International Journal of Advanced Technology and Engineering Exploration, Vol 10(104)
ISSN (Print): 2394-5443 ISSN (Online): 2394-7454
http://dx.doi.org/10.19101/IJATEE.2022.10100538

Research Article

Investigating the performance vulnerability of AODV protocol of IoT network under SYN-flood attack

Abhijit Biswas[1], Rabinder Kumar Prasad[2], Abhijit Boruah[2] and Sudipta Majumder[2*]
Department of Computer Science and Engineering, Assam University, Silchar, Assam, India-788011[1]
Department of Computer Science and Engineering, Dibrugarh University, Dibrugarh, Assam, India-786004[2]

Received: 09-December-2022; Revised: 17-July-2023; Accepted: 20-July-2023

Abstract
Wireless technology is utilized by various Internet of Things (IoT) devices, such as smartphones, drones, and cameras, to establish multiple inter-device connections simultaneously. The flexibility of wireless networks allows users to create IoT-based ad-hoc networks on demand. This connectivity enables linking hundreds to thousands of people, leading to significant enhancements in productivity and profitability. However, the flexibility of the IoT network also introduces a range of threats that require attention. The primary research methodology employed in this study was simulation to investigate and identify existing vulnerabilities or loopholes in the transport layer of IoT networks that use the ad-hoc on-demand distance vector (AODV) routing protocol. The simulation software utilized was NetSim 12.02, which is based on the Institute of Electrical and Electronics Engineers (IEEE) 802.15.4 standard. We conducted simulations of synchronization (SYN) flood attacks on IoT networks by exploiting vulnerabilities in the underlying protocol. The tests and data collection for this study were conducted using the simulation software. The research demonstrates that the transport layer of the IoT network, when utilizing the AODV protocol, can be compromised to launch a SYN-flood attack, resulting in severe performance degradation of the system. The results demonstrate that SYN-flood attack nodes result in a decrease in throughput ranging from 9.1% to 17.79%, an increase in delay ranging from 1.08% to 3.26%, and an increase in network jitter ranging from 15.4% to 22.88%. These findings can assist IoT network administrators in better planning and implementing preventive measures against SYN flood attacks on IoT networks that utilize the AODV protocol.

Keywords
IoT network, AODV, SYN-flood, Intrusion, Half-open connections, Throughput, Delay, Jitter, Transmission overhead, Idle state battery life.

1.Introduction
Technology is an absolute necessity for all people in today's world, as evidenced by the growing reliance on it in practically every area of our lives. With the fast-developing internet of things (IoT) applications, the world is undergoing significant changes [1]. In recent years, IoT has become a remarkable phenomenon. It connects physical and virtual items that have been embedded with sensors, software, and other technologies [2]. The vision of IoT is to utilize the internet to enable communication and information sharing among compatible devices worldwide. Furthermore, the IoT network comprises a collection of connected gadgets, excluding commonly used computing devices such as mobiles and desktop computers.

IoT has permeated various industries, from healthcare to large corporations [3]. It transforms the traditional computing devices in our environment into intelligent objects, thereby influencing human living standards. For instance, IoT devices can track and record vital measures like oxygen saturation (SPO_2), sleep patterns, etc., and promptly report them in case of emergencies to increase the chances of survival for those affected [4].

In the event of an accident, appropriate decisions are made to send automated and swift emergency messages to the nearest police and medical services. The IoT network greatly assists in the manufacturing and assembly processes of manufacturing industries. It also aids in the packaging and logistics of industries [5]. Routing plays a crucial role in decision-making for IoT networks. In an IoT network, "routing" refers to the act of determining a

*Author for correspondence

route for data to flow from one device to another inside the network. This route will allow the data to reach its destination. Because IoT networks typically consist of a large number of devices, which may be located in a variety of locations, it is vital to implement effective routing to guarantee that data is transferred promptly and accurately. IoT routing protocols will often take into account several criteria when choosing the most efficient route for data to follow. These criteria include the network topology, bandwidth availability, power consumption of the devices, and network congestion [6]. Different forms of IoT applications require different kinds of routing protocols since each kind of protocol has its own unique set of features [7].

Depending on what the network needs and how it works, IoT networks use different routing algorithms. Routing algorithms that are often used in IoT networks are:

- Routing protocol for low-power and lossy networks (RPL), commonly known as RPL, is a distance-vector routing system made for networks with low power and poor connectivity. It is made to work with devices that have limited memory, processing power, and power backup [8].
- Dynamic source routing (DSR) is another reactive routing system that only finds a route between two nodes when it is needed. It sends packets to the next node using source routing, where the sender gives the full path to the next node [9].
- Open shortest path first (OSPF), commonly known as OSPF, is a link-state routing system used in internet protocol (IP) networks. It is used in IoT networks where fast connection and a lot of nodes are needed [10].
- Border gateway protocol (BGP), commonly known as BGP, is a gateway system that is used to link different networks. It is used in IoT networks to connect different devices that work on their own [11].
- Ad-hoc on-demand distance vector (AODV) is a reactive routing system that is widely used in mobile ad-hoc networks (MANET), IoT networks and vehicular ad-hoc networks (VANET). It sets up a route between two nodes only when it's needed and keeps the routes as long as the two nodes are communicating with each other [12].

However, IoT-based applications are at risk due to their enormous growth in popularity and the rapid expansion of technologies. Researchers must thoroughly examine the weaknesses, security, and threats to make the envisioned IoT network possible. Vulnerabilities can be defined as flaws in the architecture or functionality of a framework that allows intruders to gain unauthorized access and carry out malicious activities that degrade system performance [13]. Intruders/attackers can infiltrate networks by exploiting known flaws in IoT devices. One such flaw is exploited in domain name system (DNS) rebinding attacks, which enable the processing and exfiltration of data from internal networks [14]. Vulnerabilities can be identified in various parts of IoT systems. Researchers have made significant efforts to discover vulnerabilities in the services and different layers of the five-layer architecture of IoT networks [15]. However, synchronization (SYN) flood attacks remain prominent in various types of networks. This attack involves bombarding the intended server with a large number of transmission control protocol (TCP) SYN packets, which are used to establish new connections [16].

The challenge in [17] was to propose a method for dynamically balancing the processing load across dispersed controllers in software-defined networking (SDN)-based fifth-generation (5G) networks while addressing the critical security issue of SYN flood attacks. The simulation results must also demonstrate the efficacy of the proposed system in detecting and mitigating SYN flood attacks, which are part of distributed denial of service (DDoS) attacks in 5G networks. Similarly, the task in [18] was to develop an efficient model for detecting and mitigating DDoS attacks induced by TCP SYN flood in SDN environments. Additionally, the objective in [19] was to develop a system capable of detecting SYN flood attacks based on the power generated during connection establishment, but this is a challenging undertaking due to the limitations and vulnerabilities of wireless sensor networks. In [20], the primary challenge is to detect TCP SYN flood attacks in a multitenant cloud environment from the perspective of a cloud service provider. The researchers face the challenge of designing a cross-layer technique that relies on machine learning to detect SYN flood attacks on IoT networks that employ RPL as their routing algorithm.

Similarly, IoT networks having AODV routing protocol may be susceptible to SYN flood attacks because it is the protocol that sets up routes between nodes that need to communicate each other. This makes it easier to keep routing tables up to date, especially in big networks where nodes are always joining and leaving. AODV is also good at quickly and efficiently finding routes. It does this by sending

route requests (RREQ) to its neighbouring nodes and then sending them to other nodes until a route is found. Because of this, it works well in networks with dynamic layouts, where the way the network is set up can change often [21].

In IoT networks that use the AODV routing system, it can be hard to find an SYN flood attack. The different ways that connected devices act and the different technologies they use make it hard to understand how an attack on an IoT network works. Along with this, there is no way to make the detection system work for all IoT framework layers because their network parameters are not the same. Keeping these challenges in mind, the proposed study focuses on multiple SYN flood attacks at the transport layers of an IoT network and tries to collect performance parameters for the network.

In this article, we examine the flaws in the IoT network's architecture while using the AODV routing protocol and provide our findings. Our study aims to find any existing vulnerabilities in the transport layer to create SYN-flood attacks and study the effects of those vulnerabilities on the network's performance when applications are using AODV as their routing protocol. The expected outcome of our research is to find existing vulnerabilities at the transport layer in the IoT network before they are exploited by malicious individuals or entities. Also, we will be able to model how those vulnerabilities can affect the network performance metrics like throughput, delay, jitter, energy consumption by IoT devices and overhead cost, etc. These findings can assist IoT network administrators in better planning and implementing preventive measures against SYN flood attacks on IoT networks that utilize the AODV protocol.

A brief introduction is provided in this section. The remaining research article is divided into several sections. Section 2 comprises a literature review where the related work conducted by other researchers is presented. These studies shed light on various vulnerabilities of the network. The materials and methodology are described in section 3, while section 4 reports the experimental results. Section 5 presents a discussion of the results. Finally, in section 6, conclusion is provided, along with the future work.

2.Literature review

Vulnerabilities of a network are loopholes in the design of the framework of the network. Intruders use the loopholes to execute malicious activity. They can access unapproved data or information [13]. Vulnerabilities may exist in many places like operating systems, programming software, control software, etc. Vulnerabilities exist because of human factors and programming complexity [15]. For reducing the programming complexity of the functioning of the IoT network, its architecture is divided into five layers [22–25] as in *Figure 1*.

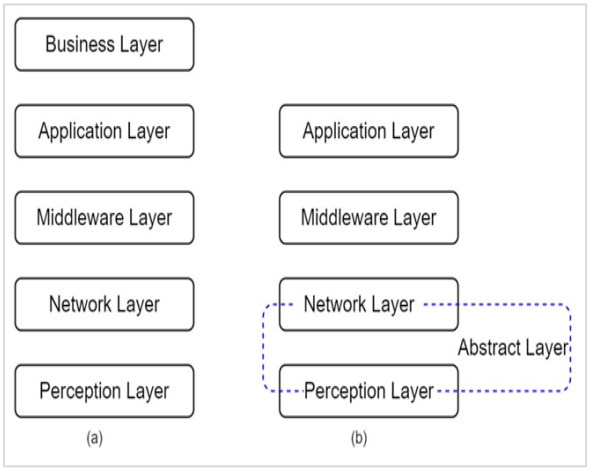

Figure 1 Versions of 5-layer architecture of IoT network

The section of literature review has subsections elaborating on performance vulnerabilities of various layers of IoT architecture, various routing protocols of IoT networks and a final review analysis.

2.1Various performance vulnerabilities of various layers of IoT architecture as mentioned by various authors are discussed, are as follows:

2.1.1Perception layer

Vulnerabilities can cause different types of attacks on the perception layer of the IoT network. Spoofing attacks, battery drainage, eavesdropping, sybil threat, malicious data injection, hardware failure, node capturing, tag cloning, and side-channel attacks are examples of attacks that exploit the perception layer's vulnerabilities [26].

Spoofing attacks are the most common attack at the perception layer of an IoT network. This attack in the perception layer of IoT networks involves sending fake or misleading data to the network while pretending to be a real sensor or device. Attackers can fake data in many ways, such as by changing a device's media access control (MAC) address, using a man-in-the-middle (MITM) attack to intercept and change data, or making fake reports from sensors. Spoofing attempts can cause a lot of trouble for IoT

networks [27]. By sharing fake information, attackers can trick the network into doing the wrong thing or making the wrong choice. For example, a faked temperature monitor in a smart home system could cause the heating or cooling to be wrong, making the people inside uncomfortable. In industrial IoT systems, fake sensor data could cause the wrong control of the process, which could damage the product or the equipment.

