A

Project Report

on

Attack Detection on IoT Network Using Machine Learning Techniques

submitted as partial fulfillment for the award of

BACHELOR OF TECHNOLOGY DEGREE

SESSION 2022-23

in

Computer Science and Engineering

By

Yatharth Sharma (1900290100200)

Vansh Kumar (1900290100185)

Under the supervision of

Prof. Himanshi Chaudhary

KIET Group of Institutions, Ghaziabad

Affiliated to

Dr. A.P.J. Abdul Kalam Technical University, Lucknow

(Formerly UPTU)

May, 2023

DECLARATION

We hereby declare that this submission is our own work and that, to the best of our knowledge and belief, it contains no material previously published or written by another person nor material which to a substantial extent has been accepted for the award of any other degree or diploma of the university or other institute of higher learning, except where due acknowledgment has been made in the text.

Signature [illegible] Vansh Kumar

Name: Yatharth Sharma, Vansh Kumar

Roll No.:1900290100200, 1900290100185

Date: 27/05/2023

CERTIFICATE

This is to certify that Project Report entitled "*Attack detection on IoT Network using Machine Learning Techniques*" which is submitted by Yatharth Sharma and Vansh Kumar in partial fulfillment of the requirement for the award of degree B.Tech. in Department of Computer Science & Engineering of Dr. A.P.J. Abdul Kalam Technical University, Lucknow is a record of the candidates own work carried out by them under my supervision. The matter embodied in this report is original and has not been submitted for the award of any other degree.

[illegible]

Date:27/05/2023 **Supervisor Name:** Himanshi Chaudhary

(Designation) Professor

ACKNOWLEDGEMENT

It gives us a great sense of pleasure to present the report of the B. Tech Project undertaken during B. Tech. Final Year. We owe special debt of gratitude to Prof. Himanshi Chaudhary, Department of Computer Science & Engineering, KIET, Ghaziabad, for her constant support and guidance throughout the course of our work. Her sincerity, thoroughness and perseverance have been a constant source of inspiration for us. It is only her cognizant efforts that our endeavors have seen light of the day.

We also take the opportunity to acknowledge the contribution of Dr. Vineet Sharma, Head of the Department of Computer Science & Engineering, KIET, Ghaziabad, for his full support and assistance during the development of the project. We also do not like to miss the opportunity to acknowledge the contribution of all the faculty members of the department for their kind assistance and cooperation during the development of our project.

We also do not like to miss the opportunity to acknowledge the contribution of all faculty members, especially faculty/industry person/any person, of the department for their kind assistance and cooperation during the development of our project. Last but not the least, we acknowledge our friends for their contribution in the completion of the project.

Date:27/05/2023

Signature: [illegible] Vansh Kumar

Name: Yatharth Sharma, Vansh Kumar

ABSTRACT

Today, the exponential growth of technology and the surge in connectivity demand has led to substantial increase in Internet of Things (IoT) devices. With the enormous increase in IoT devices, secure communication among them has become a major issue due to the increase in cyber-attacks on these connected devices. Adding hardware-based security is challenging due to the complex architecture of IoT devices and limited computation ability. Machine learning (ML) can be utilized to detect attacks on IoT devices due to enormous amount of data produced by them. In this report, we propose some Machine Learning models to detect and prevent Botnet IoT attacks. N-BaIoT dataset is used to train our models. We have analyzed the performance of models using precision, recall and F1-score by carrying out multiclass classification. Experimental result shows that some of the proposed models are efficient in detecting botnet attacks with 99% accuracy. We can also extend our model to use complete dataset and new ML techniques.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

1.	ANN	Artificial Neural Network
2.	GBN	Gaussian Naïve Bayes
3.	IoT	Internet of Things
4.	KNN	K Nearest Neighbor
5.	NIDS	Network Intrusion Detection System
6.	NLP	Natural Language Processing
7.	RNN	Recurrent Neural Network
8.	SVM	Support Vector Machine
9.	TCP	Transmission Control Protocol
10.	UDP	User Datagram Protocol

CHAPTER 1

INTRODUCTION

1.1 Introduction

IoT devices are physical, computing devices that include smart home devices, sensors, wearables, security devices and other machines that collect and exchange data over the Internet. With improvements in IoT, it has now become possible for us to meet digitally and cooperate with others. It has also connected various electrical, electronic and mechanical machines to the internet which gives us the advantage to control these machines from anywhere in the world.

With the advancement in technology, it is crucial for us to protect and our secure our devices from cyber-attack. In the past year, we have observed an increase in the number and complications of attacks. Attackers are using new and sophisticated attacks. Vendors that have been a victim of cyber-attack have increased by 27% and affected IoT devices have increased by 19% from the mid of 2021.

Currently, IoT networks are attacked through various methods and techniques. Jin Yier provided hardware- assisted protection mechanisms that are more effective in countering attacks and cause less performance overhead. However, it is difficult to install hardware-assisted security defenses in the IoT network. Thus, we intend to detect attacks on IoT devices using machine learning techniques.

1.2 Internet of Things (IoT) and Applications

IoT, or the Internet of Things, refers to the interconnected network of physical devices that are embedded with sensors, software, and connectivity capabilities that allow them to collect and exchange data with other devices and systems over the internet. These devices can include everything from smartphones and smart home appliances to industrial equipment and vehicles.

The concept behind IoT is to create a seamless network of connected devices that can improve efficiency, productivity, and convenience by enabling real-time data sharing and analysis. IoT devices can be used in a wide range of industries, including healthcare, transportation, manufacturing, and energy.

IoT (Internet of Things) applications refer to the various ways in which connected devices, sensors, and systems are utilized to collect, exchange, and act upon data in order to enhance efficiency, automate processes, and improve decision-making. Here are some examples of IoT applications across different domains:

1. **Smart Home:** IoT enables homeowners to control and automate various aspects of their homes, such as lighting, temperature, security systems, and appliances, using their smartphones or voice assistants.

2. **Industrial Automation:** IoT is extensively used in industries for tasks such as remote monitoring of equipment, predictive maintenance, supply chain optimization, and real-time inventory management.

3. **Agriculture:** IoT applications in agriculture include soil monitoring, automated irrigation systems, livestock tracking, and crop health monitoring, enabling farmers to optimize resource usage and improve yields.

4. **Healthcare:** IoT devices are employed in healthcare for remote patient monitoring, wearable health trackers, smart pills, and asset tracking in hospitals, improving patient care, and enabling early intervention.

5. **Smart Cities:** IoT technologies are applied in urban environments for traffic management, parking space optimization, waste management, air quality monitoring, and public safety systems, enhancing overall city functionality.

6. **Energy Management:** IoT solutions are used for smart grid systems, energy

consumption monitoring, and demand response programs, enabling efficient energy usage, cost savings, and integration of renewable energy sources.

7. **Retail:** IoT facilitates inventory management, customer behavior tracking, personalized marketing, and smart shelves, enhancing the shopping experience and optimizing operations for retailers.

8. **Transportation and Logistics:** IoT is utilized in fleet management, asset tracking, route optimization, and predictive maintenance of vehicles, improving efficiency, reducing costs, and enhancing safety.

9. **Environmental Monitoring:** IoT devices enable real-time monitoring of environmental factors such as air quality, water quality, weather conditions, and natural resource management, aiding in conservation efforts.

10. **Smart Wearables:** IoT-enabled wearables like smartwatches, fitness trackers, and health monitors collect and analyze data related to physical activity, sleep patterns, heart rate, and more, empowering individuals to make informed lifestyle choices.

1.3 IoT Attacks

The Internet of Things (IoT) refers to the interconnection of everyday devices with the internet, allowing for the collection, analysis, and exchange of data. IoT devices are becoming increasingly prevalent, from smart home appliances and wearables to industrial equipment and critical infrastructure. However, this increased connectivity also creates new security risks, and IoT devices are vulnerable to a range of attacks. In this article, we will discuss some of the most common types of IoT attacks and their potential consequences.

1.3.1 History Events:

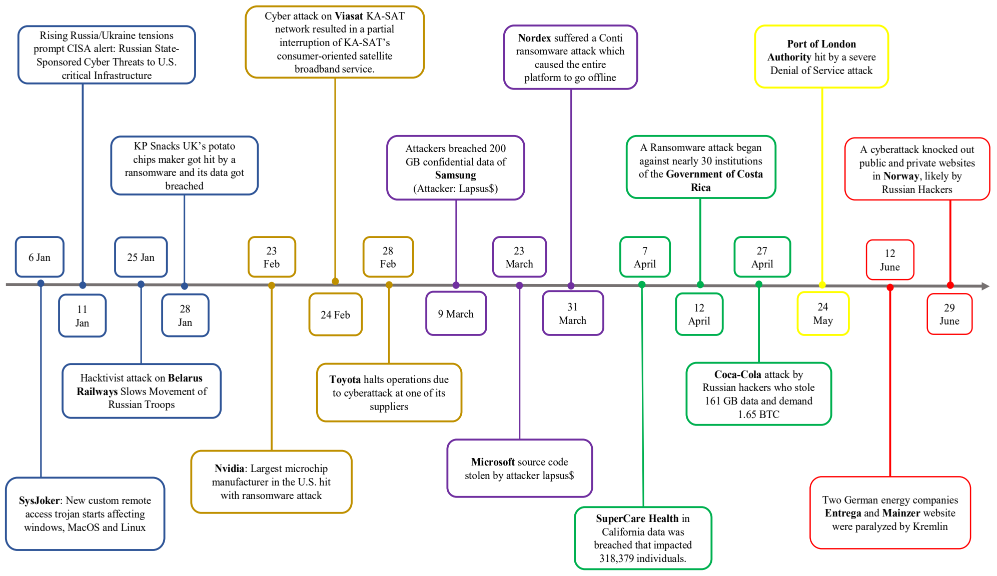
Rising Russia/Ukraine tensions prompt CISA alert: Russian State-Sponsored Cyber Threats to U.S. critical Infrastructure
Cyber attack on Viasat KA-SAT network resulted in a partial interruption of KA-SAT's consumer-oriented satellite broadband service.
Nordex suffered a Conti ransomware attack which caused the entire platform to go offline
Port of London Authority hit by a severe Denial of Service attack
KP Snacks UK's potato chips maker got hit by a ransomware and its data got breached
Attackers breached 200 GB confidential data of Samsung (Attacker: Lapsus$)
A Ransomware attack began against nearly 30 institutions of the Government of Costa Rica
A cyberattack knocked out public and private websites in Norway, likely by Russian Hackers
6 Jan
25 Jan
23 Feb
28 Feb
23 March
7 April
27 April
12 June
11 Jan
28 Jan
24 Feb
9 March
31 March
12 April
24 May
29 June
Hacktivist attack on Belarus Railways Slows Movement of Russian Troops
Toyota halts operations due to cyberattack at one of its suppliers
Coca-Cola attack by Russian hackers who stole 161 GB data and demand 1.65 BTC
Nvidia: Largest microchip manufacturer in the U.S. hit with ransomware attack
Microsoft source code stolen by attacker lapsus$
SysJoker: New custom remote access trojan starts affecting windows, MacOS and Linux
SuperCare Health in California data was breached that impacted 318,379 individuals.
Two German energy companies Entrega and Mainzer website were paralyzed by Kremlin

Figure 1 – IoT attack timeline

- **Mirai Events:**

The Mirai attacks refer to a series of distributed denial-of-service (DDoS) attacks that occurred in 2016. These attacks exploited vulnerabilities in Internet of Things (IoT) devices, such as routers, cameras, and DVRs, to create a botnet that was used to launch massive DDoS attacks. Here are the key events related to the Mirai attacks:

1. August 2016: Mirai Botnet Emerges

- The Mirai botnet is first identified by security researchers. It spreads by scanning the internet for vulnerable IoT devices and then infecting them with malware.

2. September 20, 2016: KrebsOnSecurity Targeted

- Security journalist Brian Krebs' website, KrebsOnSecurity, experiences a massive DDoS attack, reaching 620 Gbps of traffic. This attack is attributed to the Mirai botnet.

3. September 30, 2016: Dyn DNS Attack

- The Mirai botnet launches a powerful DDoS attack on Dyn, a major domain name system (DNS) provider. This attack affects numerous websites and online services, including Twitter, Reddit, Netflix, and Spotify, causing widespread disruption.

4. October 2016: Public Release of Mirai's Source Code

- The source code for Mirai is released publicly, allowing other cybercriminals to create their own variants of the malware and botnet.

5. November 2016: Mirai Botnet Creator Arrested

- Three individuals, Paras Jha, Josiah White, and Dalton Norman, are identified and arrested for their involvement in creating and operating the Mirai botnet.

6. December 2016: Plea Agreement

- Paras Jha pleads guilty to creating and operating the Mirai botnet. He cooperates with law enforcement agencies to assist in further investigations.

7. January 2017: Sentencing of Mirai Botnet Creator

- Paras Jha is sentenced to six months of home confinement, 2,500 hours of community service, and ordered to pay restitution for the damage caused by the Mirai attacks.

8. Subsequent Developments

- In the years following the Mirai attacks, there have been further instances of IoT-based botnets and DDoS attacks. Security measures have been improved, and efforts to secure IoT devices have been undertaken by manufacturers and cybersecurity experts.

It's worth noting that while the original Mirai attacks occurred in 2016, the subsequent use and evolution of IoT-based botnets have continued to be a concern in the cybersecurity landscape.

- <u>**Bashlite Events:**</u>

The Bashlite attacks, also known as the Lizkebab attacks, were a series of malware-driven distributed denial-of-service (DDoS) attacks that targeted IoT devices in 2014. The malware responsible for these attacks exploited a vulnerability in the Bash shell, leading to the name "Bashlite." Here are the key events related to the Bashlite attacks:

1. September 2014: Discovery of Bash Vulnerability

- Security researchers discover a critical vulnerability in the Bash shell, known as "Shellshock" or CVE-2014-6271. This vulnerability allows remote attackers to execute arbitrary commands on systems running vulnerable versions of Bash.

2. September 2014: Emergence of Bashlite Malware

- Cybercriminals begin exploiting the Shellshock vulnerability to infect vulnerable IoT devices, such as routers, cameras, and home automation systems, with a malware strain dubbed "Bashlite" or "Lizkebab."

