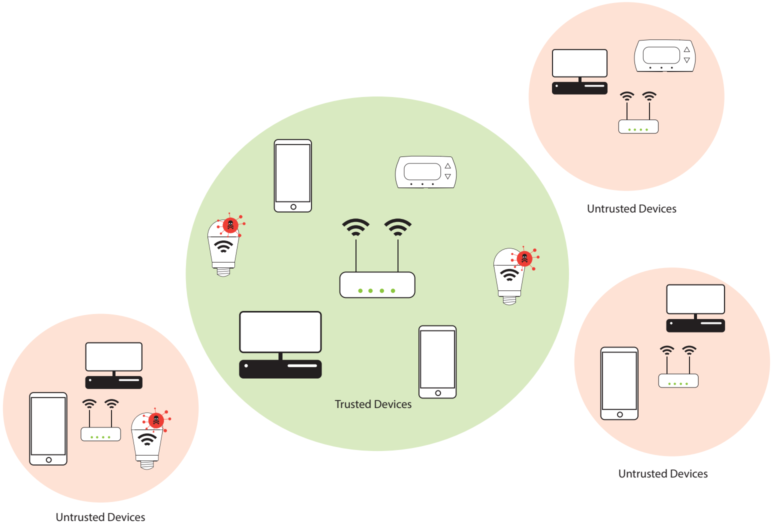


Figure 7.2: Identical Devices Are Trusted or Untrusted Based on Their Network Connection. Devices on the Internal Network Are Trusted While External Devices Are Untrusted

into question their overall trustworthiness. Instead, limiting both external internet and internal access would prevent devices from joining botnets, exposing unintended information, and spying on other devices [3,314]. These ideas were shared by Notra *et al.* who highlighted potential security benefits of segmenting IoT devices from other devices on the network, and restricting IoT devices internet access [189]. To attempt to resolve some of these issues this chapter describes a proposed solution to enable IoT devices to securely communicate, while preventing inter-device communication, compromised devices performing DDoS attakcs, and reduce the reliance on an external centralised server.

7.1 Decentralised Architecture

DHTs are known for their use in BitTorrent, InterPlanetary File System (IPFS) & Ethereum. Each of these systems connects millions of users together without requiring a single centralised computing architecture [315–317]. IoT devices by definition are required to communicate; in industrial environments sensors report to monitoring platforms, and in domestic environments smartphones interact with appliances. In a traditional architecture these interactions are achieved through routing traffic through an intermediary server [233, 318–320]. A side effect of this architecture is that IoT vendors must maintain server resources to enable the devices to communicate, and as the number of devices increases so does the demand for server resources. This demand is also often variable and difficult to predict when the IoT vendor is designing their product. An existing solution employed by websites in particular uses microservices [321] and using cloud computing platforms which can be scaled up to handle load, or down when no longer required. The advantage of this type of architecture is if there is a sudden increase in demand the site administrator can match this demand, and once the demand tails off the resources can be reduced. This works well for commercial websites as they can offset the additional server costs through advertisement, or user conversion [322].

IoT communication infrastructure on the other hand, predominantly only increase, and the current business models employed do not cater to long term use of backend servers. Therefore, without creating an additional revenue stream maintaining this communication infrastructure becomes a financial burden for the vendor. This was demonstrated in early May 2020, where a prominent IoT device vendor announced they would be moving there 4 million devices to a subscription based model otherwise the devices would cease to function [307].

Distributed architectures on the other hand, allows every computer in the network to self-organise and communicate without a centralised record structure, thereby removing any requirement for maintaining servers only to allow devices to communicate with one another. One such distributed architecture is a DHT where hashes are used as identifiers for storing data in the system. Computers in a DHT network are known as nodes, and once a node initially connects to the DHT it will first contact a bootstrap node at a fixed IP address, to get the location of another node in the network. This second node can then be used as an intermediary to connect to its neighbours to discover the whole network. Data published to the DHT is uniformly randomly distributed across all nodes in the network, as shown in Fig. 7.3. As the data is randomly distributed, receiving data from the network is achieved by the requesting node asking its neighbours if they

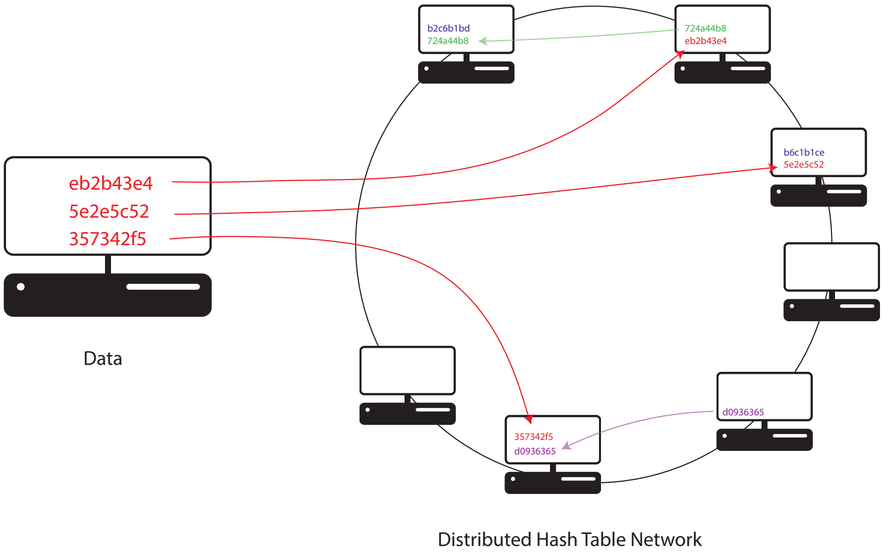


Figure 7.3: Distributed Hash Table Automatically Administering Data Throughout a Shared Network

have the requested data, the neighbours will respond either with the data itself or suggesting a neighbour which might have it. Using this design, a new node can source any data in the network through a series of hops similar to how the Domain Name Service (DNS) system works [323].

The decentralised nature of the design makes the system highly resistant to denial of service attacks and independent of organisations shutting down. If one member leaves the network, the files stored on that node will be repaired through other nodes on the network and preserved. Previous research has found that as the amount of time a node stays in the network increases over the short term, the likelihood of them staying in the network long-term increases [324]. Using this property, the network automatically reduces the amount of redundant data, while ensuring the system remains stable [324].

DHTs are also simple to use and devices are able to communicate without any specialised setup on the user's router [325]. This is also important as one of the primary reasons for the existing IoT architecture is ease of use. DHT based technology has proved this as the mainline DHT supports between 15 and 27 million users per day [315], many of which are unaware of the underlying technology [326].

7.2 Proposed Decentralised IoT Architecture

Three issues which currently affect IoT devices in a domestic setting are (1) lack of gateway access control, (2) device longevity, and (3) securely accessing devices. This section will investigate each of these issues and detail a potential solution.

7.2.1 Architecture Overview

A DHT can be used a publicly available key value store, however, in most implementations the data is stored pseudorandomly, through using the hash of the input data as an address [324]. This makes it difficult to employ interactive services as an overlay service as the device would have to continually search the network for new data. Instead, through using asymmetric cryptographic keys, fixed addresses can be determined to and used as *re-writable* data store, making interactive communication possible. The use of cryptographic keys also means that only the owner of the key would be able to publish data to that particular address and any recipient would be able to verify the data held at the address using the public key.

A very constrained low-cost low-performance IoT devices in particular struggle with generating suitably random keys, meaning they are not able to implement state of the art cyber security principles alone [15,A3]. By employing a third party device, possibly the home router, to access the DHT on IoT device's behalf, devices which otherwise could not implement strong security features are able to have their data authenticated and encrypted. Remote control of devices within the home is available through checking specific addresses on the DHT, using outbound connections avoiding NAT issues. The gateway also removes devices unfettered access to the internet as only connections to the DHT are how required. It also improves the security of all IoT devices on the network as there is no direct network exposure, and no need to rely on a vendor to secure an internet-facing device.

7.2.2 Gateway Access Control

Based on the current security and privacy considerations of connected consumer electronics devices, as discussed in Chapter 3, the author decided to build upon the idea of Virtual LAN (VLAN) based isolation, where each device is entirely isolated from every other connected device, preventing a compromised IoT device from being able to monitor or sniff other devices network traffic [3], which is currently not guaranteed [314]. Although this may be achieved through custom configuration on many domestic routers, this feature is typically not enabled by default [327]. Prior research identified the importance of having secure default settings, yet defaults are typically designed for convenience rather than security [328, 329]. A secure IoT design should, therefore, exclude the need for any manual interaction or specific configuration on different devices [330], as this would require user maintenance of the rule-sets to permit or deny access as necessary, as discussed in [189].

Instead delegating access control and authentication to a separate layer, independent of IoT equipment vendors, would allow a single place to modify rule-sets as seen in Android and iOS [331]. One criticism such permission based systems have used is their complexity [332]. Therefore, any rule-sets would have to be easy to understand. The benefits of such a system would reassure the user of the capabilities and limitations of their devices, as such a system would ensure that no back-door access is available, and preserve security against remote access. Moreover, as the solution is abstracted from the IoT device itself, many devices can be simultaneously protected, even when the vendor no longer maintains their equipment. Overall, the problems discussed present a challenge for allowing secure communication between a remote user and IoT device on their home network. Existing solutions for NAT traversal and routing to dynamic IP address typically include the use of outbound connections to a central server [232–234].

This can be resolved through using a third party device as an intermediary, possibly as part of the functionality of the router. Moreover, delegating access control to a dedicated layer facilitates updating authentication and access control logic, without requiring updates to be made to every connected device. This is a major issue for current IoT devices, for as noted they are unlikely to receive security updates [141]. By building a single authentication layer, which is secured and hardened against remote attack, many devices on the network could be protected, even when the device itself is not adequately secured.

In addition to network segregation through a separate VLAN, the security of connected devices could also be improved through disabling devices directly accessing the wider internet. Rather than relying on a device itself remaining updated against the latest threats, introducing a gateway to the VLAN would allow for this validation to be carried out prior to any packets reaching the IoT device itself. This can be achieved through configuring the router's firewall to block all incoming and outgoing traffic between the Wide Area Network (WAN) interface and the IoT VLAN. The only connections permitted from IoT devices are to the gateway application, which both acts as a local Dynamic Host Configuration Protocol (DHCP) server, and passes validated messages along to their intended recipient. These steps enable isolation of devices from one another, and from the wider internet.

7.2.3 Device Longevity

Local connectivity issues can be solved through enabling devices to operate locally without having to rely on external services. Modifying the traditional IoT architecture to position an API on the device itself would enable users and companion applications to issue commands to the device itself through using mDNS broadcasts, as demonstrated in Chapter 4. This would remove any dependency on centralised architecture and protect users against their devices becoming non-functional if the manufacturer ceased trading or elected to no longer support their product. However, this would only operate within the local network as NAT and dynamic IP addressing would restrict external access to the device's API.

7.2.4 Dynamic IP Addressing

One of the major challenges in providing such a distributed self-hosted architecture is mitigating the issues posed by standard WAN domestic internet connections. These connections typically have dynamic IP addresses, which may vary upon re-connection, since the router is issued an IP address through DHCP from a pool of IPs held by the service provider. Consequently, there may be a challenge in securely locating the user's home IP address, without requiring a centralised approach or a static IP address, which is increasingly difficult to obtain on a home connection. In the absence of a static IP, techniques such as dynamic DNS are frequently employed to resolve this issue. This system uses a static domain name, such as, *www.iotdevice.com*, and a device inside the home consistently notifies the external service of the current IP address of the home to update the address *www.iotdevice.com* points to [333, 334]. However, dynamic DNS services are typically centralised, and a user's router must notify the DNS server through an agreed API to update the A-record. Such a service could be offered by the IoT device vendor, but this would involve similar challenges as outlined above. Alternatively, a third party company could offer a dynamic DNS service as a shared resource for all IoT devices within a user's home, yet this would rely on the compatibility of IoT vendors with such a service. Finally, there is the problem of the business model. If such a service were to incur a separate monthly service fee, this may deter consumers.

These concerns are also mirrored in the local network as domestic routers also commonly issue dynamic internal IP addresses through DHCP, meaning that a IoT address may change over time, causing updates to be lost or sent to the wrong device. Therefore, to facilitate the use of multiple IoT devices within the home, mDNS discovery can be implemented on the gateway and IoT devices should advertise their presence, a description of their functionality and a unique ID using mDNS broadcasts. A third party device can then aggregate all received mDNS advertisements and use them to form a service table, and keep track of each devices current IP address (Fig. 7.4).

7.2.5 Security Design

Connecting IoT devices to the DHT creates a situation similar to a dead letter drop, where a sender leaves messages in a particular place, and then the recipient collects the message. The complexity of data storage is abstracted away from the IoT device and regulated by the network itself. This creates effectively a database from the

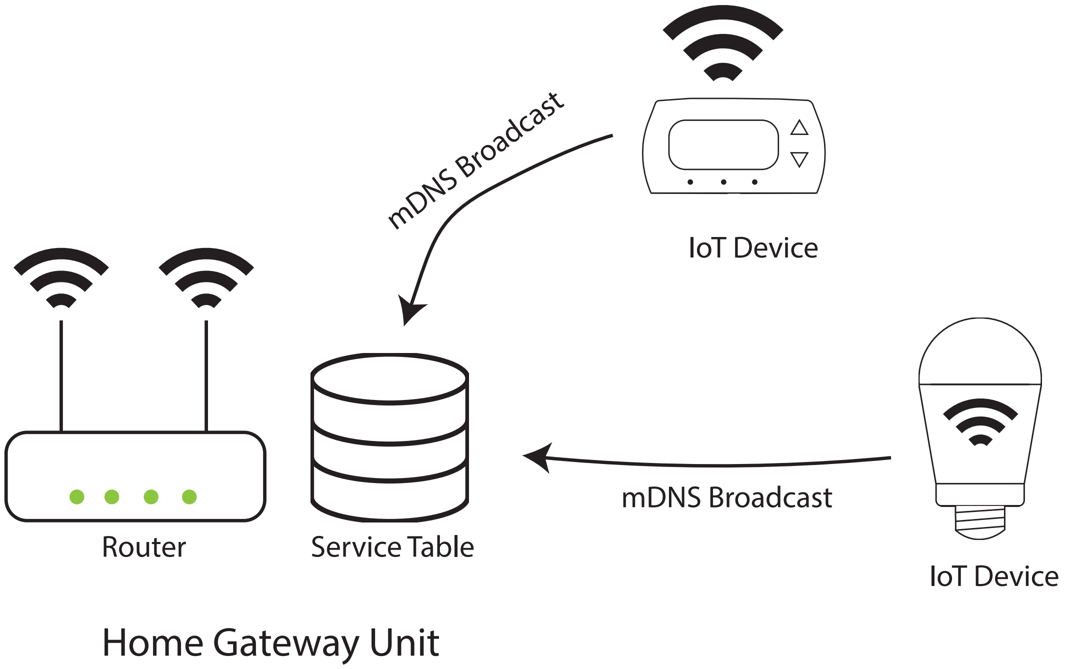


Figure 7.4: HGU's Service Table Collects Services from IoT Devices Through mDNS Broadcasts

perspective of the IoT device, where it can publish data to be received by a client application Fig. 7.5. As with the traditional architecture, connections to this *database* are outbound connections therefore authorised by default. The difference between the traditional and DHT based architectures is the inbound connections, traditional use outbound connections to authorise an inbound connection. Whereas, the DHT architecture the receiving device only makes outbound connection to the DHT for data and once any is discovered it is downloaded. This design solves the issue with dynamic IP addressing as all connections are outbound to the DHT rather than in-bound to the device. Using this method of posting to the DHT both the IoT device and any client application can fully communicate through posting and receiving from specific addresses.