Spoofing attacks are easy to carry out and can be used to get around standard security systems like firewalls and intrusion detection systems. Attackers can create fake data with tools and methods that are easy to get. This makes it hard for network managers to find and stop these attacks. Researchers used the received signal strength and the number of connected neighbours, to propose a method to find spoofing attacks in the IoT network [27–29]. In an inter-cluster network, received signal strength is used to find the spoofing attacks, pinpoint its location, and stop it. But the received signal strength isn't good at stopping intra-cluster spoofing attacks, so the number of connected neighbours was used to find, spot, and stop intra-cluster spoofing attacks. The proposed model used network simulator 2 (ns-2) to compare how well the method works when spoofing attacks are present and when they are not. Because of this, the suggested model makes it easier to find and stop spoofing. The method has the advantage that in the presence of received signal strength and received signal strength, spoofing attacks are greatly reduced and network performance is enhanced. However, without a spoofing attack, the network's performance degrades continuously.

Another popular attack at the perception layer is a jamming attack. A jamming attack is a kind of wireless attack that messes up the radio frequency (RF) messages that devices use to communicate to the network. The attacker uses a jamming device to fill the RF spectrum with noise or interference signals that can mess with or drown out the signals that IoT devices are trying to send. This makes it hard for the network to get data from the affected devices and handle it, which can cause delays and even data loss [30, 31]. A deep learning technique-based anomaly-based IDS system for IoT networks for detecting jamming attacks was proposed [32]. Specifically, a deep neural network model with filter-based feature selection in which highly correlated features are eliminated has been presented. Additionally, the model is modified using several parameters and hyperparameters. For this purpose, the UNSW-NB15 dataset containing jamming attack classes is utilised. The proposed model was accurate to the extent of 84%. The advantage of this approach is that by using generative adversarial networks to generate synthetic data of minority attacks to resolve class imbalance issues in the dataset, 91% accuracy can be attained with a balanced class dataset.

There are some limitations to the jamming attacks, such as:

- Needs to be close: Jamming devices must be close to the IoT devices for them to work. This means that the attacker has to be in the same place as the network or object they want to attack.
- Jamming devices send out a strong RF signal that can be picked up by the network or other security systems. This can let security staff know that an attack is happening so they can take steps to lessen its effects.
- Can be expensive: A device that jams signals can be expensive, and an attack may need more than one device to stop the network from working. This can make it harder for low-level attackers to do a jamming attack and reduce the number of people who can do it.

Overall, jamming attacks are a major threat to the perception layer of IoT networks, so organisations need to put in place security measures to avoid and detect them. Some of these steps are using anti-jamming technologies, putting in place intrusion detection and prevention systems, and putting in place security protocols to make sure that devices accessing the network are legitimate [31–34].

2.1.2Abstraction layer

Vulnerabilities are not just limited to the perception layer but also other layers. There is generally no or too little physical protection for IoT systems in an untrusted environment. An intruder can steal or damage sensors leading to illegal access or malicious change in data. “Address resolution protocol (ARP) priority” vulnerability is used to listen to traffic between an intelligent device and a gateway. This type of attack happens at the abstraction layer [26]. A MITM attack is a type of cyberattack in which an offender intercepts the communication between two devices or nodes on an IoT network at abstraction layer and changes or manipulates the communication [35]. A MITM attack can be very bad because it can make personal data and services less private, less trustworthy, and less available. MITM attacks have a high success rate, can get to private information, are hard to find, have a limited range, and can be found by security systems. Organisations need to put in

place security measures, such as encryption, access control, and intrusion detection and prevention systems, to stop and find these kinds of threats [34].
Any node in a VANET network can function as a router for the other nodes, and a malicious node connected to the network can inject spoofed routing tables into the other nodes, thereby influencing the network's operation. To circumvent this problem, a secure AODV routing protocol for DoS attack detection is devised [36]. The proposed method is a modified variant of the original AODV routing protocol that incorporates enhancements to the RREQ packet and route reply (RREP) packet protocols. Cryptography-based encryption and decryption are included to verify the source and destination nodes for added security. The proposed method was demonstrated using various network parameters, including packet loss, end-to-end latency, packet delivery ratio (PDR), and routing request overhead, with the help of NS-2.33 simulator. The advantage of the proposed method is that it outperforms the existing AODV routing protocol and improves network performance during black hole attacks at the abstraction layer [36, 37].

There are some limits to the DoS attacks on the abstraction layer, such as [37, 38]:

- A lot of requests are needed: The success of a DoS attack rests on how much traffic is made. To overwhelm the device or service being attacked, it may take a lot of requests or messages.
- Can be found and stopped: Firewalls and intrusion detection and prevention systems, which are used to protect networks, can find and stop DoS attacks by filtering or blocking data from suspicious sources.
- May not work on all devices: DoS attacks might not work on devices or services that have built-in defences against these kinds of attacks, like load balancers or fallback mechanisms.

DoS attacks at the abstraction layer of an IoT network are a major threat, and organizations must put in place security measures to prevent and identify them. These steps can include putting in place access controls, using intrusion detection and prevention systems, and putting in place security rules to make sure that devices that connect to the network are legitimate. Spoofing attacks, which include authentication attacks, are considered to violate of the privacy principle [39]. These attacks involve impersonating nodes. The attack is an active DoS attack if a program or a system is forcibly rejected from accessing resources. Attacks are passive attacks where one application stops another from running on the device. Tag cloning, sleep deprivation, traffic analysis attack, etc., also occur at the abstraction layer [40].

2.1.3Network layer

The sinkhole attack in an IoT environment can devastate and compromise the entire communication system. When normal nodes initiate the process of sending their packets through sinkhole nodes, the sinkhole attacker nodes begin to disrupt the network's traffic flow. In the existence of sinkhole attack nodes, the destination (e.g., sink node i.e., gateway/base station) either does not receive the required information or receives incomplete or altered information. The network performance is reduced as a consequence, and the communication's efficiency and reliability suffer. In the presence of such an attack, the throughput decreases, the end-to-end latency grows, and the PDR falls. In addition, it may negatively impact other network performance parameters. Consequently, it is of the utmost importance to provide an efficient and competent scheme for mitigating this attack in IoT environments. In paper [41], a proposed intrusion detection scheme aims to safeguard the IoT environment from sinkhole attacks. Through the exchange of messages, the resource-rich edge nodes (edge servers) detect various forms of sinkhole attacker nodes in this method. A practical demonstration of the proposed method is also provided by computing the various performance parameters using the well-known NS2 simulator. In addition, a security analysis of the proposed method is conducted to demonstrate its resistance to diverse sinkhole attacks. The proposed method obtains a detection rate of approximately 95.83% and a false positive rate of 1.03%, which are significantly higher than other comparable existing schemes. The proposed method has the benefit of being efficient in terms of computation and communication costs. Eventually, the method will be a good fit for applications that can be utilised in crucial and sensitive operations (such as surveillance, security, and monitoring systems) [41].

Blackhole and wormhole attacks are types of attacks that can occur at the network layer of an IoT network. Blackhole attacks involve intercepting and dropping data packets received by the IoT devices in the network, making them unavailable to other devices [42]. They can cause significant damage to the network, disrupt normal traffic, and prevent devices from communicating with each other, leading to a DoS condition. The blackhole attack is easy to carry

out and can be used to disrupt IoT networks quickly. Wormhole attacks involve setting up a shortcut between two points in the network, creating a tunnel that can be used to divert or manipulate traffic, intercepting the packets, and replaying them at the other end of the tunnel [43, 44]. They can cause significant damage to the network, bypass network security mechanisms, steal sensitive data, and cause a DoS condition. The wormhole attack can be difficult to detect and can bypass many security mechanisms. A self-adaptive framework into networks was introduced and thus, the data transmission process is augmented, to detect and prevent the attacks [44]. Using the proposed self-adaptive framework for the AODV routing protocol network communication protocol, the affected area is measured and rectified in the study. The performance of the network restoration technique is investigated through the usage of simulation. The proposed framework indicates promising overall performance with the aid of using accomplishing excessive flow network rate and minimum delay [35, 44]. It is found that the accuracy of the optimized simulation-based method is better than conventional methods. The energy consumption of the proposed method with 35 nodes is 7.14% superior to web-oriented architecture (WOA) and FireFly, 5.7% superior to grey wolf optimization, and 10.3% superior to particle swarm optimization.

The vulnerabilities, in this layer, are exploited to carry out attacks like jamming, false-data injection, eavesdropping, unfair access, hello flood, congestion, message queuing telemetry transport (MQTT) exploit, SYQ- flooding, etc. Desynchronization is also an example of attacks possible at the transport layer. In this attack, an attacker can break genuine linkages between two nodes by desynchronizing transmissions between them. This form of attack includes delivering forged communications like false flag messages to all sides of a participating communication channel. They will lose their ability to communicate if they are forced to lose their SYN. Session hijacking occurs when an attacker takes a user's session identifier (ID) and impersonates the legitimate user to seize control of the user's online session [26, 27].

2.1.4Middleware layer

Injection attacks on the middleware layer of IoT networks entail inserting malicious code into the communication that takes place between the apps and the devices. This can result in unauthorised access, the theft of data, or the compromising of the system. In the middleware layer of an IoT network, attackers will inject malicious code, which will often take the form of code written in structured query language (SQL), eXtensible markup language (XML), or java script. This code is then performed by the system, which grants the attacker access to the data, allows the attacker to modify the data, or grants the attacker unauthorised access to the system [45–47].

Injection attacks can result in the unauthorised access of devices or apps, as well as the theft of data and compromise of systems. Attackers have the option of utilising this attack to either take control of the IoT network or as a springboard from which to launch other attacks. Autoencoders have been presented as a novel method for detecting injection attacks [48]. They utilised the sensor data correlation in time and space, which can aid in identifying fabricated data. Moreover, the denoising autoencoders are used to clear the fabricated data. Evaluation of performance demonstrates the effectiveness of the technique in detecting injection attacks. It also surpasses a support vector machine-based method employed for the same purpose. The advantage of this approach is that it has been demonstrated that this method's data cleansing algorithm is very effective at recovering clean data from corrupted (attacked) data. Injection attacks have the potential to be very successful because they can take advantage of vulnerabilities in the middleware layer that are difficult to locate and repair. Injection attacks can be difficult to prevent because they take advantage of weaknesses at the middleware layer, which are difficult to locate and remedy [47, 48].

It is possible to help prevent vulnerabilities that can be exploited by attackers by making use of safe coding practices, constantly updating and patching the middleware software, and updating the program at regular intervals [49]. SQL injection attacks and DDoS attacks can be used by an adversary to take control of secondary nodes and even the principal ones under certain circumstances [50].

Crypto-jacking attacks at the middleware layer of IoT networks involve hackers exploiting vulnerabilities in the middleware software to install cryptocurrency mining software [51]. This allows them to use the computing resources of IoT devices to mine cryptocurrencies without the knowledge or consent of the device owner. The attacks can cause IoT devices to slow down or crash, affecting their performance and usability, consume more power and generate higher energy bills [52]. Crypto-jacking attacks are relatively easy to execute and can be highly profitable for attackers, especially if they can target a

large number of IoT devices. Crypto-jacking attacks rely on vulnerabilities in the middleware software, which can be patched or updated to prevent the attack [53].