3. October 2014: Increasing Spread of Bashlite Malware

- The Bashlite malware spreads rapidly across vulnerable IoT devices, creating a botnet. Infected devices are used to launch DDoS attacks, causing disruptions to targeted websites and online services.

4. November 2014: Law Enforcement Actions

- International law enforcement agencies, including the FBI, Europol, and the UK's National Crime Agency (NCA), collaborate to disrupt the infrastructure supporting the Bashlite attacks. They seize command-and-control (C&C) servers and take down several domains associated with the malware.

5. Subsequent Developments

- After the law enforcement actions, the spread of the Bashlite malware significantly declined. However, IoT-based botnets and DDoS attacks continued to evolve and pose significant threats in subsequent years.

It's important to note that the Bashlite attacks were one of the early instances where IoT devices were targeted at a large scale for the purpose of creating botnets. The vulnerabilities exploited in these attacks highlighted the need for improved security measures in IoT devices and raised awareness about the potential risks associated with insecure IoT deployments.

1.3.2 Types of IoT Attacks:

1. **Denial-of-service (DoS) attacks**: A DoS attack is a type of cyberattack that aims to make a device or network unavailable to legitimate users. In a DoS attack on IoT devices, attackers flood the device with traffic or send requests that cause it to crash. This can be particularly problematic for critical infrastructure, such as power plants and transportation systems.

2. **Botnet attacks:** A botnet is a network of devices that have been compromised by a hacker, allowing the hacker to control them remotely. Botnets can be used for a variety of purposes, such as launching DDoS attacks or stealing data. In an IoT botnet attack, the devices compromised are typically smart home appliances or other IoT devices.

3. **Malware attacks:** Malware is a type of software designed to infect and damage computers and other digital devices. In an IoT context, malware can be used to compromise the security of a device, allowing attackers to steal data or use the device for other malicious purposes. Malware can be installed on IoT devices through a variety of means, such as phishing attacks or exploiting vulnerabilities in the device's software.

4. **Man-in-the-middle (MitM) attacks:** A MitM attack is a type of cyberattack in which an attacker intercepts communications between two parties, allowing them to eavesdrop on the conversation or modify the data being transmitted. In an IoT context, a MitM attack can be used to gain access to sensitive information or to take control of a device.

5. **Credential stuffing attacks:** Credential stuffing is a type of cyberattack in which attackers use stolen or leaked login credentials to gain unauthorized access to user accounts. In an IoT context, credential stuffing attacks can be used to gain access to smart home devices, such as security cameras or thermostats, and potentially compromise the security of the user's entire network.
6. **Supply chain attacks:** A supply chain attack is a type of cyberattack in which attackers target a third-party vendor or supplier to gain access to their customers' networks. In an IoT context, supply chain attacks can be particularly dangerous, as many IoT devices are manufactured by third-party vendors and may contain vulnerabilities that can be exploited by attackers.

1.3.3 Potential Consequences

The potential consequences of IoT attacks can vary depending on the nature of the attack and the device or network being targeted. However, some common consequences include:

1. Data theft: Many IoT devices collect and transmit sensitive data, such as personal information, location data, and health data. If an attacker is able to compromise the security of an IoT device, they may be able to steal this data and use it for malicious purposes, such as identity theft.

2. Network disruption: If an attacker is able to launch a successful DoS or botnet attack on an IoT device or network, they can disrupt the normal operation of the device or network, potentially causing significant damage or inconvenience.

3. Device hijacking: In some cases, attackers may be able to take control of an IoT device and use it for their own purposes. For example, an attacker could take control of a smart thermostat and use it to manipulate the temperature in a home, or take control of a security camera and use it to spy on the occupants of a home or business.

4. Physical: IoT works by allowing devices to collect data through sensors and send that data to other devices or systems for analysis and processing. This data can be used to improve decision-making, optimize processes, and enable new applications and services. In certain cases, an IoT attack could result in physical damage to the device or surrounding infrastructure. For example, if an attacker were able to take control of a smart car's systems, they could potentially cause an accident or damage the car's systems.

5. Financial losses: If an IoT device is compromised, it could lead to financial losses for the user or organization. For example, an attacker could steal credit card information or use the compromised device to conduct fraudulent transactions.

However, as more and more devices become connected to the internet, the security risks associated with IoT are becoming increasingly apparent. These risks include the potential for data breaches, privacy violations, and cyber attacks. As such, it is important for users and organizations to take steps to secure their IoT devices and networks and mitigate the risk of IoT-related threats.

1.3.4 Prevention and Mitigation

Preventing and mitigating IoT attacks requires a multi-faceted approach. Some strategies that can help to prevent or mitigate IoT attacks include:

1. Strong passwords and authentication: One of the most basic steps to securing IoT devices is to use strong passwords and authentication methods. Many IoT devices come with default passwords that are easy to guess, so it is important to change these passwords to something strong and unique.

2. Regular software updates: IoT devices, like any other digital devices, are vulnerable to software vulnerabilities. Regularly updating the device's software can help to mitigate these vulnerabilities and reduce the risk of attack.

3. Network segmentation: Segmenting IoT devices onto their own network can help to isolate them from the rest of the network, reducing the risk of attack spreading from one device to another.

4. Encryption: Encrypting the data that is transmitted by IoT devices can help to protect it from interception by attackers. Many IoT devices support encryption, but it is important to ensure that encryption is enabled and properly configured.

5. Vendor vetting: When purchasing IoT devices, it is important to vet the vendor to ensure that they have good security practices in place. This includes ensuring that the vendor regularly updates the device's software, uses strong authentication and encryption methods, and has a good track record of responding to security vulnerabilities.

IoT devices are becoming increasingly prevalent, and with this increased connectivity comes new security risks. There are several types of IoT attacks that can compromise the security of IoT devices and networks, including DoS attacks, botnet attacks, malware attacks, MitM attacks, credential stuffing attacks, and supply chain attacks. Preventing and mitigating IoT attacks requires a multi-faceted approach, including strong passwords and authentication, regular software updates, network segmentation, encryption, and vendor vetting. By taking these steps, users and organizations can reduce the risk of IoT attacks and protect their devices and data.

1.4 IoT Security and Importance

IoT security refers to the protection of Internet of Things (IoT) devices and networks from unauthorized access, cyber attacks, and other security threats. With the increasing prevalence of IoT devices in homes, offices, and industries, the security risks associated with them are also increasing.

IoT devices often have limited computing resources, making it difficult to implement robust security features. They may also be vulnerable to attacks due to outdated software, weak passwords, and unencrypted communication. Additionally, IoT devices are often designed to collect and transmit sensitive data, which if compromised, can result in significant privacy and security risks.

The importance of IoT security lies in the potential harm that can be caused if these devices are compromised. Hackers can use IoT devices to launch attacks on other devices or networks, steal sensitive data, or cause physical harm. For example, a hacker could gain access to a medical device connected to the internet and change the dosage of medication, resulting in harm to the patient.

To mitigate these risks, it is important to implement strong security measures for IoT devices and networks. This can include using strong passwords and encryption, regularly updating software, and segmenting IoT devices onto separate networks. Additionally, IoT manufacturers and vendors need to prioritize security in the design and development of their products.

Overall, the security of IoT devices and networks is essential to ensure the privacy, safety, and security of individuals and organizations using these devices.

1.5 Project Description

We are going to detect attacks on IoT devices using some machine learning algorithms. In our project, we have selected 2 devices, namely Provision PT 737E Security Camera and Ecobee Thermostat.
For Botnet attack detection, we have used widely known and used Machine learning algorithms. We have employed six Machine Learning algorithms (Sequential Model, Decision Tree classification, Support vector machine (SVM), K Nearest Neighbor (KNN), Gaussian Naïve Bayes (GNB) and Random Forest).
N-BaIoT Dataset contains 10 different protocols that are utilized for botnet attacks. Thus, we have chosen multiclass classification machine learning algorithms. They can classify each of the protocols used for Mirai and Bashlite attacks.

We have unbiasedly divided the dataset into 75% training and 25% testing data using sklearn library. The selected ML models are trained using the training set and tested against the testing set.

Type of attack	Provision PT 737E Security Camera	Ecobee Thermostat
Benign	31077	13113
Gafgyt Combo	30690	13253
Gafgyt Junk	21629	15156
Gafgyt Scan	20508	13747
Gafgyt TCP	31353	14253
Gafgyt UDP	31203	15719
Mirai Ack	30277	16993
Mirai Scan	29034	17277
Mirai Syn	32873	17521
Mirai UDP	31250	15148
Mirai Udpplain	28340	13105

Table 1 - Number of Instances Used

“Table 1” represent the number of instances of different attack type we have used in our study.
The table above shows the type of attack and the number of instances of those attacks on different types of devices, including a Provision PT 737E security camera, and an Ecobee thermostat.

The different types of attacks include:

- **Benign**: In the context of cybersecurity, a benign attack is used to refer to a controlled or simulated attack that is carried out for the purpose of testing or evaluating a system's security defenses.

  Benign attacks are used in various cybersecurity-related activities such as penetration testing, vulnerability assessment, and security auditing. The objective of these activities is to identify weaknesses or vulnerabilities in a system's security defenses before they can be exploited by a malicious attacker.

A benign attack can take various forms depending on the objective of the activity. For example, a penetration test might involve an attempt to exploit a known vulnerability in a system to gain unauthorized access to its resources. A vulnerability assessment might involve scanning a system for known vulnerabilities and misconfigurations. A security audit might involve reviewing the system's security policies and procedures to identify areas that could be improved.

Benign attacks are typically carried out by trained security professionals or specialized software tools. They are conducted in a controlled environment and with the permission of the system owner or operator to ensure that no harm is done to the system or its data.

The results of a benign attack are used to inform decisions related to improving the security of a system. They may be used to prioritize security investments, identify areas where additional security controls are needed, or improve the effectiveness of existing security controls.

- **Gafgyt Combo**: "Gafgyt" or "Bashlite" is a type of malware that is used to create a network of infected devices, or a botnet, which can then be used to launch distributed denial-of-service (DDoS) attacks. A "Gafgyt combo attack" is a specific type of DDoS attack that is carried out using a combination of different techniques, often referred to as a "combo attack" or "blended attack".

  In a Gafgyt combo attack, the attacker uses multiple attack vectors simultaneously to overwhelm the target's network or servers. These attack vectors can include techniques such as UDP flood, SYN flood, HTTP flood, and DNS amplification. Each of these techniques floods the target with traffic, making it difficult or impossible for legitimate users to access the network or servers.

  The Gafgyt malware is typically spread through vulnerable Internet of Things (IoT) devices such as routers, cameras, and other smart devices that have weak or default login credentials. Once the malware infects a device, it is used to scan for other vulnerable devices and infect them as well, creating a botnet that can be controlled by the attacker.

Gafgyt combo attacks are difficult to defend against because they use multiple attack vectors, making it harder to block all of them at once. Additionally, the attacker can change the attack vectors or increase the intensity of the attack as needed, making it a persistent threat.

To defend against Gafgyt combo attacks, it is important to ensure that all IoT devices are secured with strong login credentials, and that software and firmware updates are regularly applied to address any vulnerabilities. Network and server defenses such as firewalls, intrusion detection systems, and content delivery networks can also help to detect and mitigate DDoS attacks.

- **Gafgyt Junk**: A Gafgyt junk attack, also known as a "Junk Flood" attack, is a type of distributed denial-of-service (DDoS) attack that is carried out using the Gafgyt malware. In this attack, the attacker floods the target network or servers with a large volume of junk traffic in an attempt to overwhelm the system and make it unavailable to legitimate users.

  The junk traffic used in a Gafgyt junk attack is typically generated using a tool called "hping", which sends a large volume of random or meaningless data packets to the target. This flood of traffic can quickly consume the available bandwidth of the target's network, making it difficult or impossible for legitimate traffic to reach its destination.

  Gafgyt junk attacks are typically launched using a botnet of infected Internet of Things (IoT) devices, such as routers, cameras, and other smart devices, that have weak or default login credentials. Once the malware infects a device, it is used to scan for other vulnerable devices and infect them as well, creating a large botnet that can be controlled by the attacker.

To defend against Gafgyt junk attacks, it is important to ensure that all IoT devices are secured with strong login credentials, and that software and firmware updates are regularly applied to address any vulnerabilities. Network and server defenses such as firewalls, intrusion detection systems, and content delivery networks can also help to detect and mitigate DDoS attacks.

- **Gafgyt Scan**: A Gafgyt scan attack, also known as an "IoT device scanning attack", is a type of attack that is used to identify and compromise vulnerable Internet of Things (IoT) devices. The attack is carried out using the Gafgyt malware, which is used to create a botnet of infected devices that can be controlled by the attacker.

  In a Gafgyt scan attack, the malware is used to scan the internet for vulnerable IoT devices, such as routers, cameras, and other smart devices, that have weak or default login credentials. Once a vulnerable device is identified, the malware attempts to login to the device using a list of known or default login credentials. If successful, the malware can infect the device and add it to the botnet.

  Once the botnet is established, the attacker can use it to launch a variety of attacks, including distributed denial-of-service (DDoS) attacks, spamming, and credential theft.

  To defend against Gafgyt scan attacks, it is important to ensure that all IoT devices are secured with strong login credentials, and that software and firmware updates are regularly applied to address any vulnerabilities. Network and server defenses such as firewalls, intrusion detection systems, and content delivery networks can also help to detect and mitigate attacks.

- **Gafgyt TCP**: A Gafgyt TCP attack is a type of Distributed Denial of Service (DDoS) attack that is carried out using the Gafgyt malware. The attack floods the target network or server with a large number of spoofed Transmission Control Protocol (TCP) packets, which can overwhelm the target system and cause it to become unavailable to legitimate users.

In a Gafgyt TCP attack, the malware infects a botnet of Internet of Things (IoT) devices, such as routers, cameras, and other smart devices, that have weak or default login credentials. The attacker can then use the botnet to launch the attack.

The attack works by sending a large number of spoofed TCP packets to the target system, with the aim of consuming all available network resources and making it difficult or impossible for legitimate traffic to reach the target. The attack can also be used to exploit vulnerabilities in the target system, such as buffer overflow or other software vulnerabilities.

