Authentication, Key Exchange, and Attack Detection with SDN Controller for IoT Networks

A Thesis Submitted to Auckland University of Technology in Fulfilment for the Degree of Doctor of Philosophy

By

Gaurav Pathak

School of Engineering, Computer and Mathematical Sciences /
Department of Computer Science and Software Engineering

October 2021

Declaration

I, Gaurav Pathak, hereby declare that this submission is my own work and that, to the best of my knowledge and belief, it does not contain any information published or written by another person (except where explicitly defined in acknowledgements), nor to a substantial extent has been submitted for the award of any other degree or diploma of a university or other institution of higher learning.

Candidate's signature

Abstract

The system of global connectivity of sensing devices for monitoring and automation is known as Internet of Things (IoT). Adoption of this technology has made tremendous contribution to increase the productivity, efficient monitoring, and communication in various application domains. The numerous IoT applications have driven industry to adopt the technologies facilitating the IoT implementation for their specific use cases. The rapid adoption and diverse applications of IoT have presented novel use cases and challenges. To address these challenges, multiple vendors joined the race of IoT service provision. One of the challenges in IoT adoption is low powered long distance transmission that is addressed by Low Powered Wide Area Network (LPWAN) technologies. Another challenge is the limited computing power and storage that makes these devices resource-constrained and unable to utilise state of the art security mechanisms, making them vulnerable to attacks. To fulfil the requirements of minimal processing and long node lifetime, most of the IoT devices provide minimal security features. LPWAN technologies are no exception when it comes to security provision in communication security. LPWAN technologies follow a star topology where the nodes are deployed in remote locations, and they transmit data directly to an Internet facing gateway. The end devices in the networks often use commodity hardware to achieve low cost and do not provide physical security to the devices. In addition, the communication to the gateways is wireless. Given the state of the device and communication medium, they become an easy target for the attackers in the network.

This thesis analyses the security features provided by various LPWAN technologies in detail and explains their security vulnerabilities. Based on the investigation, the requirement for better lightweight session key and attack detection mechanisms for LPWAN networks is identified. The research proposes a lightweight and security mechanism for nodes in LPWAN networks. Additionally, this study explores the applicability of Software Defined Networks (SDN) in the provision of security for LPWAN and embeds the SDN framework in the proposed security mechanism. The proposed security mechanism utilises an SDN controller as a centralised entity for key distribution and attack detection. The framework has four major components to achieve a secure security framework for LPWAN: key distribution for end nodes, node activation and session key mechanism, energy-aware adaptive encryption, and machine learning based attack detection using SDN.

The proposed framework provides a lightweight session key mechanism and a robust SDN-based attack detection mechanism for LPWAN nodes. The framework targets energy-aware operations while providing efficient security to the network by shifting the computational tasks towards the computationally efficient end of the network by leveraging the star topology of LPWAN networks. The framework is validated using various simulation tools to verify its capabilities and operations. The correctness of the key calculation employed for the session key mechanism is verified using the Mininet-WiFi emulator and the session key process includes calculating the session keys on the server

and end node using the public key information. Data flow security is verified for its protection against various attack models using the Scyther security analysis tool. In addition, the energy consumption of the session key mechanism is measured by implementing an energy model on top of the LoRaWAN protocol in the NS3 simulator. The simulation results verify that the data flow of the session key mechanism is not vulnerable to any attack model in the Scyther tool and is functioning correctly. On comparing the power consumption of the proposed session key mechanism with LoRaWAN protocol, it is confirmed that the session key mechanism has minimal impact on the end nodes.

Furthermore, along with the session key mechanism, this research has proposed an attack detection strategy. This approach is employed/tested on the publicly available dataset, AWID-CLS which include samples of three types of wireless networks attacks: “Flooding”, “Injection”, and “Impersonation”. This thesis proposes a two-tier architecture for attack detection and profiling for the security of the network. The first tier uses a binary classifier to classify “attack” and “normal” traffic from entering the Internet facing network. The second tier of attack detection mechanism implements a consolidated voting mechanism to profile the attack on the network. The dataset is divided into train and test sets for training and testing purposes for machine learning classifiers, respectively. The prediction results on the test set show that the trained classifiers have high efficiency in detecting and profiling the attacks on the network. The application of the two-tier architecture provides a distributed approach for early detection of attacks on the gateway and redirects the malicious traffic before it can enter the IoT network.

Acknowledgement

This research study has been successful because of the continuous support of many people I respect and would like to acknowledge for their support in this pursuit.

First and foremost, I would like to thank my supervisor, Professor Jairo Gutierrez. Thank you very much for your patience, support and guidance in my PhD journey. Your feedback and suggestions always encouraged me to explore and refine my work. Thanks to you, I learned so many things that will carry me further in my career and stay with me throughout my life.

Secondly, I would like to thank my secondary supervisor Dr Saeed Ur Rehman, for his detailed and critical feedback on my work. You have always made me look at things that I missed in my research. The technical suggestions provided by you were life savers for me. I appreciate the efforts you put in to help me improve the quality of my research.

Special thanks to my mentor and third supervisor, Dr Akbar Ghobakhlou for his guidance on the machine learning domain of my thesis and supporting me in carrying out the experiments that contributed to my research.

To all my colleagues at NSRG, it has been a great experience working with you all, receiving feedbacks on my early research stages that helped me rethink and refine my research questions. A big thanks to all of you.

In the journey of a PhD, one requires academic feedbacks to improve the quality of research. But equally important is the moral support provided by people around you. I would like to thank my family and friends who provided me with the moral support I needed to get through my time in PhD.

Table of Content

List of Figures

List of Tables

Glossary and Notations

LPWAN – Low Powered Wide Area Network

UART – Universal Asynchronous Receiver/Transmitter

AES – Advance Encryption Standard

RSA – Rivest–Shamir–Adleman Encryption Algorithm

ECC – Elliptic Curve Cryptography

SDN – Software Defined Network

End node – Sensor devices deployed in field for monitoring and data transmission

Gateway node – Networking device connecting end nodes external network (Internet)

ABE – Attribute Based Encryption

MAC – Message Authentication Code

BYKA – Blom-Yang Key Agreement

Master Key – Master key is used to generate private keys for each device in the network.

Private Key – Secret Key for a device that is not shared with any other device

Public Key – Public key for a device shared with other nodes for generation session keys

App – Key – A secret key unique to every application running on a device

E_{OP} – Total energy consumed in session key operations

E_{RX} – Energy consumed in packet reception

E_{MAC} – Energy consumed in MAC calculation

E_{AES} – Energy consumed in AES encryption process

E_{BK} – Energy Consumed in BYKA process

Chapter 1

Introduction

"If you think that the Internet has changed your life, think again. The Internet of things is about to change it all over again." -Brendan O'Brien (Chief Architect & Co-founder, Aria Systems)

With the evolution of wireless communication technologies and mobile computing, numerous novel use cases of network-based applications are evolving. One of the paradigms recently gaining attention is the Internet of Things (IoT). IoT can be described as a network of smart devices at a global scale that provides the facilities to automate the real world through monitoring, data collection, and data analysis [1]. IoT impact on humankind was realised when this technology was in its early stages. The numerous applications of IoT promoted academic as well as industrial research. As the area of IoT is growing, so are the challenges in IoT. The concept of IoT is an amalgamation of communication technologies with commodity hardware. Devices with sensing and transmission capabilities are connected to Internet facing entities, giving these devices global connectivity. IoT has given birth to various business models. Devices like Radio Frequency Identifiers (RFID), sensors, actuators, mobile phones, smart watches, and other intelligent devices are used for data gathering, and the data is used to extract insight. Analysed results from the data are used to make business decisions and personalise the user's experience [2].

The applications of IoT are numerous and require various modes of communications amongst the devices in the networks. IoT covers various types of communications between the entities, i.e., machine-to-machine (M2M), human-to-machine (H2M), machine-to-human (M2H) and human-to-human (H2H). M2M communication refers to the interconnection and data exchange between devices without any human intervention. M2M communications have vast applications in security systems, tracking, remote monitoring, smart grids, because of which IoT is expected to grow exponentially with the number of devices in coming years [3]. Figure 1.1 shows a representation of IoT domain interconnection as explained by Dave Evans in [4]. It can be observed how IoT can play major roles in various domains of our everyday life and bring new business opportunities to benefit both users and industries.

As IoT shows the possibilities of huge business opportunities, it is vital to understand the components of IoT networks and the role these components play in network functioning. Figure 1.2 shows an overview of a typical IoT network and how the components interact to provide services that end users are utilising. Then sensing devices generate the data and transfer it to gateways. The communication to a gateway can be direct or multi-hop, depending on the transmission technology and its range. The gateways are the Internet facing entities that forward data to a cloud platform where data is processed

and made available to end-users with some Application Programming Interfaces (APIs). Finally, the APIs provide services to the applications utilised by end-users on their smartphones, tablets or other devices used by them for IoT applications.

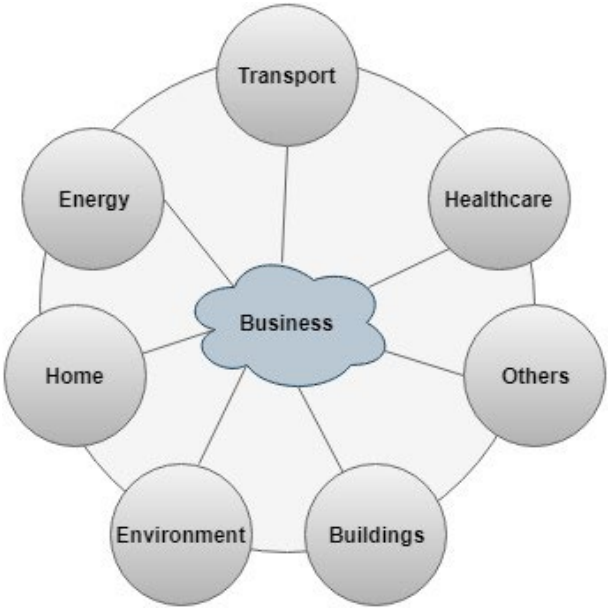

Figure 1.1: Internet of Things as Networks of Networks [3]

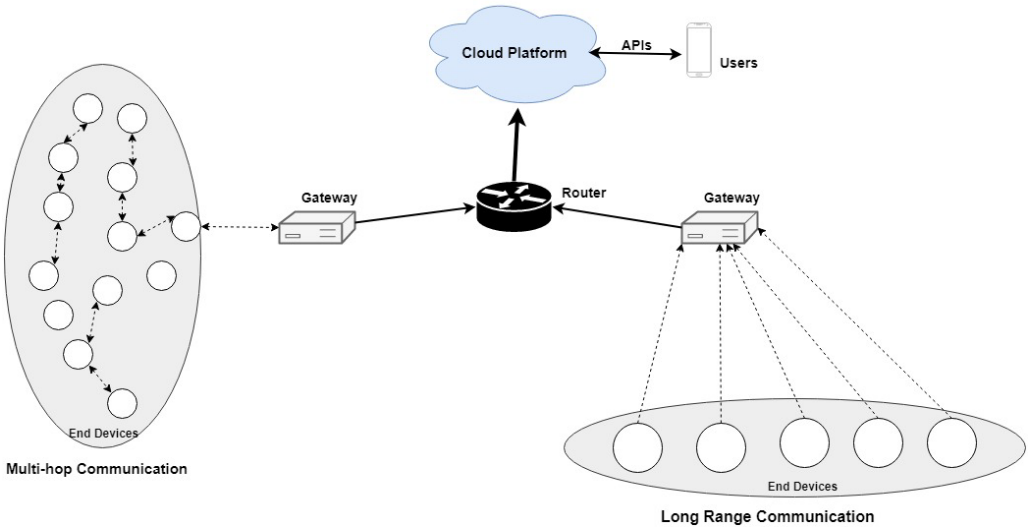

Figure 1.2: IoT Connectivity Overview

Figure 1.3 shows the four layer IoT architecture illustrating a simplified understanding of the IoT system's data flow. The first layer includes sensing devices, and it is responsible for data generation in the network; it is called the perception layer. The sensing devices in the perception layer use various transmission technologies such as 6LoWPAN, ZigBee, WiFi, or LoRaWAN to transmit

data to the gateways in the network as the communication layer. The gateways in IoT networks are Internet facing, and they forward the data from sensors to centralised servers of service providers and can be considered under the network layer. Finally, the application layer is responsible for analytics and final results that are to be used for various applications.

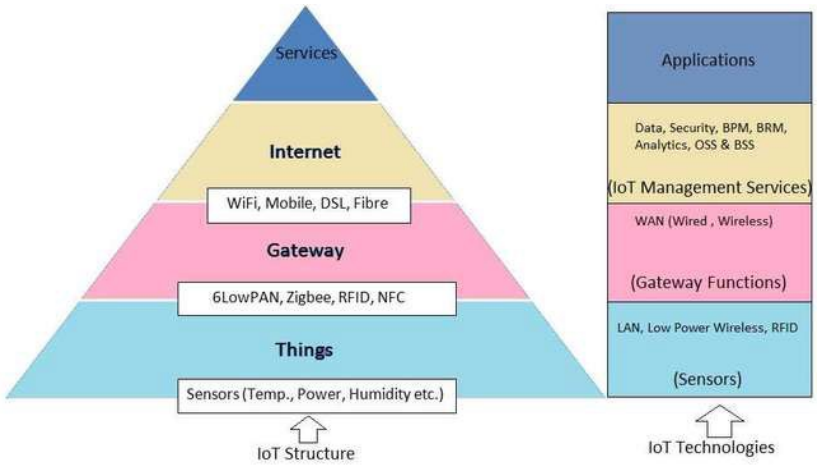

Figure 1.3: IoT layered Architecture [5]

1.1. IoT Applications

The applicability of IoT can be realised in almost everything today. The power of data-based intelligence and task automation are catalytic in the growth of IoT. IoT applications can be industry-oriented and user-facing, where applications are developed for the device to device and device to human interactions [5]. Figure 1.4 shows the domains of IoT applicability.

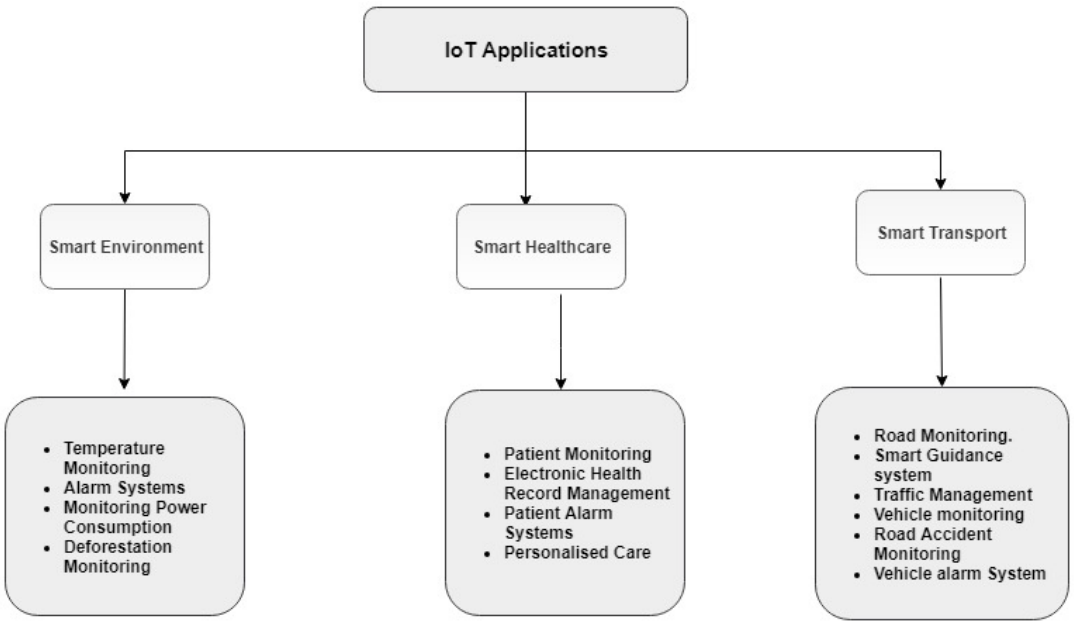

Figure 1.4: IoT Applications

1.1.1. Smart Environment

IoT facilitates the concept of smart surroundings using intelligent devices in homes, offices, industry, and other places involved in our daily lives. Sensors and actuators are distributed and used to control and monitor room temperature, power consumption, area monitoring, and alarm systems.

IoT applications are also being used for monitoring events in nature, like the monitoring of greenhouse gases, planet temperature and monitoring deforestation. Previously these were covered by sensor networks. However, the reach of sensor networks is limited as they are isolated systems disconnected from the Internet. Low-cost manufacturing of sensors and connectivity to the Internet had decreased deployment cost and massively increased the range of monitoring [6], increasing the use of IoT technologies for smart environments.

1.1.2. Smart Healthcare system

The applicability of IoT in healthcare is increasing for improved quality and reduced cost of care. IoT can offer smart personalised healthcare solutions based on an individual's biological, social, and cultural characteristics, resulting in better care of the patients. In recent times, hospitals have started using Electronic Health Record (EHR) systems where patients' medical records are kept on servers that can be accessed from any location. However, the maintenance of EHR is not as easy as it requires training and efforts to manage and update. IoT devices can play a significant role in monitoring the patients and simultaneously updating the health records using wireless transmission. Applications like real-time patient monitoring systems can be implemented and used to trigger an alarm in emergencies. IoT is being used to monitor patients in hospitals and transmit their information to a central server periodically. Kyoto University hospital has implemented real-time workflow monitoring using barcode scanners and Bluetooth transmitters and deploying barcodes on patients, nurses and supplies [7]. The automation of patient monitoring and record management tasks can benefit both hospital staff and patients, making the applicability of IoT in healthcare even more appealing.

1.1.3. Smart Transport

Recent advancements in automobile production increased the use of IoT in transportation systems as well. RFID tags and sensors are used to identify and transfer vehicle information and use it for analytics. IoT is used to track and monitor vehicles, automate toll collection, augmented mapping for directions, and the monitoring of road conditions and accidents. These smart devices are also being used for assisted driving, where the real-time information of traffic in the route can be transferred to the drivers, and better routes can be suggested [8]. The carbon emissions of the vehicles are monitored, and suggestions for repairs or maintenance are provided to drivers using sensing devices. The applicability of IoT in transportation is vast as it can cover a wide area and monitor remote locations without human interventions. IoT can play a crucial role in decreasing the maintenance cost of transportation systems.

1.2. IoT Requirements

The requirements are mostly application dependent, but all IoT applications share some common principles that effectively ease the integration of these remotely located devices and thus the overall IoT deployment gets smoother. In [9], the authors explained various aspects of IoT as shown in Figure 1.5.

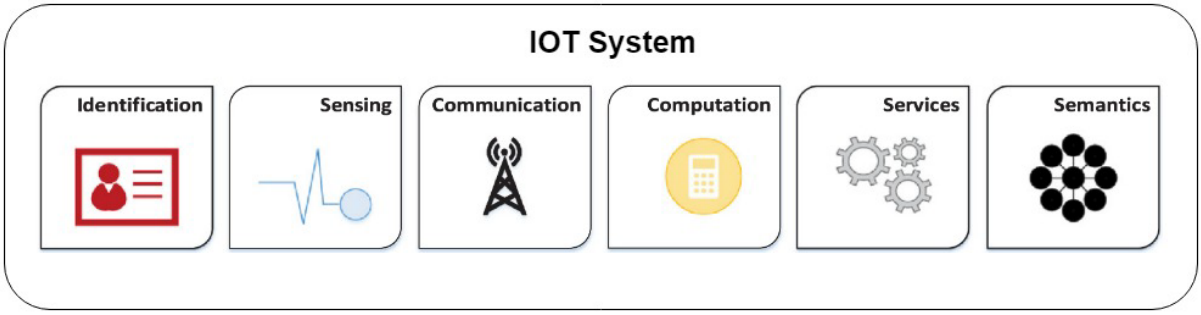

Figure 1.5: Elements of IoT System [10]

Every element in IoT has a functional role that leads to successful IoT implementation. The elements and their functions are discussed as below:

- Identification: Every device in the system must have a unique identity for unambiguous communication with other entities in the network.
- Sensing: Sensing capability is required to generate data related to the environment.
- Communication: the data generated by the sensors are to be transferred to users or other entities using communication technologies.
- Computation: Computation power is required to process the data obtained from objects in the network
- Services: The functionalities provided by the object in the system are the services.
- Semantics: The ability to extract knowledge from the data gathered by the objects to fulfil the application requirements

1.3. Challenges in IoT

IoT based applications are expected to be almost everywhere in the coming future. However, along with a massive range of applications, IoT also produce challenges in seamless service delivery to end-users. An IoT network can have billions of devices connected. These devices can be using different transmission technologies to communicate sensitive information to gateways and servers. Such scenarios brings in three significant challenges in IoT networks: Scalability, Heterogeneity and Security as described below:

Scalability: Considering the number of devices that are being targeted to be connected to the Internet under the Internet of Things, the current network architecture is not ready to accommodate the humongous number of devices yet. With the decrease in manufacturing cost of wireless devices, billions of devices are joining the Internet every year. The high volume of devices will be generating vast amounts of data and transferring data of this magnitude can cause congestion and create data aggregation challenges in the network. The increased traffic in the network will require better traffic management mechanisms [10] for the required QoS provision. Most of the devices in IoT are energy constrained but they can send minimal information periodically, and the routing mechanisms are necessary to identify the applications and handle the packets according to their requirements. Efficient data aggregation mechanisms will be required to control the amount of data uploaded to cloud backend.

Interoperability: The devices used in IoT networks use different transmission technologies and may be working on various applications at the same time. The heterogeneous nature of devices requires flexible interfacing for them to communicate. With most of the communications in IoT being machine to machine, effective interoperability is a must for all devices for seamless data transfer. IoT has numerous technologies working together, which may increase in future. The integration platforms for all these technologies must be flexible and robust to cope up with continuous changes in IoT networks.

Privacy: IoT is a major medium of data collection from users and machines. With such scale of data being transferred and stored on servers for analytics and decision making, it is necessary maintain user privacy. As the volume of data increases, it becomes challenging to differentiate sensitive information that may be of serious concern to the users from other types of data. For applications carrying sensitive user information, such as healthcare applications, user privacy must be a priority. A user's privacy breach can cause ethical and legal issues for the service providers.

Security: Out of all challenges in IoT, security is the most discussed [11, 12]. With such a tremendous number of devices and such a huge volume of data, IoT networks are likely to be attacked with malicious intent. Security consists of several aspects that must be considered while developing applications, they are:

- Confidentiality: To ensure that data is unreadable to unauthorised individuals, entities or processes.
- Integrity: It ensures that the data has not been modified before being delivered to an authorised entity.
- Authentication: It is to verify the legitimacy of the data source.
- Non- Repudiation: To ensure the non-deniability of data being sent by the sender.
- Availability: To ensure that valid users have access to the services at all times.

To fulfil all the attributes of security discussed above, several cryptographic mechanisms are used depending on application requirements. As IoT has a large variety of applications, the requirements

vary from application to applications. To fulfil the application requirements, IoT has a range of transmission technologies coming together for different applications.

1.4. LPWAN Communications for IoT

Low Powered Wide Area Networks (LPWAN) are a set of technologies targeting long-range communications. Contrary to existing long-range technologies such as WiFi and cellular technologies (3G/4G), LPWAN does not target high data rates. Instead, it focuses on larger area coverage, scalable networks, and energy efficient operations for IoT networks [13]. LPWAN technologies achieve 10-40 km coverage in rural and 1-5 km in urban areas. LPWAN devices are also extremely inexpensive, with a chipset cost of less than $2.5 and an operational cost of $1.25 per year [14]. Because of the cost effective and energy efficient characteristics of LPWAN, it perfectly fits the requirements of IoT networks. LPWAN is a comparatively new technology but has gained tremendous attention for the industry. Because of this, there have been numerous communication technologies such as LoRaWAN, SigFox, and NB-IoT [14] has been added under its umbrella. LPWAN technologies are further discussed in Chapter 2 where various LPWAN communication technologies are discussed with their advantages and disadvantages.

1.5. Motivation

IoT end nodes are usually deployed in a hostile environment where it is very challenging to re-energise the node, and they are vulnerable to various physical threats [15]. In [8, 16-19], the authors discussed some of the open issues related to security in IoT, and it is pointed out that IoT is highly vulnerable to multiple attacks. Also, authentication of the nodes, data confidentiality and integrity are considered some of the most critical issues. Nodes require numerous message exchanges with the servers to authenticate themselves and to exchange secret keys. However, sensor nodes (constrained nodes) are limited in their energy sources. An increase in the number of transmissions from end nodes can shorten the node lifetime and break network connectivity.

In [20], the authors performed a detailed analysis of wireless communications in IoT and discussed several open challenges in this area. The authors addressed the impact of security vulnerabilities because of the resource constrained devices on the users. The authors mentioned several use cases and security requirements. It has been pointed out that for IoT to be successful in areas like smart cities, healthcare and smart grids, the security requirements are relatively high. There is a lack of security in current wireless technologies for IoT because of the various factors highlighted. In [21], possible security threats are discussed for every layer of the IoT architecture. The paper pointed out that inadequate physical security is in the top ten IoT vulnerabilities. Security mechanisms of current communication layer technologies in IoT are discussed in detail along with their vulnerabilities to various attacks. The authors highlighted that the security of the communication layer protocols is very crucial as it is most

likely to be attacked. LoRaWAN is one of the most recent protocols for communication layer in IoT. Authors in [19, 22-24] have discussed how this protocol can be penetrated to send false information to a gateway using UART pins in the device. LoRaWAN's join procedure attack is vulnerable to replay attack of join accept messages which can lead to the creation of fake gateways in the network. Also, in the encryption process of end devices, the transceiver is used for encryption of the data, and the microcontroller has no knowledge of the keys used in the process. This makes it possible to send fake data from the devices using the UART pins of the transceivers.

In a large scale IoT network, it is imperative to identify and authenticate the end devices generating the data. Any unauthenticated or impersonated node can transmit false data. Public key algorithms for authentication are widely used when it comes to networks having nodes with sufficient processing capability. However, it does not apply to constrained nodes. They neither have memory nor have the processing capabilities for sophisticated operations that are required in public-key cryptography algorithms like RSA or ECC. LPWAN technologies have their own mechanism for preserving the authenticity, integrity and confidentiality of data. However, most technologies do not use session key mechanisms for data encryption; as per our preliminary research, LoRaWAN is the only transmission technology that uses session key mechanisms. However, the session key generation is not very dynamic. Furthermore, LoRaWAN uses two mechanisms for node authentication, i.e. Activation by Personalisation (ABP) and Over the Air (OTA) activation. The OTA activation mechanism provides a session key mechanism, and the ABP mechanism uses the same key for the node lifetime.

Security flaws in current transmission technologies can lead to leakage of data or, at worst, compromise the entire transmission. In some IoT applications, such as healthcare monitoring, the compromise of confidentiality and integrity could have severe implications leading to the patient's death. The security vulnerabilities of communication technologies must be attended for secure IoT service provision.

1.6. Research Questions

The thesis aims to develop a lightweight authentication and session key mechanism with an identity theft detection mechanism for compromised nodes by taking advantage of the flexibility of the Software Defined Networking framework (SDN).

RQ1. How can we design an SDN based lightweight authentication and session key exchange framework for IoT nodes?

RQ2. How can SDN controllers be used to detect node identity theft in the IoT network?

RQ3. What kind of performance trade-offs the IoT nodes may face with the addition of the new security scheme?

1.7. Research Significance

The delivery of strong security for constrained node IoT networks is a challenging proposition considering the limitations of the end nodes in the network. As discussed in earlier sections, most IoT communication protocols use static secret keys instead of session keys, which increases the odds of the deduction of secret keys used for data encryption.

This research proposes a lightweight authentication and session key exchange mechanism. The frequent changes in session keys would make the deduction of the keys difficult. However, session key exchange can cause additional transmissions for the end nodes, resulting in a faster energy drain. An SDN framework will be used as resource support for constrained IoT nodes to minimise the processing load. Most of the energy consumed by IoT nodes is due to transmissions; thus, minimising processing and transmissions at the end node would make authentication and the session key exchange mechanisms energy efficient. Also, the detection of identity theft in the network in real-time will also be addressed. Furthermore, in the case of a node being compromised in the network, the study proposes a software fingerprint-based node identification mechanism to detect identity theft, spoofing and impersonation attacks in the network. The data gathered by the SDN controller would be used to identify the impersonation attack in the network.

The major contributions of this research are summarised below where the first and the second contribution of the thesis relates to the RQ1. The third and the fifth contribution of the thesis are addressing the RQ2 and the fourth contribution relates to RQ3 discussed in section 1.6:

i. Design and implementation of a lightweight authentication mechanism for IoT nodes in LPWAN networks.
ii. Implementation of a session key mechanism is proposed for enhanced data security.
iii. Implementation of identity theft detection to identify network breach and node compromise.
iv. Employment of session management according to node energy level incorporating energy efficient security features.
v. Implementation of a two-tier machine learning based attack detection mechanism to filter attack data from the IoT network traffic.

1.8. Thesis Outline

The primary objective of this research is to introduce a lightweight session key and identity theft detection mechanism for LPWAN based IoT networks. To realise the outcomes of the research, a lightweight security framework is designed by combining multiple modules to perform specific security tasks towards a common goal of a secure LPWAN IoT network. Simulations are performed for each

module to investigate the outcomes. The research follows a constructive approach beginning from problem formulations, framework design, evaluation and validation of the research outcomes.

The architecture proposed in providing solutions is built by utilizing multiple platforms and combined by implementing multiple modules. The first module targets the session key mechanism in the framework. Simulation and emulation are used to implement the first module to realise the outcome. The simulators and emulators provide a platform to mimic large network systems and test the proposed approach that can be challenging to achieve in the real world. Various mathematical models for the proposed frameworks can be implemented and tested quickly in a simulator, facilitating faster testing of the framework outcomes. On the other hand, the second module requires the implementation of machine learning models for the detection of attacks on the networks. It uses python programming which provides rich libraries for implementations of machine learning models. The machine learning models utilise a verified publicly available data containing samples of network traffic for different classes for training and testing. Multiple simulation tools and platforms are utilised to study the output of the proposed framework. As each simulation tool and platform studies different aspects of the performance of the proposed framework. Hence, all the aspects of proposed framework cannot be tested using a single simulator or platform currently available for network simulations.

Chapter 2 explores the communication technologies currently utilised by IoT applications while categorising them based on their transmission ranges. Along with the study of technologies, their security mechanisms and security vulnerabilities are discussed in detail. Furthermore, various possible threats on each layer are discussed, along with their degree of possibility. Considering the security threats on IoT networks, the applicability of SDN and machine learning techniques are discussed for IoT networks.

Chapter 3 analyses the existing literature for security in IoT like constrained networks. The literature is classified based on the aspects of IoT security addressed by the different papers and sources.

Chapter 4 introduces the research methodology incorporated by this research. Various activities that are performed to realise the output of the study are discussed in detail. This research follows the Design Science Research methodology that aims to achieve a security framework for LPWAN based IoT networks that provides session key mechanisms and identity theft detection in the network. The session key exchange module uses public key information to generate session keys for secure communications between the centralised servers. Additionally, the attack detection mechanism uses machine learning models trained on a dataset containing network traffic samples. An SDN controller is used as a central sever acting as the key exchange authority and as an attack detector in the network.

Chapter 5 explains the experimentation procedures carried out during the study. The chapter explains all the tools utilised to validate the outcomes of the frameworks based on different aspects. Modules of

the frameworks are implemented according to features available in the simulators. The results of the experiments are analysed and discussed in detail.

Finally, Chapter 6 provides a conclusion by revisiting the research questions. Also, the research limitations are explained along with a discussion about the possible future studies that can extend this research. The structure of the thesis is shown in Figure 1.6 below.

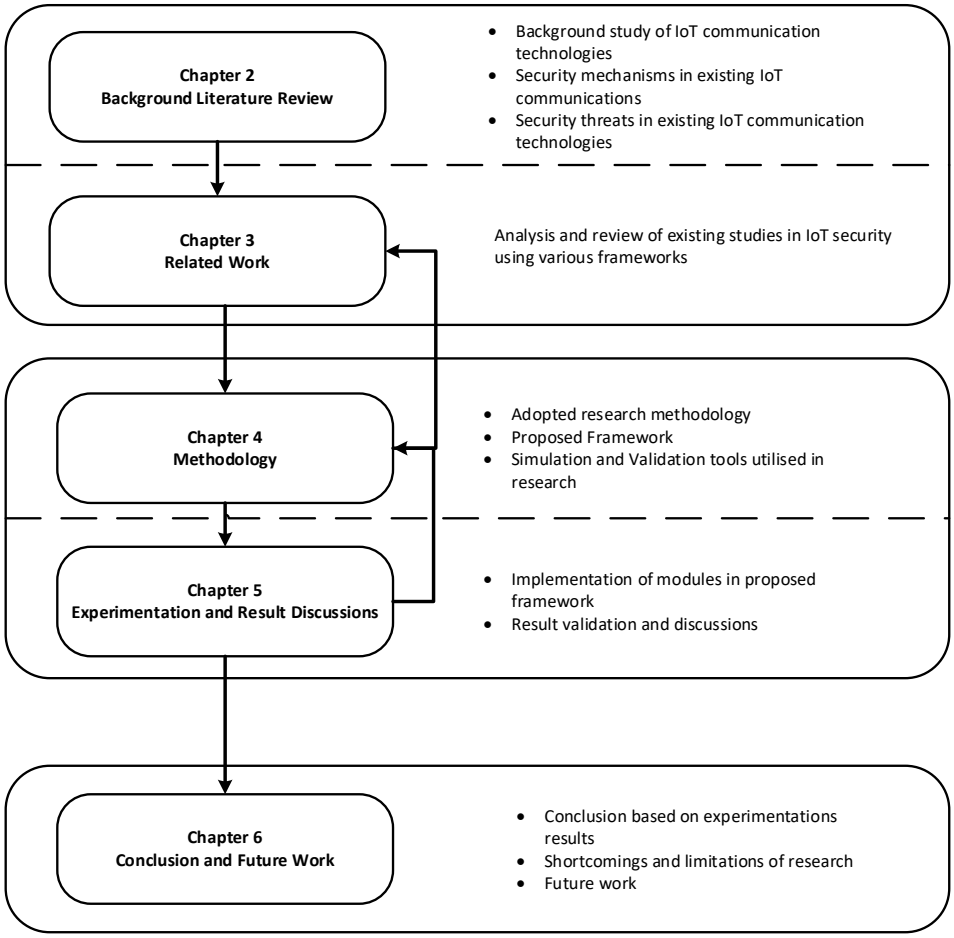

Figure 1.6: Thesis Outline

1.9. Research Publications

Pathak, G., Gutierrez, J., & Rehman, S. U. (2020). Security in low powered wide area networks: Opportunities for software defined network-supported solutions. *Electronics*, *9*(8), 1195.

Pathak, G., Gutierrez, J., & Rehman, S. U. LPWAN Key Exchange: A Centralised Lightweight Approach Sensors Journal. ***[Under Consideration].***

Chapter 2

Background Literature Review

2.1. IoT communication Technologies

As IoT covers a range of applications, various communication technologies are used with this type of networks depending on the requirements. They can be loosely categorised based on their transmission ranges, i.e. short and long-range communication technologies. Depending on the application requirements like power efficiency, data rate, security, reliability and flexibility, one can choose a suitable communication technology for the application at hand. Figure 2.1 shows the communication technologies with their data rates and range, as explained in [20].

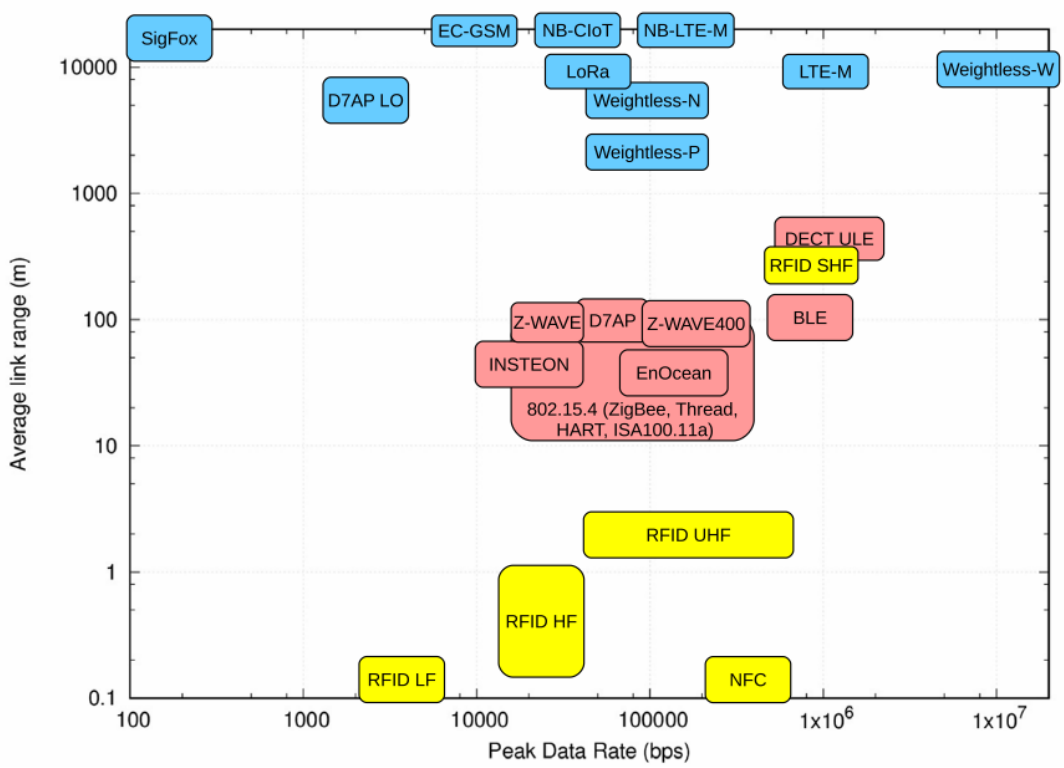

Figure 2.1: Communication Technologies based on Data rate and Range [17]

2.1.1. Short Range Communications

Technologies that fall under short-range can cover from several feet to several hundred meters. There are primarily technologies with comparatively low power consumption, but there are exceptions as well. Some of the short-range communication technologies are discussed as follows:

2.1.1.1. *Radio Frequency Identification (RFID)*

RFID [25, 26] is the first technology that communicated information from "things" to the Internet. RFID enables the identification of objects from a distance even if the object is not in the line of sight [26]. RFID uses the change in the magnetic field to identify the information being sent from the RFID source. Figure 2.2 shows an RFID tag used in devices.

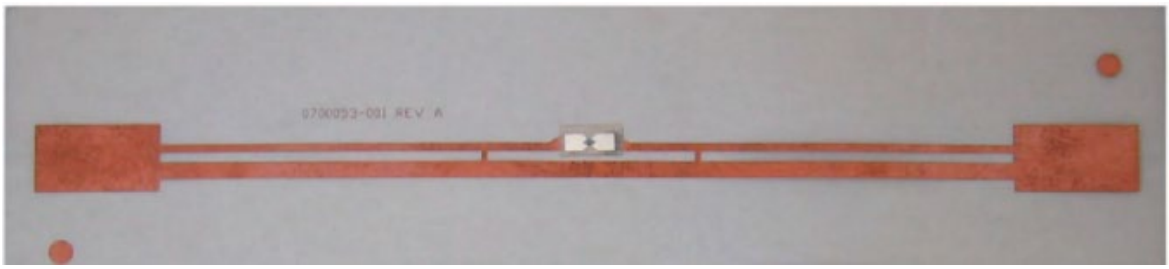

Figure 2.2: Radio Frequency Identification tag [22]

2.1.1.2. *ZigBee*

ZigBee [27] is said to be one of the most widely used technologies. It was released in 2005 by the ZigBee Alliance. It follows a layered architecture where the lower layers: Physical and Medium Access Control (MAC) sub-layers, are defined by the IEEE 802.15.4 standards, and the ZigBee alliance built the network and application layers [28]. Figure 2.3 shows the ZigBee Architecture developed by the ZigBee Alliance.

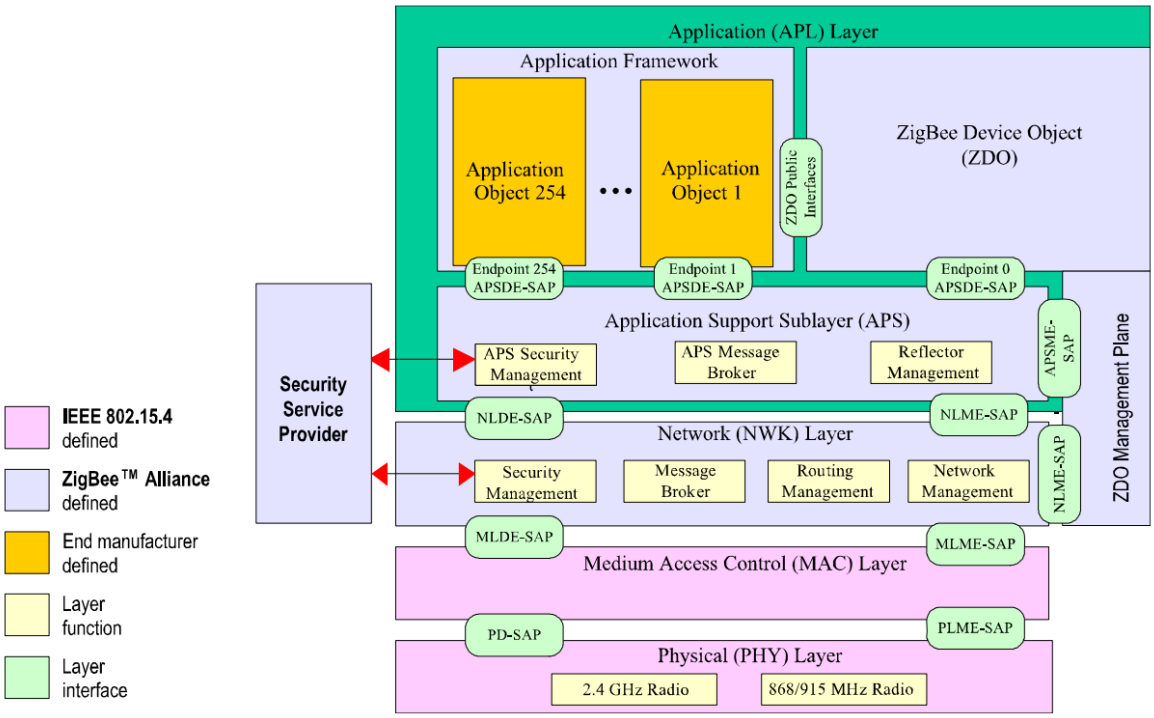

Figure 2.3: ZigBee Layered Architecture [24]

ZigBee supports star, tree and mesh topologies and has three categories of nodes in the network: End node, Router node and Coordinator node [29].

- An End Node is a standard radio frequency device located at the edge of the network.
- A Router Node is used as a relay point and has the capability of maintaining routing tables and of forwarding packets. It is not required in the star topology, but it is used in the mesh and tree topologies.
- A Coordinator Node is the root node that receives all the data either through a router or directly from end nodes. The Coordinator is also responsible for assigning a 16-bit address to a node when it joins the network.

Figure 2.4 shows the arrangement of nodes in a ZigBee network for various topologies

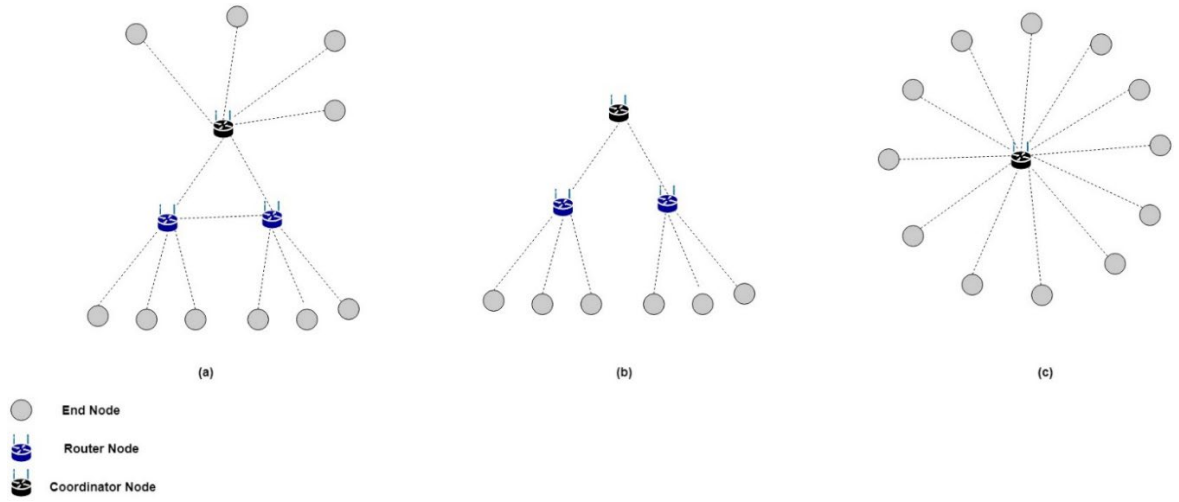

Figure 2.4: ZigBee supported topologies (a) Mesh Topology; (b) Tree Topology; (c) Star Topology

2.1.1.3. WirelessHART

WirelessHART was introduced by the HART communication foundation in 2007. Just like ZigBee, it also adopts IEEE 802.15.4 at the physical layer, but it defines its own time synchronised MAC layer [30]. The WirelessHART MAC layer is specially designed for industrial environments with strict time constraints and security features [20]. WirelessHART has the following components in the network [31]:

- Field Device: It is the instrument deployed at the edge of the network, and it is integrated with the WirelessHART communication module.
- Handheld: It is a device that is used to configure and diagnose the network.
- Gateway: It acts as the bridge between field devices and an external industrial network allowing field devices to communicate to external networks and vice versa.
- Network Manager: It is an application used to manage the network and the devices.

- Security manager: It is responsible for generating and storing the session keys in the network. It is also responsible for the secure network joining of field devices.

2.1.1.4. ISA 100.11a

Developed by the US-based non-profit organisation International Society of Automation (ISA) [32]. It is also a protocol that targets the industrial environment explicitly, just like WirelessHART. ISA 100.11a also uses IEEE 802.15.4 as the physical layer. Its datalink layer implements graph routing, frequency hopping and time-slotted time domain multiple access. Every device in the network has a specific task, and the device can be either input /output devices or forwarding devices [1]. There are significant differences between WirelessHART and ISA100.11a. WirelessHART was designed to address end-user concerns like security, reliability, and delay; ISA100.11a targets flexibility by providing multiple build options to the manufacturer.

2.1.1.5. Z-Wave

It is a wireless protocol developed by Zenesys and extended by the Z Wave alliance [33]. Unlike WirelessHART and ISA 100.11a that target industrial environments, Z-Wave targets specifically home automation. Each device is distinguished by a home id or a network id and a node id for the unambiguous identification of nodes in a common neighbourhood. A smart home that uses the Z-Wave protocol can have 232 appliances that are divided in two categories: controllers and slaves.

Slaves receive information from controllers and act upon them, and they also forward controller instructions to other nodes. The controller has all the routing information about the network. The controller also updates routing tables and strikes out “bad” routes from routing tables. There are two types of controllers: portable and static controllers. Portable controllers have the tendency to change the location and control the network remotely. The static controller must not change its location and have to be powered all the time. There are categories of slave nodes as well: Slave and routing slaves. Routing slaves are the same as slave nodes. In addition, they have the capability to send unsolicited messages to limited nodes and can store several static routes. Figure 2.5 shows the network structure of Z-Wave.

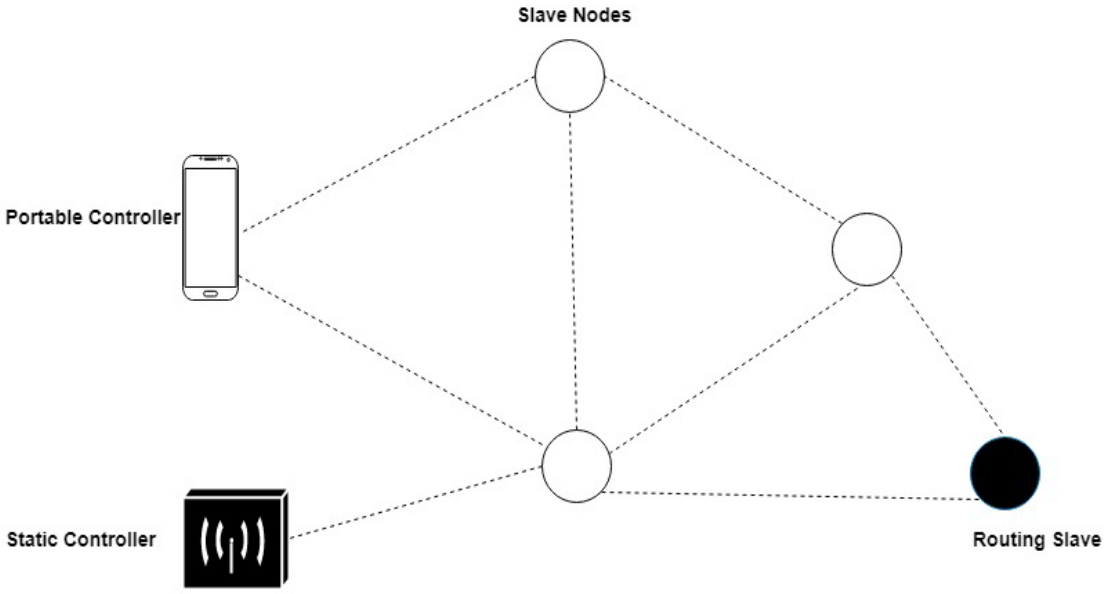

Figure 2.5: Z-Wave Network structure [33]

2.1.1.6. DASH7 Alliance Protocol

It is a wireless protocol designed for low power applications such as sensors and active RFID networks and operates on the sub-GHz ISM band [34]. DASH7 has three device classes: Endpoint Class, Sub controller Class and Gateway Class. Table 2.1 shows the device class features in DASH7.

Table 2.1: DASH7 Alliance Protocol Device class Features [29]

Device Class	**Transmits**	**Receives**	**Wake on scan cycle**	**Always on Reciever**
Endpoint	Yes	Yes	Yes	NA
Sub Controller	Yes	Yes	Yes	NA
Gateway	Yes	Yes	NA	Yes

DASH7 defines all the layers of the OSI stack, including the application and presentation layers. Figure 2.6 shows the DASH7 protocol layer overview.

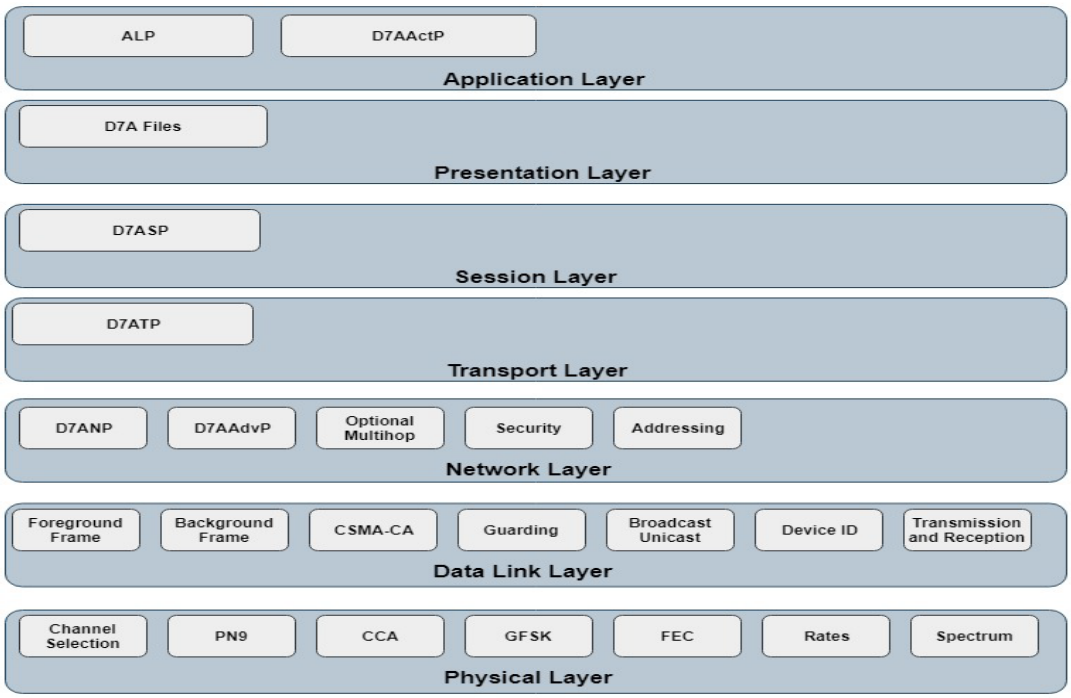

Figure 2.6: DASH7 Alliance Protocol layer overview [31]

2.1.2. Long Range Communications

Short-range technologies have advantages with low power consumption. However, their limited transmission range can be a barrier for many IoT applications. Cellular networks have indeed provided long-range coverage. However, it was not designed for machine to machine communication and to provide machine to machine services for a massive number of devices [35]. To provide a long-range and power-efficient solution to IoT applications, several commercial communication technologies compete against each other that are called LPWAN. Most of the LPWAN technologies follow star network topologies where end devices are connected directly to a gateway. Various LPWAN communication technologies are discussed in detail in the following sub-section.

2.1.2.1. SigFox

Sigfox is a French network operator founded in 2009; their network targets wireless connectivity of low powered devices. SigFox uses a proprietary ultra-narrow band (UNB) with limited uplink connection [36]. SigFox operated on publically available 192 KHz and can achieve data rates of 100 or 600 bps with a maximum payload of 12 bytes and up to a range of 10 km. SigFox is designed for small messages ranging from 0 to 12 bytes keeping the cost and autonomy of remote devices. Table 2.2 shows the sizes of some typical SigFox messages.

Table 2.2: SigFox Payload Size [34]

Payload	Size
GPS coordinates	6 bytes
Temperature	2 bytes
Speed reporting	1 byte
Object status	1 byte
Keep alive payload	0 byte

SigFox has multiple modules in its network, with task divided for each module. However, it can majorly be divided into two layers: Network Equipment and SigFox Support System.

- Network Equipment essentially covers end devices responsible for generating data of the object and base stations that receive data from objects in the network and forward it to the SigFox Support System.
- SigFox Support System is the backend of SigFox and covers the core functionalities where the messages are processed and forwarded to customer systems. APIs are provided as entry points to interact with this layer. This layer is also responsible for network monitoring, maintenance, billing and radio planning for network deployment. In addition, it provides tools for data analytics on collected data.

Figure 2.7 shows the overview of the network architecture of SigFox with functionalities provided by the two layers mentioned above.

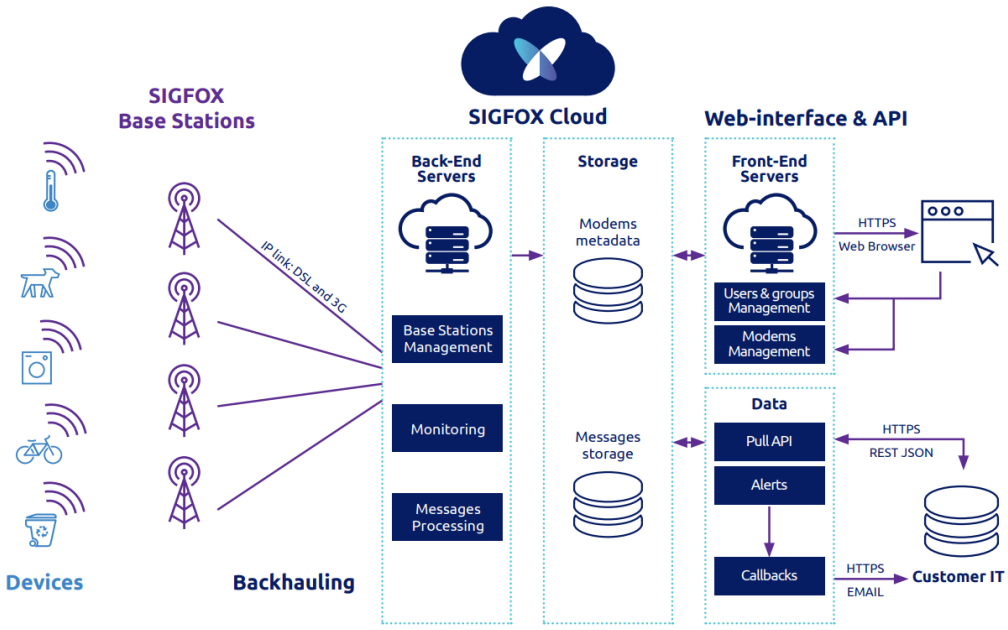

Figure 2.7: SigFox Network Architecture [34]

2.1.2.2. Weightless

Weightless is backed by the UK Company Neul, recently acquired by Huawei. The technology is described in a set of three standards which were developed by a non-profit standard organisation, "Weightless SIG" [37]. It has three standards: weightless N, P and W. Weightless-N [35] supports a star topology. It uses Ultra-Narrow Band (UNB) on a sub-GHz spectrum, uses DBPSK digital modulation with frequency hopping and has a promising range of several kilometres. Weightless-P uses a narrow band with TDMA and FDMA and can be considered an extension of Weightless-N for bidirectional communications. It uses TDMA and FDMA, and it operates on a 12.5 kHz narrow band [1]. In addition, it uses energy-efficient modulations with an adaptive data rate. Lastly, Weightless W operates in the television whitespace spectrum, and it is designed for comparatively higher data rates; the modulation techniques supported are DBPSK and 16-QAM.

2.1.2.3. NB-IoT

Narrow Band Internet of Things (NB-IoT) operates on low bandwidth. It supports a large number of devices with low data rates. It can efficiently use small parts of the re-framed spectrum. It uses repetition of transmitted data over different channels to increase the coverage. NB-IoT can provide coverage of up to 40 Km in rural areas [38]. NB-IoT can be deployed as standalone or along with the existing LTE spectrum and requires a minimum bandwidth of 180KHz [39].

2.1.2.4. LoRa-WAN

Developed by Semtech, LoRa is a wireless modulation for long-range, low-power and low-data-rate applications [40]. It follows a star topology, where nodes transmit data to their gateway directly, and the gateway forwards the data to the Internet facing components of the network. It follows the SS Chirp modulation, and it is capable of transmitting data up to 5 km in urban settings and up to 45 km in rural areas with data rates up to 50Kbps, and the life of a LoRa device can go up to 10 years. LoRa-WAN has three classes: Class A, Class B and Class C. Class A (Bi-Directional End Device): Class A devices are the lowest power-consuming devices. They support bi-directional communications where two short downlink windows follow the uplink transmission. Any communication from the server can be done in the two short downlink windows; else, it has to wait till the subsequent scheduled downlink.

- Class B (Bi-directional end-devices with scheduled receive slots): In comparison to Class A, Class B has more receive slots. It provides a scheduled receive window in addition to class A's two short downlink windows. The gateway sends a beacon to the end node to open the scheduled window for communication.

- Class C (Bi-directional end-devices with maximal receive slots): Class C devices have continuous receive windows, and it only closes when the device is transmitting. Class C is the

most power consuming device class in LoRa-WAN. It also has lower latency than Classes A and B.

Figure 2.8 shows the overview of the LoRa-WAN architecture. The network server manages the network services like routing and node activation. It receives data from gateways and forwards it to application servers from where the respective applications use the data.

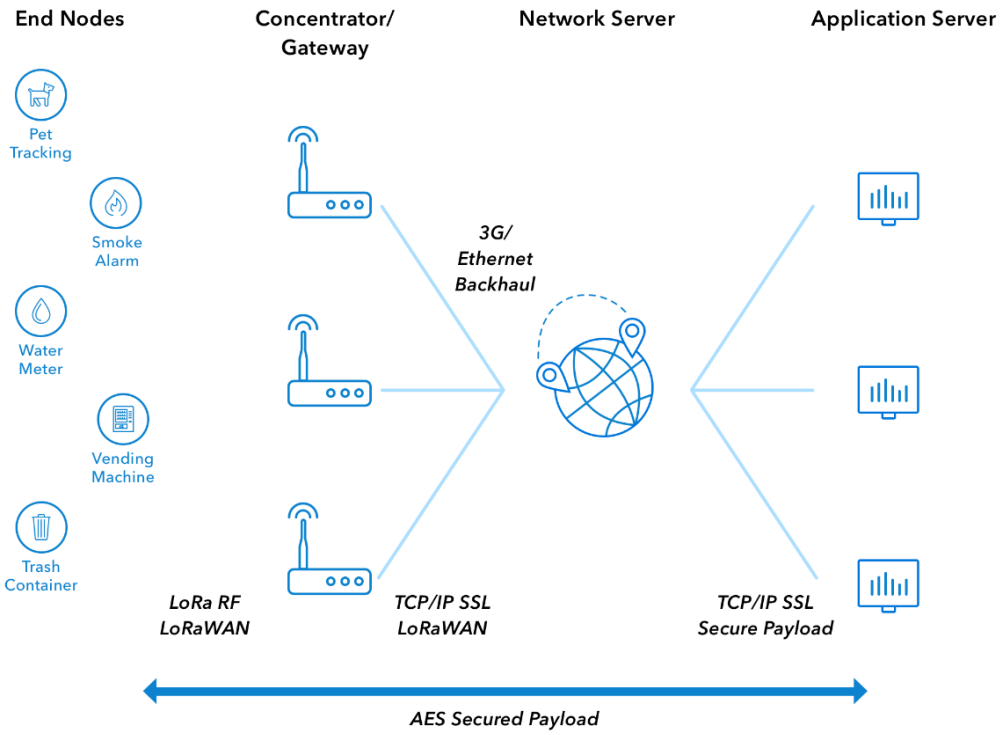

Figure 2.8: LoRaWAN Architecture [39]

2.2. Security Mechanisms in IoT Communication Technologies

2.2.1. Security in Short Range Communication Technologies

Short range technologies play a vital role in communication between devices in IoT network. This sub-section explains the security mechanisms of various short range communication technologies in IoT

2.2.1.1. RFID

RFID is the most famous technology for the automatic identification of objects. As discussed earlier, the identification tags transmit magnetic information to the receiver for identification. RFID does not provide any fundamental security feature for authentication, integrity, confidentiality, and availability unless additional security mechanisms are added to the system [41].