These connections to and from the DHT must be secured. This is achieved through using a variety of ED25519 keys. As low-cost devices struggle to create these keys the process is offloaded to a third party device — the Home Gateway Unit (HGU). The HGU owns the first keypair, where it uses the hash of the public key as the address on

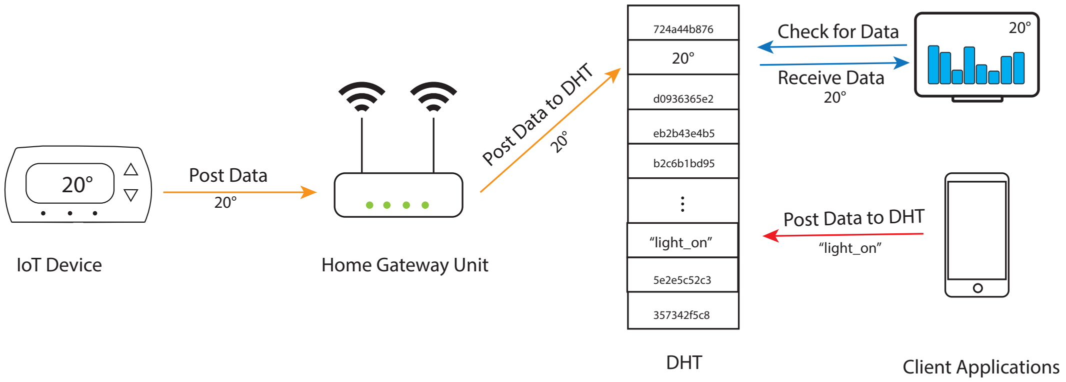


Figure 7.5: IoT Device Posting Data to an Address on the DHT for Consumers to Receive.

the DHT to store the capabilities of devices within the home. The matching private key is used to sign the messages published to the DHT. Once a IoT device is added to the home network, the HGU will generate a ED25519 key pair for the new device and the hash of the generated public key will become that devices address on the DHT. Once the device wishes to publish a message it will be signed with that devices private key. If the device relies on an external component such as a web application or smart phone application, this application will generate its own ED25519 keypair, and the public key is inserted into the HGU where the HGU will check that address on the DHT for messages from the application, and if any is found passing it along to the relevant IoT device.

Having the API on the device itself would also allow the device longevity to increase, which can be addressed through the DHT. However, constrained low-cost devices. As the HGU acts as an effective gateway for IoT devices, it can also provide access control restrictions to all devices on its network. This would prevent unauthorised devices communicating with one another as described in Section 7.2.2. There using the DHT in combination with an access device the HGU addresses all of the issues outlined.

This creates an eight step process for once a new IoT device is added to the network.

1. A keypair of public and private ED25519 keys are generated for the IoT device
2. The IoT device is added to the Wi-Fi network, and is isolated on a separate VLAN. The device sends out an mDNS broadcast detailing its capabilities. The gateway receives this broadcast message and adds it to its service table, and adds the address associated with the device to the list of addresses the HGU checks.
3. The gateway publishes the service table to the hash of the HGU's service table public key.
4. A mobile client device connects to the DHT, either directly or via a HTTP-based intermediary for performance and power consumption reasons [335], and receives the service table.
5. Once the user wishes to issue a request to an IoT device, the client creates a HTTP request, encrypts and signs the request.
6. The client then encapsulates the signed message, and publishes it to the DHT address derived from the cryptographic hash of the public key of the IoT device.
7. The HGU periodically looks for messages, at all of the registered hashes of public key addresses. The gateway also validates each request and it may apply access control restrictions to control the devices or conditions under which a given client is permitted to issue requests. Upon receiving and decrypting a request, the gateway can determine whether the sending user is permitted to carry out a given action on the device based upon rules logic set by the gateway owner (i.e. the homeowner).
8. Finally, the HTTP request is unencapsulated and transmitted to the IoT device for the device to action.

Once devices are connected to the network they will follow the following sequence Fig. 7.6.

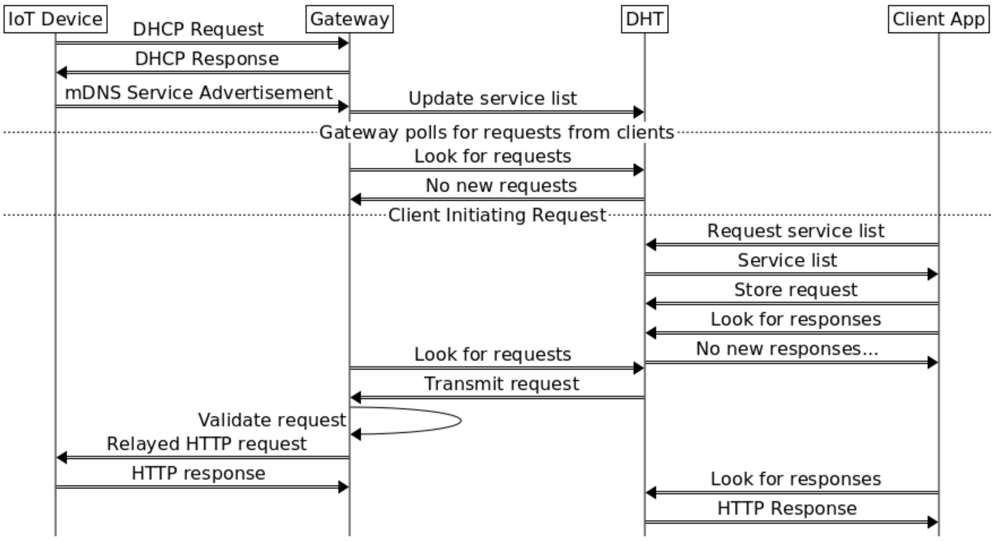


Figure 7.6: Protocol Design for Proposed Decentralised IoT Communications.

7.3 Evalution of Proposed Decentralised Architecture

An implementation of the HGU was created in NodeJS and configured to run on a laptop. Two client devices were also created to represent the different types of IoT devices, a high-performance device in the format of a Raspberry Pi, and a low-performance device using an MSP430G2553. An LED was connected to both devices to emulate an action which an IoT device would be able to actuate. The Raspberry Pi was wired directly to the ethernet network, while the MSP430 was connected to the Wi-Fi network using an ESP8266 Wi-Fi module. On setup the HGU created its own keypair for the service table, and key pairs for each of the connected IoT devices. The public key of each device was hashed, and the resulting addresses were added to the list of address the HGU would check for messages.

The mainline BitTorrent DHT was employed as it permits the storage of arbitrary mutable or immutable [1] data through the use of the BEP44 expansion protocol [336]. This part of the protocol allows messages up to 1000 bytes to be stored for two hours. This means that the DHT can be used as a temporary location to store data to be collected by the intended recipient.

[1] An object which cannot be changed

The built in security properties of the BitTorrent DHT also meant that mutable values must contain a valid signature corresponding to the private key controlling the address. This ensures that only the *owner* of the address can publish requests. Data downgrade or replay attacks are mitigated through the use of a monotonic counter field within each value, ensuring that well-behaved nodes will only accept updates to data whose version field is numerically greater than previously. This also enabled the HGU software to check it has the latest version of the data by checking the counter field.

Both devices were configured with a webserver and an REST API which would control the LED. After joining the network these devices would broadcast a mDNS message detailing the capabilities of the devices. These capabilities were then received by the HGU and added to the service table. The updated service table was then signed using the HGU's private key, and published to the hash of the HGU's public key. This allowed any new client-applications to retrieve the service table from the HGU, and get the capabilities of devices connected to the HGU. In this implementation, the capabilities were listed was the LED which could be toggled on each device.

Figure 7.7: 120 Bytes HTTP Request

Completing the implementation allowed the network overhead to be examined and compared to a traditional client server architecture. The IoT devices were configured to use HTTP and JSON for their APIs. This would create comparatively large messages given HTTP's large overhead for small messages compared to lightweight protocols such as MQTT, or Constrained Application Protocol (COAP) but HTTP is a common protocol for IoT devices due to its ease of use, and existing ecosystem. Similarly, JSON messages require opening and closing brackets, and a key value structure which is an easy to understand format but does increase the overhead. Therefore, the base level message increases. An example HTTP request to turn on the LED is shown in Fig. 7.7.

In a traditional architecture HTTP messages also require various headers at lower levels of the Open Systems Interconnection (OSI) networking stack, 32 bytes for Transmission Control Protocol (TCP) header, 20 bytes IP header, and 14 bytes of MAC header. Therefore, for a simple message of turn on the LED, which could be

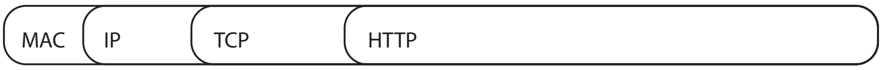


Figure 7.8: HTTP Packet with Surrounding Overhead

0 3 10 15 16 31

<table>
<tr><td colspan="3">Source Port</td><td>Destination Port</td></tr>
<tr><td colspan="4">Sequence Number</td></tr>
<tr><td colspan="4">Acknowledgement Number</td></tr>
<tr><td>Length</td><td>Reserved</td><td>Flags</td><td>Window Size</td></tr>
<tr><td colspan="3">TCP Checksum</td><td>Urgent Pointer</td></tr>
<tr><td colspan="4">Options</td></tr>
</table>

Figure 7.9: TCP Packet Header

one bit of information requires 186 bytes for an individual message. Moreover, as HTTP runs on top of TCP, which is a connection oriented protocol, it is required to setup the connection and tear down the connection which resulting in an additional an ≈128 bytes, and ≈160 bytes respectively. For a total TCP overhead of 474 bytes to send a message of turn on or off. This demonstrated that for a short message the overhead would be proportionally large Figs. 7.8 and 7.9.

This overhead would be further increased through implementing TLS to secure the connection [337]. TLS requires public keys and certificates to be transferred before any secure communication can be undertaken this is known as the TLS handshake which 4-7 kB. TLS adds a further ≈ 50 bytes per message for TLS headers. As the majority of this overhead is in setup it can be reduced by limiting the number of times the connection has to be setup, through keep-alive messages, as multiple messages can go through a single connection. However, this would also induce strain on the manufactures' server to keep connections open, and it would depend widely based on the configuration, and the use case of the IoT device. Therefore, the overhead associated with securing such a connection is variable.

When using the DHT the HTTP packet is encapsulated, therefore the HTTP message itself does not require any TCP headers. Instead, the DHT routes the

0 15	16 31
Source Port	Destination Port
Length	Header and Data Checksum

Figure 7.10: UDP Packet Header

message using UDP, a connectionless protocol, so there is no need to setup or tear down a connection reducing the overhead. Moreover, the UDP packet header is smaller Fig. 7.10.

Therefore, the light on HTTP message would require the 120 bytes which is then encrypted and signed. The encryption scheme used ed25519/Naci public box therefore induce an overhead of 16 bytes, resulting in 136 bytes. This message is then encapsulated into a UDP packet and published to the DHT requiring roughly 400 bytes for an individual post to the DHT, consisting of a 14 byte MAC layer header, a 20 byte IP header, and a 8 byte UDP header and the 136 bytes for the HTTP encapsulated DHT request. Prior to any messages being published to the DHT, the HGU would have to join the network, which would involve finding other peer nodes, so it could direct request to the correct peer, therefore the data used would be variable. As future messages would be stored in a constant location unless nodes leave the network it would not be required to find them again, therefore limiting the overhead to only posting and receiving. Requesting an acknowledgement from the DHT required 89 bytes, while receiving required 98 bytes. The initial request to find peer nodes required 134 bytes, and any ping messages would be roughly 100 bytes, and posting a HTTP request to the DHT would be variable depending on the packet transmitted but with an estimated ≈200 bytes overhead. The network performance at each stage of the system was measured using Wireshark [338].

The HGU was set to periodically poll the DHT for any new incoming requests. Each check of the DHT would require an 98 bytes outgoing message, and a 89 bytes response, therefore increasing the polling frequency also increases the data usage of HGU. Likewise, decreasing the polling frequency reduces data usage on the home internet connection at the expense of increased latency in responding to requests.

This shows that for short messages the DHT design is more efficient than the traditional architecture, with consideration that both connections have to continuously transmit either through DHT polling, or TCP keep-alive messages.

A visual representation of this is shown in Fig. 7.11, where setting up a TLS/TCP connection has a large overhead compared to our proposed DHT solution. Moreover, TLS/TCP keep alive messages require 108 bytes and are commonly set to trigger every 60 seconds, whereas the proposed solution required 187 bytes but can be varied by the user, and a check every 5 minutes would be sufficient for most out of house use cases. This means that over the course of a day, where the user was estimated to interact with their device 10 times would result in 543,024 bytes, compared to the 8,994,340 bytes of the existing solution.

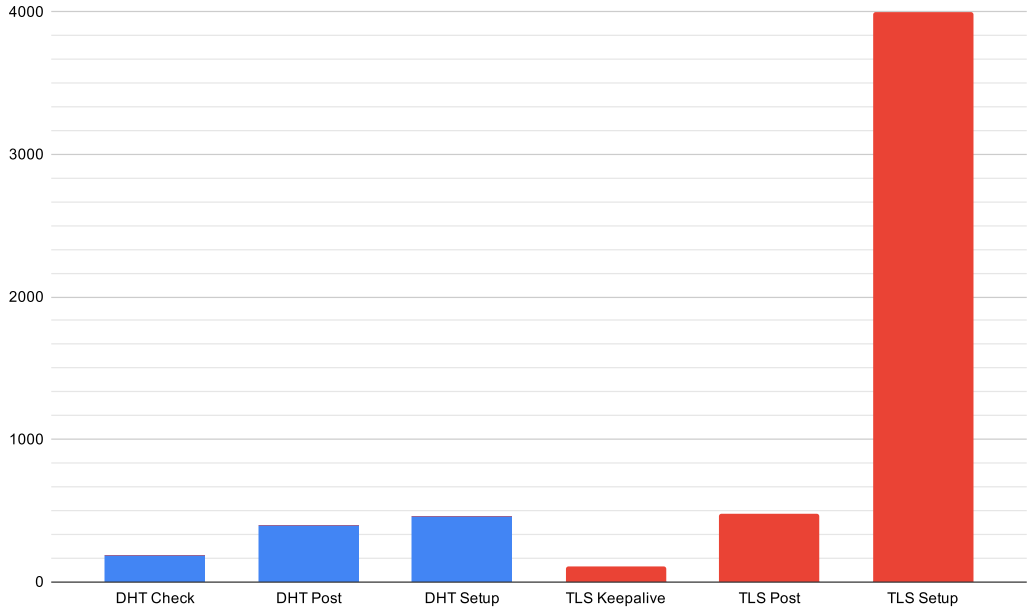


Figure 7.11: Bytes Required for Different Communication Methods

7.3.1 Device Isolation

The IoT devices are isolated from the internet using the firewall and VLAN capabilities within the HGU, making it impossible for a device on the IoT subnet to send or receive data to IP addresses other than the designated HGU address. Devices are isolated within the VLAN, preventing IoT devices from probing or communicating with each other. This forced isolation provided by the proposed architecture removes the necessity for each IoT device to carry out its own user authentication, a common attack vector on existing devices, as the HGU will complete this task. Nonetheless, this does not prevent or preclude a vendor from carrying out their own additional checks within their ecosystem, as all HTTP compliant messages would be encapsulated and passed to the device. Finally, devices without security updates can be continued to be used as they would be protected through the HGU, fulfilling a key security goal of this solution.