To defend against Gafgyt TCP attacks, it is important to ensure that all IoT devices are secured with strong login credentials, and that software and firmware updates are regularly applied to address any vulnerabilities. Network and server defenses such as firewalls, intrusion detection systems, and content delivery networks can also help to detect and mitigate DDoS attacks.

- **Gafgyt UDP**: A Gafgyt UDP attack is a type of Distributed Denial of Service (DDoS) attack that is carried out using the Gafgyt malware. The attack floods the target network or server with a large number of spoofed User Datagram Protocol (UDP) packets, which can overwhelm the target system and cause it to become unavailable to legitimate users.

  In a Gafgyt UDP attack, the malware infects a botnet of Internet of Things (IoT) devices, such as routers, cameras, and other smart devices, that have weak or default login credentials. The attacker can then use the botnet to launch the attack.

  The attack works by sending a large number of spoofed UDP packets to the target system, with the aim of consuming all available network resources and making it difficult or impossible for legitimate traffic to reach the target. The attack can also be used to exploit vulnerabilities in the target system, such as buffer overflow or other software vulnerabilities.

To defend against Gafgyt UDP attacks, it is important to ensure that all IoT devices are secured with strong login credentials, and that software and firmware updates are regularly applied to address any vulnerabilities. Network and server defenses such as firewalls, intrusion detection systems, and content delivery networks can also help to detect and mitigate DDoS attacks.

- **Mirai Ack**: The Mirai ACK attack is a type of Distributed Denial of Service (DDoS) attack that was first observed in 2016 and is carried out using the Mirai malware. The attack floods the target network or server with a large number of spoofed TCP Acknowledgment (ACK) packets, which can overwhelm the target system and cause it to become unavailable to legitimate users.

  In a Mirai ACK attack, the malware infects a botnet of Internet of Things (IoT) devices, such as routers, cameras, and other smart devices, that have weak or default login credentials. The attacker can then use the botnet to launch the attack.

  The attack works by sending a large number of spoofed TCP ACK packets to the target system, with the aim of consuming all available network resources and making it difficult or impossible for legitimate traffic to reach the target. Unlike traditional DDoS attacks, which flood the target with traffic, the Mirai ACK attack exploits the connection state between the target and its clients by sending numerous ACK packets to the target without completing the full TCP three-way handshake. This causes the target to allocate resources to track the incomplete connections, which can quickly exhaust its resources and cause it to become unavailable.

  To defend against Mirai ACK attacks, it is important to ensure that all IoT devices are secured with strong login credentials, and that software and firmware updates are regularly applied to address any vulnerabilities. Network and server defenses such as firewalls, intrusion detection systems, and content delivery networks can also help to detect and mitigate DDoS attacks.

Additionally, it is possible to filter out ACK packets that do not belong to a complete TCP connection to minimize the impact of the attack.

- **Mirai Scan**: The Mirai scan attack is a type of attack that is carried out using the Mirai malware. It involves scanning for and infecting vulnerable Internet of Things (IoT) devices, such as routers, cameras, and other smart devices, with the Mirai malware in order to create a botnet that can be used to carry out Distributed Denial of Service (DDoS) attacks.

  The Mirai malware spreads by scanning the Internet for devices that have weak or default login credentials. Once it finds a vulnerable device, it attempts to login using a list of common username and password combinations. If successful, it downloads and installs the Mirai malware onto the device, which then becomes part of the botnet.

  The Mirai scan attack is particularly effective because it targets a large number of vulnerable IoT devices that are connected to the Internet, such as home routers, cameras, and other smart devices. These devices are often poorly secured and have default login credentials that are easy to guess, making them vulnerable to attack.
  Once a botnet is created, the attacker can use it to launch large-scale DDoS attacks against a target system or network. These attacks can overwhelm the target and cause it to become unavailable to legitimate users.

  To defend against Mirai scan attacks, it is important to ensure that all IoT devices are secured with strong login credentials, and that software and firmware updates are regularly applied to address any vulnerabilities. Network and server defenses such as firewalls, intrusion detection systems, and content delivery networks can also help to detect and mitigate DDoS attacks. Additionally, it is important to monitor network traffic for signs of Mirai malware infections and to block any malicious traffic that is detected.

- **Mirai Syn**: The Mirai SYN attack is a type of Distributed Denial of Service

(DDoS) attack that is carried out using the Mirai malware. It is similar to other SYN flood attacks, in which a large number of TCP SYN requests are sent to a target system or network in order to overwhelm it and prevent legitimate users from accessing it.

In a Mirai SYN attack, the malware infects and uses vulnerable Internet of Things (IoT) devices, such as routers, cameras, and other smart devices, to flood the target with SYN requests. These requests are sent from a large number of different devices, making it difficult for the target to distinguish between legitimate and malicious traffic.

The attack works by exploiting the three-way handshake protocol used in TCP/IP communications. When a client wants to establish a connection with a server, it sends a SYN request. The server then responds with a SYN-ACK packet, and the client sends an ACK packet in response. In a SYN flood attack, the attacker sends a large number of SYN requests without sending the ACK packet, causing the server to hold open a connection for each request. This can consume all available resources on the server, preventing legitimate traffic from being processed.

The Mirai SYN attack is particularly effective because it uses a large number of infected IoT devices, making it difficult for network defenders to distinguish between legitimate and malicious traffic. Additionally, because these devices are often poorly secured and have default login credentials, they can be easily infected and used in the attack.

To defend against a Mirai SYN attack, it is important to ensure that all IoT devices are secured with strong login credentials, and that software and firmware updates are regularly applied to address any vulnerabilities. Network and server defenses such as firewalls, intrusion detection systems, and content delivery networks can also help to detect and mitigate DDoS attacks. Additionally, it is important to monitor network traffic for signs of Mirai malware infections and to block any malicious traffic that is detected.

- **Mirai UDP**: The Mirai UDP attack is a type of Distributed Denial of Service (DDoS) attack that is carried out using the Mirai malware. In this attack, the malware infects and uses vulnerable Internet of Things (IoT) devices, such as routers, cameras, and other smart devices, to flood a target with large amounts of User Datagram Protocol (UDP) traffic.

  UDP is a protocol used for sending packets of data over the internet. Unlike TCP, which establishes a connection between two devices before transmitting data, UDP is connectionless and simply sends packets of data from one device to another. This makes it faster and more efficient, but also makes it easier to abuse in DDoS attacks.

  In a Mirai UDP attack, the malware infects IoT devices and instructs them to send a large number of UDP packets to a target. These packets can be sent to any port on the target, but are often directed at ports commonly used by network protocols such as Domain Name System (DNS), Simple Network Management Protocol (SNMP), or Network Time Protocol (NTP). By flooding the target with a large amount of UDP traffic, the attackers can overwhelm the target's resources and prevent legitimate traffic from being processed.

  The Mirai UDP attack is particularly effective because it uses a large number of infected IoT devices, making it difficult for network defenders to distinguish between legitimate and malicious traffic. Additionally, because these devices are often poorly secured and have default login credentials, they can be easily infected and used in the attack.

  To defend against a Mirai UDP attack, it is important to ensure that all IoT devices are secured with strong login credentials, and that software and firmware updates are regularly applied to address any vulnerabilities. Network and server defenses such as firewalls, intrusion detection systems, and content delivery networks can also help to detect and mitigate DDoS attacks. Additionally, it is important to monitor network traffic for signs of Mirai malware infections and to block any malicious traffic that is detected.

- **Mirai Udp plain**: The Mirai malware is a type of malware that targets Internet of Things (IoT) devices, such as routers, cameras, and other smart devices. It infects these devices, turning them into a botnet that can be used for various types of attacks, including Distributed Denial of Service (DDoS) attacks.

  One type of DDoS attack that can be carried out using the Mirai malware is the Mirai UDP plain payload attack. In this attack, the malware infects vulnerable IoT devices and instructs them to send a large number of User Datagram Protocol (UDP) packets to a target. The payload of these packets is a simple sequence of characters, often just the letter "A" repeated many times.

  The goal of the Mirai UDP plain payload attack is to flood the target with a large amount of UDP traffic, overwhelming its resources and preventing legitimate traffic from being processed. Because the payload is so simple, it requires very little processing power or memory on the part of the infected IoT devices, allowing a large number of them to participate in the attack simultaneously.

  The Mirai UDP plain payload attack is particularly effective against targets that do not have proper DDoS protection in place, as it can saturate their network capacity and prevent legitimate traffic from reaching its destination. The attack is also difficult to detect and mitigate, as the payload is very simple and does not contain any specific patterns or signatures that can be used to identify the attack traffic.

  To defend against a Mirai UDP plain payload attack, it is important to ensure that all IoT devices are secured with strong login credentials and that software and firmware updates are regularly applied to address any vulnerabilities. Network and server defenses such as firewalls, intrusion detection systems, and content delivery networks can also help to detect and mitigate DDoS attacks. Additionally, monitoring network traffic for signs of Mirai malware infections and blocking any malicious traffic that is detected can help to prevent the spread of the malware and mitigate the risk of future attacks.

The table shows the number of instances of each type of attack on the different types of devices. For example, there were 30690 instances of Gafgyt Combo attacks on the Provision PT 737E, and 13105 instances of Mirai Udpplain attacks on the thermostat.
We have designed these models using Sci-kit learn and Keras.

Model	Description
Sequential Model	Number of Layers: 5 Kernel Initializer: NormalActivation: SoftMax Early Stopping: Difference inval_loss is 10^{-3}
Decision Tree	Criterion: Gini ImpuritySplitter: Best
K-nearest Neighbor	Number of Neighbour:7 Weight: Uniform
Support Vector Machine	Kernel: Linear C (Regularization Parameter): 30
Gaussian NaïveBayes	Portion of the largest variance of allfeatures: 10^{-9}
Random Forest	Number of trees:50 Criterion: Gini Impurity

Table 2 - Description of Models
“Table 2” characterize the design of our models. All the parameters used while training the model are listed in it.

This table describes various machine learning models and their configurations. Here is a breakdown of each model:

1. **Sequential Model**: This is a neural network model with five layers. The kernel initializer is set to normal, and the activation function is SoftMax. The model uses early stopping based on the difference in validation loss, which must be greater than or equal to 10-3 to stop the training process.

2. **Decision Tree**: This is a classification model that uses the Gini Impurity criterion to split the data at each node. The splitter is set to Best, which means the algorithm will try different splitting strategies to find the best one.

3. **K-nearest Neighbor**: This is a classification model that uses the k-nearest neighbors to predict the class of a given data point. The number of neighbors is set to 7, and the weight function is set to Uniform, which means all neighbors are given equal weight.

4. **Support Vector Machine**: This is a classification model that uses a linear kernel to separate the data into different classes. The regularization parameter C is set to 30, which controls the tradeoff between achieving a low training error and having a simpler model.

5. **Gaussian Naive Bayes**: This is a classification model that assumes that the features are independent and follow a Gaussian distribution. The portion of the largest variance of all features is set to 10-9.

6. **Random Forest**: This is an ensemble learning model that consists of 50 decision trees. The criterion for splitting the data is set to Gini Impurity.

To measure the performance of our model we use the following metrics:

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

$$\text{Precision} = \frac{TP}{FP+TP}$$

$$\text{Recall} = \frac{TP}{FN+TP}$$

$$\text{F1-score} = 2 \times \frac{Precision \times Recall}{Precision+Rec}$$

where:

- **True positives (TP)** are situations that the model accurately recognised as positive.

- **False positives (FP)** are situations when the model wrongly identified them as positive when they were actually negative.

- **True negatives (TN)** are situations that the model accurately recognised as negative.

- **False negatives (FN)** are occasions when the model wrongly categorised them as negative when they were actually positive.

- **Accuracy** is defined as the ratio of properly identified instances among all cases.

- **Precision** is the fraction of real positive instances among all cases projected to be positive.

- **Recall** is the percentage of true positive instances accurately recognised by the model out of all real positive cases.

- The **F1-score** is a statistic that combines precision and recall to create a single score that describes a binary classification model's performance.

CHAPTER 2

LITERATURE REVIEW

2.1 A Framework for Understanding Botnets

- A bot is a hijacked computer that can execute commands given by its master. Bots are linked together to form a botnet using a topology determined by the master.
- Botnet lifecycle is classified into formation phase, Command and control phase, attack and post-attack phase.
- The Internet Relay Chat (IRC) protocol is used by botnets to make C&C conversations easier. Based on their communication patterns, this offers a way to identify malevolent botnets.
- Determining the characteristics of botnets enables us to evaluate botnets or botnet architectures. Robustness, resilience, sustainability, exposedness, bandwidth consumption, botnet size, botnet master goals, and botnet firepower are just a few of the characteristics of a botnet that have been defined.

2.2 DDoS in the IoT: Mirai and Other Botnets

- The primary objective of the Mirai botnet is to spread the infection to improperly configured devices and carry out attacks on a target server upon receiving commands from its controller, also known as the botmaster.
- In its initial phase, Mirai conducts scans on random public IP addresses using TCP ports 23 or 2323.
- Mirai has nearly 493,000 variants, representing its diverse range of versions and iterations.
- There are five main reasons IoT devices are particularly advantageous for creating botnets Constant and unobtrusive operation, Feeble protection, Poor maintenance, Considerable attack traffic, Noninteractive or minimally

interactive user interfaces.

2.3 <u>Network-Based Detection of IoT Botnet Attacks Using Deep Autoencoders</u>

- They were pioneers in utilizing autoencoders to detect botnet attacks in IoT network traffic, providing a comprehensive approach for anomaly detection.
- MIRAI and BASHLITE, two prevalent IoT-based botnets, were thoroughly examined by the researchers.
- Due to the uncertainty of IoT manufacturers implementing specific host-based anomaly detectors, network-based approaches were considered.
- Each autoencoder consisted of an input layer with dimensions matching the dataset's features (115), followed by four hidden encoder layers progressively decreasing in size (75%, 50%, 33%, and 25% of the input layer's dimensions).
- The subsequent decoder layers mirrored the sizes of the encoders but in ascending order (starting from 33%).
- Additionally, three other commonly used algorithms for anomaly detection, namely Local Outlier Factor (LOF), One-Class SVM, and Isolation Forest, were trained.
- Autoencoders achieved a 100% True Positive Rate (TPR), while LOF and SVM exhibited similar high TPRs, outperforming Isolation Forest, which demonstrated lower and more inconsistent TPR values.