Confidentiality: In most cases, the communication between the RFID tag and a reader is not encrypted except in some ISO 14443 systems [42]. There is a high possibility of data leak if an attacker eavesdrops on the devices.

Integrity: In the case of protection against data integrity, few high-end systems (ISO 14443) provide message authentication codes (MAC). However, most systems employ checksums (CRC) that not very effective and works only against random failures.

Authentication: The RFID tags are not tamper-proof and can be manipulated. The unique identifier of the tags can be changed. Hence, destroying the authenticity of the devices.

Availability: RFID devices are highly prone to jamming attacks. Blocker tags [42] can be used to disrupt the communication between the reader and the tags.

In [43], various RFID standards are discussed along with their security features, as shown in Table 2.3

Table 2.3: RFID Technologies and features

Technology/ Standard	**Application**	**Technical Features**		**Standard Security Features**	
		Band	**Range(meters)**	**Confidentiality**	**Integrity**
EPC Class 0	Supply Chain	Ultra High Frequency (UHF)	3	None	• Parity bit • CRC
EPC Class 1 Generation 1	Supply Chain	UHF	3	None	• Parity bit • CRC
EPC Class 1 Generation 2	Supply Chain	UHF	3	One-time pad stream cipher	CRC
ISO/IEC 18000-2	Item management	Low Frequency (LF)	<0.01	• No encryption • No read protection • No authentication	• CRC • Permanent 64-bit ID • Lockable identifier code (optional)
ISO/IEC 18000-3	Item management	High Frequency (HF)	<2	• 48-bit password protection on reading operations • Reader talks first protocol	• CRC • Password protection on write command
ISO/IEC 11784/11785	Animal Tracking	LF	<0.01	• Reader talks first protocol	• Retagging Counter

Technology/ Standard	Application	Technical Features		Standard Security Features	
		Band	Range(meters)	Confidentiality	Integrity
				• Tags addresses with random numbers	• CRC
ISO/IEC 10536	Contactless smart cards	HF	<2	• Masked reader to tag communication. • Tags addresses with random numbers	CRC
ISO/IEC 15693	Vicinity smart cards	HF	<1.5	No read protection and no encryption	• Error checking on the wireless interface • Write protection (optional)

2.2.1.2. ZigBee:

ZigBee is efficient and one of the most used short-range transmission technologies for wireless sensor and IoT networks. ZigBee employs AES-124 CCM for data confidentiality, authenticity and integrity in the network. The data is encrypted and signed with a key shared amongst the devices in the network. For integrity, a MIC code of the data is calculated and transferred with the data. The receiver calculates the MIC of the data received and compares the calculated MIC with the received MIC for the integrity of the data. Table 2.4 shows the security modes provided in ZigBee.

Table 2.4: ZigBee Security Modes [44]

Security Level ID	Security level Sub field	Security Suit	Security Attributes	Data Encryption	Frame Integrity (Length of MIC)
0x00	000	None	None	OFF	NO (M=0)
0x01	001	AES-CBC-MAC-32	MIC-32	OFF	YES (M=4)
0x02	010	AES-CBC-MAC-64	MIC-64	OFF	YES (M=8)
0x03	011	AES-CBC-MAC-128	MIC-128	OFF	YES (M=16)
0x04	100	AES-CTR	ENC	ON	NO (M=0)
0x05	101	AES-CCM-32	ENC-MIC-32	ON	YES (M=4)
0x06	110	AES-CCM-64	ENC-MIC-64	ON	YES (M=8)
0x07	111	AES-CCM-128	ENC-MIC-128	ON	YES (M=16)

2.2.1.3. WirelessHART

WirelessHART focuses on industrial communications. Hence, it needs to be reliable and secure. All communications in WirelessHART are encrypted with AES-128. Similar to ZigBee, a shared key is used. The protocol also uses a four-bit counter to create a random nonce. MIC is used for data integrity in the network. The protocol uses the same key for both encryption and MIC calculation.

2.2.1.4. ISA 100.11a

ISA 100.11a [32, 45] provides similar security as WirelessHART with some additional features. It uses timestamp as protection against replay attacks on the network. And, for data integrity in the network, it uses MIC like WirelessHART. The transport layer uses a nonce to indicate the time of packet generation. The receiver checks for the valid nonce within a timeframe to check the validity of the packets. Also, ISA 100.11a provides a joining procedure for nodes where they can use public-key encryption to be authenticated and join the network for the first time. The joining procedure enables the nodes to be authenticated without sharing any information over the air hence, securing the joining procedure against multiple attacks.

2.2.1.5. Z-Wave

Z-wave is another well-known home automation communication technology used for short-range communications. The earlier Z-wave version also provides a comparatively light triple DES with a 56-bit key [46]. Z-wave has recently announced an S2 security framework that uses 128-bit AES. It divides the network into three security classes [47]: S2 access control, S2 authenticated, and S2 unauthenticated. Every class has a distinct secret key, and every device joining the PAN must belong to one of these classes. Depending on the class the device belongs they have a corresponding secret key for communication. Out of the three classes in Z-Wave, S2 access control is the most secure and S2 unauthenticated is the least secure class in the network. In order to obtain keys from the network controller, the devices use the Elliptic Curve Deffie-Helman (ECDH) for secure key exchange.

2.2.1.6. Dash7

Dash 7 uses AES symmetric key cryptography for confidentiality and node authentications in the network. The secret key is stored in the node file system prior to deployment. DASH7 provides multiple options for providing various levels of security, as shown in Table 2.5.

Table 2.5: Security classes in DASH7 [34]

Security class	Encryption/ Authentication mechanism	Description
0	NONE	No Security
1	AES-CTR	Encryption only, counter mode
2	AES-CBC-MAC-128	No encryption, Authentication, Cipher-block chaining with 128 bit MAC
3	AES-CBC-MAC-64	No encryption, Authentication, Cipher-block chaining with 64 bit MAC
4	AES-CBC-MAC-32	No encryption, Authentication, Cipher-block chaining with 32 bit MAC
5	AES-CCM-128	Authentication with CBC-MAC-128 and Encryption with Counter Mode
6	AES-CCM-64	Authentication with CBC-MAC-64 and Encryption with Counter Mode
7	AES-CCM-32	Authentication with CBC-MAC-32 and Encryption with Counter Mode

2.2.2. Security in Long Range Communication Technologies

2.2.2.1. SigFox

SigFox is the most famous long-range communications technology used for IoT devices. As it operates on power constrained devices, it focuses on energy aware activities on the nodes. Lightweight security mechanisms are used for authentication, confidentiality and integrity. SigFox uses a stored symmetric key to authenticate the node and uses sequence numbers to avoid replay attacks. These sequence number counters are auto-incremented with every message and reset after one month with 140 messages/day. The integrity of the sequence number is confirmed by using a Message Authentication Code (MAC), which is sent with the data packet. SigFox does not provide message encryption by default. However, depending on the application, customers can either use their end to end encryption mechanism or use the end to end encryption solution provided by SigFox. Figure 2.9 shows the checks performed in SigFox during message transmission. In addition to node and data security, SigFox also has a built-in Firewall, so the nodes are not directly connected to the Internet.

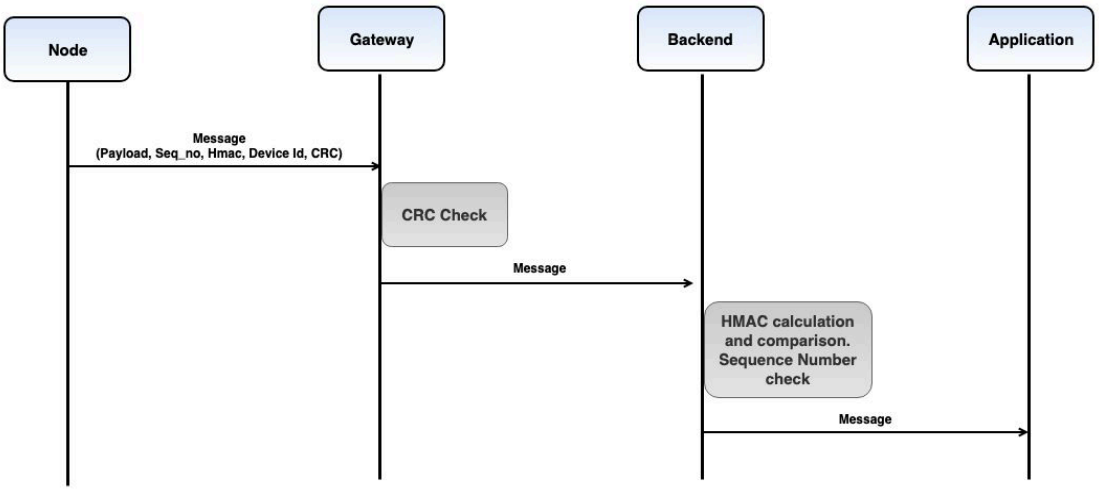

Figure 2.9: SigFox Procedure during Message Transmission [36]

2.2.2.2. LoRaWAN

LoRaWAN is another protocol with long-range transmission capabilities. It operates on LoRa [48] and aims to provide support for mobility and secure communication. LoRaWAN provides both authentication and data security. Symmetric key operations are used for node authentication and data confidentiality. In LoRaWAN, when the node tries for the first time to join the network, a join procedure is initiated. There are two types of activation procedures in LoRaWAN:

i. *Over-The-AIR-Activation (OTAA):* In this method, every node uses its 128-bit Appkey (given to the node at deployment time). The Appkey is used to calculate a four-byte Message Integrity Check(MIC) (it is the first 4 bytes of the MAC calculated on join request message) Code to sign the join request [49]. Figure 2.10 shows the procedure for OTA activation in LoRaWAN.

MAC = aes128_cmac (AppKey, MHDR | AppEUI | DevEUI | DevNonce)

MIC = MAC[0..3]

AppEUI is unique to the owner, and DevUI is unique to the device; they act as Identifiers for the application and the end device, respectively. DevNonce is a random number sent by the device to avoid the replay of the packet. The sender node does not encrypt the join request. Upon receiving the join request, the network server checks the MIC for message integrity, and it checks the DevNonce if it has already been used previously. After that, the server responds with a join accept, which is encrypted with AppKey by using the decrypting module of AES, which can be decrypted using the encrypt module of AES. Using only a decrypt module in the device makes it possible to load only one module in the end node. The MIC for join accept is generated with AppKey [49].

MAC = aes128_cmac(AppKey, MHDR | AppNonce | NetID | DevAddr | RFU | RxDelay | CFList)

MIC = MAC[0..3]

AppNonce is a random number generated by the server to produce AppSKey and NwkSKey, i.e. Application Session Key and Network Session Key, respectively [49].

NwkSKey = aes128_encrypt(AppKey, 0x01 | AppNonce | NetID | DevNonce | pad16)

AppSKey = aes128_encrypt(AppKey, 0x02 | AppNonce | NetID | DevNonce | pad16)

When the end node gets the join request from the server, it calculates the session keys with the above-explained procedure.

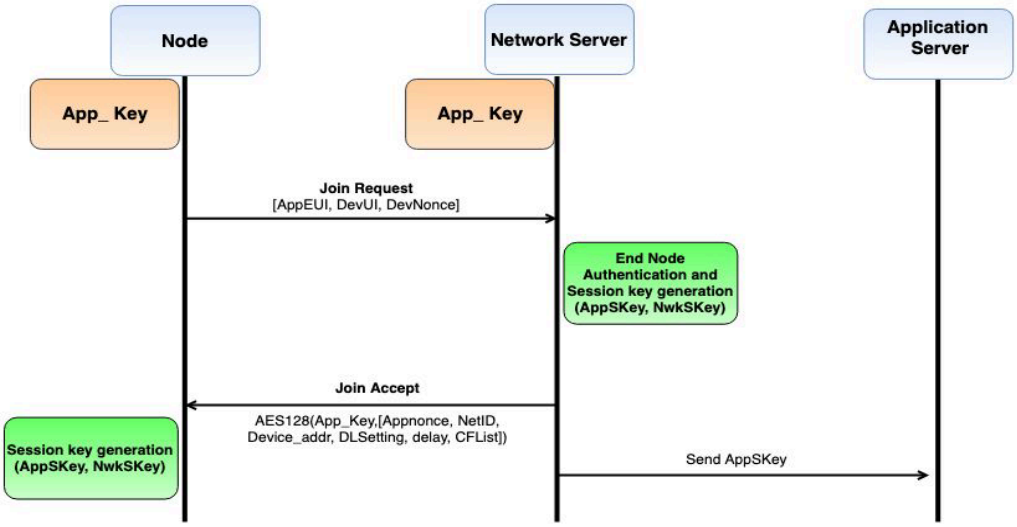

Figure 2.10: OTAA Join Procedure LoRaWAN [40]

ii. *Activation by Personalisation (ABP):* In this procedure, the nodes are deployed with secret keys, and they can directly start transmitting data without any registration procedure. This process saves time and energy but is considered less secure as the same key is used for the lifetime of the node.

Once the nodes join the network by either of the two procedures, the upcoming messages are encrypted, and MIC is calculated using the combination of network and application key, as shown in Figure 2.11.

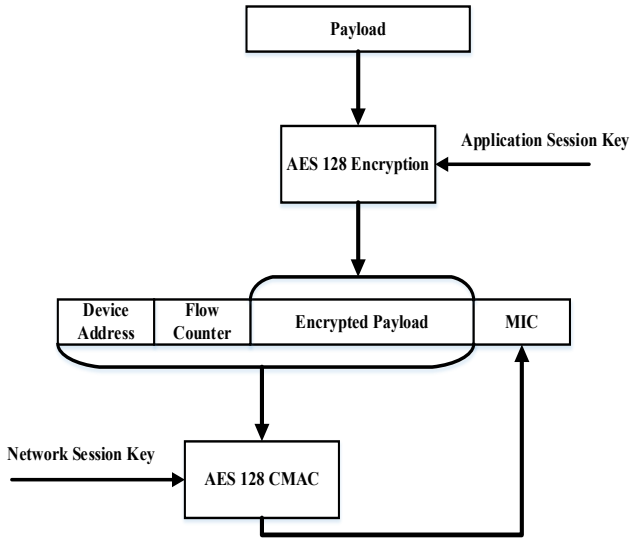

Figure 2.11: Encryption mechanism in LoRaWAN [40]

2.3. IoT Security Threats

Numerous applications of IoT in multiple domains and the use of constrained devices can make IoT a feasible target of attackers. As mentioned in chapter 1, IoT is divided into a layered framework. Authors in [21] discuss vulnerabilities of IoT layers against various attacks. This section will discuss security threats that can target IoT applications.

2.3.1. Perception Layer Security threats

The Perception layer consists of the physical objects that are responsible for sensing and for data generation. The devices are deployed in hostile and unsecured environments. These devices can be physically accessed and manipulated by attackers. Security keys can be extracted, causing grave damage to the whole network of devices. Authentication of physical devices is crucial for the security of applications using the services of the devices. An unauthenticated device can transmit false data to the applications, causing damage to integrity in the network.

In [50], threat analysis is performed on physical devices. The author explains the possibilities of attacks over the device's lifetime. Various factors, including its manufacturing, can cause device vulnerabilities are explained. It is discussed how faulty device manufacturing can leave backdoors in the devices which attackers can exploit. The second and most crucial phase is device deployment. Devices are configured initially to join and operate in a network. In a large scale deployment, there are possibilities of hasty device configuration causing security vulnerabilities.

Perception layer threats can be avoided to an extent with proper manufacturing and configurations during deployments. However, devices are still vulnerable to physical attacks.

2.3.2. Communication Layer Security Threats

The Communication layer is responsible for transmitting data from devices to gateways and further to application servers. IoT devices majorly use wireless transmission to communicate. Communication is most likely targeted by attackers as it does not require any physical access to intercept data from the wireless medium. The communication medium is prone to various attacks that can cause severe damage to the performance of IoT networks.

Possibilities of various attacks are discussed in [21, 23, 51-54]. In [1], the trade-off between energy and routing protocol is discussed. The author also discussed the requirement of an efficient routing protocol that considers the resource constrained nature of IoT network and can provide state of the art security in data transmission. Table 2.6 shows various attacks and their impacts on the networks.

Table 2.6: Attacks and Impacts on IoT networks [21]

Communication layer attack	**Impact on network**	**Possibility**
Jamming attack	Very High	High
Sinkhole attack	High	Medium
Wormhole attack	High	Medium
Hello Flooding attack	Moderate	Medium
Man in the middle attack (MITM)	High	High
Sybil Attack	High	Medium

2.3.2.1. Jamming Attack

The wireless transmission uses radio frequencies for communication. A jamming attack obstructs the nodes from communication by occupying the communication channels. Jamming attacks are divided into four categories [55], i.e. constant, deceptive, random and reactive jamming. Most wireless mac protocols use CSMA-CA, allowing them to access the channel when it is not occupied. However, in constant jamming attacks, the attackers continuously send radio signals to occupy the channel, causing nodes in the network to be unable to access the channel. In case of deceptive jamming, the attacker continuously sends packets, keeping nodes in receiving mode only. Every jamming strategy focuses on forcing the nodes to stop sending data, stopping the whole network from functioning.

2.3.2.2. Sinkhole Attack

Sinkhole attacks target the routing of the network and cause damage to the networks that are using multi-hop routing. A malicious node is planted in the network to break the network connectivity [56]. The malicious nodes advertise false route information to attract the nodes to use that path, causing the nodes in the network to start sending data through the malicious node. The sinkhole alone may not cause

grave damage. However, combined with other attacks, it can be lethal. Figure 2.12 shows the scenario of a sinkhole where the sinkhole is attracting all the network towards itself.

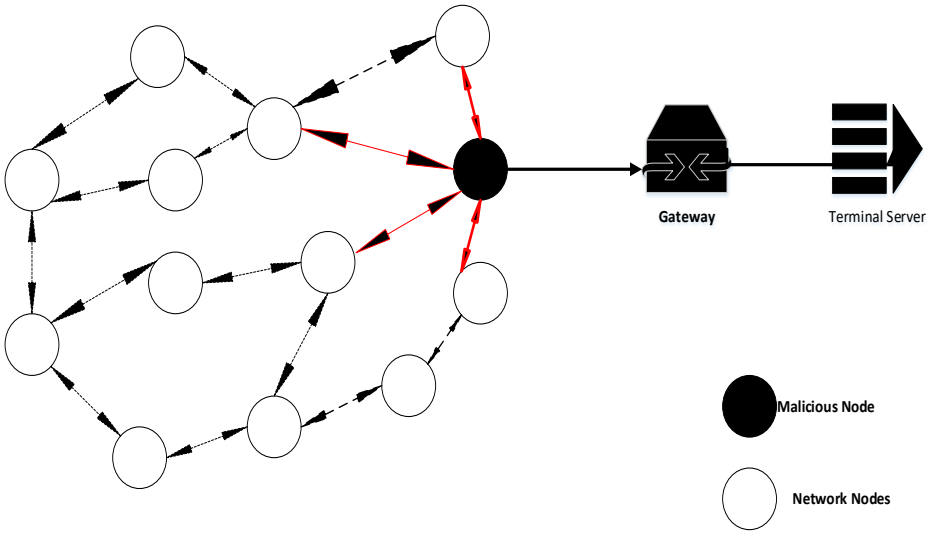

Figure 2.12: Sinkhole Attack

2.3.2.3. Wormhole Attack

A wormhole attack is a coordinated attack that requires at least two nodes [57]. The two attackers are connected through a "wormhole tunnel". The attacker at one end receives packets from nodes and forwards them to the other attacker. The other attacker, after receiving the packets, replays them back in the network. The attack does not require any cryptographic information, and the replayed packets can disrupt smooth packet transmission. Figure 2.13 shows an outline of wormhole attacks and how it impacts nodes in the network.

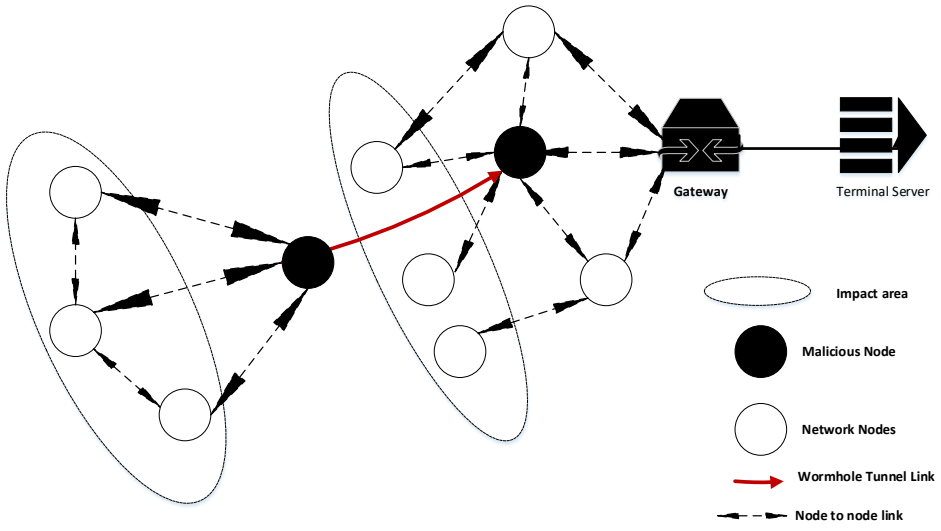

Figure 2.13: Wormhole Attack

2.3.2.4. Hello Flooding Attack

The hello flooding is a special type of Denial of Service (DoS) attack. It can be launched on various routing protocols that use "hello" packets as control packet to update network connectivity in dynamic networks [58]. The malicious node sends hello packets to a neighbour node, causing the neighbour node not to participate in routing activities for legitimate nodes in the networks. This attack is not very effective if launched with a single malicious node; however, it can cause serious damage to network connectivity with multiple attackers.

2.3.2.5. Man in the Middle Attack (MITM)

MITM attack is a passive attack where the attackers intercept data being transmitted in the network. The intruder can monitor the communications between network nodes, creating a possibility of confidentiality breach in the network. The attackers can use the sniffed packets in multiple attacks like Address Resolution Protocol (ARP) poisoning. The captured packets can be retransmitted to network nodes with a replay attack causing a breach in integrity.

2.3.2.6. Sybil Attack

The malicious nodes create multiple identities of a node in the network to create ambiguity in the network. Figure 2.14 shows an outline of a Sybil attack where the malicious node impersonates itself as multiple other nodes in the network. This attack can affect the integrity of the compromised node as the central servers will be receiving data from two nodes with the same identity and will have to either accept or discard data from both legitimate and malicious nodes.

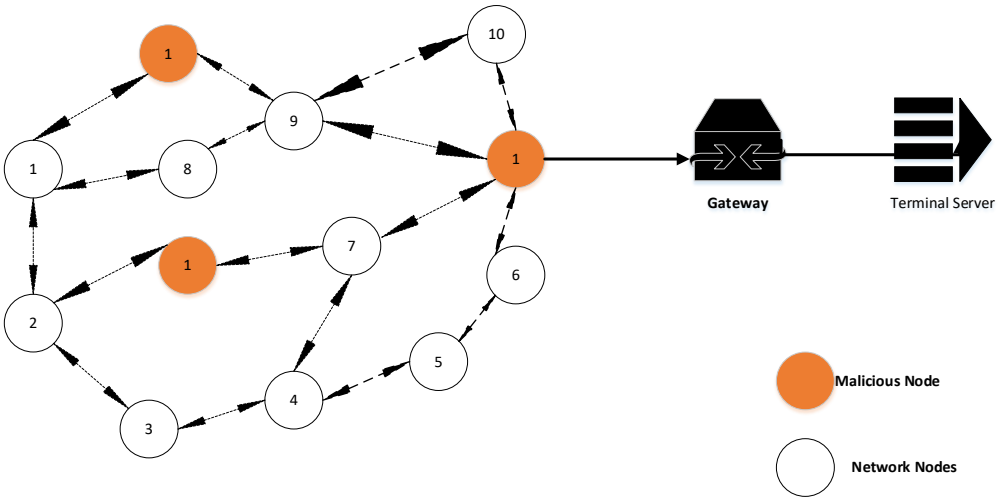

Figure 2.14: Sybil Attack

2.3.3. Business / Application Layer Threats

This layer is responsible for providing services and acts as the front end. Users Interact with this layer to provide inputs and receive outputs from the respective applications. This layer is vulnerable to attacks that can exploit applications and human errors in the provision of credentials. Following are some of the attacks that can be launched on the application layer.

2.3.3.1. Social Engineering attacks

Social engineering [59] is the art of getting users credentials by exploiting the users' psychological aspects. The targets are users with information, and attackers persuade them into leaking information. The attackers use the collected information to access the applications and launch attacks using user's access details. Technical defence mechanisms are usually useless against these attacks as the attackers use legitimate credentials to access information. The only way to avoid this is to train the users against these attacks.

2.3.3.2. Buffer Overflow Attacks

Buffer overflow attacks are one of the most common attacks launched on applications by exploiting any bugs in the applicatios. The attacker tries to overflow the memory and stop the application from functioning. This attack can significantly affect the integrity of the application, leading to a significant disruption in service provision.

As discussed above, IoT layers are prone to various attacks. Mostly the attacks target a particular layer of the system. However, some attacks target multiple layers of the IoT system, and they can severely damage the system performance.

2.4. Software Defined Networking (SDN)

SDN has introduced programmability to the networking devices that vendors usually hardcode in the traditional networking paradigm. Network programmability was introduced in the '90s with projects like Open Signalling [60], Active Networking [61], DCAN [62], NETCONF [63] and Ethane [64]. However, ForCES [65] and OpenFlow [66] are the most recent architectures used for SDN. Several groups of organisations have formed the Open Networking Foundation (ONF) [67] to standardise OpenFlow for SDN [68].

As shown in Figure 2.15, SDN decouples the control plane from the forwarding plane and allows the control plane to be programmed according to network requirements [68]. In traditional networking, the forwarding devices are controlled by predefined firmware installed by the vendors, and the control firmware cannot be reprogrammed, forcing the users to use what is provided by the vendors irrespective of the requirements. However, SDN makes it possible to program the network as required. This network

programmability feature opens new doors for other third-party software mechanisms that can be integrated with forwarding rules in the controller [69].

- Forwarding Plane/ Data Plane: The forwarding or data plane consists of forwarding (either software or hardware based) entities that use instructions or rules provided by the control plane to manage the incoming or outgoing traffic through the ports.

- Control Plane: The control plane consists of policies or rules that forwarding devices follow. The control rules are communicated to the forwarding entities using a secure channel (this depends on whether or not the switches support secure communications with the controller). The control plane is the actual brain behind the behaviour of the networks and their policies.

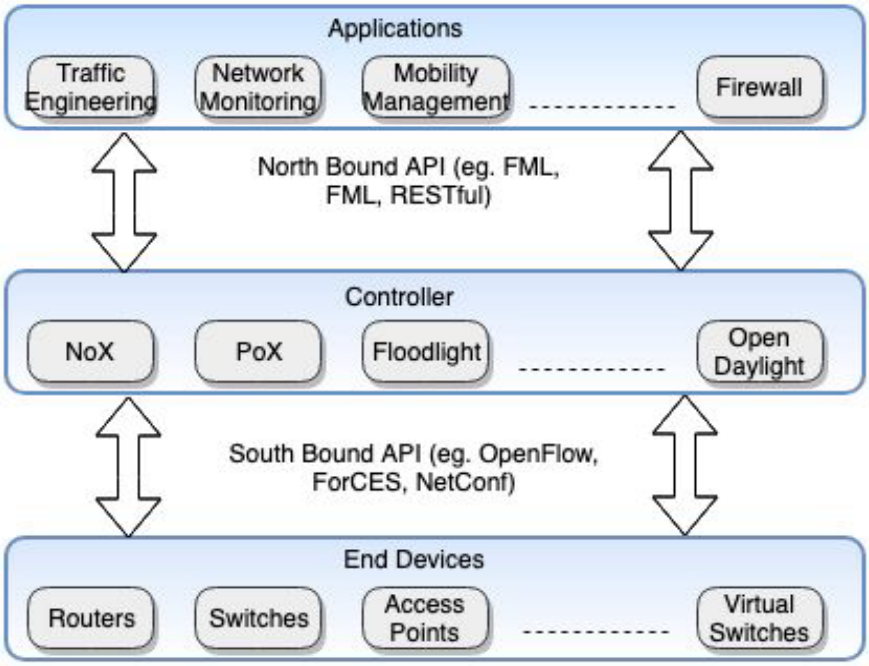

Figure 2.15: Software Defined Networking Architecture

2.4.1. Challenges and Opportunities for SDN based Security in IoT

The introduction of SDN has opened new doors in multiple areas of networking. However, there are multiple views of researchers when it comes to SDN-based security architectures. Some say that it will bring diverse opportunities for the networks, bringing revolutionary changes in the way network management and network security are handled. On the other hand, researchers claim that SDN will bring security threats to the network components. Both views have solid reasons for their claims. There are several potential SDN frameworks giving rise to opportunities as well as challenges [69-71] discussed as follows:

Centralised Control: In traditional networking, the controls are distributed over the routing devices, and the forwarding devices hold the authority to make routing decisions.

On the other hand, SDN takes out the control plane from the device and gives it to a controller that can be remote or local. This centralisation brings a single point of failure in the network. It makes the work of an attacker easier, as they have to bring just one control entity down for the network to fail. However, this centralisation of control offers a hawk eye that helps the network react to the anomalies. When it comes to routing, network firewalls and network monitoring systems, having a global view can undoubtedly decrease the reaction time [71].

SDN Standards: OpenFlow has brought flexible network customisation where the routing devices can be controlled from a remote controller by pushing flow entries into OpenFlow switches. However, OpenFlow is still in its growing phase and still has few vulnerabilities in the standards adopted by various vendors [72, 73]. The communication between switches and controllers is highly vulnerable as the latest version of OpenFlow (v 1.5.1) has made the Transport Layer Security (TLS) for communication optional. Because of which the vendors have an option not to use the TLS in their switches as TLS brings complex configuration requirements with it [72]. The absence of TLS leaves the controller switch communication open for a man in the middle attacks as the control data is being transferred as plain text. Also, there other threats like switch authentication, controller authentication, and flow verification [72]. These challenges must be addressed in the coming versions of standard protocols in SDN for the technology to be successful.

2.5. Machine Learning for Network Security

A machine learning algorithm is a computational process that takes input information and produces a desirable output without being programmed ("hardcoded") [74]. Machine learning algorithms tend to learn from a dataset to make accurate predictions in the learned domain based on specific sets of input. This capability of finding complex patterns in data gives a new dimension to network security and defence mechanisms.

Machine learning can be majorly divided into three categories; Supervised, Unsupervised, and Semi-Supervised.

Supervised Machine learning: Supervised learning [74] is used when we have a dataset with label outputs. The labelled dataset is given as input to a machine learning algorithm. The algorithm then learns to generalise based on the labelled data. Once the model is optimised to minimise the errors in prediction on the input data (training data), anonymous data is provided to the trained machine learning model to make predictions of the label based on the provided variables.

Unsupervised Learning: The significant difference between supervised and unsupervised machine learning [74] is the labelling of the dataset. In unsupervised learning, the data is not labelled. The learning algorithms automatically decide the label of the samples based on commonalities in the features of data samples. Majorly, the unsupervised learning algorithms work on the clustering principle where it clusters the data samples with common features and labels them.

Semi-Supervised Learning: In several cases, there are limited labelled samples, and most training samples are unlabelled. In such cases, semi-supervised machine learning [74] is used. The dataset is divided into two categories of labelled and unlabelled data samples. Supervised techniques are used on the labelled data samples and unsupervised clustering mechanisms are used on the unlabelled data samples in the dataset. Supervised learning is used to find a relation between data samples of the same cluster, and unsupervised learning is used to create similar data samples. In this manner, semi-supervised learning uses both supervised and unsupervised learning for classifications.

2.5.1. Supervised Learning Algorithms

Supervised learning is mainly a process to map an input set to an output set. Each sample in the input set is associated with one output set sample, and the learning algorithm has to find the best possible way for the input set to create a general mapping to the output set [75]. Supervised machine learning has shown great potential in classification and regression problems for whom labelled historical data is available.

Supervised learning is one of the most common techniques used in classification problems. It provides a massive range of techniques that can be used to address different classification tasks based on available data. Some of the most common supervised learning algorithms are discussed as follows:

2.5.1.1. *Linear Classifiers*

Linear classifiers are the simplest yet effective supervised learning technique that uses linear combinations of the features to classify the data samples into output classes [76]. For a binary classification, the linear classifiers split the data samples into two classes. However, for multiclass classifications, multiple linear classes are trained against each class available to find a decision boundary. Linear classifiers are primarily used when the classification speed is the primary focus. Some of the most common linear classification techniques are logistic regression and Naïve Bayes classifiers [77]. Both classification techniques use linear classification with different mathematical approaches to decide decision boundaries in data samples.

2.5.1.2. *Support Vector Machines (SVM)*

SVM constructs an N-dimensional hyperplane to separate the data into two categories [76]. SVM is one of the most effective techniques that can be used on high dimensional data. Even when the data

dimension is higher than the number of data samples, SVM performs exceptionally well. However, SVM fails to perform well on huge dataset because of high training time. In cases of high dimensional data, it requires precise regularisation and hyper-parameter tuning to perform well. Also, SVM struggles to perform well when there is noise in the dataset, and there is overlap in output classes.

2.5.1.3. *Decision Trees*

Decision trees follow a multistage decision making approach. It breaks down a complex decision making process into multiple simple decisions [78]. Each node in the decision tree represents a feature of the sample to be classified, and each branch carries the value of the feature based on which the branches are divided in the decision tree [77]. Decision trees are simple to understand and visualise and require less data preparation to train the model. However, decision trees tend to overfit the data and require pruning and hyper-parameter tuning to avoid overfitting [79].

2.5.1.4. *Neural Networks*

Neural Networks are one of the most well-known machine learning techniques that simulate learning in biological beings. Biological neurons are replaced by artificial neuron node, and the connections carry the weights for the connected neurons. The function of input is calculated by propagating the computed values from input neuron to output neurons with weights as intermediate parameters [80]. The neural networks are competent in doing complex classifications where non-linear learning is required. However, the Neural Networks have non-convex loss functions and can get stuck in local minima during optimisation; it can be prevented by random initial weight allocation. Also, the Neural Network is very sensitive to feature scaling and has many hyper parameters that require tuning for optimal performance [79].

2.5.2. Supervised Machine learning for IoT security

Supervised learning techniques require historical data to predict the future occurrences of attacks on the networks. Machine learning techniques are mainly used to detect attacks on networks by monitoring the network node behaviour.

2.5.2.1. Attack Detection Methodologies

Attack detection methods are majorly categorised into three categories. The first one is Signature-Based Detection of attacks: a signature string or pattern is designed for known attacks on the networks based on knowledge gathered from previous attacks from the network. The incoming traffic is compared with the stored signature in a database. In case of a matching pattern, the Intrusion Detection Systems (IDS) raises a flag of possible intrusion on the network and suitable actions for blocking the attacks are taken by the IDS. The signature based detection is also known as Misuse and Knowledge Based Detection. The signature based detection mechanisms are simple and easy to implement and are fast to react to known attacks on the networks. However, they are ineffective in the detection of novel attacks on the

network. It is also challenging to update the signature of the attacks as new attacks are introduced to the system, and it can be tedious to maintain the knowledge base.

The Second type is Stateful Protocol Analysis (SPA). SPA works on the comparison of the protocol's profile and node behaviour in the networks. It uses profiles of vendor protocols used by network devices to establish a typical behaviour of the protocol. It compares it with the incoming traffic to identify attacks on the networks. Unlike signature based protocols, it does not create a list of signatures. Instead, it analyses the protocol behaviour it is working on comparing with incoming traffic. The analysis and learning of protocols can cause an overhead. However, it protects against unknown attacks, unlike signature based IDS.

The third type of detection method is Anomaly Based Detection. It is a behaviour based detection mechanism that continuously monitors the node activities in the networks and reports any anomalous behaviour of the node in the network. Anomaly based IDS requires setting a baseline behaviour as the "normal" behaviour of nodes in the network. To define a baseline for incoming traffic, the IDS must be introduced to various protocols being used in the networks. The IDS must analyse all the protocols for their behaviour to draw a baseline for all of them. The mechanism of baseline design can be computationally extensive and time consuming. However, once the baseline is set, the anomaly based IDS can be very effective for attack or intrusion detection in the networks. An anomaly based detection system's significant advantage over a signature-based system is its capability of detecting unknown attacks on the network. As the anomaly based IDS does not depend on signatures, any node behaviour that deviates from the baseline is flagged as an attack on the network.

Anomaly based IDS can be categorised into three major categories [81]: i) Statistical Based; ii) Knowledge Based; and iii) Machine Learning Based. All three categories are discussed as follows:

Statistical Based: The statistical based anomaly detection techniques solely rely on the statistical information of various aspects of the data. It uses metric based profiling such as data rate, packet count for particular protocols, specific IP addresses. The statistical based anomaly detection uses two datasets of the network in the process. One is the current data from the network, and the second is past data to create the statistical profile. The current data is compared with the previous statistical profile to compare the network behaviour to calculate the anomaly score. If the anomaly score goes beyond a threshold, the system flags the event that occurred as an anomaly. Statistical based anomaly detection is fast and does not require huge amount of data to create the statistical profile of the network traffic. However, as it uses simple statistical rules, an attacker can be trained by an attacker so that the system starts identifying the attack as regular traffic.

Knowledge Based: Knowledge bases anomaly detection, also known as expert systems, are intended to classify audit data based on specifications (set of rules) that determines legitimate system behaviour. It works in three steps. First, the data attributes and classes are recognised from the training dataset.

Secondly, the specification is designed from the selected attributes and classes for the network traffic. Finally, the specifications is used on audit data for classification based on rules designed in the specification of the network traffic. There are several ways to design the specification for the expert systems. It can be designed by human experts to classify the data, or it can be designed by some tools using finite state machines. The main advantage of this anomaly technique is its robust and flexible properties. However, developing an optimal specification for a system can be a very challenging and time consuming process.

Machine Learning Based: Machine learning based anomaly detection techniques utilise the state of the art machine learning models to analyse the patterns in the dataset. It uses a labelled dataset, where the data samples are labelled as different classes based. The labelled dataset is provided as input to the machine learning models. The machine learning models learn for patterns in the dataset based on various features in the dataset. The machine learning based anomaly detections can utilise any machine learning algorithms that provide the best solution on the dataset. Some of the well-known machine learning algorithms are discussed in section 2.5. They are utilised for training a model, and the trained model monitors the traffic in the network. Based on the decision boundaries created during the model's learning process, it classifies the traffic based on its feature values.

Chapter 3

Related Work

There has been a considerable amount of work for security in LPWAN IoT networks. In [82], Wooyoung Jung et al. have used a lightweight SSL for sensor networks with insufficient processing power at nodes. The techniques use Elliptic Curve Cryptography (ECC) for authentication and key exchange, RC4 is used for data encryption and MD5 for hashing. The authors claim their system is fit for healthcare and military applications. However, their scheme uses rigorous algorithms to perform the tasks with high processing requirements that can increase overhead on the nodes, resulting in a faster energy drain. Datagram Transport Layer Security (DTLS) with 6LowPAN compression for the constrained devices is taken into consideration in [83] to reduce the header size of DTLS to fit in 802.15.4 maximum transmission unit(MTU). The compressed DTLS is linked with a 6LowPAN standard mechanism for security. This scheme gives the ability to implement a complicated security mechanism in constrained nodes, as by compressing, we can reduce the size of the header information. The data compression reduces the bandwidth consumption; however, the compression is still to be performed by the end nodes. In [84] Swapped Huffman tree coding is used to compress and encode(encrypt) the data with a secret key. This technique is very lightweight but may not be very resilient against sophisticated attacks on the data. The encoding patterns can be analysed, and codes for plain text can be deduced. In [85], a load relieving scheme for the nodes is proposed, where each node can share the processing burden for encryption with a set of assisting nodes. This scheme can enable the nodes to overcome the problem of limited resources by combining the resources of multiple nodes. However, this scheme can lead to complications in encryption, like maintaining the integrity of the data that is being shared among the nodes for encryption. This also will lead to an increase in the duty-cycles of the network nodes, which will lead to high power consumption. A similar approach [86] uses a proxy-based encryption scheme, where the nodes distribute the processing overhead of encryption with the proxy nodes. The technique tries to embed trust-based proxy node selection. The nodes select trusted proxy nodes and distribute their encryption load between them to save energy. It is assumed that the proxy node and the end nodes have a secure connection, which is challenging to provide in constrained networks. However, finding trusted nodes to perform the encryption and maintaining the integrity and authenticity of the data is itself a very challenging task.

Researchers have also modified the well-known encryption techniques for constrained nodes in IoT to minimise the processing overhead. In [87], the Blowfish encryption algorithm for data encryption is implemented for constrained devices. The Blowfish algorithm may provide strong data confidentiality

and has a faster calculation of the ciphertext; however, it requires good hardware capabilities and processing cost in the devices to perform the data encryption. There is also a higher memory requirement for its long key setup.

Xuanxia Yao et al. proposed attribute-based encryption(ABE) based on Elliptic Curved Cryptography rather than bilinear Diffie-Hellman pairing [88]. The proposed encryption is no-pairing ECC- based ABE, suitable for constrained devices and to secure the communication in the network. The authors claim that the proposed encryption has a lower overhead than key-policy ABE and ciphertext-policy ABE. [89] Proposed a cypher-policy ABE (Attribute-based Encryption) using some precomputation techniques. The basic idea is to precompute and store the data obtained from expensive operations to decrease the ECC computation. This scheme can save a significant amount of processing cost. However, the precomputation data can take large memory in the devices to store all the generated pairs. In [90], prevention against side-channel in existing LEA(Lightweight Encryption Algorithm) is proposed. LEA was standardised in South Korea in 2013 for IoT devices. LEA is a block encryption algorithm used to provide confidentiality in a constrained environment. It uses addition, rotation and XOR operations and it does not use S-Box lookup like AES. The bit pattern of the original data is changed to prevent the side channel attack in LEA, and the change information is kept in the extra 4 bytes to decipher the data. By doing that, the differential power analysis used for side channel attacks becomes useless. The downside of removing the S-Box is that it may lead to a decrease in the strength of the encryption and it may enable easier cryptanalysis of the ciphertext.

Symmetric key encryption is comparatively lightweight when compared with public key encryption techniques. Researchers have also proposed to embed public key cryptography to constrained IoT networks. In [91], Fagen Li et al. has proposed a solution for the secure sensor-server communication from sensors in Identity Based Cypher-text (IBC) environments to a server using a Public Key Infrastructure(PKI). The proposal is a “heterogeneous ring syncryption” technique, which can achieve integrity, authenticity, non-repudiation and confidentiality for the data. The framework uses a third party as a private key generator (PKG); it first generates a master key and later uses the node ID and the master key to generate a private key for every node. The end node receives the private key and calculates a public key with that. [92] Proposed a multi-key exchange using elliptic curve cryptography operations and an encryption key to exchange session keys between the nodes. The proposed protocol is claimed to be capable enough to handle 40 sessions at a time. The author claims the protocol to be suitable for IoT operations. However, ECC requires rigorous calculations. The constrained nodes in IoT are not capable of doing heavy processing for data encryption.

Node authenticity is an essential factor for data integrity in IoT networks. As the number of nodes in an IoT network can be very high, the authentication of those nodes can be challenging and crucial. If not authenticated, any unauthorised entity can join the network and send false data to servers. Annie Gilda

Roselin et al. [93] proposed an authentication technique for the end node's verification using MAC messages. The proposed algorithm uses a symmetric key without a pre-shared key. They have used four flights (stages) for establishing the authentication and session keys, where each flight uses existing information (PAN ID, node ID) to derive the new key. The authors suggest that the authentication is practical and lightweight. However, it has a long registration process and four tiers of key calculations from the edge routers.

A secure joining procedure for LoRa nodes is suggested in [94]. The author proposed a dual key based joining procedure for the new joining nodes in the network. They also mention that they can update the shared key in the initial deployment to enhance the security. The proposed algorithm will take more energy because two keys are preloaded, and both the session keys are calculated separately from different keys. It can be considered a reasonable trade-off for better security. However, this scheme is limited to only LoRa nodes.

In [95], a user and node authentication mechanism is proposed. The proposed scheme works in four phases; 1) Sensor registration: the sensors are deployed with predefined keys where a trusted authority generates keys. 2) User Registration: This phase is for the users to access the sensor data. The Gateway Node registers the user with the help of a username, password and a random nonce. A biometric ID is generated and stored in a smart card for future authentication. 3) Login and authentication: a smart card with a biometric Id is used for logging in by the user for sensor data access. 4) Password Change: The users can change the password without communicating with the gateway node. However, the scheme brings additional hardware requirements (smart cards). End nodes are not given much consideration and are use pre-stored keys for communication. In addition, most of the computation is performed by the gateway node creating a single point dependency.

A lightweight cloud-based concealable biometric authentication mechanism for IoT client node id is proposed [96]. The mechanism uses the biometric prints of the user to generate the authentication information. The mechanism has two phases: Enrolment and Authentication. In the enrolment phase, a user is registered with their biometric print, the print is then further pre-processed, and features are extracted out of it to be used as authentication data. In the authentication phase, the reverse process is done, and authentication data is matched with the enrollment phase's data.

To verify the authenticity of the data being received from nodes in an IoT network, a lightweight attestation mechanism "AAoT" is proposed in [97]. It uses a challenge-and-result pair to authenticate nodes and data. In the setup phase, the central entity gives a challenge to the node, and the node performs some calculations and returns the results to the authenticator. The challenge and result pair is stored in the server. To verify the authenticity of the node, a particular challenge is given to the node, and the result is compared to the data stored in the server. The scheme does not need to save any secret key information in the node. Instead, a function is used to generate results out of challenges provided by the

authenticator. Authentication preambles for LoRaWAN against DDoS attacks in class B LoRaWan are proposed in [98]. It is a mechanism against exhaustion attacks in which the attacker sends very long messages to the node in its receive cycle to exhaust the battery. The mechanism uses preamble authentication to reject unauthenticated packets without waiting for them to be received (just based on the preamble).

In [99], the authors proposed a D2D communication mechanism for LoRaWAN. The LoRaWAN standard does not support D2D communications as the devices directly transmit data to the gateway node. However, the authors claim that it can be power efficient to use D2D communications and have proposed a key establishment mechanism with help from the network server in a LoRa network. The Two devices have to initially communicate with a network server to obtain key information for the nodes to communicate. D2D communications can be very useful in power constrained networks as the nodes do not have to transmit data for long distances, and they can use less energy for data transmission. Secure communication between devices can enable power efficient transmission in LoRaWAN networks.

In [100], an authentication mechanism for sensor nodes is proposed where nodes can authenticate the gateway and any other sensor nodes in the network. The proposed algorithm focuses on resource constrained nodes and proposes an authentication mechanism for M2M communications. In the initial setup (Registration), the Mobile Service Provider (MSP) installs secret keys in gateways and every sensor node. During the first phase, the gateway is authenticated by the mobile user with the help of secret keys. In the later phases, the sensors authenticate the gateway with the help of a shared secret key and a random nonce.

Most of the research conducted depends on pre-shared keys for authentication and confidentiality of data in the network. However, using a single key for the lifetime of the communication can be vulnerable to attacks on the network. A node authentication and session key mechanism using hash chains are proposed in [101]. The nodes are divided into virtual subdomains based on their capabilities and the scope of inter-domain communications to manage the communication flow among the nodes. For authentication and key exchange mechanisms, the whole process is divided into three phases. The first phase is for initialisation, where the system administrator selects a random master key for the network controller. The controller calculates three authentication parameters using a hash chain securely stored in nodes in the second phase and is used for authentication and session key exchange in the third phase. The scheme provides a mechanism for replay attack prevention along with authentication and key exchange. However, the node will have to perform quite a few computations for authentication and implement the key exchange mechanism.

A CoAP [102] based lightweight mutual authentication mechanism is proposed in [103]. CoAP is one of the most famous application layer protocols used in IoT. The proposed scheme adds a security

mechanism to CoAP itself instead of adding an additional protocol for security. The nodes are pre-stored with secret keys. The server sends an encrypted challenge to the node, decrypted by the pre-stored secured key only, and nodes do the same to authenticate the server. An advantage of the scheme is that it incorporates the security mechanism in CoAP without adding any additional protocol overhead. However, the mechanism totally depends on the pre-stored keys, and encrypted text must be sent every time the session changes, which adds processing overhead to the nodes.

Pre-stored keys are usually used for security mechanisms. However, they require memory/storage and can be leaked on a physical attack on a node in the network, jeopardising the security of the whole network. In several pieces of research, the physical attributes of nodes are used to generate key information. In [104], an authentication and key generation mechanism using physical layer parameters is proposed. The scheme utilises physical layer parameters to generate keys. The received signal strength over multiple frequencies is measured first. The standard deviation difference from the mean of the signal strength is used to produce a quantization parameter to filter the qualified keys. Both parties use a hash of the key to check if they have obtained the same keys. In case of failure, the process is repeated.

M.N. Aman et al. [105] proposed an authentication mechanism based on a Physical Unclonable Function that works with a challenge-response mechanism. It eliminates the requirement to store keys in the node. The authors present a mutual authentication and session key mechanism for servers and IoT nodes. The server uses a challenge and response pair for every node and sends a challenge to IoT nodes for authentication. The challenge and response generation depend on FUP functions that, in turn, rely on the physical properties of the nodes. If one node wants to authenticate another node in the network, it must go through the central server for authentication.

Authentication and secure data transmission in healthcare applications for 6LoWPAN devices is the topic in [106]. Every sensor node has two pre-stored arrays of random numbers that are used to authenticate the nodes. Nodes select five random numbers from the first array and perform multiplication, and the result of the multiplication is arranged according to the order given in the second array. The result is sent to the server, and the server does the same operation at its end to check the node's authenticity. All the communications are encrypted by a pre-stored secret key shared by all the devices in the network.

A secure data sharing mechanism implemented by breaking and storing information over a group of nodes is shown in [107]. The proposed mechanism works on splitting and sharing information amongst a group of nodes rather than entrusting all the information to a single node. The mechanism works in four stages: in stage 1, a dealer distributes secret information amongst a set of nodes, a key encrypts the information based on a client's Physical Unclonable Function. In stage 2, the client retrieves the information by selecting a set of shareholders and authenticates the retrieved information. In case of

error, it moves on to stage 3, which is for error correction. In stage 4, the client does group testing if the data retrieved is correct and is from authenticated sources. However, a distributed approach like this can have a high possibility of errors because of node schedules and network scaling. This scheme is also not for sensor-based applications where the data is being generated by a device independently.

3.1. Key Agreement Mechanisms for IoT

Storing Keys in the devices and using the same key for the whole node lifetime can cause security vulnerabilities. Zero knowledge key agreement mechanisms enable two entities to calculate a shared secret key using some public information of the second party. The Diffie-Hellman [108] algorithm is one of the most famous key agreement schemes; as it is vulnerable to Man In The Middle attack, it must be used with other mechanisms to authenticate the public keys of the two parties [109]. Public key mechanisms require complex mathematical calculations, which is very challenging for IoT nodes with limited resources.

There are several lightweight key agreement algorithms suitable for resource-constrained IoT nodes. Blom's key agreement [110] uses simple mathematical calculations to generate a secret key for two entities using their public information. Blom's scheme had a limitation with the number of nodes because the data that needs to be stored in the nodes increases as the network scales. Despite the limitations, Blom's scheme is further developed into the Blundo scheme [111]. Blundo's scheme uses bi-variate polynomials for secret key calculations for two parties; IDs of the participating entities are used to calculate the secret key.

Several variants of Blom's schemes were proposed by using it in several manners [112-118]; in [109], the author proposed an extension to Blom's mechanism, i.e. the "Blom Yang key Agreement (BYKA)" algorithm uses multiple public and private keys. It uses its permutations to calculate common secret keys between the nodes. The algorithm can provide authentication and immunity from a man in middle attacks with very low overhead on end nodes.

Mutual key agreement mechanisms while preserving user anonymity are proposed in [119, 120]. The mechanisms initiate user registration. Users set an ID and password to a third party or centralised server. The server uses a hash mechanism to calculate a smart card number for user authentication

A mutual device to device authentication mechanism with forwarding secrecy is proposed for ZigBee devices [121]. The protocol uses symmetric key encryptions and enables devices to have a key agreement for a session key. The session key is changed frequently to provide forward secrecy. Pre-deployment, every node has a unique ID and a key generated by using devices' inner circuit chips. This key is considered the secret key for the device. All the devices register themselves to a controller. Access control for all the devices is also configured during device registration. For the device to device

communications, the nodes use a controller as a middleman to authenticate each other and then come to a session key agreement.

Lightweight mutual authentication and key exchange mechanisms for Wireless Body Area Network (BAN) are proposed in [122, 123]. In [122], the proposed mechanism works in three phases: initialisation, registration and authentication phase. The system has sensors and hub nodes. The sensors transmit information to the Hub Node (HN). In the initialisation phase, the administrator chooses a master key for HN and stores it in HN's memory. In the second phase, the administrator registers the sensor nodes by choosing secret identities and keys. Two authentication parameters are calculated using the secret identities, and keys are stored on both HN and the sensors' memory. The advantage is that the secret key is not stored in the nodes. The results show that the proposed scheme calculates 0.3ms and energy consumption f 0.035mJ on a 32-bit Cortex-M3 microcontroller.

In [124], a lightweight key establishment mechanism for wearable devices in IoT is proposed. It works on the principle of one-way cryptographic hash functions and XOR operations. The mechanism works in four phases: system setup, registration, authentication and key exchange, and password change phases. The authors have argued that the scheme protects against attacks like impersonation attack, insider attack, eavesdropping and password guessing attacks. The proposed algorithm also provides forward secrecy and node anonymity.

In Ruotsalainen, H. [125], a key generation and refreshment mechanism for LoRaWAN communication devices are proposed. The mechanism works in seven stages to generate AES128 keys for devices. (1) Channel probing: The devices communicate with the gateway and stores received signal strength (RSSI), signal to noise ratio (SNR), and packet counter value at both parties. (2) Measurement preselection: Gateway analyses the RSSI and SNR, and the gateway select only the values that match with antenna configuration. (3) Measurement match: The end device performs a measurement calculation based on uplink message information received from the gateway to achieving no packet drop. (4) Precorrection: The correction of synchronization measurements is performed in this stage using a cosine transformation process. (5) Quantization: The received measurements are converted into key bits. (6) Reconciliation: The errors in calculated bits are corrected in this stage. (7) Privacy affiliation: To remove the possibility of key information leak, SHA256 is used for final key generation. The authors state that security keys can be regenerated every three hours by using this scheme. However, no analysis of the energy consumption is performed. As the process requires end nodes to perform several calculations, it most probably will have an adverse effect on the node lifetime.

Han, B. [126] follow a similar approach to generate keys for LPWAN devices with a four-step procedure: (1) channel probing; (2) reconciliation; (3) quantization; and (4) privacy affiliation. The authors have used signal strength sequences to quantize into a secret key. A two-step quantization mechanism is used. In the first step, the RSSI sequence is converted into binary bits. In the second step,

a level crossing algorithm generates an initial sequence of the secret key. As the key agreement requires the end nodes to probe and collect channel information to generate the keys, there can be probable negative impacts on node lifetime. The authors have not done any analysis of the energy consumption of the scheme.

A key management and update mechanism is proposed in Xing, J. [127] for LoRa devices. Elliptic-curve Deffie–Helman (ECC-DH) is used for key agreements, and a hierarchical deterministic (HD) wallet is used for key management in the system. The server and the devices generate a public and private key pair in their HD wallet using the BIP32 algorithm [128]. After the key pair generation, the end devices register themselves to the server by sending their public keys. The server generates a root key pair based on the information received from the device and stores it for communication. The mechanism uses ECC-DH for a key agreement after receiving the public information from devices in the network. The proposed mechanism enables the devices to update the security keys for better security. The key generation mechanism adds communication and processing overhead by using eight-step for key generation. However, the authors argue that it is a reasonable trade-off for the security of the network. LPWAN networks are vulnerable to attacks like identity theft or the impersonation of legitimate nodes. These attacks can cause substantial damage to the authenticity of nodes that are transmitting data to users. It becomes essential to have a robust attack detection mechanism.

3.2. Machine Learning Based Node Identifications

The node authentication and encryption techniques are essential for data confidentiality and authenticity. Also, IoT networks are vulnerable to attacks like identity theft or the impersonation of a legitimate node. These attacks can cause substantial damage to the authenticity of nodes that are transmitting data to users. There are many attempts in constrained network environments to minimise the effect of these attacks by identifying the devices for early detection of an attack and for taking the necessary actions.

The authors in [129, 130] discuss the usability of node fingerprinting for security in a sensor network. The approach in [131] uses the time difference between the acknowledgement and authentication packets to create a node's fingerprint. In [132, 133], the neighbour information is used to create fingerprints and identify clones in the network. In [134], GTID (Georgia Tech ID cards inspire the name) is proposed for device type fingerprinting. It has four components: feature extraction, signature generation, similarity measure and enrolment. GTID primarily relies on the Inter-Arrival Time (IAT) of the packets from devices and generates their fingerprint based on the traffic rate distribution of devices. Once the signatures are generated, they are used to train artificial neural networks and patterns are registered to add the devices to the network.

A radio frequency feature based node fingerprinting is proposed using entropy in RF features in [135]. The authors have proposed to use Permutation Entropy and Dispersion Entropy on features sets to calculate a node's fingerprint. In experimentation, nine devices were used, and 900 packet samples were processed. Various machine learning models like KNN, SVM and Decision Tree were trained on selected features from samples. The models were able to achieve accuracy up to 82%.

In [136] authors introduced the "PARADIS" technique, which uses the defects in hardware to identify the nodes. Differences in individual frames in the modulation domain are analysed, and machine learning classification tools are used to identify the end nodes. Deviation in the clock skews of transmitting devices to create fingerprints to identify the devices is used [137]. The authors in [138-140] have used radio frequencies to create device fingerprints. In [141], the authors used the signal's preamble to generate node fingerprints to identify impersonators in the network. In [142], the feasibility of using signal power distribution in space to fingerprint nodes for identification is discussed. The authors argue that instead of using clock skew for identification, it is more feasible to use signal power distribution as it is harder to forge for an intruder. The authors in [143] described various statistical methods used on node and packet properties to identify 802.11 based nodes. In [144], the authors extend [143] and add an anti-forgery mechanism so that forging node fingerprint becomes more challenging. The authors argue that the mechanism works well without hindering the performance of 802.11.

3.2.1. Classification

The attack sophistication on networks is increasing; attackers use intelligent and layered techniques to launch attacks. These sophisticated attacks are difficult to detect with straightforward rule-based detection mechanisms. Hence, researchers have tried to use machine learning techniques to detect these sophisticated attacks.

In [145] the author has trained multiple machine learning and neural network models to compare their accuracy over each other. The models are trained over simulated data on various attacks like Blackhole, Greyhole, Flooding and scheduling. The neural network model trained in the experiment detected the attacks with 92.8%, 99.4%, 92.2% and 75.6% for Blackhole, Flooding, Scheduling and Greyhole attacks, respectively. However, the model was trained over simulated data, so accuracy variations are possible when implemented for real traffic.

A real-time dataset was collected in [146]. The data set "BoT-IoT" was captured from a test-bed environment, and various attacks were launched. The dataset contains several sophisticated attacks, including DoS, DDoS, and theft, with all these attacks further divided into sub-categories. There were several models trained on the datasets to detect these sophisticated attacks in the datasets. Deep neural networks were also trained to classify these attacks. The neural networks were able to attain a promising accuracy of up to 99% on the dataset. However, the deeper the neural network, the more time and

resources required to train the models. These deep neural network models will require efficient machines to classify the attacks in the network.

In [147], binary classification is done. The dataset contains attack and normal data samples. The trained classifier focuses on differentiating the traffic between attack and normal. The authors argue that their model can attain 99% accuracy in classification. However, the model is trained on DDoS attacks and can only detect single attacks. Binary classification can be simpler in comparison to multiclass classifications and consume fewer resources. However, immunity against multiple attacks is crucial in networks supporting multiple applications.

Node Identification based on radio signal irregularities is proposed in [148]. The authors proposed an Identification mechanism for nodes based on their signal irregularities. The receiver fingerprints every node based on attributes like frequency offset, I-Q imbalance, DC offset channel information. On top of these attributes, a neural network is trained to identify the nodes in the network. No secret key is to be stored in the node for authentication. The gateway performs all the processing. Although no key is required for authentication, a secret key will be needed for secure data transmission.