7.3.2 Gateway Exposure to Networks

The proposed HGU would be exposed to the internet via the WAN interface. To limit potential exploits, external connections would only be permitted to the DHT. In a production implementation each portion of the HGU software would be isolated and sandboxed from each other. As devices on the IoT network are isolated from one another, having access to the home Wi-Fi network does not automatically permit controlling and querying connected IoT devices on the network, instead allowed communication has to go via the HGU. The gateway's firewall prevents direct communication between the regular and IoT Wi-Fi networks, enforcing the use of the gateway to request authentication. This ensures that access control policies on the gateway are followed and avoids exposing insecure services running on IoT devices to the regular Wi-Fi network. IoT devices on the IoT network cannot communicate with the regular internet, and the gateway does not respond to unsolicited messages from external devices.

7.3.3 Client Software

Client applications and backend systems communicate with their IoT devices through the HGU. The HGU checks specific addresses registered by the client applications for messages for devices within the home, and once a message is discovered, it is retrieved and passed to the relevant device. This enables web services, such as IFTTT, to communicate indirectly to various IoT devices. The client application would also be required to check specific DHT addresses for messages from the IoT device.

The client may then transmit requests to the DHT, using the address of their public key which is associated to the HGU. The HGU polls this address, if the key has indeed been enrolled as an authorised key, the HGU will retrieve the message and pass it along to the IoT device.

Client software, such as a smartphone app, may also initiate direct communication with the HGU over the local Wi-Fi network. This removes the need for a DHT connection, and removes any latency associated with it, since all communication take place locally. The authentication is also carried out on the local network, as discussed in Section 7.3.2. The client software need not ever communicate directly with any devices in the home, nor need it accept any responses not received from the gateway.

7.3.4 Security Properties

One of the benefits of the proposed architecture is that it prevents any malicious user from attempting to connect to a domestic Wi-Fi network and compromise existing IoT devices. It also prevents manufacturers from creating devices which can spy on or attempt to compromise other devices within the network through enforcing device isolation. As all data is passed through the DHT, there is no direct connection for a user or external attacker to attempt to compromise the manufacturers server. Additionally, as devices are only authorised to make external connections to the DHT devices cannot join botnets such as Mirai to participate in DDoS attacks. The design of the DHT means that only the owner of a public key can publish to that specific address, so false messages cannot be introduced. Each valid message is also signed, authenticated and encrypted using public key cryptography preventing message tampering in transit.

The threat actors in the system are the user, the domestic network owner, the IoT vendor's web administrator and an external party who could impersonate either a user or the web administrator (Fig. 7.12). The domestic network owner would be able to directly interact with their device using its API, and would be able to see any messages in plaintext if they choice, but by default other devices and a regular network user would not be able to see any of the traffic of any other device on the network. The web administrator could be compromised however as there is no direct connection between deployed IoT devices and the vendor's server it would be challenging to influence the IoT device. The IoT vendor is also more secure under this design as any messages they retrieve from the DHT, are verified from coming with someone with knowledge of the private key of that particular IoT device. Whereas, in traditional architecture a IoT vendor would have to decipher the HTTP packet to gain access to a session cookie or JSON Web Token (JWT) or similar to establish the identity of the user, and this received packet could be malicious. It is worth noting that the proposed architecture would not affect the physical security of any of the existing IoT devices. This means that attacks such as spoofing, tempering or information exfiltration would still be possible through physical access to the device.

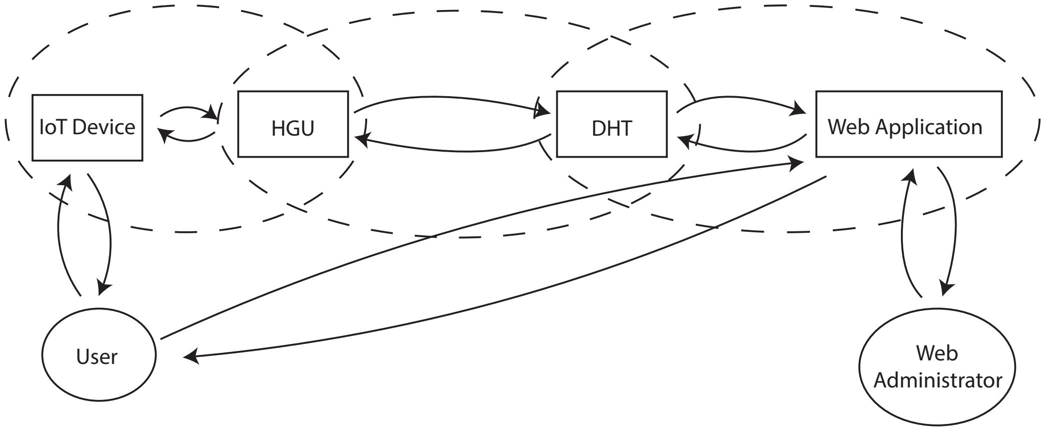


Figure 7.12: Threat Model Diagram of DHT Based System with Threat Boundaries Inserted

7.3.5 Limitations

In order for IoT devices to be compatible with the proposed architecture the traditional design of devices would have to be altered. Instead of the device reaching out to a remote server for commands, it would have to support a local API, and issue its capabilities via a mDNS broadcast to the HGU. Devices such as Phillips Hue already provides a local API on the device, but does not issue these API functions over mDNS. This would require a modification to the manufacturer's design for existing product to support the proposed decentralised architecture. A second limitation with the proposed design is that it would only be able to isolate IP devices from other IP devices, if Zigbee devices was employed in the home network, the HGU would not be able to limit these devices interacting.

Two considerations for employing a DHT based system with mobile devices is its data usage, and its energy usage. Mobile data packages regularly offer increased data quotas, but mobile data connections are likely to remain lower capacity than a fixed-line network. Periodically checking the DHT for messages would result in increased data usage and decrease the mobile device's battery life. Both Apple and Google have optimised this process of checking for messages on a central server as this is a common requirement through their Google Cloud Messaging, and Apple's Push Notification Services. The limitation of this design is that both services require HTTP messages. Therefore, it would be advantageous for a pool of nominated intermediaries to check the DHT for messages, at specific addresses and push any messages over HTTP to the mobile device Fig. 7.13. It is important to note that these intermediaries could be independently hosted and would not be able to publish messages without authorisation, and would not be able to read any messages.

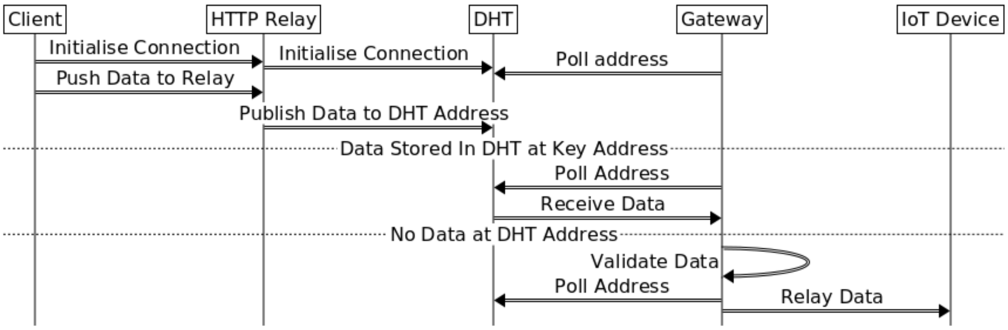


Figure 7.13: Protocol Design for Secure Remote Access

High performance devices such as the Raspberry Pi used in the test, and similar devices are capable of generating suitable random numbers are able to generate their own keypairs, therefore can implement their own security features to provide end-to-end encryption. The most common type of secure communication is TLS which as it is HTTP based it would be possible to implement over the DHT connection, however as

TLS by default has a timeout value, where the connection which is normally expected to be setup in milliseconds may take tens of seconds depending on how often the HGU is configured to poll the DHT. The HGU could either be signalled to temporarily increase the polling frequency or Datagram Transport Layer Security (DTLS) [339] which is connectionless could be implemented instead, as it has been proposed for asynchronous constrained devices [340].

The proposed system isolates each individual device from every other device, this may suggest it is no longer possible for devices within the home to interact with one another, however, currently most interaction between local devices is actually carried out between backend servers such as IFTTT. A request from one device to another would leave the first IoT device, and published to the DHT, that IoT provider, would then pass the message to the second providers backend server. Upon receiving a request to command their device, the second provider would publish this message to the DHT at its own registered address, where the HGU which notice the message and pass it to the second local IoT device. The polling frequency set by the HGU would determine how long this interaction would take place.

7.4 Conclusion

As IoT devices are commonly dependant on the desires of a central provider, software updates and cloud infrastructure have led to longevity concerns. Instead, this chapter positions the IoT device's API on the device itself, and proposes an alternative decentralised architecture. Using an intermediary HGU, IoT devices are protected from external internet connections, while enabling a persistent connection for use of the device. The separation provided by the architecture prevents IoT devices from joining botnets and protects both the IoT device user and the IoT vendor as there is no direct connection which could be exploited. An analysis of the proposal found that the network overhead was less than the existing solution, the architecture operated well with a computationally-constrained devices, and could be retrofitted to existing devices. Employing the proposed design would make IoT devices independent of centralised servers, leading to a better resiliency for long-term device use.

Chapter 8

Conclusions

The aim of this PhD study was to improve security in low-cost devices —devices which by themselves were not able to implement state of the art security features. This formed the initial research question "Can the requirement for additional components for securing IoT devices be reduced?". This focused the research into establishing the current restrictions in place for implementing state of the art security features. One of the primary restriction in domestic IoT devices is the cost of providing security. Therefore, the first contribution of this PhD study in Chapter 4 and is the creation of an architecture that allows for state of the art security with no additional hardware cost. This contribution was also influenced by the second research question "Can the deployment of secure IoT devices be simplified?". Employing an existing piece of technology, a smartphone, which is commonly available and user-friendly, enables the automated installation of strong cryptographic keys into domestic Wi-Fi based devices. This contribution is published in [A3].

The second contribution is a creation of an architecture for installing strong cryptographic key in IIoT devices using commonly available hardware. Chapter 6 describes using accessible user hardware, in the form of a smartphone, automated the process of provisioning IIoT devices, answering the second research question of "Can the deployment of secure IoT devices be simplified?". This design also meant that devices could be deployed using a distributed approach without requiring specialised knowledge. Hence, a network could be (re)deployed quickly in the event of a compromise increasing network resilience, as detailed in the publication [A2].

The third contribution of this study described in Chapter 5 is the development of a monitoring device for critical national infrastructure. Three LPWAN technologies were evaluated and LoRaWAN was found to be the most suitable candidate for a monitoring device in such an environment which requires security. This answered the research question "Can critical national infrastructure of IoT devices be secured with only passive sensing, ideally with unidirectional communications?". By allowing assets to securely report details to allow the advancement of smart grids. This contribution was also published in [A1].

Chapter 7 contribution is a decentralised communication architecture for IoT devices. This is in line with the research question "Can the dependency of external components for IoT be reduced?". Removing the requirement for a centralised server the contributed architecture increases security through isolating devices and limiting attack surfaces. The contribution also reduces the dependency for external software updates to secure IoT devices, shielding them instead.

Overall this study has shown that security features can be added to low-cost IoT devices using limited additional hardware, in both critical national infrastructure and domestic environments through a simple automated processes. Finally, the dependency of external devices was reduced through decentralisation while increasing the overall security of each system, fully addressing all research questions and concluding that yes, security can be provided with limited additional hardware for a low-cost.

Chapter 9

Further Work

9.1 Local Area & Personal Devices

The proposed domestic key transfer model (detailed in Section 4.1.1) could be expanded to operate with Bluetooth devices, such as wearable devices. Currently, Bluetooth pairing methods rely on using a shared pin, however this pin is short and can be brute forced [341]. The nature of these devices means that they are portable as they travel with the individual. Therefore, employing a similar architecture, using the smartphone to transfer generate and transfer a key to the device while in secure environment would allow the communication between the user's smartphone and the Bluetooth device to operate securely in all environments. Likewise, the design could be modified to work via a local intermediary, for devices which do not share an communication interface with smartphones. For example, Philips Hue uses Wi-Fi to communicate with a hub, and then Zigbee to the IoT device itself (the bulb). The hub acting as a relay for the phone generated key would allow each Phillips Hue lightbulb to end-to-end encrypt its traffic.

9.2 Multiple Users

The proposed domestic key transfer architecture works well for a single user or multiple users sharing the same set of keys. However, if each user required dedicated keys, for example, to provide zonal control over the lights, they would have to repeat the setup process for each light, a time-intensive task. A potential solution to this would be to have each new user detect all the switches they can see on the network through mDNS, and generate a new key for each one. However, since these keys would then be transferred over the home Wi-Fi network, if this network was compromised, then each of these keys would also be compromised. A more sophisticated solution would involve creating a master user. The master user would create a master AES keys, which would be used to initially setup up all the devices. Once a new user is to be added to the system new sub-keys would be generated and transferred to each of the required switches encrypted with the master key. The master user would then separately send each sub-key to the user's phone using Bluetooth or NFC. However, storing further keys in these constrained devices would require more resources, and would be more complex for users, therefore it would be the vendor's discretion on how to proceed in their specific case.

9.3 Wide Area Industrial Device

NB-IoT and LTE-M are likely to become more prominent in the future, particularly with private 4G/5G networks. Currently 4G/5G devices are fitted with a SIM card which contains all the necessary cryptographic keys to secure communication. However, embedded SIM cards (eSIM) is likely to replace traditional SIM cards. These eSIMs are software programmable, and therefore can be altered on deployed devices which is currently not possible with existing SIM cards. This would allow eSIM based IoT devices to change mobile operator without having to manually recall devices and replace SIM cards. This move would also mean that SIM cards would not have be inserted into devices before deployment. However, this change would mean an alternative key installation procedure would have be used. Therefore. the key transfer method proposed in Section 6.1 could be applied to NB-IoT and LTE-M based devices.

9.4 Encryption Scheme

Both of these proposed architectures used AES as an encryption cipher, other ciphers were considered however recently proposals such as Salsa20, require a random nonce and therefore would not be suitable for low-cost devices. AES-256 is still considered safe both for traditional cracking methods and quantum computing [342], however it would be beneficial to prepare an alternative cipher, if future research discovers that AES is no longer secure. Therefore, once yet to be finalised quantum ciphers are available their performance and practicality could be investigated. It should be noted that while the investigation employed AES and ED25519 the contributions are primarily of the format of architectures and are therefore implementation agnostic.

9.5 Decentralised Communication

The DHT based architecture outlined in Section 7.2.1 could be expanded to include high-performance devices which already support security features, such as TLS. To accommodate this given the low-latency requirements of TLS the architecture would have to be modified to temporarily enable rapid polling. Alternatively, DTLS could be employed given its connectionless protocol. This would be similar to existing DTLS deployments in LPWAN systems.