2.4 <u>Detecting IoT Botnet Attacks Using Machine Learning Methods</u>

- The researchers utilized network data sets from Provision 737E model security cameras, specifically the N-BaIoT data sets.
- Both supervised and unsupervised learning techniques were employed using the Weka program.
- The data set examined in this study consisted of 115 features, but utilizing all of them posed software and hardware challenges.

- To address this, the dimensionality of the data set was reduced from 115 features to 10 dimensions to ensure smooth and accurate feature extraction.
- For supervised learning, the J48 algorithm, a decision tree classification algorithm, was utilized, achieving an accuracy rate of 99.95%.
- Unsupervised learning was performed using the Expectation Maximization algorithm, resulting in the formation of three clusters, with an accuracy percentage of 76.73%.
- The results were compared with nine other studies that employed various machine learning algorithms and datasets.

2.5 Security Attacks in IoT: A Survey

- IoT technology operates across three layers: perception, network, and application layers.
- The perception layer's goal is to gather environmental data using sensors and transmit it to the network layer. The network layer's goal is to transmit data from the perception layer to designated information processing systems through reliable networks like the internet or mobile networks. The application layer is responsible for achieving the IoT's objective of creating a smart environment.
- Adversaries can target IoT systems through physical vulnerabilities, such as damaging or tampering with nodes, exploiting flaws in routing and network protocols from within the network, deploying malicious software, or launching attacks on encryption strategies.
- Based on these vulnerabilities, attacks are classified into four categories: physical attacks, network attacks, software attacks, and encryption attacks.
- Here is the list of some security attacks:

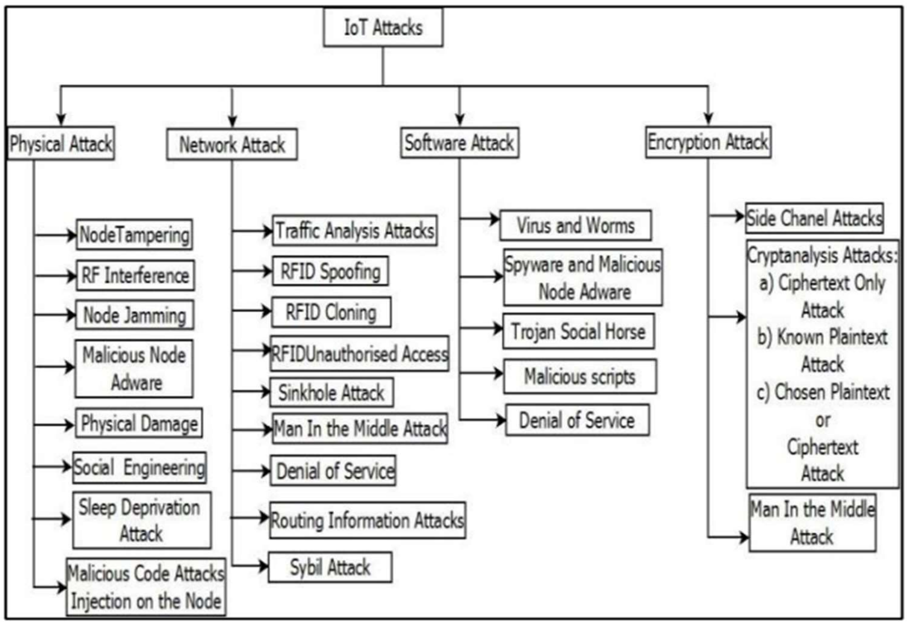

Figure 2 – List of IoT Attacks

2.6 Hawkware: Network Intrusion Detection based on Behavior Analysis with ANNs on an IoT Device

- NIDS is utilized for analyzing network data traffic and detecting attacks in networked systems. Researchers prefer NIDS for IoT security due to the interconnected nature of IoT systems, which form an ecosystem rather than standalone devices.
- Two deployment strategies for NIDS are centralized and distributed. In a distributed approach, NIDS is placed at multiple strategic points such as routers or gateways instead of being centralized in one location.
- Hawkware is a lightweight distributed NIDS that employs an ANN-based approach to detect attacks on IoT devices without relying on deep packet inspection (DPI), achieving higher accuracy compared to the latest NIDS solutions.
- Hawkware demonstrates resilience against advanced attacks, including traffic mimicry and system call mimicry, by correlating behavioral patterns.
- Hawkware consists of five components and their functions are summarized

as follows:

1. the packet analyzer (PA) analyzes network packet headers and extracts relevant features;
2. the system call logger (SCL) records the device behavior and extracts features related to incoming/outgoing network packets;
3. the feature preprocessor (FP) aggregates both extracted features and transfer them as inputs to Hawknet;
4. the Hawknet controller (HC) examines the Hawknet's outputs and determines the existence of intrusions;
5. the Hawknet quantifies the degree of anomaly.

CHAPTER 3

PROPOSED METHODOLOGY

3.1 Machine Learning and Process

The machine learning (ML) process involves several steps that enable computers to learn from data and make predictions or decisions based on that data. The steps typically involved in the ML process are as follows:

1. **Data Collection:** In this step, the relevant data for the problem at hand is collected and prepared for analysis. This can involve cleaning and transforming the data to make it suitable for use in ML algorithms.

2. **Data Preprocessing:** The collected data is then preprocessed to eliminate any errors or inconsistencies. This can involve tasks such as data normalization, feature scaling, and data imputation.

3. **Feature Engineering:** In this step, the relevant features for the problem at hand are identified and extracted from the preprocessed data. These features are then transformed into a format suitable for use in ML algorithms.

4. **Model Selection:** The next step is to select an appropriate ML model for the problem at hand. This involves evaluating different models and selecting the one that is best suited to the data and the problem.

5. **Model Training:** Once a suitable model has been selected, it is trained on the preprocessed data. This involves feeding the data into the model and adjusting its parameters to minimize the error or loss function.

6. **Model Evaluation:** After the model has been trained, it is evaluated to determine its performance. This involves using a separate dataset to test the model's ability to make accurate predictions or decisions.

7. Model Deployment: Once the model has been evaluated and its performance is deemed satisfactory, it is deployed in a real-world setting to make predictions or decisions based on new data.

Overall, the ML process involves several iterative steps that allow machines to learn from data and make decisions or predictions based on that data. The process requires careful attention to data quality, feature selection, model selection, and evaluation to ensure that the model is accurate, robust, and suitable for the problem at hand.

3.1.1 Machine Learning Models(used)

1. Sequential model:

- A sequential model is a type of neural network architecture in deep learning that is designed to process a sequence of data. It is widely used for tasks such as natural language processing, speech recognition, and time-series analysis.
- In a sequential model, the input data is processed sequentially, with each input being fed into the model one at a time. The model consists of a sequence of layers, with each layer processing the input data and passing the output to the next layer in the sequence.

- The simplest type of sequential model is the feedforward neural network, which consists of a single input layer, one or more hidden layers, and an output layer. In a feedforward network, the input data is fed forward through the layers, with each layer transforming the input data and passing it to the next layer until it reaches the output layer.

- Another type of sequential model is the recurrent neural network (RNN), which is designed to process sequences of data with a temporal component, such as time-series data or natural language text. In an RNN, the output of each layer is fed back as input to the same layer in the next time step, allowing the model to incorporate information from previous time steps.

- A variant of the RNN is the long short-term memory (LSTM) network, which is designed to address the issue of vanishing gradients in standard RNNs. An LSTM network uses memory cells to store information from previous time steps, allowing it to retain information over long sequences.

- Overall, sequential models are powerful tools for processing sequences of data and are widely used in deep learning applications. They are highly flexible and can be adapted to a wide range of tasks by adjusting the number and type of layers in the model.

2. Decision Tree:

- A decision tree is a type of supervised machine learning algorithm that is used for classification and regression analysis. It works by recursively partitioning the data into subsets based on the values of one or more input variables, until a leaf node is reached that contains a decision or prediction.

- The basic structure of a decision tree consists of a root node, which represents the entire dataset, and a series of internal nodes, which represent decisions based on the values of input variables. Each internal node has one or more child nodes, which represent the subsets of the data that satisfy the decision. The leaf nodes represent the final decisions or predictions based on the values of the input variables.

- The decision tree algorithm works by selecting the input variable that best separates the data into subsets with the least amount of impurity or uncertainty. The most common measures of impurity used in decision trees are entropy and Gini index. The algorithm then recursively partitions the data based on the selected variable until a stopping criterion is met, such as a minimum number of samples in a leaf node or a maximum depth of the tree.

- Decision trees have several advantages, including their simplicity, interpretability, and ability to handle both categorical and numerical input variables. They can also be used for feature selection by identifying the most important input variables for the problem at hand.

- However, decision trees can also suffer from several limitations, including overfitting, which occurs when the tree is too complex and fits the training data too closely, and instability, which occurs when small changes in the data can result in large changes in the structure of the tree.

- Overall, decision trees are a powerful tool for classification and regression analysis in machine learning, and are widely used in a variety of domains, including finance, healthcare, and marketing.

3. K-nearest Neighbour:

- K-Nearest Neighbor (KNN) is a supervised machine learning algorithm used for classification and regression analysis. It is a non-parametric algorithm that makes predictions based on the distance between a new data point and existing data points in a dataset.

- The basic idea behind KNN is that similar data points tend to be close to each other in a high-dimensional space. To make a prediction for a new data point, KNN looks at the K nearest neighbors in the dataset, where K is a user-defined parameter. The prediction is then based on the majority class or average value of the K nearest neighbors.

- The distance between data points is typically measured using Euclidean distance or other distance metrics such as Manhattan distance or cosine similarity. The choice of distance metric can have a significant impact on the performance of the algorithm, and different metrics may be more appropriate for different types of data.

- One of the advantages of KNN is its simplicity and ease of implementation. It also requires no training time, as all the training data is used to make predictions at test time. KNN can handle both numerical and categorical data, making it a versatile algorithm that can be used in a wide range of applications.

- However, KNN can suffer from several limitations, including the need to store the entire training dataset, which can be computationally expensive for large datasets. The performance of the algorithm can also be sensitive to the choice of K, and the algorithm may not perform well when the dataset has imbalanced classes or noisy data.

- Overall, KNN is a useful algorithm for classification and regression analysis, and is widely used in areas such as pattern recognition, image recognition, and recommendation systems.

4. Support vector machine:

- Support Vector Machine (SVM) is a supervised machine learning algorithm that is used for classification and regression analysis. It works by finding a hyperplane in a high-dimensional space that maximally separates the classes or best fits the regression line.

- The basic idea behind SVM is to transform the input data into a higher-dimensional space where the classes can be separated by a hyperplane. The hyperplane is chosen to maximize the margin, which is the distance between the hyperplane and the nearest data points of each class. This maximization of margin helps to ensure that the classifier is robust to noise and generalizes well to new data.

- SVM can handle both linear and non-linear classification and regression problems. In the case of non-linear problems, SVM uses a kernel function to transform the data into a higher-dimensional space where it can be separated by a hyperplane. The most commonly used kernel functions are polynomial and radial basis function (RBF) kernels.

- One of the advantages of SVM is its ability to handle high-dimensional data and deal with noise and outliers effectively. It also has a strong theoretical foundation, which helps to ensure that the model is robust and generalizes well to new data.

- However, SVM can also suffer from several limitations, including the need to select appropriate kernel functions and tune hyperparameters, which can be time-consuming and require domain expertise. It can also be computationally expensive for large datasets, and the interpretability of the model can be limited.

- Overall, SVM is a powerful algorithm for classification and regression analysis, and is widely used in applications such as image classification, bioinformatics, and text classification.

5. Gaussian Naïve Bayes:

- Gaussian Naive Bayes is a supervised machine learning algorithm used for classification. It is based on Bayes' theorem, which states that the probability of a hypothesis (in this case, a class label) given the evidence (in this case, the features of a data point) is proportional to the likelihood of the evidence given the hypothesis, multiplied by the prior probability of the hypothesis.

- The "Naive" part of the algorithm refers to the assumption that all the features of a data point are independent of each other, given the class label. This assumption simplifies the calculation of the likelihood and makes the algorithm computationally efficient.

- Gaussian Naive Bayes assumes that the distribution of the features for each class label is Gaussian (i.e., normally distributed). This means that the algorithm estimates the mean and standard deviation of each feature for each class label, and uses these estimates to calculate the likelihood of a data point belonging to each class label.

- To make a prediction for a new data point, Gaussian Naive Bayes calculates the posterior probability of the data point belonging to each class label, and chooses the class label with the highest probability as the prediction.

- One of the advantages of Gaussian Naive Bayes is its simplicity and ease of implementation. It is also computationally efficient and can handle high-dimensional data well. It can perform well in situations where the number of features is large compared to the number of data points.

- However, Gaussian Naive Bayes assumes that the features are independent, which may not be true in some cases. It can also be sensitive to outliers and may not perform well when the distribution of the data is not Gaussian.

- Overall, Gaussian Naive Bayes is a useful algorithm for classification, especially in situations where the number of features is high and the data is well-behaved. It is widely used in applications such as spam filtering, text classification, and sentiment analysis.

6. Random forests:

- Random Forest is a supervised machine learning algorithm used for classification and regression analysis. It is an ensemble method that combines multiple decision trees to improve the accuracy and robustness of the model.

- The basic idea behind Random Forest is to create multiple decision trees using different subsets of the training data and different subsets of the features. Each decision tree is trained on a random subset of the data and features, and makes a prediction based on the majority vote of the trees in the forest.

- Random Forest uses a technique called "bagging" (bootstrap aggregating) to generate the different subsets of the data and features. Bagging randomly samples the training data with replacement, creating multiple datasets of the same size as the original. The decision trees are then trained on these datasets, and the final prediction is based on the majority vote of the trees.

- Random Forest also uses a technique called "feature bagging" or "random subspace method" to generate the different subsets of features. Feature bagging randomly selects a subset of the features for each decision tree, creating different decision trees that use different combinations of features.