A deep learning architecture is used to detect anomalous structures in traffic datasets in the so-called social internet of things [149]. The NSL-KDD dataset is used to train, test and validate the deep learning model. The algorithm works in two steps. First, the data samples are classified into two categories of attack and normal traffic. In the second phase, multiclass classification is performed on categories of the attacks that include DoS, Probe, R2L and U2R attacks. Two types of models, deep and shallow models, are trained over the dataset. The results indicate that the deep models outperform shallow models with accuracy in both binary and multiclass classifications. However, a downside of the deep architecture is that it takes too long to train and run the models. Also, the experiment was conducted by deploying the model on a centralized configuration (i.e. on a single fog node) and a distributed configuration, i.e. using multiple fog nodes. The results show that the model implemented on a distributed system can detect sophisticated attacks better than those used in a centralized system.

3.2.2. Anomaly Detection

Anomaly based network IDS are quite capable of detecting outliers or anomalous traffic in the network. There have been several projects conducted by researchers using various methods. In [150], collaborative anomaly detection is proposed by the authors. The UNSW-NB15 [151] dataset is used to train and evaluate the accuracy of proposed models. The authors argue that the model can achieve a promising accuracy of up to 96.7%, and it is ready to be implemented on a cloud system handling large scale data. However, a system implemented on the cloud will detect higher layer attacks, but attacks on communication layers may go undetected. AN IoT environment is focused in [152], where a lightweight anomaly detection mechanism for constrained networks is proposed. The proposed anomaly detection

approach uses game theory to activate a detection mechanism. It is argued that the proposed mechanism is lightweight and energy conscious, and highly accurate. The authors argue that the nodes in sensor networks and IoT networks should check for anomalies continuously. The continuous anomaly check can increase the overhead and decrease the lifetime of the network. [152] Proposes to use a Nash Equilibrium (NE) to calculate an equilibrium state when the nodes will train models and build signatures for anomaly detection to save resources in the nodes. The Anomaly Detection System for IoT (HADES-IoT) is proposed in [153]. HADES focuses on low performance overhead, and it is suitable for IoT systems. It is a host-based detection system implemented on a Linux kernel as most of the IoT based devices are Linux based. The system creates a whitelist of devices that are allowed in the system after profiling them. The authors tested the system on seven IoT devices and argued that it is highly effective and suitable for energy constrained IoT nodes. Maede Zolanvari et al. performed a vulnerability analysis on IoT in [154] and discussed how vulnerabilities in IoT systems are different from vulnerabilities in traditional networks. The authors built a dataset that contains normal traffic, backdoor traffic, command injection and SQL injection traffic where 99.81% is normal traffic and 0.19% is attacks' traffic. Multiple machine learning models like Random Forest, Decision Tree, KNN, ANN, SVM, Logistic Regression, and Naïve Bayes were trained on the datasets. In conclusion the authors emphasise on the abilities of machine learning algorithms in detecting anomalous traffic in the network. A use case of smart building is taken by authors in [155] for detection of anomaly in application traffic. A context aware anomaly-based detection mechanism is proposed that focuses on Building Automation Systems (BAS). The Building Automation Control networks (BACnet) protocol that is used for building automation control is analysed and the security threats are discussed in detail. The authors used a building at the University of Arizona as a suitable testbed for experimentation. The proposed system was able to detect attacks in real-time and proved to be highly accurate. In another attempt of intrusion detection for WiFi networks, SVM is proposed as a potential classifier. MAC layer information is analysed from MAC headers and useful features are extracted to train the classifier.

A two-tier classification mechanism for network intrusion is proposed in [156]. The NSL-KDD [157] dataset is used to train and test the classification model. The proposed model has two modules, i.e. a dimension reduction and a classification module. The dimension reduction module is used to filter out useful features, as using every feature in the dataset can cause an error in decision making. Principal Component Analysis (PCA) and Linear Discriminant Analysis (LDA) are used in the first module to extract features out of the dataset. After choosing the required features with the first module, a classification is performed in the second module. Two well-known classifiers, K-Nearest Neighbours and Naive Bayes are used. The results show that the proposed algorithm efficiently detects attacks where user machines are compromised or attacks where a user tries to use root privileges to exploit the network.

Table 3.1 summarises the key contributions discussed in the literature for constrained networks along with their possible shortcomings for the researches using traditional approaches for security. Table 3.2 and table 3.3 summarise the contributions and possible shortcomings of key agreement and machine learning based security approaches in constrained networks respectively.

Table 3.1: Summary of Key Contributions from Literature in Constrained Networks Using Traditional Approach

Security Domain	**Paper Name**	**Contributions**	**Shortcomings**
Authentication	A three-factor anonymous authentication scheme for wireless sensor networks in Internet of Things environments [95]	Proposes Authentication mechanism for nodes and users with 4 phases. The proposed algorithm covers user authentication to ensure that no unauthenticated users can log in to the system to access the sensor data from the gateway.	It brings additional hardware requirement (smart cards). End nodes are not given much consideration and are using the pre-stored key for communication. In addition, most of the computation is performed by the gateway node creating a single point dependency.
Authentication	A lightweight machine learning-based authentication framework for smart IoT devices [96]	The paper proposes a lightweight cloud-based concealable biometric authentication mechanism for IoT client nodes. The algorithm uses biometric print. However, feature extraction and random matrix generation authentication data can be updated even with the same biometric prints.	Focus is given only to user authentication. Sensors in the networks are not targeted to make the whole system secure.
Authentication	AAoT: Lightweight attestation and authentication of low-resource things	A lightweight mutual authentication and attestation procedure for “things” in the network to	The number of challenge and result pair stored per node will be limited. How

Security Domain	Paper Name	Contributions	Shortcomings
	in IoT and CPS [97]	provide integrity and tamperproof features. The scheme does not need to save any key information in the node. Instead, a function is used to generate results out of challenges provided by the authenticator	the scheme will scale is not discussed.
Authentication	Authenticated Preambles for Denial of Service Mitigation in LPWANs [98]	Introduction of authentication preambles for LoRaWAN. Analysis of DDoS attack in class B LoRaWan. The solution is simple and effective for B Class devices. Authors argue that the scheme is power efficient	The Methodology is focusing on Class B nodes in LoRaWAN networks.
Authentication	Secure mutual authentication and automated access control for IoT smart home using cumulative Keyed-hash chain [101]	Node Identification and session key exchange mechanism is proposed using hash chains. The scheme provides a mechanism for replay attack prevention along with authentication and key exchange.	Node has to perform quite a few computations for authentication and key exchange mechanism.
Authentication	A payload-based mutual authentication scheme for Internet of Things [103]	CoAP base lightweight mutual authentication without adding additional protocol layer, unlike DTLS. The advantage of the scheme is that it incorporates the security	The mechanism depends on the pre-stored and encrypted text that has to be sent every time the session changes, adding processing overhead on the nodes.

Security Domain	Paper Name	Contributions	Shortcomings
		mechanism in CoAP without adding any additional protocol overhead.	
Authentication	A Novel Physical-layer Security Scheme for Internet of Things [104]	Authentication and key generation mechanism using physical layer parameters. The use of physical layer properties can avoid pre storing of keys in the nodes	Nodes may require special hardware to generate multiple frequencies used to obtain the signal strength in the mechanism.
Authentication	Design of a Secure Password-Based Authentication Scheme for M2M Networks in IoT Enabled Cyber-Physical Systems [100]	An authentication mechanism for sensor nodes is proposed where nodes can authenticate the gateway and any other sensor nodes in the network. The mechanism is lightweight and suitable for constrained nodes, and secret keys are never transferred over the air. The hash mechanism is used to authenticate the data sent by each entity	The mechanism depends on the pre-stored key. In the case of a sensor to sensor authentication, the nodes have to communicate to the gateway for getting authentication.
Authentication	Security of 6LoWPAN IoT Networks in Hospitals for Medical Data Exchange [106]	Authentication and secure data transmission in healthcare applications for 6LoWPAN devices is proposed. The scheme is lightweight as there is no complex calculation in the mechanism.	However, sharing same key with every device is not considered very safe in the network. The mechanism perform. The Encryption process mentioned is not an AES but simple mod operation. Which can be easy to break.

Security Domain	Paper Name	Contributions	Shortcomings
Authentication	Mutual Authentication in IoT Systems Using Physical Unclonable Functions [105]	Proposed an authentication mechanism based on a physical Uncolonable Function that works on a challenge-response mechanism. It eliminates the requirement to store keys in the node. The mechanism does not store any keys in the memory, and any data generated for authentication is temporary and deleted once the authentication is successful.	As mentioned by the authors themselves that the protocol has high latency for delay-sensitive applications.
Confidentiality	A secure and robust scheme for sharing confidential information in IoT systems [107]	Secure data sharing mechanism proposed by breaking and storing information over a group of nodes. Distributing information can be beneficial in case of node compromise.	There can be high chances of errors because of node schedules and also as the network scales. This scheme is also not for sensor based application where the data is being generated by a device independently.
Confidentiality	SSL-Based Lightweight Security of IP-Based Wireless Sensor Networks [82]	The authors proposed a lightweight SSL based transmission for sensor network using ECC for key exchange and authentication, RC4 for data encryption, and MD5 for hashing mechanisms.	The use of public-key cryptographic mechanisms such as ECC can have higher requirements of processing and energy. The paper does not discuss how using the three mechanisms for SSL in sensor networks

Security Domain	Paper Name	Contributions	Shortcomings
			impacts the node lifetime in the network.
Network Integrity	6LoWPAN Compressed DTLS for CoAP [83]	A security mechanism for 6LoWPAN networks is proposed on top of the DTLS security mechanism. The header of DTLS is compressed to achieve the Minimal Transmission Unit (MTU) in 6LoWPAN networks. The compression mechanism of the DTLS header enables the network to use the sophisticated security features of DTLS in constrained IoT networks.	The compression mechanism enables the constrained nodes to use DTLS. However, the nodes are required to compress the headers for every packet, causing an overhead on the nodes that can lead to a lower lifetime of the node.
Confidentiality	Swapped Huffman tree coding application for low-power wide-area network (LPWAN) [84]	The authors target the challenge of huge data transmission and security in IoT. Swapped Huffman Tree coding technique is used to perform compression and encoding of data for security and compression.	The proposed mechanism performs Huffman tree encoding for every data packet. The research does not explain how it affects the node lifetime and its sophisticated defence cryptanalysis.
Confidentiality	C-CP-ABE: Cooperative Ciphertext Policy Attribute-Based Encryption for the	Considering the resource constrained nature of the nodes in IoT networks, a cooperative encryption approach is proposed. The	Data division and joining can be complicated scenario. There can be a possibility of data integrity

Security Domain	Paper Name	Contributions	Shortcomings
	Internet of Things [85]	nodes share the burden of data encryption to avoid overhead on a single node.	breach while sharing and recollecting the data.
Confidentiality	Enhancing the security of the IoT LoraWAN architecture [86]	The nodes share the encryption overhead by choosing trusted proxy nodes in the network. The data is distributed amongst the proxy nodes for encryption.	The proposed mechanism presumes a secure connection between end nodes and the proxy nodes, which is challenging to achieve in constrained networks.
Confidentiality	An implementation of data encryption for Internet of Things using blowfish algorithm on FPGA [87]	A well-known encryption algorithm, “Blowfish”, is implemented on constrained FPGA devices. Blowfish is a faster and powerful encryption algorithm and is suitable for secure transmission in IoT networks.	The Blowfish algorithms have requirements for compatible hardware to work. In addition, it has a higher memory requirement to store its long key setup in the node.
Confidentiality	Lightweight Attribute-Based Encryption for the Internet of Things [89]	The proposed mechanism focuses on minimising the processing overhead of data encryption on the node. Pre-Computed encryption information is stored in the node to minimise the computations required for encryption.	The proposed approach is effective in decreasing computations. However, introduces additional memory requirement at end nodes to store the pre-computed encryption data.

Security Domain	Paper Name	Contributions	Shortcomings
Network Integrity	An improved LEA block encryption algorithm to prevent side-channel attack in the IoT system [90]	The proposed mechanism uses an LEA encryption algorithm to protect IoT nodes from side-channel attack. It changes the bit pattern of the original data and stores the change information is kept in extra 4 bits added to the data.	The LEA encryption removes the S-Box and uses XOR and rotation operations for encryption. That can lead to a decrease in encryption strength.
Authentication	TTP Based High-Efficient Multi-Key Exchange Protocol [93]	A multi-staged node authentication technique is used to authenticate the nodes using MAC layer information of the nodes. The proposed mechanism does not require any pre-shared information between nodes and server for authentication.	The process of registration is long and requires multiple exchanges between nodes and edge routers for authentication.
Authentication	A Dual Key-Based Activation Scheme for Secure LoRaWAN [94]	The paper outlines the limitations of current LoRaWAN node activation mechanisms where a single key is used to generate session key with the network server. The application server is not involved in session keys. In the proposed scheme, two keys are used to generate session keys. Moreover, two separate session keys are	The dual key procedure requires the nodes to store two keys instead of one. Furthermore, the scheme is limited to LoRaWAN platform.

Security Domain	Paper Name	Contributions	Shortcomings
		generated at the network and application server.	
Authentication	A Secure Device-to-Device Link Establishment Scheme for LoRaWAN [99]	A secure D2D communication scheme for LoRaWAN is proposed. D2D communication can be very useful in power constrained networks as the nodes do not have to transmit data for long-distance and use less energy for data transmission. Secure communication between devices can enable power-efficient transmission in LoRaWAN networks	As both devices communicate with the network server to obtain key information. It can be challenging to synchronise node clocks, and it will become even challenging as the number of nodes increase. As nodes do not have any information about each other.

Table 3.2: Summary of Key Agreement Based Contributions from Literature in Constrained Networks

Security Domain	Paper Name	Contributions	Shortcomings
Authentication	Anonymous mutual IoT interdevice authentication and key agreement scheme based on the ZigBee technique [121]	The protocol uses symmetric key encryptions and enables devices to a key agreement for a session key. The session key is changed frequently to provide forward secrecy.	Using a controller as an intermediary can cause a delay in the authentication process.

Security Domain	Paper Name	Contributions	Shortcomings
		The session keys can provide strong forward secrecy in the network	
Authentication	A lightweight anonymous user authentication and key establishment scheme for wearable devices [124]	A four-step key establishment mechanism is proposed for wearable devices using XOR and hash operations.	The proposed mechanisms have a long registration process and require multiple transmissions between node and gateway.
Authentication	Experimental Investigation on Wireless Key Generation for Low-Power Wide-Area Networks [125]	A key generation and refreshment mechanism for LoRaWAN is designed. The proposed mechanism works in seven stages. It uses the physical layer parameters like Signal to Noise Ratio (SNR) and RSSI to calculate the keys on LoRaWAN nodes.	The proposed mechanism can achieve a secure key generation at long ranges. However, the seven-step procedure can create overhead on the nodes. The impact of key generation on node lifetime is not discussed in the research.
Authentication	An Improved Secure Key Management Scheme for LoRa System [127]	The proposed mechanism follows an eight-step procedure of node registration and key generation using ECC with the Diffie-Hellman algorithm. It provides secure key generation using public information to achieve superior security for LoRa nodes.	The eight-step procedure and ECC, a public key algorithm, can add additional overhead on the end nodes and shorten the node lifetime.

Table 3.3: Summary of Key Contributions from Literature in Constrained Networks Using Machine Learning Techniques

Security Domain	**Paper Name**	**Contributions**	**Shortcomings**
Authentication	RF-PUF: Enhancing IoT Security Through Authentication of Wireless Nodes Using In-Situ Machine Learning [148]	Node Identification based on radio signal irregularities. All the burden of processing is shifted to the gateway, and no key is stored in node memory	Although no key is required for authentication, the secret key will be needed to secure data transmission.
Authentication	Implications of radio fingerprinting on the security of sensor networks [129]	The proposed mechanism uses radio parameters from captured packets to create a fingerprint of a node. The fingerprint can be used to identify every node uniquely and prevent the network from various attacks.	The fingerprinting mechanism has a probability of incorrect identifications of the nodes in the network. The authors mention in the research that the mechanism needs improvement for higher accuracy.
Network Integrity	Real-time detection of clone attacks in wireless sensor networks [132]	A clone attack detection mechanism is proposed for sensor networks. A social fingerprint of the network is created based on the neighbour information of the node. The social fingerprint is used to identify the real node from the clone nodes.	The use of social fingerprinting can create challenges in a dynamic network where the topology changes.
Authentication	GTID: A technique for physical device and device type fingerprinting [134]	A physical layer information based fingerprinting mechanism is proposed to identify	The fingerprinting mechanism focuses on the devices already in the networks and detects the

Security Domain	Paper Name	Contributions	Shortcomings
		unknown devices to the networks with the help of Artificial Neural Networks (ANN). The applicability of the fingerprinting mechanisms for the authentication of nodes in the networks is discussed.	intruder based on its signature. However, as the network changes, new devices are introduced in the networks with no previous information. How will the proposed fingerprinting mechanism will cope with that change is not discussed in the research?
Authentication	Physical layer authentication of Internet of Things wireless devices through permutation and dispersion entropy [135]	Radio frequency fingerprinting of devices is proposed. The permutation Entropy mechanism on top of the fingerprinting is discussed for node authentication in the network.	The fingerprinting mechanism targets IoT networks that are highly diverse and constantly changing. The fingerprinting mechanism requires historical data sample from devices for Identification. The fingerprinting mechanism may face challenges in keeping up with the dynamic nature of the networks as new nodes join the network with no historical data samples.

3.3. SDN and IoT Security

As discussed in section 2.4, SDN provides multiple features that can be used to administer the network efficiently. Because of the versatility of SDN, multiple types of research are being conducted on SDN applicability in different areas of the network, and network security is no exception to this. Table 3.5 shows a few of the attempts to amalgamate SDN with IoT security.

Table 3.4: SDN based security in IoT networks

Paper Name	**Brief review/ Contribution**	**Shortcomings**
Black SDN for The Internet of Things [158]	An SDN architecture for IoT is proposed to manage the traffic and network resources to provide better services in the network. The architecture shows a multilayer controller design to manage the heterogeneous networks in the IoT environment. The architecture comes up with the concept of resource matching that identifies the requirement of the application and confirms the capability of the network to fulfil the requirement. This architecture manages to get better throughput, end to end delay and jitter than existing load balancing and bin packing approaches.	The resource matching technique will need to send requests to the controller for a new application, adding an extra delay in the process.
A Hierarchical Security Framework for Defending Against Sophisticated Attacks on Wireless Sensor Networks in Smart Cities [159]	A hierarchy-based framework is proposed for the higher security of wireless sensor networks. Detection of attacks is divided into two parts, sensor end and base station end. The sensor nodes perform low-level detection with basic rules that require less computation. The base station performs detection with sophisticated rules with higher computation requirements. Once the attack is detected, SDN and network virtualisation are used to mitigate the attacks on the network.	The involvement of sensor nodes in detection will require the active participation of sensor nodes, where they will require gathering data and using decision-making algorithms to detect attacks on the network. Sensor nodes are energy constrained. The more end nodes participate in the detection mechanism, the more energy is consumed.
Identity-Based Authentication Scheme	An authentication mechanism proposed for a heterogeneous environment. The technique is divided into three phases: 1) Gateway public key certification; 2)	The controller is powerful enough to perform the complicated operations of ECC. However, it is not very

Paper Name	Brief review/ Contribution	Shortcomings
for the Internet of Things [160]	Things Registration; 3) Authentication Phase. Gateway is certified by the controller in the first phase. In the second phase, end nodes register their identities to the controller using the controller's public key with ECC. The third phase is for node authentication, where the node's public key is used to send encrypted data to the end node.	efficient to perform such rigorous calculations at the constrained node.
Dynamic Attack Detection and Mitigation in IoT using SDN [161]	The attack on end nodes is detected by continuously analysing network traffic. SDN controller continuously monitors traffic patterns from nodes and uses learning modules from stored attack models. An anomaly in traffic pattern is detected using a machine learning algorithm to identify an attack on the network.	Using learning modules of past data can have limitations with accuracy; the controller will also have to gather a considerable amount of data to perform anomaly detection in network traffic and increase the attack detection time.
Software Defined Intelligent Building [162]	A lightweight security mechanism was proposed for intelligent buildings. Mutual authentication between the SDN controller and sensors is proposed. Controller and devices share a secret key before any communications. In the first phase, the device authenticates the controller where devices generate a Nonce and controller calculates HMAC on receiving the nonce and data including the secret key in MAC controller sends it to end node. The end node again calculates the HMAC on its end to compare and verify the authenticity of the controller. Nodes are verified by the	

Paper Name	Brief review/ Contribution	Shortcomings
	controller in the same manner, using Nonce and secret key.	
S^2 Net: A Security Framework for Software Defined Intelligent Building Networks [163]	An extension of [162], SDN based network model. SDN controller is introduced in the control layer to interact with smart devices and applications as an intermediary. Usually, the SDN controller interacts with a switch to control, but SNET (controller) directly interacts with the end devices. The proposed model tries to attain integrity, confidentiality, authenticity, and lightweight with session keys management.	
A Secure IoT Architecture for Smart Cities [67]	The author proposes a secure architecture for IoT using, Black network, an SDN controller and a key management registry. The black network is used to add privacy, the key registry is used for key management, and SDN controllers are used for secure routing in the network.	The proposed black network uses traditional security approaches for IoT nodes that are not very efficient for IoT networks.
IOT SENTINEL: Automated Device-Type Identification for Security Enforcement in IoT[164]	IOT SENTINEL is proposed as a system that can identify the types of devices connected and enforce the rules to minimise the damage from node compromise. Device fingerprint is created by using previous network traffic. SDN controllers are used to identifying the fingerprints of the new devices and enforce security rules on them.	Using only traffic data for device fingerprint can have errors in classification and identify the devices; multiple criteria should be used to identify devices instead.

Chapter 4

Methodology

This chapter discusses the process of designing the methodology for the research conducted in this thesis. The chapter begins with a general definition of research and attempts to understand the type of research appropriate for this thesis. Further, the chapter explores various research approaches to understand and follow a suitable research approach to follow to achieve the goals of this thesis. After exploring and understanding the research approaches, a justification for the research approach's choice is provided. After the finalisation of the approach, the methodology for the research is explained along with its justification. Subsequently, the research methods utilised to achieve the final goal of the thesis are explained and justified for their use.

4.1. Different types of research

Research, in general, can refer to a search for knowledge. It can also be defined as a scientific and organised search for meaningful information on a particular topic [165]. Research leads to the discovery of different forms of knowledge related to practices in various fields.

In general, research relates to the discovery of knowledge. However, as we look closely at the research, it can be categorised into different types; in [166] categorisation of research is explained according to different perspectives:

1. Application perspective of research: Considering the prospect of the application of the research, it can be categorised into two categories: pure research and applied research. Pure research focuses on formulating a theory to generalise and understand a phenomenon. Pure research mostly deals with theories and developing hypotheses. It may not have immediate practical applications in the near future. On the contrary, applied research concerns the practical challenges affecting individuals or masses.
2. The objective perspective of research: from the point of the objectives of the research, it can be categorised into four classes: descriptive, correlation, exploratory, and explanatory. Descriptive research attempts to describe or explain the properties of events being studied. It does not concern the questions of “how”,” when”, and “why”. It focuses on the question of “what”. Correlation research focuses on relationships of events in which one situation or event impacts another event and then another; it finds the dependence of phenomenon on other phenomena, and that’s why it is called correlation research. The next category is explanatory research that addresses the “why” or “how” of the phenomenon. How one event

relates to another occurrence? It explains the details of the dependencies of the events on one another. Finally, exploratory research focuses on understanding or exploring an area or subject.

Considering both perspectives of the research, this thesis can be categorised as applied research because it attempts to address a practical task that is a current requirement of IoT communications technologies. The security of communication technology can have a huge impact on the applications that depend on these technologies and the users who benefit from those IoT applications.

From the Objective prospect of research, this research has multiple objectives it attempts to achieve. The thesis proposes an initial security framework using SDN and explores the framework's impact on various IoT network performance metrics.

4.2. Process of Research

Every research, irrespective of its category, requires following a process to achieve its objectives. The research process provides a general guide that any research can follow to achieve its goals and answer the research questions. In [166], the research process is explained to have three phases covered by eight operational steps. The steps are not necessarily in a particular sequence and can be iterative, depending on the research requirements. The three phases of research are deciding, planning, and undertaking. The three phases focus on addressing different requirements of the research. The first phase is the initial phase of the research, where the researcher explores and attempts to understand what can be done. The first phase includes exploring the research areas and designing a hypothesis based on the literature review for the initial formulation of the research problem. The phase dives deeper into the literature and attempts to conceptualise the research design. A more extensive literature review is performed to design methods or frameworks for data collection and the research tools utilised for data collection. By the end of phase two, the research proposal is formulated, and research components are more concrete as to what needs to be done and how it can be done. The third and final phase is about conducting a study on the collected information. It focuses on results obtained from the data processing and verifying the hypotheses formulated for the research. The third phase utilises mathematical tools to interpret the information gathered from the collected data and to obtain insights from it. By the end of phase three, a research report is formulated containing details of the research.

This thesis also follows a similar research process to organise the work, and it has a streamlined operational understanding. Figure 4.1 shows the research process followed by this research to organise and achieve the objective of the study.

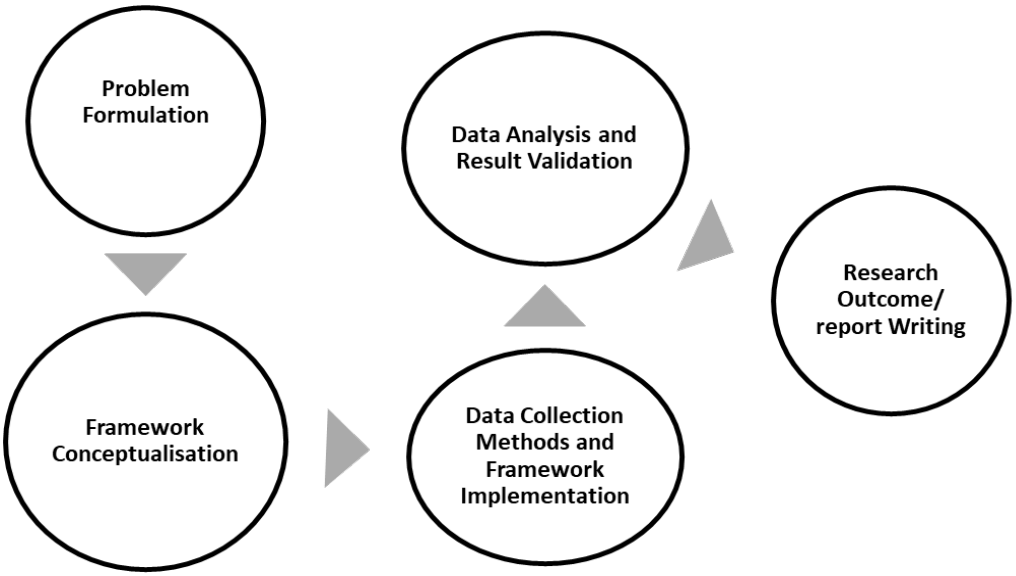

Figure 4.1: Research Process

As mentioned in the definition, research is an organised way of discovering knowledge. Depending on the subject area and type of research, there are a few approaches that can be utilised to realise the goals of the research.

4.3. Research Approach

The research process provides a general idea of the flow or steps that are carried out for the organised execution of the research. To provide a more specific idea about the research procedure, research approaches are utilised. A research approach is a plan that provides a funnelled vision for data collection, analysis, and result interpretations. The selection of the research approach relies on the research problem, researcher's experience and the area of study. In general, research approaches are categorised as qualitative, quantitative, and mixed methods. This section attempts to explore all three approaches and determine the most appropriate approach for this research.

1. Qualitative Research Approach: Qualitative research approach deals with the subjective evaluations of opinions, behaviours, and attitude [165]. It targets human behaviour and social problems [167]. The research is designed mostly to collect data through participants and the researchers attempt to find the meaning from the data based on their interpretations. The conclusion from the collected data can be subjective and dependent on the researchers understanding and experience in the study area.
2. Quantitative Research Approach: Quantitative approach follows a more mathematical approach towards data collection and analysis. It focuses on realising the relationship between various variables [167]. The variables are extracted using various instruments and provides a numeric output. The numeric data can be then analysed using statistical methods. The researcher then

interprets the statistical information to provide an outcome and conclusion of the research. The quantitative approach can be divided into subcategories [165] of inferential, experimental, and simulation. Inferential research is used when data is collected through surveys and depends on the population and their opinion. Experimental research collects data by conducting experiments in a controlled environment and by manipulating various variables to analyse their impact on each other. The third approach is the simulation approach which collects data in an even more controlled environment. An artificial environment generates data based on mathematical models for various variables to analyse the correlation between variables.

3. Mixed Method Approach: The mixed method approach follows a hybrid research approach by mixing quantitative and qualitative data. It follows both philosophical assumptions and theoretical frameworks to conclude the outcome of the research based on the collected data. It relies on the fundamental assumption that the amalgamation of both quantitative and qualitative data results in better insight than provided by either a qualitative or a quantitative approach alone.

4.4. Research Design

In addition to selecting research approaches, the researchers also need to select one of the three research designs [168]. Research design provides the strategy to address the research questions. As the scope of research has grown, the number of research designs have also increased for the researchers. In this sub-section three key research designs are discussed:

1. Quantitative Research Design: The quantitative research design follows the postpositivist worldview and mostly follows an experimental approach. Quantitative research designs can be categorised further into four major types: descriptive, correlational, quasi-experimental, and experimental [168]. The four design branches have different methods, and features they provides to conduct experiments. The descriptive design focuses on describing and interpreting the information collected about a sample's natural phenomenon or properties in their natural environments. The findings of descriptive research are useful in explaining a natural phenomenon that is novel and little knowledge is available. However, it fails to provide the cause of the phenomenon that can be used to establish a hypothesis for further research. Correlational design analyses the relationships between variables in the collected data. The research findings are conveyed using statistical methods as positive, negative, or no correlation between the variables of the data.

   Experimental and quasi-experimental designs focus on the analysis of intervention by manipulating the independent variables of the experiments. The quasi-experimental design follows comparatively less rigorous experiments and fits in scenarios where it is not feasible to

conduct randomised controlled experiments. Hence, it fits well for healthcare research studies [168].

2. Qualitative Research Design: the qualitative research design follows a non-experimental approach and relies on data provided by the research participants. Qualitative research is also categorised as narrative, phenomenological, grounded theory, and case study-based research. Narrative research relates well with the study of humanities, where the researcher studies the lives of individuals. The researcher collects information about participants' daily lives and narrates their stories according to the researcher's interpretation. Phenomenological research enquiry describes the experience and prospects of participants about a certain scenario. The research follows an interview approach where participants can describe the impact of a phenomenon on them. Then the researcher conveys a collective interpretation of the collected data from participants. On the other hand, grounded theory designs an abstract theory considering the view of the participant and then refines the theory by conducting data collection in multiple stages. Finally, case studies are one of the most common designs followed by many types of research where the researchers derive in-depth knowledge about a case. The cases are usually events, processes, or activities. The researchers collect data about the case by using various data collection techniques and in-depth knowledge about the case from various prospects.
3. Mixed Method Research Design: Mixed method involves the combination of both quantitative and qualitative research methods. The qualitative data collection is open, and quantitative research targets close-ended questions. The mixed method uses multiple methods to collect data from the participant with an amalgamation of both open and close-ended questions to the research participants. The mixed method focuses on combining both types of questions using various research designs. The first method is the convergent mixed method, where the researcher collects both types of data simultaneously and assimilates the information based on their interpretations to produce overall results. The second is the explanatory sequential mixed methods, where the qualitative data is collected first, and interpretations are made based on the data. Once the researchers have formulated results from the qualitative data, those results are used as a basis of collection for quantitative data collection to achieve further understanding. The reverse of the explanatory sequence is the third method, the exploratory sequence mixed methods, where the quantitative data is collected first. Results formulated from the first phase of data collection is then used to formulate open-ended questions for qualitative data collection in the research [168].

4.5. Selecting a Research Approach

Section 4.3 explains the traditional approaches used for researches to study natural phenomenon around us. The traditional research approaches adopted by science aim to understand reality by developing

theories to explain certain phenomena. Natural science research consists of two major activities: discovery and justification [169]. The discovery is the phase when scientists are formulating theories and making claims. In the second phase the validation and justification about the claims are provided based on data collected using various methods adopted by research.

Information Technology (IT) researches, on the other hand, are concerned more with manmade phenomenon. It focuses on systems created by humans, such as network or information systems [169]. This thesis also falls under the category of IT research as it aims to understand and enhance LPWAN based IoT network security mechanisms. To achieve the research objectives, it is crucial to adopt a research approach that aligns with the research requirements and provides effective methods for the research. The traditional approaches used in the social sciences and natural sciences try to formulate a theory and understand the problems. However, the information sciences are better suited to a research methodology that tries to create a product as a solution to an existing problem; Design Science Research (DSR) introduces the concept of building a model or a product in the research as a solution [170-174].

The concept of developing a model or a framework aligns with the requirements of this thesis, where a security framework is designed to equip LPWAN IoT networks with authentication and attack detection mechanisms. Hence, DSR appears as an ideal approach to be adopted for this research.

4.6. Design Science Research Methodology

Jay F. Nunamaker et al. [170] proposed a multi-methodological approach that describes how the final product is developed using multiple activities like theory building, observation and experimentation, and how they are interdependent. Figure 4.2 shows different activities included in DSR. Each activity includes multiple tasks towards completing the final output of the research proposed in [170].

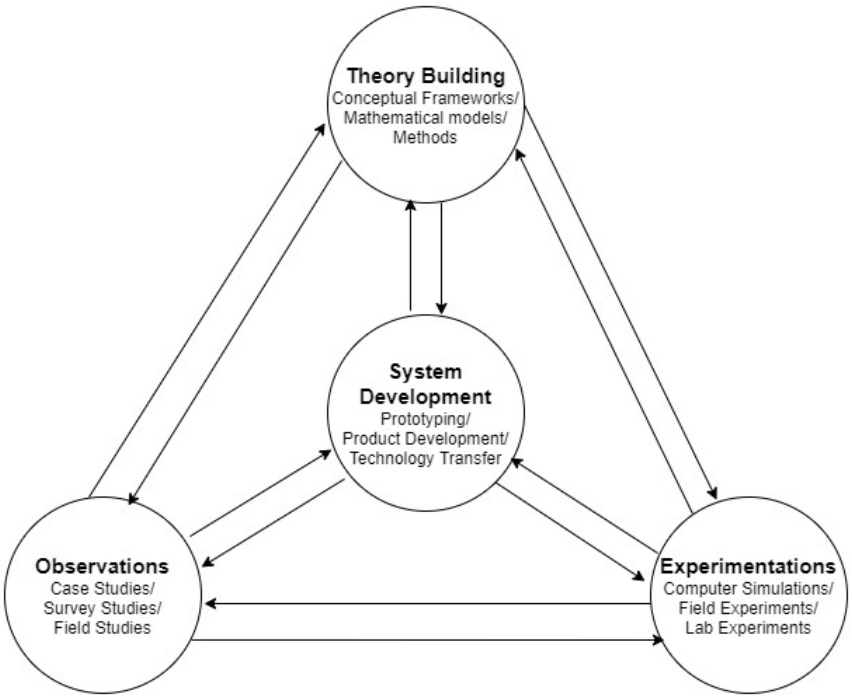

Figure 4.2: A Multi-methodological Approach for Information Science Research [162]

Figure 4.2 formulates an iterative approach that carries out several activities based on feedback from each step in the direction of system development that can be used as evidence or proof in achieving the research objective.

Theory Building: Theory building's objective is to create a knowledge base that can be utilised in formulating theories, framework conceptualisation, mathematical or simulation models. The theory building requires rigorous literature survey and analysis. The knowledge collected during the literature review serves as the foundation of the theory or it can be used for framework building. The theories, frameworks, or models built during this phase can act as an initial guiding principle of the research.

During the theory building phase of this research, a detailed study of standard communication technologies utilised in IoT applications is performed. The features and security mechanisms of communication mechanisms are analysed as discussed in chapter 2 of this thesis. Further, the literature analysis related to security mechanisms of constrained network environments such as LPWAN and Wireless Sensor Networks is carried out and discussed in chapter 3 of this thesis. The literature analysis in chapters 2 and 3 act as the knowledge base for this research and it is utilised for the problem formulation and research question design parts of this thesis. The literature review also explores the SDN framework and its applicability in IoT security which further provides insights towards developing an SDN-based security framework.

Similarly, the literature studies how machine learning mechanisms can be utilised for attack detection and how they can be embedded within the SDN framework for early attack detection. Finally, considering the insights from the literature, an initial framework for node authentication and attack

detection for LPWAN-based IoT networks using SDN is designed to address the research objectives and answering the research questions. Further, the proposed framework can be refined based on feedback from other phases of the research.

Experimentations: Experimentation consists of the research strategies used to prove the theories built in the previous phase. It consists of activities such as lab experiments and simulations. It acts as the data generator that is utilised in the observation phase. Hence, we can say it acts as a bridge between the research's observation and theory building phase. The experiment designs are guided by the theories designed and to enable the system development.

This thesis utilises multiple experimentations targeting different components of the proposed framework. Simulation, emulation and machine learning frameworks are utilised to validate the components of the framework. The detailed experiment design and experimentation tools are discussed in details later in the chapter. Based on the results received in the experimentation phase and interpretations from the results, the theories and model are refined for the efficient functioning of the framework.

Observations: In the observation phase, the researcher attempts to collect information when there is little knowledge about the subject. For observation, case studies, field studies, or sample survey can be conducted to get a general idea of the components involved in the research. It can help formulate an initial hypothesis that can be tested in an experiment or act as a base to formulate a theory for investigation. This research carries out an initial literature survey towards the IoT networks and communication technologies to identify the glitches in the existing technologies. During the initial literature survey, IoT communication technologies are lacking in providing security mechanisms to the applications, which lead to further investigations in that direction for theory building and research question formulations.

System Development: System Development sits at the centre of all the phases of the research, and every phase is carried out in the direction of system development. In general, it consists of five stages: concept design, architecture construction, prototyping, product development, and technology transfer. The concept design is to combine the theoretical and technical advancements into a possible practical application. Prototyping has used a proof of concept to validate the feasibility of the application. The research can be stopped if the application fails the feasibility test during prototyping. If the prototyping is successful, the product then goes to development, and finally, the technology transfer represents the final success for the research. The system development serves at the centre of the research that interacts with other research techniques to create a robust research platform.

A finalised security framework for authentication and attack detection on LPWAN IoT networks is produced in this thesis. The framework is conceptualised as the literature reveals the requirement of a lightweight authentication and session key mechanism for IoT networks. BYKA key agreement

provided a possible solution that can be extended to be used as a session key mechanism for LPWAN IoT networks. Based on the requirements and technical understanding, the framework was conceptualised and architected. To validate the feasibility and the correctness of the framework, it is divided into two modules. Both modules are validated separately because of their disjoint nature. The framework is refined based on the results from experimentation and finalised once performance of both the modules is optimised. The modules are programmed on simulators and emulators as prototypes and for performance validation. The final framework is presented as output with all the artefacts once each component and module is optimised by iterating over framework refinement and result in analysis from experiments conducted on simulation tools.

This thesis exercises adaptation of the multi-methodological model proposed in [170] with modifications in research activities aligning with the research aim, as shown in Figure 4.3.

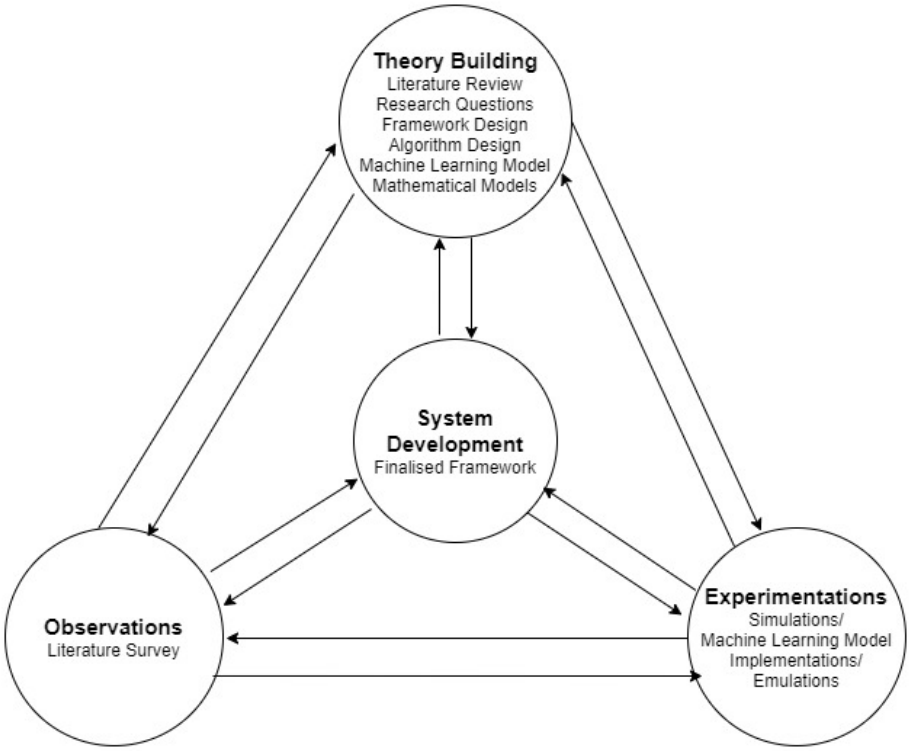

Figure 4.3: Adaptation of Multi-methodological Approach for this Research

Salvatore T. March and Gerald F. Smith [169] say that the framework for IT research lies in the interconnection of design and natural sciences. Their research includes both utility and theory to get the final research output. They proposed a framework with two dimensions one is a research activity, and the other is the research output. The framework tries to relate the design science research activities and the research output.

In [169], DRS outputs or artefacts are discussed to be of four types: construct, models, methods, and implementation, and for each output, there are four natural science research activities: build, evaluate, theorise, and justify. Every output requires at least one of these activities to be performed to achieve the outputs.

Figure 4.4 shows the relation between the activity that will be performed and the research output for this research using the framework of March & Smith [169]. The research outputs are divided into four categories construct model, method and instantiation. A different set of research activities are to be performed for different research outputs.

Research Output	Research Activities: Build	Evaluate	Theorize	Justify
Construct	Y	Y		
Model	Y	Y	Y	Y
Method	Y	Y	Y	
Instantiation		Y		Y

Figure 4.4: Relation between Research activity and research output

Figure 4.5 shows the flow of DSR activities that are to be performed to finalise the output of the current research. The initial phase targets the extensive literature review for finding research gaps in current technologies being used for LPWAN security, and a problem statement is defined. Research question formulation is followed by problem identification, questions that are going to be answered by research are formulated based on problem statements. The framework proposed aims to provide answers to the design research questions, and further experimentation will be conducted, and the results will be analysed.

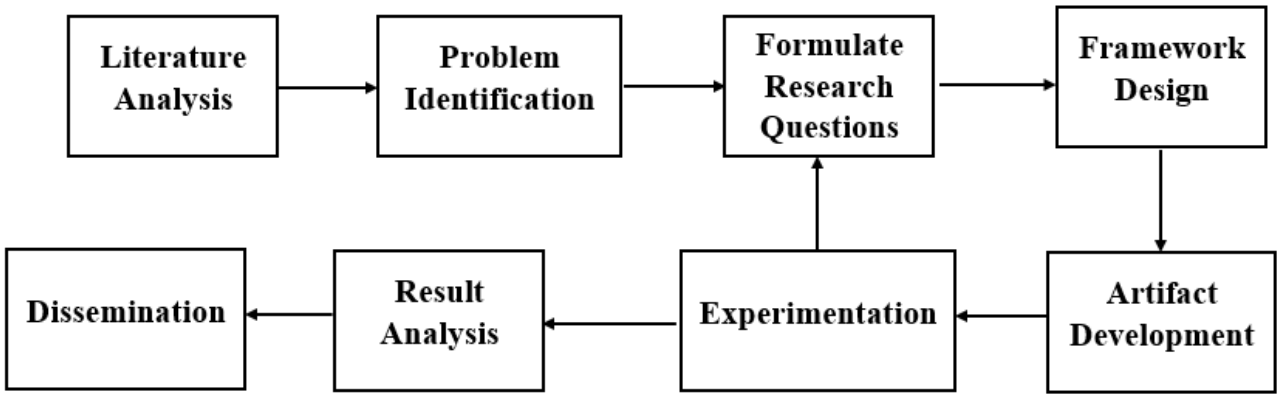

Figure 4.5: Research methodology

Chapter 5

Security Framework Design

5.1. Proposed Framework

The proposed framework incorporates the principles of DSR to address the research questions developed on the grounds of literature review. The following sub-sections describe the process of framework design, working and how it can contribute as an energy efficient, secure platform for LPWAN networks used in IoT.

5.1.1. Artefact Design

As discussed in [169], DSR produces research outputs as artefacts. Figure 4.5 shows the four general artefacts and their relationship with each other that helped in the evolution of this research. The research begins by constructing a knowledge base and designing research questions based on gaps identified in the literature review.

The literature review identifies research gaps such as inefficient modules for node authentications and session key management in current security mechanisms used by IoT devices for data transfers. IoT devices are being used in many applications for automation and monitoring in multiple areas. Some IoT applications like health monitoring might carry sensitive data that should not be forged or leaked to an unauthorised user. The problems such as lack of node authentication and session key mechanism for end nodes, energy-aware security mechanism and protection against attacks like spoofing or node identity theft were identified in current security mechanisms. At the same time, the applicability of the SDN framework is analysed based on existing literature. The literature shows that SDN can be applied as a centralised entity in networks to carry out security related tasks. Considerations on how SDN can be utilised to perform security tasks in LPWAN networks are given, and research questions are formulated based on the consideration.

The research questions developed a path for exploration that opened a variety of perspectives, research areas and responses to answer these questions. This research majorly focuses on two methodologies to answer these questions. First is the necessity of a lightweight session key mechanism where the computationally intensive operations can be shifted to resource efficient end of the network. And second is the requirement of a robust attack detection mechanism for early detection of impersonation attacks at the network layer. SDN showed promising potential in addressing both the challenges to provide security for LPWAN networks. Hence, the framework utilises the SDN controller as a central entity for

session key management and attack detection. The design and functioning of the proposed framework are discussed in the next subsection.

5.1.2. Framework Design

The proposed framework consists of six components as described below along with their functionalities:

i. End Nodes: These are the end/ terminal nodes in the network deployed in the fields. These nodes act as the data generators in the network and their numbers can tens of thousands in the network. Most of the end nodes in the network are limited with resources (processing and energy). These nodes must be authenticated before their data is entertained in the network.

ii. Gateways: Gateway nodes play a significant role in the network by acting as bridge between the end and external IP networks. Gateway nodes are also deployed in fields but are much less in number in comparison to end nodes. A single gateway node can manage hundreds of end nodes as the gateway nodes have better resources than the end nodes.

iii. SDN Enabled Switches: The Gateway nodes relay the network data to SDN enabled switches. The SDN enabled switches are different from traditional switches. These devices are programmable and support SDN communication protocols such as OpenFlow. The switches receive forwarding rules from the server based on the programs determined by administrators. The flexibility of programmability in these switches enhances packet control in the networks and makes it possible to identify various packet signatures.

iv. SDN Controller: SDN controller acts as the brain of the network as it is responsible for the behaviour of SDN enabled switches that forward the data in the network. The SDN controller performs major operations of key management and identity theft/ impersonation attack detection in the network in the proposed framework. The SDN controller is programmed with a key management mechanism and a multi-tier machine learning model for attack detection in the network. The SDN controller pushes the forwarding rules for packets in SDN enabled switches based on packet signatures, and the switch saves the rule for a period and forwards the packets accordingly.

v. Key Base: In the proposed framework, key bases act as a database for the generated keys in the network. It is directly connected to the SDN controller and stores all the key tokens that are being generated during key agreement in the network for each node in the network

vi. Application Server: Application servers are responsible for providing services to the end users. The end users connect to the application servers to fetch the information from the end nodes in the networks. The application servers are responsible for providing smart analytical results and user-friendly representations of the data which it receives from the end nodes in the network.

Figure 5.1 shows an outline of the proposed framework, including all the components as discussed above. The detailed working of the framework is explained in the next section:

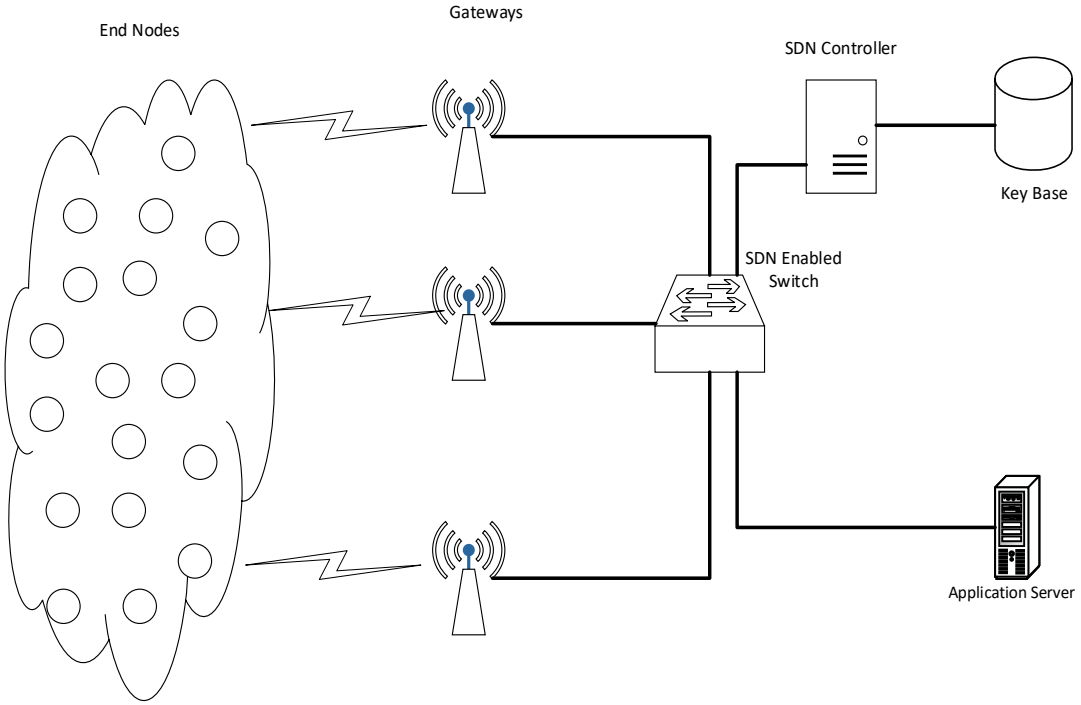

Figure 5.1: Proposed Initial Framework

5.2. Framework Functioning

Information flow across various components of the proposed IoT based system is show in Figure 5.1. The system architecture is divided into four phases: i) Key Distribution for end nodes. ii) Node activation and session key agreement. iii) Energy-aware adaptive encryption mechanisms. iv) Real-time radiometric monitoring for identity theft attack detection. Figure 5.2 illustrates the interactions among these different phases achieving the research objective.

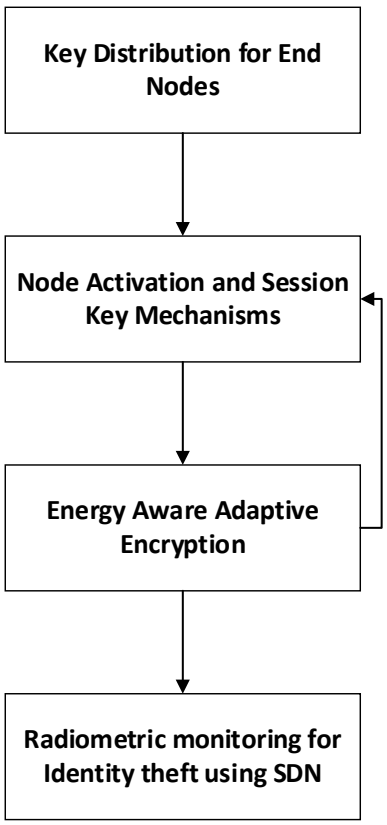

Figure 5.2: Phases in Proposed Framework

5.2.1. Key Distribution for End Nodes

Key distribution is the initial phase which is initiated before the deployment of the nodes in the fields. Each node is assigned two secret keys which is stored in the memory before deployment. The two pre-stored keys are 1) Application Key: Application keys are assigned by the application server in the network and is unique for applications in the network. The application key is used only to sign the message before sending the packets to the network. The application key is used to generate a MIC code of the data which is sent along with each data packet. The same process of MIC code generation is followed at the receiving end and compared against the MIC sent with the packet to ensure there is no tampering in the data. As the application key is unique to the applications, a node can have multiple application keys if it runs multiple applications (Note: the number of applications on one node is very limited as its constrained node environment). 2) Private Key: The private keys are assigned by the SDN controller and are unique to every node in the network. The sole purpose of the private key is to generate the session keys for the node. They are never used for either encryption or message signing. The generated session keys are unique to every data transmission session by the node and are used to encrypt the data by using AES-128 before sending it to the network gateway.

As mentioned earlier, the key distribution for end nodes is carried out before deploying nodes. It follows the procedure similar BYKA scheme [109]. In this research, the SDN controller is used as the trusted authority for a key generation securely transferred to respective nodes.

BYKA scheme uses Blom's key agreement mechanism [110] for primitive operations of key calculation. But, further enhance it by incorporating multiple public and master keys for private key generation.

There are N master keys where each key is a random (m x m) matrix, n public keys, q is the prime modulus for the public key, p is the prime modulus for other key operations. The values of N, m, n, p and q are considerably small but capable enough to provide 128-bit equivalent security.

Notations used in BYKA procedure:

S: Seed value for the public key

V: Public key, an (m X 1) column vector

R: Set of integers for the generation of pairwise key between two nodes

M: Master key, a secret symmetric (m X m) matrix stored in SDN controller

p: Prime modulus for key operation

n: Number of public keys assigned to each node

N: Number of master keys

The public keys of a node are Vandermonde matrices and which are calculated as follows:

$$V_i^T = [1, s_i, s_i^2, s_i^3 \dots s_i^{m-1}](mod q) \qquad (1)$$

s_i = Node ID + i-1, for i=1,2,…,n

The private key set S = {K_{11}…, K_{nN}} for every node is a permutation of N master keys and n public keys. In the proposed scheme, we assume that the SDN controller has a database of node IDs. The SDN controller calculates private keys for all the node IDs and securely transfers them to every node before its deployment, and is calculated as follows:

$$K_{ij} = V_i^T M_j (mod p) \qquad (2)$$

For i=1, …, n and j=1,…,N

5.2.2. Node Activation and Session key Mechanism

The proposed approach is inspired by the BYKA scheme for node authentication and activation [109]. However, it takes a different approach in session key generation by exploiting the capabilities of the SDN controller.

The existing BYKA scheme mainly focuses on secure key agreement between sensor nodes without the involvement of any third party. The end nodes utilises the pre-loaded information to generate the encryption key. The BYKA scheme targets the sensor networks with no standard topology and no centralised entity. The nodes cannot independently generate new keys as the stored information in all the nodes in the network are static. Because of which same node-pairs will generate the same key for every communication session. To implement a session key mechanism, they will require to exchange the session keys after encrypting it with the generated pairwise key using BYKA. The encryption and decryption of the session keys during the data exchange increases the number of transmissions and the number of operation for a session key exchange.

The main intent of the proposed approach is to utilise the star topology of the LPWAN networks as every node transmits data to the base station directly without using any multi-hop routing. This property of LPWAN has been exploited to randomise the generated session key. Virtual base stations are created with different IDs at the SDN controller. These virtual IDs and end node IDs are used as a pair for a key agreement. As node pairs will be different for every session, the generated key during the key agreement will also be different every time, which will create a session key mechanism without having to transfer any additional secret information over the air. Creating a virtual base station require resources like memory and processing power, so the number of virtual base stations depends on the platform where the SDN controller is deployed. Also, let us consider a scenario where a malicious entity is creating virtual base stations without the master key (only know to the SDN controller). They will not be able to decrypt the data.

Figure 5.3 shows the key agreement process in the BYKA scheme. The session key for each session is calculated using the BYKA scheme as follows:

The pairwise key set is calculated using permutations with a private key set of the node using the following process:

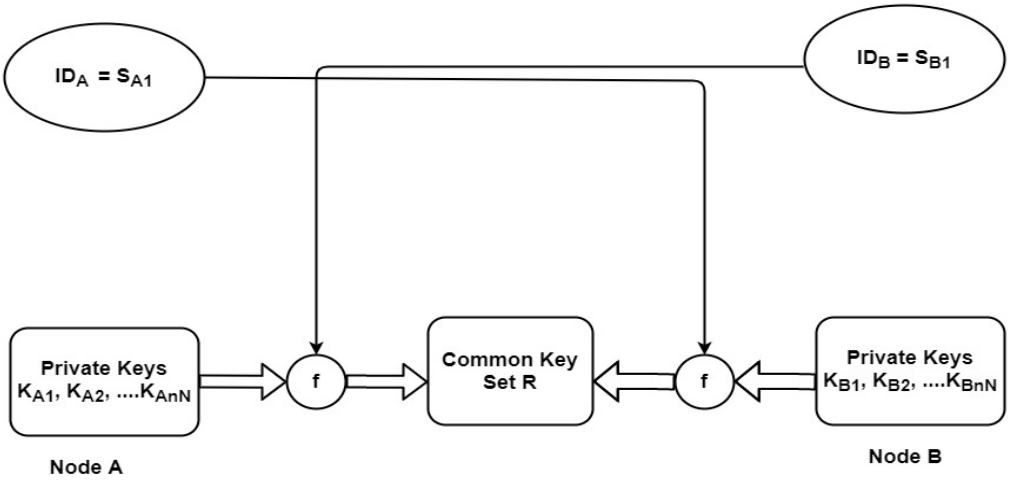

Figure 5.3: The BYKA Process [106]

For *i, k=1,..., n,* and *j=1,...., N.*

Node A:

$$s_{B_k} = ID_B + k - 1$$

$$V_{B_k}^T = [1, s_{B_k}, \dots s_{B_k}^{m-1}]$$

$$R_{A_{ijk}} = \left\{K_{A_{ij}} V_{B_k}\right\} = \{ \left(V_{A_i}^T M_j\right) V_{B_k} \}(mod\ p) \qquad (3)$$

Node B:

$$s_{A_k} = ID_A + k - 1$$

$$V_{A_k}^T = [1, s_{A_k}, \dots s_{A_k}^{m-1}]$$

$$R_{B_{ijk}} = \left\{K_{B_{ij}} V_{A_k}\right\} = \{ \left(V_{B_i}^T M_j\right) V_{A_k} \}(mod\ p) \qquad (4)$$

After calculation of all the elements of R (elements in R for both nodes are the same, but not in the same order), the pairwise key can be calculated by:

1. Multiplying them together.
2. Sorting the set elements.
3. Counting of occurrence of an integer.

5.2.3. The Session Key Extension to BYKA

The proposed strategy for LPWAN is an extension to the existing BYKA scheme that highly exploits the star structure of these networks where computationally efficient devices are on the internet-facing end of the network. The processing is offloaded from end nodes to these efficient devices to minimise

the processing at the end node. The extended BYKA scheme will be using a central key management server for all session key management operations our proposal. The server will generate all the private keys and store all the master keys using all the operations discussed in the BYKA scheme.

5.2.3.1. Public Key Data Management for Session Keys

BYKA uses node ID's to generate public keys for the nodes, and these public keys in combination with the private keys are used to generate secret keys. In the proposed extension, a mechanism to randomise the public key data is implemented. The server employs two data structures to maintain the randomness of the public key data.

The server maintains a set $\mathbf{E = \{ID_1, ID_2, ID_3, \ldots\}}$ consisting of all the nodes in the network. A different set U contains unused node ID's $\mathbf{U \cap E = \emptyset}$.

Step 1: The server chooses a random ID from U.

Step 2: The chosen random ID is broadcasted in the network for pairwise key generation using BYKA.

Step 3: The ID used earlier is removed from U and is stored in a different set of used IDs'.

A new random ID is chosen from U to broadcast for a new pairwise key generation for every session. The randomisation of ID at one end causes the change in node pair for every session. Hence, for every session, the pairwise key will be different. If a new node joins the network such that $\mathbf{U \cap E \neq \emptyset}$, that particular node ID is removed from U while maintaining the condition of $\mathbf{U \cap E = \emptyset}$**.** This condition of mutual exclusion enables the system to ensure that there is a new node pair during in session key calculation for every session, resulting in a new key every session.

As the transmission of data from the server to the node is wireless, there are probabilities of other attacks like a replay of the captured packet and a data modification attack. To avoid attacks, some additional measures are added to the key exchange mechanisms. Figure 5.4 shows the sequence of operations performed for session key generation.

5.2.3.2. Infrastructure setup

The Server (SDN Controller):

The server stores the master key M;

$\mathbf{E = \{ID_1, ID_2, ID_3, \ldots, ID_n\}}$**,** E is the set of ID's of the nodes in the network

U is a set of random IDs such that $\mathbf{U \cap E = \emptyset}$**.**

The Node:

Nodes are preloaded with and APP_Key and private keys.

The App_key is shared between server and nodes and is unique to all the applications in the networks, and the private key is calculated using the Master Key by the server.

Step 1: the server initiates the communication with a message a Random Nonce (R_D) taken from the set **U**, a timestamp (T_S). AES128 message authentication code (MAC) calculated with App_key shared by both node and the server.

$$Server\ Message = aes128\_mac[App\_key, R_D| T_S]$$

Step 2: The gateway forwards the message to the end node

Step 3: Upon receiving the message from the gateway, the node verifies the MAC to confirm the integrity of the message.

Step 4: R_D is used to calculate a public key of the sender using equation (1).