Bibliography

[1] H. Liu, T. Spink, and P. Patras, "Uncovering Security Vulnerabilities in the Belkin WeMo Home Automation Ecosystem," in *2019 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*. IEEE, 2019, pp. 894–899.

[2] M. Ye, N. Jiang, H. Yang, and Q. Yan, "Security analysis of internet-of-things: A case study of august smart lock," in *2017 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*. IEEE, 2017, pp. 499–504.

[3] J. Wurm, K. Hoang, O. Arias, A.-R. Sadeghi, and Y. Jin, "Security analysis on consumer and industrial IoT devices," in *2016 21st Asia and South Pacific Design Automation Conference (ASP-DAC)*. IEEE, 2016, pp. 519–524.

[4] H. Lin and N. W. Bergmann, "IoT privacy and security challenges for smart home environments," *Information*, vol. 7, no. 3, p. 44, 2016.

[5] L. David. (2018, May) Hijacking Philips Hue. [Online]. Available: https://www.pentestpartners.com/security-blog/hijacking-philips-hue/

[6] K. Zetter. (2012, Jul.) Flaw In Home Security Cameras Exposes Live Feeds To Hackers. [Online]. Available: https://www.wired.com/2012/02/home-cameras-exposed/

[7] A. Greenberg, "Hack Brief: 'Devil's Ivy' Vulnerability Could Afflict Millions Of IoT Devices," Wired, Tech. Rep., 2018. [Online]. Available: https://www.wired.com/story/devils-ivy-iot-vulnerability/

[8] L. Franceschi-Bicchierai, "Hacker Obtained Children's Headshots and Chatlogs from Toymaker VTech', Motherboard," 2015. [Online]. Available: https://www.vice.com/en_us/article/yp3zev/hacker-obtained-childrens-headshots-and-chatlogs-from-toymaker-vtech

[9] J. Cox and S. Cole, "How hackers are breaking into ring cameras," 2019. [Online]. Available: https://www.vice.com/en_us/article/3a88k5/how-hackers-are-breaking-into-ring-cameras

[10] D. Goodin, "Creepy IoT teddy bear leaks ¿2 million parents' and kids' voice messages," 2017. [Online]. Available: https://arstechnica.com/information-technology/2017/02/creepy-iot-teddy-bear-leaks-2-million-parents-and-kids-voice-messages/

[11] (2016, Jul.) Pets left hungry as smart feeder breaks. BBC News. [Online]. Available: http://www.bbc.co.uk/news/technology-36912992

[12] J. Blythe and S. Johnson, "The Consumer Security Index for IoT: A protocol for developing an index to improve consumer decision making and to incentivize greater security provision in IoT devices," *IET Conference Proceedings*, pp. 4 (7 pp.)–4 (7 pp.)(1), January 2018.

[13] E. Leverett, R. Clayton, and R. Anderson, "Standardisation and Certification of the 'Internet of Things'," *Proceedings of WEIS 2017*, 2017.

[14] "Consumer Internet of Things Security Labelling Survey Research Findings," Department for Digital, Culture, Media and Sport (DCMS), Tech. Rep., 2019. [Online]. Available: https://assets.publishing.service.gov.uk/government/uploads/system/uploads/attachment_data/file/798543/Harris_Interactive_Consumer_IoT_Security_Labelling_Survey_Report.pdf

[15] O. Garcia-Morchon, S. Kumar, and M. Sethi, "Internet of Things (IoT) Security: State of the Art and Challenges," Request for comments, Tech. Rep., 2019, accessed 11th November 2019. [Online]. Available: https://rfc-editor.org/rfc/rfc8576.txt

[16] L. Harriss and C. West, “Cyber security of consumer devices,” Parliamentary Office of Science and Technology, London, UK, Tech. Rep., Feb. 2019. [Online]. Available: https://researchbriefings.parliament.uk/ResearchBriefing/Summary/POST-PN-0593

[17] L. Jun-tao and L. Hong-jian, “Design optimization of Amazon robotics,” *Automation, Control and Intelligent Systems*, vol. 4, no. 2, p. 48, 2016.

[18] A. Razaghpanah, A. A. Niaki, N. Vallina-Rodriguez, S. Sundaresan, J. Amann, and P. Gill, “Studying tls usage in android apps,” in *Proceedings of the 13th International Conference on emerging Networking EXperiments and Technologies*, 2017, pp. 350–362.

[19] M. O. Ojo, S. Giordano, G. Procissi, and I. N. Seitanidis, “A review of low-end, middle-end, and high-end iot devices,” *IEEE Access*, vol. 6, pp. 70 528–70 554, 2018.

[20] D. J. Bernstein, “Curve25519: new Diffie-Hellman speed records,” in *International Workshop on Public Key Cryptography.* Springer, 2006, pp. 207–228.

[21] ——, “Chacha, a variant of salsa20,” in *Workshop Record of SASC*, vol. 8, 2008, pp. 3–5.

[22] B. Schneier, J. Kelsey, D. Whiting, D. Wagner, C. Hall, and N. Ferguson, “Twofish: A 128-bit block cipher,” *NIST AES Proposal*, vol. 15, no. 1, pp. 23–91, 1998.

[23] J. Daemen and V. Rijmen, “AES proposal: Rijndael,” Tech. Rep., October 1999. [Online]. Available: https://www.cryptosoft.de/docs/Rijndael.pdf

[24] A. Francillon and C. Castelluccia, “Tinyrng: A cryptographic random number generator for wireless sensors network nodes,” in *2007 5th International Symposium on Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks and Workshops.* IEEE, 2007, pp. 1–7.

[25] Z. Najm, D. Jap, B. Jungk, S. Picek, and S. Bhasin, “On comparing side-channel properties of AES and ChaCha20 on microcontrollers,” in *2018 IEEE Asia Pacific Conference on Circuits and Systems (APCCAS).* IEEE, 2018, pp. 552–555.

[26] "Electronic toy maker VTech settles FTC allegations that it violated children's privacy law and the FTC Act," Federal Trade Commission, Tech. Rep., January 2018. [Online]. Available: https://www.ftc.gov/news-events/press-releases/2018/01/electronic-toy-maker-vtech-settles-ftc-allegations-it-violated

[27] R. Nigam. CVE-2015-4400 : Backdoorbot Network Configuration Leak on a Connected Doorbell. fortinet. (22 January 2016). [Online]. Available: https://www.fortinet.com/blog/threat-research/cve-2015-4400-backdoorbot-network-configuration-leak-on-a-connected-doorbell.html

[28] J. Cox, "We tested ring's security. it's awful, motherboard," 2019. [Online]. Available: https://www.vice.com/en_uk/article/epg4xm/we-tested-rings-security-its-awful

[29] E. Ramirez, J. Brill, M. K. Ohlhausen, and J. D. Wright, "In the Matter of Treadnet, INC., a corporation."

[30] D. Shonka, L. D. Berger, and K. H. Moriarty, "Federal Trade Commission v. D-LINK Corporation."

[31] "2020 Unit 42 IoT Threat Report," Palo Alto Networks, Tech. Rep., 2020, 10th March 2020. [Online]. Available: https://unit42.paloaltonetworks.com/iot-threat-report-2020/

[32] M. Singh, J. Trivedi, P. Maan, and J. Goyal, "Smartphone Battery State-of-Charge (SoC) Estimation and battery lifetime prediction: State-of-art review," in *2020 10th International Conference on Cloud Computing, Data Science Engineering (Confluence)*, 2020, pp. 94–101.

[33] L. Ferrigno, A. Pietrosanto, and V. Paciello, "Low-cost visual sensor node for BlueTooth-based measurement networks," *IEEE Transactions on Instrumentation and Measurement*, vol. 55, no. 2, pp. 521–527, 2006.

[34] O. Arias, J. Wurm, K. Hoang, and Y. Jin, "Privacy and security in internet of things and wearable devices," *IEEE Transactions on Multi-Scale Computing Systems*, vol. 1, no. 2, pp. 99–109, 2015.

[35] B. Schneier, *Applied Cryptography.* Wiley, 1996.

[36] W. Trappe, *Introduction to cryptography with coding theory.* Pearson Education India, 2006.

[37] W. Mehuron, "Data encryption standard (des)," United States of America National Institute of Standards and Technology, Tech. Rep., 1999. [Online]. Available: https://csrc.nist.gov/csrc/media/publications/fips/46/3/archive/1999-10-25/documents/fips46-3.pdf

[38] R. Davis, "The data encryption standard in perspective," *IEEE Communications Society Magazine*, vol. 16, no. 6, pp. 5–9, November 1978.

[39] M. Curtin and J. Dolske, "A brute force search of DES keyspace," in *8th Usenix Symposium, January*, 1998, pp. 26–29.

[40] J. Kilian and P. Rogaway, "How to protect des against exhaustive key search," in *Annual International Cryptology Conference.* Springer, 1996, pp. 252–267.

[41] P. Karn, P. Metzger, and W. Simpson, "The esp triple des transform," Network Working Group, Tech. Rep., 1995. [Online]. Available: https://tools.ietf.org/html/rfc1851

[42] V. Rijmen and J. Daemen, "Advanced encryption standard," *Proceedings of Federal Information Processing Standards Publications, National Institute of Standards and Technology*, pp. 19–22, 2001.

[43] J. Nechvatal, E. Barker, L. Bassham, W. Burr, M. Dworkin, J. Foti, and E. Roback, "Report on the development of the Advanced Encryption Standard (AES)," *Journal of Research of the National Institute of Standards and Technology*, vol. 106, no. 3, p. 511, 2001.

[44] L. Chen, L. Chen, S. Jordan, Y.-K. Liu, D. Moody, R. Peralta, R. Perlner, and D. Smith-Tone, *Report on post-quantum cryptography.* US Department of Commerce, National Institute of Standards and Technology, 2016, vol. 12.

[45] A. Biryukov, O. Dunkelman, N. Keller, D. Khovratovich, and A. Shamir, "Key Recovery Attacks of Practical Complexity on AES-256 Variants with up to 10 Rounds," in *Advances in Cryptology – EUROCRYPT 2010*, H. Gilbert, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 299–319.

[46] A. Biryukov and D. Khovratovich, "Related-key cryptanalysis of the full AES-192 and AES-256," in *International Conference on the Theory and Application of Cryptology and Information Security.* Springer, 2009, pp. 1–18.

[47] A. Bogdanov, D. Khovratovich, and C. Rechberger, "Biclique cryptanalysis of the full AES," in *International Conference on the Theory and Application of Cryptology and Information Security.* Springer, 2011, pp. 344–371.

[48] J. Alawatugoda, D. Jayasinghe, and R. Ragel, "Countermeasures against bernstein's remote cache timing attack," *2011 6th International Conference on Industrial and Information Systems*, Aug 2011. [Online]. Available: http://dx.doi.org/10.1109/ICIINFS.2011.6038038

[49] A. Kahate, *Cryptography and Network Security.* Tata McGraw-Hill Publishing, 2008.

[50] A. J. Menezes, J. Katz, P. C. Van Oorschot, and S. A. Vanstone, *Handbook of applied cryptography.* CRC press, 1996.

[51] M. Dworkin, "Recommendation for block cipher modes of operation. methods and techniques," National Inst of Standards and Technology Gaithersburg MD Computer security Div, Tech. Rep., 2001. [Online]. Available: https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-38a.pdf

[52] H. Lipmaa, P. Rogaway, and D. Wagner, "Comments to NIST concerning AES modes of operations: CTR-mode encryption," in *National Institute of Standards and Technologies*, 2000.

[53] M. J. Dworkin, *SP 800-38D. Recommendation for block cipher modes of operation: Galois/Counter Mode (GCM) and GMAC.* National Institute of Standards & Technology, 2007. [Online]. Available: https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-38d.pdf

[54] D. McGrew and J. Viega, "The Galois/counter mode of operation (GCM)," *submission to NIST Modes of Operation Process*, vol. 20, 2004.

[55] M. J. Dworkin, *SP 800-38D. Recommendation for block cipher modes of operation: Galois/Counter Mode (GCM) and GMAC Appendx C.* National Institute of Standards & Technology, 2007.

[56] S. Gueron, A. Langley, and Y. Lindell, "AES-GCM-SIV: Specification and Analysis." *IACR Cryptology ePrint Archive*, vol. 2017, p. 168, 2017.

[57] J. C. Lee, V. C. Leung, K. H. Wong, J. Cao, and H. C. Chan, "Key management issues in wireless sensor networks: current proposals and future developments," *IEEE Wireless Communications*, vol. 14, no. 5, pp. 76–84, 2007.

[58] R. L. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," *Communications of the ACM*, vol. 21, no. 2, pp. 120–126, 1978.

[59] M. O. Rabin, "Digitalized signatures and public-key functions as intractable as factorization," Massachusetts Inst of Tech Cambridge Lab for Computer Science, Tech. Rep., 1979. [Online]. Available: https://apps.dtic.mil/dtic/tr/fulltext/u2/a078415.pdf

[60] P. McMahon, "Sesame v2 public key and authorization extensions to kerberos," *Proceedings of the Internet Society 1995 Symposium on Network and Distributed Systems Security*, 1995.

[61] R. L. Rivest, A. Shamir, and L. M. Adleman, "Cryptographic communications system and method," Sep. 20 1983, US Patent 4,405,829.

[62] M. E. Hellman and R. C. Merkle, "Public key cryptographic apparatus and method," Aug. 19 1980, US Patent 4,218,582.

[63] C. P. Schnorr, "Method for identifying subscribers and for generating and verifying electronic signatures in a data exchange system," Feb. 19 1991, US Patent 4,995,082.

[64] M. E. Hellman and S. C. Pohlig, "Exponentiation cryptographic apparatus and method," Jan. 3 1984, US Patent 4,424,414.

[65] M. E. Hellman, B. W. Diffie, and R. C. Merkle, "Cryptographic apparatus and method," Apr. 29 1980, US Patent 4,200,770.

[66] J. Callas, L. Donnerhacke, H. Finney, and R. Thayer, "OpenPGP message format," Network Working Group, Tech. Rep., November 1998. [Online]. Available: https://www.ietf.org/rfc/rfc2440.txt

[67] V. S. Miller, "Use of elliptic curves in cryptography," in *Advances in Cryptology — CRYPTO '85 Proceedings*, H. C. Williams, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 1986, pp. 417–426.

[68] N. Koblitz, "Elliptic curve cryptosystems," *Mathematics of computation*, vol. 48, no. 177, pp. 203–209, 1987.

[69] S. Blake-Wilson, N. Bolyard, V. Gupta, C. Hawk, and B. Moeller, "Elliptic curve cryptography (ECC) cipher suites for transport layer security (TLS)," RFC 4492, May, Tech. Rep., 2006. [Online]. Available: https://tools.ietf.org/html/rfc4492

[70] D. Stebila and J. Green, "Elliptic curve algorithm integration in the secure shell transport layer," Internet Requests for Comments, RFC Editor, Tech. Rep., Dec. 2009. [Online]. Available: http://www.rfc-editor.org/rfc/rfc5656.txt

[71] Z. Durumeric, F. Li, J. Kasten, J. Amann, J. Beekman, M. Payer, N. Weaver, D. Adrian, V. Paxson, M. Bailey *et al.*, "The matter of heartbleed," in *Proceedings of the 2014 conference on internet measurement conference*, 2014, pp. 475–488.