- One of the advantages of Random Forest is that it is a robust and accurate algorithm that can handle high-dimensional data and deal with noise and outliers effectively. It also has a low risk of overfitting, which occurs when a model becomes too complex and fits the training data too closely, leading to poor generalization to new data.

- However, Random Forest can be computationally expensive and may require more resources than other algorithms. It is also less interpretable than other algorithms, making it difficult to understand how the model makes its predictions.

- Overall, Random Forest is a powerful algorithm for classification and regression analysis that is widely used in applications such as image recognition, text classification, and medical diagnosis. It is particularly useful when dealing with complex and high-dimensional data, and when accuracy and robustness are important.

3.1.2 Machine Learning Applications

Machine learning has numerous applications across various industries and domains. Some of the popular applications of machine learning are:

1. Image and Speech Recognition: Machine learning algorithms have been trained on a large dataset of images and audio samples, which have enabled them to accurately recognize images and speech. These applications are used in facial recognition, voice assistants, and security systems.

2. Natural Language Processing (NLP): NLP is the ability of machines to understand, interpret, and respond to human language. Machine learning models are used to analyze and interpret text and speech data, and extract valuable insights. NLP is used in chatbots, voice assistants, and sentiment analysis.

3. Recommendation Systems: Machine learning models are used to predict and recommend products or services to users based on their past behavior and preferences. These systems are used in e-commerce, music and video streaming platforms, and social media.

4. Fraud Detection: Machine learning algorithms are used to detect fraudulent activities in financial transactions by analyzing patterns and identifying anomalies. These systems are used in banking and finance to prevent fraudulent activities.

5. Healthcare: Machine learning is used in healthcare to analyze medical data and provide insights into patient health. Machine learning models are used to diagnose diseases, identify risk factors, and develop treatment plans.

6. Predictive Maintenance: Machine learning models are used to predict equipment failures by analyzing data from sensors and identifying patterns. This enables proactive maintenance and reduces downtime in industries such as manufacturing and transportation.

7. Autonomous Vehicles: Machine learning models are used in self-driving cars to analyze data from sensors and make decisions in real-time. This technology has the potential to transform the transportation industry.

8. Gaming: Machine learning models are used in gaming to develop intelligent agents that can learn and adapt to the player's behavior. This enables more immersive and challenging gameplay.

9. Agriculture: Machine learning models are used in agriculture to predict crop yields, detect plant diseases, and optimize irrigation. This enables more efficient and sustainable agriculture practices.

These are just a few examples of the numerous applications of machine learning across different industries and domains.

3.1.3 IoT and ML Integration

The integration of IoT and machine learning (ML) is becoming increasingly important as IoT devices continue to generate large volumes of data. Machine learning algorithms are used to analyze this data and extract valuable insights, which can be used to improve the performance of IoT devices and the systems they operate in.

There are several ways in which IoT and ML can be integrated:

1. Predictive Maintenance: IoT sensors can be used to monitor the performance of equipment and detect anomalies in real-time. Machine learning models can then be used to analyze this data and predict equipment failures before they occur. This enables proactive maintenance and reduces downtime.

2. Anomaly Detection: IoT devices generate a lot of data, and it can be difficult to identify anomalies manually. Machine learning algorithms can be trained to identify patterns in this data and detect anomalies automatically. This can be used to improve security, detect fraudulent activities, and identify potential problems before they occur.

3. Energy Management: IoT devices can be used to monitor energy usage in real-time. Machine learning models can then be used to analyze this data and optimize energy consumption, reducing costs and improving energy efficiency.

4. Smart Buildings: IoT devices can be used to monitor the performance of buildings and adjust systems automatically to optimize energy consumption and improve occupant comfort. Machine learning models can be used to analyze this data and make predictions about future performance.

5. Agriculture: IoT devices can be used to monitor crop yields, soil moisture, and other environmental factors. Machine learning models can then be used to analyze this data and provide insights into crop health, enabling farmers to optimize crop yields and reduce costs.

6. Autonomous Vehicles: IoT sensors can be used to monitor road conditions, traffic, and weather. Machine learning algorithms can then be used to analyze this data and make decisions in real-time, enabling self-driving cars to operate safely and efficiently.

7. Personalized Healthcare: IoT devices can be used to monitor patient health in real-time, generating large volumes of data. Machine learning models can then be used to analyze this data and provide personalized treatment plans based on individual patient needs.

These are just a few examples of how IoT and machine learning can be integrated to improve performance, reduce costs, and enhance user experiences. As more IoT devices are deployed, the integration of machine learning will become increasingly important to enable more intelligent and efficient systems.

3.2 Experimental Dataset (N-BaIoT)

The N-BaIoT dataset [11] is a benchmark dataset for evaluating machine learning models in the context of Internet of Things (IoT) security. It was introduced in 2018 by researchers from the University of New Brunswick, Canada.

The dataset contains 9.2 million network flows generated by a network of IoT devices, which were infected with a variety of malware families. The devices include home automation, entertainment, environmental monitoring, and health monitoring devices. The dataset was created by setting up a testbed network with a variety of IoT devices, and then infecting some of the devices with malware to generate traffic.

The dataset contains nine classes of traffic, including benign traffic and eight malware families. The malware families included in the dataset are Bashlite, Mirai, Gafgyt, Hajime, Tsunami, Wifatch, Doflo, and IoT Reaper. Each malware family exhibits different attack behaviors and targets different types of IoT devices.

The N-BaIoT dataset provides a comprehensive evaluation of machine learning models for IoT security. It is designed to help researchers and practitioners develop and evaluate new security solutions for IoT devices, which are becoming increasingly important in our daily lives. Machine learning models can be trained on the dataset to identify patterns in the network traffic generated by IoT devices and detect malware infections.

The N-BaIoT dataset is a significant contribution to the field of IoT security because it provides a realistic and diverse set of traffic patterns that can be used to evaluate machine learning models. In addition to the malware families and benign traffic, the dataset includes different types of IoT devices, communication protocols, and network topologies. This diversity reflects the complexity of IoT networks in the real world, where devices from different manufacturers may use different protocols to communicate with each other.

One of the challenges in developing machine learning models for IoT security is the limited resources of IoT devices, such as processing power, memory, and energy. Therefore, machine learning models for IoT security must be designed to be lightweight and efficient. The N-BaIoT dataset can be used to evaluate the performance of machine learning models under these constraints.

Another challenge in developing machine learning models for IoT security is the lack of labeled datasets. The N-BaIoT dataset provides labeled data that can be used to train and test machine learning models. This is particularly important for supervised learning algorithms, which require labeled data to learn patterns in the network traffic.

In addition to developing machine learning models for IoT security, the N-BaIoT dataset can be used to evaluate other security solutions, such as intrusion detection systems, firewalls, and anomaly detection systems. The dataset can also be used to evaluate the effectiveness of different security measures, such as network segmentation, access control, and encryption.

Overall, the N-BaIoT dataset is a valuable resource for the research community to evaluate the performance of machine learning models in the context of IoT security. It provides a realistic and diverse set of traffic patterns that can be used to train and test machine learning models for detecting malware infections in IoT devices.

3.3 Data Pre-processing

The objective of preprocessing is to formulate the data into a form that will help in boosting the efficiency of the machine learning models. We have used the Standardization of data that converts the structure of the dataset into one common format of data.

This can be achieved using the subsequent equations:

$$x_{standardized} = \frac{x - \mu}{\sigma}$$

Where, $\mu = \frac{1}{N}\sum_{i=1}^{N} x_i$,

$$\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - \mu)^2}$$

where:

- $x_{\text{standardized}}$ is the standardized value.
- x is the original value.
- μ is the mean value.
- σ is the standard deviation value.

3.4 Experiment Setup

Machine learning model's performance is significantly influenced by the hardware. We used the same hardware throughout the experiment.

The following configurationswere used to conduct the experiment:

1) Hardware - Intel® Core™ i5-8265U CPU @ 1.8 GHz, 8 GB RAM, NVIDIA GeForce MX230 GPU@ 1.5 GHz, 2GB Memory running on Windows 10
2) Software – Python 3.7

We have used some python libraries to assist us in ourimplementation:

- ***Keras*****:** It is an API developed by Google using python language for implementing neural networks.
- ***Numpy***: NumPy is a library for Python language that can be used for complex mathematical operations. NumPy provides support for working with large-sized multi-dimensional vectors and arrays.
- ***Pandas***: Pandas is a Python package that provides fast, flexible, and expressive data structures designed to work with "labeled" data not only with ease, but also intuitively. It is built on top of the NumPy package.
- ***Sklearn***: Scikit-learn provides a range of supervised and unsupervised learning algorithms. David Cournapeau started developing sklearn as a Google Summer of Code (GSoC) project in 2007.
- ***Tensorflow***: TensorFlow is an open-source software library for high-performance numerical computation. It bundles together a slew of machine learning and deep learning models and algorithms. It allows developers to create dataflow graphs and can be deployed on most devices.

CHAPTER 4

RESULTS AND DISCUSSION

ML Model	Metrics	Benign	Bashlite					Mirai					Accuracy
			Combo	Junk	Scan	TCP	UDP	UDP	ACK	Scan	SYN	UDP plain	
Sequential	Precision	1	0.97	1	0.5	0	1	0.98	1	1	0.98	1	0.901
	Recall	1	0.98	1	1	0	0.98	0.97	0.99	1	1	1	
	F1-score	1	0.98	1	0.67	0	0.99	0.97	1	1	0.99	1	
KNN	Precision	1	1	0.99	1	1	1	1	0.98	1	1	1	0.996
	Recall	1	0.99	1	1	1	1	0.98	0.99	1	1	1	
	F1-score	1	1	1	1	1	1	0.99	0.99	1	1	1	
SVM	Precision	1	0.98	0.84	1	0.5	0	1	1	1	1	1	0.888
	Recall	1	0.81	0.98	1	1	0	1	1	1	1	1	
	F1-score	1	0.89	0.91	1	0.66	0	1	1	1	1	1	
Random Forest	Precision	1	1	1	1	1	0.95	1	1	1	1	1	0.994
	Recall	1	1	1	1	0.94	1	1	1	1	1	1	
	F1-score	1	1	1	1	0.97	0.97	1	1	1	1	1	
Gaussian Naïve Bayes	Precision	1	0.51	0.7	0.68	0.5	0	1	0.82	1	1	1	0.758
	Recall	0.56	0.97	0.07	1	1	0	0.97	1	1	1	0.81	
	F1-score	0.71	0.67	0.13	0.81	0.66	0	0.98	0.9	1	1	0.89	
Decision Tree	Precision	1	1	1	1	1	0.95	1	1	1	1	1	0.994
	Recall	1	1	1	1	0.94	1	1	1	1	1	1	
	F1-score	1	1	1	1	0.97	0.97	1	1	1	1	1	

Table 4 - Classification Report of Provision PT 737E Security Camera

It can be seen in “Table 3”, the performance of six machine learning methods: Sequential, Support Vector Machine (SVM), K-Nearest Neighbor (KNN), decision tree,random forest, and Gaussian Naive Bayes (GNB).

These are the performance metrics for different machine learning models (Sequential, KNN, SVM, Random Forest, Gaussian Naive Bayes, and Decision Tree) for detecting various types of attacks (Benign, Mirai, Bashlite, UDP, ACK, Scan, SYN, UDPplain, Combo, Junk, and TCP).

Each metric (precision, recall, and F1-score) measures a different aspect of the model's performance:

- **Precision**: measures the percentage of correct positive predictions out of all positive predictions. In other words, it tells us how many of the predicted attacks were actually correct. A higher precision indicates fewer false positives.

- **Recall**: measures the percentage of correct positive predictions out of all actual positives. In other words, it tells us how many of the actual attacks were correctly predicted. A higher recall indicates fewer false negatives.

- **F1-score**: is the harmonic mean of precision and recall, and it provides an overall measure of the model's performance. It ranges from 0 to 1, where a higher score indicates better performance.

Looking at the table, we can see that each model performs differently for different types of attacks. For example, the SVM model performs well for detecting benign traffic, while the Random Forest model performs well for detecting most types of attacks. The Gaussian Naive Bayes model performs relatively poorly compared to the other models.

Overall, the choice of model and its parameters should be carefully tuned depending on the specific task and dataset. These performance metrics can help in selecting the most appropriate model for a given problem.

ML Model	**Metrics**	**Benign**	**Bashlite**					**Mirai**					**Accuracy**
			Combo	**Junk**	**Scan**	**TCP**	**UDP**	**UDP**	**ACK**	**Scan**	**SYN**	**UDP plain**	
Sequential	Precision	1	1	1	0.48	0	1	0.89	1	1	1	1	0.893
	Recall	1	0.89	1	1	0	1	1	1	1	1	1	
	F1-score	1	0.94	1	0.65	0	1	0.94	1	1	1	1	
KNN	Precision	1	1	0.99	1	1	1	0.99	0.98	1	1	1	0.995
	Recall	1	0.99	1	1	1	1	0.98	0.99	1	1	1	
	F1-score	1	0.99	0.99	1	1	1	0.98	0.98	1	1	1	
SVM	Precision	1	0.98	0.89	1	0.5	0.53	1	1	1	1	1	0.901
	Recall	1	0.86	0.98	1	0	1	1	1	1	1	1	
	F1-score	1	0.92	0.93	1	0	1	1	1	1	1	1	
Random Forest	Precision	1	1	1	1	1	0.98	1	1	1	1	1	0.998
	Recall	1	1	1	1	0.98	1	1	1	1	1	1	
	F1-score	1	1	1	1	0.99	0.99	1	1	1	1	1	
Gaussian Naïve Bayes	Precision	1	1	0.56	0.99	0	0	0.91	0.99	0.36	1	0.91	0.737
	Recall	1	0.12	1	1	0	0	0.99	0.95	1	1	0.99	
	F1-score	1	0.22	0.71	0.99	0	0	0.95	0.97	0.53	1	0.98	
Decision Tree	Precision	1	1	1	1	1	0.98	1	1	1	1	1	0.998
	Recall	1	1	1	1	0.98	1	1	1	1	1	1	
	F1-score	1	1	1	1	0.99	0.99	1	1	1	1	1	

Table 5 - Classification Report of EcoBee Thermostat

It can be seen in Table 4", the performance of six machine learning methods: Sequential, Support Vector Machine (SVM), K-Nearest Neighbor (KNN), decision tree, random forest, and Gaussian Naive Bayes (GNB). Decision tree and Random forest have almost same results.