Step 5: The node uses its private key to calculate the session key S_k using equation (3).

Step 6: The node uses S_k to encrypt the data with AES128, appends the encrypted data with its ID, calculated the MAC using App_key, and sends it to the server.

$$Node\ Message = Aes128\_mac(App\_Key, E(Esk, Data),|Node\ ID)$$

Step 7: Upon receiving the message from the node, the MAC of the message is verified.

Step 8: If the integrity of the message is intact. The node ID sent by the end node is used to generate a public key using equation (1).

Step 9: The R_D sent in step 1 is used to generate a private key from the server-side by using equations (1) and (2).

Step 10: The private key generated in step 9 is used to generate session key S_k (same as node side) using equation (4). The generated S_k is used to decrypt the message from the node.

Step 11: For every session, a different value from U is picked to generate a new node pair, resulting in a new session key generation.

The operations for key calculation performed at the node and server-side are shown below:

Node A (End Node):

$$s_{B_k} = R_D + k - 1$$

$$V_{B_k}^T = [1, s_{B_k}, \dots s_{B_k}^{m-1}]$$

$$R_{A_{ijk}} = \left\{K_{A_{ij}} V_{B_k}\right\} = \left\{ \left(V_{A_i}^T M_j\right) V_{B_k} \right\} (mod\ p)$$

Node B (Server):

$$s_{A_k} = ID_A + k - 1$$

$$V_{A_k}^T = [1, s_{A_k}, \dots s_{A_k}^{m-1}]$$

$$s_{B_k} = R_D + k - 1$$

$$V_{B_k}^T = [1, s_{B_k}, \dots s_{B_k}^{m-1}]$$

$$K_{B_{ij}} = V_{B_{ki}}^T M_j (mod p)$$

$$R_{B_{ijk}} = \left\{K_{B_{ij}} V_{A_k}\right\} = \{ \left(V_{B_i}^T M_j\right) V_{A_k} \} (mod\ p)$$

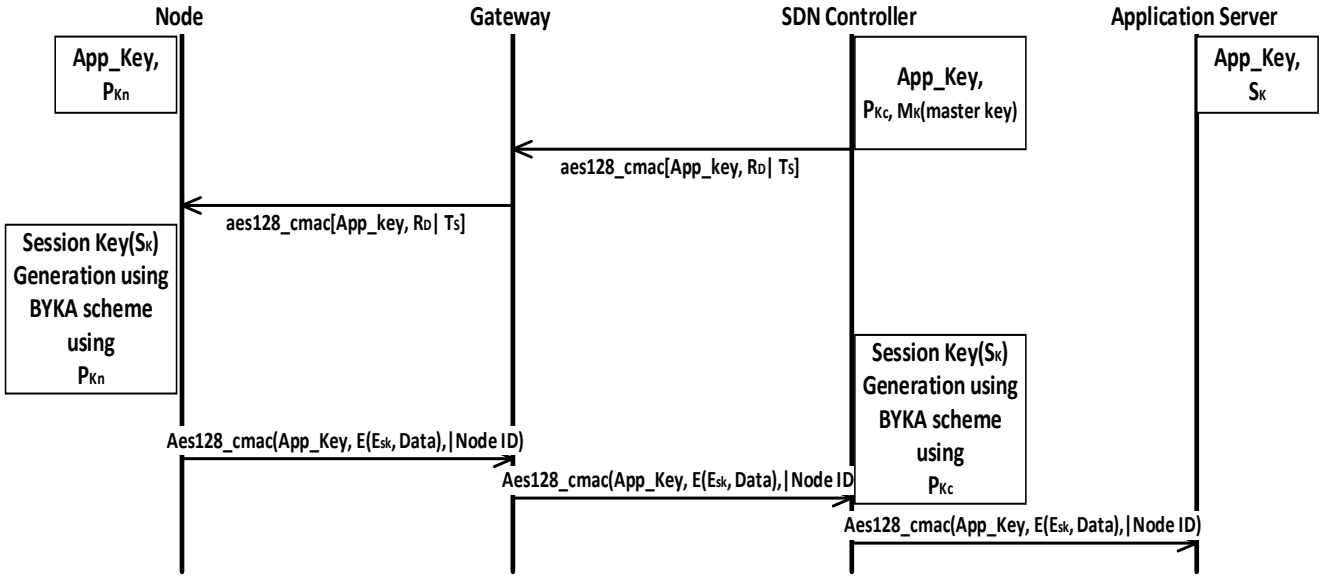

Figure 5.4: Sequence Diagram for node activation and Session Key Generation

5.2.4. Energy-Aware Adaptive Encryption

End nodes in LPWAN networks are energy constrained because of which security mechanisms must be energy efficient. As discussed before, the authentication mechanism minimises the transmissions from end nodes. Moreover, the SDN controller monitors the number of transmissions performed by nodes to estimate the energy level of the node. As the SDN controller initiates the session key mechanism, session lengths can be customised based on the applications running on the end node. With the customisation of session length for the nodes, the overhead of session key calculation is controlled, resulting in further optimisation of the node energy consumption even further.

The energy required to transmit the data is ETx, to receive the data is ERx, BYKA calculation is EBK, for AES MAC calculation is EMAC and for AES encryption is EAES.

As shown in Figure 5.4, for every transmission, the end node performs two operations before transmitting data to the server. Hence, the Energy of Operation (EOP) for each transmission can be calculated as follows:

$$\mathrm{E_{OP}} = \mathrm{SUM}(\mathrm{E}_{RX}, \mathrm{E_{BK}}, \mathrm{E_{MAC}}, \mathrm{E_{AES}}) \qquad (5)$$

In (5), it is assumed that for every transmission, a new session is created. However, if the length of the sessions is decided based on the number of transmissions *m* for the total *n* number of transmissions. Then the node energy consumption will be reduced and calculated as follows:

$$\mathrm{E_{OP}} = \sum_{i=1}^{n}(\mathrm{E}_{RXi} + \mathrm{E_{MACi}} + \mathrm{E_{AESi}}) + \sum_{j=1}^{n/m} \mathrm{E_{BKj}} \qquad (6)$$

5.2.5. Attack detection in the network using SDN

Authentication and session key mechanisms are effective ways to secure networks from unauthorised access of data. The BYKA scheme provides an effective authentication mechanism but, is vulnerable to identity theft/impersonation attacks. As the nodes are deployed in an un-monitored environment, physical attacks on these devices are likely to happen. As these devices are not physically secured, attackers can extract the secret keys from the devices are act as a legitimate node, which can cause damage to the integrity of the networks. The attacker impersonate as a network node and transmit false information to the application server without being detected because they have the keys and can act as a legal node in the network.

To overcome these limitations, in the proposed framework, a dynamic real-time attack detection mechanism is proposed to be implemented on the SDN controller to monitor incoming traffic in the network.

There are several techniques that can be used for attack detections in IoT networks. Mechanisms in [136, 139, 175] depict how radio properties can identify nodes in the networks. The proposed framework adopted a similar approach and uses node behaviour and data fields to identify normal and attack traffic. The prime focus of the attack detection mechanism is the early detection of attacks in the network before the attackers can damage the network. As a result, it is vital to deploy the attack detection mechanism close to the end nodes. The gateway nodes have more capability than the nodes but are still limited. Hence, the attack detection mechanisms should be light enough to be deployed on gateway devices. Considering all the limitations and the requirements, the attack detection is divided into two tiers. The first tier is a simple, lightweight binary machine learning classifier responsible for filtering attack from the normal traffic in the network. Tier two implements a more complex machine learning model at SDN controller to profile the attack traffic received from the gateway. The controller redirects the data to designated ports based on the type of the traffic (normal, attack) by pushing ruled into SDN enabled switch.

Two Tier Network Attack Detection: The proposed two-tier attack detection mechanism is divided into two stages; i) Early detection; ii) Attack Profiling. The machine learning model implementation is for the wireless transmissions in IoT networks. Hence, the attack detection mechanism uses features of wireless traffic to detect the attack on the network. To implement the two tier mechanism AWID [176] dataset is used to train and test the detection mechanism.

5.2.5.1. The Dataset

To implement a machine learning based solution for attack detection, an appropriate dataset is required to train the machine learning models. As the research question for this thesis focuses on impersonation attacks, a dataset with impersonation attack samples must be chosen to train the classifiers. Considering the requirement of impersonation attack samples, AWID dataset is chosen for experiments in this thesis. The AWID dataset is generated on a testbed consisting of access points, smartphones and laptops as clients using 802.11 for data transmission [176]. The dataset has two sets of data: AWID-CLS and AWID-ATK. The difference in both the datasets is the number of classes, AWID-CLS has four classes, and AWID-ATK has sixteen (16) classes.

The dataset is generated over the period of one hour, 60% of the time, normal traffic was transmitted, and 40% of the time, the network was exploited. The total dataset contains 37,817,835 records, out of which 1,085,372 are attack traffic. Most of the studies are performed on 5% of the AWID-CLS dataset, containing 1,795,575 and 575,562 in the training and testing set respectively. The samples in the dataset belong to one of the following classes: "normal", "flooding", "impersonation", and "injection". Table 5.1 shows the composition of the AWID dataset

Table 5.1: AWID-CLS Dataset class composition

Class	**Train Set Instances**	**Test Set Instances**
Normal	1,633,190	530,785
Injection	65,379	16,682
Impersonation	48,522	20,079
Flooding	48,484	16,682

5.2.5.2. Attack Profiles

As discussed in section 2.3, wireless networks are vulnerable to several attacks. However, the AWID-CLS dataset focuses on three major attacks on wireless networks as discussed below:

Flooding Attack: Flooding attacks are a type of Denial of Service (DoS) attack that abruptly increases the management frames per unit time; it aims to exhaust network resources resulting in performance degradation for legitimate users. Flooding attacks may look straightforward, but they can do serious

damage to network performance [176]. For instance, the de-authentication flooding attack is one of the most effective DoS attacks on wireless networks. However, it is very challenging to identify it accurately. The other type of flooding attack is probe flooding. In IEEE 802.11, probe request to access points are transmitted in an insecure way, because of which they can be captured and read by an adversary. In probe flooding, the captured probe requests are flooded in the network, drastically decreasing the network's throughput [177].

Injection Attack: Injection attack focuses on rushing valid encrypted packets into the network. Injection attacks include ARP injection and Fragmentation attacks. ARP attacks can be further used for password cracking of clients in the network [178]. The ARP injection attack is used to force the network to regenerate Initialisation Vectors (IV), and these newly generated IVs are captured to feed key cracking algorithms. The attacker constructs an ARP packet and sends it to an access point in the network. The access point further broadcasts the ARP packet in the network. For every ARP request, a new IV is generated that can be used by attacker for cracking the key.

Impersonation Attack: Impersonation attacks are common in wireless networks where the attackers act as a legitimate node of the network to obtain unauthorised access of the wireless network. There are various ways to launch an impersonation attack like cloning a device, rogue access point, spoofing and replay attack [179]. In various cases, attackers use the honeypot approach to launch an *evil twin* or *Rogue Access Point*. Attackers create fake access points with inviting names or with the name of an existing SSID. Once the clients connect to the fake access points, the attackers can control their system or extract information, including security keys of their original SSID.

Impersonation attacks do not affect the network performance like DoS attacks. However, they can cause a serious breach in network security by damaging node authenticity and data credibility. Furthermore, as the nodes do not show any anomalous activities in the network, it can be challenging to spot an impersonator.

The proposed approach focuses on separating the attack and normal traffic in the first tier while minimising the computational costs of the classifier. Then, in the second tier, the attacks are profiled based on their attack patterns to take suitable actions against the originators of the attack traffic. Figure 5.5 shows the outline of the proposed two-tier attack detection mechanism.

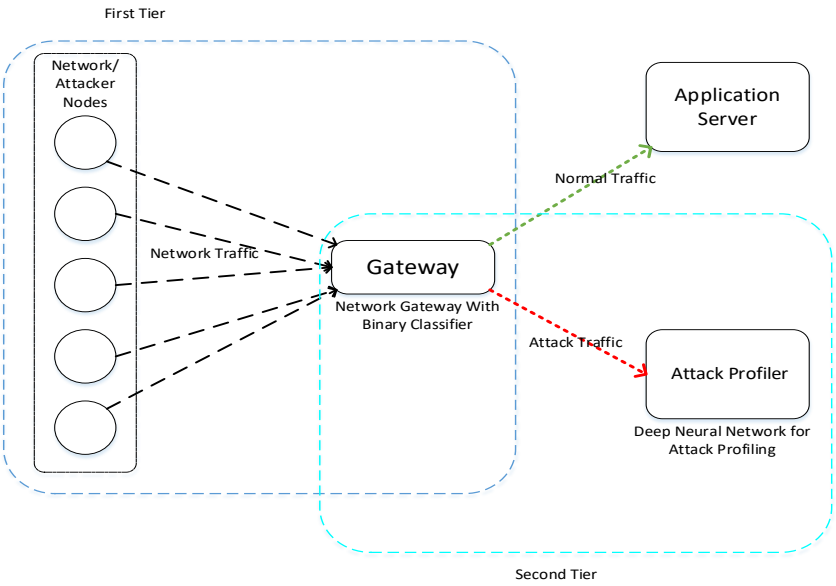

Figure 5.5: Two-Tier Network Attack detection

5.3. Features of The Extended BYKA Scheme

As mentioned in sub-section 5.2.3, the extended BYKA scheme takes advantage of the star topology of LPWAN. BYKA is originally designed for wireless sensor networks. Hence, it has to use static keys stored in the node to generate the secret key during key agreement. However, LPWAN is not restricted to such limitations. Therefore, there are several advantages of extending BYKA for LPWAN that follows a star topology.

5.3.1. Prevention Against Replay Attacks

In wireless transmission technologies, the possibilities of packet sniffing are always possible by an adversary. An attacker can use a transceiver to capture the packets sent by nodes to the gateways or vice versa. The possibility of using these captured attacks to be injected back into the network without being detected can breach the network's integrity and data authenticity. These type of attacks are called replay attacks, where old packets are replayed in the networks. Current LPWAN techniques provides various mechanisms to counter the possibility of replay attacks in the networks by using frame counters or Nonces. However, LoRaWAN is the only LPWAN technology that provides the session key mechanism when it comes to using dynamic keys for data transmission. As mentioned in sub-section 2.2.2.2, LoRaWAN uses join request to initiate the session keys from end nodes; and the end nodes receive join accept message from servers as a reply. LoRaWAN uses DevNonce to avoid the replay of join request messages. However, there is no mechanism available to avoid the replay of join accept messages [180]. The replay of join accept in the network can cause an end node to trust a fake gateway.

In the proposed session key mechanism, the sessions are initiated from the central server, and the packets consist of timestamps that identifies the packet's age. Thus, the nodes can compare the

timestamps with their clocks and decide whether the packet is fresh or being replayed by an attacker.

5.3.2. Robust Session Control

The BYKA scheme is designed for wireless sensor networks where nodes have to communicate with each other to relay the data to the gateway node. The originator of the data initiates the key exchange with whom it wants to communicate. Similarly, every node initiates the key agreement process with the successor node in the routing path as the data moves towards the gateway. However, the proposed mechanism targets the LPWANs where a gateway directly receives data from the end nodes and is aware of the node scheduling. Hence, the proposed scheme uses the gateway to initiate the sessions with the end nodes. As the central servers are aware of the applications running on the nodes in the network, the sessions can be customised based on the application's security requirements. The centralised session control provides the power to consider multiple factors based on the status of the end nodes (application, remaining energy, transmission frequency, location). The granular session control can be used to minimise or maximise the session length based node status to optimise the energy consumption of the end node.

5.3.3. Lightweight

BYKA is originally designed for sensor networks where the sensor nodes can use it for key exchange for data encryption. It uses a simple matrix-based calculation using significantly smaller prime numbers for the key calculations. The simple operations of BYKA and the reduced number of transmissions in the proposed framework by shifting the extensive calculations towards the SDN controller makes the proposed key exchange framework usable in IoT environment where constrained nodes are used for operations. The lightweight nature of the proposed framework lowers the burden of complex operations from end nodes, enabling the nodes to have a longer lifetime with better security.

5.3.4. Node Authentication

The proposed framework session key mechanism enables the automatic authentication of the end node without additional transmission. Only the nodes pre-loaded with a private key from the central server can generate the same session key as the SDN controller on receiving the public information. Providing an implicit authentication of the nodes in the network.

5.3.5. Minimal Transmission Requirements

Considering the limitations of the end nodes for limited energy availability, the proposed session key mechanism tries to shift most of the session key responsibilities towards the SDN controller.

Unlike LoRaWAN session key mechanism, the proposed mechanism makes the central SDN controller responsible for initiating the session key considering the sleep cycles of the nodes running different applications. Hence, the nodes only have to receive the session key information, generate the session keys, apply encryption, and start transferring data. The SDN controller is responsible for validating the data authenticity by using the session key generated at its end. In case the SDN controller finds any errors, it simply discards the packets. Hence, the transmissions from the end nodes are minimal and can maximise the lifetime of the nodes while using the session key mechanism.

5.3.6. Key Localisation

Using a single secret key for all the nodes in the network can cause severe damage to the network in case even a single node gets compromised. The proposed mechanism uses the BYKA scheme ensuring that all the nodes get a distinct private key to generate the session key upon receiving information from the server. If a node's private key gets compromised in a physical attack on the nodes, only that node gets affected. The attackers can use the compromised node to impersonate as a legal node in the network and send false information to the server. Considering this situation, the proposed mechanism incorporates machine learning attack detection as a second line of defence deployed on an SDN controller. It filters the impersonator's data from entering the internet facing segment of the network.

5.4. Trade-offs for Proposed Security Framework

The proposed framework aims to provide a dynamic session key and attack detection mechanism for LPWAN based IoT networks. However, adding security mechanisms to existing technologies that do not provide such features can have several trade-offs for adding new features.

i) *Additional Transceiver Overhead:* The proposed session key mechanism aims to minimise the number of transceiver operations on the end nodes. As most of the existing LPWAN protocols do not provide a session key mechanism, they do not have to do any additional data transmission and reception for a key generation or key transfer operations, In other words, they can directly encrypt and send data by using the pre-stored key in the end node. However, the proposed session key mechanism requires the node to do some transceiver operations to receive the public key information from the server. Transceiver operations are the most intense operations concerning high energy requirements and can shorten the node lifetime depending on frequency of such operations. Hence, the implementation of proposed session key mechanisms will have certain effects on the node lifetime. However, by regulating the frequency of the session key operations, the transceiver operations can be moderated to not exert the end node excessively. A balanced trade-off between the number of transceiver operations and security is maintained

by analysing the security requirements of the applications running on the end nodes and regulating the sessions of the applications accordingly from the SDN controller. Further, the SDN controller can also monitor the number of transmissions made by a node and estimate the remaining energy of the nodes. Based on the remaining energy of the end nodes, the SDN controller can regulate the sessions to extend the node lifetime in the network.

ii) *Continuous Listening at End Node:* In constrained network environments such as IoT, nodes have to last for a long period of time on battery power. In some applications, the nodes are deployed in remote locations, so it is challenging to recharge the batteries frequently. Considering the adverse working conditions, the end nodes are designed to minimise the energy consumption and only perform operations when required. The end nodes mostly stay in sleep mode to save energy and wake up when they are required to either receive or transmit data. In the current LPWAN technologies, the end nodes are mostly involved in sending data. The end nodes do not receive any data from server side except in LoRaWAN. Where LoRaWAN sends a reply from the server during OTAA node activation. However, in the proposed session key mechanism, the end nodes have to receive public key information from the SDN controller to generate the session keys for encrypting data before transmissions. The proposed approach will require the nodes to continuously listen to the channel to receive the SDN controller data. In order to listen to the channel, the nodes have to stay in an awake state for a longer period of time than usual, causing the transceiver to draw extra energy from the battery. The longer awake cycle will eventually cause the battery to drain quicker than usual, shortening the operational period of the node in the network.

iii) *Processing Overhead at End Node:* The session key mechanism will require the end nodes to calculate the session keys using the public key which they receive from the SDN controller. In the standard protocols, end nodes do not perform session key operations or utmost perform only when nodes join the network. Hence, there is no overhead of session key operations. As we add the functionality of session key mechanism, the nodes will have to do session key operations frequently, causing them to use the battery power. The session key operations can be regulated by managing the session lengths and frequency based on the security requirements of the applications running on the end nodes.

iv) *Increased Payload Size:* The proposed session key mechanism requires the end nodes to send some additional information to the SDN controller used to calculate the session key at the server end. This requires the nodes to send more information than the standard protocols in LPWAN. Sending more data means that the transmitter needs to be functional for a long period for each transmission, leading it to draw more energy and resulting in a shorter lifetime of the nodes.

v) *Additional Delay:* The proposed framework targets a secure transmission and prevention from possible attacks on the network. To achieve the secure transmission, the session keys are introduced so that the encryption keys can be updated frequently and attackers cannot collect

enough information to perform cryptanalysis. Attack prevention is introduced by implementation a machine learning based attack detection system at SDN controller to filter the malicious traffic from network. For session keys, the nodes need to perform additional operations that introduce operational delay at end nodes. Moreover, the attack detection parses every incoming packet as input to the attack detection system. The filtering process also adds delay in the transmission of the data packets to the application server. Hence, adding the proposed security framework may cause additional delays or packets to reach the destination. However, the proposed framework performs the traffic filtering at the network's gateway on the network layer. The machine learning algorithm utilises the mac layer information and uses a multi-tier approach to minimise the delay in packet delivery.

Chapter 6

Experimentation and Results Analysis

In the previous chapters, we discussed the specifics of this research, the tools and techniques to be used for the implementation of the proposed framework. As mentioned in Chapter 5, this research is divided into four phases. The first three phases implement the session key mechanism of the framework and the fourth phase deals with solving the classification problem into attack traffic and normal traffic.

In this chapter, the experimentation is carried out to verify the working and impact of the proposed security framework. The experiments are conducted separately on both the module to validate and verify the impact of session key exchange mechanism and attack detection framework.

6.1. Simulations Tools for Experimentation

The proposed framework has mainly two modules coupled together; i) Node Authentication and Key Exchange; ii) Identity Theft (Impersonation Attack) Detection. Multiple tools are available to validate the correctness and effectiveness of each module implemented in the proposed architecture. Various available tools are studied to understand their applicability at various level. After analysing the use cases of these tools, three simulation/emulation tools were identified to be used for experimentation. Following are the tools analysed for wireless network simulation:

Network Simulators:

i) Network Simulator (NS) 2: NS2 [181] is one of the most popular open source discrete event simulators for wireless networks. NS2 support several network protocols for wireless networks and energy models. For Wireless Sensor Networks (WSN), it provides 802.11 and 802.15.4 mac protocols. NS2 is written in C++ and OTCL languages. NS2 libraries are in C++, while OTCL is used to control the simulation environment. For visualisation of networks, NS2 provide Network AniMator (NAM). While NS2 provides an accurate simulation environment for wireless networks, it does not have the capability of running real-time Operating System (OS) command executions.

ii) Omnet ++: Omnet ++ [182] is another discrete event simulator available for wireless networks. It supports standard wireless and wired IP networks along with various WSN protocols. Like NS2, it also uses C++ for simulation modelling. In addition, it also provides TinyOS simulation support. However, Omnet++ lacks the support for sophisticated energy models and mac layer protocols required in WSN.

iii) Riverbed Modeller (OPNET): OPNET [183] is a commercially available network simulation tool created originally for military purpose but later was highly accepted as commercial network modelling tool. It was designed to support static networks because of which it provide extensive and accurate support for fixed network hardware and protocols. The higher versions of OPNET also started providing supports for wireless protocols such as 802.15.4 used in WSN. In addition to the matured library for simulation modelling, it also provides an interactive user interface to create simulation environments.

iv) Cooja Simulator: Cooja simulator [184] is a Java based network simulator for WSN. It provides a capable platform to simulate large WSN. It provides an interactive user interface with underlying support for wireless communication protocols for WSN. In addition, Cooja also provides the capability to integrate with external tools. Cooja is designed for the Contiki Operating System (OS) and provides all the features to emulate a mote with Contiki OS.

v) Common Open Research Emulator (CORE): CORE [185] is a platform for network emulation. It provides the features to emulate PC, wireless hosts, routers, and links. CORE emulations are exactly like live networks, and it also provides the feature to connect the emulation network with external networks. In addition, the CORE provides support for wireless networks with 802.11. However, it lacks the availability of an energy model and other WSN mac layer support.

vi) NS3: NS3 [186] is another open source discrete event simulator and emulator for the wireless network built in C++ with optional Python scripting. NS3 provides support both for IP and non-IP based networks and provides the capability to connect NS3 emulated network with an external network like CORE. NS3 also provides support for OpenFlow, an SDN protocol, making it possible to implement SDN networks in NS3.

vii) Mininet-WiFi: Mininet-WiFi [187] is another lightweight emulator that uses wireless networks supporting 802.11 transmission for nodes in the network. It uses virtualised Linux kernel to act as network devices like PC, routers, switch, and access points. It provides extensive support for SDN with multiple controller integration like PoX, Ryu and Open Floodlight. It also provides running Linux command on every virtualised node in the network, making it possible to run custom scripts on every network node.

Table 6.1: Comparison of Wireless Simulation/Emulation Tools

Simulation/Emulation Tool	Wireless Support	Real-time OS commands	SDN Support	Energy Model Support
NS2	802.11, 802.15.4	No	No	Yes
Omnet++	802.11	No	No	Yes
OPNET	802.11, 802.15.4	No	No	Yes
Cooja	802.11, 802.15.4	No	No	Yes
CORE	802.11	Yes	Yes	No
NS3	802.11, 802.15.4, 6LowPAN, LoRaWAN	No	Yes	Yes
Mininet-Wifi	802.11	Yes	Yes	No

Table 6.1 summarises wireless simulation tools based on features required for experimentations in this research. Several tools are providing various features for wireless network simulations. Most of the simulation tools provide simulations for wireless protocols such as 802.11, 802.15. However, NS3 is the only tool providing the simulations of LPWAN protocol LoRaWAN. SDN framework implementation being one of the research the research focuses, NS3, Mininet-WiFi, and CORE are fitting tools that provide the SDN support. In order to meet the research objective of having energy efficient key session mechanism, the availability of the energy model in LPWAN protocols is given high considerations in choosing the tool. NS3 is the only tools that provide LPWAN protocols LoRaWAN support and an energy model for it. Hence, NS3 is chosen for the validation of energy consumption of the proposed session key mechanism.

Furthermore, as it is also required to test the correctness of the key exchange protocol on SDN framework, Mininet-WiFi is chosen because it seamlessly integrates with SDN protocol OpenFlow and supports various SDN controllers. It emulates wireless nodes as Linux kernel and provides functionality to execute high level programming. Python is used on the wireless nodes in Mininet-WiFi for implementing the session key mechanism to test the correctness of the algorithm.

6.1.1. Security Protocol Analysis Tools

i) Failures-Divergence Refinement (FDR)/Casper: Casper [188] is a compiler to analyse security protocols. It takes the flow description of a security protocol as input. The input is then converted into Content Security Policy (CSP) description and the description is further verified with FDR3 [189]. It can be used to identify loopholes in a security protocol or to show that no attacks are possible

ii) Automated Validation of Internet Security Protocols and Applications (AVISPA): AVISPA [190] is an automatic validation tool for network security algorithms. The protocol specification is written in High−Level Protocol Specification Language (HLPSL) converted into Intermediate Format (IF) by HSPL2IF. The converted specification is verified against four checkers for the correctness of flow and attacks. The final outcome is provided about possible loopholes and attacks possible on the protocol.

iii) Scyther: Scyther [191] is a formal analysis tool for security protocols. Scyther takes protocol description in Security Protocol Description Language (.spdl). It is assumed that cryptographic functions are perfect, and the adversary learns nothing from encrypted messages unless they have a decryption key. This tool is used to find vulnerabilities in the models used for investigating security protocols. The protocol is formulated in the form of "roles" and "events". The "roles" are designed based on the knowledge and operations performed by an entity in the protocol model. "Event" describes the events of sending and receiving of the data. Based on the operations, the roles make claims regarding secrecy, integrity, and authenticity. These claims are verified over several threat models defined in Scyther. Based on the analysis, possible attacks are shown on the security protocol model.

The security protocol analysis tools are utilised to perform the semantic testing of the algorithm. It tests the data flow of the algorithms against various attacks and highlights if any attack is possible on the sequence along with the process of how that attack would take place. There are three well-known tools listed that are used to perform semantic analysis of security protocols. All of them utilise different input languages to frame a protocol and to run attack models on them. This research utilises Scyther [191] as a security analysis tool because of its diverse attack models and intuitive input mechanism. The security protocol mechanism uses roles of different entities in security protocol, making it easier to simulate the roles of server and end nodes. The events are used to represent the data exchange. Scyther provides easy to uses graphical user interface that provides detailed results and traces run by the tool to validate the security protocol.

6.1.2. Machine Learning Tools and Techniques for Attack Detection

The correctness of the first module, the session key mechanism, is validated in network simulators and security analysis tools. The second module of the framework deals with the Identity Theft attack (or Impersonation attack) detection in the network. The attack detection mechanism incorporates a machine learning approach to learn from an existing dataset and use the learned parameters to classify the incoming traffic in the network based on the traffic attributes.

Many tools and frameworks are available to implement machine learning-based classifiers, such as Konstanz Information Miner (KNIME), Weka, RapidMiner, Apache Machine Learning Library

(MLLIB), and Orange. These tools provide easy to use user interfaces for fast and easy implementations of machine learning algorithms.

i) *Weka [192, 193]:* Weka is a Java based tool that provides a graphical user interface to model machine learning algorithms. It provides extensive data exploration and manipulation features that facilitates understanding and data pre-processing. Weka is designed to take input from multiple sources ranging from simple CSV files to complex SQL queries to a database. It provides all the classes of machine learning algorithms such as classification, regression, and clustering. One of the major advantages of the Weka tool is that it is an open source and can be maintained easily by the contribution of researchers. Secondly, it includes almost all the standard machine learning algorithms that let the researchers conduct various experiments using different algorithms and choose the best suitable for their use case without writing complex codes for testing purposes. The third advantage is the java implementation that enables the platform independence and lets it run on any operating system. However, despite being a handy tool for the quick implementation of machine learning algorithms, several limitations of this tool can sometimes create challenges in implementing machine learning techniques. The first disadvantage is that all the data being processed by the tool is held in main memory, creating a limitation on the dataset size used in Weka. Weka can be very useful as the first step of machine learning protocol testing. However, it will fail to be useful in scenarios where the machine learning algorithms are highly customised from the standard ones.

ii) *KNIME [194]:* It is an environment that provides an interface for the interactive execution of a machine learning pipeline. It provides a graph like structure to run various flows of different steps on a machine learning algorithm. The users can utilise a Graphical User Interface (GUI) to assemble different data pre-processing steps, machine learning algorithms, and result calculations as part of a pipeline. These pipelines can be executed to further explore the results. The tools enable a quicks deployment of a data pipeline to test several pre-processing and machine learning algorithms intuitively and easily. Each node in the flow of KNIME performs an operation it receives from the previous step in the dataflow. The connectors in the dataflow show the order of the operations being performed on the dataset. Moreover, KNIME provides an integration with Weka and R statistics software to enhance the data analysis experience of the users. It also provides the feature of exporting the pipeline and Predictive Model Markup Language (PMML) file and can be used in R, SAS and other tools that support PMML files. However, in the current version, the pre-processing steps are not included in the PMML file.

iii) *Rapid Miner [195]:* Rapid Miner is a commercial product and provides similar platforms like KNIME. Rapid Miner provides a user-friendly platform for flow design to implement and analyse performances of machine learning algorithms on the dataset. The data pipeline can be created by adding steps and connecting them in the desired order for processing. The output from one block

is passed as an input to the immediate next block. It provides an easy drag and drops feature to add and remove pipeline steps that enable users to create flows for data processing easily. In addition to the easy and quick pipeline deployment, it provides comprehensive result analysis tools with various filters and visuals.

iv) *Apache Machine Learning Library (MLlib) [196]:* MLlib is a distributed machine learning library provided by Apache Spark that can handle huge amount of data. As Spark is designed to handle huge amounts of data, MLlib being designed has given huge consideration to iterative computations required while handling huge datasets. The open-source community of Spark keep up with the high increase in speed and complexity of ML models by continuous improvement in their libraries. MLlib provides powerful features for machine learning development such as: i) Methods and Utilities: It provides implementations of standard machine learning algorithms like linear models, naïve Bayes, ensemble methods for classifications and regression. It also provides implementations of unsupervised learning algorithms like K-Means clustering and Principal Component Analysis (PCA). It offers various utilities like convex optimisation techniques, feature extraction methods and tools for statistical analysis; ii) Optimisation: MLlib provide mechanisms to optimise distributed systems for distributed learning and predictions. It implements an efficient map and reduces algorithms for efficient distribution of data over worker nodes for calculations while maintaining the integrity of the algorithms. iii) Pipeline API: To implement a successful machine model, preparing high quality of data is imperative which involves various steps like data exploration, data cleaning, feature engineering, data quality and finally preparing the training-testing datasets. MLlib provides pipeline API to create a sequence of these steps that can be performed on the dataset to train a model and make predictions. iv) Extensive Documentations: Spark provides a detailed documentation of all the features provides by MLlib for easy understanding and implementation.

v) *Scikit-Learn [79]:* Scikit-Learn is a python based machine learning library specially designed to for machine learning research. It has extensive and full support for Python language as Python is widely used for scientific computing. Scikit-learn utilises the versatile and rich environment of python programming to implement state of the art machine learning algorithms. Scikit-Learn utilises python libraries to perform various operations related to machine learning implementations. Python library NumPy is used to lay down the underlying data structure. The Input data is represented as NumPy arrays. Scipy, another Python library, is used for sparse matrix representation, statistical functions, and linear algebra operations. Finally, Cython, a language for including C in Python, is utilised to introduce precompiled code as a boilerplate for faster execution of machine learning algorithms. Scikit-Learn provides a huge collection of machine learning implementations while providing the flexibility to choose from the available optimisation techniques. As it uses Python as the programming language for implementations, it provided integrations into advanced framework such as TenserFlow, Karas, and Pytorch where highly

customised machine learning models can be designed using Scikit-Learn building block. In addition to implementations of state-of-the-art machine learning algorithms, it provides a rich collection of data pre-processing packages that are used for data transformation suitable for machine learning algorithms. The toolkit provides features such as data scaling, normalisation, data encoding, and imputation of missing values. Other than out of the box transformers, it also facilitates custom transformer implementations id required for effective data pre-processing for machine learning algorithms sensitive to data variance. Being an open-source library, it is easily accessible and regularly updated by the machine learning community, keeping it up to date with the latest machine learning algorithms. Furthermore, the detailed documentation of the library makes it easier to understand for quick use of sophisticated machine learning algorithms.

Considering all the above mentioned machine learning tools. There are various advantages of every tool in various aspects. Some of them provide interactive user interfaces for easy and quick prototyping of machine learning models. Tools like MLLIB show great potential in handling a huge amount of data and create highly custom machine learning models. Scikit-Learn provides the capability of building custom machine learning models with extensive support of Python for data pre-processing.

In this research, a two-tier attack detection framework with a consolidated voting mechanism implemented that makes predictions by summing the results from various binary classifiers is aimed to be tested on a wireless network dataset. Taking into consideration the structure of data, and the problem to be solved, this research adopts a programmatic approach to implement the machine learning classifier for attack detection in the network by using Python programming language.

Python is one of the most popular and convenient programming languages available to implement machine learning-based solutions. Python provides a huge range of libraries such as Scikit-Learn, Pytorch, TensorFlow, and Karas to implement complex machine learning and deep learning architectures. Before implementing any machine learning classifier, the pre-processing of the data must be done for the machine learning classifier to perform effectively. Python provides libraries such as NumPy, Pandas for data cleaning and wrangling. In addition, Python provides tools like “imblearn” to handle data imbalance in the dataset. SHAP is available in Python to explain and visualise the learning process of the machine learning algorithms. Also, matplotlib in python provides a flexible platform to visualise results produced by machine learning classifiers.

All the steps beginning from data cleaning till explaining the machine learning models and interpreting the results which form the basis for the classification are described in following subsection.

6.1.3. Machine Learning based Attack Detection Experimentation Process

Data pre-processing is considered one of the most important aspects of any machine learning based project. As it increases the quality of data which has a great impact on the prediction accuracy of a

classifier [197]. In this section, we will discuss about the various data manipulation / pre-processing steps applied on AWID-CLS dataset as shown in Figure 6.1.

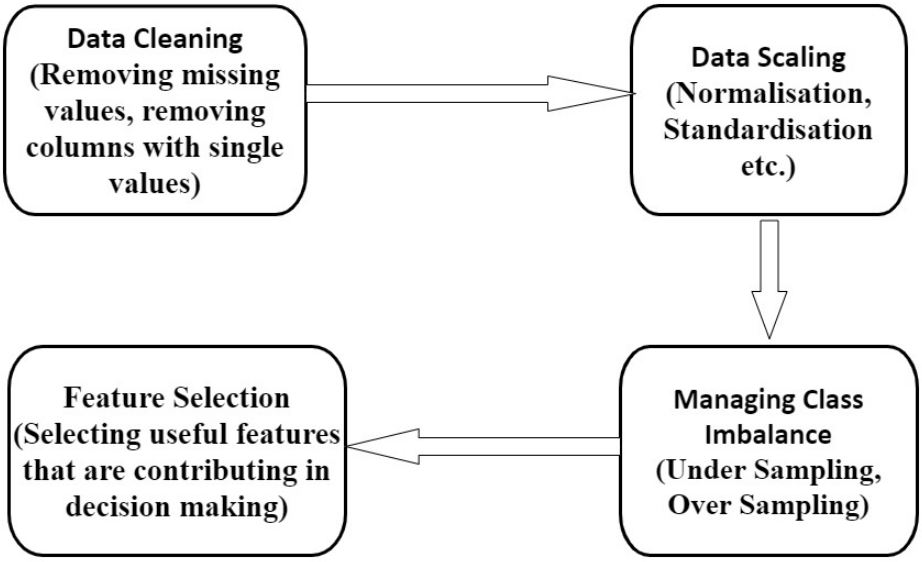

Figure 6.1: Data Pre-processing Steps

6.1.3.1. Data Cleaning

Data in the real world are mostly unclean and heterogeneous in nature. The AWID dataset was created in a lab environment, and a separate device was used in monitor mode with a T-shark to capture the network traffic. Many attributes of T-shark captures did not apply to packets in the monitored network, so there were plenty of missing values that were replaced by "?" in the dataset. As the dataset included lots of missing values, data cleaning was a very crucial step here. Ignorance in handling the missing data can lead to wrong predictions or classifications and can also cause high bias for the model being used. The dataset consist of 154 attributes with 1795575 records. Figure 6.2 shows the snapshot of the original AWID-CLS dataset before data cleaning. There were 72 attributes with more than 50% of missing values. Due to high proportion of missing values, these attributes were dropped off. If we do not remove these missing values, they cause various issues. Firstly, we already have less information about these attributes and secondly if not handled properly, they have capacity to devour the true data. As this dataset would be propagated further in many iterations, the adverse impact would be huge. Next, three attributes amongst others were converted from hexadecimal to integer values. Some attributes such as feature 0 and 2 had no distinct values and even they were excluded as such data is incapable of contributing towards better decision making. In summary, with an objective of increasing the quality of data, we ended up using 23 out of 154 attributes for further steps.

	0	1	2	3	4	5	6	7	8	9	...	145	146	147	148	149	150	151	152	153	154
0	0	?	0.0	1.393661e+09	0.000000	0.000000	0.000000	261	261	0	...	?	?	?	?	?	?	?	?	?	normal
1	0	?	0.0	1.393661e+09	0.024271	0.024271	0.024271	185	185	0	...	?	?	?	?	?	?	?	?	?	normal
2	0	?	0.0	1.393661e+09	0.001631	0.001631	0.025902	185	185	0	...	?	?	?	?	?	?	?	?	?	normal
3	0	?	0.0	1.393661e+09	0.055325	0.055325	0.081227	159	159	0	...	?	?	?	?	?	?	?	?	?	normal
4	0	?	0.0	1.393661e+09	0.000415	0.000415	0.081642	54	54	0	...	?	?	?	?	?	?	?	?	?	normal
5	0	?	0.0	1.393661e+09	0.000005	0.000005	0.081647	40	40	0	...	?	?	?	?	?	?	?	?	?	normal
6	0	?	0.0	1.393661e+09	0.016692	0.016692	0.098339	261	261	0	...	?	?	?	?	?	?	?	?	?	normal
7	0	?	0.0	1.393661e+09	0.000142	0.000142	0.098481	40	40	0	...	?	?	?	?	?	?	?	?	?	normal
8	0	?	0.0	1.393661e+09	0.028067	0.028067	0.126548	185	185	0	...	?	?	?	?	?	?	?	?	?	normal
9	0	?	0.0	1.393661e+09	0.001801	0.001801	0.128349	185	185	0	...	?	?	?	?	?	?	?	?	?	normal

Figure 6.2: AWID-CLS Dataset Snapshot

6.1.3.2. Data Scaling

Data or Feature Scaling is the next important step in machine learning modelling. Machine learning algorithms only understands numeric values – if few features are in the range of thousands, few other in tens, then machine learning model gives more weightage to high ranging numbers. On analysing we realised that our dataset contains outliers which stands true as it a dataset with network-based attack information. Applying feature scaling techniques based on mean and variance would not consider the effect of these outliers. As a result, we chose to apply Robust Scaler. Robust scaler preserves the outliers in the dataset, hence is very effective for our case [79].

The Robust Scaler uses an inter-quartile range between first quartile (Q1) and third quartile (Q3) to scale the data using the following formula where x_i is a sample of the dataset:

$$x_i = \frac{(x_i - \mathrm{Q}_1(x))}{(\mathrm{Q}_3(x))} - Q_1(x)$$

6.1.3.3. Managing Dataset Imbalance

As shown in Table 5.1, the composition of the dataset is highly imbalanced with 1,633,190 samples for "Normal" class, 65,397 samples for "Injection", 48,522 samples for "Impersonation" and 48,484 samples for "Flooding" class. The number of records with Normal traffic are much higher as compared to the number of the records with attack instances. Therefore, training a machine learning model on an imbalanced dataset can reduce the precision of the model [198]. Oversampling and undersampling are the highly used strategies to deal with imbalanced datasets.

Oversampling is a method in which new samples are generated for the minority class of the dataset. There are several techniques like Random over-sampling, SMOTE [199] and ADASYN [200]. Random over-sampling simply duplicates the instances of the minority class in the dataset. On the other hand, SMOTE and ADASYN generate data points with K-Nearest neighbours between two existing data points. In addition to the mentioned oversampling techniques, data augmentation using neural networks like Generative Adversarial Networks (GANs) [201] are gaining popularity on image datasets. In

GANs, two neural networks compete against each other to learn the data distribution. Later, the learned distribution is used to generate synthetic data to balance the dataset.

Undersampling works on the contrary of oversampling. The instances of majority class are trimmed down to level with the minority class. Thus, removing the samples with majority from the training set reduces the skew present in the class distribution.

In AWID-CLS dataset, the number of "normal" data samples is excessive compared to all the attack samples combined. The biased sample size can cause the trained model to overfit. Two standard techniques to manage the data imbalance are discussed. In case of oversampling, the samples of the attack classes are to be generated from the sample distribution of the attack classes. However, synthesising the samples can cause the change in properties of the attacks. With this consideration, we decided to use under-sampling to manage the imbalanced dataset. The "normal" samples are removed randomly to match the number of "attack" packets using the undersampling technique. Although undersampling reduces the number of samples in the dataset, it maintains the originality of the attribute values, thus increasing the confidence in machine learning model results as they were built on true data.

6.1.3.4. Feature Selection

After the dataset cleaning process, we decided to use 23 features shown in Table 6.2 for training the machine learning model as mentioned earlier. Other features were excluded as they could be a noise to the machine learning model. It is not ensured that even these 23 features are the best in distinguishing the attack traffic from the normal one. Hence, the next important step is feature selection technique. The aim is to provide rich knowledge to the model with minimal features while training for building simple models and making faster classification decisions. For example, amongst these 23 features, there would be features which are high correlated to each other. Such correlated features cannot improve models and sometimes may cause instability.

The Random Forest uses Gini indexing to calculate the information gain while using each feature in the process of decision making. Figure 6.3 shows the feature importance in the current dataset.

Table 6.2: AWID-CLS dataset features after data cleaning

Feature Number	**Dataset Feature**	**Feature Name**	**Feature Description**	**Feature Value Data Type**
1	3	frame.time_epoch	Time epoch of packet arrival.	Time offset
2	4	frame.time_delta	Time delta from previous captured frame.	Time offset

Feature Number	Dataset Feature	Feature Name	Feature Description	Feature Value Data Type
3	5	frame.time_delta_displayed	Time delta from previous displayed frame.	Time offset
4	6	frame.time_relative	Time from the reference of the first frame.	Time offset
5	7	frame.len	Frame length of the captured packet.	Unsigned Integer
6	8	frame.cap_len	Frame length stored I captured file.	Unsigned Integer
7	37	radiotap.mactime	MAC timestamp	Unsigned Integer
8	46	radiotap.datarate	Data rate (Mb/s)	Floating Point (single decimal)
9	47	radiotap.channel.freq	Channel Frequency	Unsigned Integer
10	49	radiotap.channel.type.cck	Complementary Code Keying flag.	Boolean
11	50	radiotap.channel.type.ofdm	Orthogonal Frequency-Division Multiplexing (OFDM) flag.	Boolean
12	60	radiotap.dbm_antsignal	Antenna Signal Strength	Signed Integer
13	63	wlan.fc.type_subtype	WLAN frame control type subtype	Unsigned Integer
14	65	wlan.fc.type	WLAN frame control type	Unsigned Integer
15	66	wlan.fc.subtype	WLAN frame control sub type	Unsigned Integer
16	67	wlan.fc.ds	Frame Control Distribution System indicator	Unsigned Integer
17	68	wlan.fc.frag	Frame control more fragment flag	Boolean
18	69	wlan.fc.retry	Retry flag	Boolean

Feature Number	Dataset Feature	Feature Name	Feature Description	Feature Value Data Type
19	70	wlan.fc.pwrmgt	Power management flag	Boolean
20	71	wlan.fc.moredata	Mored data flag	Boolean
21	72	wlan.fc.protected	Protected flag	Boolean
22	74	wlan.duration	Transmission duration	Unsigned Integer
23	75	wlan.ra	Receiver's address	Ethernet or other MAC address (Hexadecimal)

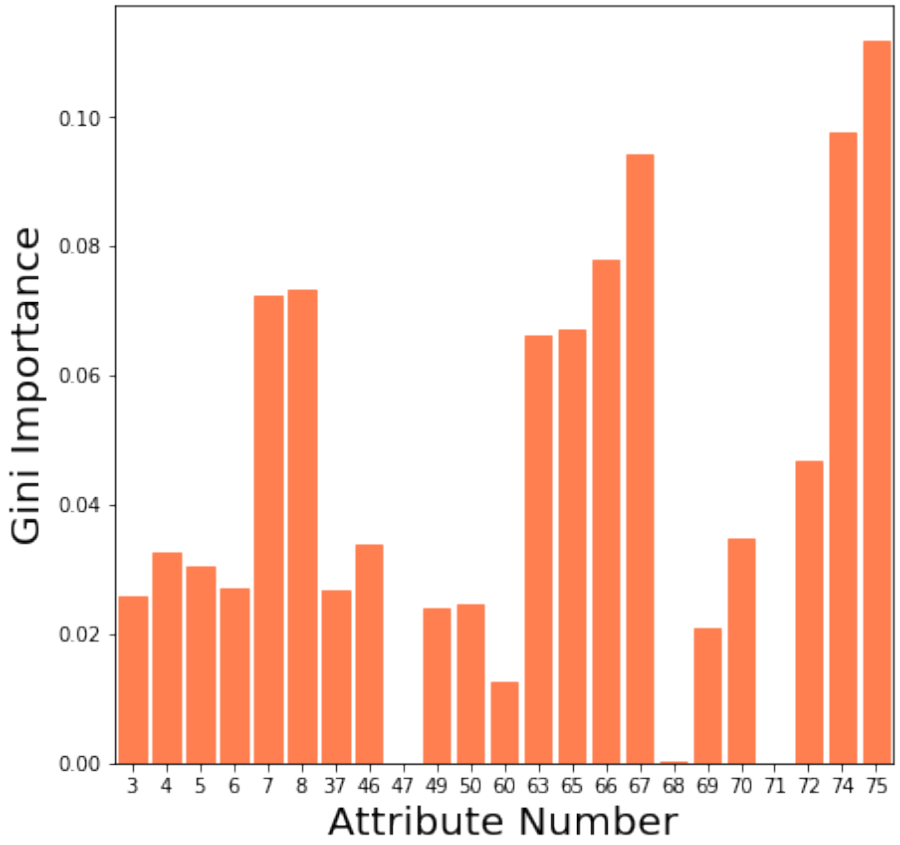

Figure 6.3: Feature Importance in AWID-CLS

Figure 6.3 shows that some of the features are not contributing at all to decision making. Figure 6.4 shows the Pearson correlation computes on the 23 features after data cleaning. The dense orange and blue shade in Figure 6.4 denotes high positive and negative correlation between those features respectively. Also, some features that are monotonically increasing in the dataset as they represent time

(feature 3, 5 and 37). Since the model must work in any session which is independent of the time of use, dropping these features before training is necessary.

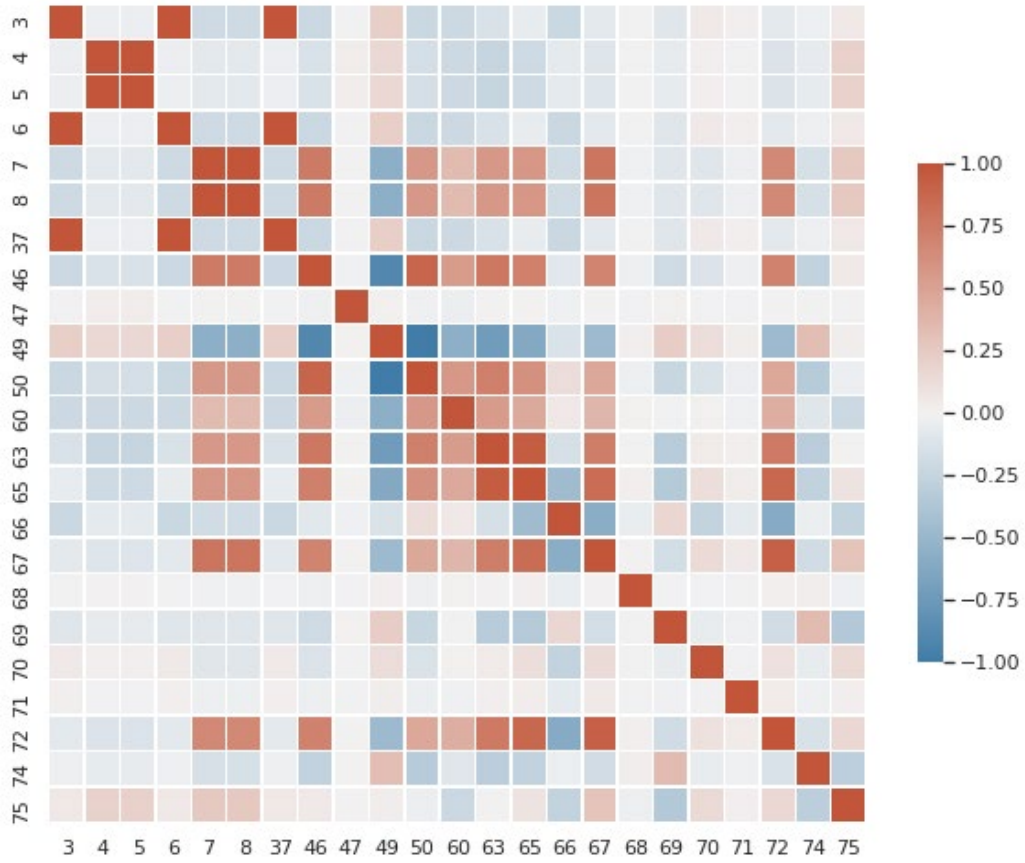

Figure 6.4: Correlation Matrix of Dataset

After considering all the factors mentioned above, 15 out of 23 features listed as below were finalised for the first-tier classifier to filter attack and normal:

Feature Number: 4, 6, 8, 46, 49, 60, 63, 65, 66, 68, 69, 71, 72, 74, 75

6.1.4. Attack Classification

Better data is better than using complex algorithms; and data cleaning is the main foundation of machine learning. After data cleaning and feature selection, the data is ready to be fed as an input to the classifiers. There are various machine learning algorithms that can be applied for categorising the type of traffic. In this research, we have considered the two most widely used methods namely: Ensemble and Neural Network model.

Ensemble methods construct a set of multiple classifiers using different splits of the same training dataset and same algorithm or same dataset with different algorithms, and then uses weighted voting between the classifiers to make a final decision [202]. We experimented with two tree based ensemble models: Random Forest and Extra Tree classifier [79]. The classifiers are trained on pre-processed

datasets with ten estimators (trees), and the results are discussed in the coming sections. Figure 6.5 describes the method of ensemble (voting).

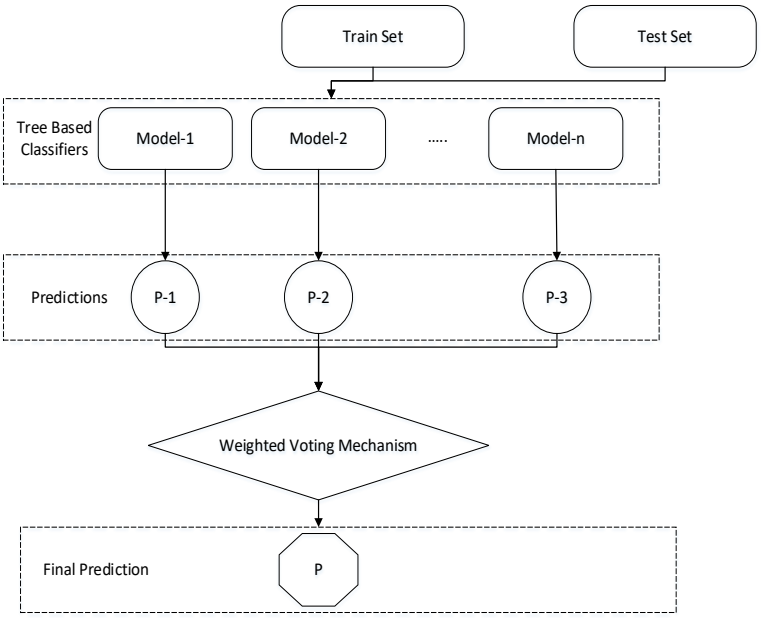

Figure 6.5: Ensemble Method Architecture

Artificial Neural Networks (ANN) are one of the most widely used classification methods to detect complex patterns in a dataset, especially when the data is non-linear in nature. It follows a multi-layer architecture i.e., input layer consists of the input dataset, where the number of input nodes in same as the number of attributes in the dataset, the hidden layer (one or multiple) consists of artificial neurons which consists of non-linear activation functions and finally the output layer consisting of artificial neurons makes the classification decision. Figure 6.6 shows the general architecture of a neural network. In this research Multi-layer Perception classifier (MLP) which is a type of neural network is used for experiments. Various ANN architectures with different number of layers, number of neurons and learning rates were used. The efficiency of neural network models with various parameters was compared. Finally, considering the input and output dimension of 15 and 1 respectively, the neural network with four layers with 15 nodes in the input layer, 50 in first hidden layer and 10 nodes in the second hidden layer, and 1 node in output layer. We used Adam optimisation technique that incorporates different learning rate and can handle sparse gradients while training. The learning rate of 0.001 is used during the training process of the neural network. For both of the hidden layers and the output layer, Tanh activation function was used.

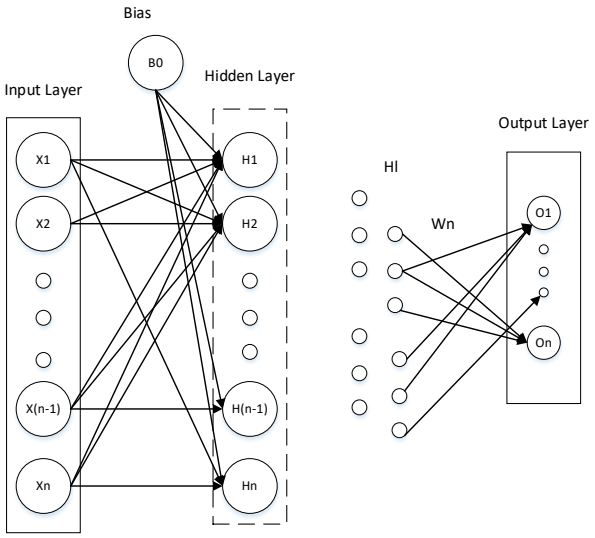

Figure 6.6: Neural Network General Architecture

6.1.5. Machine Learning Model Output Explanation and Interpretation

The experimentation utilises two different types of supervised machine learning techniques for the attack detection module of the proposed framework. Both machine learning techniques utilise the dataset features differently for classifications. Understanding the underlying mechanism of what is the model is learning is equally important other than interpreting the accuracies of the model [203]. To explain the behaviour of the machine learning models, SHapley Additive exPlanations (SHAP) [204] is utilised. SHAP provides insights on the feature impact on different machine learning techniques and how the values of the feature impact the machine learning models. It follows a game theory approach to explain the output of the machine learning models [204]. It provides an additive feature importance measure that relates to "Local Accuracy", "Missingness", and "Consistency" [203].

The python library for SHAP is used to calculate the SHAP values of the classifiers trained on the datasets. This library will help us analyse the contribution of different features in making the classification decision describing the relationship between the attribute and the class label.

6.1.6. Model Evaluation Metrics

After training, the models are evaluated using various model evaluation metrics. However, every performance metric cannot be used for all classifiers. Furthermore, every performance metric has some shortcomings that might get overlooked if the evaluation metric is not selected carefully. It is crucial to consider the application and the data on which the classifier is trained while selecting the performance metric for a classifier [205]. Some of the performance metrics used for ML classifiers are as follows:

i) Confusion Matrix: It shows the relationship between True Positive (TP), True Negative (TN), False Positive (FP) and False Negative (FN). It provides deep knowledge about the performance of the model. As Figure 6.7 shows, TP is when the sample is a positive (or 1) and is identified

as positive. FN is when the sample is positive but being falsely identified as negative by the classifier. FP is when a negative sample falsely identified as positive and TN is when a negative sample being correctly identified as negative.

Actual / Predicted	Positive (1)	Negative (0)
Positive (1)	TP	FP
Negative (0)	FN	TN

Figure 6.7: Confusion Matrix

ii) False Positive Rate (FPR): False positive rate is defined by the number of negative samples incorrectly classified as positive by the classifier.

$$FPR = \frac{FP}{(FP + TN)}$$

iii) Recall: Recall is the capability of a model to classify the positive samples from all the available samples and is also called sensitivity.

$$Recall = \frac{TP}{(TP + FN)}$$

iv) Precision: Precision depicts the correctness of the classifier in not labelling a sample positive when the true label is negative.

$$Precision = \frac{TP}{(TP + FP)}$$

v) F1-Score: F1-score is weighted harmonic mean of precision and recall with the best value as 1 and worst as 0.

$$F1 - Score = \frac{2 * Precision * Recall}{(Precision + Recall)}$$

vi) Accuracy: The accuracy metric is the ratio of correct predictions over the total number samples.

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN}$$

vii) Receiver Operating Characteristic (ROC) Curve: ROC curve is a performance metric that plots a curve between True Positive Rate (TPR).

6.2. Key Exchange Mechanism

6.2.1. Correctness of Key Exchange Mechanism

The key exchange mechanism uses the SDN controller as a trusted entity for key generation. The SDN controller is responsible for session management in the network. To verify the applicability of the proposed mechanism in a Software Defined Network, Mininet-WiFi [187] emulator is used.

Mininet-WiFi is a wireless network emulator built on top of Mininet emulator [206] for rapid prototyping of wireless SDN. Mininet-WiFi provides lightweight network nodes by virtualising Linux kernel. In our experimental framework setup, we emulate a wireless network with one SDN controller to validate and demonstrate the efficiency of the proposed key exchange algorithm. Figure 6.8 illustrates design/structure of the wireless topology created in Mininet-WiFi to implement SDN based key exchange mechanism with parameters shown in Table 6.3.

Table 6.3: Simulation Parameters in Mininet Emulator

Parameters	**Value**
Number of Nodes	3 (2 end devices, 1 SDN controller)
SDN Controller	POX Controller
Wireless Transmission	802.11 (WiFi)
SDN Protocol	OpenFlow
Device Kernel	Linux

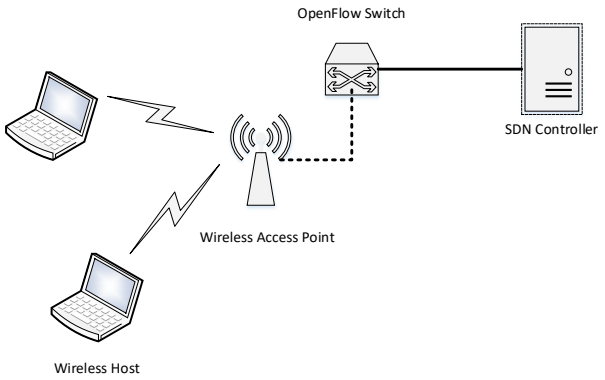

Figure 6.8: Mininet-WiFi Experiment Topology

The algorithm explained in subsection 5.2.3 is implemented on the POX SDN controller using Python 3.7 to validate the correctness and applicability of the proposed framework in a wireless SDN network. The successful key exchange between nodes is shown in Figure 6.9. “Node h1” is acting as an SDN controller responsible for initiating the session key mechanism. “STA 2” is the wireless station for

session key exchange. The screenshot shows that both entities (Node h1 and STA 2) get a common key calculated by using the public key information.