2.1.5Application layer

There are various types of security concerns or vulnerabilities at the operation layer and application layer. The security concern raising due to the vulnerabilities being unauthorized access, fake information, MITM, end-to-end encryption attack, integration attack, DoS attacks and illegal intervention attacks. The application layer is in charge of the services provided to clients. This layer addresses each program's own set of security concerns. Various types of attacks and threats that exploit application layer vulnerabilities are data tampering, malware attack, SQL injection, cross-site script, code injection, etc. [27]. The issue of IoT security sensor manipulation in an office setting was addressed [54]. Data from real-world environments is aggregated, and two machine-learning techniques are utilized to detect sensor tampering. Initially, a real-time view of the traffic patterns is used to train an unsupervised machine learning method based on isolation forests for anomaly detection. Subsequently, a novel anomaly detection system using machine learning generates labels based on traffic patterns and employs a decision tree supervised method. The achieved accuracy of the isolation forest is 84% based on the silhouette metric. Furthermore, the supervised machine learning model, evaluated through 10 cross-validations for decision trees, yielded the highest classification accuracy of 91.62% with the lowest false positive rate. Attacks at the application layer are constrained by the fact that they take a certain level of technical expertise and the ability to write and run malicious code, as well as the fact that security measures like firewalls and intrusion detection systems can identify and reduce the impact of such attacks.

Attacks on the application layer can lead to the theft of sensitive information, unauthorized entry to the system, and the application not working as it should. Attacks at the application layer can also compromise the integrity and privacy of the data sent between the devices and the application. To stop these kinds of attacks, it's important to be cautious. Session hijacking happens because of vulnerabilities in the network layer [55, 56].

2.2Explanation of the exploitation of various routing protocols' vulnerabilities in IoT networks by malicious nodes

- A malicious node erroneously asserts that it is the quickest path to the target node and then discards all data packets along the way. As a result, legitimate nodes are unable to communicate with one another, leading to a DoS attack. The attack is called a “black hole” attack. Researchers in [57–60] proposed exponential smoothing, social computing, and federated learning-based approach for detecting black hole attacks.
- Another challenge faced by IoT networks is when two or more hostile nodes establish a tunnel between themselves beyond the regular communication range and then send data packets through this tunnel instead of following the usual routing method. This attack can be used as a springboard for additional attacks, such as packet manipulation or dropping. This attack is called a “wormhole attack”. Researchers in [61] proposed deep learning and cost estimation techniques to detect wormhole attacks in IoT networks.
- In a sybil attack, a malicious node establishes many dummy accounts and acts as though it is multiple other nodes in the network. This gives the attacker complete command over the network and the ability to conduct a wide variety of attacks, such as denial of service and selective forwarding. In [62-64], collaborative edge computing, bloom filter, unclonable function, and time-varying channels-based approaches were proposed for detecting sybil attacks on IoT networks.
- A malicious node can launch a “selective forwarding attack” by only sending specific data packets to other nodes and discarding the rest. This has applications in both obfuscating sensitive information and preventing communication between specific nodes. A “heart-beat” communication hybrid optimization algorithm to demonstrate the possibility of selective forwarding attacks in IoT-based networks was elaborated in [63, 64].

The literature provides an in-depth exploration of the vulnerabilities that can exist in an IoT network. In the research, the weaknesses that can be found in an IoT network are looked at in depth. The literature study shows that the IoT architecture is structured into several layers. Each layer has unique vulnerabilities.

The vulnerabilities are largely in the underlying protocols. These vulnerabilities are used by malicious entities. At the perception layer, for example, attackers develop spoofing attacks, battery depletion, eavesdropping, sybil attacks, and malicious data injection-like attacks. Various researchers have proposed various techniques to deal with such flaws or vulnerabilities. For instance, in [29], a strategy was proposed to detect spoofing attacks at the perception layer of an IoT network. This approach uses received signal strength and the number of connected neighbors to identify the attacks, leading to improved network performance. However, in the absence of an attack, network performance may suffer due to transmission costs. In [32], a novel anomaly-based IDS solution for IoT networks was presented to prevent attacks. This approach utilizes generative adversarial networks to produce synthetic data on minority attacks, addressing class imbalance issues in the dataset. The network layer of an IoT network can be vulnerable to various attacks, such as sinkholes, blackholes, false-data injection, eavesdropping, unfair access, hello floods, congestion, etc., caused by flaws in the underlying protocols. To identify blackhole and wormhole threats in IoT networks, a self-adaptive architecture was proposed for the AODV routing protocol network [44]. A simulation program was used to analyze the performance of the network restoration technique, revealing that the optimized simulation-based method outperforms conventional methods in terms of accuracy. Additionally, the middleware layer of IoT networks can also be targeted by attacks, including data injection and crypto-jacking. In [48], a machine-learning-based approach was suggested to detect bogus data injection attacks in industrial IoT. The model makes use of autoencoders. The model has a high rate of accuracy. IoT sensor manipulation can occur in an office setting, and a mechanism has been devised to check if the IoT sensors are tampered [54]. Our research aims to examine the IoT network framework for any such existing vulnerabilities. We are focusing on the transport layer of the IoT reference model because this layer is responsible for end-to-end communication in IoT networks. This layer provides services like reliability and congestion avoidance. This layer also guarantees that packets will be delivered in the same order that they are sent. If vulnerabilities exist in this layer of the IoT architecture, then intruders or attackers will be able to degrade the performance of the network without raising any alarm. Our objective is to detect and exploit existing vulnerabilities in this layer to carry out SYN flood attacks before they are exploited by intruders. We will also examine the effects of the detected vulnerabilities on the performance of the IoT network. The next section represents the methodology of our investigation for vulnerabilities, associated attacks, and their effect on the IoT network performance. We have presented the necessary steps taken to record performance metrics in the next section.

3.Materials and methods

IoT's main component is wireless sensors, IPv6 over low power personal area network (LoWPAN) gateway, and a router. For simulation purposes, we have been using a network simulator known as Netsim 12.02. The software has all the necessary tools to simulate the IoT environment. The sensors are tiny, power-efficient devices used to collect essential signals. The sensor we used here is wireless. The LoWPAN gateways are devices that create a low-power wireless personal network based on the IP version 6 (IPv6). A router is a device that routes packets received from LoWPAN gateways to the wired node.

We chose simulation as a study method because it allows us to accurately forecast the behavior of all IoT network components. It aids in the investigation of the flaws in the underlying protocols and their impact. These flaws should be pursued, investigated, and eliminated to prevent severe damage to IoT networks from malicious entities or organizations.

In this article, we have demonstrated the creation of an SYN flood attack for an IoT device network using AODV as its routing protocol. The SYN-flood attack is a type of DoS attack that exploits the vulnerability of the TCP/IP protocol. SYN packets are normally used to establish connections between sensors and a wired node. Once the SYN packet and its acknowledgment packet are exchanged between the sensor and the wired node, they become eligible for data transfer. However, the vulnerability arises when a malicious node can misuse the exchange of SYN and acknowledgment packets, leading to a severe attack on the network. A hostile sensor sends an SYN packet to the wired node, resulting in an acknowledgment being sent back from the wired node to the malicious sensor. However, the sensor does not reply with an acknowledgment packet, resulting in a half-open connection. The malicious sensor repeats this process to create numerous half-open connections, exhausting the resources available at the wired node. Additionally, we have measured, analyzed, and evaluated the performance of the SYN-

flood attack on the AODV routing protocol in the IoT network. The flowchart for the performance evaluation of the AODV routing protocol under an SYN-flood attack in the IoT network is depicted in *Figure 2*. The following is the detailed explanation for each step in the flowchart for investigating the performance vulnerability of the AODV protocol in an IoT network under an SYN-flood attack.

This is the beginning of the flowchart, signifying the start of the research procedure.

- Start: This step indicates the start of the design and implementation of an IoT simulation module in Netsim, a simulation software tool used for simulating and analyzing IoT networks. This is a crucial step for the research team to construct a virtual environment of the IoT network and test the performance of the system under various conditions.
- Design & implementation of IoT simulation module in Netsim: This step requires the design and implementation of an IoT simulation module in Netsim, a simulation software tool used for simulating and analyzing IoT networks. This is a crucial step for the research team to construct a test-bed environment of the IoT network and test the performance of the system under various conditions.
- Design & implementation of SYN flood attack for IoT system: In this step, the research team plans and executes an SYN-flood attack against the under-investigation IoT system. This step is essential for analyzing the system's performance under attack and identifying the AODV protocol's vulnerabilities. In this step, the research team constructs simulation scenarios numbered 1 through 5 to test the performance vulnerability of the AODV protocol of the IoT network under SYN-flood attack. These simulation scenarios will aid the research team in simulating various attack scenarios and evaluating numerous performance metrics.
- Implementation of various scenarios for attack: This is a conditional step that occurs if the implementation of attack scenarios for the IoT system is effective. If the attack scenarios fail, the research team may need to revise their approach and make modifications to ensure successful implementation.
- Performance metrics computation: This step involves measuring various performance metrics of the IoT network under the attack scenarios, such as throughput, jitter, delay, half-open connections, transmitted overhead, and energy consumed. These metrics will assist the research team in determining the effect of the SYN-flood attack on the performance of the AODV protocol in the IoT network.

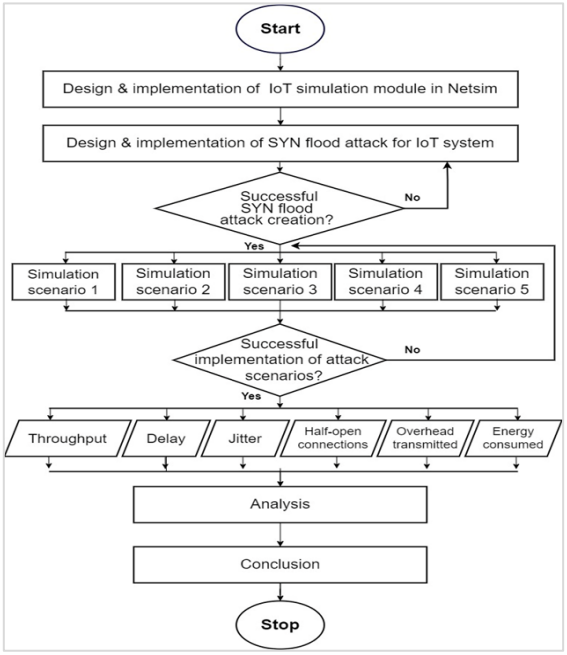

Figure 2 Process flow diagram of our study

- Result analysis: The results of the performance metrics measured in step 5 are analyzed to determine the effect of the SYN-flood attack on the performance of the AODV protocol within the IoT network. The analysis assists the research team in identifying the effect of the vulnerability of the network under attack.
- Conclusion: The research team concludes the performance vulnerability of the AODV protocol of the IoT network under the SYN-flood attack based on its analysis of the results. This step will assist the research team in determining the effect of SYN-flood attacks in the AODV protocol in IoT networks.
- Stop: The end of the flowchart, signifying the conclusion of the research procedure. At this point, the research team has concluded its investigation and has a comprehensive comprehension of the system's behavior during the SYN-flood attack.

The algorithm for creation of SYN flood attack for AODV routing protocol in IoT network is shown below. The novelty of this algorithm lies in its specific adaptation to the AODV routing protocol in IoT networks, where it effectively creates and tracks

malicious sensors that are capable of launching SYN flood attacks.