[72] Changes between 0.9.8n and 1.0.0. OpenSSL. Accessed: 1 April 2020. [Online]. Available: https://www.openssl.org/news/changelog.html

[73] OpenSSH Release Notes. OpenSSH. Accessed: 1 April 2020. [Online]. Available: https://www.openssh.com/releasenotes.html

[74] D. Shumow and N. Ferguson, "On the possibility of a back door in the NIST SP800-90 Dual Ec Prng," in *Proc. Crypto*, vol. 7, 2007.

[75] O. Raso, P. Mlynek, R. Fujdiak, L. Pospichal, and P. Kubicek, "Implementation of Elliptic Curve Diffie Hellman in ultra-low power microcontroller," pp. 662–666, 2015.

[76] B. Schneier, "A primer on authentication and digital signature," *Computer Security Journal*, vol. 10, pp. 63–63, 1994.

[77] D. J. Bernstein, N. Duif, T. Lange, P. Schwabe, and B.-Y. Yang, "High-speed high-security signatures," *Journal of cryptographic engineering*, vol. 2, no. 2, pp. 77–89, 2012.

[78] T. Dierks and C. Allen, "Rfc2246: The TLS protocol version 1.0," 1999.

[79] Qualy SSL Labs. Accessed on 17th March 2020. [Online]. Available: https://www.ssllabs.com/ssl-pulse/

[80] B. Möller, T. Duong, and K. Kotowicz, "This POODLE bites: exploiting the SSL 3.0 fallback," Google, Tech. Rep., 2014.

[81] P. G. Sarkar and S. Fitzgerald, "Attacks on SSL a comprehensive study of BEAST, CRIME, TIME, BREACH, Lucky 13 & RC4 biases," iSECpartners, Tech. Rep., 2013.

[82] T. Yoo, J.-S. Kang, and Y. Yeom, "Recoverable Random Numbers in an Internet of Things Operating System," *Entropy*, vol. 19, no. 3, 2017, cited By 3.

[83] K. Munro. IoT Encryption: The challenge of missing entropy. Pen Test Partners. (7 Dec 2016). [Online]. Available: https://www.pentestpartners.com/security-blog/iot-encryption-the-challenge-of-missing-entropy/

[84] A. Greenberg. (2018) The untold story of NotPetya, the most devastating cyberattack in history.

[85] ——, *Sandworm: A New Era of Cyberwar and the Hunt for the Kremlin's Most Dangerous Hackers.* Doubleday Publishing Group, 2019.

[86] J. Liebow-Feeser, "LavaRand in Production: The Nitty-Gritty Technical Details," Cloudflare, Tech. Rep. [Online]. Available: https://blog.cloudflare.com/lavarand-in-production-the-nitty-gritty-technical-details/

[87] Z. Gutterman, B. Pinkas, and T. Reinman, "Analysis of the linux random number generator," in *Proceedings - IEEE Symposium on Security and Privacy*, vol. 1, 2006, pp. 371–385.

[88] N. Sullivan. Ensuring Randomness with Linux's Random Number Generator. Cloudflare. Accessed 15 Jan 2019. [Online]. Available: https://blog.cloudflare.com/ensuring-randomness-with-linuxs-random-number-generator/

[89] K. Wallace, K. Moran, E. Novak, G. Zhou, and K. Sun, "Toward sensor-based random number generation for mobile and IoT devices," *IEEE Internet of Things Journal*, vol. 3, no. 6, pp. 1189–1201, 2016.

[90] B. B. Visser, "Additional source of entropy as a service in the Android user-space," Ph.D. dissertation, MA thesis. Radboud University, 2015.

[91] K. Mowery, M. Wei, D. Kohlbrenner, H. Shacham, and S. Swanson, "Welcome to the Entropics: Boot-Time Entropy in Embedded Devices," in *2013 IEEE Symposium on Security and Privacy*, May 2013, pp. 589–603.

[92] *SenSys '16: Proceedings of the 14th ACM Conference on Embedded Network Sensor Systems CD-ROM.* New York, NY, USA: Association for Computing Machinery, 2016.

[93] J. Voris, N. Saxena, and T. Halevi, “Accelerometers and Randomness: Perfect Together,” in *Proceedings of the Fourth ACM Conference on Wireless Network Security*, ser. WiSec ’11, ACM. New York, NY, USA: Acm, 2011, pp. 115–126. [Online]. Available: http://doi.acm.org/10.1145/1998412.1998433

[94] N. Sornin, “LoRaWAN™ 1.0.2 Specification,” LoRa Alliance, Tech. Rep., 2017. [Online]. Available: https://lora-alliance.org/sites/default/files/2018-05/lorawan1_0_2-20161012_1398_1.pdf

[95] S. Tomasin, S. Zulian, and L. Vangelista, “Security Join Procedure for Internet of Things Networks,” in *2017 IEEE Wireless Communications and Networking Conference Workshops (WCNCW)*, Mar. 2017, pp. 1–6.

[96] K. Wertheim, “Embryology and morphology of friction ridge skin,” *The fingerprint sourcebook*, pp. 103–126, 2011.

[97] F. Chabaud and A. Joux, “Differential collisions in SHA-0,” in *Annual International Cryptology Conference.* Springer, 1998, pp. 56–71.

[98] X. Wang, Y. L. Yin, and H. Yu, “Finding collisions in the full SHA-1,” in *Annual international cryptology conference.* Springer, 2005, pp. 17–36.

[99] M. Stevens, E. Bursztein, P. Karpman, A. Albertini, and Y. Markov, “The first collision for full SHA-1,” in *Annual International Cryptology Conference.* Springer, 2017, pp. 570–596.

[100] J. Song, R. Poovendran, J. Lee, and T. Iwa, “The AES-CMAC Algorithm,” Internet Requests for Comments, RFC Editor, RFC 4493, June 2006. [Online]. Available: https://tools.ietf.org/html/rfc4493

[101] A. Dorri, S. S. Kanhere, R. Jurdak, and P. Gauravaram, “Blockchain for IoT security and privacy: The case study of a smart home,” in *2017 IEEE international conference on pervasive computing and communications workshops (PerCom workshops).* IEEE, 2017, pp. 618–623.

[102] K. Christidis and M. Devetsikiotis, “Blockchains and Smart Contracts for the Internet of Things,” *IEEE Access*, vol. 4, pp. 2292–2303, 2016.

[103] Z. Zheng, S. Xie, H. Dai, X. Chen, and H. Wang, “An overview of blockchain technology: Architecture, consensus, and future trends,” in *Proceedings - 2017 IEEE 6th International Congress on Big Data, BigData Congress 2017*, Z. J. Karypis G., Ed. Institute of Electrical and Electronics Engineers Inc., 2017, pp. 557–564.

[104] M. Khan and K. Salah, “IoT security: Review, blockchain solutions, and open challenges,” *Future Generation Computer Systems*, vol. 82, pp. 395–411, 2018.

[105] M. Conoscenti, A. Vetro, and J. C. De Martin, “Blockchain for the Internet of Things: A systematic literature review,” in *2016 IEEE/ACS 13th International Conference of Computer Systems and Applications (AICCSA).* Ieee, 2016, pp. 1–6.

[106] R. Böhme, N. Christin, B. Edelman, and T. Moore, “Bitcoin: Economics, technology, and governance,” *Journal of Economic Perspectives*, vol. 29, no. 2, pp. 213–38, May 2015.

[107] Y. Xu and Y. Huang, “Segment blockchain: A size reduced storage mechanism for blockchain,” *IEEE Access*, vol. 8, pp. 17 434–17 441, 2020.

[108] G. Ramachandran and B. Krishnamachari, “Blockchain for the IoT: Opportunities and Challenges,” 05 2018.

[109] D. Puthal, S. P. Mohanty, P. Nanda, E. Kougianos, and G. Das, “Proof-of-authentication for scalable blockchain in resource-constrained distributed systems,” in *2019 IEEE International Conference on Consumer Electronics (ICCE).* IEEE, 2019, pp. 1–5.

[110] Raspberry pi datasheet. Raspberry Pi Organisation. Accessed: 13 January 2019. [Online]. Available: https://www.raspberrypi.org/app/uploads/2018/11/Raspberry_Pi_3A_product_brief.pdf

[111] Mixed Signal Microcontroller. Texas Instruments. Accessed: 13 January 2019. [Online]. Available: https://www.ti.com/lit/ds/symlink/msp430g2553.pdf?ts=1595406297499

[112] A. Jain, R. Bansal, A. Kumar, and K. Singh, "A comparative study of visual and auditory reaction times on the basis of gender and physical activity levels of medical first year students," *International Journal of Applied Basic Medical Research*, 2014.

[113] H. Lee, H. Mun, and Y. Lee, "Comparing response time of home iot devices with or without cloud," in *2020 IEEE International Conference on Consumer Electronics (ICCE)*, Jan 2020, pp. 1–6.

[114] R. Vilalta, R. Ciungu, A. Mayoral, R. Casellas, R. Martinez, D. Pubill, J. Serra, R. Munoz, and C. Verikoukis, "Improving security in Internet of Things with software defined networking," in *2016 IEEE Global Communications Conference (GLOBECOM)*. IEEE, 2016, pp. 1–6.

[115] Z. Li, A. Goyal, and Y. Chen, "Honeynet-based botnet scan traffic analysis," in *Botnet Detection*. Springer, 2008, pp. 25–44.

[116] K. Wang, C.-Y. Huang, S.-J. Lin, and Y.-D. Lin, "A fuzzy pattern-based filtering algorithm for botnet detection," *Computer Networks*, vol. 55, no. 15, pp. 3275–3286, 2011.

[117] G.-H. Lai, C.-M. Chen, R.-Y. Tzeng, C.-S. Laih, and C. Faloutsos, "Botnet detection by abnormal IRC traffic analysis," in *Proceedings of the Fourth Joint Workshop on Information Security (JWIS)*, 2009.

[118] D. Zhao, I. Traore, B. Sayed, W. Lu, S. Saad, A. Ghorbani, and D. Garant, "Botnet detection based on traffic behavior analysis and flow intervals," *Computers & Security*, vol. 39, pp. 2–16, 2013.

[119] A. Sivanathan, D. Sherratt, H. H. Gharakheili, V. Sivaraman, and A. Vishwanath, "Low-cost flow-based security solutions for smart-home IoT devices," in *2016 IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS).* IEEE, 2016, pp. 1–6.

[120] "Spectrum value of 800MHz, 1800MHz and 2.6GHz A DotEcon and Aetha Report for Ofcom," DotEcon and Aetha, Tech. Rep., 2012. [Online]. Available: https://www.ofcom.org.uk/__data/assets/pdf_file/0016/51208/spectrum-value.pdf

[121] Z. Zhang, Z. Zhao, H. Guan, D. Miao, and Z. Tan, "Study of signaling overhead caused by keep-alive messages in lte network," in *2013 IEEE 78th vehicular technology conference (VTC Fall).* IEEE, 2013, pp. 1–5.

[122] "Sigfox coverage map," Sigfox, Mar. 2018. [Online]. Available: https://www.sigfox.com/en/coverage

[123] "Short Range Devices operating in the 863 - 870 MHz Short Range Devices operating in the 863 - 870 MHz frequency band," The Office of Communications (Ofcom), Tech. Rep., 2010. [Online]. Available: https://www.ofcom.org.uk/__data/assets/pdf_file/0025/38095/final_report.pdf

[124] LoRa Link Budget Sensitivity Calculations Example Explained. [Online]. Available: http://www.techplayon.com/lora-link-budget-sensitivity-calculations-example-explained/

[125] E. Aras, N. Small, G. S. Ramachandran, S. Delbruel, W. Joosen, and D. Hughes, "Selective Jamming of LoRaWAN using Commodity Hardware," *CoRR*, vol. abs/1712.02141, 2017. [Online]. Available: http://arxiv.org/abs/1712.02141

[126] N. Sornin, "LoRaWAN™ 1.0.3 Specification," LoRa Alliance, Tech. Rep., 2017. [Online]. Available: https://lora-alliance.org/sites/default/files/2018-05/lorawan1_0_2-20161012_1398_1.pdf

[127] ——, "LoRaWAN™ 1.1 Specification," LoRa Alliance, Tech. Rep., 2017. [Online]. Available: https://lora-alliance.org/sites/default/files/2018-04/lorawantm_specification_-v1.1.pdf

[128] M. S. Cesar Cerrudo, Esteban Martinez Fayo, "LoRaWAN Networks Susceptible to Hacking: Common Cyber Security Problems, How to Detect and Prevent Them," Ioactive, Tech. Rep., Jan. 2020, accessed: 27 Jun 2020. [Online]. Available: https://act-on.ioactive.com/acton/attachment/34793/f-87b45f5f-f181-44fc-82a8-8e53c501dc4e/1/-/-/-/-/LoRaWAN%20Networks%20Susceptible%20to%20Hacking.pdf

[129] X. Yang, E. Karampatzakis, C. Doerr, and F. Kuipers, "Security vulnerabilities in LoRaWAN," in *2018 IEEE/ACM Third International Conference on Internet-of-Things Design and Implementation (IoTDI)*. IEEE, 2018, pp. 129–140.

[130] N. Sornin, "LoRaWAN 1.1 Specification," Dept. Elect. LoRa Alliance, Inc. 3855 SW 153rd Drive, Beaverton, OR 97003, Version 1.1 2017.

[131] R. Ratasuk, B. Vejlgaard, N. Mangalvedhe, and A. Ghosh, "NB-IoT system for M2M communication," in *2016 IEEE Wireless Communications and Networking Conference Workshops (WCNCW)*, 2016, pp. 428–432.

[132] ——, "NB-IoT system for M2M communication," in *2016 IEEE Wireless Communications and Networking Conference*, April 2016, pp. 1–5.

[133] M. Lauridsen, I. Z. Kovács, P. Mogensen, M. Sorensen, and S. Holst, "Coverage and capacity analysis of LTE-M and NB-IoT in a rural area," in *2016 IEEE 84th Vehicular Technology Conference (VTC-Fall)*. IEEE, 2016, pp. 1–5.

[134] "Emergency Services Network: overview," United Kingdom Home Office, Tech. Rep., 2020, accessed on 4 April 2020. [Online]. Available: https://www.gov.uk/government/publications/the-emergency-services-mobile-communications-programme/emergency-services-network

[135] "Scottish 4G Infill Program: Progress Update," Scottish Goverment, Tech. Rep., 2020, accessed: 30th March 2020. [Online]. Available: https://www.gov.scot/publications/scottish-4g-infill-programme-progress-update/

[136] "Mobile Infastructure Project Impact and Benefits Report," UK Goverment, Department for Culture Media & Sport, Tech. Rep., accessed: 30th March 2020. [Online]. Available: https://assets.publishing.service.gov.uk/government/uploads/system/uploads/attachment_data/file/651008/MIP_Impact_and_Benefits_Report.pdf

[137] E. Spafford and A. Dewdney, "Computer recreations: Of worms, viruses and core war," *Scientific American*, pp. 1999–2007, 1989.

[138] A. Shostack, "Experiences threat modeling at microsoft." *MODSEC@ MoDELS*, vol. 2008, 2008.

[139] J. Hong, "The State of Phishing Attacks," *Commun. ACM*, vol. 55, no. 1, p. 74–81, Jan. 2012. [Online]. Available: https://doi.org/10.1145/2063176.2063197

[140] M. Weiß, B. Heinz, and F. Stumpf, "A cache timing attack on aes in virtualization environments," in *Financial Cryptography and Data Security*, A. D. Keromytis, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 314–328.