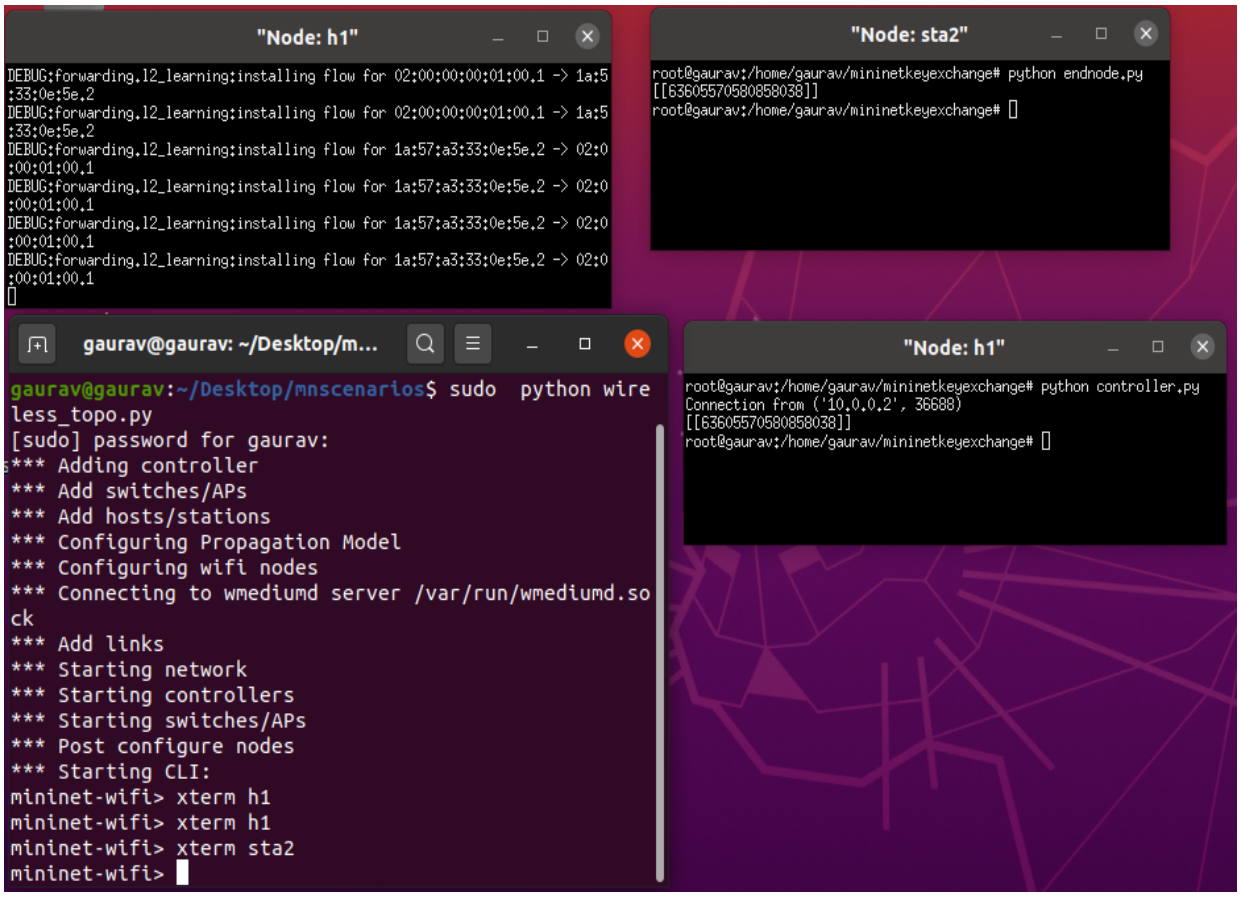

Figure 6.9: Session Key Exchange Validation in Mininet-WiFi

6.2.2. Semantic Analysis

The proposed key exchange mechanism has various aspects that require verification of effective operations. The first aspect is the secure agreement of session keys without losing the integrity of the public key information transmitted between the server and the nodes over the air. A security verification tool named Scyther [191, 207] is used to verify the successful transmission of key information.

Scyther is a tool used for the analysis of security protocol models. It is assumed that cryptographic functions are reliable, meaning the adversary learns nothing from encrypted messages unless they have a decryption key. The tool is used to find loopholes in the model of the security protocol. The protocol is formulated in the form of “roles” and “events”. “Roles” are designed based on the knowledge and operations an entity performs in the protocol model. “Events” the activities of sending and receiving of the data. Based on the operations, the “roles” make “claims” regarding secrecy, integrity, and authenticity. These claims are verified over several threat models defined in Scyther. Based on the analysis, Scyther shows the possible attacks on the security protocol model.

The security model shown in Figure 5.4 is implemented in Scyther's input language to verify the secrecy of the session key, server, and device agreement against possible attack models. Figure 6.10 shows the verifications of the claim and possible attacks on the session key agreement. The screenshot shows the verified claims and no possible attacks available in the attack model of Scyther.

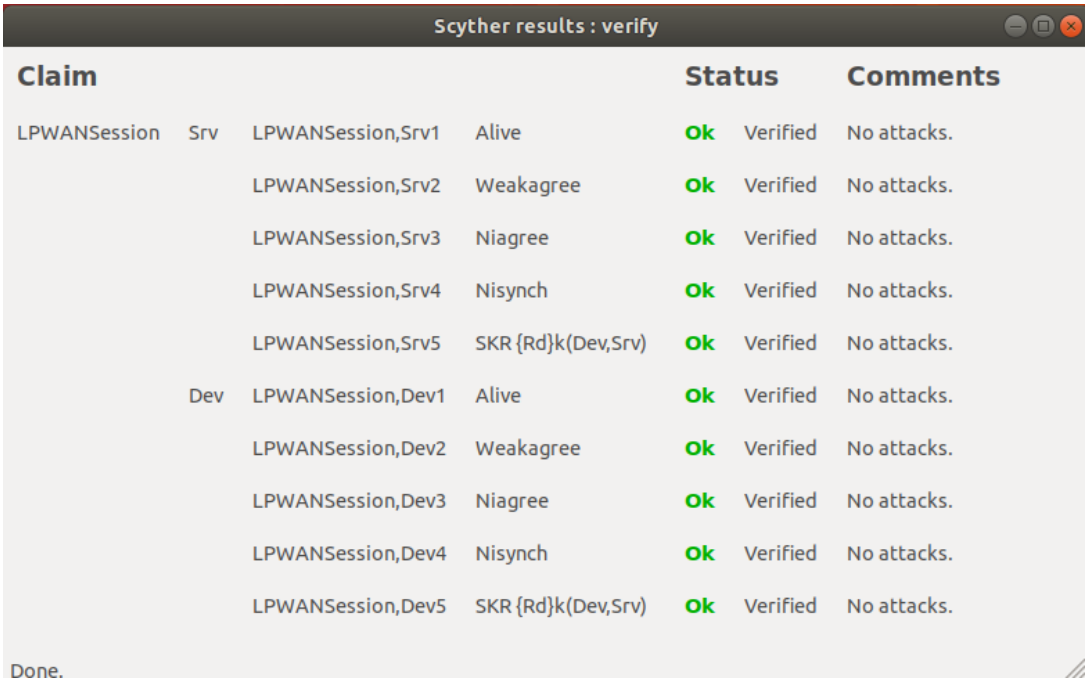

Figure 6.10: Semantic Analysis Results of LPWAN Session Key Mechanism

6.2.3. Computational Analysis

In addition to the semantic analysis of the security, the next important step is to analyse the computational cost while executing the proposed framework. The use of computationally intensive security mechanisms can shorten the end node lifetime, causing disruptions in network function. Since our research focus is LPWAN based IoT networks where we have constrained nodes, the operational overhead should be less.

Hence, our objective is to minimise the overall overhead on nodes by reducing the number of transmissions and processing overhead for end nodes. To investigate the energy consumption of the proposed scheme, we have created an energy model in NS3 based on the energy consumption of various operations such as data transmission, data reception, data encryption, and BYKA key calculation. The sections below throw light on the analysis of energy consumption by the proposed architecture.

6.2.4. Simulation Parameters:

In our experimentation, we have modified the energy model of LoRaWAN in Network Simulator 3(NS3) [208] to simulate the effects of the calculations on the end nodes. The energy model follows the standard energy consumption for transmission and reception of data packets for transceivers in LoRaWAN. In addition, the energy consumptions for cryptographic activities are added for each

transmission in the NS3 energy model of LoRaWAN. The energy model for AES encryption and MAC operations is derived from cryptographic algorithms' energy consumption described in [209]. The implementation of energy consumption for the BYKA key agreement is based on the experimental framework on sensor nodes described in [210].

In NS3 a simulation with a single gateway and end node is designed to analyse the energy consumption and node lifetime in the LoRaWAN network. The end node transmits a packet every five seconds to the gateway while energy consumption is calculated according to equation (5). Table 6.4 shows the simulation parameters used in experiments in NS3 simulator.

Table 6.4: Simulation Parameters in NS3 Simulator

Parameters	**Value**
Simulation Time	24 Hours
Initial Energy of Node	1,000 J
Supply Voltage	3.3 V
Packet Transmission Current	0.028 A
Packet Reception Current	0.0112 A
Number of Gateways	1
Number of Nodes	1
Data Rate	12 packets/Minute

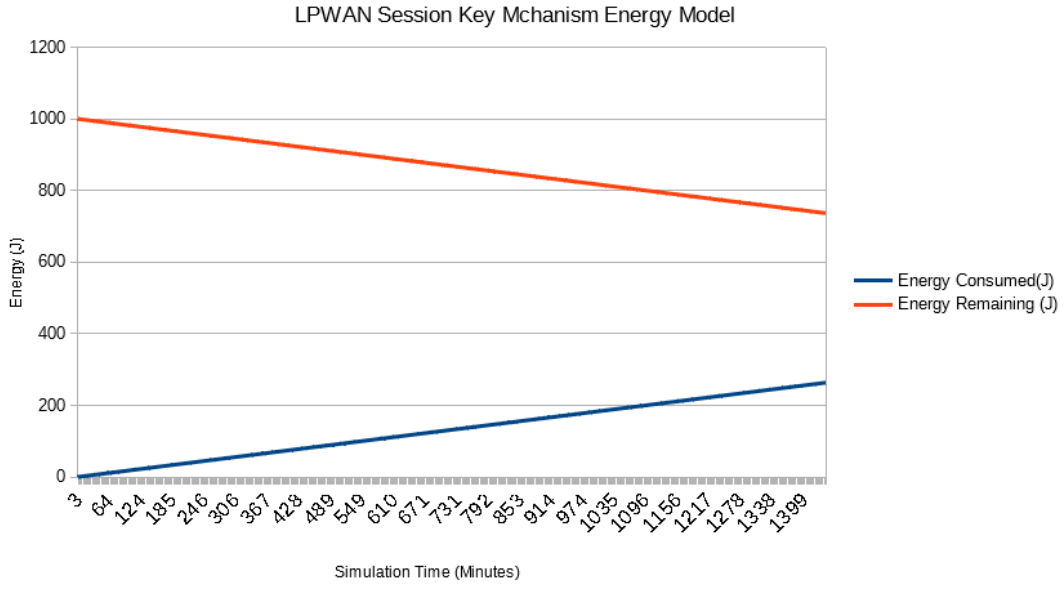

Figure 6.11: Energy Consumption of LPWAN Session Key Mechanism

Figure 6.11 shows the energy consumption of the end node over twenty-four hours of transmissions with a data rate of 12 packets per minute. Our experiment considered every transmission as an individual

session, and for every transmission different key generation operations were performed. Therefore, the number of packets for each session can be decreased, and the overhead of session key generation can be reduced on the end nodes as per security requirements by the following equation (6).

To show the benefits of our proposed algorithm on the energy consumption, we considered the worst-case scenario of maximum overhead on nodes. If the energy consumption of the end node remains the same as shown in Figure 6.11, the node can remain functional for one and a half months with 10,000 J energy which is equivalent to an 850 mAh battery with a 3 Volt supply.

In the previous scenario the transmission time was 24 hours, however, this study further investigates the impact of overhead caused by the proposed session key mechanism. In the second level of investigation, we conducted experiments to compare the energy consumption with and without session key mechanism on LoRaWAN protocol. For this comparison, the simulation time was extended to 30 days.

Table 6.5 compares LoRaWAN protocol with and without session key mechanism based on different data rates on end nodes. It is observed that with high data rates on the end nodes, the change in the session key for each packet causes extensive overhead on the end nodes. However, as the data rate decreases and the frequency of session key decreases, the difference between power consumption with and without session key mechanism narrows. It can be observed in Table 6.5 that the power consumption with session key in the case of 1 packet/minute is more than double the power consumption without session key mechanism. The difference between them comes down to almost 50% where data rate 1 packet/30 minutes. For applications with 1 packet / 6 hours, the difference between the energy consumptions is just one Joule over a month. Moreover, as we move on to the cases of even lower data rate, the difference between the power consumption in both cases becomes negligible.

Table 6.6 shows a comparison of the number of transceiver operations required by some existing and the proposed session key mechanisms in LoRaWAN. The table shws that the proposed session key mechanism achieves successful session key mechanism wit just a single transmission, reducing the transceiver operations to a half of the existing session key mechanisms for LoRaWAN, and five times less than session key mechanism discussed in [211].

IoT nodes are usually designed to last longer while sending data with low data rates (depending on the applications running on the nodes). Considering the session key mechanism's impact with high data rates, changing session key for each packet may not be ideal. However, suppose the session length is increased to even one hour. In that case, the impact of the session key mechanism can be minimised significantly and made suitable to run on applications with high data rates. On the other hand, applications with lower data rate can afford to change the session with each packet without having any severe impact on the power consumption of the end nodes. Hence, the application nodes with lower

data can have a similar lifetime as standard LPWAN protocols while utilising the session key mechanism.

Table 6.5: LoRaWAN Energy Consumption Analysis

Data Rate	**Energy Consumption (With Session Key Mechanism)**	**Energy Consumption (Without Session Key Mechanism)**
1 packet/min	692.16	272.122
1 packet/30 min	35.4716	21.4703
1 packet/hour	24.1464	17.1457
1 packet/6 hours	14.6801	13.5133
1 packet/12 hours	13.6992	13.1158
1 packet/24 hours	13.1527	12.8611
1 packet/7 days	10.992	10.9531
1 packet/15 days	10.6205	10.6011

Table 6.6: Transceiver Operation Comparison with Existing Session Key Mechanisms

Session Key Mechanisms	**Number of Transceiver Operations**
An enhanced key management scheme for LoRaWAN [211]	5
A Dual Key-Based Activation Scheme for Secure LoraWAN [94]	2
LoRaWAN OTA Activation [40]	2
Proposed Session Key Mechanism	1

6.3. Identity Theft Detection

The second module of the proposed framework is to develop a prevention mechanism against impersonation attacks on the LPWAN networks by distinguishing attack from normal traffic using machine learning techniques. As discussed in sub-section 5.2.5.1, the research utilises various machine learning mechanisms on AWID-CLS traffic dataset created on a WiFi network. The data quality is improved using various pre-processing steps as mentioned in sub-section 6.1.3. At this point, the data is prepared to be fed as an input to the machine learning model for the classification purpose.

6.3.1. Training and Testing Attack Classifiers

The proposed framework for attack detection in IoT networks adopts (is based on) two-tier implementation strategy. The first tier of the attack detection platform focuses on distinguishing between attack and normal traffic and hence this tier deals with binary classification problem. It is implemented on the network gateways allowing early filtering of attack traffic from entering the network. After being classified as attack traffic, the second tier further distinguishes between the three types of attacks mentioned in the dataset. This tier deals with the multiclass classification problem.

The training and testing processes are carried out on a Windows machine with an Intel i5-6500 CPU and 16 GB of RAM is used for machine learning modelling. Sci-Kit Learn [79] a Python machine learning library is used for building the ML models as mentioned in sub-section 6.1.2.

In the following sections, the machine earning implementation details and the results produced in both tiers are discussed.

6.3.2. First Tier Classifier for Attack Detection

AWID-CLS dataset is trained on three different models for the binary classification. Different models are evaluated on the test dataset using various performance metrics. The training and testing of the trained classifiers are discussed as follows:

Random Forest: As mentioned earlier, three classifiers were tested on the dataset using various performance metrics for their classification. The first classification algorithm tested is Random Forest.

Random Forest is an Ensemble Learning method that uses several tree classifiers on sub-samples of the dataset and averages the outputs of these multiple trees for better accuracy and to avoid overfitting [79]. The total training time taken to train the Random Forest model was 2.559 seconds.

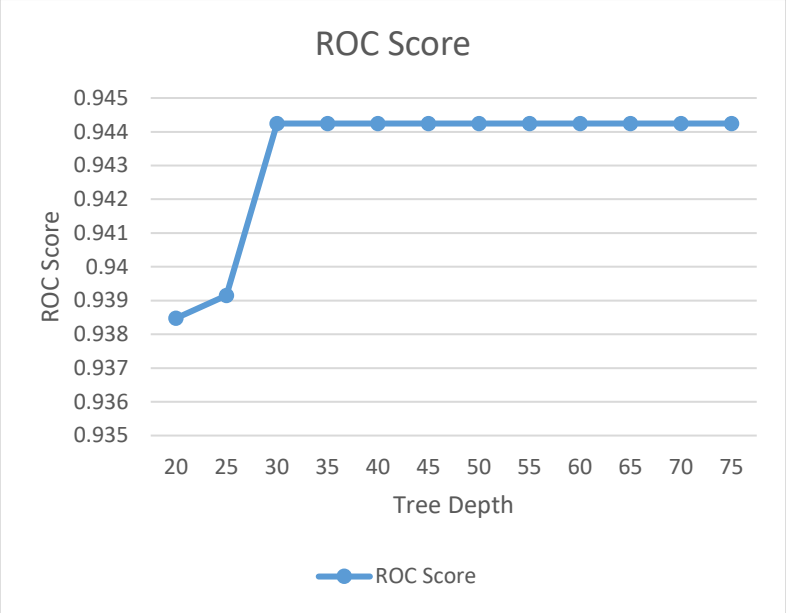

Figure 6.12: ROC values with Random Forest Tree Depth

The Random Forest tree classifier has various parameters that can impact the output of the classifier. One of the most important parameters is tree depth, as it plays a major role in classifiers size [79]. Hence, the tree depth was considered for optimisation during the training process for tree based algorithm. Random Forest classifier was trained on multiple tree depth to achieve the best performance with minimal tree depth. Figure 6.12 shows that the performance of Random Forest reaches saturation at tree depth of 30 and remains constant till 75. Hence, tree depth of 30 is considered the best parameter for tree depth.

After training the Random Forest with multiple parameters, finally Random Forest with 10 trees and tree depth of 30 is finalised based on its results. Figure 6.13 illustrates the contribution of each feature on the classifier's predictions. X-axis shows the magnitude (mean) of feature contribution and the Y-axis shows the feature number (refer to Table 6.2 to map feature number to feature names). It can be observed that feature 8 is the highest contributor while feature 71 is the lowest contributor for each class for the Random Forest classifier.

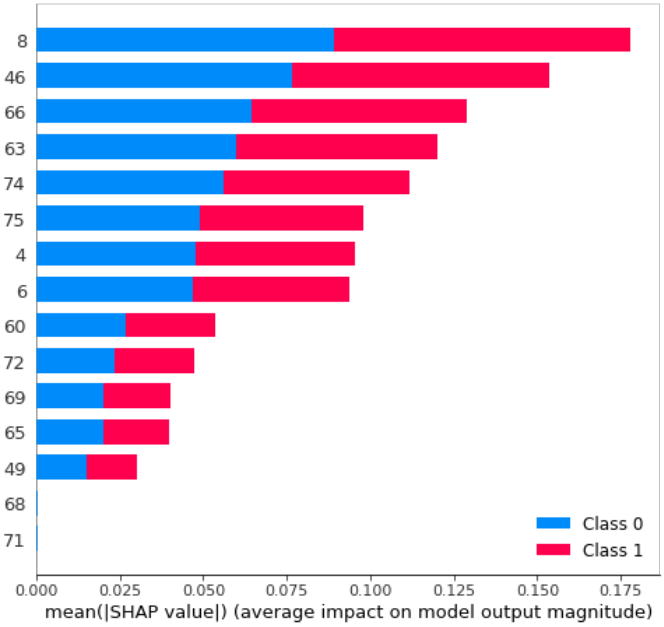

Figure 6.13: Average Feature Impact Random Forest Classifier

Figure 6.13 shows the summary of the impact of dataset features on the classifier's output. Along with feature importance, the interpretability of model's individual predictions is equally crucial. Figure 6.14 provides further detail on how different sample values impact the output of the classifier. Every sample can have a positive or a negative impact on a class. Figure 6.14 shows the feature SHAP values calculated for class 1, i.e., normal traffic in the network. It shows the positive or negative impact of a feature values on class 1 decision making. As it is calculated on a binary classifier, the negative impact on class 1 means a positive impact for class 0 classification.

It can be observed in the figure how the classifier's output shows the relationship between the feature values and class prediction. Feature 46 representing data rate information of the packet shows that most of the time when the data rate value is higher in the dataset, it has a positive impact on class 1. In other words, most of the data samples with higher data rate falls in "normal" category.

Similarly, we can deduce the relationship between other MAC layer information and the predictions of Random Forest classifier. Feature 60 represents signal strength information in the dataset. According to Figure 6.14, lower signal strength values impact positively on class 1. From this finding, we can conclude that attackers send information in the network with higher signal strength (using more transmission power). Additionally, feature 4 that characterises the time difference between packets from a source shows that "Normal" packets are mostly has a lower time delta between packets than attack traffic.

Feature 69 shows the number of retries for a packet. The number of retries is mostly high for "normal" traffic, and this can probably be happening because of lower received signal strengths, as shown by feature 60 discussed earlier. Finally, feature 11 is the representative of frame control information about "more data" flag. The dataset shows that "more data" flag values has a positive impact on class 1.

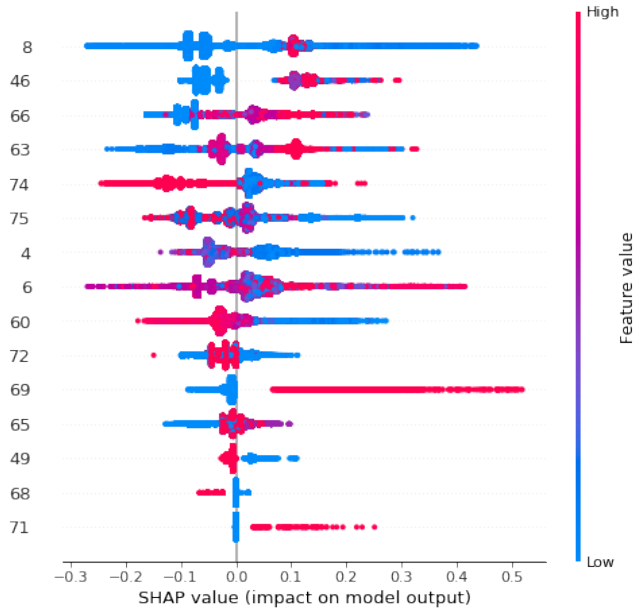

Figure 6.14: Feature Impact on Class 1 in Random Forest

Figure 6.15 provide further details on the dependence of dataset features deduced by the Random Forest classifier. The SHAP value of the feature shows how responsible that feature is for model output change. In each subplot of Figure 6.15, each dot represents a sample value from the dataset, Y-axis shows the SHAP value of the feature. The X-axis shows the values of the feature, and the colour map corresponds to a second feature selected that may have interaction with the feature that is being plotted. Figure 6.14 provides an overall summary of relationship between the feature values and model output i.e., the impact of volume and range of positive and negatives values on the model output. Figure 6.15 reveals the reasoning behind model's behaviour for each sample by considering the impact of each feature on classifier's output for each instance.

The subplot for feature 4 shows its relation to feature 66 and the SHAP values. The inference from this subplot coincides with the one from Figure 6.14: lower values in time delta in packets positively impact class 1 ("Normal" data). Feature 8 shows captured packet length and has a mostly positive impact on class 1 except for the packets smaller in length.

Another interesting behaviour is seen for feature 49 which contains Boolean value about channel type with Complementary Code Keying (CCK). Feature 49 value "0" (False) Pushes the output of the classifier as class 1 whereas the negative values lead to class 0 ("Attack") predictions. Similarly, feature 69 also shows a clear separation of positive and negative impacts. It shows that higher transmission retries values in dataset positively impact class 1 outcome. Similarly, other features can also be observed for their impact on the Random Forest classifiers output during the training process.

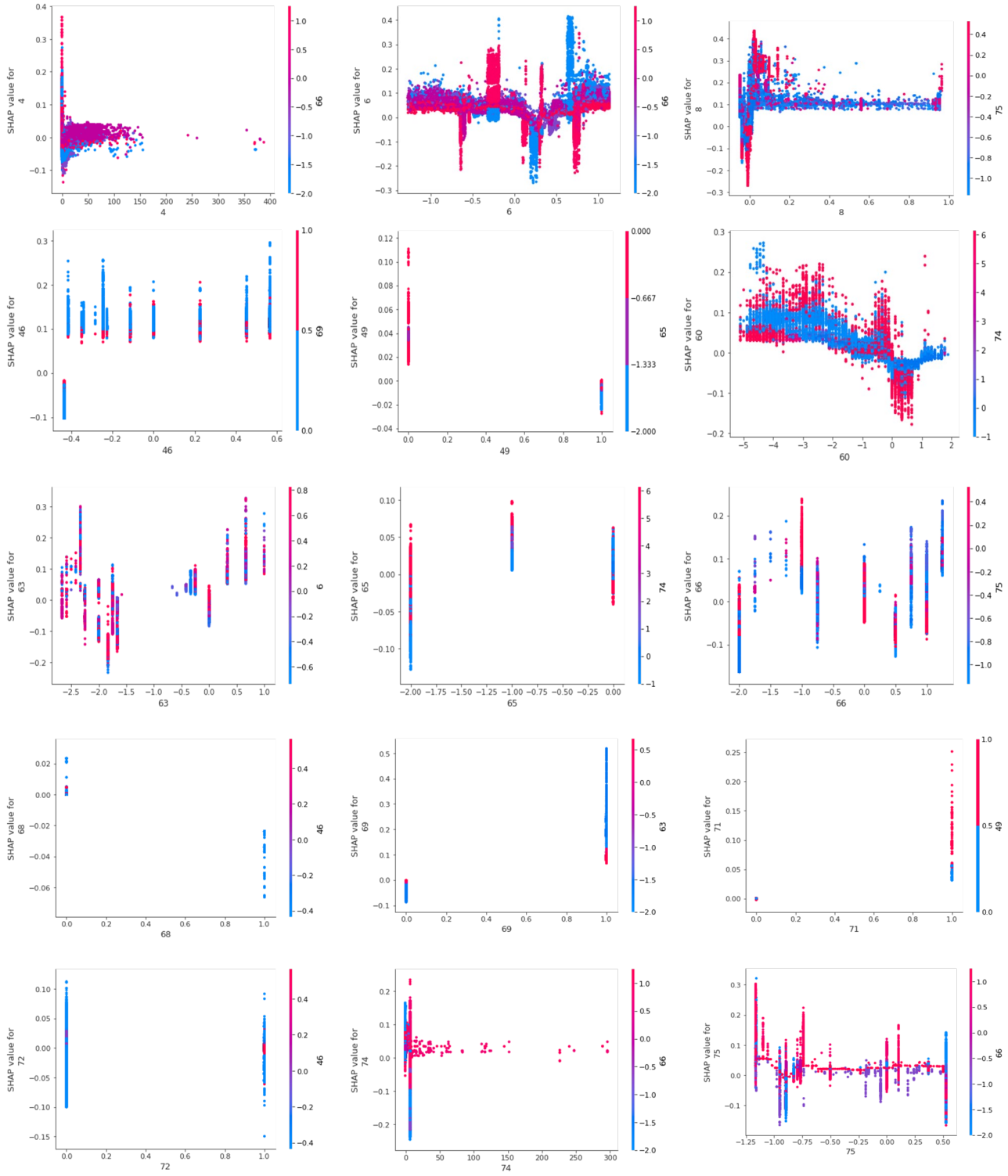

Figure 6.15: Random Forest Classifier Feature Dependence

At this point, a detailed analysis on the feature importance for the trained Random Forest classifier is provided. The internal strategies for decision making of Random Forest is revealed during its learning process i.e., the training process. Now we will discuss about the model performance on the unseen data i.e., test samples.

Figure 6.16 shows the confusion matrix for the trained Random Forest classifier on the test set. The figure shows that most of the normal and attack packets are correctly identified except for a few incorrect (1.5%) “Attack” and normal traffic classifications. The FPR for the Random Forest classifier is 0.1036, 0.9411 as TPR and 0.9463 F1-Score.

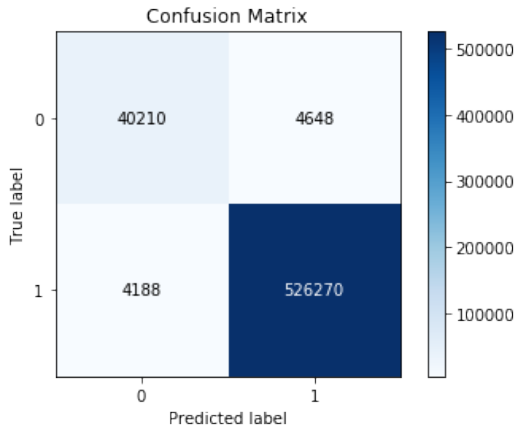

Figure 6.16: Confusion Matrix for Random Forest Classifier

Extra Tree: The Extra Tree or Extremely Randomised Tree is another ensemble based machine learning technique. The training process of the Extra tree classifier is similar to Random Forest Classifier. As tree depth plays an important role in the model’s efficiency and size, the tree depth is optimised based on the ROC score of the model. Figure 6.17 shows the performance of the Extra Tree classifier as the tree depth increases. The figure shows that the best performance is achieved when tree depth is 25. The performance of the classifier does not increase as we increase the tree depth. Considering the tree depth and performance, the best value for the tree depth parameter is 25.

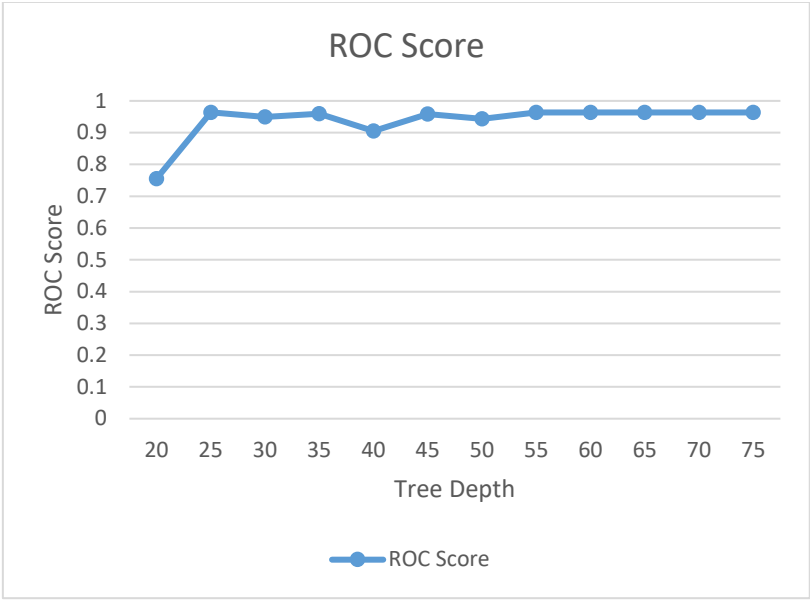

Figure 6.17: ROC values with Extra Tree Depth

Figure 6.18 shows the overall feature contribution for the trained Extra Tree Classifier. It can be observed that the feature learning is different for Extra Tree classifier and Random Forest as the feature contribution in Extra Tree is different to that of Random Forest.

Feature 49 containing Boolean value about the channel modulation information for CCK is identified as the most informative and discriminating feature by the Extra Tree model. The second most influential is feature 66, containing frame control subtype. On the other hand, Random Forest as shown in Figure 6.13 relies mostly on feature 8 that is, frame captured length and feature 3, the data rate. Thus, the least two influential features are the same for both classifiers.

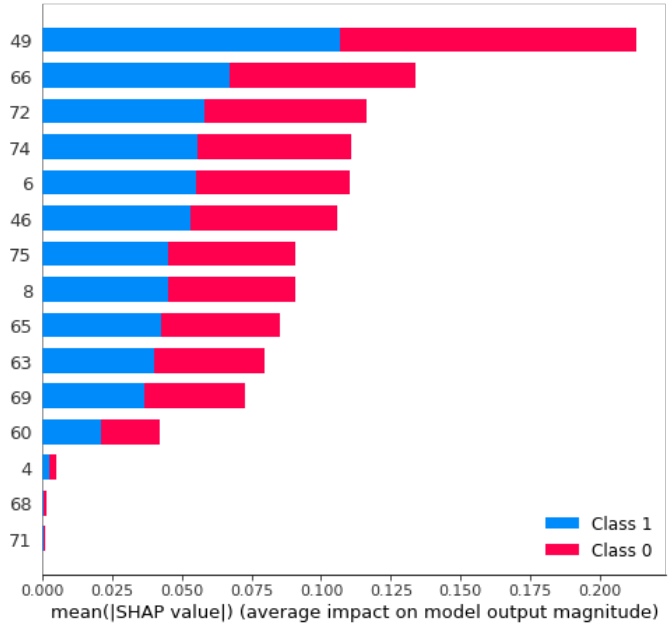

Figure 6.18: Average Feature Impact Extra Tree Classifier

Figure 6.19 shows further details of feature impact on Extra Tree classifier for class 1, i.e., “normal” data samples. Feature 4 has the most influence on the Extra tree decision making as discussed above. Figure 6.19 shows how the values of feature 4 impact class 1 (“normal”) samples. The low values for feature 4 have a positive impact on class 1.

It can be observed in Figure 6.19 that the overall implications are similar to Figure 6.14. However, the Extra Tree classifier provides a clear relationships with feature values and its impact on output class for some features. Other than feature 49, feature 68 also shows a clear relationship between values and decision class. The impact of Feature 69 is similar to Random Forest. The higher values of the feature are pushing the decision of the model towards class 1.

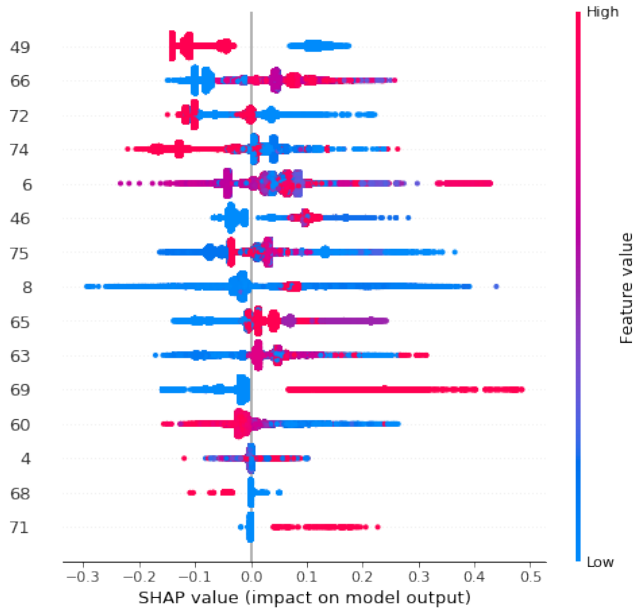

Figure 6.19: Feature Impact for Class 1 in Extra Tree Classifier

In Figure 6.20, further details on feature impact on Extra Trees decision making are shown. Each subplot shows the impact of every feature sample in the dataset on the output of the classifier. The final implications of Figure 6.20 are similar to Figure 6.15. However, the dependence of the Extra Tree classifier on the dataset feature is different. Like Random Forest, the Extra Tree Classifier is trained on the same pre-processed dataset and tested on the test set. The Extra-Tree Classifier achieved better results than Random Forest, which can be seen by comparing the confusion metrics shown in Figure 6.16 and Figure 6.21. Extra Tree classifier achieves TPR 0.9631, FPR 0.0724, and F1- Score 0.9755.

The confusion matrix in Figure 6.21 shows that the Extra Tree classifier shows a great capability in correctly identifying “normal” traffic with very few “normal” samples incorrectly identified. Also, the Extra Tree has a much lower FPR than the Random Forest classifier. Hence, it can be deduced that the Extra Tree classifier is better at classifying bot “attack” and “normal” traffic with higher F1-Score than the Random Forest classifier.

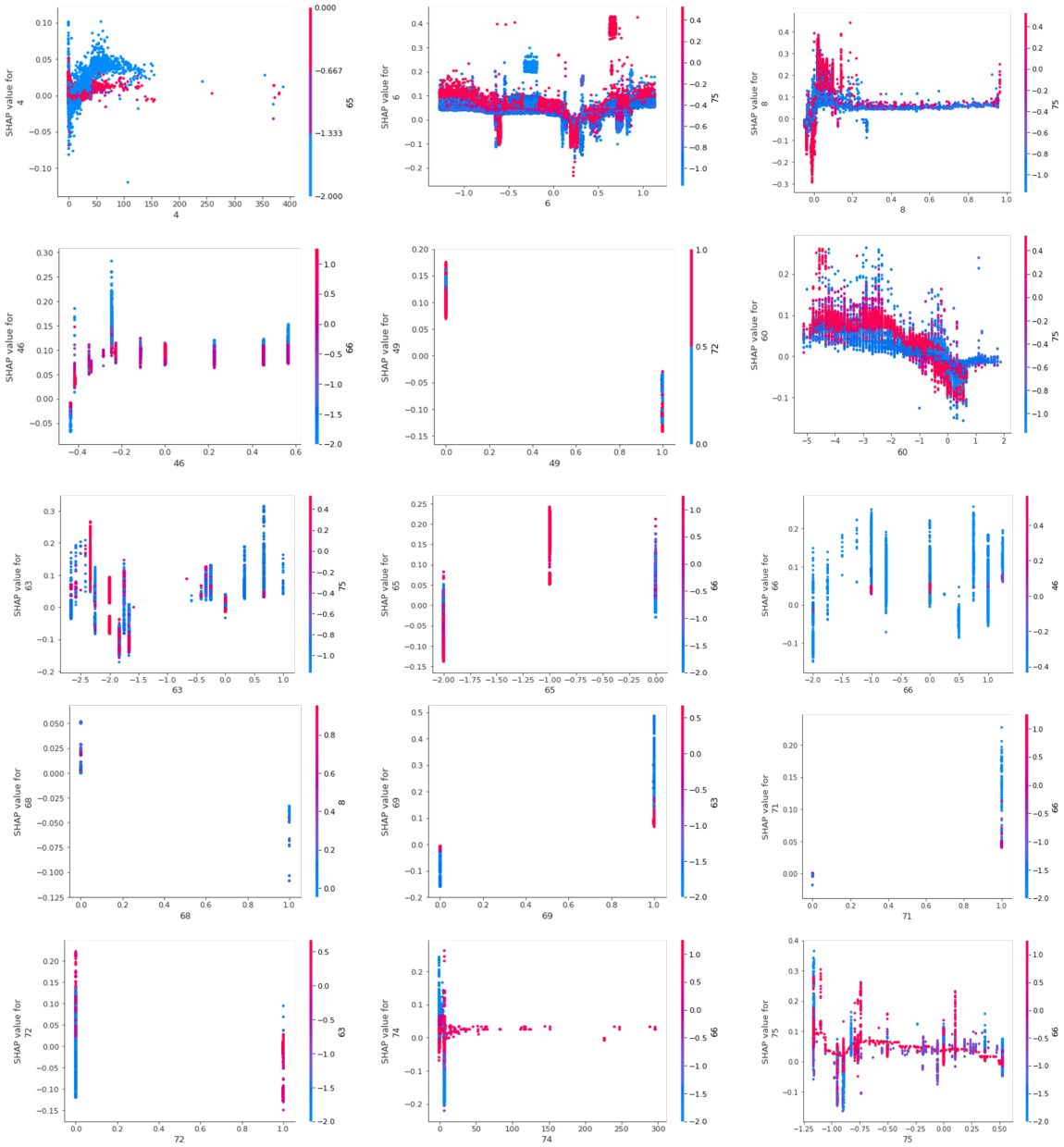

Figure 6.20: Extra Tree Classifier Feature Dependence

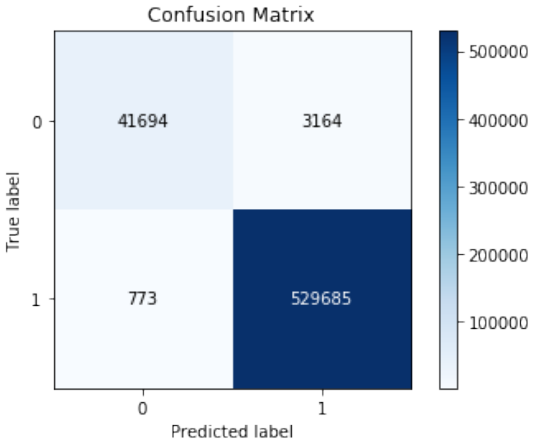

Figure 6.21: Confusion Matrix for Extra Tree Classifier

Neural Networks: The third type of machine learning technique employed for this research Neural Networks. Figure 6.22 shows the feature contribution in the decision making of the Neural Network classifier trained on the pre-processed dataset. The Neural Network classifier has different decision making than the other two previously trained classifiers. The Neural Network finds feature 63, the information about the frame controller subtype as the most distinguishing feature in classifying the samples. The second most informative features are 66 and 74, representing frame control subtype and the duration of the transmission respectively. The Least contribution features are feature 49, 68 and 71. On the contrary, Figure 6.18 shows that feature 49 is the highest contributor for extra tree's decision making. "attack" traffic are 68 and 71.

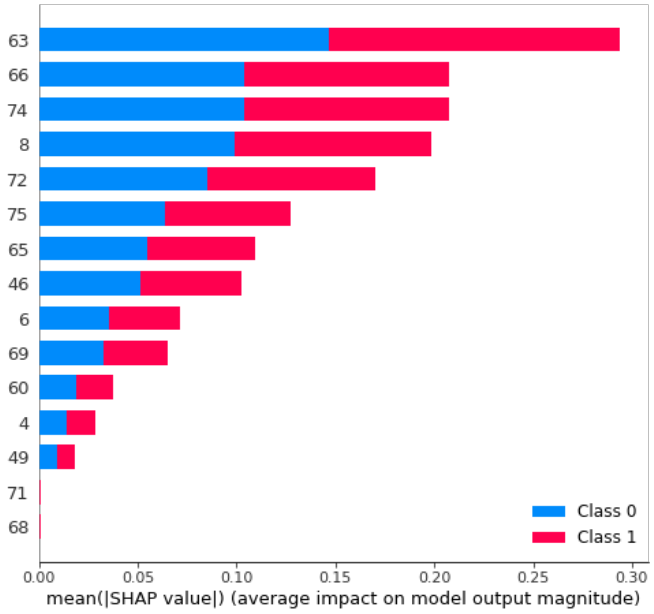

Figure 6.22: Average Feature Impact Neural Network Classifier

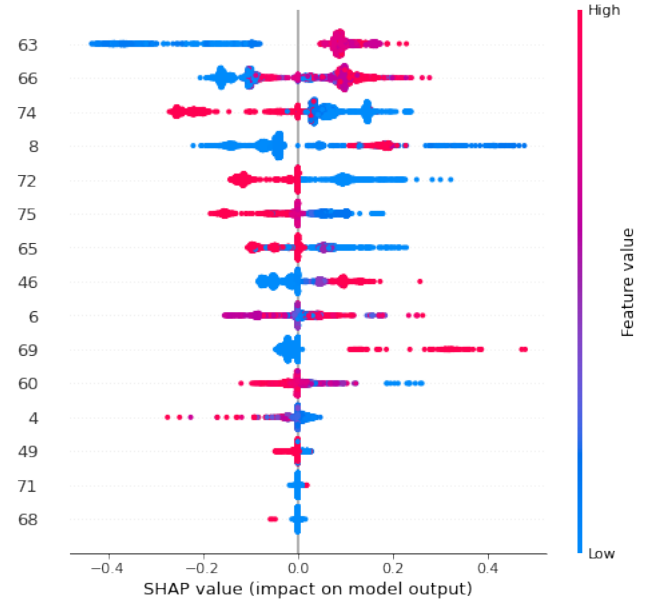

Figure 6.23: Feature Impact on Class 1 Neural Network

Figure 6.22 and Figure 6.23 Provides more information about the relationship between SHAP values and feature values. The SHAP values for Neural Networks are computed on 2000 random samples from the training dataset because of computational constraints. Figure 6.23 shows a clear relationship between the feature values and the classifier's output. Feature 63 with high range in the training instances have a positive impact on class 1 and pushes the decision towards class 0. On the other hand, higher values of feature 72 and 74 impacts negatively on class 1 decision making. Figure 6.24 shows how every sample in the dataset contributes by plotting the SHAP value for samples of every feature of the dataset in each subgraph. In addition to the feature contribution, Figure 6.24 also shows how the Neural Network finds a correlation between two features in the dataset.

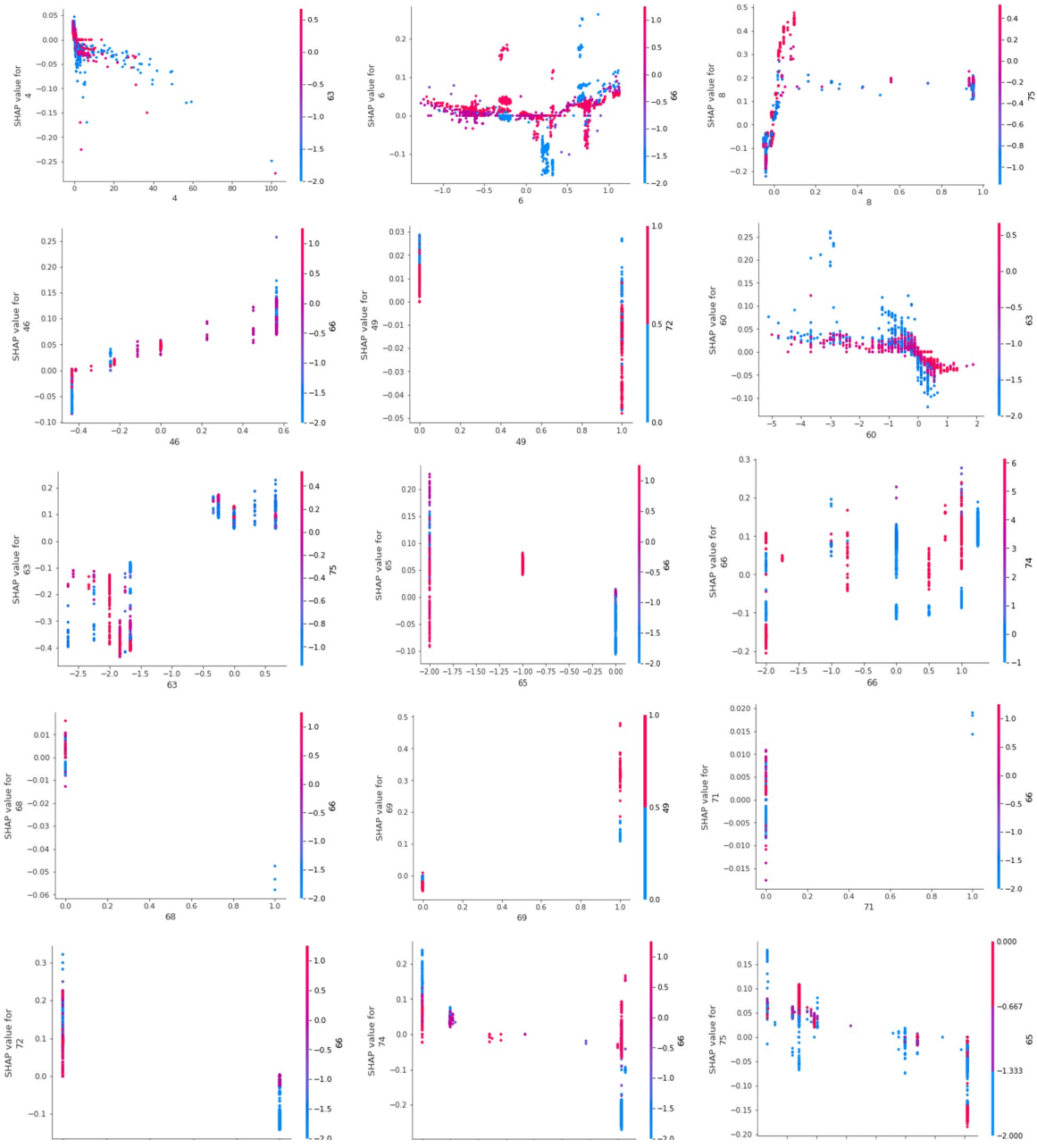

Figure 6.24: Neural Network Classifier Feature Dependence

Figure 6.25 shows the confusion matrix of the trained Neural Network classifier on the test set. Compared to the performance of Random Forest and Extra Tree classifier, the Neural Network has lower performance. However, it has a lower FPR than the other two classifiers. It achieves TPR 0.9573, FPR 0.0166, and F1- Score 0.9319.

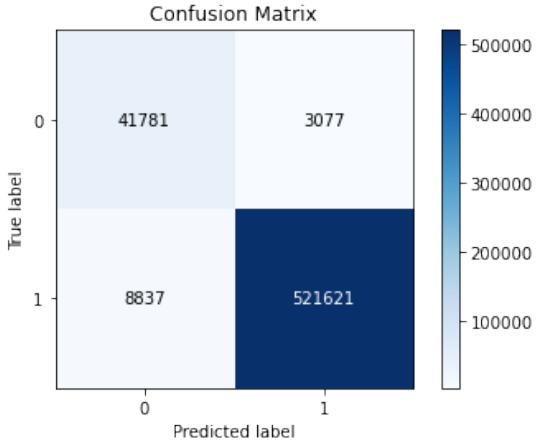

Figure 6.25: Confusion Matrix for Neural Network Classifier

After training all the three classifiers on the pre-processed dataset, their correctness is tested on the test set, a different set of samples unseen to the trained model. Table 6.7 shows the comparison of the performance of the three classifiers based on various performance metrics and their training time. In addition to performance metrics, Table 6.7 also shows the training time of the three classifiers on a windows machine with 16 GB RAM and Intel i5-6500 CPU. The table shows that the Extra Tree classifier produces the most promising results on the test set out of all the classifiers. On the other hand, as seen from the performance metrics, Neural Network performs poorly in comparison with other classifiers, the exception being the FPR value. Low FPR for Neural Network means that it identifies the attack data better than other classifiers

The objective of the first tier classifier is to separate “normal” and “attack” from incoming traffic on the network gateway. As the classifier in tier-1 is deployed on network layer, the classifiers are trained only on MAC layer information available on the gateway. The gateway devices in networks have comparatively more resources than end nodes. However, they still have limited computational power and hence it is vital to deploy a lightweight machine learning model on the gateways. The requirement of the lightweight classifier caused us to train a binary classifier only for “attack” and “normal” traffic rather than training a multiclass classifier for all the attack classes in the dataset. The binary classifiers are simple and have lower computational requirements. Considering the memory requirements, the Extra Tree classifier proves to be the lightest amongst three trained classifiers in tier-1 capable of achieving the highest performance amongst the three classifiers. This makes the Extra Tree classifier the fittest model to be deployed as tier -1 classifier on gateways.

Table 6.7: Classification Model Results for Classifiers trained in Tier-1

Classification Method	**Training Time**	**ROC-AUC Score**	**Precision**	**Recall (TPR)**	**FPR**	**F1-Score**
Random Forest	2.559 sec	0.944244	0.948458	0.941178	0.103615	0.946339
Extra Tree	1.567 sec	0.963111	0.988770	0.963111	0.072473	0.975526
Neural Network	181.068 sec	0.957373	0.909776	0.957373	0.01665	0.931961

6.3.3. Second Tier Classifier for Attack Profiling

The objective of tier-1 classification is to segregate the "attack" and "normal" traffic. However, there are three types of attacks in the dataset. The tier-2 is designed to identify the type of attacks to take appropriate actions against the network attacks. The tier-1 classifier deployed on the gateway redirects the "attack" traffic to the tier-2 classifier for attack profiling.

Table 6.8: Dataset Features for Tier-2 Classifiers

Dataset Feature	**Feature Name**
4	frame.time_delta
8	frame.cap_len
47	radiotap.channel.freq
60	radiotap.dbm_antsignal
63	wlan.fc.type_subtype
65	wlan.fc.type
66	wlan.fc.subtype
67	wlan.fc.ds
68	wlan.fc.frag
70	wlan.fc.pwrmgt
71	wlan.fc.moredata
72	wlan.fc.protected
74	wlan.duration
75	wlan.ra
140	Wlan.wep.key
153	data.len

Figure 6.26 shows the architecture design for tier-2 classifier used to profile attacks in the networks. The architecture takes the pre-processed attack training dataset. The training set has additional features as compared to tier-1 phase are added to the model at this stage with an assumption that the model will perform better. Table 6.8 shows the features used in training the tier-2 classifiers. The training set is divided into three subsets with combinations of two classes in each subset. A separate binary classifier is trained on each subset of the dataset. In addition, one multiclass classifier is trained on the whole dataset. Thus, in total four classifiers are trained on attack classes. A majority voting mechanism is designed for the binary classifiers trained on subsets of attack data in the next step. The testing set is given to the all the four classifiers for predictions. Each classifier provides a label (attack type) to the test samples. Finally, the class predicted by the majority of the classifiers is assigned as the label to the test samples. Now, there can be a scenario where the binary classifiers in the voting mechanism cannot come to majority example. In other words, each classifier can predict different class as we have three attack categories and three classifiers. In this situation, a multiclass classifier is used as a tie-breaker to come to a final concrete result.

Similar to tier-1, tier-2 also implements three classifiers for training and testing phase. Three different subsets with each subset containing two types of attack samples are created. The fourth subset includes the complete training set. Using multiple types of classifiers help in developing a more generalised outcome for the tier-2 classifiers for attack profiling.

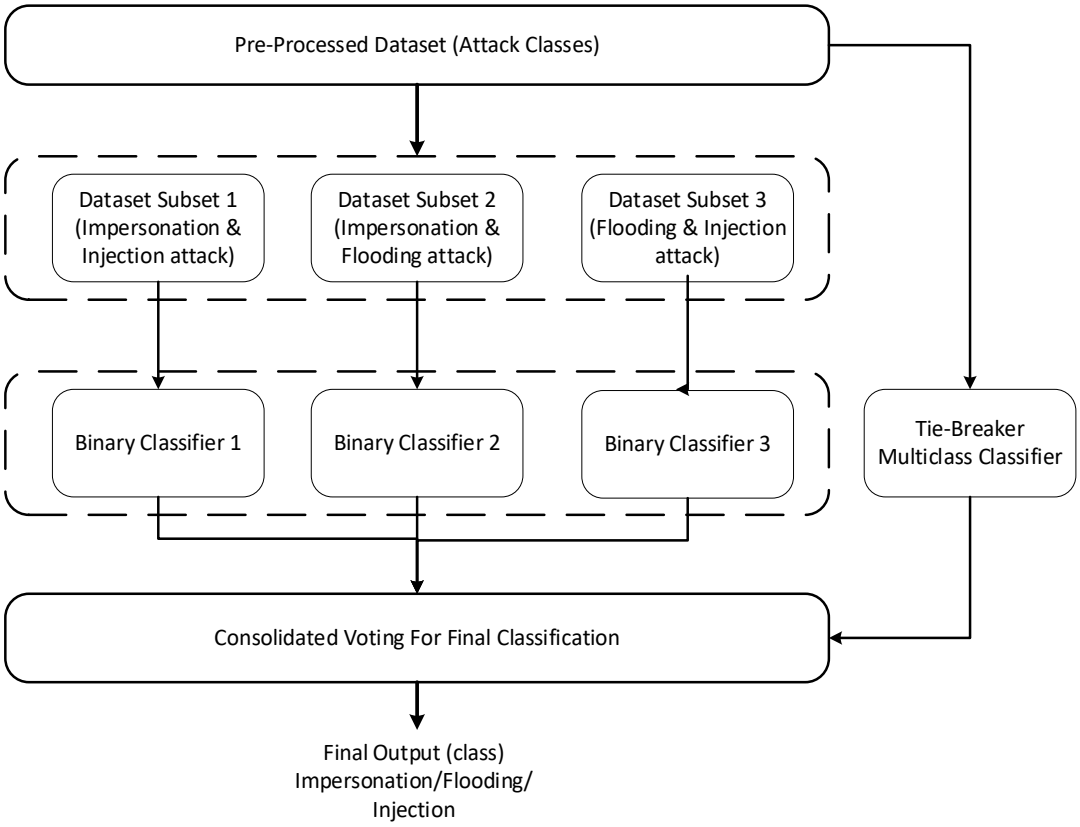

Figure 6.26: Tier-2 Classifier's Architecture

6.3.3.1. Impersonation and Injection Attack Classifiers

The first subset of the dataset take has instances of two classes of "attack" samples: "Injection" and "Impersonation" attack.

Table 6.9 shows the number of samples from each class in the training and testing set in the subset. As Table 6.9 shows, this "attack" data subset does not have a considerable data imbalance; hence the data subset does not need to be balanced. Furthermore, the whole "attack" dataset is pre-processed and scaled before the subsets are created. Therefore, there is no requirement to scale the data subset again.

Table 6.9: Impersonation and Injection Attack Data-Subset Class Instances

Class Name	**Train Set Instances**	**Test Set Instances**	**Class Label**
Injection	65,379	16,682	2
Impersonation	48,522	20,079	1

Radom Forest: The first classifier trained on the "Impersonation" and "Injection" attack data subset is the ensemble based Random Forest classifier. Provide interpretability for Random Forest model behaviour, SHAP values are calculated for each classifier. Figure 6.27 shows the overall contribution of dataset features on the decision making of the classifier. The highest contribution on "Impersonation" and "Injection" classes are made by feature 75 and feature 74 containing information about the receiver's address and transmission duration of the packets, respectively. There are several features in the subset that are not contributing towards classifier's decision-making. Such as feature 68, 8, 47, and 71 where Feature 68 and 71 represents frame control information; feature 8 represents information about data length captured, and feature 47 samples represent the channel frequency information.

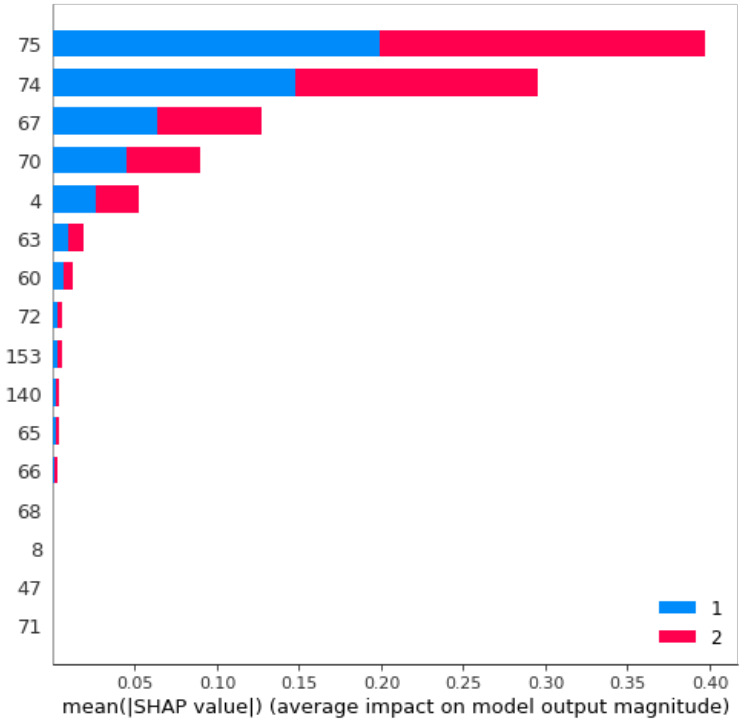

Figure 6.27: Average Feature Impact Classifier 1 (Random Forest)

Figure 6.28 shows the association of feature values and classifier's decision for class 1 ("Impersonation") attack samples. It shows that feature 75 contributes positively towards class 1 when the samples have a higher value. Similarly, feature 74 has information about transmission duration that contributes positively towards class 1 when it has lower values. Alternatively, it can be said that samples with lower transmission duration are more likely to be an "impersonation" attack and with a higher duration "injection" attack. Feature 70 also shows the same trend where higher values are contributing positively towards class 1. Feature 72, 153, 140, and 65 contribute positively towards class 1 in an opposite manner, the lower values for these features pushes the decision making of the Random Forest classifier towards class 1.

The SHAP describes how and what the classifier has learned from the training set. After training the classifier on two classes, the classifier is validated on the test set. During validation, the test set with all the three classes of attack data are used to check how the classifier classifies the unknown samples.

Figure 6.29 shows the confusion matrix of the Random Forest classifier on "Impersonation" and "Injection". It shows that the classifier accurately classifies the samples of class 2 ("Injection" attack). However, class-1 has considerable True Negatives in the classification and are labelled as class 2, which means that the classifier has difficulties creating a decision boundary for many cases.