Algorithms 1: Create SYN flood attack sensor for AODV routing protocol in IoT network.
Input: List of IDs of all the sensors in the IoT network.
Output: A set of malicious sensors capable of carrying out SYN flood attack.
Initialize the list of malicious sensors (malicious_node) with the IDS of the attacker sensors.
Define a helper function "is_malicious_node" that verifies whether a given device ID is included in the catalogue of malicious sensors. If true, return 1; otherwise, return 0

Define the "syn_flood()" function:
a. Create a socket address with the IP address of the target node ("anySocketAddr").
b. Call "get_Remotesocket()" to acquire a socket (s) for transmitting SYN packets. This function must return a socket depending on the device ID and socket address provided.
c. If the socket s is not already created:
To create a new socket, use the "socket_creation()" function.
Using the "tcp_connect()" function, connect the new socket to the target sensor.
d. If the socket exists already:
Based on the event details, set the local device ID, remote device ID, and socket ID (sId) of the socket.
To transmit a SYN packet, invoke the "send_syn_packet()" function.
e. Create a timer event to schedule the next SYN packet transmission after a delay of 1000 simulation units time.

Define the "send_syn_packet(s)" function:
a. Create a SYN packet using the function "create_syn()".
b. Set the initial sequence number(ISS) in the TCP control block (TCB) of the socket
c. Change the state of the TCP connection to "TCPCONNECTION_SYNC_SENT".
d. Increment the number of SYN retries.
e. Refresh TCP metrics.
f. Use the "send_to_network()" function to send the SYN packet to the network.
g. Using the "add_timeout_event()" function, add a timeout event for the SYN transmission.

Define the "socket_creation()" function:
a.Create a new socket identifier (s_id).
b.Determine the socket interface (sId) using the event details.
c.Create a new socket by calling "tcp_create_socket()".
d.Configure the local and remote socket addresses for the newly created socket.
e. Assign local and remote device identifiers.

The algorithm begins by inputting the IDs of the malicious sensors into the list of attacking sensors (malicious_node). This phase establishes the list of sensor IDs that will be deemed malicious during a SYN flood attack. The algorithm defines the "is_malicious_node(devid)" utility function. This function accepts a device ID (devid) as input and determines if it is present in the "malicious_node" list. The function returns 1 if the device ID is found in the list, signifying that the device is malicious. If not, it returns 0. The algorithm specifies the "syn_flood ()" function, which executes the SYN flood attack. The stages of this function are:

a. In this step, the IP address of the target node is used to generate a socket address called "anySocketAddr". This socket address will be used to specify the SYN packets' destination.

b. call "get_Remotesocket()" function to acquire a socket (s) for transmitting SYN packets. This function should return a socket based on the supplied device ID and socket address.

c. If the socket s does not already exist (i.e., it is NULL), the algorithm creates a new socket by calling "socket_creation()". This new socket will be used for SYN flooding. After constructing the socket, the algorithm calls the "tcp_connect()" function to establish a TCP connection with the target sensor.

d. If the socket already exists, based on the event details, the algorithm sets the local device ID, remote device ID, and socket ID (sId) of the socket. This phase verifies that the socket is configured correctly for a SYN flood attack. The "send_syn_packet()" function is then called to send a SYN packet over the existing socket.

e. Finally, a timer event is created to schedule the next SYN packet transmission 1000 simulation units after the delay. This periodic scheduling ensures the continuous transmission of SYN packets during an attack.

The algorithm specifies the "send_syn_packet(s)" function, which is responsible for transmitting a SYN

packet over the specified interface. The stages of this function are:
a. The function uses the "create_syn()" function to generate a SYN packet. This function generates a SYN packet with the required preamble data.
b. The ISS is specified in the socket's TCB. This sequence number corresponds to the TCP connection's ISS.
c.The connection's state is altered to "TCPCONNECTION_SYN_SENT". This state indicates that a SYN packet has been transmitted and the socket is awaiting the ACK.
d. The quantity of SYN retry attempts increases. This counter tracks the number of times a SYN packet was transmitted without the expected response.
e.TCP metrics are modified to reflect the transmission of the SYN packet. This step involves updating TCP connection statistics, including the number of SYN packets sent.
f.The function "send_to_network()" is invoked to transmit the SYN packet to the network. This function sends the payload to its intended destination.
g.The "add_timeout_event()" function adds an additional timeout event. This event is used to manage situations in which the expected response (such as an Acknowledgement (ACK)) is not received within a specified time limit.

The algorithm concludes by defining the "socket_creation()" function, which is accountable for creating a new socket for the SYN flood attack. Let's examine the stages contained within this function:
a. A new socket ID is created (s_id). This ID will be utilised to identify the socket uniquely.
b. Based on the event details; the socket interface (sId) is determined. The socket interface provides access to functions and data structures particular to sockets.
c. A request is made to the "tcp_create_socket()" function to create a new socket. This function creates a new socket and allots memory for its data structures.
d.The local and remote socket addresses are configured for the newly created socket. With a port value of 0, the local socket address is set to the IP address of the device where the socket was created. The remote socket address is set to the IP address of the target node, along with the port number 0.
e. The socket is allocated both the local and remote device IDs. The local device ID is set to the device ID contained in the event details, whereas the remote device ID is assigned to the target node.
f. The newly constructed socket is returned as the function's final output.

The wireless sensors, the LoWPAN gateway, the router, and the wired node are used to create a network to transmit sensed data from the sensor to the wired node. *Table 1* provides information about the numbers of each device used in the simulation.

Table 1 Devices used

S. No.	Device used	Quantity
1	LoWPAN gateway	1
2	Wireless sensor	10
3	Router	1
4	Wired node	1

Table 2 shows the different parameters used in a LoWPAN gateway, which is a device that connects LoWPANs to the Internet.

- Flood time (in microseconds) is the amount of time that the network is flooded with data to find new nodes. It is set to 100 microseconds in this case.
- The method used to keep a network from getting too crowded when multiple devices send data at the same time. "New_reno" is used in this case.
- Maximum segment size: The largest amount of data that can be sent in a single section. It is set to 1460 bytes in this case.
- Transport layer protocol: The system that lets devices send and receive data. User datagram protocol (UDP) is used in this case.
- The amount of time a device has to wait after ending a link before it can use the same port again. It is set to 120 seconds in this case.
- Routing protocol: The protocol used to find the best way for data to get from source to target. AODV protocol is used in this case.
- Modulation is the process of turning digital data into analogue signs that can be sent. Offset Quadrature Phase-Shift Keying (O-QPSK) is used in this case.
- CCA mode: In this mode, the device listens to the channel to see if it is busy before sending data. This is called "carrier sense only mode." This method is used in this case.
- Type of device: The kind of device that is used in the network. "LoWPAN Gateway" is used in this case.

Table 2 LoWPAN gateway parameters and properties

S. No.	Parameter	Value
1	Flood time (micro sec.)	*Value*
2	Congestion control algorithm	100
3	Maximum segment size	New_reno
4	Transport layer Protocol	1460
5	Time-wait timer (Sec.)	UDP
6	VRouting protocol	120
7	Modulation technique	AODV
8	CCA mode	O-QPSK
9	Device type	Carrier sense only

The properties of various devices used in the simulation are shown in *Table 3*, *Table 4*, and *Table 5* for wireless sensors, router, and wired node respectively.

Table 3 Wireless sensor properties

S. No.	Parameter	Value
1	Device type	IoT sensor
2	Mobility model	No_mobility
3	Congestion control algorithm	New_reno
4	Maximum segment size	1460
5	Protocol	UDP
6	Routing protocol	AODV
7	Unit back off period (symbols)	20
8	Connection medium	Wireless
9	Power source	Battery
10	IdleModeCurrent (mA)	3.3
11	Initial energy (mAH)	0.5
12	Voltage (V)	3.6

Table 4 Router properties

S. No.	Parameter	Value
1	Device type	IoT router
2	Congestion control algorithm	New_reno
3	Maximum segment size	1460
4	Protocol	UDP

Table 5 Wired node properties

S. No.	Parameter	Value
1	Device type	Wired node
2	Interface count	1
3	Congestion control algorithm	New_reno
4	Protocol	Ethernet

Table 6 shows various simulation scenarios used to measure the performance of AODV routing protocols in IoT networks under SYN-flood attacks. We have used five different scenarios. Scenario 1 doesn't have any attack-node. This scenario is used to measure the AODV routing protocol's performance in normal conditions. There is one SYN-flood attack-node in simulation scenario -1, two in simulation scenario -2, three in simulation scenario -3, and four in simulation scenario -4, respectively. All other significant properties are the same for all the simulation scenarios. The mobility of attack nodes in simulation scenario -1 to simulation scenario -4 is fixed. The wireless sensors form an ad-hoc network using the AODV as a routing protocol. All the malicious nodes can put themselves in the ad-hoc link created.

Figure 3 shows simulation scenario -0, where there is no attack-node or sensor. This scenario is used to access the network's performance when there is no malicious sensor placed in it. Simulation scenario -1 is shown in *Figure 4* which has one malicious sensor. The sensor exploits the vulnerability of the network to carry out SYN-flood attacks on the IoT network. Similarly, we have *Figure 5*, *Figure 6*, and *Figure 7* showing simulation scenario -2, simulation scenario -3, and simulation scenario -4, respectively.

Each simulation scenario has different numbers of attack-nodes or sensors. simulation scenario -2, simulation scenario -3, and simulation scenario -4 have two, three, and four counts of malicious or attacking sensors, respectively.

As mentioned in *Figure 2*, we carried out simulations to measure various network performance metrics like throughput, delay, and jitter. The throughput of the given system is the number of packets received by the receivers in a unit of time. We can express the same mathematically as follows:

$$Throughput\ (t) = \frac{\sum number\_of\_packets_{received}}{\sum total\_time} \quad (1)$$

In Equation 1, the numerator (∑number_of_packets_received) represents the total number of packets received by the IoT network. In other terms, it equals the number of packets successfully delivered to their intended destinations.

The denominator (total_time) represents the total transmission time for these packets. It measures the time between the beginning of the transmission and the reception of the final payload.