It seems that provided table contains various metrics for different machine learning models on a dataset related to network security. The dataset seems to contain different types of network attacks and benign traffic. The table shows various metrics such as accuracy, precision, recall, and F1-score for different models such as Sequential, KNN, SVM, Random Forest, Gaussian Naive Bayes, and Decision Tree.

The metrics such as accuracy, precision, recall, and F1-score are commonly used in evaluating the performance of machine learning models. Accuracy measures how many predictions are correct, while precision measures how many of the predicted positive cases are actually positive. Recall measures how many actual positive cases are correctly predicted, while F1-score is a harmonic mean of precision and recall, which gives a balanced measure of model performance.

Looking at the table, it seems that Random Forest has the highest precision, recall, and F1-score for most of the attacks and benign traffic, which indicates that it is performing well on this dataset. However, it is important to note that the performance of a model can depend on various factors such as the quality and size of the dataset, the features used in the model, and the hyperparameters of the model. Therefore, it is always a good practice to evaluate different models on various datasets and select the one that performs best on the task at hand.

Both the table shows how well different machine learning models perform at classifying different types of network traffic as either benign or associated with various types of malicious activity. The rows represent the models used, while the columns show different performance metrics used to evaluate their performance.

For each type of network traffic, such as Bashlite or Mirai SYN, the table shows the accuracy, precision, recall, and F1-score for each machine learning model. These metrics indicate how well the model was able to correctly classify the network traffic as either benign or malicious.

In general, higher values for accuracy, precision, recall, and F1-score indicate better performance by the model. However, the specific thresholds for what constitutes "good" performance can vary depending on the context and the goals of the analysis.

There could be several reasons why a GNB (Gaussian Naive Bayes) model is not performing well. Here are some possible reasons:

1. **Feature Independence Violation:** GNB assumes that features are independent of each other. If this assumption is violated, it can lead to poor performance. If the features in the dataset are not truly independent, it might be beneficial to consider other algorithms that can capture dependencies between features.

2. **Incorrect Assumption of Gaussian Distribution:** GNB assumes that the features follow a Gaussian distribution. If the features in the dataset have a different distribution (e.g., multimodal or skewed), GNB may not be appropriate and could result in poor performance.

3. **Insufficient Training Data:** GNB, like any machine learning algorithm, requires an adequate amount of training data to learn patterns and make accurate predictions. If the dataset is small or imbalanced, it can lead to poor performance.

4. **Irrelevant or Noisy Features:** GNB can be sensitive to irrelevant or noisy features. If the dataset contains features that do not provide meaningful information for the classification task or include noisy data, it can negatively impact the model's performance.

5. **Class Imbalance:** If the classes in the dataset are imbalanced, meaning some classes have significantly more instances than others, GNB may have difficulty accurately predicting the minority class.

6. **Violation of Assumption of Conditional Independence:** GNB assumes that features are conditionally independent given the class label. If this assumption is violated, it can lead to poor performance. In such cases, more advanced models that can capture dependencies between features, such as decision trees or

ensemble methods, might yield better results.

7. **Outliers in the Data:** GNB can be sensitive to outliers since it relies on the mean and variance of features. If the dataset contains outliers, they can distort the estimation of the distribution parameters and negatively affect the model's performance.

8. **Inadequate Preprocessing:** Insufficient or improper preprocessing of the data can impact the performance of any machine learning model, including GNB.

9. **Overfitting or Underfitting:** GNB, like other models, can suffer from overfitting or underfitting. Overfitting occurs when the model learns the training data too well but fails to generalize to new, unseen data. Underfitting happens when the model is too simple to capture the underlying patterns in the data.

10. **Inherent Limitations of GNB:** GNB is a simple and fast algorithm that makes strong assumptions about the data. However, these assumptions might not hold in all scenarios. It may struggle with complex relationships or highly correlated features. If the data exhibits such characteristics, using more advanced algorithms that can capture nonlinear relationships or handle feature interactions might be more suitable.

There can be various reasons why a Support Vector Machine (SVM) may not be performing well. Here are some possible reasons:

1. **Insufficiently Scaled Features:** SVMs are sensitive to the scale of features. If the features have significantly different scales, it can affect the SVM's performance. Make sure to scale or normalize the features appropriately before training the SVM to ensure each feature contributes equally to the model.

2. **Improper Kernel Selection:** SVMs utilize different types of kernels (e.g., linear, polynomial, radial basis function) to map the data into higher-dimensional spaces. The choice of kernel can greatly impact the model's performance. If the chosen

kernel is not appropriate for the underlying data distribution or problem, the SVM may not perform well.

3. **Incorrect Hyperparameter Settings:** SVMs have several hyperparameters that need to be properly tuned to achieve optimal performance. If the hyperparameters such as the regularization parameter (C) or the kernel parameters are not set correctly, the SVM might be underfitting or overfitting the data. Performing a hyperparameter search using techniques like grid search or random search can help find the optimal hyperparameter values.

4. **Imbalanced Data:** If the dataset is imbalanced, meaning one class has significantly more instances than the others, SVMs may not perform well. SVMs strive to find a decision boundary that maximizes the margin, which can be biased towards the majority class.

5. **Insufficient Training Data:** SVMs, like other machine learning algorithms, require an adequate amount of training data to learn accurate decision boundaries. If the dataset is small or contains limited informative samples, the SVM's performance may suffer.

6. **Noisy or Outlier-Prone Data:** SVMs can be sensitive to noisy or outlier-prone data points. Outliers can disrupt the optimization process or mislead the SVM's decision boundary. Cleaning the data by removing or correcting noisy data points or considering outlier detection techniques can help mitigate this issue.

7. **Non-Linearly Separable Data:** SVMs are inherently designed to handle linearly separable data. If the data is not linearly separable and the separation requires a non-linear decision boundary, a linear SVM may not perform well.

8. **Curse of Dimensionality:** When the number of features is significantly larger than the number of samples, SVMs may struggle to find meaningful patterns in the data. This is known as the "curse of dimensionality."

9. **Violation of Assumptions:** SVMs have certain assumptions, such as the data being independent and identically distributed and the classes being separable by a hyperplane. If these assumptions are violated, the SVM's performance can be negatively impacted. Understanding the characteristics of the data and whether it aligns with the assumptions of SVMs is crucial in assessing their performance.

10. **Inherent Data Complexity:** Some datasets inherently possess complex patterns or non-linear relationships that are challenging for SVMs to capture.

To improve the performance of Gaussian Naive Bayes (GNB) and Support Vector Machine (SVM) models, we can consider the following solutions:

For Gaussian Naive Bayes (GNB):

1. **Feature Engineering:** Analyze and preprocess the features to enhance their relevance and quality. This can involve techniques such as feature selection, dimensionality reduction (e.g., Principal Component Analysis), or creating new informative features.

2. **Addressing Feature Independence Violation:** If the assumption of feature independence is violated, we can explore more advanced models that can capture dependencies between features, such as decision trees, random forests, or gradient boosting algorithms.

3. **Handling Non-Gaussian Distributions:** If the data exhibits non-Gaussian distributions, we can consider transforming the features to make them more Gaussian-like. Techniques such as logarithmic, exponential, or power transformations may be useful in such cases.

4. **Data Augmentation:** If the dataset is limited, we can employ data augmentation techniques to generate synthetic samples, thereby increasing the size of the training set. This can help improve the model's ability to generalize.

5. **Balancing Class Distribution:** If the classes are imbalanced, we can address the issue by employing techniques like oversampling the minority class, undersampling the majority class, or using appropriate class weights during training to ensure equal representation and prevent bias towards the majority class.

6. **Regularization:** Introducing regularization techniques, such as adding a penalty term to the likelihood function, can help prevent overfitting in GNB. This can be particularly useful when dealing with high-dimensional data or cases with a limited number of training samples.

For Support Vector Machines (SVM):

1. **Feature Scaling:** Ensure that the features are properly scaled or normalized before training the SVM. This can be achieved by techniques like standardization or min-max scaling to ensure that each feature contributes equally to the model.

2. **Hyperparameter Tuning:** Perform an extensive search or use optimization techniques, such as grid search or random search, to find the optimal values for hyperparameters like the regularization parameter (C) and kernel parameters. This can significantly impact the SVM's performance.

3. **Kernel Selection:** Experiment with different types of kernels and their parameters to find the one that best captures the underlying patterns in the data. Linear, polynomial, radial basis function (RBF), and sigmoid kernels are commonly used, but other specialized kernels may be appropriate for specific datasets.

4. **Non-Linear SVMs:** If the data is not linearly separable, consider employing non-linear SVM variants like the kernel SVM, which can map the data into higher-dimensional spaces to find non-linear decision boundaries.

5. **Handling Outliers:** Outliers can adversely affect SVM performance. Consider removing or correcting outliers, or explore robust SVM formulations that are less

affected by outliers.

6. **Ensemble Methods:** Combine multiple SVMs or use ensemble techniques like bagging or boosting to improve performance. This can help capture diverse patterns in the data and reduce the risk of overfitting.

7. **More Complex Models:** For complex datasets with intricate patterns, consider using more advanced models like deep learning algorithms or ensemble methods, which can capture non-linear relationships and handle high-dimensional data more effectively.

It's important to note that the performance of GNB and SVM models can also depend on the specific characteristics of the dataset and problem. Experimenting with different approaches, analyzing the results, and iteratively refining the models can help us achieve better performance

CHAPTER 5

CONCLUSION AND FUTURE SCOPE

The growth of IoT is phenomenal, with the market having an incredible compound per annum growth rate. According to some researchers, it is estimated that the industry of the Internet of Things is likely to increase by 18% to 14.4 billion active devices. When the Russia-Ukraine crisis ends and international trade gets back on track, it is likely that by 2025, there will be approximately 27 billion connected IoT devices. Decision Tree, K nearest neighbor and random forest showed extremely high accuracy in detecting Mirai and Bashlite Botnet attacks. But, IoT attacks are becoming more complex with the advancement in technology and knowledge about connected systems. Cybercriminals have now become more efficient and can attack different components of an IoT network. Thus, we should devise new methods and techniques to counter IoT cyberattacks.

Future work of this paper would be to:

1) Extend our models by using complete dataset and at the same time ensure that the data remain balanced to avoid overfitting.
2) Hyperparameter tune our models to achieve higher F1 scores in predicting Bashlite attacks.
3) Use new Machine Learning techniques that provided advanced and accurate models for detecting botnet attacks.

REFERENCES

1. J. Ploennigs, J. Cohn and A. Stanford-Clark, "The Future of IoT," in IEEE Internet of Things Magazine, vol. 1, no. 1, pp. 28-33, SEPTEMBER 2018, doi: 10.1109/IOTM.2018.1700021.

2. Nozomi Networks "OT/IoT Security Report", (2022) https://www.cisa.gov/uscert/sites/default/files/ICSJWGArchive/QNL_SEP_22/Nozomi-Networks-OT-IoT-Security-Report-ES2022-1H_508c.pdf.

3. Y. Jin, "Towards Hardware-Assisted Security for IoT Systems," 2019 IEEE Computer Society Annual Symposium on VLSI (ISVLSI), 2019, pp. 632-637, doi: 10.1109/ISVLSI.2019.00118.

4. Leonard, Justin & Xu, Shouhuai & Sandhu, Ravi. (2009). "A Framework for Understanding Botnets," Proceedings - International Conference on Availability, Reliability and Security, ARES 2009. 917-922, doi: 10.1109/ARES.2009.65.

5. K. Angrishi, "Turning internet of things (iot) into internet of vulnerabilities (iov): IoT botnets," 2017., doi: 10.48550/arXiv.1702.03681.

6. C. Kolias, G. Kambourakis, A. Stavrou and J. Voas, "DDoS in the IoT: Mirai and Other Botnets," in Computer, vol. 50, no. 7, pp. 80-84, 2017, doi: 10.1109/MC.2017.201.

7. M. Antonakakis, T. April, M. Bailey, M. Bernhard, E. Bursztein, J. Cochran, Z. Durumeric, J. A. Halderman, L. Invernizzi, M. Kallitsis, D. Kumar, C. Lever, Z. Ma, J. Mason, D. Menscher, C. Seaman, N. Sullivan, K. Thomas, and Y. Zhou, "Understanding the mirai botnet," in Proc. of USENIX Security Symposium, 2017.

8. Y. M. P. Pa, S. Suzuki, K. Yoshioka, T. Matsumoto, T. Kasama, and C. Rossow, "Iotpot: Analysing the rise of iot compromises," in Proc. Of USENIX Workshop on Offensive Technologies, 2015.

9. P. P. Shinde and S. Shah, "A Review of Machine Learning and Deep Learning Applications," 2018 Fourth International Conference on Computing Communication Control and Automation (ICCUBEA), 2018, pp. 1-6, doi: 10.1109/ICCUBEA.2018.8697857.

10. C. Sinclair, L. Pierce and S. Matzner, "An application of machine learning to network intrusion detection," Proceedings 15th Annual Computer Security Applications Conference (ACSAC'99), 1999, pp. 371- 377, doi: 10.1109/CSAC.1999.816048.

11. Yair Meidan, Michael Bohadana, Yael Mathov and other, N-Baiot dataset, http://archive.ics.uci.edu/ml/datasets/detection_of_IoT_ botnet_attacks_N_BaIoT.

12. Y. Meidan et al., "N-BaIoT—Network-Based Detection of IoT Botnet Attacks Using Deep Autoencoders," in IEEE Pervasive Computing, vol. 17, no. 3, pp. 12-22, Jul.-Sep. 2018, doi: 10.1109/MPRV.2018.03367731.

13. C. OKUR and M. DENER, "Detecting IoT Botnet Attacks Using Machine Learning Methods," 2020 International Conference on Information Security and Cryptology (ISCTURKEY), 2020, pp. 31-37, doi: 10.1109/ISCTURKEY51113.2020.9307994.

14. Dr. D. Ramyachitra, P. Manikandan, IMBALANCED DATASET CLASSIFICATION AND SOLUTIONS: A REVIEW, International Journal of Computing and Business Research (IJCBR) ISSN (Online) : 2229-6166.