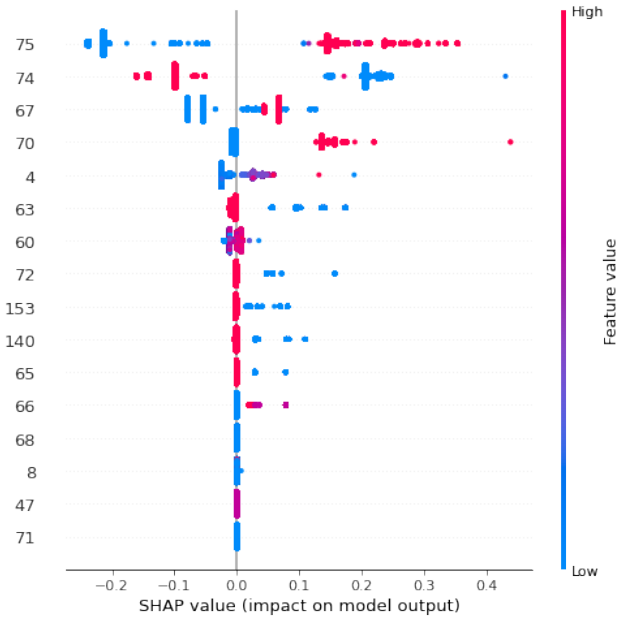

Figure 6.28: Feature Impact on Class 1 Classifier 1 (Random Forest)

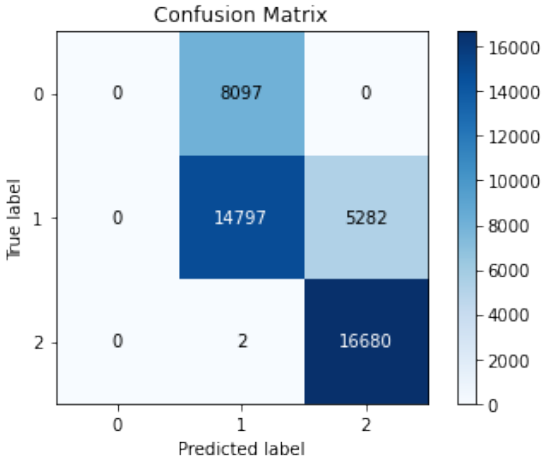

Figure 6.29: Confusion Matrix Classifier 1 (Random Forest)

Extra Tree: The second classifier tested for the "Impersonation" and "Injection" attack dataset is another ensemble based machine learning model named Extra Tree Classifier. It is trained on the same pre-processed as Random Forest classifier. Figure 6.30 shows the learning of the Extra Tree classifier from various features in the data subset and how they impact the classifier's decision making.

Figure 6.30 shows that the Extra Tree classifier gives different priorities to various features during the learning process as compared to the Random Forest classifier. Feature 74 has been identified as the most contributing attribute in deciding the class label for the training process. The second most important feature is feature 67 that contains the information about frame control Distributed System Status. And the third most contributing feature is feature 75 which represents the information about the receiver's address.

The Random Forest classifier finds feature 75, 67, 74 in the dataset as the top three impactful features but in a different order. The Random Forest relies most on the receiver's address while keeping feature 74 and feature 67 at positions second and third, respectively.

Figure 6.31 provides further details on the relationship of feature values and decision making of the Extra Tree classifier. Although the order of feature contribution is different for the Extra Tree classifier, the relationship of the feature values and the classifier's decision for "impersonation" class is the same as Random Forest. The higher values for the transmission duration shown by feature 74 are impacting negatively on class 1, meaning the impersonation attack samples mostly have lower transmission duration than the injection attacks. Feature 67 signifies which direction the packets are travelling. The higher values of feature 67 are mostly positively impacting class 1 as they were in Random Forest. Feature 70, containing the value of power management flags with higher values are more likely to be "impersonation" attack than "injection" attack as shown with the SHAP value in Figure 6.31. Further, for features 63, 60, and 65 samples with lower value impacts positively on "impersonation" attack samples.

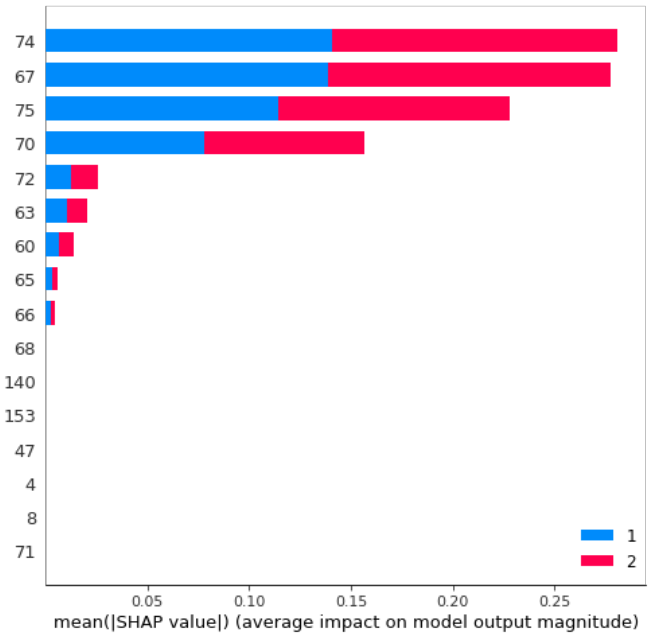

Figure 6.30: Average Feature Impact Classifier 1 (Extra Tree)

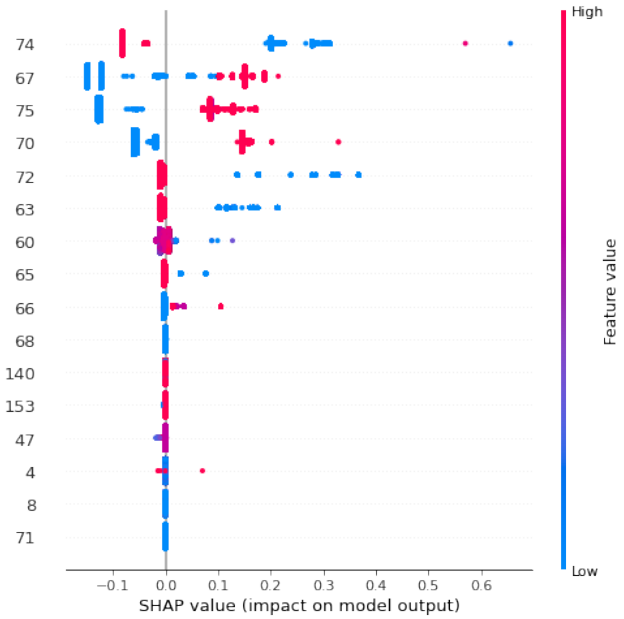

Figure 6.31: Feature Impact on Class 1 Classifier 1 (Extra Tree)

Figure 6.32 shows the confusion matrix on how the classifier performs on the test set with all three classes in the dataset. It shows that the Extra Tree classifier has better performance in classifying both "Impersonation" and "Injection" attack. It has much lower number of True Negatives for "impersonation" attack than Random Forest. It is also evident that both the Random Forest and Extra Tree classifier label all the "Flooding" attack data unknown to the classifier as "Impersonation" attack. This signifies that "Flooding" and "impersonation" attack samples have similar properties that are being used by the classifiers to decide on label 1, the "Impersonation" attack.

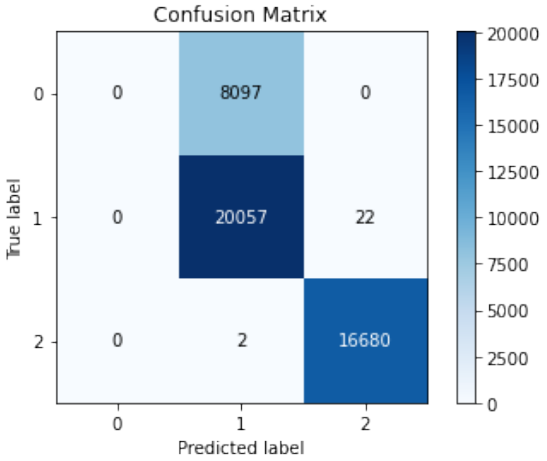

Figure 6.32: Confusion Matrix Classifier 1 (Extra Tree)

Neural Networks: After Experimenting with Random Forest and Extra Tree classifier on the first subset of the dataset, a different type of machine learning technique i.e., neural network is tested on the same data subset of "Impersonation" and "Injection" attack. A three layer Neural Network was

trained with 16 input nodes, 50, and 10 nodes in hidden, and 3 nodes in output layers with Adam optimiser and tanh activation function (same as tier-1 neural network classifier).

Figure 6.33 shows how the Neural Network learns from the features in the dataset. It shows that Neural Network is highly influenced by feature 75 and 74 while having low impacts from all the other feature. It can be observed that the Neural Network classifier's decision-making uses almost all the features available in the dataset. On the contrary, Extra Tree and Random Forest classifiers ignore the less likely features to contribute to the decision making.

Figure 6.33 shows that the Neural Network decision making is also getting highly influenced by feature 74, similar to Extra Tree. In other words, transmission duration of the packet in the dataset contains enough information that has helped the classifiers in categorising different attacks. The second most important feature is 74, followed by feature 70. However, the rest of all the features have a very low impact on Neural Network's decision making.

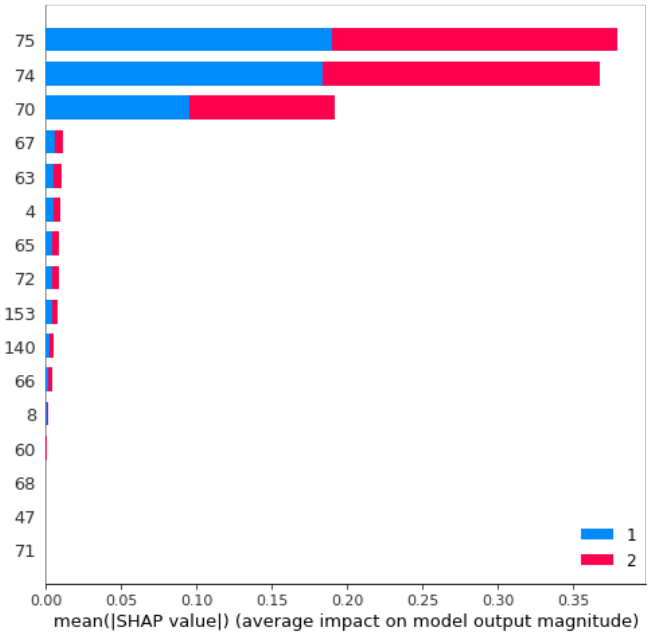

Figure 6.33: Average Feature Impact Classifier 1 (Neural Network)

Figure 6.34 shows the dependence of class 1 decision of the values of feature samples. Neural Networks have the same correlation for class 1 decision making and feature values as ensemble methods. Feature 74 contains the value of transmission duration impacts positively. All the tree trained classifiers implicate that "Impersonation" attackers have lower transmission duration than "Injection" attacks on the network.

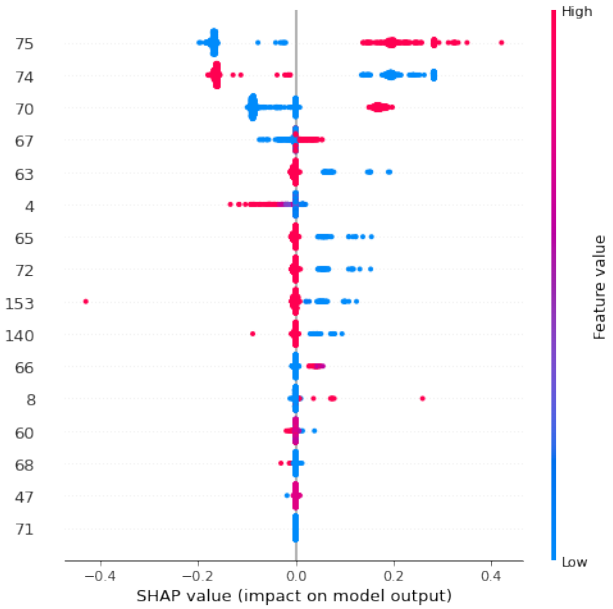

Figure 6.34: Feature Impact for Class 1 Classifier 1 (Neural Network)

Features 70 and 67 have positive contribution with high value data samples. On the contrary, features 63, 65, 72, 140, and 153 with lower values favour the model towards "Impersonation". The only feature that is not contributing to the decision making of Neural Network is feature 71.

After the training, the test set is introduced to the trained classifier to make predictions. Figure 6.35 shows the confusion matrix of the Neural Network. It can be seen that True Negative for "Impersonation" attack is the highest amongst all the three other classifier. The performance in the classification of "Injection" attack is the same for all three classifiers. However, Neural Network fails to identify the "Impersonation" attack samples effectively. All the class 0 ("flooding" attack) samples unknown to the Neural Network are classified as class 1, which is the common observation in other two classifiers as well.

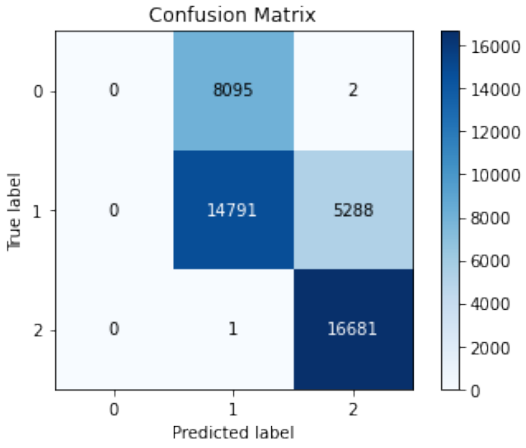

Figure 6.35: Confusion Matrix Classifier 1 (Neural Network)

In the above sections we described how the three different classifiers learnt from various features in the training dataset for their decision making process. Table 6.10 shows the brief summary about the various features and their level of contribution to the classifiers' decision making. It can be observed that some features like 74, 75 and 70 of the features constantly play important role in decision making, whereas many features are not utilised by some of the classifiers. For example, the Extra Tree depends on the least number of feature in this data subset, whereas the Neural Network relies on most of the features in the data subset.

For "Impersonation" and "Injection" attacks, the most decisive feature is proven to be feature 74, containing the information about transmission duration, followed by feature 75 that carries information about the receiver's address. On the other hand, feature 70 representing frame control information about power management consistently contributes to all the classifiers. It is assumed that signal strength (feature 60) carries important information about the network nodes. However, it has a comparatively lower contribution in classification of the "Impersonation" and "Injection" attacks. Feature 71, on the other hand, is not contributing to the decision making of any classifier.

In this subset, the results on "Flooding" attack samples are ignored as these classifiers are not trained on "flooding" attack samples .Looking at the performance of the three trained classifier, it is evident that even after learning from the least number of features, Extra-Tree Classifier is the most effective classifier amongst the three trained classifiers. It has the lowest True Negatives and False Positives for "Impersonation" and "Injection" attack samples. On the other hand, the Neural Network has the lowest performance amongst the tree three with a high True Negative for "Impersonation" attack samples. Neural Network could not learn effectively from the features and hence could not detect patterns to distinguish the two attack grows. Because of this, the performance of Neural Network is comparatively poor.

Table 6.10: Feature Contribution Summary for Classifiers on "Impersonation" and "Injection" Attack Data-Subset

Dataset Feature	**Feature Name**	**Random Forest**	**Extra Tree**	**Neural Network**
4	frame.time_delta	Average	Nil	Very Low
8	frame.cap_len	Nil	Nil	Very Low
47	radiotap.channel.freq	Nil	Nil	Very Low
60	radiotap.dbm_antsignal	Low	Low	Very Low
63	wlan.fc.type_subtype	Low	Low	Low
65	wlan.fc.type	Very Low	Very Low	Low
66	wlan.fc.subtype	Very Low	Very Low	Very Low
67	wlan.fc.ds	Average	Very High	Low
68	wlan.fc.frag	Nil	Nil	Very Low

Dataset Feature	Feature Name	Random Forest	Extra Tree	Neural Network
70	wlan.fc.pwrmgt	Average	Average	Average
71	wlan.fc.moredata	Nil	Nil	Nil
72	wlan.fc.protected	Very Low	Low	Low
74	wlan.duration	High	Very High	High
75	wlan.ra	Very High	High	Very High
140	Wlan.wep.key	Very Low	Nil	Low
153	data.len	Very Low	Nil	Very Low

6.3.3.2. Impersonation and Flooding attack Classifier

The second subset of data comprises samples of "Impersonation" and "Flooding" attack from the original attack subset. The pre-processed attack dataset is filtered for these two classes to train binary classifiers. Table 6.11 shows the summary of the data subset with the number of test and train samples. In each attack group the dataset is also already cleaned during the pre-processing steps. Further, we discuss how the three classifiers perform on this data subset and how the classification algorithms learn from the features.

Table 6.11: Impersonation and Flooding Attack Data-Subset Class Instances

Class Name	Train Set Instances	Test Set Instances	Class Label
Impersonation	48,522	20,079	1
Flooding	48,484	8,097	0

Random Forest: Similar to the first subset of "Impersonation" and "Injection" attack samples, the first classifier trained on the current data subset is the Random Forest ensemble classifier.

Figure 6.36 provide details of the overall impact of various features on the classifier's decision making process. It shows that the most impactful feature for "Flooding" and "Impersonation" classes is feature 153. Feature 153 contains information about the data length of the packet. The second most impacting feature is feature 8 which contains information about frame length. Interestingly, these two particular features had almost no contribution in the decision making of Random Forest for "Impersonation" and "Injection" attack data-subset. The addition of "Flooding" samples causes feature 153 and feature 8 to contribute heavily implicate how the packet's data length and frame length are different from the average for "Flooding" samples. Also, feature 63 and 65 had very low contribution in decision making of Random Forest in the absence of "Flooding" attack samples but have high contribution with flooding attack data samples.

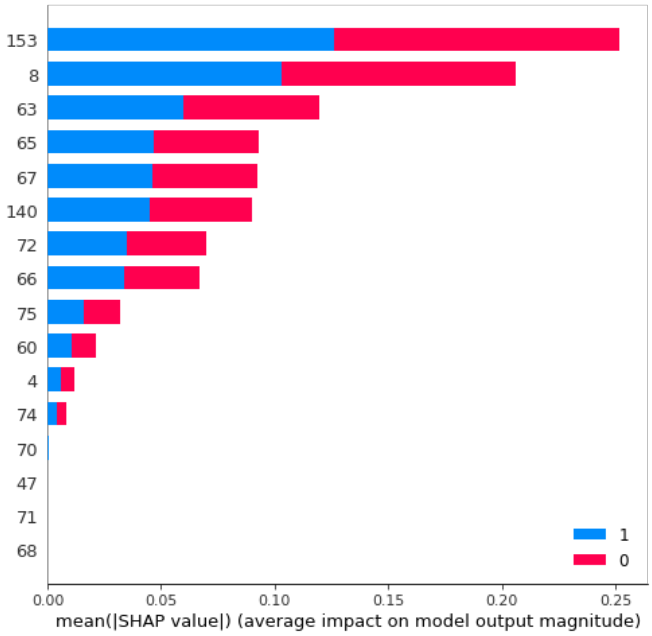

Figure 6.36: Average Feature Impact Classifier 2 (Random Forest)

Figure 6.37 shows the relationship of sample values and decision making of Random Forest in current dataset. Figure 6.37 shows the impact on class 0 (“Flooding” attack) with correlation to feature value of each sample. The lower values of feature 153 have a positive impact on the class 0 decisions of Random Forest. Similar trends can be seen with feature 8 where the lower values have a positive impact on class 0. Based on implications from Figure 6.37, data with low data length and frame length are most likely to be “Flooding” attack samples. Similarly, lower values of features 65, 67, 140, and 72 positively impact class 0 decisions of the Random Forest classifier. The low contributing features in the “Impersonation” and “Injection” data-subset stated are contributing in “Flooding” and “Impersonation” attack data subset in classifications. It show that some features are very specific to certain types of attacks.

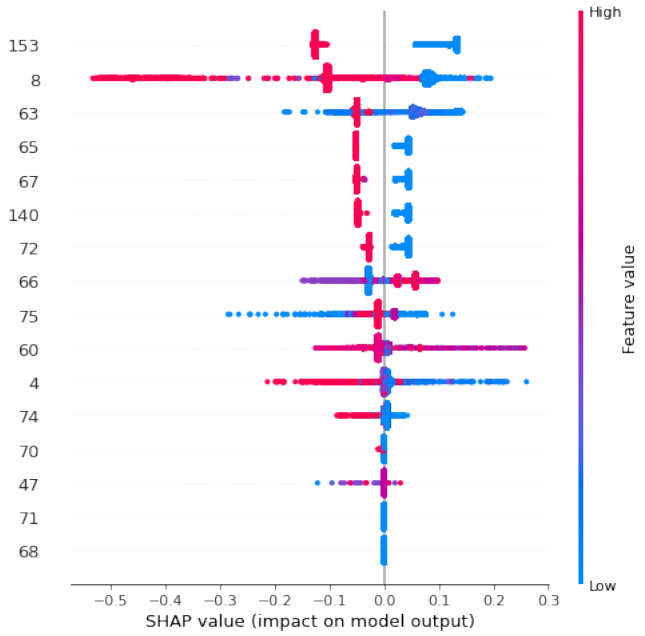

Figure 6.37: Feature Impact on Class 0 Classifier 2 (Random Forest)

Figure 6.38 shows the confusion matrix of the Random Forest classifier trained on the second attack data subset. The current data subset contains class 0 and class 1. The samples of Class 2 are not known to the current Random Forest classifier. The confusion matrix in Figure 6.38 shows that the Random Forest classifier has promising performance on both "Flooding" and "Impersonation" attack samples. Unlike with the "Impersonation" and "Injection" attack, the current Random Forest has much lower False Positives and True Negatives. Also, almost all the samples from unknown class (class 2) are being classified as class 1. Which means all "Injection" attack samples are being classified as "Impersonation" attack.

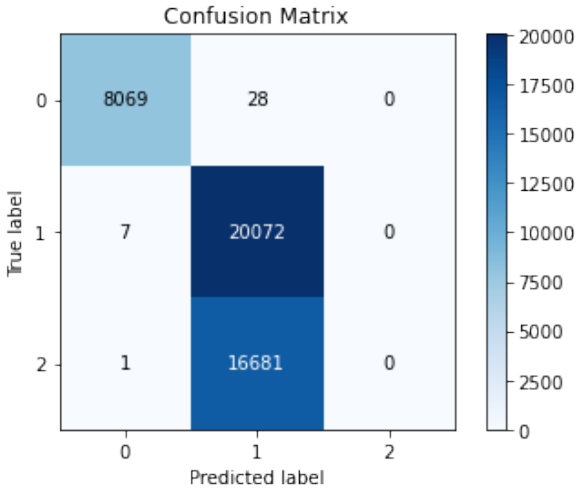

Figure 6.38: Confusion Matrix Classifier 2 (Random Forest)

Extra Tree: The second classifier tested of "Flooding" and "Impersonation" attack dataset is Ensemble based Extra Tree ensemble based classifier. Figure 6.39 shows the overall summary of various features with their level of contribution towards the classification. Similar to Random Forest, feature 153

remains the most impactful feature in the Extra Tree classifier's decision-making. However, feature 8 contributes comparatively less for Extra Tree than Random Forest. In addition, feature 70 that holds the information about power management was not being used by Random Forest, is seen to be contributing much more in this case. On the other hand, feature 140 that was seen contributing highly to Random Forest's decision making is not being used by the Extra Tree classifier.

The Extra Tree classifier was not using all the features in the "Impersonation" and "Injection" attack data-subset. The features that were not contributing in the Extra Tree classifier for first data-subset

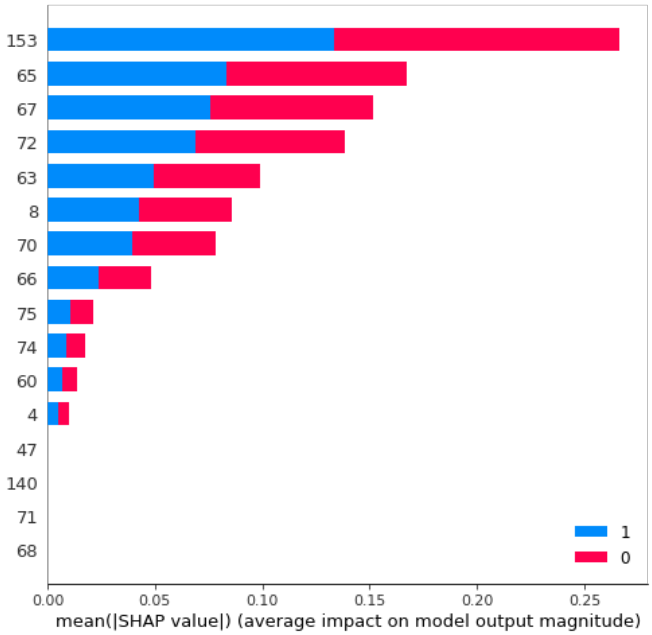

Figure 6.39: Average Feature Impact Classifier 2 (Extra Tree)

Feature contribution of Extra Tree and Random Forest have many differences shows how different each model learn from the different features. Figure 6.40 shows more details on the learning of the Extra Tree classifier from the data-subset features by highlighting the feature's value impact on the classifier's decision making.

The range of positive and negative values of the features impacting the classifier's decision is similar to the Random Forest classifier. Feature 153 lower values have a positive contribution towards the "flooding" attack and favours the "Impersonation" attack with higher values. Feature 8 had a high contribution in Random Forest and is contributing comparatively lower in Extra Tree. However, similar to Random Forest, feature 8 contributes positively towards "Flooding" with lower valued samples and favours "Impersonation" attack samples with higher values. Also, feature 140 had a high contribution for the Random Forest classifier of the second data subset which was not being used by the Extra Tree classifier on first data subset. In the case of Random Forest, the lower valued samples of feature 14 contributed positively to class 0 decisions. However, it does not show any effect on Extra Tree classifiers decision making. Other than feature 140, Extra Tree also does not use features 47, 71, and

68. Similar behaviour was seen in the decision making process of Random Forest for classifying samples with “Impersonation” and “Flooding” attacks.

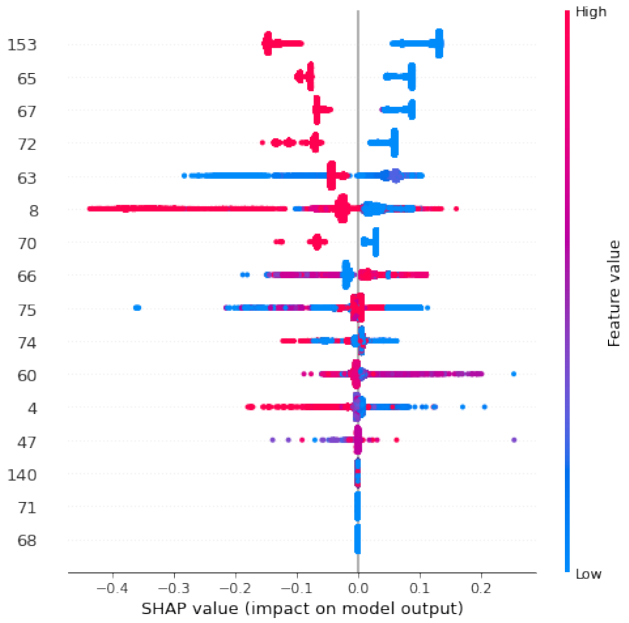

Figure 6.40: Feature Impact on Class 0 Classifier 2 (Extra Tree)

Figure 6.41 shows the confusion matrix of the Extra Tree classifier trained on the second data subset. The test-set samples are fed to the classifier, and the confusion matrix is calculated on predicted and actual samples. As shown in Figure 6.41, the Extra Tree classifier for class 0 and class 1 (“Flooding” and “Impersonation”) performs well on the test set. It has very low true negatives and false positive for both “Flooding” and “Impersonation” attack samples. However, compared to Random Forest, the True negative for “Flooding” attack (class 0) samples are higher. The Extra Tree shows better performance in classifying “Impersonation” attack (class 1) samples. Similar to Random Forest, the unknown class 2 (“Injection” attack) samples are labelled as class 1.

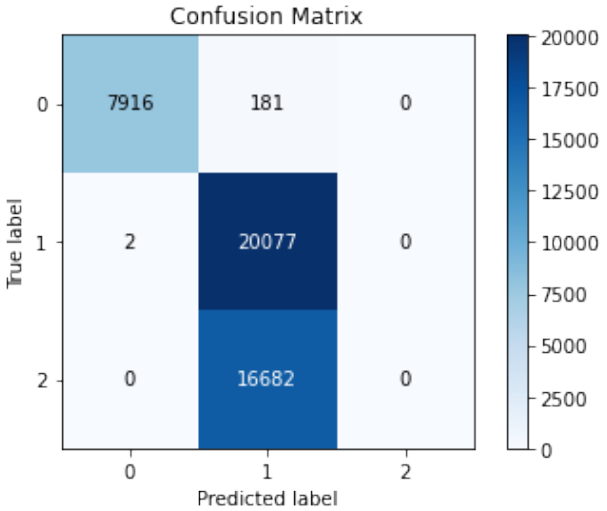

Figure 6.41: Confusion Matrix Classifier 2 (Extra Tree)

Neural Network: The third classifier trained on the second subset of data is Neural Network. The two Ensemble based classifiers trained on "Flooding" and "Impersonation" attack dataset have shown promising results. The Neural Network is fed the same data to learn. The Neural Networks shows a different learning behaviour, as shown in Figure 6.42. It shows a different level of contribution each feature is showing towards the learning process of the model. Unlike the ensemble methods, the Neural Network decision making is highly impacted by transmission duration information of the data packet contained in feature 74. Feature 74 had a low contribution in the decision making of Extra Tree and Random Forest Classifiers. Neural Network gives high importance to the receiver address in the decision making. On the other hand, the ensemble method's decision making does not learn from the receiver's address.

Similar to the previous modelling of "Impersonation" and "Injection" using Neural Network, the current model also tries to learn from all the given features. Most of the features can be seen contributing fairly to Neural Networks decision making. On the other hand, ensemble methods have comparatively fewer high contributing features in decision making.

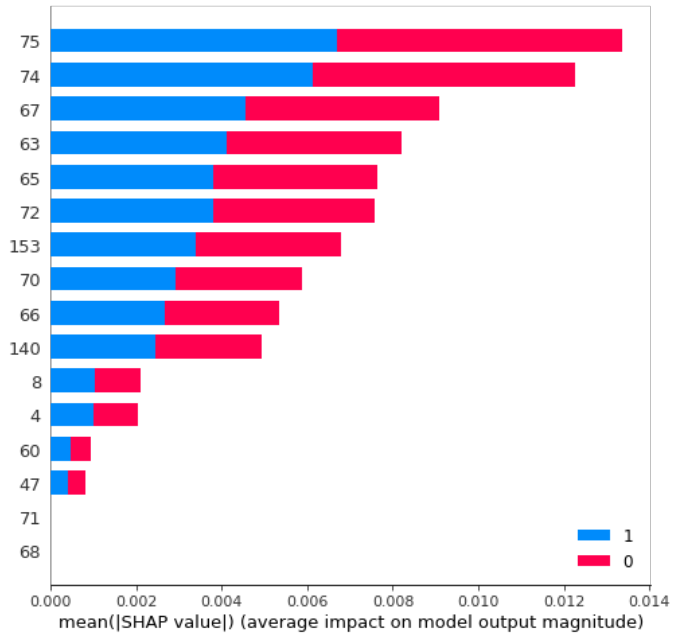

Figure 6.42: Average Feature Impact Classifier 2 (Neural Network)

Figure 6.43 has more details on feature contribution, how they impact class 0 decision making in Neural Network. Neural Network shows similar behaviour with respect to feature values and its decision making as ensemble methods. Despite of different feature contributions for each classifier, the relationship between the SHAP values and the feature values for each sample is the same across all the classifiers.

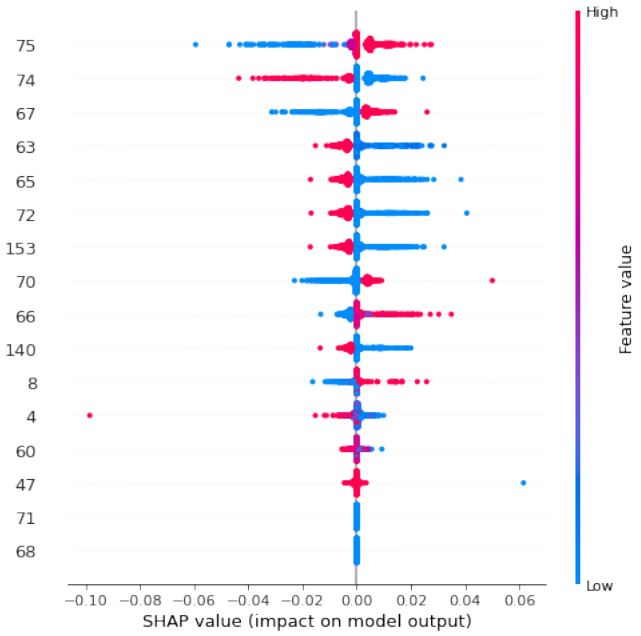

Figure 6.43: Feature Impact on Class 0 Classifier 2 (Neural Network)

The Neural Network's learning follows a different route in weighing the features to decide the output label. Figure 6.44 shows the confusion matrix for Neural Network trained on the second subset of data. The confusion matrix shows that Neural Network performs much lower in identifying class 0 and class 1 samples than Extra Tree and Random Forest classifiers. The Neural Network performs comparatively well on class 0 ("flooding" attack) samples. However, it has a high True Negative for class 1 samples ("Impersonation" attack). Class 2 is the unknown class to the Neural Network, the Neural Network classifies it as Class 1 similar to ensemble based classifiers.

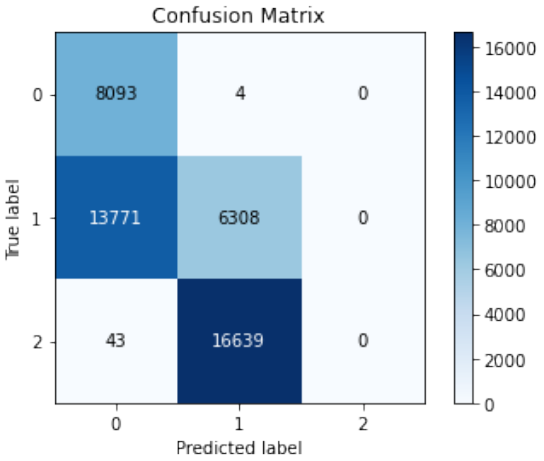

Figure 6.44: Confusion Matrix Classifier 2 (Neural Network)

The tree classifiers tested on the second dataset of "Impersonation" and "Flooding" attack samples. The performance of every classifier varies on the test set. Comparing the confusion matrix for all the three classifiers, it is evident that Random Forest performs best in classifying "Flooding" and "Impersonation" attacks. Neural Network performs the worst out of the three classifiers. It has very

high false positives for class 0. The Neural Network shows incapability in classifying the "Impersonation" attack samples.est impact for decision making.

Table 6.12 shows the summary of how the features are impacting all the classifiers. The table shows how all the classifiers are using features with different importance. In some cases, the features are not being used by any classifier. For the second dataset of "Impersonation" and "Flooding" samples, all the classifiers do not give much importance to the time delta between packets represented by feature 4. Also, several features are not being utilised by any classifier, i.e. feature 68 and feature 71. The captured frame length in feature 8 contributes high in ensemble methods, and Neural Network gives it less importance. Feature 65 that carries the frame control information, is being utilised by all classifiers.

Considering the performance of the three classifiers, the ensemble methods are performing much better, and the Random Forest classifier proves most capable in classifying "Flooding" and "Impersonation" attack samples. The Random Forest uses frame length and data length features with the highest impact for decision making.

Table 6.12: Feature Contribution Summary for Classifiers on "Impersonation" and "Flooding" Attack Data-Subset

Dataset Feature	**Feature Name**	**Random Forest**	**Extra Tree**	**Neural Network**
4	frame.time_delta	Low	Very Low	Low
8	frame.cap_len	Very High	Average	Low
47	radiotap.channel.freq	Nil	Nil	Very Low
60	radiotap.dbm_antsignal	Low	Very Low	Low
63	wlan.fc.type_subtype	High	Average	Average
65	wlan.fc.type	High	High	High
66	wlan.fc.subtype	Average	Average	Average
67	wlan.fc.ds	High	High	High
68	wlan.fc.frag	Nil	Nil	Nil
70	wlan.fc.pwrmgt	Very low	Average	Average
71	wlan.fc.moredata	Nil	Nil	Nil
72	wlan.fc.protected	Average	High	Average
74	wlan.duration	Low	Low	Very High
75	wlan.ra	Low	Low	High
140	wlan.wep.key	High	Nil	Average
153	data.len	Very High	Very High	Average

6.3.3.3. Flooding and Injection Attack Classifier

The third and final attack subset of the data is for "Flooding" and "Injection" attack samples. Table 6.13 shows the overview of the current subset of dataset. The "Flooding" class is labelled as 0 and "Injection" as 2. As the data is already pre-processed, it can directly be fed to the classification algorithm for training. In the third subset, the number of samples for "Injection" attack are higher than the "Flooding" attack samples. However, the bias is not very high as the class 0 samples are 42.5% of the totals samples. Since the imbalance is not too severe, class balancing is not considered.

Table 6.13: Flooding and Injection Attack Data-Subset Class Instances

Class Name	Train Set Instances	Test Set Instances	Class Label
Flooding	48,484	8,097	0
Injection	65,379	16,682	2

Random Forest: Following a similar pattern as before, Ensemble based method Random Forest is trained first on the third data subset. Figure 6.45 shows the summary of the Random Forest's learning from the features in the dataset.

The Random Forest classifier decision making is mostly impacted by the captured frame length stored in feature 8, followed by the data length values stored in feature 153. Comparing the learning of Random Forest in the second subset of "Impersonation" and "Flooding" attack samples, the feature impact on decision making for Random Forest is almost similar. The Random Forest gets highly impacted by the data length and frame length fields of the received packets in both cases. However, the case is different in the first subset of "Impersonation" and "Injection" attack samples.

The similarity of learning in the Random Forest classifier becomes clearer as the samples of "Flooding" attack are added to the data subsets. This indicates that the "Flooding" attack samples have difference in the properties of data from "Injection" and "Impersonation" attack, and the classifiers have clearer decision boundaries between "Flooding" and other attack samples in the dataset. Features for both the classes have clear difference, this means the quality of data is good and hence classification is effective i.e., the decision making.

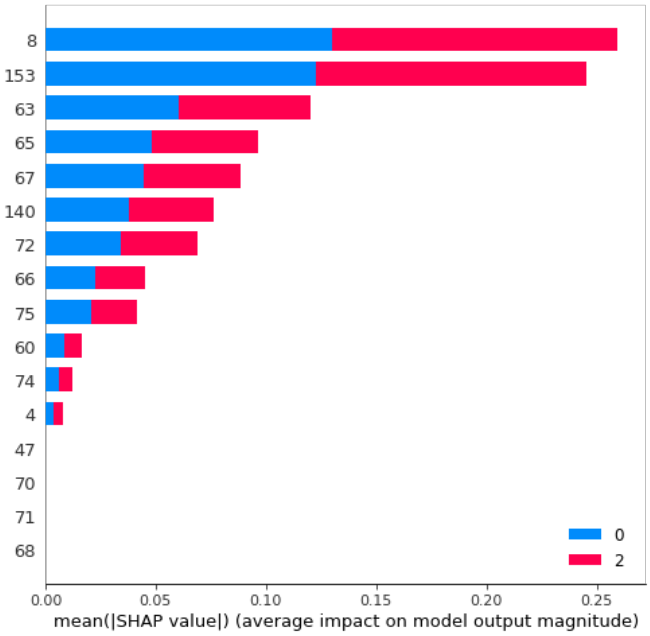

Figure 6.45: Average Feature Impact Classifier 3 (Random Forest)

Figure 6.46 has more details on feature contribution, how they impact class 0 decision making in Random Forest. It shows that the samples with smaller frame length and data length contribute positively towards class 0. The feature value and decision making correlation for Random Forest on the third data subset also similar to that for the second data subset. Other than the top two contributing features, feature 63, 65, 67, 140, 72 with lower values positively impact class 0.

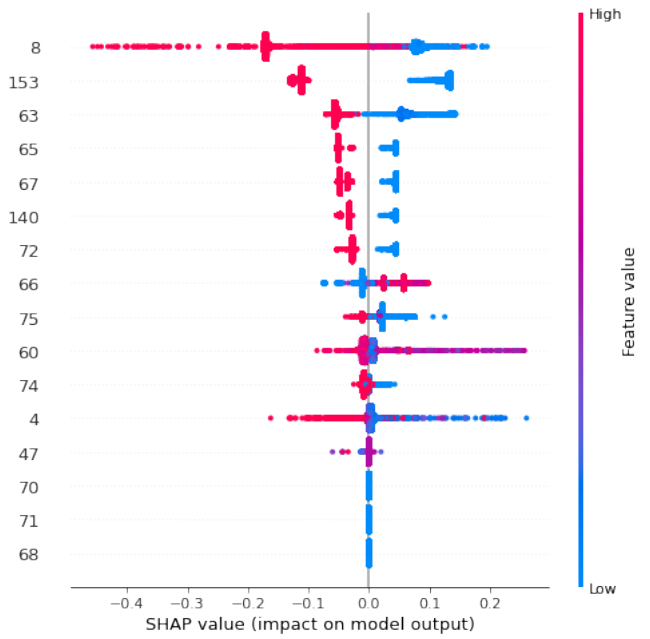

Figure 6.46: Feature Impact on Class 0 Classifier 3 (Random Forest)

Figure 6.47 shows the confusion matrix for the attack test set for Random Forest trained on the third data subset. In the case of “Flooding” and “Injection” attack samples (class 0 and class 2), the Random Forest shows exceptional results in the classification of both classes. The model performance in this

case is better than the Random Forest model while trying to classify Flooding" and "Impersonation" attack samples. It can be observed that most of the class 1 samples that were unknown to the classifiers are mostly classified as class 2. But there are several samples classified as class 0.

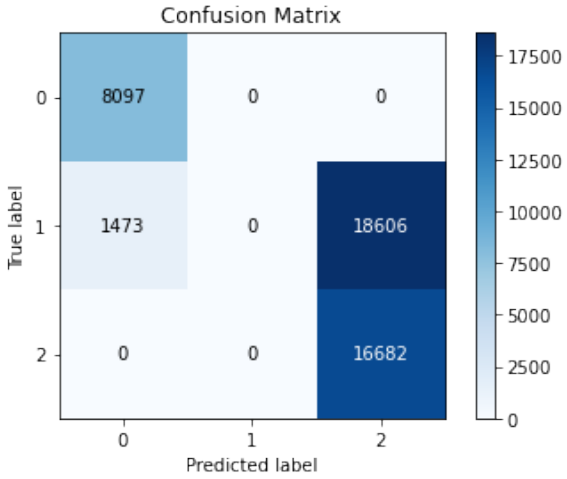

Figure 6.47: Confusion Matrix Classifier 3 (Random Forest)

Extra Tree: The second classifier trained on the third subset of data is the Extra Tree classifier. Figure 6.48 summarises the learning of the Extra Tree classifier from features in the third subset. The top contributing feature for Extra Tree is feature 153 which contains information about the data length of the captured packet. The Extra Tree classifier shows similarity in learning from features with the Extra Tree classifier on the second data subset. Both Random Forest and Extra Tree classifier's show similarity learns from the same features for their decision making process. "Flooding" attack samples. However, the Extra Tree classifier has does not find feature 140 to be containing distinguishing information which had a high contribution in the case of the Random Forest classifier. It uses feature 70, which was not utilised in the case of the Random Forest classifier.

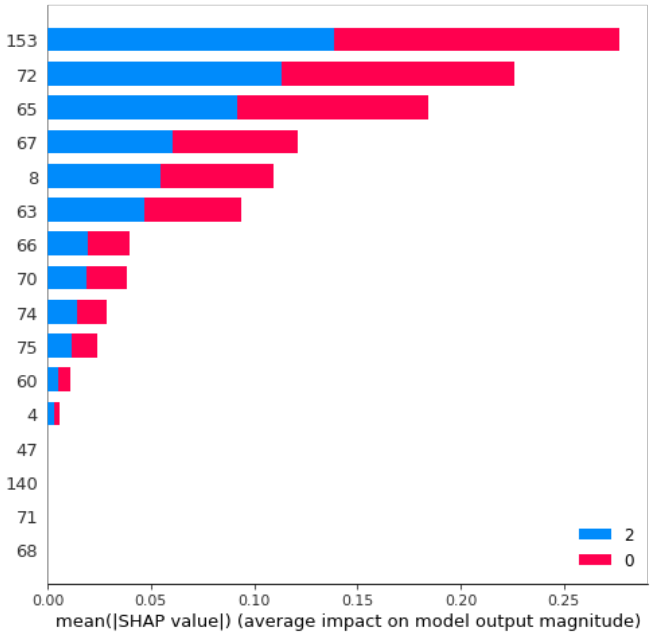

Figure 6.48: Average Feature Impact Classifier 3 (Extra Tree)

Considering details shown in Figure 6.49 on feature values impact on the class 0 ("Flooding" attack) decision making and comparing it with the Random Forest classifier shown in Figure 6.46 on third data-subset; the Random Forest classifier has a clearer decision boundary based on how it differentiates between the classes while learning from the features. For instance, feature 67 in Figure 6.46 shows that Random Forest finds a clear correlation where lower values positively impact class 0. Similarly the Random Forest shows clear learning from feature 75 values and how they impact the class 0 decision making as shown in Figure *6.46*: Feature Impact on Class 0 Classifier 3 (Random Forest)Figure 6.46.

As mentioned earlier, the Extra tree ignores feature 140 that had a high contribution in Random Forest. However, it uses feature 70 for decision making that Random Forest did not use, as shown in Figure 6.45 and Figure 6.46. Figure 6.49 shows that even though feature 70 contributes to decision making, the classifier uses all the feature values in favour of class 0 decision making. That means that the classifier does not find information to utilise feature 70 well in decision making.

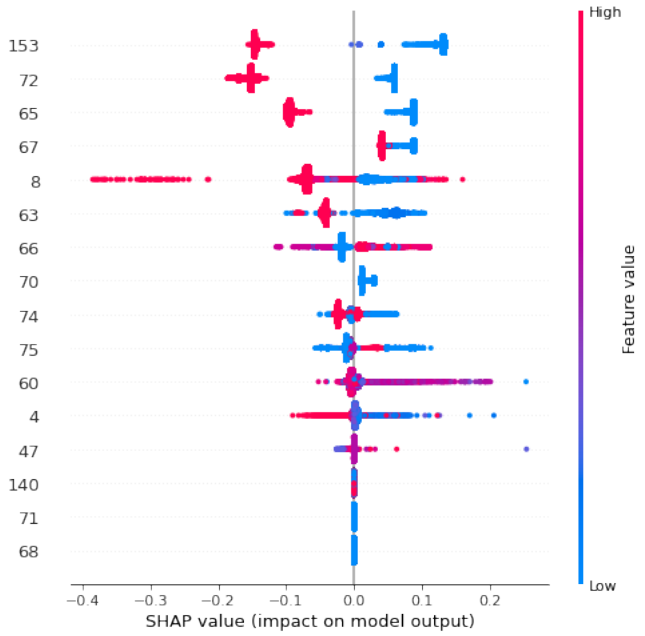

Figure 6.49: Feature Impact on Class 0 Classifier 3 (Extra Tree)

Figure 6.50 shows the confusion matrix for the trained Extra Tree Classifier on the test set. The performance of Extra Tree classifiers is comparatively lower than Random Forest classifier trained on the same data subset. The Extra Tree has comparatively higher false positives for class 0 samples. However, for class 2, they both perform the same. If we compare the performance on class 0 with Extra Tree trained on the second data-subset of "Flooding" and "Impersonation" attack samples, the current Extra Tree classifier has comparatively lower performance. As for class 1 that was not known to the classifier, both Random Forest and Extra Tree have the same results in the confusion matrix.

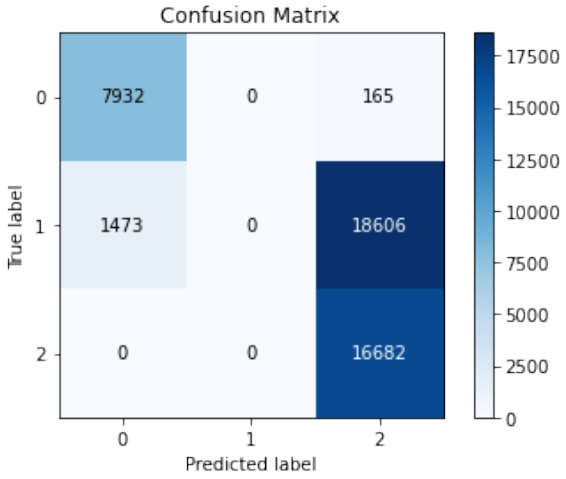

Figure 6.50: Confusion Matrix Classifier 3 (Extra Tree)

Neural Network: The Neural Network is trained after the ensemble methods to test how it performs on the third subset comprising "Flooding" and "Injection" attack samples. Figure 6.51 summarises the learning of the Neural Network from the features of the dataset. The contribution rank of features is different from ensemble based classifiers. The ensemble methods were affected mostly by the data

length or frame length of the captured packets. However, the Neural Network relies more on feature 75 and feature 74 containing information about the receiver's address and transmission duration, respectively. The Neural Network mostly utilises the features that were not contributing much in ensemble based classifiers. Also, the neural network utilises a higher number of features for decision making. The only feature that is not contributing in the case of Neural Network is feature 71. In the second subset, the Neural Network relies on more features and prioritises transmission duration over all the other features for decision making.

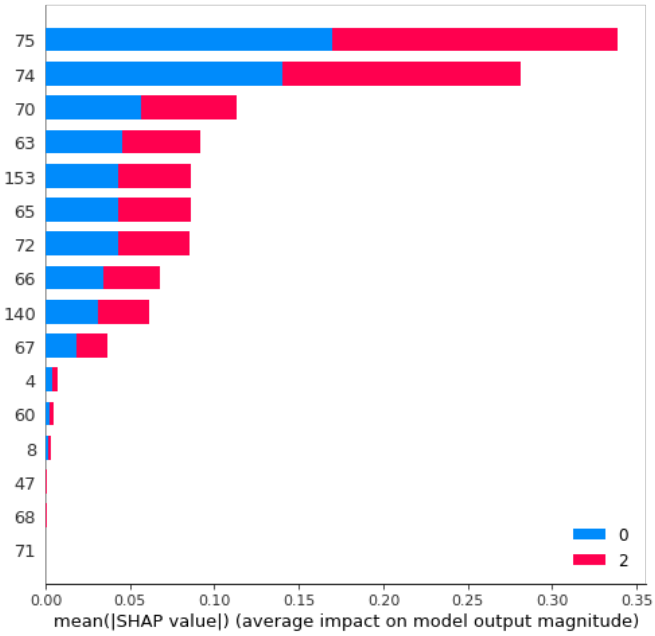

Figure 6.51: Average Feature Impact Classifier 3 (Neural Network)

Explaining the Neural Network further, Figure 6.52 shows details on how different feature values are impacting class 0 decisions. Features 75 and 74 on the top in Figure 6.52 shows that lower values for feature 75 and higher values of feature 74 contribute negatively for class 0. However, the Neural Network is not able to clearly distinguish between the feature values contributing positively to class 0 decisions. However, the Ensemble-based classifiers can differentiate how the values contribute to class 0 in both directions. As Figure 6.52 suggests, for most features in the case of Neural Network, it is not able to find how a feature contributes positively towards class 0.

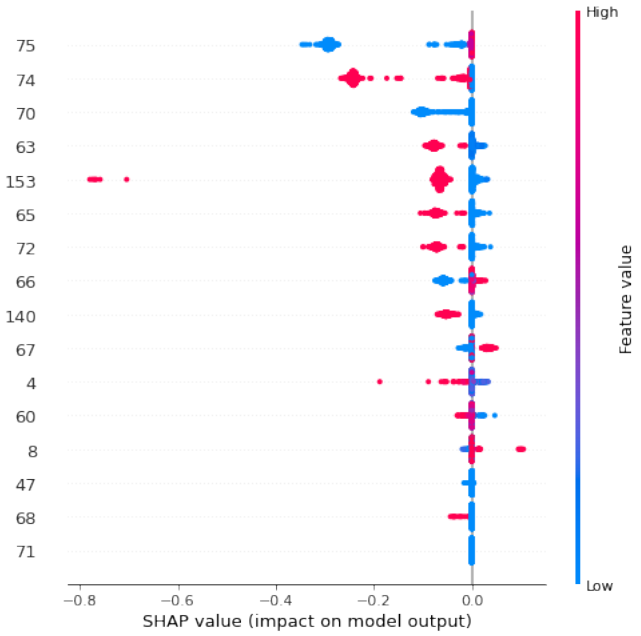

Figure 6.52: Feature Impact on Class 0 Classifier 3 (Neural Network)

As shown in Figure 6.53 as a confusion matrix on the test set. The Neural Network performs the same as the ensemble methods on the test set and has very high performance in detecting both class 0 and class 2 sample correctly. Moreover, for the unknown class 1 to the classifier, it shows the same results as ensemble based classifiers.

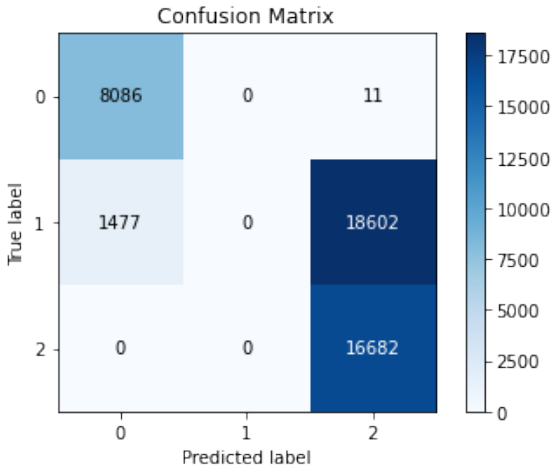

Figure 6.53: Confusion Matrix Classifier 3 (Neural Network)

Table 6.14 shows the summary of how the features contribute to the classifiers trained on the third data subset. The table shows how the three classifiers use features for their decision making. Feature 71 that store the value of frame control more data flag is not being used by any classifier trained on the third subset. Features 74 and 75 are not utilised by any ensemble classifiers trained. However, it contributes a lot to Neural Networks. Feature 153 and feature 8 is being utilised most by ensemble, but underutilised by the Neural Network. However, the performance of all the three classifiers is almost similar for the third subset.

Table 6.14: Feature Contribution Summary for Classifiers on "Injection" and "Flooding" Attack Subset

Dataset Feature	Feature Name	Random Forest	Extra Tree	Neural Network
4	frame.time_delta	Very Low	Very Low	Very Low
8	frame.cap_len	Very High	Average	Very Low
47	radiotap.channel.freq	Nil	Nil	Very Low
60	radiotap.dbm_antsignal	Low	Very Low	Low
63	wlan.fc.type_subtype	High	Average	Average
65	wlan.fc.type	High	High	Average
66	wlan.fc.subtype	Low	Low	Average
67	wlan.fc.ds	High	Average	Low
68	wlan.fc.frag	Nil	Nil	Very Low
70	wlan.fc.pwrmgt	Nil	Low	Average
71	wlan.fc.moredata	Nil	Nil	Nil
72	wlan.fc.protected	Average	High	Average
74	wlan.duration	Very Low	Nil	Very High
75	wlan.ra	Low	Low	Very High
140	wlan.wep.key	Average	Nil	Average
153	data.len	Very High	Very High	Average

6.3.3.4. All Attack Classifier

Once the classifiers are trained on each subset of data, finally, a multi-class classifier on all the attack classes is trained as a tie breaker classifier if all the classifiers classify the test samples to different classes and don't come to a majority agreement in the classifiers voting. Table 6.15 shows the composition of the attack dataset in AWID-CLS dataset, since the dataset is fairly balanced and was pre-processed, it is directly fed to the three multi-class classifiers trained on the attack dataset.

Table 6.15: Attack Dataset Class Composition

Class Name	Train Set Instances	Test Set Instances	Class Label
Injection	65,379	16,682	2
Impersonation	48,522	20,079	1
Flooding	48,484	8,097	0

Random Forest: The first multi-class classifier trained on the attack dataset is the Random Forest classifier. Figure 6.54 shows a summary of feature impact on Random Forest's decision making. It shows that Random Forest learns a lot from feature 153 that stores information about the data length of

the captured packet. This feature contributes highest towards class 0 and lowest for class 1. The second most contributing feature is 67 containing the information about the direction of the frame. Feature 67 is the least contribution to class 0 and mostly contributes to class 1. The figure shows some features like 74 and 70 that only contribute for class 1 and 2 and not for class 0. Feature 8 that stores information about frame length has not contributed to class 1. And out of all features, four features are not being used by the classifiers for the decision making.

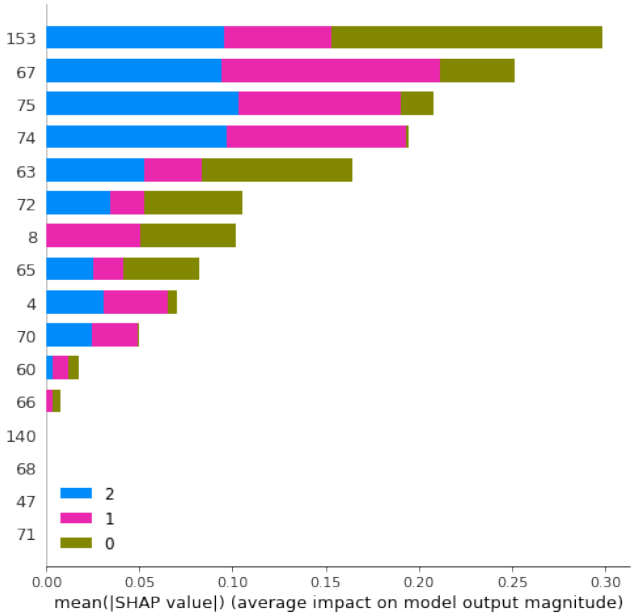

Figure 6.54: Average Feature Impact Multi-Class Random Forest

Figure 6.55 shows further details on how the features contribute to each class in the multi-class Random Forest classifier. Figure 6.55(a) describes the feature impact on class 0 during the training process. It shows that feature 153 has a high positive impact on class 0 when the values are lower. However, in most cases, the higher values of the feature push the decision making towards other classes. The same is the case with feature 63. It can also be observed in Figure 6.55(a) that a few samples with values for feature 153 and 63 are contributing positively for class 0, which can cause difficulty in generalisation of classifier for class 0. The higher values of feature 8 have a very high negative impact on class 0. The other features have comparatively lower contribution in class 0 decision making.

Figure 6.55(b) shows learning for class 1. Feature 67 with high values have a positive impact on class 1 decision making. The second most contributing feature is feature 74 that contains the information about the transmission duration of the packet. The lower values of the transmission duration influence the Random Forest classifier towards class 1. Feature 153 contributing the most for class 0 shows a mixed correlation between values and the influence for class 1. Feature 4 that stores the frame time delta, also shows a clear relationship between its values and contribution for class 1. The higher values of the time delta influence the classifier towards class 1.

Figure 6.55(c) shows the information on how Random Forest learns for class 2. In the case of class 2 the Random Forest is not utilising many features. It has detected a correlation between class 2 and feature 75 that contains the information about the receiver's address of the packet. The second most contributing feature is 74, which is common for class 1 and class 2. However, the correlation with class 2 and feature 74 values is the opposite of class 1. Feature 153 also contributes to class 2. However, the correlation of values and class 2 shows similarity with class 1. Figure 6.55(c) shows that the classifier cannot clearly draw a decision boundary for class 2. It struggles to find a clear correlation between feature values and class 2 and does not utilise most of the features for class 2.

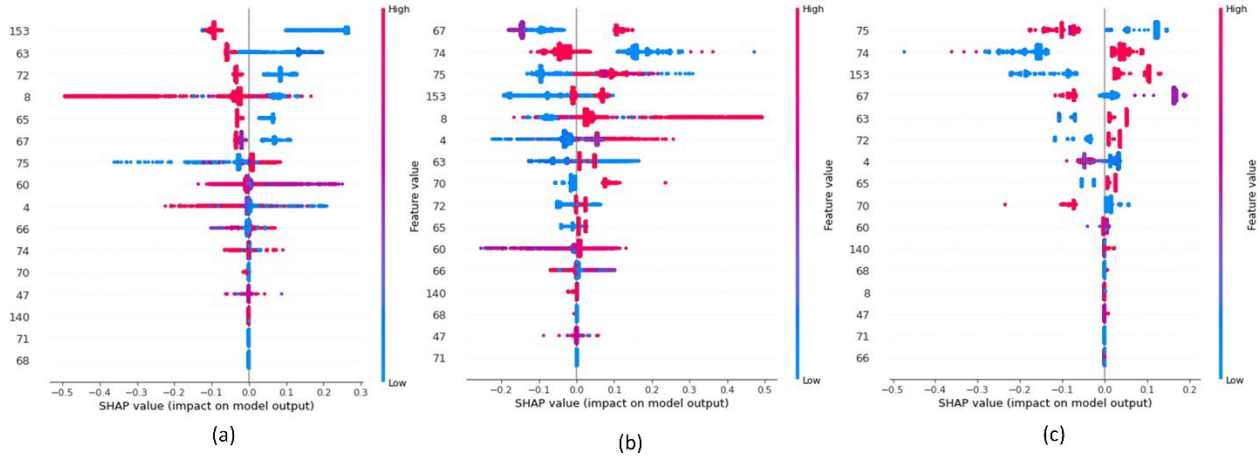

Figure 6.55: Feature Impact Multi-Class Random Forest for Attack Classes (a) Class 0 (b) Class 1 (c) Class 2

After training the Random Forest on the attack dataset, it was validated on the attack test set. Figure 6.56 shows the confusion matrix for the multi-class Random Forest classifier. Figure 6.56 shows that it performs very well on class 0 with very low false positives and true negatives. It means that "Flooding" attack samples are very well detected.