Table 6 Simulation scenarios

S. No.	Scenario name	LoWPAN gateway properties	Wireless sensor properties	Router properties	wired node properties	No. of attack (SYN-flood) node
1	simulation scenario -0	Same as table 2	Same as table 3	Same as table 4	Same as table 5	0
2	simulation scenario -1					1
3	simulation scenario -2					2
4	simulation scenario -3					3
5	simulation scenario -4					4

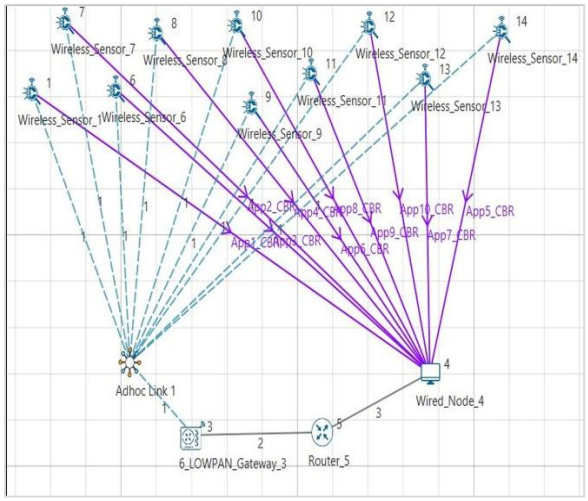

Figure 3 Simulation scenario -0 with no attack-node

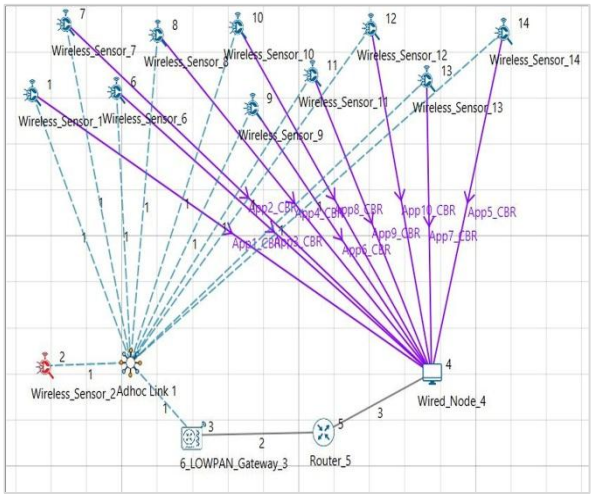

Figure 4 Simulation scenario -1 with one attack-node

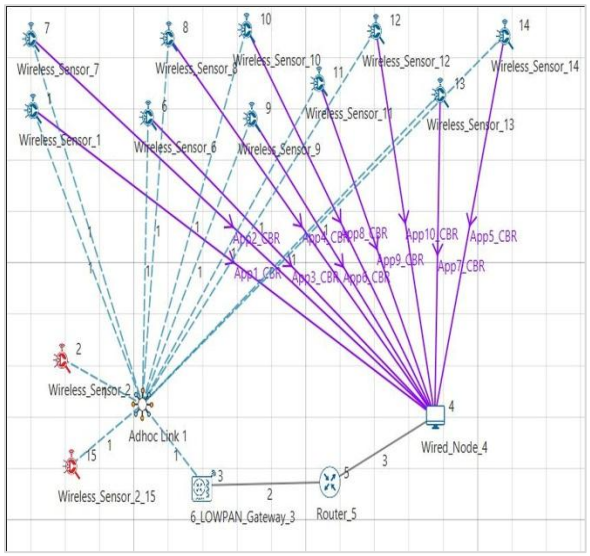

Figure 5 Simulation scenario -2 with two attack-nodes

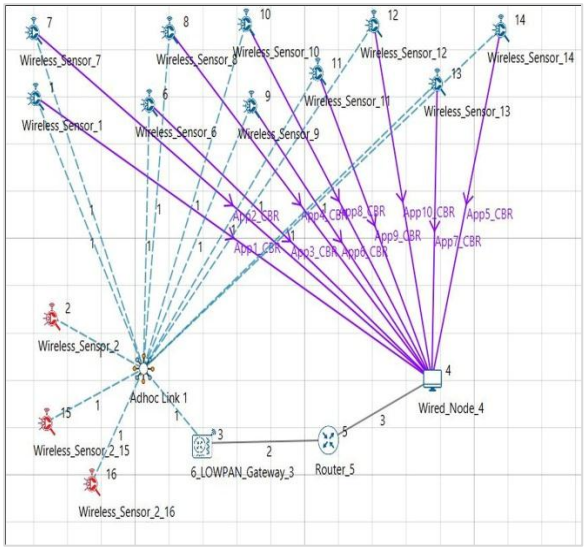

Figure 6 Simulation scenario -3 with three attack- nodes

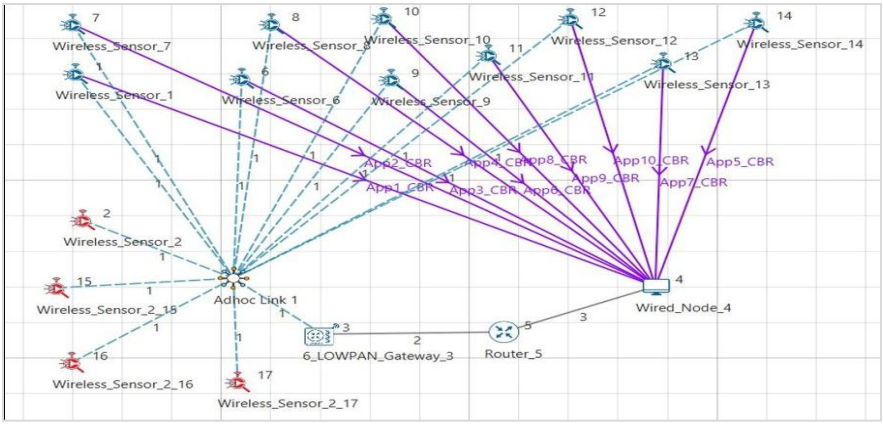

Figure 7 Simulation scenario -4 with four attack-node

To calculate the throughput, we add the total number of received packets and divide it by the total time. This ratio indicates the rate of data transfer or throughput by providing the average number of messages received per unit of time. Using Equation 1, we can evaluate the effectiveness and functionality of an IoT network. A higher throughput value indicates that the network can handle and transmit a greater volume of data more rapidly. A lower throughput, on the other hand, may indicate network congestion, packet loss, or delays in data transmission. The nodal delay is the amount of time it takes for a data payload to traverse a single network node. The nodal delay can be mathematically expressed as follows (Equation 2):

$$d_{nodal} = d_{queuing} + d_{processing} + d_{transmission} + d_{propagation} \quad (2)$$

Several factors contribute to this delay, which can be broken down into four components: queuing delay, processing delay, transmission delay, and propagation delay. Queuing delay refers to the amount of time a message spends in a queue before it can be processed by the node. Multiple packets may arrive simultaneously in an IoT network, and if the node is processing other packets, the inbound packets must wait in a queue. The queuing delay depends on the network's congestion level and the priority assigned to various packets.

Processing delay represents the time needed to process the transmission by the node. In an IoT network, the processing delay may entail tasks such as analyzing the packet's content, performing any necessary computations, and making decisions based on the received data. The processing delay is determined by the complexity of the processing duties and the capabilities of the node's hardware and software.

Transmission delay refers to the time required to transmit the payload from the source node to the destination node over the network link. It depends on the magnitude of the packet and the network link's available bandwidth. In an IoT network, factors such as the distance between nodes, the quality of the wireless connection, and network congestion can affect the transmission delay.

Propagation delay represents the time it takes for a signal or packet to travel from a node to its destination owing to the physical properties of the communication medium. In an IoT network, the propagation delay depends on factors such as the distance between nodes and the pace at which the signal propagates through the medium (e.g., radio waves in wireless networks or light waves in optical networks). Understanding and optimizing these components of nodal delay Equation 2 is necessary for designing and managing IoT networks to ensure efficient data transmission, minimal delay, and dependable device-to-device communication.

The variance in packet delay is referred to as jitter. Jitter for any packet is equal to the difference between the current packet's end-to-end delay and the prior packet's end-to-end delay. Jitter for any packet can be expressed as follows (Equation 3):

$$ǰ_{packet} = ǰ_{current\_packet1_{end-to-end\ delay}} - ǰ_{current\_packet2_{end-to-end\ delay}} \quad (3)$$

Where ǰ_packet1 represents the time at which packet 1 arrived at its destination and ǰ_packet2 represents the time at which packet 2 arrived at its destination. To calculate jitter using Equation 3, we must measure

the arrival times of multiple packets at the destination and then determine the absolute difference in arrival times between consecutive packets. This absolute difference provides the jitter value, which indicates how much the arrival times of individual packets vary.

The average packet-jitter for the complete application can express as follows:

$$ǰ_{pacentire-applicaion} = \frac{\sum ǰ_{packet}}{\sum receveied_{packets_{(success)}} - 1} \quad (4)$$

Where $ǰ_{(pacentire\text{-}applicaion)}$ represents the average packet jitter for the complete application. In other words, Equation 4 measures the average delay or time required for a packet to propagate within an IoT network from its source to its destination. $\sum ǰ_{packet}$ represents the aggregate of the individual transmission delays. It indicates that we must calculate the sum of the latencies for each of the sent packets. $\sum receveied_{packets_{(success)}}$ represents the total number of received packets that were delivered successfully to their destination. It implies that we must calculate the total number of packets that the intended recipient effectively received. The subtraction of 1 is intended to eliminate the first packet from the calculation. This may be done to exclude any initialization or setup time that may distort the delay measurement. Additional performance indicators like idle-state energy consumption of the number of half-open connections and overhead transmitted in wireless sensors are also measured. Half-open connections are created as a result of the SYN-flood attack. SYN-flood creates numerous half-open connections. These half-open connections seize available ports of the target node resulting in the exhaustion of the node's capacity to address genuine requests. Overhead transmitted is the total number of control overheads required to send packets and maintain a link. A wireless sensor is idle when it is not part of any active transmission or receiving of packets.

4.Results

Figure 8 shows a box analysis of throughputs under different attack scenarios. For simulation scenario -0, the median is 0.0101, the mean is 0.010047 and there are few outliers. For simulation scenario -1, the median is 0.008885, the mean is 0.009136 and there are no outliers. For simulation scenario -2, the median is 0.008763, the mean is 0.008763and there are no outliers, for simulation scenario -3, the median is 0.008377, the mean is 0.008377 and there are no outliers. Similarly, for simulation scenario -4, the median is 0.00826, the mean is 0.00826 and there are no outliers. We can observe that both the median and mean is decreasing for each simulation case indicating the number of attack sensors have impact on the performance of the network.

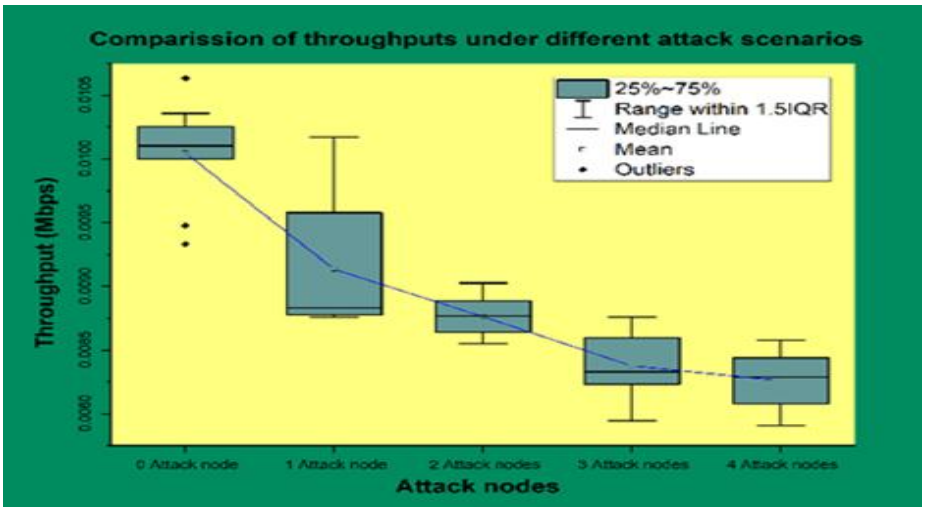

Figure 8 Average throughput (in Mbps) in various attack scenarios

Figure 9 shows the box plot presentation of the delays in the network caused by the attack sensors. We can observe that the median and mean of the delay also increase as the number of attack sensors increases in the simulation scenarios. The standard deviation of the delay values for each situation can be used to find out how different the values in the table are. The standard deviation is a way to measure how

spread out or variable a set of data is. Based on the table's delay numbers, here are the standard deviations for each attack scenario: simulation scenario -0: 0.065E+7, Simulation scenario -1: 0.156E+7, Simulation scenario -2: 0.124E+7, Simulation scenario -3: 0.087E+7 and Simulation scenario -4: 0.092E+7. Here all the values are in Micro Sec. These numbers show that the delay values vary in different ways depending on the type of attack. Simulation scenario -1 has the biggest standard deviation, which means that the delay values are more spread out and have a wider range. Simulation scenario -0, on the other hand, has the smallest standard deviation, which means that the delay numbers are closer to the mean.