15. M. Alshamkhany, W. Alshamkhany, M. Mansour, M. Khan, S. Dhou and F. Aloul, "Botnet Attack Detection using Machine Learning," 2020 14th International Conference on Innovations in Information Technology (IIT), 2020, pp. 203-208, doi: 10.1109/IIT50501.2020.929906.

<u>**APPENDIX1**</u>

Attack detection on Internet of Things devices using machine learning techniques

Yatharth Sharma
Department of Computer Science and Engineering
KIET Group of Institutions
Ghaziabad, India
yatharth570@gmail.com

Vansh Kumar
Department of Computer Science and Engineering
KIET Group of Institutions
Ghaziabad, India
krvansh30@gmail.com

Himanshi Chaudhary
Department of Computer Science and Engineering
KIET Group of Institutions
Ghaziabad, India
himanshi.chaudhary@kiet.edu

Abstract—**Today, the exponential growth of technology and the boom in connection demand has resulted in significant growth of Internet of Things (IoT) devices. In light of the enormous increase in smart devices, secure communication between them has become a major challenge due to an increase in cyber-attacks. Adding hardware-based security is challenging due to the complex architecture of IoT devices and limited computation ability. Since IoT devices produce enormous amount of data, machine learning (ML) can be utilized to detect attacks on IoT devices. In this study, we propose some Machine Learning models to detect and prevent Botnet IoT attacks. The network-Based Detection of Bait-and-Switch IoT Attacks (N-BaIoT) dataset is used to train our models. We have analyzed the performance of models using precision, recall and F1-score by carrying out multiclass classification. Experimental result shows that some of the proposed models are efficient in detecting botnet attacks with 99% accuracy. We can also extend our model to use complete dataset and new ML techniques.**

Keywords— IoT, machine learning, attack Detection, Botnet, N-BaIoT, Mirai, Bashlite.

I. Introduction

Kevin Ashton in 1999 independently came up with the term "Internet of Things" or IoT. IoT devices are physical computing devices that include smart home devices, sensors, wearables, security devices and other machines. They collect and exchange data online.
[1] With IoT improvements, it has now become possible to meet digitally and cooperate with others. It has also connected various electrical, electronic and mechanical machines to the internet. This gives us the advantage of controlling these machines from anywhere in the world.

With the advancement in technology, it is crucial for us to protect and secure our devices from cyber-attack. In the past year, we have observed an increase in attack numbers and complications. Attackers use new and sophisticated attacks. Vendors that have been a victim of cyber-attack have increased by 27% and affected IoT devices have increased by 19% from the mid of 2021[2].

Currently, IoT networks are attacked through various methods and techniques. [3] Jin Yier provides hardware-assisted protection mechanisms that are more effective in defending against threats and provide less performance overhead. However, it is difficult to install hardware-assisted security defenses in the IoT network. We intend to predict attacks on IoT devices via machine learning techniques.

II. IoT Attacks

IoT attacks are cyber-attacks that access sensitive user information using any IoT device. Attackers typically install malware on the device, damage the device, or access other company personal information. Since IoT devices are not built with proper security as a result they pose a huge security attack. Devices, Communication Channels, and applications and software are some attacking zones.
Different forms of IoT attacks are:

- Physical Tampering – Attackers physically control the devices and steal data from them.
- Eavesdropping – The attacker can take advantage of a weak network between an IoT device and the server to steal data.
- Man-in-the-middle-attack – Cybercriminals can access the data passed between the device and the server by exploiting insecure networks.
- Malicious code injection – The attacker injects malicious code into an application via a web page field.
- Botnet attacks – In this, botnets (networks of infected computers) are used to perform malicious activities and cyberattack.

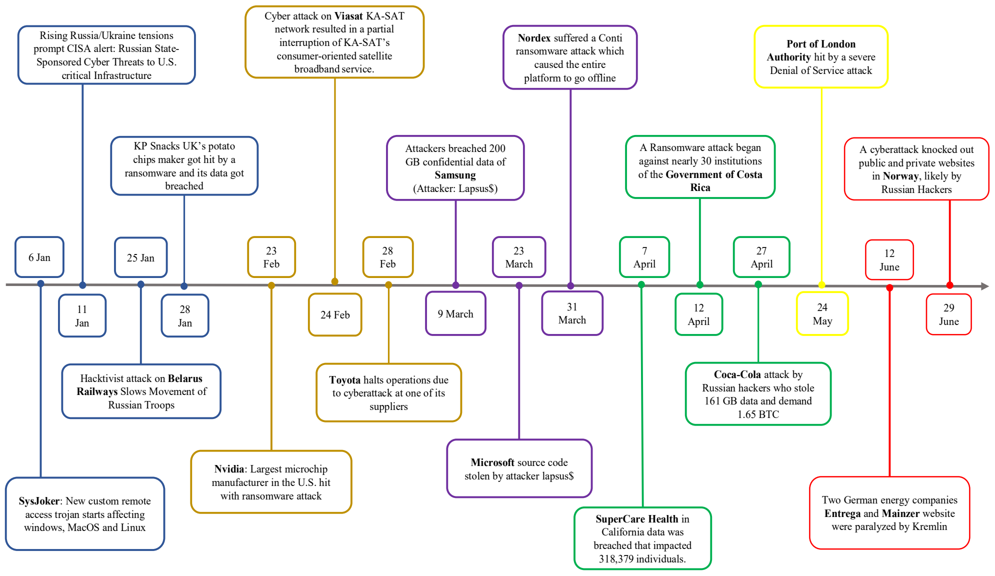
Rising Russia/Ukraine tensions prompt CISA alert: Russian State-Sponsored Cyber Threats to U.S. critical Infrastructure
Cyber attack on Viasat KA-SAT network resulted in a partial interruption of KA-SAT's consumer-oriented satellite broadband service.
Nordex suffered a Conti ransomware attack which caused the entire platform to go offline
Port of London Authority hit by a severe Denial of Service attack
KP Snacks UK's potato chips maker got hit by a ransomware and its data got breached
Attackers breached 200 GB confidential data of Samsung (Attacker: Lapsus$)
A Ransomware attack began against nearly 30 institutions of the Government of Costa Rica
A cyberattack knocked out public and private websites in Norway, likely by Russian Hackers
6 Jan
25 Jan
23 Feb
28 Feb
23 March
7 April
27 April
12 June
11 Jan
28 Jan
24 Feb
9 March
31 March
12 April
24 May
29 June
Hacktivist attack on Belarus Railways Slows Movement of Russian Troops
Toyota halts operations due to cyberattack at one of its suppliers
Coca-Cola attack by Russian hackers who stole 161 GB data and demand 1.65 BTC
Nvidia: Largest microchip manufacturer in the U.S. hit with ransomware attack
Microsoft source code stolen by attacker lapsus$
SysJoker: New custom remote access trojan starts affecting windows, MacOS and Linux
SuperCare Health in California data was breached that impacted 318,379 individuals.
Two German energy companies Entrega and Mainzer website were paralyzed by Kremlin

Fig.1 2022's first-half cyber happenings

In this study, we detect IoT Botnet attacks. There are four phases in the botnet lifecycle - the occurrence phase, the command-and-control phase, the attack phase, and the post-attack phase [4]. Botnets are mostly used for financial theft, information theft, sabotage of services, etc.

Distributed Denial-of-Service (DDoS), Spambots, and Brute force attacks are its types.

DDoS is one of the most vulnerable attacks and can be launched from botnets.

Security specialists have devoted their efforts to describe those botnets and develop countermeasures [5], [6], [7], [8].

Bashlite Botnet (first founded in 2014 [5]), and Mirai Botnet (first founded in 2016 [7]) are the most famous families of IoT Botnets.

BASHLITE (also known as Gafgyt) is a botnet that launches DDoS attacks. It not only exploits upcoming IoT vulnerabilities, but also employs botnet strategies like using Tor to cloak its activities. The most common method to infect recently acquired devices is to leverage Metasploit modules. It targets devices like DVR(s).

Mirai botnet is malware that infects networked devices like security cameras, and smart home devices, converting the network into remotely controlled bots. Mirai botnets are used by hackers to target systems in mass DDoS attacks. Fig. 1 demonstrates cyber-attack on various industries in the first half of 2022. Among them manufacturing, energy and healthcare are the most affected industries.

In this study, we will devise methods to detect Bashlite and Mirai Botnet attacks using machine learning methods.

III. MACHINE LEARNING

If a picture of a Lion or Tiger is shown, will you be able to classify them? Yes, because since childhood, we are taught by our elders how a tiger or lion looks and we can visualize after feeding some data that is machine learning. Machine Learning is a process by which a system improves performance from experience, in other words, the science of machine learning has enabled computers to learn without explicit programming. Tom Mitchell proposed that machine learning is the study of algorithms that:

- improve their performance P
- at some tasks T
- with experience E

There are seven steps involved in implementing machine learning models

1) Data is collected. The accuracy of our model depends on the amount and quality of data. This stage generally results in a representation of data that will be used for training.
2) Data is pre-processed, cleaned, randomized to avoid ambiguity, and then prepared for training.
3) Dataset was split into testing and training set, using different distributions/approaches. One of the most used distributions is K-fold cross-validation.
4) Choosing a model. Based on the methods and way of learning, machine learning is categorized

into Supervised, Unsupervised and Reinforcement Learning.

5) Train the model. This selected model is trained. This step aims to produce a model that can make predictions correctly.
6) Assessing the model utilizing metrics to assess the model's performance. This model is validated using previously unknown data.
7) Parameter tuning refers to hyperparameter tuning. Hyperparameters cannot be learned during the training process. These can be fixed before the actual training process begins. They are used to control the learning rate. They play a crucial role in the fitting of machine learning algorithms. Two well-known strategies for hyperparameter tuning are GridSeachCV and RandomizedSearchCV.

Machine learning algorithms have demonstrated their efficiency and effectiveness in numerous use cases and applications. [9] P. P. Shinde, Pramila P acknowledged the use of machine learning for classification, predicting denial of service attack, etc. And they have been successfully implemented. [10] C. Sinclair, L. Pierce and S. Matzner used genetic algorithm and decision tree for the production of rules to detect complex network intrusions to maximize the utility of the system.

IV. IoT And ML Integration

IoT and machine learning (ML) are complementary technologies that can be integrated to provide significant benefits in various applications. IoT devices generate an enormous volume of data that is leveraged to improve decision-making and enhance user experiences. However, analyzing this data can be challenging without the use of advanced analytical techniques such as ML.

Integrating IoT and ML involves using machine learning
algorithms. This can help to identify patterns, predict future events, and optimize processes in real-time. For example, in the industrial sector, IoT sensors can collect data from machinery, and ML algorithms can be used to analyze this data to detect anomalies, predict equipment failures, and optimize maintenance schedules.

To integrate IoT and ML, there are several steps that need to
be taken, including:

- Data collection: IoT devices need to be set up to collect and transmit relevant data to a centralized database or cloud platform.
- Data preprocessing: The collected data needs to be cleaned, filtered, and transformed into a format suitable for ML analysis.
- Model selection: The appropriate ML model needs to be selected based on the type of data, the problem being addressed, and the desired outcomes.
- Training: The selected ML model needs to be trained on the preprocessed data to learn patterns and make predictions.
- Deployment: The trained ML model needs to be deployed in the IoT system to analyze real-time data and provide insights.
- Continuous improvement: The ML model needs to be continuously monitored and updated to improve its accuracy and effectiveness.

V. Proposed Approach

In this section, we will illustrate the dataset used, data preprocessing, and an experimental scenario.

A. Experimental Dataset (N-BaIoT)

The absence of publicly available botnet datasets, especially for IoT is addressed with the N-BaIoT dataset [11]. It implies actual traffic data acquired from nine commercial IoT devices that were infected by Mirai and Bashlite attacks and consists of 115 features. The malicious traffic includes several types of bait-and-switch attacks. Each network flow file in the N-BaIot dataset contains a range of features, including protocol type, source and destination IP addresses and ports, payload size and packet count, and flow duration. The dataset is labeled with three different classes of traffic: benign, attack, and background.

Table I demonstrates the name of the devices used to generate this dataset.

TABLE I. DEVICES IN THE N-BAIOT DATASET

S.No.	Device Name
1	Danmini Doorbell
2	Ecobee Thermostat
3	Ennio Doorbell
4	Philips B120N10 Baby Monitor
5	Provision PT 737E Security Camera
6	Provision PT 838 Security Camera
7	Samsung SNH 1011 N Webcam
8	SimpleHome XCS7 1002 WHT Security Camera
9	SimpleHome XCS7 1003 WHT Security Camera

The dataset was used in previous studies to develop and evaluate intrusion detection systems for IoT devices and thus, is already pre-processed and no further cleaning is exercised on the data [12][13]. However, we have taken a fraction of some attack types of the dataset to make it balanced. This is discussed by Dr. D. Ramyachitra [14] since there was an unequal distribution of classes in the dataset.

B. Data Pre-processing

The objective of pre-processing is to formulate the data into a form that will help in boosting the efficiency of machine learning models. We have used the Standardization of data that converts the structure of the dataset into one common format of data. It's a data pre-processing technique for machine learning used to scale data with a standard deviation of 1 and a mean of 0 for the data. This helps to remove any bias in the data and ensures that all features have the same scale. Standardization is applied column-wise in the dataset. This can be achieved using the subsequent equations:

$$x_{standardized} = \frac{x - \mu}{\sigma}$$

Where, $\mu = \frac{1}{N}\sum_{i=1}^{N} x_i$,

$$\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - \mu)^2}$$

where:

- $x_{\text{standardized}}$ is the standardized value.
- x is the original value.
- μ is the mean value of the column.
- σ is the standard deviation value of the column.