Similarly, for class 2, it classifies the samples correctly most of the time. However, in the case of class 1, it has a very high true negative where class 1 samples are misclassified as class 2. This implicates that when the Random Forest is trained on all the attack classes, it struggles to differentiate between "Impersonation" and "Injection" attack samples.

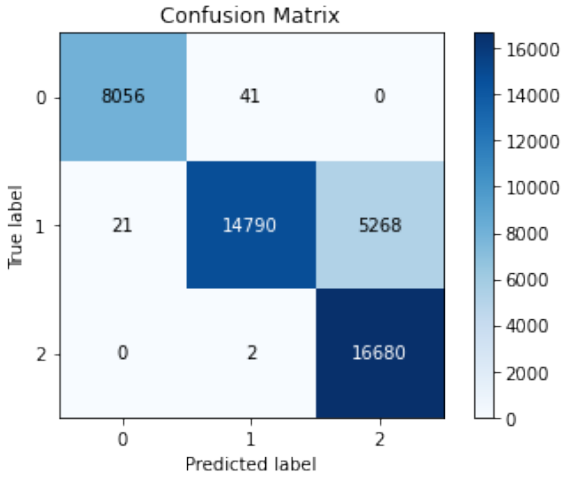

Figure 6.56: Confusion Matrix Multi-Class Random Forest Classifier

Extra Tree: The second multi-class classifier trained on the attack dataset is the Extra Tree classifier. Figure 6.57 summarises how Extra Tree learns from features for all the classes. In the case of Extra Tree, feature 74 is the most influential factor in its decision making. It mostly impacts class 1 and class 2. It has a very low contribution for class 0, which means that Extra Tree uses transmission duration values to differentiate between "Impersonation" and "Injection" attack samples. Unlike Random Forest, Extra Tree has not given much importance to Feature 153. The second most contributing feature is the same for both Extra Tree and Random Forest, i.e., feature 67. Features 72 and 65 are the third and fourth most influential features. However, they contribute higher towards class 0 and class 2 and have less contribution for class 1.

Several features only contribute to the learning of class 0 and class 2, for example, features 140, 60, and 4. Feature 8 has a very low influence on class 2 decision making. It is evident from Figure 6.57 that the features 153, 63, 65 and 72 have a comparatively stronger influence on class 0 and class 2. However, class 1 has a lower contribution by most features in the attack dataset.

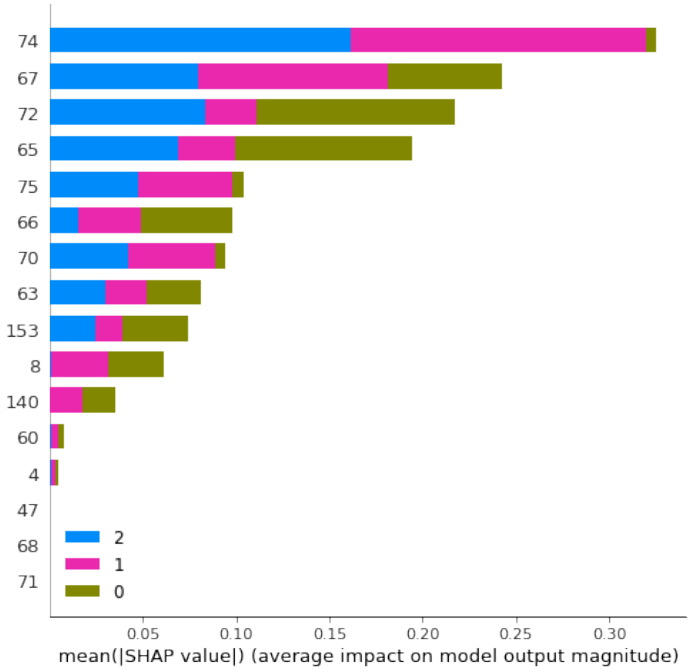

Figure 6.57: Average Feature Impact Multi-Class Extra Tree Classifier

Figure 6.58 shows how feature values influence the decision making of the Extra Tree classifier for the three classes available in the attack dataset. Figure 6.58(a) shows the feature value impact on the model output of class 0. For class 0, features 72, 65, and 67 are the top three influencers. The model can find the relationship between the feature value and how they influence the classifier's output for class 0. The top three contributing features have a positive impact on the class 0 decision with lower values. Feature 153 also shows a clear relationship with values and class 0 decisions as the lower values of the feature 153 have a positive impact on class 0 outputs. For all the other features, the model does not find a relationship between class 0 and feature values as shown in Figure 6.58(a).

Figure 6.58(b) shows how Extra Tree relationship between feature values and SHAP values for class 1 during the training process. The most noteworthy feature for class 1 is feature 74 with information about the transmission duration of the captured packet. The lower duration packets have a positive impact on class 1. In other words, packets with lower transmission duration have a high probability of being an "Impersonation" attack sample in the dataset. The second most influential is feature 67 with higher values mostly positively contributing to class 1. Other than the top two contribution features, the higher values of features 70, 4, 140, and 153 have a positive impact on class 1 output. The rest of the features do not show a clear correlation between their values and class 1 output of the Extra Tree classifier.

The Extra Tree Classifier's learning for class 2 is shown in Figure 6.58(c). Feature 74 is the most influential feature and shows a clear relationship with class 2. The higher values of feature 74 influences positively to class 2. The second most influential feature 72, also shows the same relation with class 2 decisions. The Extra Tree fails to use many features of the dataset in class 2 decision making and relies

only on a few of them. In some cases, like feature 67, 60, 4, and 140 the relation to the feature values is unclear to the model output as shown in Figure 6.58(c). Similar to multi-class Random Forest trained on attack dataset, Extra Tree also is not effective in creating a decision boundary for class 2.

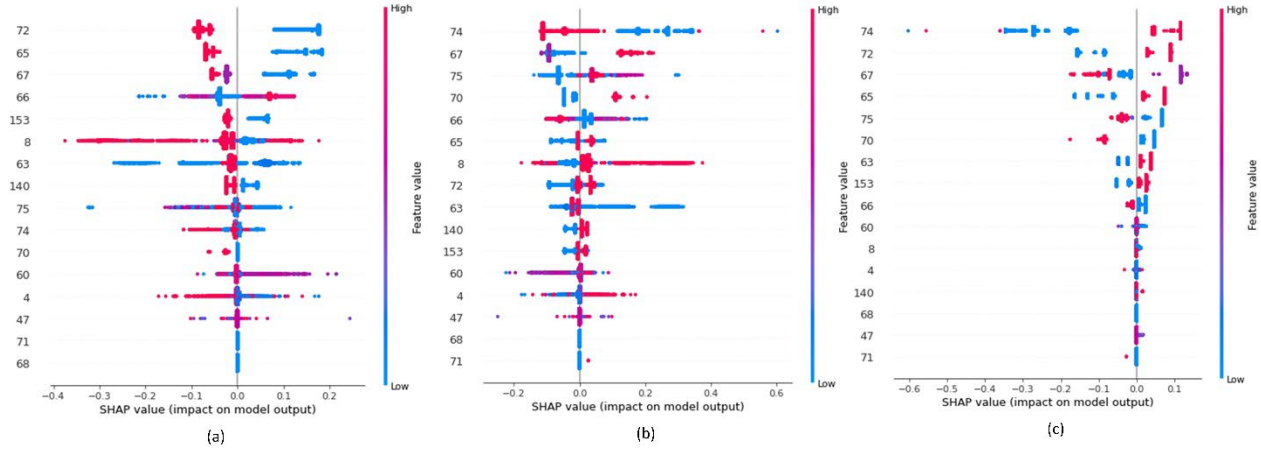

Figure 6.58: Feature Impact Multi-Class Extra Tree for Attack Classes (a) Class 0 (b) Class 1 (c) Class 2

Figure 6.59 shows the confusion matrix of the multi-class Extra Tree classifier on the attack test set. In comparison to Random Forest, Extra Tree has lower performance for class 0. It has a higher number of samples incorrectly identified as class 1. For class 1 and class 2, both Random Forest and Extra Tree have similar performance. They both struggle to correctly identify the class 1 samples as they have a high True Negative for class 1.

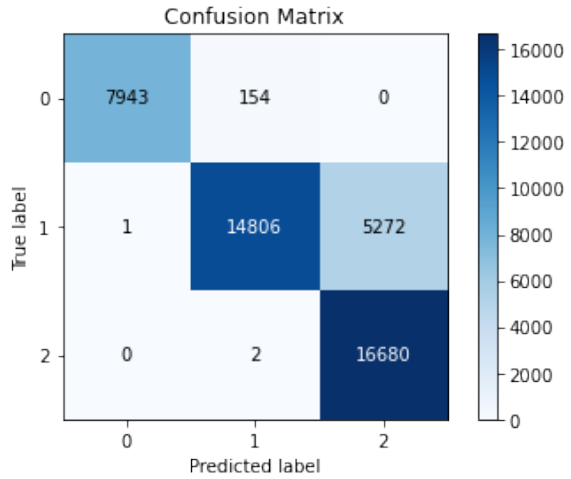

Figure 6.59: Confusion Matrix Multi-Class Extra Tree Classifier

Neural Network: The multi-class Neural Network is trained after the ensemble methods on the attack dataset to explore the learning and performance of Neural Networks on the attack dataset. Figure 6.60 show the summary of feature contribution for the attack classes in the dataset. The Neural Networks are utilising more features for decision making than the ensemble based methods.

The highest contributing feature 8, containing the values for captured frame length has a very low contribution in class 2 decision making. However, it highly contributes to class 1 and class 0. The second most influential feature is feature 74 that has information about the transmission duration of the packet. However, the difference between the impacts of the top two contributors is huge, implicating that the Neural Network has a high reliance on feature 8 for decision making, and other features have comparatively lower contribution. Also, most of the features have a low contribution for class 2 decision making. Meaning like ensemble methods, Neural Network also finds it difficult to detect pattern for class 2 samples.

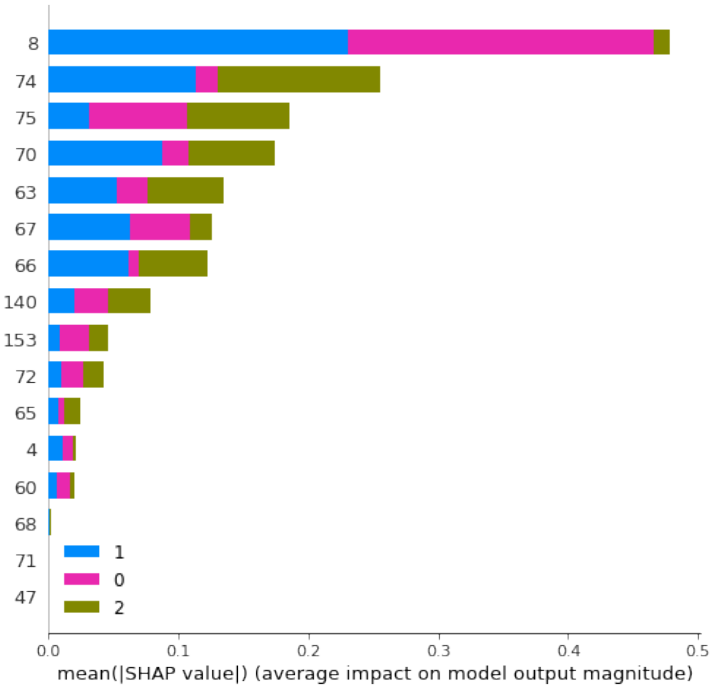

Figure 6.60: Average Feature Impact Multi-Class Neural Network Classifier

Figure 6.61 shows further details on how feature values impact decisions of all the classes. Figure 6.61(a) shows how the Neural Networks is learning for class 0 from the feature values in the dataset. Class 0 is getting the most influenced by the values of feature 8. The higher values of feature 8 have a positive impact on class 0. The neural network finds a high correlation between the receiver's address and class 0 in feature 75. Neural Network shows a higher inclination towards the top 3 features and consumes the other features with low impact.

Figure 6.61(b) shows the learning of Neural Network for class 1. The highest contributing is feature 8, and its higher values have a positive impact on class 1. The second most influential feature is 74 that contains information on the transmission duration of the packet. The lower values of feature 74 have a positive influence on class 1 decision making. Feature 70 also shows how its high values positively impact class 1 decisions of Neural Network.

Figure 6.61(c) shows the learning of Neural Network for class 2. The highest impact is made by feature 74 in the case of class 2. The higher values of feature 74 influences the Neural Network decision towards class 2. The second highest is feature 75, containing the receiver's address.

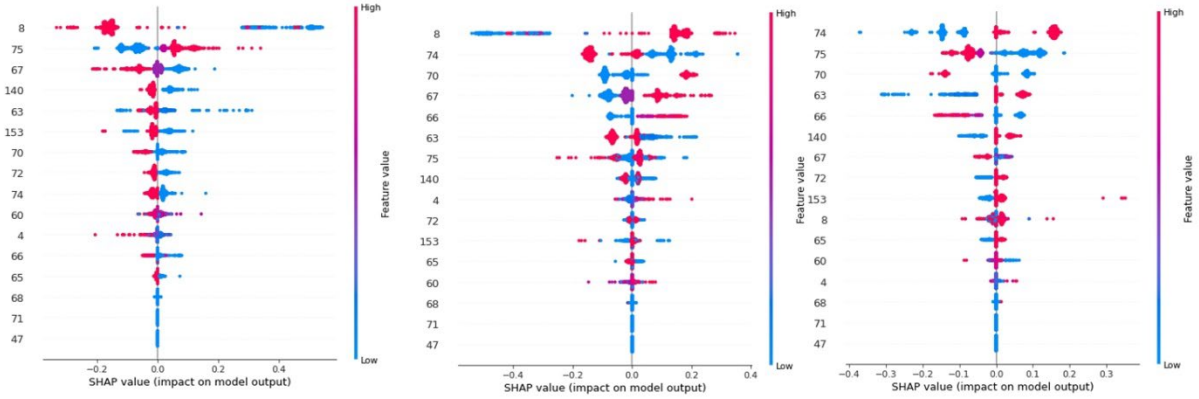

Figure 6.61: Feature Impact Multi-Class Neural Network for Attack Classes (a) Class 0 (b) Class 1 (c) Class 2

Figure 6.61 shows that Neural Network finds a clear correlation for all the features in each class output. However, it relies on few features highly while underutilising most of the features for decision making for each class. Figure 6.62 shows the confusion matrix on the attack test set for Neural Network. The Neural Networks performance is comparatively lower than the ensemble methods. It is evident from the confusion matrix that Neural Network struggles in differentiating between the different attacks classes in the dataset. The confusion matrix shows that the best performance of Neural Network is for class 0. However, it still has comparatively higher False Positives and Negatives for class 0 than Extra Tree and Random Forest trained on the same dataset. For class 1 majority of the samples are misclassified as class 2, which means that Neural Network poorly performs on creating decision boundary for class 1 and class 2. The poor performance of Neural Network is caused as it fails to learn any patterns from the features for different attacks. Figure 6.60 shows that it relies on very few features heavily and underutilises most of the dataset feature. The top contribution features in the Neural Network do not show patterns for some classes in the dataset causing instability in the decision making.

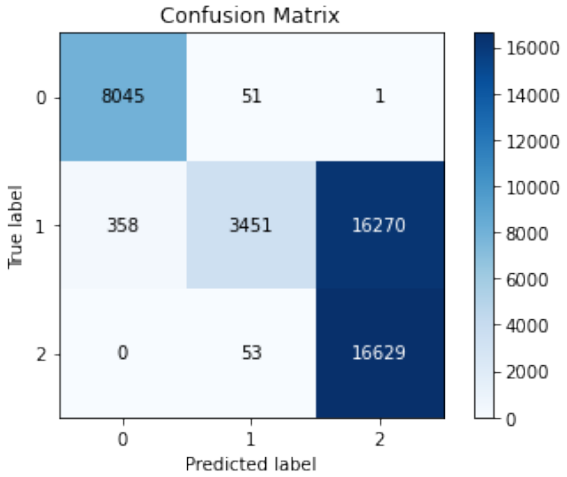

Figure 6.62: Confusion Matrix Multi-Class Neural Network Classifier

The Multi-class classifiers have to learn for all the classes in the dataset simultaneously from all the available samples. In the learning process of multi-class classifiers, there are possibilities that samples of one class can have an influence on the other classes. Because of noise caused by samples belonging to a particular class, the multi-class classifier can struggle in finding optimal decision boundaries for all the classes.

Table 6.16 summarises how each classifier utilises the features for their decision making. All multiclass classifiers have shown a pattern in the receiver's address attribute stored in feature 75 by showing high utilisation. The other feature that has a strong influence on the decision making of all the classifiers is feature 74 with values of transmission duration of the captured packets. Considering the confusion matrices from Figure 6.56, Figure 6.59, and Figure 6.62, it is evident that the ensemble methods perform much better with their learning approach than the Neural Network. Both Extra Tree and Random Forest classifiers have almost similar performances on the attack classes.

Table 6.16: Feature Contribution Summary for Multi-Class Classifiers on Attack Dataset

Dataset Feature	**Feature Name**	**Random Forest**	**Extra Tree**	**Neural Network**
4	frame.time_delta	Low	Very Low	Low
8	frame.cap_len	Average	Low	Very High
47	radiotap.channel.freq	Nil	Nil	Nil
60	radiotap.dbm_antsignal	Very Low	Very Low	Very Low
63	wlan.fc.type_subtype	High	Average	Average
65	wlan.fc.type	Average	High	Low
66	wlan.fc.subtype	Very Low	Average	Low
67	wlan.fc.ds	Very High	High	Average
68	wlan.fc.frag	Nil	Nil	Low
70	wlan.fc.pwrmgt	Low	Average	Average

Dataset Feature	Feature Name	Random Forest	Extra Tree	Neural Network
71	wlan.fc.moredata	Nil	Nil	Nil
72	wlan.fc.protected	Average	High	Low
74	wlan.duration	High	Very High	High
75	wlan.ra	High	Average	High
140	wlan.wep.key	Nil	Low	Low
153	data.len	Very High	Average	Low

6.3.3.5. Consolidated Voting Classifier

The concept of dividing the dataset into three sub-sets of two classes each and then training separate binary classes on each subset of the dataset simplifies the learning of classifiers. The binary classifiers show a better tendency to create a decision boundary as only two types of samples are available in the dataset subset. Since the third class is not available, its samples are not hindering the decision making for other classes.

The consolidated voting mechanism combines three previously discussed binary classifiers trained on all three subsets of data and a multi-class classifier trained on all the attack classes available in the dataset. Four classifiers using same machine learning algorithm are brought together in the voting mechanism where three are binary classifiers trained on different subsets of data containing two attack classes and fourth is a multiclass classifier trained on all attacks in dataset.

The attack test set is passed to all the binary classifiers trained for the consolidated voting mechanism. Each classifier makes its decisions and votes on the decided class. The class that receives the most votes is projected as the final decision of that sample by the consolidated framework. Suppose a class does not receive a mandate and all binary classifiers vote for different classes. In that case, the multiclass classifier is used to break the tie and finalise the decision of the voting framework. Figure 6.63, Figure 6.64, and Figure 6.65 show the confusion matrix of Random Forest, Extra Tree, and Neural Network consolidated voting classifier on the attack test set.

The confusion matrix for all the three voting classifiers achieves the same results on the attack test set. The consolidated voting approach shows impressive results in classifying all the available attack classes in the attack dataset. The consolidated voting by using multiple binary classifiers on the data subset proves to be much more effective in comparison to the multiclass classifiers trained on attack dataset discussed in sub-section 6.3.3.4.

Interestingly, all the classifiers achieve the same results when clustered in proposed consolidated voting mechanisms irrespective of their differences in their learning and results on the test set. It proves that

the three binary classifiers, when put together, balances the shortcomings of each other and find an optimal decision boundary for each class in the attack dataset.

Figure 6.56, Figure 6.59, and Figure 6.62 show that all the classifiers struggle to classify "Impersonation" attack samples labelled as class 1. The false positive for the "Impersonation" attack is high for all of the multi-class classifiers. The results from Figure 6.56, Figure 6.59, and Figure 6.62 prove that the detection of the "Impersonation" attack is more challenging than the other two attack classes in the dataset. However, the consolidated voting mechanism shows very low false positives for "Impersonation" attack samples. It shows high competence in detecting "Impersonation" attacks that are otherwise challenging to identify by all the three multi class classifiers accurately on their own.

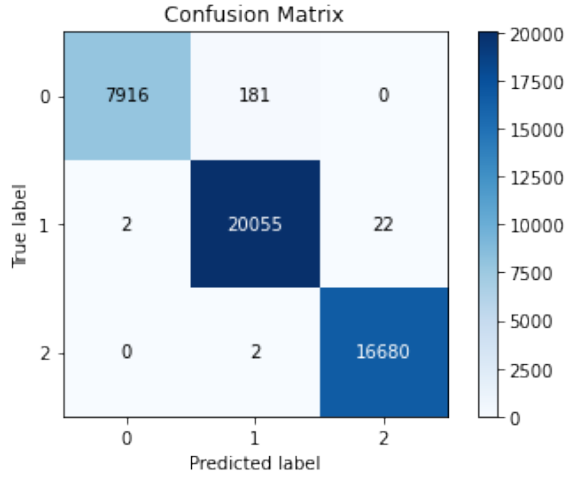

Figure 6.63: Confusion Matrix for Random Forest Classifier with Consolidated Voting

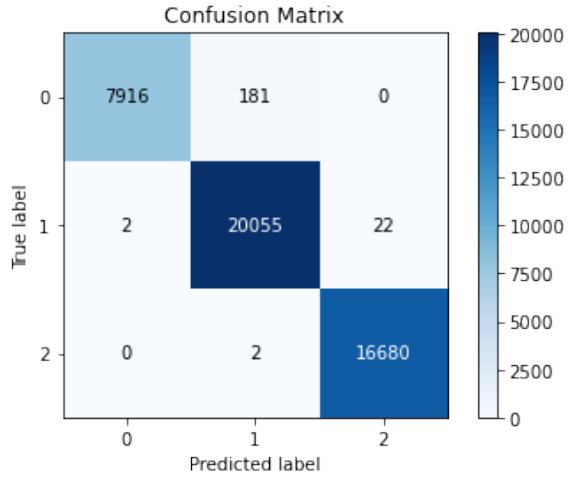

Figure 6.64: Confusion Matrix for Extra Tree Classifier with Consolidated Voting

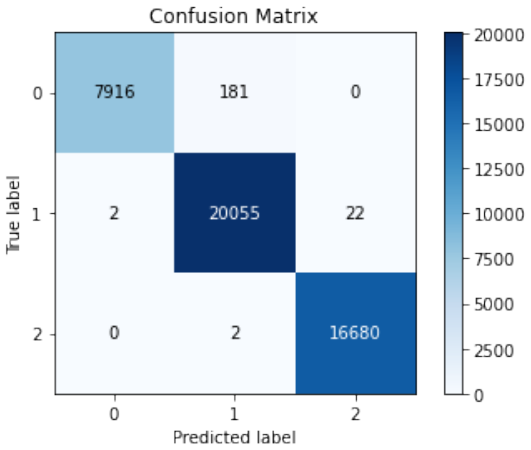

Figure 6.65: Confusion Matrix for Neural Network Classifier with Consolidated Voting

Chapter 7

Conclusion and Future Work

This thesis explores the security mechanisms available in LPWAN communication technologies used in IoT networks. The literature shows that LPWAN networks use static keys for data authentication and confidentiality and create vulnerabilities in LPWAN networks. The literature highlights that LPWAN network nodes are susceptible to physical attacks. The attackers can gain the key information of the node by physically accessing the nodes, which creates a possibility of an impersonator in the network.

Considering the finding from literature, this thesis aims to propose a lightweight key exchange mechanism for LPWAN networks; and an effective attack detection mechanism that can help in early attack detection on LPWAN and IoT networks. The proposed framework explores the possibility of using the SDN framework for security in LPWAN networks. The proposed key exchange mechanism uses BYKA, which was designed for sensor networks. BYKA key exchange mechanism is extended to integrate with a centralised SDN controller and provide a session key mechanism. Furthermore, a two-tier machine learning based attack detection mechanism is proposed to defend against network attacks.

The proposed session key mechanism follows a communication flow between the server and the end nodes. To test the vulnerabilities in the communication flow of the session key mechanism, a security analysis tool "Scyther" is utilised. The communication flow is tested across various attack models available in Scyther. Considering the results of simulations, the proposed communication flow proves to have no vulnerabilities from the attack models available in Scyther. It also shows that the session key remains confidential in during communications for session key mechanism.

Achieving a secure data transmission is a vital feature for a network. However, in case of low powered networks, it is necessary to achieve the security while being energy efficient because of the limited energy available to the network nodes. Hence, to further validate the proposed session key mechanism, it is tested on the NS3 simulator for its energy consumptions. The LoRaWAN energy model is modified, and energy modules for key exchange and encryption mechanism are added to the LoRaWAN energy model. The experiment is performed for worst case scenarios so that performance can be tested on worst case scenarios. The proposed session key mechanism proves to be energy efficient and show the possibility of a node lifetime of one and a half months with an 850mAH battery on a 3 Volt supply.

In addition, the proposed session key mechanism is implemented on Mininet-WiFi to validate its correctness. The Mininet-WiFi emulator is used with the POX controller to facilitate the session key

exchange in an SDN based wireless network. The experimentation proves the applicability of the SDN framework in the security of LPWAN and IoT networks.

The second module of the thesis was to provide a prevention mechanism against impersonation, flooding and injection attacks on IoT networks. AWID-CLS, a public dataset, is used to train and validate the proposed attack detection framework. A two-tier attack detection and profiling mechanism are proposed for attack detection on IoT networks. The proposed attack detection approach uses a binary classifier to differentiate between "attack" and "normal" traffic in tier-1 and use multiple binary classes into a voting mechanism to profile network attacks on the network in tier-2.

There have been various encounters with challenges in the research and its implementations that led to alternate paths to peruse the outcomes and accomplish this study. Based on the outcomes of the proposed framework, this report is being concluded. However, there are outstanding issues and must be addressed in the future, as discussed in the next section.

7.1. Research Contributions

This thesis focuses on the research questions derived from the literature review. Three research questions were derived from the gaps identified from the literature, and the thesis aims to contribute around the three research questions. The major contributions of the thesis are as follows:

7.1.1. An SDN based Lightweight Session Key Mechanism for an IoT Network

The first research question states, "How can we design an SDN based lightweight authentication and session key exchange framework for IoT nodes?" The literature highlights the requirement of a lightweight session key mechanism as most of the transmission technologies in IoT use a single static key for node authentication and data encryption. The use of a static key over a long period is not recommended as it creates a possibility of the key being deduced from the data being transferred over the air. The literature also shows LPWAN transmission standards to be one of the most popular amongst IoT vendors. Hence, this thesis targets the LPWAN networks for the session key mechanism.

A lightweight session key mechanism using the BYKA scheme is implemented that exploit the star topology of the LPWAN. The proposed session key mechanism requires only one transceiver operation from the end node and uses lightweight mathematical operations for the session key calculations. The lightweight operations and minimal transmission for the end nodes facilitates a longer node lifetime in the network. The experimentation shows that the proposed session key mechanism is secure against all the attack models in Scyther security analysis tool and provides a correct calculation of the session keys. In addition, the simulations in NS3 show that even with a high data rate and frequent session changes, the proposed session key mechanism is energy efficient.

The proposed session key mechanism also shows the flexibility to tune the session lengths based on the application requirements and consumes similar energy as LPWAN standard technologies at a session length of 12 hours. The flexible nature of the proposed session key mechanism enables it to be used with any LPWAN standard technology and IoT applications respective to their requirements. Also, the proposed session key mechanism is implemented in an SDN controller that interacts with the network at a lower level (network layer), which provides more control over the network operations and monitoring. The capability of the SDN controller to monitor and control the network at a granular level can be utilised more in Identity theft detection, which is the focus of the second research question of the thesis.

7.1.2. Exploring the Opportunities of SDN based Identity Theft Detection

The BYKA scheme provides a strong and flexible session key mechanism for power constrained IoT networks. However, the BYKA key exchange mechanisms is vulnerable to identity theft or Impersonation attacks. The attackers can acquire the secret key information from the end nodes by accessing them physically and then impersonating a network node to transmit false data. The Impersonation attack breaches the integrity and data authenticity of the network. The proposed security framework aims to provide early detection and flexible actions for attack data and decides to utilise the SDN framework. The second research question states, "How can SDN controllers be used to detect node identity theft in the IoT network?".

For Impersonation attack detection, machine learning algorithms are utilised. A public dataset AWID-CLS is used as it contains the samples of impersonation attacks along with flooding and injection attacks on a wireless network.

The proposed attack detection is divided into two phases. The first phase is for gateway devices, where the classification between attack and normal traffic takes place. The first phase uses a simple and lightweight binary classifier that requires less memory and computation and can be deployed on an IoT gateway device. Based on the classification of the first phase, the SDN controller can decide to redirect to the network or phase two, where a consolidated voting mechanism profiles the attack. The experimentation shows that the proposed two-tier classification mechanism is highly efficient in detecting and profiling all the attacks, including impersonation attacks.

The use of the SDN controller provides the opportunity for early detection and filtering of attack traffic at the network layer. The SDN also provides granular control over traffic and can manage traffic based on a specific rule provided by network admin. The literature also suggests that using machine learning techniques with SDN can provide a flexible and efficient attack detection and prevention mechanism for IoT networks.

7.1.3. Trade-offs With Proposed SDN based session key and attack detection mechanism

The utilisation of SDN controller and the proposed security framework brings flexibility, control and better security for LPWAN based IoT networks. However, the addition of security and SDN controller also bring some trade-offs with the benefits. The third research question aims to analyse and discuss the trade-offs of the proposed security framework stating, "What kind of performance trade-offs may the nodes face with the addition of the new security scheme?". The proposed security framework aims to provide security to low powered IoT networks that are currently using a static key for authentication and data encryption. The addition of a security mechanism on top of the current standards brings some performance overheads and trade-offs. This thesis also provides analysis and discussion on those trade-offs in section 5.4. It is discussed that the implementation of the proposed security framework will bring additional processing overhead to the nodes as they will require to perform an additional transceiver and computational operations per session. Also, using an SDN controller as a key management authority and as an attack detection mechanism can bring some additional delays as the data needs to travel through the SDN controller. However, considering the advantages of the proposed security framework, it is argued that the trade-offs are affordable in exchange for higher security of the network.

In summary, this research addresses the following research questions:

1. How can we design an SDN based lightweight authentication and session key exchange framework for IoT nodes?

2. How can SDN controllers be used to detect node identity theft in the IoT network?

3. What kind of performance trade-offs the nodes may face with the addition of the new security scheme?

7.2. Research Limitations and Future Work

The proposed framework targets to provide a lightweight and robust security or LPWAN. However, there are a few limitations that should be taken under consideration for future work, discussed as follows:

7.2.1. Optimisation of and Implementation of Energy Aware Key Exchange Initiation from Controller

Power efficiency is one of the most important features that must be considered for security mechanisms in the IoT environment. The proposed session key mechanism can achieve that to an extent. However, as the result shows, the applications with high data rate need to have a better session management mechanism from the server side to regulate the power consumption of the end nodes. The proposed session key mechanism can be extended further to provide a more robust session mechanism that can

monitor the node status to their application requirements and remaining energy and manage the session length accordingly. Having flexible session lengths can provide more consideration to the energy consumption on the nodes to extend their lifetime. In addition to flexible session lengths, the servers can also implement a monitoring mechanism for node's duty cycles based on applications running on them. Having a monitoring mechanism for the duty cycle can enable the servers to initiate the sessions according to the duty cycle of the nodes, which can prevent the nodes from continuous listening to the channel. If the nodes receive data according to their duty cycle, they will not require additional transceiver operations for the session key mechanism.

7.2.2. Scalable and Robust Implementation of Key Exchange Mechanism

This thesis focused on providing a lightweight session key mechanism for LPWAN. However, the implementation provided in this thesis is a prototype and requires further refinements around efficient public and private key management. As the number of nodes increase, the vanilla implementations will start to face challenges. Further implementations around the reusability of the public key at SDN controller can facilitate support to large scaled IoT networks. In addition, a distributed approach can be followed where multiple SDN controllers can be utilised to support multiple subgroups of the network. Following a distributed architecture for the session key mechanism can provide load balancing on the SDN controller and reusability of public keys within subgroups.

7.2.3. Public Key Broadcast Optimisation

The proposed session key mechanism requires the server to initiate the session by broadcasting the public keys in the network. Any node that receives the public key can generate a session key for data encryption. The SDN controller also uses an application key to sign the public key to control the nodes participating in the session. However, as the network size increases, it can be challenging to manage nodes participating in session key generation. The network can be divided into multiple multicast groups to address the issue, and the SDN controllers can use multicasting instead of broadcasting to limit the number of nodes participating in a session. The implementation of multicasting can also help in node localisation and group nodes based on various features such as applications, data rate, and power supply.

7.2.4. Connectivity of SDN Controller and Controller Placement

The proposed framework leverages the functionalities of SDN to detect and block attacks at the network layer. At the same time, the SDN controller is also being utilised as a key management entity. The network relies on the SDN controller for all the security mechanisms. Hence, it is crucial to manage the connectivity of the SDN controller. The SDN controller placement can play an important role in achieving optimal connectivity to minimise delays in session key and attack detection mechanism. As it plays an important role for network devices, it should be close to the network devices. However, it

also needs to be secured physically as it also stores the network key information. An optimal location for the SDN controller needs to be calculated for the effective functioning of the framework.

7.2.5. Forward Secrecy

The proposed session key mechanism uses a secret key stored at the end node and public key information to generate the session keys. Given a situation where an attacker acquires the secret key of a node and all the public keys, they can read all the data transferred from that node. The proposed session key mechanism must be extended to provide features to update the stored keys in end nodes periodically to provide forward secrecy.

7.2.6. Consolidation of Key Exchange and Attack Detection Framework

The proposed framework has two modules, one performs the tasks related to the session key mechanism, and the other is for attack detection on the network. Both modules are validated on different simulators during experimentation as all aspects could not be validated on a single simulation tool. The machine learning models for attack detection are designed and tested on public data and require training on real network traffic. Both modules are required to be consolidated and deployed on an SDN controller in a real network testbed for them to work together effectively.

7.2.7. Attack detection mechanisms for more type of attacks.

The proposed two-tier attack detection and profiling mechanism is currently trained on three attacks available in AWID-CLS dataset. The attack detection module can be trained for more attack traffic for better efficiency in attack detection for the network. In addition, the attack detection mechanism should be extended to support online learning so that the classifier can learn as it functions on the network.

References

[1] D. Airehrour, J. Gutierrez, and S. K. Ray, "Secure routing for internet of things: A survey," *Journal of Network and Computer Applications,* vol. 66, pp. 198-213, 2016/05/01/ 2016.

[2] M. Centenaro, L. Vangelista, A. Zanella, and M. Zorzi, "Long-range communications in unlicensed bands: the rising stars in the IoT and smart city scenarios," *IEEE Wireless Communications,* vol. 23, no. 5, pp. 60-67, 2016.

[3] R. Ratasuk, B. Vejlgaard, N. Mangalvedhe, and A. Ghosh, "NB-IoT system for M2M communication," in *2016 IEEE wireless communications and networking conference*, 2016, pp. 1-5: IEEE.

[4] D. Evans. (2011). *The Internet of Things; How the Next Evolution of the Internet Is Changing Everything*. Available: https://www.cisco.com/c/dam/en_us/about/ac79/docs/innov/IoT_IBSG_0411FINAL.pdf

[5] I. Lee and K. Lee, "The Internet of Things (IoT): Applications, investments, and challenges for enterprises," *Business Horizons,* vol. 58, no. 4, pp. 431-440, 2015/07/01/ 2015.

[6] S. Agrawal and M. L. Das, "Internet of Things—A paradigm shift of future Internet applications," in *Engineering (NUiCONE), 2011 Nirma University International Conference on*, 2011, pp. 1-7: IEEE.

[7] P. A. Laplante and N. Laplante, "The internet of things in healthcare: Potential applications and challenges," *IT Professional,* vol. 18, no. 3, pp. 2-4, 2016.

[8] L. Atzori, A. Iera, and G. Morabito, "The internet of things: A survey," *Computer networks,* vol. 54, no. 15, pp. 2787-2805, 2010.

[9] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash, "Internet of things: A survey on enabling technologies, protocols, and applications," *IEEE communications surveys & tutorials,* vol. 17, no. 4, pp. 2347-2376, 2015.

[10] A. Čolaković and M. Hadžialić, "Internet of Things (IoT): A review of enabling technologies, challenges, and open research issues," *Computer Networks,* vol. 144, pp. 17-39, 2018/10/24/ 2018.

[11] T. Xu, J. B. Wendt, and M. Potkonjak, "Security of IoT systems: Design challenges and opportunities," in *2014 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2014, pp. 417-423: IEEE.

[12] L. Farhan, S. T. Shukur, A. E. Alissa, M. Alrweg, U. Raza, and R. Kharel, "A survey on the challenges and opportunities of the Internet of Things (IoT)," in *2017 Eleventh International Conference on Sensing Technology (ICST)*, 2017, pp. 1-5: IEEE.

[13] Y. Song, J. Lin, M. Tang, and S. Dong, "An Internet of energy things based on wireless LPWAN," *Engineering,* vol. 3, no. 4, pp. 460-466, 2017.

[14] K. Mekki, E. Bajic, F. Chaxel, and F. Meyer, "A comparative study of LPWAN technologies for large-scale IoT deployment," *ICT express,* vol. 5, no. 1, pp. 1-7, 2019.

[15] M. M. Hossain, M. Fotouhi, and R. Hasan, "Towards an analysis of security issues, challenges, and open problems in the internet of things," in *Services (SERVICES), 2015 IEEE World Congress on*, 2015, pp. 21-28: IEEE.

[16] F. Adelantado, X. Vilajosana, P. Tuset-Peiro, B. Martinez, J. Melia-Segui, and T. Watteyne, "Understanding the limits of LoRaWAN," *IEEE Communications magazine,* vol. 55, no. 9, pp. 34-40, 2017.

[17] A.-A. A. Boulogeorgos, P. D. Diamantoulakis, and G. K. Karagiannidis, "Low power wide area networks (lpwans) for internet of things (iot) applications: Research challenges and future trends," *arXiv preprint arXiv:1611.07449,* 2016.

[18] G. Margelis, R. Piechocki, D. Kaleshi, and P. Thomas, "Low Throughput Networks for the IoT: Lessons learned from industrial implementations," in *2015 IEEE 2nd World Forum on Internet of Things (WF-IoT)*, 2015, pp. 181-186.

[19] E. Aras, G. S. Ramachandran, P. Lawrence, and D. Hughes, "Exploring the security vulnerabilities of lora," in *Cybernetics (CYBCONF), 2017 3rd IEEE International Conference on*, 2017, pp. 1-6: IEEE.

[20] F. Montori, L. Bedogni, M. Di Felice, and L. Bononi, "Machine-to-machine wireless communication technologies for the Internet of Things: Taxonomy, comparison and open issues," *Pervasive and Mobile Computing,* vol. 50, pp. 56-81, 2018/10/01/ 2018.

[21] P. I. Radoglou Grammatikis, P. G. Sarigiannidis, and I. D. Moscholios, "Securing the Internet of Things: Challenges, threats and solutions," *Internet of Things,* vol. 5, pp. 41-70, 2019/03/01/ 2019.

[22] S. Zulian, "Security threat analysis and countermeasures for lorawan join procedure," University of Padova Masters Thesis, 2016.

[23] X. Yang, "LoRaWAN: Vulnerability Analysis and Practical Exploitation," Delft University of Technology Masters thesis., 2017.

[24] X. Yang, "LoRaWAN: Vulnerability Analysis and Practical Exploitation," 2017.

[25] S. Shepard, *RFID: radio frequency identification*. McGraw Hill Professional, 2005.

[26] R. Want, "An introduction to RFID technology," *IEEE pervasive computing,* no. 1, pp. 25-33, 2006.

[27] Z. Alliance, "ZigBee Home Automation, Public Application Profile Specification," *Document number 075367r01ZB. ZigBee Profile: 0x0104, Revision 25, Version 1.0, USA,* 2007.

[28] Z. Alliance, "ZIGBEE SPECIFICATION," 2012, Available: http://www.zigbee.org/wp-content/uploads/2014/11/docs-05-3474-20-0csg-zigbee-specification.pdf.

[29] P. Baronti, P. Pillai, V. W. Chook, S. Chessa, A. Gotta, and Y. F. Hu, "Wireless sensor networks: A survey on the state of the art and the 802.15. 4 and ZigBee standards," *Computer communications,* vol. 30, no. 7, pp. 1655-1695, 2007.

[30] J. Song *et al.*, "WirelessHART: Applying wireless technology in real-time industrial process control," in *2008 IEEE Real-Time and Embedded Technology and Applications Symposium*, 2008, pp. 377-386: IEEE.

[31] S. Petersen and S. Carlsen, "WirelessHART Versus ISA100.11a: The Format War Hits the Factory Floor," *IEEE Industrial Electronics Magazine,* vol. 5, no. 4, pp. 23-34, 2011.

[32] M. Nixon and T. R. Rock, "A Comparison of WirelessHART and ISA100. 11a," *Whitepaper, Emerson Process Management,* pp. 1-36, 2012.

[33] M. B. Yassein, W. Mardini, and A. Khalil, "Smart homes automation using Z-wave protocol," in *2016 International Conference on Engineering & MIS (ICEMIS)*, 2016, pp. 1-6: IEEE.

[34] DASH7 Alliance Wireless Sensor and Actuator Network Protocol version 1.2 [Online]. Available: https://dash7-alliance.org/product/dash7-alliance-protocol-specification-v1-2/

[35] L. Vangelista, A. Zanella, and M. Zorzi, "Long-Range IoT Technologies: The Dawn of LoRa™ " in *Future Access Enablers for Ubiquitous and Intelligent Infrastructures*, Cham, 2015, pp. 51-58: Springer International Publishing.

[36] R. Sanchez-Iborra and M.-D. Cano, "State of the Art in LP-WAN Solutions for Industrial IoT Services," *Sensors,* vol. 16, no. 5, p. 708, 2016.

[37] L. Vangelista, A. Zanella, and M. Zorzi, "Long-Range IoT Technologies: The Dawn of LoRa™ ", Cham, 2015, pp. 51-58: Springer International Publishing.

[38] J. d. C. Silva, J. J. P. C. Rodrigues, A. M. Alberti, P. Solic, and A. L. L. Aquino, "LoRaWAN — A low power WAN protocol for Internet of Things: A review and opportunities," in *2017 2nd International Multidisciplinary Conference on Computer and Energy Science (SpliTech)*, 2017, pp. 1-6.

[39] A. Adhikary, X. Lin, and Y.-P. E. Wang, "Performance evaluation of NB-IoT coverage," in *2016 IEEE 84th Vehicular Technology Conference (VTC-Fall)*, 2016, pp. 1-5: IEEE.

[40] L. A. T. Committee. (2017). *LoRaWAN™ 1.1 Specification*. Available: https://lora-alliance.org/sites/default/files/2018-04/lorawantm_specification_-v1.1.pdf

[41] H. Pohl, "Taxonomie und Modellbildung in der Informationssicherheit," *Datenschutz und Datensicherheit (DuD),* vol. 28, no. 11, pp. 678-685, 2004.

[42] H. Knospe and H. Pohl, "RFID security," *Information Security Technical Report,* vol. 9, no. 4, pp. 39-50, 2004/12/01/ 2004.

[43] T. Phillips, T. Karygiannis, and R. Kuhn, "Security standards for the RFID market," *IEEE Security & Privacy,* vol. 3, no. 6, pp. 85-89, 2005.

[44] B. Yang, "Study on security of wireless sensor network based on ZigBee standard," in *2009 International Conference on Computational Intelligence and Security*, 2009, vol. 2, pp. 426-430: IEEE.

[45] M. S. Costa and J. Amaral, "Analysis of wireless industrial automation standards: ISA-100.11 a and WirelessHART," *InTech Magazine,* 2012.

[46] M. Knight, "Wireless security - How safe is Z-wave?," *Computing & Control Engineering Journal,* vol. 17, no. 6, pp. 18-23, 2006.

[47] I. Unwala, Z. Taqvi, and J. Lu, "IoT Security: ZWave and Thread," in *2018 IEEE Green Technologies Conference (GreenTech)*, 2018, pp. 176-182.

[48] M. Bor, J. Vidler, and U. Roedig, "LoRa for the Internet of Things," presented at the Proceedings of the 2016 International Conference on Embedded Wireless Systems and Networks, Graz, Austria, 2016.

[49] R. Miller, "LoRa Security: Building a secure LoRa solution," *White Paper. MWR Labs,* 2016.

[50] T. Pecorella, L. Brilli, and L. Mucchi, "The role of physical layer security in IoT: A novel perspective," *Information,* vol. 7, no. 3, p. 49, 2016.

[51] L. Wallgren, S. Raza, and T. Voigt, "Routing attacks and countermeasures in the RPL-based internet of things," *International Journal of Distributed Sensor Networks,* vol. 9, no. 8, p. 794326, 2013.

[52] M. Nawir, A. Amir, N. Yaakob, and O. B. Lynn, "Internet of Things (IoT): Taxonomy of security attacks," in *2016 3rd International Conference on Electronic Design (ICED)*, 2016, pp. 321-326: IEEE.

[53] I. Andrea, C. Chrysostomou, and G. Hadjichristofi, "Internet of Things: Security vulnerabilities and challenges," in *2015 IEEE Symposium on Computers and Communication (ISCC)*, 2015, pp. 180-187: IEEE.

[54] D. Airehrour, J. A. Gutierrez, and S. K. Ray, "SecTrust-RPL: A secure trust-aware RPL routing protocol for Internet of Things," *Future Generation Computer Systems,* vol. 93, pp. 860-876, 2019/04/01/ 2019.

[55] S. Vadlamani, B. Eksioglu, H. Medal, and A. Nandi, "Jamming attacks on wireless networks: A taxonomic survey," *International Journal of Production Economics,* vol. 172, pp. 76-94, 2016/02/01/ 2016.

[56] C. Cervantes, D. Poplade, M. Nogueira, and A. Santos, "Detection of sinkhole attacks for supporting secure routing on 6LoWPAN for Internet of Things," in *2015 IFIP/IEEE International Symposium on Integrated Network Management (IM)*, 2015, pp. 606-611: IEEE.

[57] R. Mehta and M. M. Parmar, "Trust based mechanism for Securing IoT Routing Protocol RPL against Wormhole &Grayhole Attacks," in *2018 3rd International Conference for Convergence in Technology (I2CT)*, 2018, pp. 1-6.

[58] O. Sbai and M. Elboukhari, "A simulation analyse of MANET's RREQ Flooding and HELLO Flooding attacks with ns-3," presented at the Proceedings of the 2nd International Conference on Networking, Information Systems & Security, Rabat, Morocco, 2019.

[59] K. Krombholz, H. Hobel, M. Huber, and E. Weippl, "Advanced social engineering attacks," *Journal of Information Security and Applications,* vol. 22, pp. 113-122, 2015/06/01/ 2015.

[60] A. T. Campbell, I. Katzela, K. Miki, and J. Vicente, "Open signaling for ATM, internet and mobile networks (OPENSIG'98)," *ACM SIGCOMM Computer Communication Review,* vol. 29, no. 1, pp. 97-108, 1999.

[61] S. Bhattacharjee, K. L. Calvert, and E. W. Zegura, "An architecture for active networking," in *High Performance Networking VII*: Springer, 1997, pp. 265-279.

[62] J. E. van der Merwe and I. M. Leslie, "Switchlets and dynamic virtual ATM networks," in *Integrated Network Management V*, 1997, pp. 355-368: Springer.

[63] R. Enns, "Rfc4741: Netconf configuration protocol," *Std. Track, http://www. ietf. org/rfc/rfc4741. txt,* 2006.

[64] M. Casado, M. J. Freedman, J. Pettit, J. Luo, N. McKeown, and S. Shenker, "Ethane: Taking control of the enterprise," in *ACM SIGCOMM Computer Communication Review*, 2007, vol. 37, no. 4, pp. 1-12: ACM.

[65] E. Haleplidis *et al.*, "Network programmability with ForCES," *IEEE Communications Surveys & Tutorials,* vol. 17, no. 3, pp. 1423-1440, 2015.

[66] N. McKeown *et al.*, "OpenFlow: enabling innovation in campus networks," *ACM SIGCOMM Computer Communication Review,* vol. 38, no. 2, pp. 69-74, 2008.

[67] S. Chakrabarty and D. W. Engels, "A secure IoT architecture for Smart Cities," in *2016 13th IEEE Annual Consumer Communications & Networking Conference (CCNC)*, 2016, pp. 812-813.

[68] B. A. A. Nunes, M. Mendonca, X. N. Nguyen, K. Obraczka, and T. Turletti, "A Survey of Software-Defined Networking: Past, Present, and Future of Programmable Networks," *IEEE Communications Surveys & Tutorials,* vol. 16, no. 3, pp. 1617-1634, 2014.

[69] M. Dabbagh, B. Hamdaoui, M. Guizani, and A. Rayes, "Software-defined networking security: pros and cons," *IEEE Communications Magazine,* vol. 53, no. 6, pp. 73-79, 2015.

[70] M. Dacier, S. Dietrich, F. Kargl, and H. König, "Network Attack Detection and Defense: Security Challenges and Opportunities of Software-Defined Networking," *Dagstuhl Reports,* vol. 6, no. 9, pp. 1-28, 2016.

[71] M. C. Dacier, H. König, R. Cwalinski, F. Kargl, and S. Dietrich, "Security Challenges and Opportunities of Software-Defined Networking," *IEEE Security & Privacy,* vol. 15, no. 2, pp. 96-100, 2017.

[72] K. Benton, L. J. Camp, and C. Small, "OpenFlow vulnerability assessment," presented at the Proceedings of the second ACM SIGCOMM workshop on Hot topics in software defined networking, Hong Kong, China, 2013.

[73] R. Klöti, V. Kotronis, and P. Smith, "OpenFlow: A security analysis," in *2013 21st IEEE International Conference on Network Protocols (ICNP)*, 2013, pp. 1-6.

[74] I. El Naqa and M. J. Murphy, "What Is Machine Learning?," in *Machine Learning in Radiation Oncology: Theory and Applications*, I. El Naqa, R. Li, and M. J. Murphy, Eds. Cham: Springer International Publishing, 2015, pp. 3-11.

[75] E. Carrizosa and D. Romero Morales, "Supervised classification and mathematical optimization," *Computers & Operations Research,* vol. 40, no. 1, pp. 150-165, 2013/01/01/ 2013.

[76] T. O. Ayodele, "Types of machine learning algorithms," *New advances in machine learning,* vol. 3, pp. 19-48, 2010.

[77] F. Osisanwo, J. Akinsola, O. Awodele, J. Hinmikaiye, O. Olakanmi, and J. Akinjobi, "Supervised machine learning algorithms: classification and comparison," *International Journal of Computer Trends and Technology (IJCTT),* vol. 48, no. 3, pp. 128-138, 2017.

[78] S. R. Safavian and D. Landgrebe, "A survey of decision tree classifier methodology," *IEEE transactions on systems, man, and cybernetics,* vol. 21, no. 3, pp. 660-674, 1991.

[79] F. Pedregosa *et al.*, "Scikit-learn: Machine learning in Python," *the Journal of machine Learning research,* vol. 12, pp. 2825-2830, 2011.

[80] C. C. Aggarwal, "Neural networks and deep learning," *Springer,* vol. 10, pp. 978-3, 2018.

[81] V. Jyothsna, R. Prasad, and K. M. Prasad, "A review of anomaly based intrusion detection systems," *International Journal of Computer Applications,* vol. 28, no. 7, pp. 26-35, 2011.

[82] W. Jung, S. Hong, M. Ha, Y. J. Kim, and D. Kim, "SSL-Based Lightweight Security of IP-Based Wireless Sensor Networks," in *2009 International Conference on Advanced Information Networking and Applications Workshops*, 2009, pp. 1112-1117.

[83] S. Raza, D. Trabalza, and T. Voigt, "6LoWPAN Compressed DTLS for CoAP," in *2012 IEEE 8th International Conference on Distributed Computing in Sensor Systems*, 2012, pp. 287-289.

[84] Y. S. Jang, M. R. Usman, M. A. Usman, and S. Y. Shin, "Swapped Huffman tree coding application for low-power wide-area network (LPWAN)," in *2016 International Conference on Smart Green Technology in Electrical and Information Systems (ICSGTEIS)*, 2016, pp. 53-58.

[85] L. Touati, Y. Challal, and A. Bouabdallah, "C-CP-ABE: Cooperative Ciphertext Policy Attribute-Based Encryption for the Internet of Things," in *2014 International Conference on Advanced Networking Distributed Systems and Applications*, 2014, pp. 64-69.

[86] S. Naoui, M. E. Elhdhili, and L. A. Saidane, "Enhancing the security of the IoT LoraWAN architecture," in *2016 International Conference on Performance Evaluation and Modeling in Wired and Wireless Networks (PEMWN)*, 2016, pp. 1-7.

[87] K. N. Prasetyo, Y. Purwanto, and D. Darlis, "An implementation of data encryption for Internet of Things using blowfish algorithm on FPGA," in *2014 2nd International Conference on Information and Communication Technology (ICoICT)*, 2014, pp. 75-79.

[88] X. Yao, Z. Chen, and Y. Tian, "A lightweight attribute-based encryption scheme for the Internet of Things," *Future Generation Computer Systems,* vol. 49, pp. 104-112, 2015/08/01/ 2015.

[89] N. Oualha and K. T. Nguyen, "Lightweight Attribute-Based Encryption for the Internet of Things," in *2016 25th International Conference on Computer Communication and Networks (ICCCN)*, 2016, pp. 1-6.

[90] J. Choi and Y. Kim, "An improved LEA block encryption algorithm to prevent side-channel attack in the IoT system," in *2016 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA)*, 2016, pp. 1-4.

[91] F. Li, Z. Zheng, and C. Jin, "Secure and efficient data transmission in the Internet of Things," *Telecommunication Systems,* vol. 62, no. 1, pp. 111-122, 2016.

[92] K. L. Tsai, Y. L. Huang, F. Y. Leu, and I. You, "TTP Based High-Efficient Multi-Key Exchange Protocol," *IEEE Access,* vol. 4, pp. 6261-6271, 2016.

[93] A. G. Roselin, P. Nanda, and S. Nepal, "Lightweight Authentication Protocol (LAUP) for 6LoWPAN Wireless Sensor Networks," in *2017 IEEE Trustcom/BigDataSE/ICESS*, 2017, pp. 371-378.

[94] J. Kim and J. Song, "A Dual Key-Based Activation Scheme for Secure LoRaWAN," *Wireless Communications and Mobile Computing,* vol. 2017, 2017.

[95] X. Li, J. Niu, S. Kumari, F. Wu, A. K. Sangaiah, and K.-K. R. Choo, "A three-factor anonymous authentication scheme for wireless sensor networks in internet of things environments," *Journal of Network and Computer Applications,* vol. 103, pp. 194-204, 2018.

[96] P. Punithavathi, S. Geetha, M. Karuppiah, S. K. H. Islam, M. M. Hassan, and K.-K. R. Choo, "A lightweight machine learning-based authentication framework for smart IoT devices," *Information Sciences,* vol. 484, pp. 255-268, 2019/05/01/ 2019.

[97] W. Feng, Y. Qin, S. Zhao, and D. Feng, "AAoT: Lightweight attestation and authentication of low-resource things in IoT and CPS," *Computer Networks,* vol. 134, pp. 167-182, 2018/04/07/ 2018.

[98] I. Suciu, J. C. Pacho, A. Bartoli, and X. Vilajosana, "Authenticated Preambles for Denial of Service Mitigation in LPWANs," in *Ad-hoc, Mobile, and Wireless Networks*(Lecture Notes in Computer Science, 2018, pp. 199-210.

[99] J. Kim and J. Song, "A Secure Device-to-Device Link Establishment Scheme for LoRaWAN," *IEEE Sensors Journal,* vol. 18, no. 5, pp. 2153-2160, 2018.

[100] K. Renuka, S. Kumari, D. Zhao, and L. Li, "Design of a Secure Password-Based Authentication Scheme for M2M Networks in IoT Enabled Cyber-Physical Systems," *IEEE Access,* vol. 7, pp. 51014-51027, 2019.

[101] M. Alshahrani and I. Traore, "Secure mutual authentication and automated access control for IoT smart home using cumulative Keyed-hash chain," *Journal of Information Security and Applications,* vol. 45, pp. 156-175, 2019/04/01/ 2019.

[102] K. Hartke, "Observing resources in the constrained application protocol (CoAP)," 2015.

[103] M. A. Jan, F. Khan, M. Alam, and M. Usman, "A payload-based mutual authentication scheme for Internet of Things," *Future Generation Computer Systems,* vol. 92, pp. 1028-1039, 2019/03/01/ 2019.

[104] Y. Jiang and S. Chen, "A Novel Physical-layer Security Scheme for Internet of Things," *Journal of Physics: Conference Series,* vol. 1176, p. 042090, 2019/03 2019.

[105] M. N. Aman, K. C. Chua, and B. Sikdar, "Mutual Authentication in IoT Systems Using Physical Unclonable Functions," *IEEE Internet of Things Journal,* vol. 4, no. 5, pp. 1327-1340, 2017.

[106] A. Yeole, D. R. Kalbande, and A. Sharma, "Security of 6LoWPAN IoT Networks in Hospitals for Medical Data Exchange," *Procedia Computer Science,* vol. 152, pp. 212-221, 2019/01/01/ 2019.

[107] L. Bu, M. Isakov, and M. A. Kinsy, "A secure and robust scheme for sharing confidential information in IoT systems," *Ad Hoc Networks,* vol. 92, p. 101762, 2019/09/01/ 2019.

[108] G. Al-Aali, B. Boneau, and K. Landers, "Diffie-hellman key exchange," *Proceedings of CSE 331, Data Structures Fall 2000,* vol. 67, 2000.

[109] M. L. Yang, *An authenticated key agreement scheme for sensor networks: a thesis submitted to Auckland University of Technology in fulfilment of the requirements for the degree of Doctor of Philosophy, 2014*. 2014.

[110] R. Blom, "An optimal class of symmetric key generation systems," in *Workshop on the Theory and Application of of Cryptographic Techniques*, 1984, pp. 335-338: Springer.

[111] C. Blundo, A. De Santis, A. Herzberg, S. Kutten, U. Vaccaro, and M. Yung, "Perfectly-secure key distribution for dynamic conferences," in *Annual international cryptology conference*, 1992, pp. 471-486: Springer.

[112] N. Chen, J. B. Yao, and G. J. Wen, "An improved LU matrix key pre-distribution scheme for wireless sensor networks," in *Advanced Computer Theory and Engineering, 2008. ICACTE'08. International Conference on*, 2008, pp. 503-507: IEEE.

[113] W. Du, J. Deng, Y. S. Han, P. K. Varshney, J. Katz, and A. Khalili, "A pairwise key predistribution scheme for wireless sensor networks," *ACM Transactions on Information and System Security (TISSEC),* vol. 8, no. 2, pp. 228-258, 2005.

[114] L. Eschenauer and V. D. Gligor, "A key-management scheme for distributed sensor networks," in *Proceedings of the 9th ACM Conference on Computer and Communications Security*, 2002, pp. 41-47: ACM.

[115] H.-S. Lee, Y.-R. Lee, and J.-H. Lee, "Multiparty key agreement protocol based on symmetric techniques," *Communications of the Korean Mathematical Society,* vol. 18, no. 1, pp. 169-179, 2003.

[116] J. Lee and D. R. Stinson, "Deterministic key predistribution schemes for distributed sensor networks," in *International Workshop on Selected Areas in Cryptography*, 2004, pp. 294-307: Springer.

[117] J. Zhou and M. He, "An improved distributed key management scheme in wireless sensor networks," in *International Workshop on Information Security Applications*, 2008, pp. 305-319: Springer.

[118] H. Y. Chien, R.-C. Chen, and A. Shen, "Efficient key pre-distribution for sensor nodes with strong connectivity and low storage space," in *Advanced Information Networking and Applications, 2008. AINA 2008. 22nd International Conference on*, 2008, pp. 327-333: IEEE.

[119] J. Shen, T. Zhou, F. Wei, X. Sun, and Y. Xiang, "Privacy-preserving and lightweight key agreement protocol for V2G in the social internet of things," *IEEE Internet of things Journal,* vol. 5, no. 4, pp. 2526-2536, 2017.

[120] M. Nikooghadam, R. Jahantigh, and H. Arshad, "A lightweight authentication and key agreement protocol preserving user anonymity," *Multimedia Tools and Applications,* vol. 76, no. 11, pp. 13401-13423, 2017/06/01 2017.

[121] M. Alshahrani, I. Traore, and I. Woungang, "Anonymous mutual IoT interdevice authentication and key agreement scheme based on the ZigBee technique," *Internet of Things,* vol. 7, p. 100061, 2019/09/01/ 2019.