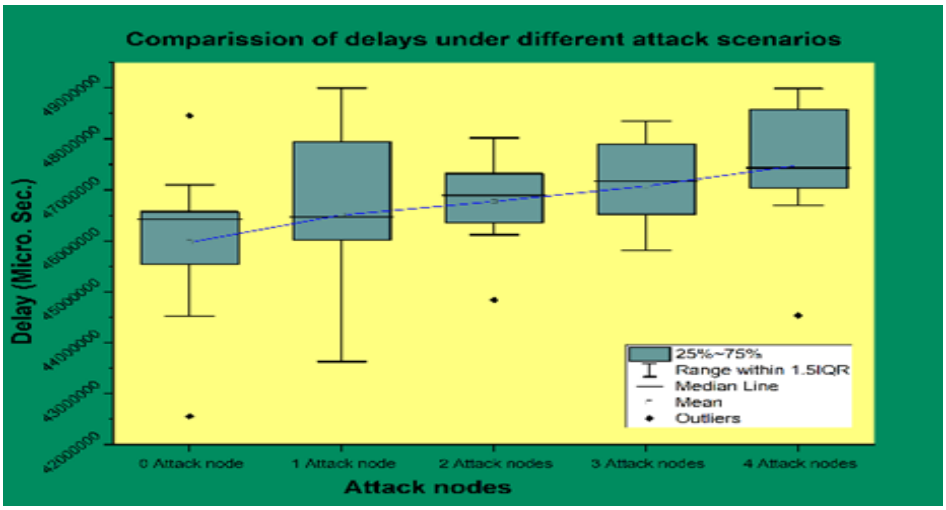

Figure 9 Average delay (in Micro Sec.) in various attack scenarios

Figure 10 based on the box plots shows a comparison of jitters under different attack scenarios. We can see that the distribution of jitter values in each attack situation (Simulation scenario -0 to Simulation scenario -4) is right-skewed with some outliers. The straight line in each box shows what the median number is. The boxes themselves show the interquartile range (IQR), which has 50% of the data. The whiskers go from the box to the minimum and highest values that are within 1.5 times the IQR. values outside of the whiskers are called "outliers."

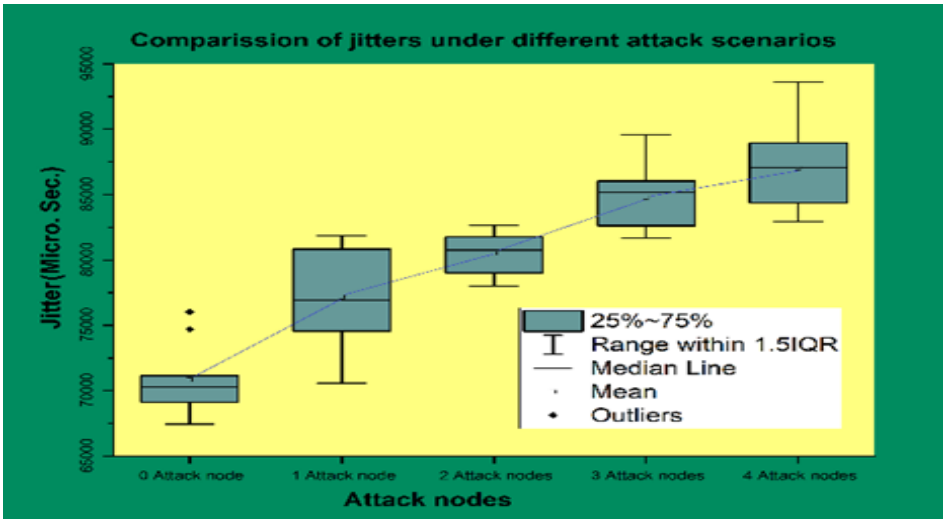

Figure 10 Average jitter (in Micro Sec.) in various attack scenarios

In Scenario (Simulation)-0, the data are a little bit clumped together, with a few outliers. About 67,000 microseconds is the average amount of jitter. Simulation scenario -1 has a slightly larger spread of values, with a median of about 77,000 microseconds and a few outliers. Simulation scenario -2 has a wider spread of values, with a median of about 80,000 microseconds. Scenario (Simulation) 3 has the most different numbers, with an average of about 85,000 microseconds. Simulation scenario -4 also has a wide range of numbers, with the average being about 87,000 microseconds. We can see that as the attack cases get worse (i.e., move from Simulation scenario -0 to Simulation scenario -4), the jitter values tend to go up, which means that there is more variation in how long it takes for packets to be sent.

Figure 11 display the number of half-open connections generated in response to various attack scenarios and time intervals. Each row represents a sensor number, while each column represents a time interval in seconds during which the number of half-open connections is recorded.

At 10 seconds, for sensor number 2 in Simulation scenario -1, a total of 20,017 half-open connections were created. Similarly, at 90 seconds for simulation scenario -2, 99,901 half-open connections were created for sensor number 10. The number of half-open connections can indicate how many connections to the target system are being established but not yet concluded. This can be an important metric for evaluating the impact of an attack on a system's availability. The table demonstrates that as the severity of the attack scenario increases from Simulation scenario -1 to Simulation scenario -4), the number of half-open connections increases. In addition, as time passes, the number of half-open connections for every scenario and sensor increases.

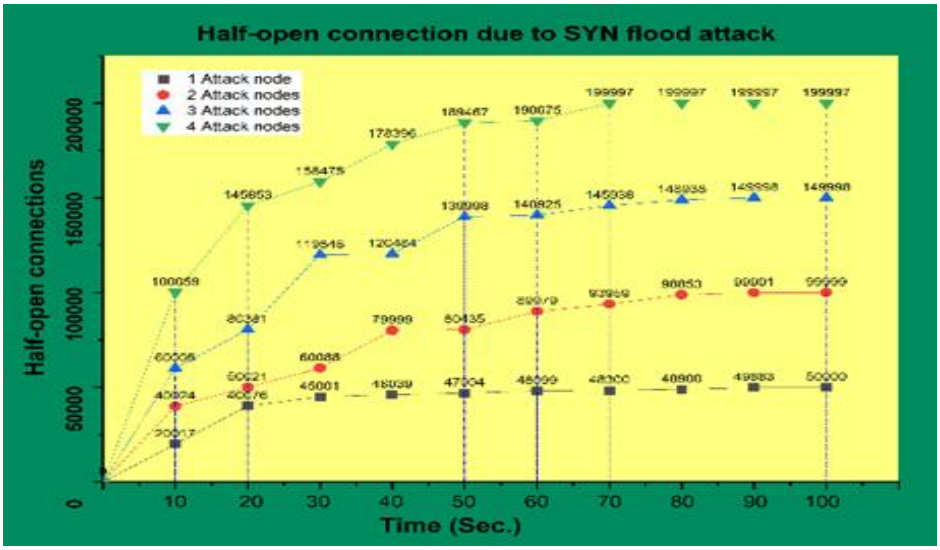

Figure 11 Number of half-open connections created in various attack scenarios

Figure 12 shows energy consumption by sensors in an idle state in various attack scenarios. The table and the figure indicate that as the number of attack nodes increases, so does the average energy consumption of the sensors in the idle state. The average energy consumption drops to 938.49 mj when there are no attack nodes. When one attack node is present, the average energy consumption rises to 1013.1 mj. It increases to 1052.21 mj when two attack nodes are present. The energy consumption continues to rise with values of 1060.12 mj and 1068.02 mj for three and four attack nodes, respectively. This suggests that the presence of attack nodes has a significant effect on the energy consumption of sensors in an inactive state, with a greater number of attack nodes resulting in a higher energy consumption. This can have significant implications for the design and operation of sensor networks, as increased energy consumption can reduce sensor lifetimes and network dependability.

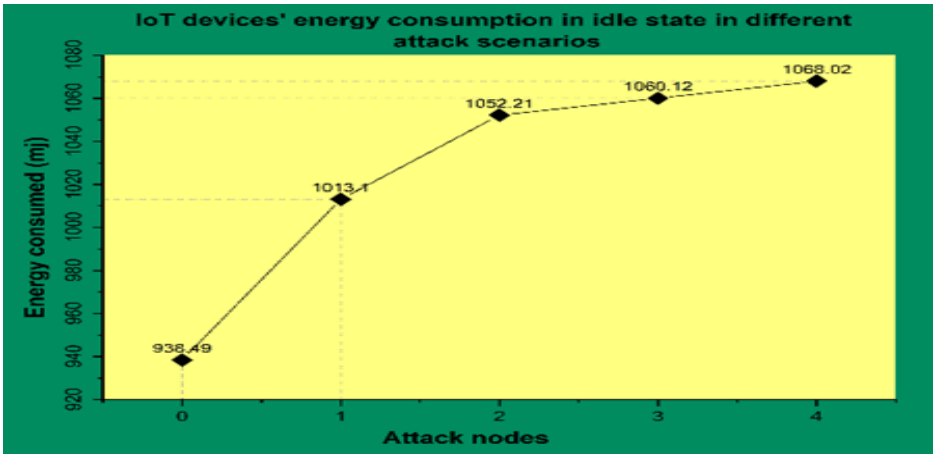

Figure 12 Energy consumption by sensors in idle state in various attack

As shown in *Figure 13*, the transmitted overhead increases as the number of attack nodes rises. The average overhead transmitted when there are no attack nodes is 683,590 bytes. When four attack nodes are present, however, the average overhead transmitted increases to 3,399,348 bytes. This increase in transmitted overhead is likely due to the increased communication requirements between sensors and attack nodes. The attack nodes may be transmitting additional packets to the sensors to disrupt the network. This additional communication creates additional overhead and can negatively influence the network's overall performance.

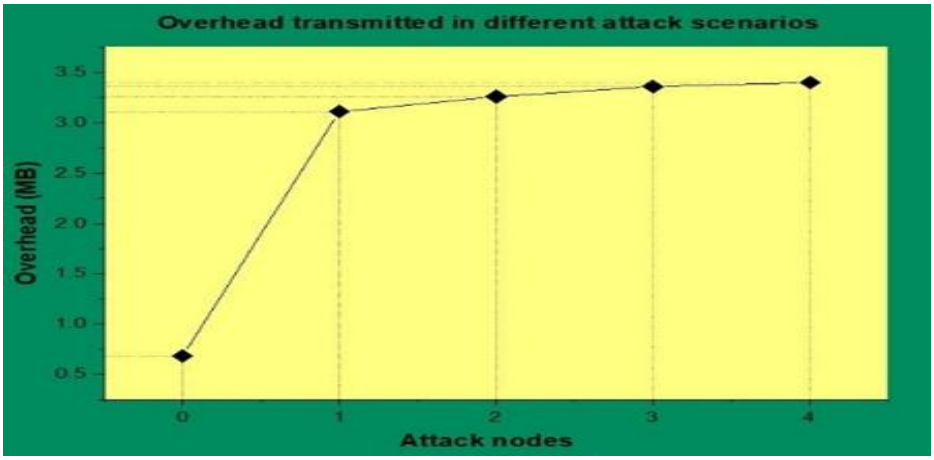

Figure 13 Overhead transmitted in various attack scenarios

5.Discussions

This study explores the consequences of SYN-flood attacks on IoT networks and establishes the possibility of such attacks in IoT networks. In an SYN-flood attack, a malicious node maliciously requests a connection establishment with a genuine wireless sensor. However, the malicious node does not reply to the SYN message from the wireless sensor, thereby keeping the connection half-open. The malicious node then creates a large number of half-open connections with other wireless sensors in the network in a similar manner. Due to these half-open connections between the genuine wireless sensors and the malicious node, other legitimate sensor nodes are unable to establish connections with the compromised sensor. This results in service deprivation or denial. *Table 6* presents the simulation scenarios used in our investigation. Simulation scenario -0 has no attack nodes, while Simulation scenario -1 has one SYN-flood attack node. Similarly, Simulation scenario -2, Simulation scenario -3, and Simulation scenario -4 have two, three, and four SYN-flood attack nodes respectively. *Figure 8* demonstrates the effects of different numbers of SYN-flood attack nodes on IoT networks in terms of throughput. Throughput is calculated

using Equation 1. The existence of attack nodes negatively affects the throughput. The graphic clearly illustrates a significant drop in the throughput of IoT networks due to the presence of malicious nodes in various simulated scenarios. Additionally, the mean and median of the throughputs of all wireless sensors decrease as a result of the attack.