C. Experiment setup

Machine learning models' performance is significantly influenced by hardware. We used the same hardware throughout the experiment. The following configurations were used to conduct the experiment:

1) Hardware - Intel® Core™ i5-8265U CPU @ 1.8 GHz, 8 GB RAM, NVIDIA GeForce MX230 GPU @ 1.5 GHz, 2GB Memory running on Windows 10
2) Software – Python 3.7

We have used some python libraries to assist us in our implementation:

Keras: It is an API developed by Google using python language for implementing neural networks.
Numpy: NumPy is a library for Python language that can be used for complex mathematical operations. NumPy provides support for working with large-sized multi-dimensional vectors and arrays.
Pandas: A python library that imparts fast, flexible, and expressive data structures designed to work with classified data not only with ease but also intuitively. It is built on top of the NumPy package.
Sklearn: Scikit-learn provides a range of supervised and unsupervised learning algorithms. David Cournapeau started developing sklearn as a Google Summer of Code (GSoC) project in 2007.
Tensorflow: TensorFlow is an open-source software library for high-performance numerical computation. It compiles numerous machine learning models, deep learning models and various algorithms. It allows developers to create dataflow graphs and can be deployed on most devices.

VI. EXPERIMENT RESULT

We have selected 2 devices, namely Provision PT 737E Security Camera and Ecobee Thermostat.
For botnet attack detection, we have used widely known and used machine learning algorithms. We have employed six Machine Learning algorithms (Sequential Model, Decision Tree classification, Support vector machine (SVM), K-Nearest Neighbor (KNN), Gaussian Naïve Bayes (GNB) and Random Forest. M.Alshamkhany [15] applied four classifiers to the Bot-IoT dataset, namely Naïve Bayes, K-Nearest Neighbor, Support Vector Machine, and Decision Trees.

N-BaIoT Dataset contains 10 different protocols that are utilized for botnet attacks. Thus, we have chosen multiclass classification machine learning algorithms. They can classify each of the protocols used for Mirai and Bashlite attacks.
We have unbiasedly divided the dataset into 75% training and 25% testing data using the Sklearn library. The selected ML models are trained via the training set and tested against the testing set. Table II represent the number of instances of different attack type we have used in our study.

TABLE II. NUMBER OF INSTANCES USED IN THIS STUDY

Type of attack	Provision PT 737E Security Camera	Ecobee Thermostat
Benign	31077	13113
Gafgyt Combo	30690	13253
Gafgyt Junk	21629	15156
Gafgyt Scan	20508	13747
Gafgyt TCP	31353	14253
Gafgyt UDP	31203	15719
Mirai Ack	30277	16993
Mirai Scan	29034	17277
Mirai Syn	32873	17521
Mirai UDP	31250	15148
Mirai Udpplain	28340	13105

We have designed these models using Sci-kit learn and Keras. Table III characterizes the design of our models. All the parameters used while training the model are listed in it.

TABLE III. DESCRIPTION OF OUR MODELS

Model	Description
Sequential Model	Number of Layers: 5 Kernel Initializer: Normal Activation: Softmax Early Stopping: Difference in val loss is 10^{-3}
Decision Tree	Criterion: Gini Impurity Splitter: Best
K-nearest Neighbour	Number of Neighbour:7 Weight: Uniform
Support Vector Machine	Kernel: Linear C(Regularization Parameter): 30
Gaussian Naïve Bayes	Portion of the largest variance of all features: 10^{-9}
Random Forest	Number of trees:50 Criterion: Gini Impurity

To measure the performance of our model we use the following metrics:

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

$$\text{Precision} = \frac{TP}{FP+TP}$$

$$\text{Recall} = \frac{TP}{FN+TP}$$

$$\text{F1-score} = 2 \times \frac{Precision \times Recall}{Precision+Rec}$$

where:

- True positives (TP) are situations that the model accurately recognized as positive.
- False positives (FP) are situations when the model wrongly identified them as positive when they were actually negative.
- True negatives (TN) are situations that the model accurately recognized as negative.
- False negatives (FN) are occasions when the model wrongly categorized them as negative when they were actually positive.
- Accuracy is defined as the ratio of properly identified instances among all cases.
- Precision is the fraction of real positive instances among all cases projected to be positive.
- Recall is the percentage of true positive instances accurately recognized by the model out of all real positive cases.
- The F1-score is a statistic that combines precision and recall to create a single score that describes a binary classification model's performance.

TABLE IV. CLASSIFICATION REPORT OF PROVISION PT 737E SECURITY CAMERA.

ML Model	Metrics	Benign	Combo	Junk	Bashlite Scan	TCP	UDP	UDP	ACK	Mirai Scan	SYN	UDP plain	Accuracy
Sequential	Precision	1	0.97	1	0.5	0	1	0.98	1	1	0.98	1	
	Recall	1	0.98	1	1	0	0.98	0.97	0.99	1	1	1	0.901
	F1-score	1	0.98	1	0.67	0	0.99	0.97	1	1	0.99	1	
KNN	Precision	1	1	0.99	1	1	1	1	0.98	1	1	1	
	Recall	1	0.99	1	1	1	1	0.98	0.99	1	1	1	0.996
	F1-score	1	1	1	1	1	1	0.99	0.99	1	1	1	
SVM	Precision	1	0.98	0.84	1	0.5	0	1	1	1	1	1	
	Recall	1	0.81	0.98	1	1	0	1	1	1	1	1	0.888
	F1-score	1	0.89	0.91	1	0.66	0	1	1	1	1	1	
Random Forest	Precision	1	1	1	1	1	0.95	1	1	1	1	1	
	Recall	1	1	1	1	0.94	1	1	1	1	1	1	0.994
	F1-score	1	1	1	1	0.97	0.97	1	1	1	1	1	
Gaussian Naïve Bayes	Precision	1	0.51	0.7	0.68	0.5	0	1	0.82	1	1	1	
	Recall	0.56	0.97	0.07	1	1	0	0.97	1	1	1	0.81	0.758
	F1-score	0.71	0.67	0.13	0.81	0.66	0	0.98	0.9	1	1	0.89	
Decision Tree	Precision	1	1	1	1	1	0.95	1	1	1	1	1	
	Recall	1	1	1	1	0.94	1	1	1	1	1	1	0.994
	F1-score	1	1	1	1	0.97	0.97	1	1	1	1	1	

Table IV shows the classification report with metrics for Provision PT 737E Security Camera.

Table V shows the classification report with metrics for Ecobee Thermostat.

The dataset consists of 10 attacks and Benign. Our model considers them as multiple classes. Each of the 6 machine learning models produced diverse results. [13] C. OKUR and M. DENER were able to get nearly the same accuracy, but they used binary classification whereas our model can classify attack type into one of the 11 types with similar accuracy.

TABLE V. CLASSIFICATION REPORT OF ECOBEE THERMOSTAT

ML Model	Metrics	Benign	Combo	Junk	Bashlite Scan	TCP	UDP	UDP	ACK	Mirai Scan	SYN	UDP plain	Accuracy
Sequential	Precision	1	1	1	0.48	0	1	0.89	1	1	1	1	
	Recall	1	0.89	1	1	0	1	1	1	1	1	1	0.893
	F1-score	1	0.94	1	0.65	0	1	0.94	1	1	1	1	
KNN	Precision	1	1	0.99	1	1	1	0.99	0.98	1	1	1	
	Recall	1	0.99	1	1	1	1	0.98	0.99	1	1	1	0.995
	F1-score	1	0.99	0.99	1	1	1	0.98	0.98	1	1	1	
SVM	Precision	1	0.98	0.89	1	0.5	0.53	1	1	1	1	1	
	Recall	1	0.86	0.98	1	0	1	1	1	1	1	1	0.901
	F1-score	1	0.92	0.93	1	0	1	1	1	1	1	1	
Random Forest	Precision	1	1	1	1	1	0.98	1	1	1	1	1	
	Recall	1	1	1	1	0.98	1	1	1	1	1	1	0.998
	F1-score	1	1	1	1	0.99	0.99	1	1	1	1	1	
Gaussian Naïve Bayes	Precision	1	1	0.56	0.99	0	0	0.91	0.99	0.36	1	0.91	
	Recall	1	0.12	1	1	0	0	0.99	0.95	1	1	0.99	0.737
	F1-score	1	0.22	0.71	0.99	0	0	0.95	0.97	0.53	1	0.98	
Decision Tree	Precision	1	1	1	1	1	0.98	1	1	1	1	1	
	Recall	1	1	1	1	0.98	1	1	1	1	1	1	0.998
	F1-score	1	1	1	1	0.99	0.99	1	1	1	1	1	

VII. Result Discussion

It can be seen in Table IV and Table V, a comparison of six machine learning methods: Sequential, Support Vector Machine (SVM), K-Nearest Neighbor (KNN), decision tree, random forest, and Gaussian Naive Bayes (GNB). Decision tree and Random forest have almost same results. KNN, Decision tree and random forest have achieved an accuracy of approx. 99%, with a striking high F1-score.

SVM shows poor performance in detecting Bashlite TCP and UDP attacks.

Sequential model achieved an accuracy of 90%, but its F1-score dropped down to 0.67 in Bashlite scan and it is unable to predict Bashlite TCP since it had a 0 F1-score.

GNB has achieved an accuracy of 75% approx., which is the worst among all the implemented models, it was able to predict Mirai scan and Mirai syn with a high F1 score but was not able to predict Bashlite UDP and TCP.

Moreover, Mirai attack was predicted correctly by almost all the models with high accuracy, but only KNN, Decision Tree and Random Forest were able to predict Bashlite attack with high accuracy.

VIII. Conclusion and Future Works

The growth rate of the IoT industry is difficult to quantify precisely because it encompasses a wide range of technologies, devices, and applications across various industries. However, according to market research projections, between 2021 and 2026, the global IoT market is expected to grow by around 25% compound annual growth rate (CAGR). This growth is being driven by the increasing adoption of connected devices and the rising demand for automation and data analytics in various industries. The IoT market is expected to reach a value of $1.5 trillion by 2026, with significant growth expected in areas such as smart homes, industrial automation, healthcare, and transportation. Decision Tree, K nearest neighbor and random forest showed extremely high accuracy in detecting Mirai and Bashlite Botnet attacks. But, IoT attacks are becoming more complex with the advancement in technology and knowledge about connected systems. Cybercriminals have now become more efficient and can attack different components of an IoT network. Thus, we should devise new methods and techniques to counter IoT cyberattacks.

Future work of this paper would be to

1) Extend our models by using the complete dataset and at the same time ensure that the data remain balanced to avoid overfitting.
2) Hyperparameter tune our models to achieve higher F1 scores in predicting Bashlite attacks.
3) Use new machine learning techniques that provide advanced and accurate models for detecting botnet attacks.

IX. References

[1] J. Ploennigs, J. Cohn and A. Stanford-Clark, "The Future of IoT," in *IEEE Internet of Things Magazine*, vol. 1, no. 1, pp. 28-33, SEPTEMBER 2018, doi: 10.1109/IOTM.2018.1700021.

[2] Nozomi Networks "OT/IoT Security Report", (2022) https://www.cisa.gov/uscert/sites/default/files/ICSJWG-Archive/QNL_SEP_22/Nozomi-Networks-OT-IoT-Security-Report-ES-2022-1H_508c.pdf.

[3] Y. Jin, "Towards Hardware-Assisted Security for IoT Systems," 2019 *IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, 2019, pp. 632-637, doi: 10.1109/ISVLSI.2019.00118.

[4] Leonard, Justin & Xu, Shouhuai & Sandhu, Ravi. (2009). "A Framework for Understanding Botnets," Proceedings - *International Conference on Availability, Reliability and Security, ARES 2009*. 917-922, doi: 10.1109/ARES.2009.65.

[5] K. Angrishi, "Turning internet of things (iot) into internet of vulnerabilities (iov): IoT botnets," 2017., doi: 10.48550/arXiv.1702.03681.

[6] C. Kolias, G. Kambourakis, A. Stavrou and J. Voas, "DDoS in the IoT: Mirai and Other Botnets," in Computer, vol. 50, no. 7, pp. 80-84, 2017, doi: 10.1109/MC.2017.201.

[7] M. Antonakakis, T. April, M. Bailey, M. Bernhard, E. Bursztein, J. Cochran, Z. Durumeric, J. A. Halderman, L. Invernizzi, M. Kallitsis, D. Kumar, C. Lever, Z. Ma, J. Mason, D. Menscher, C. Seaman, N. Sullivan, K. Thomas, and Y. Zhou, "Understanding the mirai botnet," in Proc. of USENIX Security Symposium, 2017.

[8] Y. M. P. Pa, S. Suzuki, K. Yoshioka, T. Matsumoto, T. Kasama, and C. Rossow, "Iotpot: Analysing the rise of iot compromises," in Proc. Of USENIX Workshop on Offensive Technologies, 2015.

[9] P. P. Shinde and S. Shah, "A Review of Machine Learning and Deep Learning Applications," *2018 Fourth International Conference on Computing Communication Control and Automation (ICCUBEA),* 2018, pp. 1-6, doi: 10.1109/ICCUBEA.2018.8697857.

[10] C. Sinclair, L. Pierce and S. Matzner, "An application of machine learning to network intrusion detection," *Proceedings 15th Annual Computer Security Applications Conference (ACSAC'99),* 1999, pp. 371-377, doi: 10.1109/CSAC.1999.816048.

[11] Yair Meidan, Michael Bohadana, Yael Mathov and other, N-Baiot dataset, http://archive.ics.uci.edu/ml/datasets/detection_of_IoT_botnet_attacks_N_BaIoT

[12] Y. Meidan et al., "N-BaIoT—Network-Based Detection of IoT Botnet Attacks Using Deep Autoencoders," in *IEEE Pervasive Computing*, vol. 17, no. 3, pp. 12-22, Jul.-Sep. 2018, doi: 10.1109/MPRV.2018.03367731.

[13] C. OKUR and M. DENER, "Detecting IoT Botnet Attacks Using Machine Learning Methods," *2020 International Conference on Information Security and Cryptology (ISCTURKEY),* 2020, pp. 31-37, doi: 10.1109/ISCTURKEY51113.2020.9307994.

[14] Dr. D. Ramyachitra, P. Manikandan, IMBALANCED DATASET CLASSIFICATION AND SOLUTIONS: A REVIEW, International Journal of Computing and Business Research (IJCBR) ISSN (Online) : 2229-6166

[15] M. Alshamkhany, W. Alshamkhany, M. Mansour, M. Khan, S. Dhou and F. Aloul, "Botnet Attack Detection using Machine Learning," *2020 14th International Conference on Innovations Information Technology (IIT),* 2020, pp. 203-208, doi: 10.1109/IIT50501.2020.9299061.