[122] X. Li, M. H. Ibrahim, S. Kumari, A. K. Sangaiah, V. Gupta, and K.-K. R. Choo, "Anonymous mutual authentication and key agreement scheme for wearable sensors in wireless body area networks," *Computer Networks,* vol. 129, pp. 429-443, 2017/12/24/ 2017.

[123] M. Kompara, S. K. H. Islam, and M. Hölbl, "A robust and efficient mutual authentication and key agreement scheme with untraceability for WBANs," *Computer Networks,* vol. 148, pp. 196-213, 2019/01/15/ 2019.

[124] A. Gupta, M. Tripathi, T. J. Shaikh, and A. Sharma, "A lightweight anonymous user authentication and key establishment scheme for wearable devices," *Computer Networks,* vol. 149, pp. 29-42, 2019/02/11/ 2019.

[125] H. Ruotsalainen, J. Zhang, and S. Grebeniuk, "Experimental Investigation on Wireless Key Generation for Low-Power Wide-Area Networks," *IEEE Internet of Things Journal,* vol. 7, no. 3, pp. 1745-1755, 2020.

[126] B. Han, S. Peng, X. Wang, and B. Wang, "Distributed Physical Layer Key Generation for Secure LPWAN Communication," in *2019 IEEE 25th International Conference on Parallel and Distributed Systems (ICPADS)*, 2019, pp. 225-232.

[127] J. Xing, L. Hou, K. Zhang, and K. Zheng, "An Improved Secure Key Management Scheme for LoRa System," in *2019 IEEE 19th International Conference on Communication Technology (ICCT)*, 2019, pp. 296-301: IEEE.

[128] P. Wuille, "Bip32: Hierarchical deterministic wallets," *h ttps://github. com/genjix/bips/blob/master/bip-0032. md,* 2012.

[129] K. B. Rasmussen and S. Capkun, "Implications of radio fingerprinting on the security of sensor networks," in *Security and Privacy in Communications Networks and the Workshops, 2007. SecureComm 2007. Third International Conference on*, 2007, pp. 331-340: IEEE.

[130] Q. Xu, R. Zheng, W. Saad, and Z. Han, "Device fingerprinting in wireless networks: Challenges and opportunities," *IEEE Communications Surveys & Tutorials,* vol. 18, no. 1, pp. 94-104, 2016.

[131] B. Sieka, "Active fingerprinting of 802.11 devices by timing analysis," in *Consumer communications and networking conference, 2006. CCNC 2006. 3rd IEEE*, 2006, vol. 1, pp. 15-19: IEEE.

[132] K. Xing, F. Liu, X. Cheng, and D. H. Du, "Real-time detection of clone attacks in wireless sensor networks," in *Distributed Computing Systems, 2008. ICDCS'08. The 28th International Conference on*, 2008, pp. 3-10: IEEE.

[133] W. T. Zhu, J. Zhou, R. H. Deng, and F. Bao, "Detecting node replication attacks in wireless sensor networks: A survey," *Journal of Network and Computer Applications,* vol. 35, no. 3, pp. 1022-1034, 2012/05/01/ 2012.

[134] S. V. Radhakrishnan, A. S. Uluagac, and R. Beyah, "GTID: A technique for physical device and device type fingerprinting," *IEEE Transactions on Dependable and Secure Computing,* vol. 12, no. 5, pp. 519-532, 2015.

[135] G. Baldini, R. Giuliani, G. Steri, and R. Neisse, "Physical layer authentication of Internet of Things wireless devices through permutation and dispersion entropy," in *2017 Global Internet of Things Summit (GIoTS)*, 2017, pp. 1-6: IEEE.

[136] V. Brik, S. Banerjee, M. Gruteser, and S. Oh, "Wireless device identification with radiometric signatures," in *Proceedings of the 14th ACM international conference on Mobile computing and networking*, 2008, pp. 116-127: ACM.

[137] T. Kohno, A. Broido, and K. C. Claffy, "Remote physical device fingerprinting," *IEEE Transactions on Dependable and Secure Computing,* vol. 2, no. 2, pp. 93-108, 2005.

[138] D. B. Faria and D. R. Cheriton, "Detecting identity-based attacks in wireless networks using signalprints," in *Proceedings of the 5th ACM workshop on Wireless security*, 2006, pp. 43-52: ACM.

[139] R. M. Gerdes, T. E. Daniels, M. Mina, and S. Russell, "Device Identification via Analog Signal Fingerprinting: A Matched Filter Approach," in *NDSS*, 2006.

[140] N. Patwari and S. K. Kasera, "Robust location distinction using temporal link signatures," in *Proceedings of the 13th annual ACM international conference on Mobile computing and networking*, 2007, pp. 111-122: ACM.

[141] S. U. Rehman, K. W. Sowerby, P. H. J. Chong, and S. Alam, "Robustness of radiometric fingerprinting in the presence of an impersonator," in *2017 IEEE 28th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*, 2017, pp. 1-5.

[142] X. Mei, D. Liu, K. Sun, and D. Xu, "On Feasibility of Fingerprinting Wireless Sensor Nodes Using Physical Properties," in *2013 IEEE 27th International Symposium on Parallel and Distributed Processing*, 2013, pp. 1112-1121.

[143] J. Cache, "Fingerprinting 802.11 implementations via statistical analysis of the duration field," *Uninformed. org,* vol. 5, 2006.

[144] C. Maurice, S. Onno, C. Neumann, O. Heen, and A. Francillon, "Improving 802.11 fingerprinting of similar devices by cooperative fingerprinting," in *Security and Cryptography (SECRYPT), 2013 International Conference on*, 2013, pp. 1-8: IEEE.

[145] I. Almomani, B. Al-Kasasbeh, and M. Al-Akhras, "WSN-DS: a dataset for intrusion detection systems in wireless sensor networks," *Journal of Sensors,* vol. 2016, 2016.

[146] N. Koroniotis, N. Moustafa, E. Sitnikova, and B. Turnbull, "Towards the development of realistic botnet dataset in the Internet of Things for network forensic analytics: Bot-IoT dataset," *Future Generation Computer Systems,* vol. 100, pp. 779-796, 2019/11/01/ 2019.

[147] E. Hodo *et al.*, "Threat analysis of IoT networks using artificial neural network intrusion detection system," in *2016 International Symposium on Networks, Computers and Communications (ISNCC)*, 2016, pp. 1-6.

[148] B. Chatterjee, D. Das, S. Maity, and S. Sen, "RF-PUF: Enhancing IoT Security Through Authentication of Wireless Nodes Using In-Situ Machine Learning," *IEEE Internet of Things Journal,* vol. 6, no. 1, pp. 388-398, 2019.

[149] A. A. Diro and N. Chilamkurti, "Distributed attack detection scheme using deep learning approach for Internet of Things," *Future Generation Computer Systems,* vol. 82, pp. 761-768, 2018/05/01/ 2018.

[150] N. Moustafa, G. Creech, E. Sitnikova, and M. Keshk, "Collaborative anomaly detection framework for handling big data of cloud computing," in *2017 Military Communications and Information Systems Conference (MilCIS)*, 2017, pp. 1-6.

[151] N. Moustafa and J. Slay, "UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set)," in *2015 Military Communications and Information Systems Conference (MilCIS)*, 2015, pp. 1-6.

[152] H. Sedjelmaci, S. M. Senouci, and M. Al-Bahri, "A lightweight anomaly detection technique for low-resource IoT devices: A game-theoretic methodology," in *2016 IEEE International Conference on Communications (ICC)*, 2016, pp. 1-6: IEEE.

[153] D. Breitenbacher, I. Homoliak, Y. L. Aung, N. O. Tippenhauer, and Y. Elovici, "HADES-IoT: A Practical Host-Based Anomaly Detection System for IoT Devices," in *Proceedings of the 2019 ACM Asia Conference on Computer and Communications Security*, 2019, pp. 479-484: ACM.

[154] M. Zolanvari, M. A. Teixeira, L. Gupta, K. M. Khan, and R. Jain, "Machine Learning-Based Network Vulnerability Analysis of Industrial Internet of Things," *IEEE Internet of Things Journal,* vol. 6, no. 4, pp. 6822-6834, 2019.

[155] Z. Pan, S. Hariri, and J. Pacheco, "Context aware intrusion detection for building automation systems," *Computers & Security,* vol. 85, pp. 181-201, 2019/08/01/ 2019.

[156] H. H. Pajouh, R. Javidan, R. Khayami, D. Ali, and K.-K. R. Choo, "A two-layer dimension reduction and two-tier classification model for anomaly-based intrusion detection in IoT backbone networks," *IEEE Transactions on Emerging Topics in Computing,* 2016.

[157] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A detailed analysis of the KDD CUP 99 data set," in *2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications*, 2009, pp. 1-6: IEEE.

[158] S. Chakrabarty, D. W. Engels, and S. Thathapudi, "Black SDN for the Internet of Things," in *2015 IEEE 12th International Conference on Mobile Ad Hoc and Sensor Systems*, 2015, pp. 190-198.

[159] J. Wu, K. Ota, M. Dong, and C. Li, "A Hierarchical Security Framework for Defending Against Sophisticated Attacks on Wireless Sensor Networks in Smart Cities," *IEEE Access,* vol. 4, pp. 416-424, 2016.

[160] O. Salman, S. Abdallah, I. H. Elhajj, A. Chehab, and A. Kayssi, "Identity-based authentication scheme for the Internet of Things," in *2016 IEEE Symposium on Computers and Communication (ISCC)*, 2016, pp. 1109-1111.

[161] S. S. Bhunia and M. Gurusamy, "Dynamic attack detection and mitigation in IoT using SDN," in *2017 27th International Telecommunication Networks and Applications Conference (ITNAC)*, 2017, pp. 1-6.

[162] R. Y. Xu, X. Huang, J. Zhang, Y. Lu, G. Wu, and Z. Yan, "Software Defined Intelligent Building," *Int. J. Inf. Sec. Priv.,* vol. 9, no. 3, pp. 84-99, 2015.

[163] N. Xue, X. Huang, and J. Zhang, "S2Net: A Security Framework for Software Defined Intelligent Building Networks," in *2016 IEEE Trustcom/BigDataSE/ISPA*, 2016, pp. 654-661.

[164] M. Miettinen, S. Marchal, I. Hafeez, N. Asokan, A.-R. Sadeghi, and S. Tarkoma, "IoT Sentinel: Automated device-type identification for security enforcement in IoT," in *Distributed Computing Systems (ICDCS), 2017 IEEE 37th International Conference on*, 2017, pp. 2177-2184: IEEE.

[165] C. R. Kothari, *Research methodology: Methods and techniques*. New Age International, 2004.

[166] R. Kumar, *Research methodology: A step-by-step guide for beginners*. Sage, 2018.

[167] J. W. Creswell and J. D. Creswell, *Research design: Qualitative, quantitative, and mixed methods approaches*. Sage publications, 2017.

[168] J. Bloomfield and M. J. Fisher, "Quantitative research design," *Journal of the Australasian Rehabilitation Nurses Association,* vol. 22, no. 2, pp. 27-30, 2019.

[169] S. T. March and G. F. Smith, "Design and natural science research on information technology," *Decision support systems,* vol. 15, no. 4, pp. 251-266, 1995.

[170] J. F. Nunamaker Jr, M. Chen, and T. D. Purdin, "Systems development in information systems research," *Journal of management information systems,* vol. 7, no. 3, pp. 89-106, 1990.

[171] A. R. Hevner, "A three cycle view of design science research," *Scandinavian journal of information systems,* vol. 19, no. 2, p. 4, 2007.

[172] R. H. Von Alan, S. T. March, J. Park, and S. Ram, "Design science in information systems research," *MIS quarterly,* vol. 28, no. 1, pp. 75-105, 2004.

[173] K. Peffers, T. Tuunanen, M. Rothenberger, and S. Chatterjee, "A Design Science Research Methodology for Information Systems Research," *J. Manage. Inf. Syst.,* vol. 24, no. 3, pp. 45-77, 2007.

[174] R. Reubens, "To Craft, By Design, for Sustainability: Towards holistic sustainability design for developing-country enterprises," 2016.

[175] J. Franklin, D. McCoy, P. Tabriz, V. Neagoe, J. V. Randwyk, and D. Sicker, "Passive Data Link Layer 802.11 Wireless Device Driver Fingerprinting," in *USENIX Security Symposium*, 2006, vol. 3, pp. 16-89.

[176] C. Kolias, G. Kambourakis, A. Stavrou, and S. Gritzalis, "Intrusion detection in 802.11 networks: empirical evaluation of threats and a public dataset," *IEEE Communications Surveys & Tutorials,* vol. 18, no. 1, pp. 184-208, 2015.

[177] D. N. Ratnayake, H. B. Kazemian, S. A. Yusuf, and A. B. Abdullah, "An intelligent approach to detect probe request attacks in IEEE 802.11 networks," in *Engineering Applications of Neural Networks*: Springer, 2011, pp. 372-381.

[178] Y. Qin, B. Li, M. Yang, and Z. Yan, "Attack Detection for Wireless Enterprise Network: a Machine Learning Approach," in *2018 IEEE International Conference on Signal Processing, Communications and Computing (ICSPCC)*, 2018, pp. 1-6: IEEE.

[179] M. E. Aminanto, R. Choi, H. C. Tanuwidjaja, P. D. Yoo, and K. Kim, "Deep abstraction and weighted feature selection for Wi-Fi impersonation detection," *IEEE Transactions on Information Forensics and Security,* vol. 13, no. 3, pp. 621-636, 2017.

[180] T. C. Dönmez and E. Nigussie, "Security of lorawan v1. 1 in backward compatibility scenarios," *Procedia computer science,* vol. 134, pp. 51-58, 2018.

[181] T. Issariyakul and E. Hossain, "Introduction to network simulator 2 (NS2)," in *Introduction to network simulator NS2*: Springer, 2009, pp. 1-18.

[182] G. Pongor, "Omnet: Objective modular network testbed," in *Proceedings of the international workshop on modeling, analysis, and simulation on computer and telecommunication systems*, 1993, pp. 323-326: Society for Computer Simulation International.

[183] X. Chang, "Network simulations with OPNET," in *WSC'99. 1999 Winter Simulation Conference Proceedings.'Simulation-A Bridge to the Future'(Cat. No. 99CH37038)*, 1999, vol. 1, pp. 307-314: IEEE.

[184] A. Velinov and A. Mileva, "Running and testing applications for Contiki OS using Cooja simulator," 2016.

[185] J. Ahrenholz, C. Danilov, T. R. Henderson, and J. H. Kim, "CORE: A real-time network emulator," in *MILCOM 2008-2008 IEEE Military Communications Conference*, 2008, pp. 1-7: IEEE.

[186] G. Carneiro, "NS-3: Network simulator 3," in *UTM Lab Meeting April*, 2010, vol. 20, pp. 4-5.

[187] R. R. Fontes, S. Afzal, S. H. Brito, M. A. Santos, and C. E. Rothenberg, "Mininet-WiFi: Emulating software-defined wireless networks," in *2015 11th International Conference on Network and Service Management (CNSM)*, 2015, pp. 384-389: IEEE.

[188] G. Lowe, "Casper: A compiler for the analysis of security protocols," in *Proceedings 10th Computer Security Foundations Workshop*, 1997, pp. 18-30: IEEE.

[189] T. Gibson-Robinson, P. Armstrong, A. Boulgakov, and A. W. Roscoe, "FDR3 — A Modern Refinement Checker for CSP," in *Tools and Algorithms for the Construction and Analysis of Systems*, Berlin, Heidelberg, 2014, pp. 187-201: Springer Berlin Heidelberg.

[190] A. Armando *et al.*, "The AVISPA tool for the automated validation of internet security protocols and applications," in *International conference on computer aided verification*, 2005, pp. 281-285: Springer.

[191] C. J. Cremers, "The Scyther Tool: Verification, falsification, and analysis of security protocols," in *International conference on computer aided verification*, 2008, pp. 414-418: Springer.

[192] G. Holmes, A. Donkin, and I. H. Witten, "Weka: A machine learning workbench," in *Proceedings of ANZIIS'94-Australian New Zealnd Intelligent Information Systems Conference*, 1994, pp. 357-361: IEEE.

[193] E. Frank *et al.*, "Weka-a machine learning workbench for data mining," in *Data mining and knowledge discovery handbook*: Springer, 2009, pp. 1269-1277.

[194] M. R. Berthold *et al.*, "KNIME-the Konstanz information miner: version 2.0 and beyond," *AcM SIGKDD explorations Newsletter,* vol. 11, no. 1, pp. 26-31, 2009.

[195] S. Dwivedi, P. Kasliwal, and S. Soni, "Comprehensive study of data analytics tools (RapidMiner, Weka, R tool, Knime)," in *2016 Symposium on Colossal Data Analysis and Networking (CDAN)*, 2016, pp. 1-8: IEEE.

[196] X. Meng *et al.*, "Mllib: Machine learning in apache spark," *The Journal of Machine Learning Research,* vol. 17, no. 1, pp. 1235-1241, 2016.

[197] H. Yin and K. Gai, "An empirical study on preprocessing high-dimensional class-imbalanced data for classification," in *2015 IEEE 17th International Conference on High Performance Computing and Communications, 2015 IEEE 7th International Symposium on Cyberspace Safety and Security, and 2015 IEEE 12th International Conference on Embedded Software and Systems*, 2015, pp. 1314-1319: IEEE.

[198] G. E. Batista, R. C. Prati, and M. C. Monard, "A study of the behavior of several methods for balancing machine learning training data," *ACM SIGKDD explorations newsletter,* vol. 6, no. 1, pp. 20-29, 2004.

[199] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: synthetic minority over-sampling technique," *Journal of artificial intelligence research,* vol. 16, pp. 321-357, 2002.

[200] H. He, Y. Bai, E. A. Garcia, and S. Li, "ADASYN: Adaptive synthetic sampling approach for imbalanced learning," in *2008 IEEE international joint conference on neural networks (IEEE world congress on computational intelligence)*, 2008, pp. 1322-1328: IEEE.

[201] A. Makhzani, J. Shlens, N. Jaitly, I. Goodfellow, and B. Frey, "Adversarial autoencoders," *arXiv preprint arXiv:1511.05644,* 2015.

[202] H. Kim, H. Kim, H. Moon, and H. Ahn, "A weight-adjusted voting algorithm for ensembles of classifiers," *Journal of the Korean Statistical Society,* vol. 40, pp. 437-449, 2011.

[203] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," in *Proceedings of the 31st international conference on neural information processing systems*, 2017, pp. 4768-4777.

[204] S. M. Lundberg *et al.*, "From local explanations to global understanding with explainable AI for trees," *Nature Machine Intelligence,* vol. 2, no. 1, pp. 56-67, 2020/01/01 2020.

[205] M. Hossin and M. Sulaiman, "A review on evaluation metrics for data classification evaluations," *International Journal of Data Mining & Knowledge Management Process,* vol. 5, no. 2, p. 1, 2015.

[206] B. Lantz, B. Heller, and N. McKeown, "A network in a laptop: rapid prototyping for software-defined networks," in *Proceedings of the 9th ACM SIGCOMM Workshop on Hot Topics in Networks*, 2010, pp. 1-6.

[207] C. J. F. Cremers, *Scyther: Semantics and verification of security protocols*. Eindhoven university of Technology Eindhoven, Netherlands, 2006.

[208] M. C. Davide Magrin. (2019). *An ns-3 module for simulation of LoRaWAN networks.* Available: https://apps.nsnam.org/app/lorawan/

[209] J.-P. Kaps and B. Sunar, "Energy comparison of AES and SHA-1 for ubiquitous computing," in *International Conference on Embedded and Ubiquitous Computing*, 2006, pp. 372-381: Springer.

[210] M. L. Yang, "An authenticated key agreement scheme for sensor networks," Auckland University of Technology, 2014.

[211] J. Han and J. Wang, "An enhanced key management scheme for LoRaWAN," *Cryptography,* vol. 2, no. 4, p. 34, 2018.

Appendix

Source Codes

A1 LoRaWAN Scenario for Validation of Energy Consumption

LoRaWANScenario.cc

```
/*
 *This script simulates a simple network LoRaWAN network with single node to implement energy
*model.
 */

// Header Files from NS3 LoRAWAN module

#include "ns3/end-device-lora-phy.h"
#include "ns3/gateway-lora-phy.h"
#include "ns3/class-a-end-device-lorawan-mac.h"
#include "ns3/gateway-lorawan-mac.h"
#include "ns3/simulator.h"
#include "ns3/log.h"
#include "ns3/constant-position-mobility-model.h"
#include "ns3/lora-helper.h"
#include "ns3/mobility-helper.h"
#include "ns3/node-container.h"
#include "ns3/position-allocator.h"
#include "ns3/periodic-sender-helper.h"
#include "ns3/command-line.h"
#include "ns3/basic-energy-source-helper.h"
#include "ns3/lora-radio-energy-model-helper.h"
#include "ns3/file-helper.h"
#include "ns3/names.h"
#include <algorithm>
#include <ctime>

using namespace ns3;
using namespace lorawan;

NS_LOG_COMPONENT_DEFINE ("LoraEnergyModelExample");

int main (int argc, char *argv[])
{

  // Set up logging
  LogComponentEnable ("LoraEnergyModelExample", LOG_LEVEL_ALL);
  // LogComponentEnable ("LoraRadioEnergyModel", LOG_LEVEL_ALL);
  // LogComponentEnable ("LoraChannel", LOG_LEVEL_INFO);
  // LogComponentEnable ("LoraPhy", LOG_LEVEL_ALL);
  // LogComponentEnable ("EndDeviceLoraPhy", LOG_LEVEL_ALL);
  // LogComponentEnable ("GatewayLoraPhy", LOG_LEVEL_ALL);
  // LogComponentEnable ("LoraInterferenceHelper", LOG_LEVEL_ALL);
  // LogComponentEnable ("LorawanMac", LOG_LEVEL_ALL);
```

```
  // LogComponentEnable ("EndDeviceLorawanMac", LOG_LEVEL_ALL);
  // LogComponentEnable ("ClassAEndDeviceLorawanMac", LOG_LEVEL_ALL);
  // LogComponentEnable ("GatewayLorawanMac", LOG_LEVEL_ALL);
  // LogComponentEnable ("LogicalLoraChannelHelper", LOG_LEVEL_ALL);
  // LogComponentEnable ("LogicalLoraChannel", LOG_LEVEL_ALL);
  // LogComponentEnable ("LoraHelper", LOG_LEVEL_ALL);
  // LogComponentEnable ("LoraPhyHelper", LOG_LEVEL_ALL);
  // LogComponentEnable ("LorawanMacHelper", LOG_LEVEL_ALL);
  // LogComponentEnable ("OneShotSenderHelper", LOG_LEVEL_ALL);
  // LogComponentEnable ("OneShotSender", LOG_LEVEL_ALL);
  // LogComponentEnable ("LorawanMacHeader", LOG_LEVEL_ALL);
  // LogComponentEnable ("LoraFrameHeader", LOG_LEVEL_ALL);
  LogComponentEnableAll (LOG_PREFIX_FUNC);
  LogComponentEnableAll (LOG_PREFIX_NODE);
  LogComponentEnableAll (LOG_PREFIX_TIME);

  /************************
  *  Create the channel  *
  ************************/

  NS_LOG_INFO ("Creating the channel...");

  // Create the lora channel object
  Ptr<LogDistancePropagationLossModel> loss = CreateObject<LogDistancePropagationLossModel>
();
  loss->SetPathLossExponent (3.76);
  loss->SetReference (1, 7.7);

  Ptr<PropagationDelayModel> delay = CreateObject<ConstantSpeedPropagationDelayModel> ();

  Ptr<LoraChannel> channel = CreateObject<LoraChannel> (loss, delay);

  /************************
  *  Create the helpers  *
  ************************/

  NS_LOG_INFO ("Setting up helpers...");

  MobilityHelper mobility;
  Ptr<ListPositionAllocator> allocator = CreateObject<ListPositionAllocator> ();
  allocator->Add (Vector (100,0,0));
  allocator->Add (Vector (0,0,0));
  mobility.SetPositionAllocator (allocator);
  mobility.SetMobilityModel ("ns3::ConstantPositionMobilityModel");

  // Create the LoraPhyHelper
  LoraPhyHelper phyHelper = LoraPhyHelper ();
  phyHelper.SetChannel (channel);

  // Create the LorawanMacHelper
  LorawanMacHelper macHelper = LorawanMacHelper ();

  // Create the LoraHelper
  LoraHelper helper = LoraHelper ();
```

```
/************************
*  Create End Devices  *
************************/

NS_LOG_INFO ("Creating the end device...");

// Create a set of nodes
NodeContainer endDevices;
endDevices.Create (1);

// Assign a mobility model to the node
mobility.Install (endDevices);

// Create the LoraNetDevices of the end devices
phyHelper.SetDeviceType (LoraPhyHelper::ED);
macHelper.SetDeviceType (LorawanMacHelper::ED_A);
NetDeviceContainer endDevicesNetDevices = helper.Install (phyHelper, macHelper, endDevices);

/**********************
 *  Create Gateways  *
 **********************/

NS_LOG_INFO ("Creating the gateway...");
NodeContainer gateways;
gateways.Create (1);

mobility.SetPositionAllocator (allocator);
mobility.Install (gateways);

// Create a netdevice for each gateway
phyHelper.SetDeviceType (LoraPhyHelper::GW);
macHelper.SetDeviceType (LorawanMacHelper::GW);
helper.Install (phyHelper, macHelper, gateways);

macHelper.SetSpreadingFactorsUp (endDevices, gateways, channel);

/*********************************************
 *  Install applications on the end devices  *
 *********************************************/

// OneShotSenderHelper oneShotSenderHelper;
// oneShotSenderHelper.SetSendTime (Seconds (10));

// oneShotSenderHelper.Install (endDevices);

PeriodicSenderHelper periodicSenderHelper;

// Packet interval
periodicSenderHelper.SetPeriod (Hours (1));

periodicSenderHelper.Install (endDevices);

/************************
 * Install Energy Model *
 ************************/
```

```
  BasicEnergySourceHelper basicSourceHelper;
  LoraRadioEnergyModelHelper radioEnergyHelper;

  // configure energy source
// Initial Energy for end node
  basicSourceHelper.Set ("BasicEnergySourceInitialEnergyJ", DoubleValue (1000));
  basicSourceHelper.Set ("BasicEnergySupplyVoltageV", DoubleValue (3.3));

// Current withdrawn for transceiver operations in Ampere
  radioEnergyHelper.Set ("StandbyCurrentA", DoubleValue (0.0014));
  radioEnergyHelper.Set ("TxCurrentA", DoubleValue (0.028));
  radioEnergyHelper.Set ("SleepCurrentA", DoubleValue (0.0000015));
  radioEnergyHelper.Set ("RxCurrentA", DoubleValue (0.0112));

  radioEnergyHelper.SetTxCurrentModel ("ns3::ConstantLoraTxCurrentModel",
                                "TxCurrent", DoubleValue (0.028));

  // install source on EDs' nodes
  EnergySourceContainer sources = basicSourceHelper.Install (endDevices);
  Names::Add ("/Names/EnergySource", sources.Get (0));

  // install device model
  DeviceEnergyModelContainer deviceModels = radioEnergyHelper.Install
     (endDevicesNetDevices, sources);

  /**************
   * Get output *
   **************/
  FileHelper fileHelper;
  fileHelper.ConfigureFile ("battery-level", FileAggregator::SPACE_SEPARATED);
  fileHelper.WriteProbe ("ns3::DoubleProbe", "/Names/EnergySource/RemainingEnergy", "Output");


  /****************
  *  Simulation  *
  ****************/
// Total Simulation time
  Simulator::Stop (Hours (720));
  Simulator::Run ();
  Simulator::Destroy ();
  return 0;
}
```

A2 LoRaWAN Energy Model

LoRaWANEnergyModel.h

```
/*
This Program was provided free under GNU license and is modified as required in this research
*/
#ifndef LORA_RADIO_ENERGY_MODEL_H
#define LORA_RADIO_ENERGY_MODEL_H

#include "ns3/device-energy-model.h"
#include "ns3/traced-value.h"
```

```
#include "end-device-lora-phy.h"
#include "lora-tx-current-model.h"

namespace ns3 {
namespace lorawan {

/**
 * \ingroup energy
 */
class LoraRadioEnergyModelPhyListener : public EndDeviceLoraPhyListener
{
public:
  /**
   * Callback type for updating the transmit current based on the nominal tx power.
   */
  typedef Callback<void, double> UpdateTxCurrentCallback;

  LoraRadioEnergyModelPhyListener ();
  virtual ~LoraRadioEnergyModelPhyListener ();

  /**
   * \brief Sets the change state callback. Used by helper class.
   *
   * \param callback Change state callback.
   */
  void SetChangeStateCallback (DeviceEnergyModel::ChangeStateCallback callback);

  /**
   * \brief Sets the update tx current callback.
   *
   * \param callback Update tx current callback.
   */
  void SetUpdateTxCurrentCallback (UpdateTxCurrentCallback callback);

  /**
   * \brief Switches the LoraRadioEnergyModel to RX state.
   *
   * \param duration the expected duration of the packet reception.
   *
   * Defined in ns3::LoraEndDevicePhyListener
   */
  void NotifyRxStart (void);

  /**
   * \brief Switches the LoraRadioEnergyModel to TX state and switches back to
   * STANDBY after TX duration.
   *
   * \param duration the expected transmission duration.
   * \param txPowerDbm the nominal tx power in dBm
   *
   * Defined in ns3::LoraEndDevicePhyListener
   */
  void NotifyTxStart (double txPowerDbm);

  /**
```

```
   * Defined in ns3::LoraEndDevicePhyListener
   */
  void NotifySleep (void);

  /**
   * Defined in ns3::LoraEndDevicePhyListener
   */
  void NotifyStandby (void);


private:
  /**
   * A helper function that makes scheduling m_changeStateCallback possible.
   */
  void SwitchToStandby (void);

  /**
   * Change state callback used to notify the LoraRadioEnergyModel of a state
   * change.
   */
  DeviceEnergyModel::ChangeStateCallback m_changeStateCallback;

  /**
   * Callback used to update the tx current stored in LoraRadioEnergyModel based on
   * the nominal tx power used to transmit the current frame.
   */
  UpdateTxCurrentCallback m_updateTxCurrentCallback;
};


/**
 * \ingroup energy
 * \brief A WiFi radio energy model.
 *
 * 4 states are defined for the radio: TX, RX, STANDBY, SLEEP. Default state is
 * STANDBY.
 * The different types of transactions that are defined are:
 *  1. Tx: State goes from STANDBY to TX, radio is in TX state for TX_duration,
 *     then state goes from TX to STANDBY.
 *  2. Rx: State goes from STANDBY to RX, radio is in RX state for RX_duration,
 *     then state goes from RX to STANDBY.
 *  3. Go_to_Sleep: State goes from STANDBY to SLEEP.
 *  4. End_of_Sleep: State goes from SLEEP to STANDBY.
 * The class keeps track of what state the radio is currently in.
 *
 * Energy calculation: For each transaction, this model notifies EnergySource
 * object. The EnergySource object will query this model for the total current.
 * Then the EnergySource object uses the total current to calculate energy.
 *
 */
class LoraRadioEnergyModel : public DeviceEnergyModel
{
public:
  /**
   * Callback type for energy depletion handling.
```

```
 */
typedef Callback<void> LoraRadioEnergyDepletionCallback;

/**
 * Callback type for energy recharged handling.
 */
typedef Callback<void> LoraRadioEnergyRechargedCallback;

/**
 * \brief Get the type ID.
 * \return the object TypeId
 */
static TypeId GetTypeId (void);
LoraRadioEnergyModel ();
virtual ~LoraRadioEnergyModel ();

/**
 * \brief Sets pointer to EnergySouce installed on node.
 *
 * \param source Pointer to EnergySource installed on node.
 *
 * Implements DeviceEnergyModel::SetEnergySource.
 */

//declaration of key exchange energy

double KeyExchangeEnergy (double time_duration);


void SetEnergySource (Ptr<EnergySource> source);

/**
 * \returns Total energy consumption of the wifi device.
 *
 * Implements DeviceEnergyModel::GetTotalEnergyConsumption.
 */
double GetTotalEnergyConsumption (void) const;

// Setter & getters for state power consumption.
/**
 * \brief Gets idle current.
 *
 * \returns idle current of the lora device.
 */
double GetStandbyCurrentA (void) const;
/**
 * \brief Sets idle current.
 *
 * \param idleCurrentA the idle current
 */
void SetStandbyCurrentA (double idleCurrentA);
/**
 * \brief Gets transmit current.
 *
 * \returns transmit current of the lora device.
```

```
 */
double GetTxCurrentA (void) const;
/**
 * \brief Sets transmit current.
 *
 * \param txCurrentA the transmit current
 */
void SetTxCurrentA (double txCurrentA);
/**
 * \brief Gets receive current.
 *
 * \returns receive current of the lora device.
 */
double GetRxCurrentA (void) const;
/**
 * \brief Sets receive current.
 *
 * \param rxCurrentA the receive current
 */
void SetRxCurrentA (double rxCurrentA);
/**
 * \brief Gets sleep current.
 *
 * \returns sleep current of the lora device.
 */
double GetSleepCurrentA (void) const;
/**
 * \brief Sets sleep current.
 *
 * \param sleepCurrentA the sleep current
 */
void SetSleepCurrentA (double sleepCurrentA);

/**
 * \returns Current state.
 */
EndDeviceLoraPhy::State GetCurrentState (void) const;

/**
 * \param callback Callback function.
 *
 * Sets callback for energy depletion handling.
 */
void SetEnergyDepletionCallback (LoraRadioEnergyDepletionCallback callback);

/**
 * \param callback Callback function.
 *
 * Sets callback for energy recharged handling.
 */
void SetEnergyRechargedCallback (LoraRadioEnergyRechargedCallback callback);

/**
 * \param model the model used to compute the lora tx current.
 */
```

```
  // NOTICE VERY WELL: Current  Model linear or constant as possible choices
  void SetTxCurrentModel (Ptr<LoraTxCurrentModel> model);

  /**
   * \brief Calls the CalcTxCurrent method of the tx current model to
   *        compute the tx current based on such model
   *
   * \param txPowerDbm the nominal tx power in dBm
   */
  // NOTICE VERY WELL: Current  Model linear or constant as possible choices
  void SetTxCurrentFromModel (double txPowerDbm);

  /**
   * \brief Changes state of the LoraRadioEnergyMode.
   *
   * \param newState New state the lora radio is in.
   *
   * Implements DeviceEnergyModel::ChangeState.
   */
  void ChangeState (int newState);

  /**
   * \brief Handles energy depletion.
   *
   * Implements DeviceEnergyModel::HandleEnergyDepletion
   */
  void HandleEnergyDepletion (void);

  /**
   * \brief Handles energy recharged.
   *
   * Implements DeviceEnergyModel::HandleEnergyChanged
   */
  void HandleEnergyChanged (void);

  /**
   * \brief Handles energy recharged.
   *
   * Implements DeviceEnergyModel::HandleEnergyRecharged
   */
  void HandleEnergyRecharged (void);

  /**
   * \returns Pointer to the PHY listener.
   */
  LoraRadioEnergyModelPhyListener * GetPhyListener (void);


private:
  void DoDispose (void);

  /**
   * \returns Current draw of device, at current state.
   *
   * Implements DeviceEnergyModel::GetCurrentA.
```

```
   */
  double DoGetCurrentA (void) const;

  /**
   * \param state New state the radio device is currently in.
   *
   * Sets current state. This function is private so that only the energy model
   * can change its own state.
   */
  void SetLoraRadioState (const EndDeviceLoraPhy::State state);

  Ptr<EnergySource> m_source; ///< energy source

  // Member variables for current draw in different radio modes.
  double m_txCurrentA; ///< transmit current
  double m_rxCurrentA; ///< receive current
  double m_idleCurrentA; ///< idle current
  double m_sleepCurrentA; ///< sleep current
  // NOTICE VERY WELL: Current  Model linear or constant as possible choices
  Ptr<LoraTxCurrentModel> m_txCurrentModel; ///< current model

  /// This variable keeps track of the total energy consumed by this model.
  TracedValue<double> m_totalEnergyConsumption;

  // State variables.
  EndDeviceLoraPhy::State m_currentState;  ///< current state the radio is in
  Time m_lastUpdateTime;          ///< time stamp of previous energy update

  uint8_t m_nPendingChangeState; ///< pending state change
  bool m_isSupersededChangeState; ///< superseded change state

  /// Energy depletion callback
  LoraRadioEnergyDepletionCallback m_energyDepletionCallback;

  /// Energy recharged callback
  LoraRadioEnergyRechargedCallback m_energyRechargedCallback;

  /// EndDeviceLoraPhy listener
  LoraRadioEnergyModelPhyListener *m_listener;
};

} // namespace ns3

}
#endif /* LORA_RADIO_ENERGY_MODEL_H */
```

LoRaWANEnergyModel.cc

```
/*
This Program was provided free under GNU license and is modified as required in this research
*/
#include "ns3/log.h"
#include "ns3/simulator.h"
#include "ns3/pointer.h"
```

```
#include "ns3/energy-source.h"
#include "lora-radio-energy-model.h"


namespace ns3 {
namespace lorawan {

NS_LOG_COMPONENT_DEFINE ("LoraRadioEnergyModel");

NS_OBJECT_ENSURE_REGISTERED (LoraRadioEnergyModel);

TypeId
LoraRadioEnergyModel::GetTypeId (void)
{
  static TypeId tid = TypeId ("ns3::LoraRadioEnergyModel")
    .SetParent<DeviceEnergyModel> ()
    .SetGroupName ("Energy")
    .AddConstructor<LoraRadioEnergyModel> ()
    .AddAttribute ("StandbyCurrentA",
                   "The default radio Standby current in Ampere.",
                   DoubleValue (0.0014),      // idle mode = 1.4mA
                   MakeDoubleAccessor (&LoraRadioEnergyModel::SetStandbyCurrentA,
                                       &LoraRadioEnergyModel::GetStandbyCurrentA),
                   MakeDoubleChecker<double> ())
    .AddAttribute ("TxCurrentA",
                   "The radio Tx current in Ampere.",
                   DoubleValue (0.028),       // transmit at 0dBm = 28mA
                   MakeDoubleAccessor (&LoraRadioEnergyModel::SetTxCurrentA,
                                       &LoraRadioEnergyModel::GetTxCurrentA),
                   MakeDoubleChecker<double> ())
    .AddAttribute ("RxCurrentA",
                   "The radio Rx current in Ampere.",
                   DoubleValue (0.0112),       // receive mode = 11.2mA
                   MakeDoubleAccessor (&LoraRadioEnergyModel::SetRxCurrentA,
                                       &LoraRadioEnergyModel::GetRxCurrentA),
                   MakeDoubleChecker<double> ())
    .AddAttribute ("SleepCurrentA",
                   "The radio Sleep current in Ampere.",
                   DoubleValue (0.0000015),      // sleep mode = 1.5microA
                   MakeDoubleAccessor (&LoraRadioEnergyModel::SetSleepCurrentA,
                                       &LoraRadioEnergyModel::GetSleepCurrentA),
                   MakeDoubleChecker<double> ())
    .AddAttribute ("TxCurrentModel", "A pointer to the attached tx current model.",
                   PointerValue (),
                   MakePointerAccessor (&LoraRadioEnergyModel::m_txCurrentModel),
                   MakePointerChecker<LoraTxCurrentModel> ())
    .AddTraceSource ("TotalEnergyConsumption",
                     "Total energy consumption of the radio device.",
                     MakeTraceSourceAccessor (&LoraRadioEnergyModel::m_totalEnergyConsumption),
                     "ns3::TracedValueCallback::Double")
  ;
  return tid;
}

LoraRadioEnergyModel::LoraRadioEnergyModel ()
```

```
{
  NS_LOG_FUNCTION (this);
  m_currentState = EndDeviceLoraPhy::SLEEP;      // initially STANDBY
  m_lastUpdateTime = Seconds (0.0);
  m_nPendingChangeState = 0;
  m_isSupersededChangeState = false;
  m_energyDepletionCallback.Nullify ();
  m_source = NULL;
  // set callback for EndDeviceLoraPhy listener
  m_listener = new LoraRadioEnergyModelPhyListener;
  m_listener->SetChangeStateCallback (MakeCallback (&DeviceEnergyModel::ChangeState, this));
  // set callback for updating the tx current
  m_listener->SetUpdateTxCurrentCallback (MakeCallback
(&LoraRadioEnergyModel::SetTxCurrentFromModel, this));
}

LoraRadioEnergyModel::~LoraRadioEnergyModel ()
{
  NS_LOG_FUNCTION (this);
  delete m_listener;
}

void
LoraRadioEnergyModel::SetEnergySource (Ptr<EnergySource> source)
{
  NS_LOG_FUNCTION (this << source);
  NS_ASSERT (source != NULL);
  m_source = source;
}

double
LoraRadioEnergyModel::GetTotalEnergyConsumption (void) const
{
  NS_LOG_FUNCTION (this);
  return m_totalEnergyConsumption;
}

double
LoraRadioEnergyModel::GetStandbyCurrentA (void) const
{
  NS_LOG_FUNCTION (this);
  return m_idleCurrentA;
}

void
LoraRadioEnergyModel::SetStandbyCurrentA (double idleCurrentA)
{
  NS_LOG_FUNCTION (this << idleCurrentA);
  m_idleCurrentA = idleCurrentA;
}

double
LoraRadioEnergyModel::GetTxCurrentA (void) const
{
  NS_LOG_FUNCTION (this);
```

```
  return m_txCurrentA;
}

void
LoraRadioEnergyModel::SetTxCurrentA (double txCurrentA)
{
  NS_LOG_FUNCTION (this << txCurrentA);
  m_txCurrentA = txCurrentA;
}

double
LoraRadioEnergyModel::GetRxCurrentA (void) const
{
  NS_LOG_FUNCTION (this);
  return m_rxCurrentA;
}

void
LoraRadioEnergyModel::SetRxCurrentA (double rxCurrentA)
{
  NS_LOG_FUNCTION (this << rxCurrentA);
  m_rxCurrentA = rxCurrentA;
}

double
LoraRadioEnergyModel::GetSleepCurrentA (void) const
{
  NS_LOG_FUNCTION (this);
  return m_sleepCurrentA;
}

void
LoraRadioEnergyModel::SetSleepCurrentA (double sleepCurrentA)
{
  NS_LOG_FUNCTION (this << sleepCurrentA);
  m_sleepCurrentA = sleepCurrentA;
}

EndDeviceLoraPhy::State
LoraRadioEnergyModel::GetCurrentState (void) const
{
  NS_LOG_FUNCTION (this);
  return m_currentState;
}

void
LoraRadioEnergyModel::SetEnergyDepletionCallback (
  LoraRadioEnergyDepletionCallback callback)
{
  NS_LOG_FUNCTION (this);
  if (callback.IsNull ())
    {
      NS_LOG_DEBUG ("LoraRadioEnergyModel:Setting NULL energy depletion callback!");
    }
  m_energyDepletionCallback = callback;
```

```
}

void
LoraRadioEnergyModel::SetEnergyRechargedCallback (
  LoraRadioEnergyRechargedCallback callback)
{
  NS_LOG_FUNCTION (this);
  if (callback.IsNull ())
    {
      NS_LOG_DEBUG ("LoraRadioEnergyModel:Setting NULL energy recharged callback!");
    }
  m_energyRechargedCallback = callback;
}

void
LoraRadioEnergyModel::SetTxCurrentModel (Ptr<LoraTxCurrentModel> model)
{
  m_txCurrentModel = model;
}

void
LoraRadioEnergyModel::SetTxCurrentFromModel (double txPowerDbm)
{
  if (m_txCurrentModel)
    {
      m_txCurrentA = m_txCurrentModel->CalcTxCurrent (txPowerDbm);
    }
}

void
LoraRadioEnergyModel::ChangeState (int newState)
{
  NS_LOG_FUNCTION (this << newState);

  Time duration = Simulator::Now () - m_lastUpdateTime;
  NS_ASSERT (duration.GetNanoSeconds () >= 0);     // check if duration is valid

  // energy to decrease = current * voltage * time
  double energyToDecrease = 0.0;
  double supplyVoltage = m_source->GetSupplyVoltage ();
  switch (m_currentState)
    {
    case EndDeviceLoraPhy::STANDBY:
      energyToDecrease = duration.GetSeconds () * m_idleCurrentA * supplyVoltage;
      break;
    case EndDeviceLoraPhy::TX:
      energyToDecrease = (duration.GetSeconds () * m_txCurrentA * supplyVoltage) +
(KeyExchangeEnergy (duration.GetSeconds ()));
      break;
    case EndDeviceLoraPhy::RX:
      energyToDecrease = duration.GetSeconds () * m_rxCurrentA * supplyVoltage;
      break;
    case EndDeviceLoraPhy::SLEEP:
      energyToDecrease = duration.GetSeconds () * m_sleepCurrentA * supplyVoltage;
      break;
```

```
    default:
      NS_FATAL_ERROR ("LoraRadioEnergyModel:Undefined radio state: " << m_currentState);
    }

  // update total energy consumption
  m_totalEnergyConsumption += energyToDecrease;

  // update last update time stamp
  m_lastUpdateTime = Simulator::Now ();

  std::cout<<"energy cunsumed at: \t" <<int (Simulator::Now ().GetMinutes ()) << "\t is: \t"<<
m_totalEnergyConsumption<<"\n";

  m_nPendingChangeState++;

  // notify energy source
  m_source->UpdateEnergySource ();

  // in case the energy source is found to be depleted during the last update, a callback might be
  // invoked that might cause a change in the Lora PHY state (e.g., the PHY is put into SLEEP mode).
  // This in turn causes a new call to this member function, with the consequence that the previous
  // instance is resumed after the termination of the new instance. In particular, the state set
  // by the previous instance is erroneously the final state stored in m_currentState. The check below
  // ensures that previous instances do not change m_currentState.

  if (!m_isSupersededChangeState)
    {
      // update current state & last update time stamp
      SetLoraRadioState ((EndDeviceLoraPhy::State) newState);

      // some debug message
      NS_LOG_DEBUG ("LoraRadioEnergyModel:Total energy consumption is " <<
                    m_totalEnergyConsumption << "J");
    }

  m_isSupersededChangeState = (m_nPendingChangeState > 1);

  m_nPendingChangeState--;
}

// Function added to update energy consumption for encryption and key exchange

double
LoraRadioEnergyModel::KeyExchangeEnergy (double time_duration)  //added for key exchange
energy addition
{
  double time_calculation = 0.290; //time it takes for BYKA calculation (in seconds)
  double byka_current = 0.0087;    // current supply for BYKA calculation (in Ampere)
  double byka_voltage = 3.1;       //voltage required for BYKA (in Volt)
  double supplyVoltage = m_source->GetSupplyVoltage ();
  double recv_energy = time_duration * m_rxCurrentA * supplyVoltage;
  double byka_energy = time_calculation * byka_current * byka_voltage;
  double mac_energy = 0.00000000033*408;  //energy/bit * number of bits in packet
  double encryption_energy = 0.00000000022 *408;
```

```
  return recv_energy + byka_energy + mac_energy + encryption_energy;
//this returns total energy from receiving the packet from server and BYKA calculation energy

}


void
LoraRadioEnergyModel::HandleEnergyDepletion (void)
{
  NS_LOG_FUNCTION (this);
  NS_LOG_DEBUG ("LoraRadioEnergyModel:Energy is depleted!");
  // invoke energy depletion callback, if set.
  if (!m_energyDepletionCallback.IsNull ())
    {
      m_energyDepletionCallback ();
    }
}

void
LoraRadioEnergyModel::HandleEnergyChanged (void)
{
  NS_LOG_FUNCTION (this);
  NS_LOG_DEBUG ("LoraRadioEnergyModel:Energy changed!");
}


void
LoraRadioEnergyModel::HandleEnergyRecharged (void)
{
  NS_LOG_FUNCTION (this);
  NS_LOG_DEBUG ("LoraRadioEnergyModel:Energy is recharged!");
  // invoke energy recharged callback, if set.
  if (!m_energyRechargedCallback.IsNull ())
    {
      m_energyRechargedCallback ();
    }
}

LoraRadioEnergyModelPhyListener *
LoraRadioEnergyModel::GetPhyListener (void)
{
  NS_LOG_FUNCTION (this);
  return m_listener;
}

/*
 * Private functions start here.
 */

void
LoraRadioEnergyModel::DoDispose (void)
{
  NS_LOG_FUNCTION (this);
  m_source = NULL;
  m_energyDepletionCallback.Nullify ();
```

```
}

double
LoraRadioEnergyModel::DoGetCurrentA (void) const
{
  NS_LOG_FUNCTION (this);
  switch (m_currentState)
    {
    case EndDeviceLoraPhy::STANDBY:
      return m_idleCurrentA;
    case EndDeviceLoraPhy::TX:
      return m_txCurrentA;
    case EndDeviceLoraPhy::RX:
      return m_rxCurrentA;
    case EndDeviceLoraPhy::SLEEP:
      return m_sleepCurrentA;
    default:
      NS_FATAL_ERROR ("LoraRadioEnergyModel:Undefined radio state:" << m_currentState);
    }
}

void
LoraRadioEnergyModel::SetLoraRadioState (const EndDeviceLoraPhy::State state)
{
  NS_LOG_FUNCTION (this << state);
  m_currentState = state;
  std::string stateName;
  switch (state)
    {
    case EndDeviceLoraPhy::STANDBY:
      stateName = "STANDBY";
      break;
    case EndDeviceLoraPhy::TX:
      stateName = "TX";
      break;
    case EndDeviceLoraPhy::RX:
      stateName = "RX";
      break;
    case EndDeviceLoraPhy::SLEEP:
      stateName = "SLEEP";
      break;
    }
  NS_LOG_DEBUG ("LoraRadioEnergyModel:Switching to state: " << stateName <<
                " at time = " << Simulator::Now ().GetSeconds () << " s");
}

// ---------------------------------------------------------------------- //

LoraRadioEnergyModelPhyListener::LoraRadioEnergyModelPhyListener ()
{
  NS_LOG_FUNCTION (this);
  m_changeStateCallback.Nullify ();
  m_updateTxCurrentCallback.Nullify ();
}
```

```
LoraRadioEnergyModelPhyListener::~LoraRadioEnergyModelPhyListener ()
{
  NS_LOG_FUNCTION (this);
}

void
LoraRadioEnergyModelPhyListener::SetChangeStateCallback
(DeviceEnergyModel::ChangeStateCallback callback)
{
  NS_LOG_FUNCTION (this << &callback);
  NS_ASSERT (!callback.IsNull ());
  m_changeStateCallback = callback;
}

void
LoraRadioEnergyModelPhyListener::SetUpdateTxCurrentCallback (UpdateTxCurrentCallback
callback)
{
  NS_LOG_FUNCTION (this << &callback);
  NS_ASSERT (!callback.IsNull ());
  m_updateTxCurrentCallback = callback;
}

void
LoraRadioEnergyModelPhyListener::NotifyRxStart ()
{
  NS_LOG_FUNCTION (this);
  if (m_changeStateCallback.IsNull ())
    {
      NS_FATAL_ERROR ("LoraRadioEnergyModelPhyListener:Change state callback not set!");
    }
  m_changeStateCallback (EndDeviceLoraPhy::RX);
}

void
LoraRadioEnergyModelPhyListener::NotifyTxStart (double txPowerDbm)
{
  NS_LOG_FUNCTION (this << txPowerDbm);
  if (m_updateTxCurrentCallback.IsNull ())
    {
      NS_FATAL_ERROR ("LoraRadioEnergyModelPhyListener:Update tx current callback not
set!");
    }
  m_updateTxCurrentCallback (txPowerDbm);
  if (m_changeStateCallback.IsNull ())
    {
      NS_FATAL_ERROR ("LoraRadioEnergyModelPhyListener:Change state callback not set!");
    }
  m_changeStateCallback (EndDeviceLoraPhy::TX);
}

void
LoraRadioEnergyModelPhyListener::NotifySleep (void)
{
  NS_LOG_FUNCTION (this);
```

```
  if (m_changeStateCallback.IsNull ())
    {
      NS_FATAL_ERROR ("LoraRadioEnergyModelPhyListener:Change state callback not set!");
    }
  m_changeStateCallback (EndDeviceLoraPhy::SLEEP);
}

void
LoraRadioEnergyModelPhyListener::NotifyStandby (void)
{
  NS_LOG_FUNCTION (this);
  if (m_changeStateCallback.IsNull ())
    {
      NS_FATAL_ERROR ("LoraRadioEnergyModelPhyListener:Change state callback not set!");
    }
  m_changeStateCallback (EndDeviceLoraPhy::STANDBY);
}

/*
 * Private function state here.
 */

void
LoraRadioEnergyModelPhyListener::SwitchToStandby (void)
{
  NS_LOG_FUNCTION (this);
  if (m_changeStateCallback.IsNull ())
    {
      NS_FATAL_ERROR ("LoraRadioEnergyModelPhyListener:Change state callback not set!");
    }
  m_changeStateCallback (EndDeviceLoraPhy::STANDBY);
}

}
} // namespace ns3
```

A3 Session Key Mechanism Syntactic Analysis Simulation

```
usertype Timestamp;
usertype String;
const HelloWorld: String;
protocol LPWANSession(Dev,Srv)

{
role Srv {
fresh Rd:Nonce;
fresh T1: Timestamp;
send_!T1(Srv, Dev, {Rd, T1}k(Dev,Srv));
macro SKey={Rd}k(Dev,Srv);
recv_!2(Dev, Srv, {HelloWorld}k(Dev,Srv));
claim(Srv,Alive); //assures the Aliveness of Srv
claim(Srv,Weakagree); //minimum agreement check between partners according to Srv
claim(Srv,Niagree); //validates the non-injective agreement according to Srv
claim(Srv,Nisynch); //validates the non-injective synchronization according to Srv
claim (Srv,SKR,SKey); //validate the secrecy of AppSKey according to Srv
```

```
}

role Dev {
var T1: Timestamp;
var Rd: Nonce;
recv_!T1(Srv, Dev, {Rd, T1}k(Dev,Srv));
macro SKey={Rd}k(Dev,Srv);
send_!2(Dev, Srv, {HelloWorld}k(Dev,Srv));
claim(Dev,Alive); //assures the Aliveness of Dev
claim(Dev,Weakagree); //minimum agreement check between partners according to Dev
claim(Dev,Niagree); //validates the non-injective agreement according to Dev
claim(Dev,Nisynch); //validates the non-injective synchronization according to Dev
claim (Dev,SKR,SKey); //validate the secrecy of AppSKey according to Dev
}
}
```

A4 BYKA Correctness Validation (Mininet-WiFi) Scenario

MininetToppology.py

```
#!/usr/bin/python

from mininet.node import Controller, OVSKernelSwitch, Host, RemoteController
from mininet.log import setLogLevel, info
from mn_wifi.net import Mininet_wifi
from mn_wifi.node import Station, OVSKernelAP, OVSSwitch
from mn_wifi.cli import CLI
from mn_wifi.link import wmediumd
from mn_wifi.wmediumdConnector import interference
from subprocess import call

class InbandController( RemoteController ):

    def checkListening( self ):
        #"Overridden to do nothing."
        return

def myNetwork():

    net = Mininet_wifi(topo=None,
                       build=False,
                       link=wmediumd,
                       wmediumd_mode=interference,
                       ipBase='10.0.0.0/8')

    info( '*** Adding controller\n' )
    c0 = net.addController(name='c0',
                           controller=InbandController,
                           ip='10.0.0.3',
                           protocol='tcp',
                           port = 6633)

    info( '*** Add switches/APs\n')
    ap1 = net.addAccessPoint('ap1', cls=OVSKernelAP, ssid='ap1-ssid',
```

```
                    channel='1', mode='g', position='474.0,277.0,0')
    s2 = net.addSwitch('s2', cls=OVSSwitch, inband=True)

    info( '*** Add hosts/stations\n')
    sta1 = net.addStation('sta1', ip='10.0.0.1/8',
                          position='271.0,384.0,0')
    sta2 = net.addStation('sta2', ip='10.0.0.2/8',
                          position='620.0,394.0,0')
    h1 = net.addHost('h1', cls=Host, ip='10.0.0.3/8', defaultRoute=None)

    info("*** Configuring Propagation Model\n")
    net.setPropagationModel(model="logDistance", exp=3)

    info("*** Configuring wifi nodes\n")
    net.configureWifiNodes()

    info( '*** Add links\n')
    net.addLink(sta1, ap1)
    net.addLink(ap1, sta2)
    net.addLink(s2, ap1)
    net.addLink(s2, h1)

    #net.plotGraph(max_x=1000, max_y=1000)

    info( '*** Starting network\n')
    net.build()
    info( '*** Starting controllers\n')
    for controller in net.controllers:
        controller.start()

    info( '*** Starting switches/APs\n')
    net.get('ap1').start(net.controllers)
    net.get('s2').start(net.controllers)
    s2.cmd('ifconfig s2 inet 10.0.0.10')

    info( '*** Post configure nodes\n')

    CLI(net)
    net.stop()


if __name__ == '__main__':
    setLogLevel( 'info' )
    myNetwork()
```

A5 Sessions Key Mechanism Implementation (BYKA Extension)

Send.py

```
## This file implements sending of data between nodes using socket

import socket
import time

def sendPublicKey(publicKey):
```

```
    s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    s.bind((socket.gethostname(), 1234))
    s.listen(5)
    while True:
        clientsocket, address = s.accept()
        print(f"Connection from {address}")
        clientsocket.send(bytes(str(publicKey), "utf-8"))
        clientsocket.close()
        if address:
            break
```

Receive.py

```
## This file implements receiving of data between nodes using socket

import socket
import time

def recvPublicKey():
    s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    while True:
        try:
            s.connect((socket.gethostname(), 1234))
            msg = s.recv(1024)
            return int(msg.decode("utf-8"))
        except OSError as ex:
            time.sleep(1)
```

Byka.py

```
## This file implements the BYKA key calculation

import numpy as np
import random


m = 16
n = 6
N = 7
p = 31
q = 65521


def genMasterKey(m, N, p):
    masteKeySet = []
    masterKey = np.empty(shape=[m, m], dtype=int)
    for k in range(0, N):
        for i in range(0, m):
            for j in range(0, m):
                random.seed(i + j)
                masterKey[i][j] = random.randrange(p)
                masterKey[j][i] = masterKey[i][j]
        masteKeySet.append(masterKey)
    return masteKeySet
```

```
# serverMasterKey is stored in server and is never transfered anywhere

serverMasterKey = genMasterKey(m, N, p)


def genPublicKey(ID, m, n, q):
    publicKeySet = []
    publicKeyVector = np.empty(shape=[m, 1], dtype='int64')
    for i in range(0, n):
        seed = ID + (i + 1)
        for j in range(0, m):
            publicKeyVector[j] = pow(seed, j)
        publicKeyVector = np.mod(publicKeyVector, q)
        publicKeySet.append(publicKeyVector)
    return publicKeySet


def genPrivateKey(n, N, p, ID):
    publicK = genPublicKey(ID, m, n, q)
    privateKey = []
    for i in range(0, n):
        for j in range(0, N):
            privateKey.append(np.dot(publicK[i].T, serverMasterKey[j]))
    return np.array(privateKey)


n1PrivateKey = np.array(genPrivateKey(n, N, p, 1))
n2PrivateKey = genPrivateKey(n, N, p, 2)
def genPairwiseKey(privateKey, n, N, ID2):
    publicK = genPublicKey(ID2, m, n, q)
    pairwiseKey = []
    for i in range(0, n * N):
        for j in range(0, n):
            pairwiseKey.append(np.dot(privateKey[i], publicK[j]))
    return sum(np.array(pairwiseKey))


def serverOperations ( m,n,N,p,q, ID1= 3,ID2 = 2):
    pkID1 = genPrivateKey(n, N, p, ID1)
    pkID2 = genPrivateKey(n, N, p, ID2)
    pairwiseID2 = genPairwiseKey(pkID1, n,N, ID2)
    pairwiseID1 = genPairwiseKey(pkID2, n,N, ID1)
    if (pairwiseID2==pairwiseID1):
        return pairwiseID1
    print("not matching")
    return False


serverOperations(m,n,N,p,q)
```

Controller.py

```
## This file implements the tasks performed by SDN controller
```

```
import byka as bk
import senddata as sender
import numpy as np

m = 16
n = 6
N = 7
p = 31
q = 65521

usedID = set({})
E = {1,2}
U = set(np.random.randint(100, size=1))
U = U.symmetric_difference(E) # for mutual exclusion

IDtosend = U.pop()
usedID.add(IDtosend)

sender.sendPublicKey(IDtosend)

print(bk.serverOperations(m,n,N,p,q, IDtosend, 2))
```

EndNode.py

```
## This file implements the tasks performed by end nodes
import numpy as np
import recvdata as receiver

m = 16
n = 6
N = 7
p = 31
q = 65521

publicID = receiver.recvPublicKey()

privateKey = np.load("n2PrivateKey.npy", allow_pickle=True)

def genPublicKey(ID, m, n, q):
    publicKeySet = []
    publicKeyVector = np.empty(shape=[m, 1], dtype='int64')
    for i in range(0, n):
        seed = ID + (i + 1)
        for j in range(0, m):
            publicKeyVector[j] = pow(seed, j)
        publicKeyVector = np.mod(publicKeyVector, q)
        publicKeySet.append(publicKeyVector)
    return publicKeySet

def genPairwiseKey(privateKey, n, N, ID2):
    publicK = genPublicKey(ID2, m, n, q)
    pairwiseKey = []
    for i in range(0, n * N):
        for j in range(0, n):
```

```
            pairwiseKey.append(np.dot(privateKey[i], publicK[j]))
    return sum(np.array(pairwiseKey))

print(genPairwiseKey(privateKey, n, N, publicID))
```