Table 7 compares the effect of SYN-flood attacks on overhead transmission (in MB) in IoT networks with other network types [65].We observe that the overhead transmission for IoT networks due to SYN-flood attacks is higher compared to other networks under similar attacks. This situation is concerning because IoT networks typically have lower bandwidth than other networks.

Figure 9 indicates the influence of SYN-flood attacks on network delay. The genuine sensor nodes require multiple attempts to establish connections due to the presence of unsolicited SYN-flood packets in the network. Consequently, the overall delay for the network, as well as for individual sensors, increases. Delay is calculated using Equation 2. The trend of increased delay is evident in *Table 8*. From *Figure 8*, we can easily observe that the mean and median delay for the sensors increase in successive simulation scenarios.

The presence of malicious SYN-flood attack nodes in the IoT network affects the network's jitter. The attack leads to higher jitters in the network, as observed in *Figure 10*. Jitter is calculated using Equation 3 and Equation 4. The SYN-flood attack nodes significantly reduce the performance of the network by drastically increasing the jitter in every simulation scenario.

Figure 11 shows the number of half-open connections opened by the SYN-flood attack nodes. The figure illustrates a significant increase in half-open connections every second, and the rate of creation of half-open connections is higher in scenarios with a higher number of attack nodes.

The idle state energy consumption of each wireless sensor is shown in *Figure 12*. The attack nodes constantly attempt to send fake connection requests to genuine sensors. Although there is no actual data communication between them, the connections remain half-open. As a result, the wireless sensors spend their precious battery life listening to the communication channel, leading to decreased power levels even in an idle state.

Figure 13 demonstrates the overhead transmitted in various simulation scenarios. The overhead packets are responsible for maintaining the network. Due to the attack, the overhead transmission increases tremendously, resulting in higher congestion in the network. This, in turn, increases the delay and jitter of the network, thereby reducing the throughput.

Table 8 compares the effect of SYN-flood attacks in delay (Millisec.) in IoT networks and other network types [65]. The effect of SYN-flood attacks on delay in IoT networks and other network types is displayed in *Table 8*. SYN-flood attacks are a form of DoS attack that exploits the TCP three-way handshake vulnerability. A large number of SYN packets are sent to the target server in this type of attack, but the perpetrator does not complete the handshake, leaving the server waiting for a response. The table demonstrates that as the number of attack-nodes increases, so does the delay in IoT networks. When there are no attack-nodes, for instance, the delay in IoT networks is 46,000 milliseconds. However, the delay increases to 46500 milliseconds with one attack-node and 47500 milliseconds with four attack-nodes. This implies that the greater the number of attack-nodes in an IoT network, the greater the delay caused by SYN-flood attacks.

The table also compares the impact of SYN-flood attacks on delay in IoT networks with the effect of blackhole attacks on delay. The blackhole attack is a method of redirecting attack traffic to a "blackhole" or null interface. The null interface discards all incoming packets, effectively preventing attack traffic from reaching its destination. As shown in the table, the delay in IoT networks subject to a blackhole attack is considerably less than the delay in IoT networks subject to a SYN flood attack. For instance, the delay in IoT networks without a black hole attack node is 58 milliseconds, whereas, with one, the delay is only 65 milliseconds. This implies that SYN deluge attacks are more damaging than blackhole attacks.

The table also contrasts the delay caused by SYN-flood attacks in 5G mmWave networks with and without the Round Robin and Proportionally Fair scheduling algorithms. The Round Robin algorithm allocates resources equally to all users, whereas the Proportional Fair algorithm prioritizes users with poor channel conditions to enhance the quality of service. The delay in 5G mmWave networks with the Round Robin scheduling algorithm is greater than the delay with the Proportional Fair scheduling

algorithm, as shown in the table. For example, with four attack-nodes, the Round Robin scheduling algorithm incurs a delay of 2245.411 milliseconds, whereas the Proportional Fair scheduling algorithm incurs a delay of only 2240.051 milliseconds. This indicates that the Proportional Fair scheduling algorithm is more effective at reducing the impact of SYN-flood attacks on delay in 5G mmWave networks. In conclusion, the table provides insightful information regarding the effect of SYN-flood attacks on delay in various network types. These insights can assist network administrators and security experts in developing effective strategies to defend IoT networks against SYN-flood attacks. The effect of SYN-flood attacks on jitter in IoT networks and other network types is displayed in *Table 9*. Jitter is the variation in network delay between packets, and it is a crucial performance metric for real-time applications. Jitter increases as the number of attack nodes increases, indicating a decline in network performance. The table demonstrates that the SYN-flood attack has a considerable impact on the jitter in an IoT network, with the jitter increasing from 70.8 milliseconds to 87.0 milliseconds as the number of attack nodes increases from 0 to 4. This indicates that the attack causes a substantial variation in the delay between packets, which can have a negative impact on the performance of real-time applications.

In contrast, SYN-flood attacks in 5G millimeter wave (mmWave) networks employing Round Robin and Proportional Fair algorithms have a significantly lesser impact on jitter. As the number of attack nodes increases from 0 to 4, the jitter increases from 5.19 to 8.37 milliseconds. This suggests that these networks are more resistant to SYN-flood attacks and more suitable for real-time applications. Overall, the table emphasizes the significance of network resilience and the need to evaluate the impact of security attacks on network performance using various performance metrics, such as delay and jitter. In addition, it emphasizes the need for comprehensive security mechanisms in IoT networks to mitigate the performance impact of attacks.

Table 7 Effect of SYN flood attacks in overhead transmission (in MB) in IoT networks and other networks

No. of attack-nodes	SYN attack in IoT network	Blackhole attack	No. of attack-nodes	SYN attack in IoT network
0	0.683	0.70	0.683	0.63
1	3.11	2.8	3.10	3.14
2	3.26	N.A	3.25	3.21
3	3.36	N.A	3.36	3.31
4	3.40	N.A	4.40	3.38

Table 8 Effect of SYN-flood attacks in delay (Millisec.) in IoT networks and other network types

No. of attack-nodes	SYN attack in IoT network	Blackhole attack in IoT network	SYN attack with Round Robin algo. in 5G mmWave network	SYN attack with proportional fair algo. in 5G mmWave network
0	46000	58	2156.647	2142.647
1	46500	65	2190.939	2175.939
2	46800	N.A	2227.517	2180.534
3	47100	N.A	2234.770	2234.770
4	47500	N.A	2245.411	2240.051

Table 9 Effect of SYN-flood attacks in jitter (Millisec.) IoT networks and other network types

No. of attack-nodes	SYN attack in IoT network	SYN attack with Round Robin algo. in 5G mmWave network	SYN attack with proportional fair algo. in 5G mmWave network
0	70.8	5.19	5.23
1	77.2	5.36	5.36
2	80.5	6.74	6.88
3	84.8	7.45	7.44
4	87.0	8.33	8.37

Our investigation's critical evaluations are as follows:

1. An IoT network framework has vulnerabilities that attackers can exploit. Our team exploited one of these flaws to launch an SYN-flood attack. When an SYN packet arrives at wireless sensors, a compromised IoT network creates half-open connections. When other sensors send a series of SYN packets to the compromised sensor, it opens numerous connections without sending any data. Consequently, the attack consumes all available resources, preventing genuine service connections.

2. The average throughput of the IoT network has significantly decreased. As the number of attack nodes increases, the drop in throughput becomes more pronounced. In various attack scenarios, the attack results in a 9.1% to 17.79% decrease in throughput.
3. There is a slight increase in network delay due to the attack. In various attack scenarios, the attack leads to a 1.08% to 3.26% increase in delay.
4. The SYN-flood attack type causes an increase in network jitter. The attack results in a 15.4% to 22.88% increase in network jitter.
5. The attack results in an increased number of half-open connections, leading to the exhaustion of available ports. This port exhaustion deprives genuine nodes of connections. Additionally, the excessive number of half-open connections requires a higher number of overhead transmissions, resulting in increased traffic and congestion in the network.
6. The attack has resulted in increased energy consumption by sensors in an idle state in various attack scenarios.

5.1 Limitations

- The attacking sensors can easily degrade the performance of the network. However, if the attacking node wishes to attack a specific sensor, it must be aware of the network of the target node. Therefore, the attacking sensor must be mindful of the neighbourhood of the target user equipment.
- Continuously transmitting packets (SYN) causes a high battery drain. Therefore, the duration of the attack is dependent on the power source (battery) of the attacking sensor.

A complete list of abbreviations is shown in *Appendix I.*

6. Conclusion and future work

The existence of a vulnerability was demonstrated in this paper in IoT networks that utilized the AODV protocol as their routing protocol. This vulnerability resided in the transport layer of the IoT network framework, enabling it to be exploited by intruders to create multiple half-open connections. Exploiting this vulnerability allowed intruders to launch SYN-flood attacks, which had a severe impact on the network's performance. The SYN-flood attack leads to deteriorated performance metrics, including increased delay and jitter, as well as decreased throughput. Specifically, the attack resulted in a decrease in throughput ranging from 9.1% to 17.79%, an increase in delay ranging from 1.08% to 3.26%, and an increase in jitter ranging from 15.4% to 22.88% within the network. Additionally, the attack increased energy consumption for idle wireless sensors and significantly raised overhead transmissions due to the increased number of half-open connections resulting from the SYN-flood attack.

In future research, methods for mitigating the impact of SYN-flood attacks in IoT networks will be investigated. Additionally, there is a plan to develop an intrusion detection system specifically tailored for detecting and eliminating such attacks in IoT-based networks. This will involve the utilization of machine learning and deep learning techniques.

Acknowledgment

None.

Conflicts of interest

The authors have no conflicts of interest to declare.

Author's contribution statement

Abhijit Biswas: Conceptualization, creation of test-bed of IoT network scenarios, review of the original draft of the manuscript. **Rabinder kumar Prasad**: Conceptualization, investigation, performed the analysis. **Abhijit Boruah:** performed the analysis & writing the original draft and editing the manuscript. **Sudipta Majumder:** Conceptualization, Investigation, supervision and revised the article for critical intellectual content and approved the final draft of the manuscript.

References

[1] Jha AV, Appasani B, Ghazali AN. Performance evaluation of network layer routing protocols on wireless sensor networks. In international conference on communication and electronics systems 2019 (pp. 1862-5). IEEE.

[2] Tiwary A, Mahato M, Chidar A, Chandrol MK, Shrivastava M, Tripathi M. Internet of things (IoT): research, architectures and applications. International Journal on Future Revolution in Computer Science & Communication Engineering. 2018; 4(3):23-7.