[141] N. Lomas. (2015, Jan.) The FTC Warns Internet Of Things Businesses To Bake In Privacy And Security. [Online]. Available: https://techcrunch.com/2015/01/08/ftc-iot-privacy-warning/

[142] "Cyber essentials scheme: Overview," National Cyber Security Centre UK Government, Tech. Rep., 2014, accessed 10th May 2020. [Online]. Available: https://www.ncsc.gov.uk/cyberessentials/overview

[143] E. Bertino and N. Islam, "Botnets and Internet of Things Security," *Computer*, vol. 50, no. 2, pp. 76–79, 2017.

[144] D. Bardnard-wills, L. Marinos, and S. Portesi, “Threat landscape and good practice guide for smart home and converged media,” European Union Agency for Network and Information Security (ENISA), Attiki, Greece, Tech. Rep. 978-92-9204-096-3, 2014. [Online]. Available: https://www.enisa.europa.eu/publications/threat-landscape-for-smart-home-and-media-convergence/at_download/fullReport

[145] C. Lévy-Bencheton, E. Darra, G. Tétu, G. Dufay, and M. Alattar, “Security and resilience of smart home environments good practices and recommendations,” European Union Agency for Network and Information Security (ENISA), Attiki, Greece, Tech. Rep. 978-92-9204-141-0, 2015. [Online]. Available: https://www.enisa.europa.eu/publications/security-resilience-good-practices/at_download/fullReport

[146] P. Morgner and Z. Benenson, “Exploring security economics in IoT standardization efforts,” *arXiv preprint arXiv:1810.12035*, 2018. [Online]. Available: https://arxiv.org/pdf/1810.12035.pdf

[147] M. Thomas, “Written Evidence to the Joint Committee on the National Security Strategy, 'UK national security in a digital world',” 2 2017”. [Online]. Available: http://data.parliament.uk/writtenevidence/committeeevidence.svc/evidencedocument/national-security-strategy-committee/cyber-security-uk-national-security-in-a-digital-world/written/47405.pdf

[148] “Common position on cybersecurity,” Infineon, NXP, STMicroElectronics, European Union Agency for Network and Information Security, Tech. Rep., Dec. 2016. [Online]. Available: https://www.enisa.europa.eu/publications/enisa-position-papers-and-opinions/infineon-nxp-st-enisa-position-on-cybersecurity

[149] H. Bauer, O. Burkacky, and C. Knochenhauer, "Security in the Internet of Things," McKinsey & Company, Tech. Rep., May 2017. [Online]. Available: https://www.mckinsey.com/~/media/McKinsey/Industries/Semiconductors/Our%20Insights/Security%20in%20the%20Internet%20of%20Things/Security-in-the-Internet-of-Things.ashx

[150] G. A. Akerlof, "The market for "lemons": Quality uncertainty and the market mechanism," in *Uncertainty in economics.* Elsevier, 1978, pp. 235–251.

[151] R. W. Hahn and A. Layne-Farrar, "The law and economics of software security," *Harvard Journal of Law and Public Policy*, vol. 30, 2006.

[152] G. Paul and J. Irvine, "Investigating the security of android security applications," in *9th CMI Conference on Smart Living, Cyber Security and Privacy*, 2016.

[153] (2107) Teardown - google home. IHS Markit. [Online]. Available: https://benchmarking.ihsmarkit.com/587659/teardown-google-home

[154] K. Cao and A. K. Jain, "Hacking mobile phones using 2d printed fingerprints," *Michigan State University, Tech. Report. MSU-CSE-16-2*, 2016.

[155] I. Cherapau, "Towards understanding how touch id impacts users' authentication secrets selection for iphone lock," Ph.D. dissertation, University of British Columbia, 2015.

[156] S. Laplante, "The european union's general product safety directive: Another call for us exporters to comply with the iso 9000 series," *Syracuse J. Int'l L. & Com.*, vol. 22, p. 155, 1996.

[157] V. Gupta, C. Saini, M. Oberoi, G. Kalra, and M. I. Nasir, "Semmelweis reflex: An age-old prejudice," *World Neurosurgery*, 2019.

[158] Y. Zou, A. H. Mhaidli, A. McCall, and F. Schaub, "" i've got nothing to lose": Consumers' risk perceptions and protective actions after the equifax data breach," in *Fourteenth Symposium on Usable Privacy and Security ({SOUPS} 2018)*, 2018, pp. 197–216.

[159] S. Gressin, "The equifax data breach: What to do," Federal Trade Commission, Tech. Rep., September 2017. [Online]. Available: https://www.consumer.ftc.gov/blog/2017/09/equifax-data-breach-what-do

[160] C. Millard, W. K. Hon, and J. Singh, "Internet of things ecosystems: unpacking legal relationships and liabilities," in *2017 IEEE International Conference on Cloud Engineering (IC2E)*. IEEE, 2017, pp. 286–291.

[161] Accessed 24 March 2020. [Online]. Available: https://www.ftc.gov/news-events/press-releases/2019/07/equifax-pay-575-million-part-settlement-ftc-cfpb-states-related

[162] D. Harding, "Oral evidence: Cyber security: Protection of personal data online, high court 587," UK Government, Tech. Rep. [Online]. Available: http://data.parliament.uk/writtenevidence/committeeevidence.svc/evidencedocument/culture-media-and-sport-committee/cyber-security-protection-of-personal-data-online/oral/26312.html

[163] A. Marotta, F. Martinelli, S. Nanni, A. Orlando, and A. Yautsiukhin, "Cyber-insurance survey," *Computer Science Review*, vol. 24, pp. 35–61, 2017.

[164] A. Bhatia, Z. Yusuf, D. Ritter, and N. Hunke, "Who will win the iot platform wars?" BCG Perspectives, Tech. Rep., 2017.

[165] P. Cartwright, "Understanding and protecting vulnerable financial consumers," *Journal of Consumer Policy*, vol. 38, no. 2, pp. 119–138, 2015.

[166] "How FSCS protects your money," Financial Services Compensation Scheme, Tech. Rep., 2015. [Online]. Available: https://www.fscs.org.uk/globalassets/disclosure-materials/fscs_2015_online-leaflet.pdf

[167] H. Taneja, "The era of "move fast and break things" is over," Harvard Business Review, Tech. Rep., 2019. [Online]. Available: https://hbr.org/2019/01/the-era-of-move-fast-and-break-things-is-over

[168] M. O'Neill *et al.*, "Insecurity by design: Today's IoT device security problem," *Engineering*, vol. 2, no. 1, pp. 48–49, 2016.

[169] B. Lee and J.-H. Lee, "Blockchain-based secure firmware update for embedded devices in an internet of things environment," *The Journal of Supercomputing*, vol. 73, no. 3, pp. 1152–1167, 2017.

[170] Learn when you'll get android updates on pixel phones & nexus devices. Google. [Online]. Available: https://support.google.com/nexus/answer/4457705?hl=en

[171] "Taking charge of the IoT's security vulnerabilities," Canonical, Tech. Rep., 2017. [Online]. Available: https://pages.ubuntu.com/rs/066-EOV-335/images/IoTSecurityWhitepaper-FinalReport.pdf

[172] K. E. Vaniea, E. Rader, and R. Wash, "Betrayed by updates: how negative experiences affect future security," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 2014, pp. 2671–2674.

[173] P. Morgner, F. Freiling, and Z. Benenson, "Opinion: Security lifetime labels-Overcoming information asymmetry in security of IoT consumer products," in *Proceedings of the 11th ACM Conference on Security & Privacy in Wireless and Mobile Networks*, 2018, pp. 208–211.

[174] S. Ala-Kurikka. (2015, Mar.) Lifespan of consumer electronics is getting shorter, study finds. [Online]. Available: https://www.theguardian.com/environment/2015/mar/03/lifespan-of-consumer-electronics-is-getting-shorter-study-finds

[175] A. Gilbert, "The time that tony fadell sold me a container of hummus," Apr. 2016. [Online]. Available: https://arlogilbert.com/the-time-that-tony-fadell-sold-me-a-container-of-hummus-cb0941c762c1

[176] A. Hern. (2016, April) Revolv devices bricked as google's nest shuts down smart home company. [Online]. Available: https://www.theguardian.com/technology/2016/apr/05/revolv-devices-bricked-google-nest-smart-home

[177] B. Schoon. Youtube Leanback. Accessed: 20th March 2020. [Online]. Available: https://9to5google.com/2019/09/16/youtube-leanback-tv-interface-disabled-soon/

[178] Nest. Accessed: 13-Aug-2018. [Online]. Available: https://nest.com/whats-happening/whats-happening-to-the-works-with-nest-program

[179] Google wave. Accessed: 13-Aug-2018. [Online]. Available: https://support.google.com/answer/1083134?hl=en

[180] Google reader. Accessed: 13-Aug-2018. [Online]. Available: https://support.google.com/answer/1083134?hl=en

[181] "Developer survey results 2019." StackOverflow, Tech. Rep., 2019. [Online]. Available: https://insights.stackoverflow.com/survey/2019/most-loved-dreaded-and-wanted

[182] A. Dunkels, B. Gronvall, and T. Voigt, "Contiki-a lightweight and flexible operating system for tiny networked sensors," in *29th annual IEEE international conference on local computer networks.* IEEE, 2004, pp. 455–462.

[183] Q. Cao and T. Abdelzaher, "Liteos: a lightweight operating system for c++ software development in sensor networks," in *Proceedings of the 4th international conference on Embedded networked sensor systems*, 2006, pp. 361–362.

[184] L. Kelion. Why amazon knows so much about you. [Online]. Available: https://www.bbc.co.uk/news/extra/CLQYZENMBI/amazon-data

[185] C. Cadwalladr and E. Graham-Harrison. (2018) Revealed: 50 million Facebook profiles harvested for Cambridge Analytica in major data breach.

[186] J. Angwin, T. Parris Jr, and S. Mattu. (2016, December) Facebook doesn't tell users everything it really knows about them. ProPublica.

[187] B. D. Weinberg, G. R. Milne, Y. G. Andonova, and F. M. Hajjat, “Internet of things: Convenience vs. privacy and secrecy,” *Business Horizons*, vol. 58, no. 6, pp. 615–624, 2015.

[188] Y. Lee, W. Hsiao, C. Huang, and S. T. Chou, “An integrated cloud-based smart home management system with community hierarchy,” *IEEE Transactions on Consumer Electronics*, vol. 62, no. 1, pp. 1–9, 2016.

[189] S. Notra, M. Siddiqi, H. H. Gharakheili, V. Sivaraman, and R. Boreli, “An experimental study of security and privacy risks with emerging household appliances,” in *Communications and Network Security (CNS), 2014 IEEE Conference on.* IEEE, 2014, pp. 79–84.

[190] A. Tierney. (2016, Feb.) Pwning CCTV cameras. [Online]. Available: https://www.pentestpartners.com/blog/pwning-cctv-cameras/

[191] D. Lodge, “Hacking the IP camera (part 1),” Apr. 2015. [Online]. Available: https://www.pentestpartners.com/blog/hacking-the-ip-camera-part-1/

[192] ——, “Hacking the IP camera (part 2),” Oct. 2015. [Online]. Available: https://www.pentestpartners.com/blog/hacking-the-aldi-ip-cctv-camera-part-2/

[193] “Senate Bill No. 327, Chapter 886,” California Senate Legislature, Sacramento, CA, Tech. Rep. 327, 2018. [Online]. Available: https://leginfo.legislature.ca.gov/faces/billTextClient.xhtml?bill%5Fid=201720180SB327

[194] “Government response to the regulatory proposals for consumer internet of things (iot) security consultation,” United Kingdom Department for Digital, Culture, Media and Sport, Tech. Rep., 2020. [Online]. Available: https://www.gov.uk/government/consultations/consultation-on-regulatory-proposals-on-consumer-iot-security/outcome/government-response-to-the-regulatory-proposals-for-consumer-internet-of-things-iot-security-consultation

[195] ETSI TS 103 645, "Cyber security for consumer internet of things," European Telecommunications Standards Institute, Cedex, France, Tech. Rep. 1.1.1, 2019. [Online]. Available: https://www.etsi.org/deliver/etsi_ts/103600_103699/103645/01.01.01_60/ts_103645v010101p.pdf

[196] M. Warner, "S.734 - internet of things cybersecurity improvement act of 2019," Senate of the United States, Washington, D.C., Tech. Rep. 215, 2019, accessed: 5 November 2019. [Online]. Available: https://www.congress.gov/bill/116th-congress/senate-bill/734/

[197] "Code of Practice for Consumer IoT Security," Department for Digital, Culture, Media and Sport (DCMS), in conjunction with the National Cyber Security Centre (NCSC), London, United Kingdom, Tech. Rep. 1, 2018. [Online]. Available: https://assets.publishing.service.gov.uk/government/uploads/system/uploads/attachment%5Fdata/file/773867/Code%5Fof%5FPractice%5Ffor%5FConsumer%5FIoT%5FSecurity%5FOctober%5F2018.pdf

[198] General Data Protection Regulation, "Regulation (eu) 2016/679 of the european parliament and of the council of 27 april 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing directive 95/46," European Parliament, Tech. Rep. 1-88, 2016. [Online]. Available: http://data.europa.eu/eli/reg/2016/679/oj

[199] A. D. Thierer, "The Internet of Things and Wearable Technology: Addressing Privacy and Security Concerns without Derailing Innovation," *Richmond Journal of Law and Technology*, vol. 21, 2015. [Online]. Available: http://jolt.richmond.edu/v21i2/article6.pdf

[200] "Vizio to pay $2.2 million to ftc, state of new jersey to settle charges it collected viewing histories on 11 million smart televisions without users' consent," Federal Trade Commission, Tech. Rep., February 2017, press release. [Online]. Available: https://www.ftc.gov/news-events/press-releases/2017/02/vizio-pay-22-million-ftc-state-new-jersey-settle-charges-it

[201] L. Franceschi-Bicchierai, "How this internet of things stuffed animal can be remotely turned into a spy device," 2017. [Online]. Available: https://www.vice.com/en_us/article/qkm48b/how-this-internet-of-things-teddy-bear-can-be-remotely-turned-into-a-spy-device

[202] P. Tobin, M. McKeever, J. Blackledge, M. Whittington, and B. Duncan, "UK Financial Institutions Stand to Lose Billions in GDPR Fines: How can They Mitigate This?" in *The British Accounting and Finance Association Scottish Area Group Conference, BAFA, Ed., Aberd*, 2017.

[203] "The CNIL's restricted committee imposes a financial penalty of 50 Million euros against GOOGLE LLC," Commission nationale de l'informatique et des libertés, Tech. Rep., 2019. [Online]. Available: https://www.cnil.fr/en/cnils-restricted-committee-imposes-financial-penalty-50-million-euros-against-google-llc

[204] "Intention to fine british airways £183.39m under gdpr for data breach," Information Commissioner's Office, Tech. Rep., 2019. [Online]. Available: https://ico.org.uk/about-the-ico/news-and-events/news-and-blogs/2019/07/ico-announces-intention-to-fine-british-airways/

[205] "Statement: Intention to fine marriott international, inc more than £99 million under gdpr for data breach," Information Commissioner's Office, Tech. Rep., 2019. [Online]. Available: https://ico.org.uk/about-the-ico/news-and-events/news-and-blogs/2019/07/statement-intention-to-fine-marriott-international-inc-more-than-99-million-under-gdpr-for-data-breach/

[206] B. Barrett, "Hack Brief: Marriott Got Hacked. Yes, Again," Wired, Tech. Rep. [Online]. Available: https://www.wired.com/story/marriott-hacked-yes-again-2020/

[207] "Statistics: Fines by type of violation," Enforcement Tracker, Tech. Rep. [Online]. Available: https://www.enforcementtracker.com/?insights

[208] M. Goddard, "The EU General Data Protection Regulation (GDPR): European Regulation that has a Global Impact," *International Journal of Market Research*, vol. 59, no. 6, pp. 703–705, 2017. [Online]. Available: https://doi.org/10.2501/IJMR-2017-050

[209] "Directive (eu) 2016/1148 of the european parliament and of the council of 6 july 2016 concerning measures for a high common level of security of network and information systems across the union," European Parliament, Tech. Rep. 19.7, 2016.

[210] J. Veaney, "RIIO-ED2 Framework Decision," Office of Gas and Electricity Markets, Tech. Rep., 2020. [Online]. Available: https://www.ofgem.gov.uk/system/files/docs/2020/01/riio-ed2_framework_decision_jan_2020.pdf

[211] "Progress of the 2016–2021 national cyber security programme," National Audit Office, Tech. Rep., March 2019. [Online]. Available: https://www.nao.org.uk/wp-content/uploads/2019/03/Progress-of-the-2016-2021-National-Cyber-Security-Programme.pdf

[212] S.Singh, "Consumer IoT Market worth $104.4 billion by 2023" MarketsandMarkets, Northbrook, Illinois, SE 6810 2018.

[213] D. Bryan. Lack of Web and API Authentication in INSTEON Hub. Trustwave SpiderLabs. (8 January 2013). [Online]. Available: https://www.trustwave.com/en-us/resources/security-resources/security-advisories/?fid=18876

[214] S. Margaritelli. Reversing The Smarter Coffee Iot Machine Protocol To Make Coffee Using The Terminal. (09 October 2016). [Online]. Available: https://www.evilsocket.net/2016/10/09/IoCOFFEE-Reversing-the-Smarter-Coffee-IoT-machine-protocol-to-make-coffee-using-terminal/

[215] G. Kambourakis, C. Kolias, and A. Stavrou, "The Mirai botnet and the IoT Zombie Armies," in *MILCOM 2017 - 2017 IEEE Military Communications Conference (MILCOM)*, Oct. 2017, pp. 267–272.

[216] S. Briciu *et al.*, “Variable and fixed costs in company management,” *Annales Universitatis Apulensis Series Oeconomica*, vol. 1, no. 10, pp. 1–14, 2008.

[217] R. Mahmoud, T. Yousuf, F. Aloul, and I. Zualkernan, “Internet of things (iot) security: Current status, challenges and prospective measures,” in *2015 10th International Conference for Internet Technology and Secured Transactions (ICITST)*, 2015, pp. 336–341.

[218] Microchip. (17 July 2019). [Online]. Available: https://www.microchip.com/wwwproducts/en/ATECC508A

[219] “Global mobile consumer survey 2018,” deloitte, Tech. Rep., 2018, accessed 19 March 2019. [Online]. Available: http://www.deloitte.co.uk/mobileuk/assets/img/download/Global-Mobile-Consumer-Survey-2018-UK-Cut-State-of-the-smartphone.pdf

[220] G. M. Toschi, L. B. Campos, and C. E. Cugnasca, “Home automation networks: A survey,” *Computer Standards & Interfaces*, vol. 50, pp. 42–54, 2017.

[221] S. Bluetooth, “Bluetooth market update 2019,” Tech. Rep., 2019. [Online]. Available: https://www.bluetooth.com/wp-content/uploads/2018/04/2019-Bluetooth-Market-Update.pdf

[222] M. Siekkinen, M. Hiienkari, J. K. Nurminen, and J. Nieminen, “How low energy is bluetooth low energy? comparative measurements with zigbee/802.15.4,” in *2012 IEEE Wireless Communications and Networking Conference Workshops (WCNCW)*, 2012, pp. 232–237.

[223] M. P. Ebrom, “Method of connecting an appliance to a wifi network,” European Patent EP2 814 273A1, Jun 10, 2013.

[224] Marvell, *88DE3006 High-Definition Secure Media Processor System-on-Chip (SoC)*, 2012.

[225] A. K. Yau, K. G. Paterson, and C. J. Mitchell, “Padding oracle attacks on CBC-mode encryption with secret and random IVs,” in *International Workshop on Fast Software Encryption.* Springer, 2005, pp. 299–319.

[226] N. J. Al Fardan and K. G. Paterson, “Lucky thirteen: Breaking the TLS and DTLS record protocols,” in *2013 IEEE Symposium on Security and Privacy.* IEEE, 2013, pp. 526–540.

[227] M. Bellare, T. Kohno, and C. Namprempre, “Breaking and provably repairing the ssh authenticated encryption scheme: A case study of the encode-then-encrypt-and-mac paradigm,” *ACM Transactions on Information and System Security (TISSEC)*, vol. 7, no. 2, pp. 206–241, 2004.

[228] H. Krawczyk, “The order of encryption and authentication for protecting communications (or: How secure is SSL?),” in *Annual International Cryptology Conference.* Springer, 2001, pp. 310–331.

[229] V. Thangavelu, D. M. Divakaran, R. Sairam, S. S. Bhunia, and M. Gurusamy, “DEFT: A Distributed IoT Fingerprinting Technique,” *IEEE Internet of Things Journal*, vol. 6, no. 1, pp. 940–952, Feb. 2019.

[230] D. Nouichi, M. Abdelsalam, Q. Nasir, and S. Abbas, “IoT Devices Security Using RF Fingerprinting,” in *2019 Advances in Science and Engineering Technology International Conferences (ASET)*, Mar. 2019, pp. 1–7.

[231] H. Jafari, O. Omotere, D. Adesina, H. Wu, and L. Qian, “IoT Devices Fingerprinting Using Deep Learning,” in *MILCOM 2018 - 2018 IEEE Military Communications Conference (MILCOM)*, Oct. 2018, pp. 1–9.

[232] R. Mahy, P. Matthews, and J. Rosenberg, “Traversal using relays around NAT (TURN): Relay extensions to session traversal utilities for NAT (STUN),” Internet Requests for Comments, RFC Editor, Rfc 5766, Apr. 2010, accessed 20th May 2018. [Online]. Available: http://www.rfc-editor.org/rfc/rfc5766.txt

[233] C. H. A. Keranen and J. Rosenberg, "Interactive Connectivity Establishment (ICE): A Protocol for Network Address Translator (NAT) Traversal," Internet Requests for Comments, RFC Editor, Tech. Rep. 8445, May 2018. [Online]. Available: http://www.rfc-editor.org/rfc/rfc8445.txt

[234] J. Rosenberg, R. Mahy, P. Matthews, and D. Wing, "Session traversal utilities for NAT (STUN)," Internet Requests for Comments, RFC Editor, Rfc 5389, Oct. 2008, http://www.rfc-editor.org/rfc/rfc5389.txt. [Online]. Available: http://www.rfc-editor.org/rfc/rfc5389.txt

[235] J. Feichtner, "A comparative study of misapplied crypto in android and ios applications," in *16th International Conference on Security and Cryptography (SECRYPT 2019)*. SciTePress, 2019.

[236] C. Ziegler. Google unveils 'bouncer' service to automatically detect android market malware. The Verge. Accessed: 20 July 2019. [Online]. Available: https://www.theverge.com/2012/2/2/2766674/google-unveils-bouncer-service-to-automatically-detect-android-market

[237] Sign your app. Google. Accessed: 20 July 2019. [Online]. Available: https://developer.android.com/studio/publish/app-signing

[238] Google Cloud. Accessed on 18th May 2020. [Online]. Available: https://cloud.google.com/hsm/

[239] Amazon Web Services. Accessed on 18th May 2020. [Online]. Available: https://aws.amazon.com/cloudhsm/

[240] IBM. Accessed on 18th May 2020. [Online]. Available: https://www.ibm.com/cloud/hardware-security-module

[241] Micosoft Azure. Accessed on 18th May 2020. [Online]. Available: https://azure.microsoft.com/en-us/services/azure-dedicated-hsm

[242] P. Bhat and K. Dutta, “A survey on various threats and current state of security in android platform,” *ACM Comput. Surv.*, vol. 52, no. 1, Feb. 2019. [Online]. Available: https://doi.org/10.1145/3301285

[243] P. M. Hartmann, M. Zaki, N. Feldmann, and A. Neely, “Capturing value from big data–a taxonomy of data-driven business models used by start-up firms,” *International Journal of Operations & Production Management*, 2016.

[244] A. J. Pyles, X. Qi, G. Zhou, M. Keally, and X. Liu, “Sapsm: Smart adaptive 802.11 psm for smartphones,” in *Proceedings of the 2012 ACM conference on ubiquitous computing*, 2012, pp. 11–20.

[245] “IBSS with Wi-Fi Protected Setup Technical Specification,” Wi-Fi Alliance, Tech. Rep. [Online]. Available: https://www.wi-fi.org/download.php?file=/sites/default/files/private/IBSS_with_Wi-Fi_Protected_Setup_Technical_Specification_v1.0.0.pdf

[246] H. Lipmaa, P. Rogaway, and D. Wagner, “CTR-mode encryption,” in *First NIST Workshop on Modes of Operation*, vol. 39, 2000.

[247] M. Day, “Using power solutions to extend battery life in MSP430 applications,” *Analog Applications*, 2009.

[248] S. Tedeschi, D. Rodrigues, C. Emmanouilidis, J. Erkoyuncu, R. Roy, and A. Starr, “A cost estimation approach for IoT modular architectures implementation in legacy systems,” *Procedia Manufacturing*, vol. 19, pp. 103–110, 2018, proceedings of the 6th International Conference in Through-life Engineering Services, University of Bremen, 7th and 8th November 2017.

[249] E. B. Barker and A. L. Roginsky, “Transitions: Recommendation for Transitioning the Use of Cryptographic Algorithms and Key Lengths,” National Institute of Standards and Technology, Tech. Rep., 2015. [Online]. Available: 10.6028/nist.sp.800-131ar1

[250] Phillips hue dimmer switch. Accessed: 1 April 2020. [Online]. Available: https://www.philips-hue.com/en-us/p/hue-dimmer-switch/046677473372

[251] ESP8266EX Datasheet. Espressif. Accessed: 13 January 2019. [Online]. Available: https://www.espressif.com/sites/default/files/documentation/0a-esp8266ex_datasheet_en.pdf

[252] L. Guegan and A.-C. Orgerie, "Estimating the end-to-end energy consumption of low-bandwidth IoT applications for WiFi devices," 2019.

[253] L. Florio and K. Wierenga, "Eduroam, providing mobility for roaming users," in *Proceedings of the EUNIS 2005 Conference, Manchester*, 2005.

[254] F. K. Shaikh and S. Zeadally, "Energy harvesting in wireless sensor networks: A comprehensive review," *Renewable and Sustainable Energy Reviews*, vol. 55, pp. 1041–1054, 2016.

[255] A. Kara, "Protecting privacy in remote-patient monitoring," *Computer*, vol. 34, no. 5, pp. 24–27, May 2001.

[256] S. Kekade, C.-H. Hseieh, M. M. Islam, S. Atique, A. M. Khalfan, Y.-C. Li, and S. S. Abdul, "The usefulness and actual use of wearable devices among the elderly population," *Computer methods and programs in biomedicine*, vol. 153, pp. 137–159, 2018.

[257] L. Jenstad and J. Moon, "Systematic review of barriers and facilitators to hearing aid uptake in older adults," *Audiology Research*, vol. 1, no. 1, 2011.

[258] S. Meadows. (2017) Why care costs are spiralling at up to twice inflation. [Online]. Available: https://www.telegraph.co.uk/money/consumer-affairs/care-costs-spiralling-twice-inflation/

[259] B. Teja, I. Daniel, G. H. Pink, A. Brown, and D. J. Klein, "Ensuring adequate capital investment in canadian health care," *CMAJ*, vol. 192, no. 25, pp. E677–E683, 2020.

[260] E. Rachkidi, E. H. Cherkaoui, M. Ait-idir, N. Agoulmine, N. C. Taher, M. Santos, and S. Fernandes, “Towards Efficient Automatic Scaling and Adaptive Cost-Optimized eHealth Services in Cloud,” in *2015 IEEE Global Communications Conference (GLOBECOM)*, 2015, pp. 1–6.

[261] M. Morgan, “A Sustainable Lighting Strategy for Perth,” Perth and Kinross Council, Tech. Rep., December 2015.

[262] “Orkney transmission project – Conditional decision on Final Needs Case,” The Office of Gas and Electricity Markets (OFGEM), Tech. Rep., 2019. [Online]. Available: https://www.ofgem.gov.uk/system/files/docs/2019/09/conditional_decision_on_orkney_final_needs_case_2.pdf

[263] J. Ambler, “RIIO-ED1,” OFGEM, Tech. Rep., 2018. [Online]. Available: https://www.ofgem.gov.uk/system/files/docs/2019/03/riio-ed1_annual_report_2017-18.pdf

[264] “DNO - Smart Grid Communications Requirements,” Telent and Energy Networks Association, Tech. Rep., 2011. [Online]. Available: https://www.energynetworks.org/assets/files/electricity/engineering/telecoms/eitc/restricted/Reference%20Doc/Telent.pdf

[265] T. Newman, “Pole mounted fault passage indicators,” in *IEE Colloquium on Improving Supply Security on 11 kV Overhead Networks*, May 1990, pp. 7/1–7/2.

[266] K. Sliverstein, “In Light Of The Russian Attack, Utilities Seek New Defenses To Protect Critical Infrastructure,” Forbes, Tech. Rep., 2018. [Online]. Available: https://www.forbes.com/sites/kensilverstein/2018/03/18/in-light-of-the-russian-attack-utilities-seek-new-defenses-to-protect-critical-infrastructure/

[267] C. Glenn, D. Sterbentz, and A. Wright, “Cyber Threat and Vulnerability Analysis of the U.S. Electric Sector,” Idaho National Laboratory, Tech. Rep. [Online]. Available: https://www.energy.gov/sites/prod/files/2017/01/f34/Cyber%20Threat%20and%20Vulnerability%20Analysis%20of%20the%20U.S.%20Electric%20Sector.pdf

[268] D. E. Whitehead, K. Owens, D. Gammel, and J. Smith, "Ukraine cyber-induced power outage: Analysis and practical mitigation strategies," in *2017 70th Annual Conference for Protective Relay Engineers (CPRE)*. IEEE, 2017, pp. 1–8.

[269] "Control systems security program," Department of Homeland Security, Tech. Rep., 2007, accessed 21 March 2020. [Online]. Available: https://cdn.muckrock.com/foia_files/14f00304-Documents.pdf

[270] "European clocks lose six minutes after dispute saps power from electricity grid," The Guardian, Tech. Rep., 2018. [Online]. Available: https://www.theguardian.com/world/2018/mar/08/european-clocks-lose-six-minutes-dispute-power-electricity-grid

[271] I. Gilchrist, "Backup Power For Mobile Telecommunications: Market Analysis For Alternatively Fuelled Generators In Brazil, Argentina And Mexico," Master's thesis, Simon Fraser University, 2014.

[272] T. Jowitt. (2008) Backup power 'not required' in U.K. base stations. Tech World. [Online]. Available: https://www.networkworld.com/article/2284316/network-security/backup-power--not-required--in-u-k--base-stations.html

[273] T. Shields. (2017) Harvey Knockout of Cell Service Revives Talk of Backup Power. Bloomberg. [Online]. Available: https://www.bloomberg.com/news/articles/2017-08-30/harvey-s-knockout-of-cell-service-revives-calls-for-backup-power

[274] "Electricity Distribution (DPCR5): Glossary of Terms - Regulatory Instructions and Guidance: Version 2," The Office of Gas and Energy Markets (OFGEM), Tech. Rep., 2011, 03 June 2011. [Online]. Available: https://www.ofgem.gov.uk/ofgem-publications/46615/dpcr5glossarymaster1.pdf

[275] "Black start strategy," National Grid, Tech. Rep., 2017. [Online]. Available: https://www.nationalgrid.com/sites/default/files/documents/High%20Level%20Black%20Start%20Strategy.pdf

[276] “Connected nations 2019,” Ofcom, Tech. Rep., 2019. [Online]. Available: https://www.ofcom.org.uk/research-and-data/multi-sector-research/infrastructure-research/connected-nations-2019

[277] R. S. Sinha, Y. Wei, and S.-H. Hwang, “A survey on LPWA technology: LoRa and NB-IoT,” *ICT Express*, vol. 3, no. 1, pp. 14–21, 2017.

[278] R. Ratasuk, N. Mangalvedhe, A. Ghosh, and B. Vejlgaard, “Narrowband lte-m system for m2m communication,” in *2014 IEEE 80th vehicular technology conference (VTC2014-Fall)*. IEEE, 2014, pp. 1–5.

[279] F. Euchner, “Hunting the Sigfox: Wireless IoT Network Security,” Chaos Computer Club e.V., 2018, accessed: 03 Mar 2020. [Online]. Available: https://doi.org/10.5446/39352

[280] “Make things come alive in a secure way,” Sigfox, Tech. Rep., 2017, accessed 19 March 2019. [Online]. Available: https://www.sigfox.com/sites/default/files/1701-SIGFOX-White_Paper_Security.pdf

[281] LoRaWAN distance world record broken, twice. 766 km (476 miles) using 25mW transmission power. 30 July 2019. [Online]. Available: https://www.thethingsnetwork.org/article/lorawan-distance-world-record

[282] L. Ratliff, Tech. Rep., 2019, loRaWAN Live Berlin Presentation. [Online]. Available: https://lora-alliance.org/sites/default/files/2019-06/technical_track_final_web.pdf

[283] D. Margrave, “GSM Security and Encryption,” *George Mason University*, pp. 1–5, 1999.

[284] D. Forsberg, G. Horn, W.-D. Moeller, and V. Niemi, *LTE security*. John Wiley & Sons, 2012.

[285] J. Cao, M. Ma, H. Li, Y. Zhang, and Z. Luo, “A survey on security aspects for lte and lte-a networks,” *IEEE communications surveys & tutorials*, vol. 16, no. 1, pp. 283–302, 2013.

[286] A. Mahboob and J. Zubairi, "Intrusion avoidance for scada security in industrial plants," in *2010 International Symposium on Collaborative Technologies and Systems.* IEEE, 2010, pp. 447–452.

[287] 450MHz Alliance, 2020, accessed 24th April 2020. [Online]. Available: https://450alliance.org/

[288] "Radio spectrum policy group work programme for 2020 and beyond," Joint Radio Company, Tech. Rep., 2019. [Online]. Available: https://www.jrc.co.uk/Plugin/Publications/assets/pdf/ICT-JRC-response-to-the-Public.pdf

[289] R. El Chall, S. Lahoud, and M. El Helou, "LoRaWAN network: radio propagation models and performance evaluation in various environments in Lebanon," *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 2366–2378, 2019.

[290] H. Mukkamala, "Arm Expands IoT Connectivity and Device Management Capabilities with Stream Technologies Acquisition," Jun. 2018. [Online]. Available: https://www.arm.com/company/news/2018/06/arm-expands-iot-connectivity-and-device-management-capabilities-with-stream-technologies-acquisition

[291] B. C. Fargas and M. N. Petersen, "GPS-free geolocation using LoRa in low-power WANs," in *2017 Global Internet of Things Summit (GIoTS)*, Jun. 2017, pp. 1–6.

[292] W. Xu, S. Jha, and W. Hu, "Exploring the Feasibility of Physical Layer Key Generation for LoRaWAN," in *2018 17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/ 12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE)*, 2018, pp. 231–236.

[293] "Communications Market Report 2019," The Office Of Communications (Ofcom), Tech. Rep., July 2019. [Online]. Available: https://www.ofcom.org.uk/__data/assets/pdf_file/0028/155278/communications-market-report-2019.pdf

[294] "Radio equipment regulations 2017," United Kingdom Department for Business, Energy & Industrial Strategy, Tech. Rep., Dec. 2017. [Online]. Available: https://assets.publishing.service.gov.uk/government/uploads/system/uploads/attachment_data/file/664824/radio-equipment-regulations-2017-guidance.pdf

[295] J. M. Kahn and J. R. Barry, "Wireless infrared communications," *Proceedings of the IEEE*, vol. 85, no. 2, pp. 265–298, 1997.

[296] J.-h. Choi, E.-b. Cho, T.-G. Kang, and C. G. Lee, "Pulse width modulation based signal format for visible light communications," in *OECC 2010 Technical Digest*. Ieee, 2010, pp. 276–277.

[297] S. Park, D. Kim, and H. Cha, "Reducing energy consumption of alarm-induced wake-ups on android smartphones," in *Proceedings of the 16th International Workshop on Mobile Computing Systems and Applications*, 2015, pp. 33–38.

[298] E. Tsimbalo, X. Fafoutis, and R. J. Piechocki, "CRC error correction in IoT applications," *IEEE Transactions on Industrial Informatics*, vol. 13, no. 1, pp. 361–369, 2016.

[299] C. Yoon, S. Lee, Y. Choi, R. Ha, and H. Cha, "Accurate power modeling of modern mobile application processors," *Journal of Systems Architecture*, vol. 81, pp. 17–31, 2017.

[300] Opto Semiconductors. Accessed 1st March 2020. [Online]. Available: https://dammedia.osram.info/media/resource/hires/osram-dam-14425800/CW2%20FKAQ81.Z1_EN.pdf

[301] M. Eldefrawy, I. Butun, N. Pereira, and M. Gidlund, "Formal security," *Computer Networks*, vol. 148, pp. 328–339, 2019.

[302] L. K. Grover, "A fast quantum mechanical algorithm for database search," in *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, 1996, pp. 212–219. [Online]. Available: https://arxiv.org/pdf/quant-ph/9605043.pdf

[303] M. A. Arfaoui, M. D. Soltani, I. Tavakkolnia, A. Ghrayeb, M. Safari, C. Assi, and H. Haas, "Physical layer security for visible light communication systems: A survey," *IEEE Communications Surveys & Tutorials*, 2020.

[304] M. A. Arfaoui, A. Ghrayeb, and C. Assi, "Secrecy rate closed-form expressions for the SISO VLC wiretap channel with discrete input signaling," *IEEE Communications Letters*, vol. 22, no. 7, pp. 1382–1385, 2018.

[305] L. Sharma and M. Mathuria, "Mobile banking transaction using fingerprint authentication," in *2018 2nd International Conference on Inventive Systems and Control (ICISC)*, 2018, pp. 1300–1305.

[306] Postscapes. (2017, Jan.) IoT Devices & Products. [Online]. Available: http://www.postscapes.com/internet-of-things-award/winners/#most-popular

[307] (2020) Introducing wink subscription. Wink. [Online]. Available: https://blog.wink.com/wink-blog/2020/5/6/introducing-wink-subscription

[308] M. Patton, E. Gross, R. Chinn, S. Forbis, L. Walker, and H. Chen, "Uninvited Connections: A Study of Vulnerable Devices on the Internet of Things (IoT)," in *Intelligence and Security Informatics Conference (JISIC), 2014 IEEE Joint*, Sep. 2014, pp. 232–235.

[309] R. Bodenheim, J. Butts, S. Dunlap, and B. Mullins, "Evaluation of the ability of the shodan search engine to identify internet-facing industrial control devices," *International Journal of Critical Infrastructure Protection*, vol. 7, no. 2, pp. 114–123, 2014.

[310] B. Genge and C. Enăchescu, "ShoVAT: Shodan-based vulnerability assessment tool for Internet-facing services," *Security and communication networks*, vol. 9, no. 15, pp. 2696–2714, 2016.

[311] C. Kolias, G. Kambourakis, A. Stavrou, and J. Voas, "DDoS in the IoT: Mirai and other botnets," *Computer*, vol. 50, no. 7, pp. 80–84, 2017.

[312] O. Klaba. (2016, Sep.) Twitter. [Online]. Available: https://twitter.com/olesovhcom/status/779297257199964160

[313] N. Woolf. DDoS attack that disrupted internet was largest of its kind in history, experts say. The Guardian. [Online]. Available: https://www.theguardian.com/technology/2016/oct/26/ddos-attack-dyn-mirai-botnet

[314] D. Lodge. (2016, Jan.) Steal your Wi-Fi key from your doorbell? IoT WTF! [Online]. Available: https://www.pentestpartners.com/blog/steal-your-wi-fi-key-from-your-doorbell-iot-wtf/

[315] L. Wang and J. Kangasharju, "Measuring large-scale distributed systems: case of bittorrent mainline DHT," in *IEEE P2P 2013 Proceedings*. IEEE, 2013, pp. 1–10.

[316] A. Hill, "Adding the next million peers to IPFS," Textile, Tech. Rep. [Online]. Available: https://medium.com/textileio/adding-the-next-million-peers-to-ipfs-76d356352d14

[317] "Ethereum by the numbers," ConsenSys, Tech. Rep., accessed 30th November 2019. [Online]. Available: https://media.consensys.net/ethereum-by-the-numbers-3520f44565a9

[318] A. Johnston, S. Donovan, R. Sparks, C. Cunningham, and K. Summers, "Session initiation protocol (SIP) basic call flow examples," Network Working Group, Tech. Rep., 2003. [Online]. Available: https://tools.ietf.org/html/rfc3665

[319] C. Holmberg, S. Hakansson, and G. Eriksson, "Web real-time communication use cases and requirements," Internet Engineering Task Force, Tech. Rep., 2015. [Online]. Available: https://tools.ietf.org/html/rfc7478

[320] J. Rosenberg, R. Mahy, P. Matthews, and D. Wing, "Session traversal utilities for NAT (STUN)," RFC 5389 (Proposed Standard), Tech. Rep., 2008. [Online]. Available: https://tools.ietf.org/html/rfc5389

[321] A. Balalaie, A. Heydarnoori, and P. Jamshidi, "Microservices architecture enables devops: Migration to a cloud-native architecture," *IEEE Software*, vol. 33, no. 3, pp. 42–52, 2016.

[322] K.-c. Lee, B. Orten, A. Dasdan, and W. Li, "Estimating conversion rate in display advertising from past erformance data," in *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2012, pp. 768–776.

[323] P. Mockapetris and K. J. Dunlap, "Development of the domain name system," in *Symposium proceedings on Communications architectures and protocols*, 1988, pp. 123–133.

[324] P. Maymounkov and D. Mazieres, "Kademlia: A peer-to-peer information system based on the XOR metric," in *International Workshop on Peer-to-Peer Systems.* Springer, 2002, pp. 53–65.

[325] S. Raval, *Decentralized applications: harnessing Bitcoin's blockchain technology.* " O'Reilly Media, Inc.", 2016.

[326] H. Hietanen, A. Huttunen, and H. Kokkinen, "Criminal friends of entertainment: Analysing results from recent peer-to-peer surveys," *Scripted*, vol. 5, p. 31, 2008.

[327] T. Mirzoev and S. White, "The Role of Client Isolation in Protecting Wi-Fi Users from ARP Spoofing Attacks," *arXiv preprint arXiv:1404.2172*, April 2014.

[328] K.-P. Yee, "Aligning security and usability," *IEEE Security & Privacy*, vol. 2, no. 5, pp. 48–55, 2004.

[329] S. Furnell, A. Jusoh, and D. Katsabas, "The challenges of understanding and using security: A survey of end-users," *Computers & Security*, vol. 25, no. 1, pp. 27–35, 2006. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0167404805002038

[330] D. W. M. Boucadair, R. Penno, " Universal Plug and Play (UPnP) Internet Gateway Device - Port Control Protocol Interworking Function (IGD-PCP IWF)," Internet Requests for Comments, RFC Editor, Rfc 6970, July 2013. [Online]. Available: http://www.rfc-editor.org/rfc/rfc6970.txt

[331] Y. Zhauniarovich and O. Gadyatskaya, "Small changes, big changes: an updated view on the android permission system," in *International Symposium on Research in Attacks, Intrusions, and Defenses.* Springer, 2016, pp. 346–367.

[332] A. P. Felt, E. Ha, S. Egelman, A. Haney, E. Chin, and D. Wagner, "Android permissions: User attention, comprehension, and behavior," in *Proceedings of the eighth symposium on usable privacy and security*, 2012, pp. 1–14.

[333] J. Wang, "Method and system for providing wireless mobile server and peer-to-peer services with dynamic dns update," Sep. 2003, US Patent 6,614,774. [Online]. Available: https://www.google.com/patents/US6614774

[334] B. Wellington, "Secure domain name system (DNS) dynamic update," Network Working Group, Tech. Rep., 2000, accessed 21th November 2019. [Online]. Available: https://rfc-editor.org/rfc/rfc3007.txt

[335] G. Paul, P.-L. Dubouilh, and J. Irvine, "Performance Challenges of Decentralised Services," in *Proceedings of Vehicular Technology Conference (VTC Fall), 2015 IEEE 82nd*, Sep. 2015, pp. 1–4.

[336] A. Norberg and S. Siloti. (2016, Aug.) BEP44: Storing arbitrary data in the DHT. Version b13d4e0f52da7808c46a61f473b10db076d1cabb. [Online]. Available: http://bittorrent.org/beps/bep%5F0044.html

[337] J. Mattsson, "Overview and Analysis of Overhead Caused by TLS," Network Working Group, Tech. Rep., 2014. [Online]. Available: https://tools.ietf.org/id/draft-mattsson-uta-tls-overhead-01.html

[338] Wireshark. [Online]. Available: https://www.wireshark.org/

[339] T. Kothmayr, C. Schmitt, W. Hu, M. Brünig, and G. Carle, "DTLS based security and two-way authentication for the Internet of Things," *Ad Hoc Networks*, vol. 11, no. 8, pp. 2710–2723, 2013.

[340] I. You, S. Kwon, G. Choudhary, V. Sharma, and J. T. Seo, "An enhanced LoRaWAN security protocol for privacy preservation in IoT with a case study on a smart factory-enabled parking system," *Sensors*, vol. 18, no. 6, p. 1888, 2018.

[341] F. Xu, W. Diao, Z. Li, J. Chen, and K. Zhang, "Badbluetooth: Breaking android security mechanisms via malicious bluetooth peripherals." in *NDSS*, 2019.

[342] E. Barker, "Guideline for using cryptographic standards in the federal government: Cryptographic mechanisms," National Institute of Standards and Technology, Tech. Rep., 2020. [Online]. Available: https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-175Br1.pdf