[3] González-zamar MD, Abad-segura E, Vázquez-cano E, López-meneses E. IoT technology applications-based smart cities: research analysis. Electronics. 2020; 9(8):1-36.

[4] Raykarmakar K, Harrison S, Ghata S, Das A. Medical internet of things: techniques, practices and applications. Medical Internet of Things: Techniques, Practices and Applications. 2021.

[5] Pipkin DL. Halting the hacker: a practical guide to computer security. Prentice Hall Professional; 2003.

[6] Mavaluru D, Enduri MK, Thiyagarajan A, Anamalamudi S, Srinivasan K, Carie CA, et al. An AI fuzzy clustering-based routing protocol for vehicular

image recognition in vehicular ad hoc IoT networks. Soft Computing. 2023:1-12.
[7] Nourildean SW, Hassib MD, Abd MY. AD-Hoc routing protocols in WSN-WiFi based IoT in smart home. In 15th international conference on developments in esystems engineering 2023 (pp. 82-7). IEEE.
[8] Adarbah HY, Moghadam MF, Maata RL, Mohajerzadeh A, Al-badi AH. Security challenges of selective forwarding attack and design a secure ECDH-based authentication protocol to improve RPL security. IEEE Access. 2022; 11:11268-80.
[9] Prasath N, Sreemathy J. Optimized dynamic source routing protocol for MANETs. Cluster Computing. 2019; 22(Suppl 5):12397-409.
[10] Darville C, Höfner P, Ivankovic F, Pam A. Advanced models for the OSPF routing protocol. Electronic Proceedings in Theoretical Computer Science, EPTCS. 2022; 355:3-26.
[11] Kim D, Andalibi V, Camp J. Protecting IoT devices through localized detection of BGP hijacks for individual things. In security and privacy workshops 2021(pp. 260-7). IEEE.
[12] Mistareehi H, Salameh HB, Manivannan D. An on-board hardware implementation of AODV routing protocol in VANET: design and experimental evaluation. In international conference on internet of things: systems, management and security 2022 (pp. 1-6). IEEE.
[13] Bertino E, Martino L, Paci F, Squicciarini A, Bertino E, Martino LD, et al. Web services threats, vulnerabilities, and countermeasures. Security for Web Services and Service-Oriented Architectures. 2010:25-44.
[14] Bashir AK, Rawat DB, Wu J, Imran MA. Guest editorial security, reliability, and safety in IoT-enabled maritime transportation systems. IEEE Transactions on Intelligent Transportation Systems. 2023; 24(2):2275-81.
[15] Swamy SN, Kota SR. An empirical study on system level aspects of internet of things (IoT). IEEE Access. 2020; 8:188082-134.
[16] Ghanti S, Naik GM. Defense techniques of SYN flood attack characterization and comparisons. International Journal of Network Security. 2018; 20(4):721-9.
[17] Fan CI, Wang JH, Shie CH, Tsai YL. Software-defined networking integrated with cloud native and proxy mechanism: detection and mitigation system for TCP SYN flooding attack. In 17th international conference on ubiquitous information management and communication 2023 (pp. 1-8). IEEE.
[18] Sheibani M, Konur S, Awan I. DDoS attack detection and mitigation in software-defined networking-based 5G mobile networks with multiple controllers. In 9th international conference on future internet of things and cloud 2022 (pp. 32-9). IEEE.
[19] Kumar BS, Gowda KK. Detection and prevention of TCP SYN flooding attack in WSN using protocol dependent detection and classification system. In international conference on data science and information system 2022 (pp. 1-6). IEEE.
[20] Kumar R, Lal SP, Sharma A. Detecting TCP SYN flood attack in the cloud. Journal of Software. 2017; 12(7):493-506.
[21] Zrelli A, Khlaifi H, Ezzedine T. Performance evaluation of AODV and OAODV for several WSN/IoT applications. In international conference on software, telecommunications and computer networks 2019 (pp. 1-6). IEEE.
[22] Silva BN, Khan M, Han K. Internet of things: a comprehensive review of enabling technologies, architecture, and challenges. IETE Technical Review. 2018; 35(2):205-20.
[23] Atzori L, Iera A, Morabito G. The internet of things: a survey. Computer Networks. 2010; 54(15):2787-805.
[24] Li S, Xu LD, Zhao S. The internet of things: a survey. Information Systems Frontiers. 2015; 17:243-59.
[25] Gubbi J, Buyya R, Marusic S, Palaniswami M. Internet of things (IoT): a vision, architectural elements, and future directions. Future Generation Computer Systems. 2013; 29(7):1645-60.
[26] Butun I, Österberg P, Song H. Security of the internet of things: vulnerabilities, attacks, and countermeasures. IEEE Communications Surveys & Tutorials. 2019; 22(1):616-44.
[27] Hassija V, Chamola V, Saxena V, Jain D, Goyal P, Sikdar B. A survey on IoT security: application areas, security threats, and solution architectures. IEEE Access. 2019; 7:82721-43.
[28] Makhdoom I, Abolhasan M, Lipman J, Liu RP, Ni W. Anatomy of threats to the internet of things. IEEE Communications Surveys & Tutorials. 2018; 21(2):1636-75.
[29] Khan F, Al-atawi AA, Alomari A, Alsirhani A, Alshahrani MM, Khan J, et al. Development of a model for spoofing attacks in internet of things. Mathematics. 2022; 10(19):1-16.
[30] Fadele AA, Othman M, Hashem IA, Yaqoob I, Imran M, Shoaib M. A novel countermeasure technique for reactive jamming attack in internet of things. Multimedia Tools and Applications. 2019; 78:29899-920.
[31] Ambika N. Tackling jamming attacks in IoT. Internet of Things (IoT) Concepts and Applications. 2020:153-65.
[32] Sharma B, Sharma L, Lal C, Roy S. Anomaly based network intrusion detection for IoT attacks using deep learning technique. Computers and Electrical Engineering. 2023; 107:108626.
[33] Namvar N, Saad W, Bahadori N, Kelley B. Jamming in the internet of things: a game-theoretic perspective. In global communications conference 2016 (pp. 1-6). IEEE.
[34] Lin J, Yu W, Zhang N, Yang X, Zhang H, Zhao W. A survey on internet of things: architecture, enabling technologies, security and privacy, and applications. IEEE Internet of Things Journal. 2017; 4(5):1125-42.
[35] Inayat U, Zia MF, Mahmood S, Khalid HM, Benbouzid M. Learning-based methods for cyber

attacks detection in IoT systems: a survey on methods, analysis, and future prospects. Electronics. 2022; 11(9):1-20.
[36] Kumar A, Varadarajan V, Kumar A, Dadheech P, Choudhary SS, Kumar VA, et al. Black hole attack detection in vehicular ad-hoc network using secure AODV routing algorithm. Microprocessors and Microsystems. 2021; 80:103352.
[37] Pelechrinis K, Iliofotou M, Krishnamurthy SV. Denial of service attacks in wireless networks: the case of jammers. IEEE Communications Surveys & Tutorials. 2010; 13(2):245-57.
[38] Baig ZA, Sanguanpong S, Firdous SN, Nguyen TG, So-in C. Averaged dependence estimators for DoS attack detection in IoT networks. Future Generation Computer Systems. 2020; 102:198-209.
[39] Gamundani AM. An impact review on internet of things attacks. In international conference on emerging trends in networks and computer communications 2015 (pp. 114-8). IEEE.
[40] Balamurugan B, Biswas D. Security in network layer of IoT: possible measures to preclude. In security breaches and threat prevention in the internet of things 2017 (pp. 46-75). IGI Global.
[41] Pundir S, Wazid M, Singh DP, Das AK, Rodrigues JJ, Park Y. Designing efficient sinkhole attack detection mechanism in edge-based IoT deployment. Sensors. 2020; 20(5):1-27.
[42] Chugh K, Aboubaker L, Loo J. Case study of a black hole attack on LoWPAN-RPL. In the sixth international conference on emerging security information, systems and technologies 2012 (pp. 157-62).
[43] Pongle P, Chavan G. Real time intrusion and wormhole attack detection in internet of things. International Journal of Computer Applications. 2015; 121(9):1-9.
[44] Kamaleshwar T, Lakshminarayanan R, Teekaraman Y, Kuppusamy R, Radhakrishnan A. Self-adaptive framework for rectification and detection of black hole and wormhole attacks in 6lowpan. Wireless Communications and Mobile Computing. 2021; 2021:1-8.
[45] Deogirikar J, Vidhate A. Security attacks in IoT: a survey. In international conference on IoT in social, mobile, analytics and cloud 2017 (pp. 32-7). IEEE.
[46] Kaur B, Dadkhah S, Shoeleh F, Neto EC, Xiong P, Iqbal S, et al. Internet of things (IoT) security dataset evolution: challenges and future directions. Internet of Things. 2023:100780.
[47] Moudoud H, Mlika Z, Khoukhi L, Cherkaoui S. Detection and prediction of FDI attacks in IoT systems via hidden Markov model. IEEE Transactions on Network Science and Engineering. 2022; 9(5):2978-90.
[48] Aboelwafa MM, Seddik KG, Eldefrawy MH, Gadallah Y, Gidlund M. A machine-learning-based technique for false data injection attacks detection in industrial IoT. IEEE Internet of Things Journal. 2020; 7(9):8462-71.
[49] Habib AA, Hasan MK, Alkhayyat A, Islam S, Sharma R, Alkwai LM. False data injection attack in smart grid cyber physical system: issues, challenges, and future direction. Computers and Electrical Engineering. 2023; 107:108638.
[50] Tankard C. Advanced persistent threats and how to monitor and deter them. Network Security. 2011; 2011(8):16-9.
[51] Mercy PP, Basil XS, Jose A, Kathrine GJ, Andrew J. Variants of crypto-jacking attacks and their detection techniques. In international conference on applications and techniques in information security 2022 (pp. 71-87). Singapore: Springer Nature Singapore.
[52] Singh SK, Kumar S. Blockchain technology: introduction, integration, and security issues with IoT. Applications of Blockchain and Big IoT Systems: Digital Solutions for Diverse Industries. 2022.
[53] Aziz Al Kabir M, Elmedany W, Sharif MS. Securing IoT devices against emerging security threats: challenges and mitigation techniques. Journal of Cyber Security Technology. 2023:1-25.
[54] Pathak AK, Saguna S, Mitra K, Åhlund C. Anomaly detection using machine learning to discover sensor tampering in IoT systems. In international conference on communications 2021 (pp. 1-6). IEEE.
[55] Kaushik K, Singh V, Manikandan VP. A novel approach for an automated advanced MITM attack on IoT networks. In international conference on advancements in interdisciplinary research 2022 (pp. 60-71). Cham: Springer Nature Switzerland.
[56] Dogan-tusha S, Althunibat S, Qaraqe M. A novel sybil attack detection mechanism for mobile IoT networks. In global communications conference 2022 (pp. 1838-43). IEEE.
[57] Anjum A, Olufowobi H. Towards mitigating blackhole attack in NDN-enabled IoT. In international conference on consumer electronics 2023 (pp. 1-6). IEEE.
[58] Wang C, Chen J, Yang Y, Ma X, Liu J. Poisoning attacks and countermeasures in intelligent networks: status quo and prospects. Digital Communications and Networks. 2022; 8(2):225-34.
[59] Sanders K, Yau SS. An effective approach to protecting low-power and lossy IoT networks against blackhole attacks. In international conferences on internet of things (ithings) and green computing & communications (GreenCom) and Cyber, Physical & Social Computing (CPSCom) and Smart Data (SmartData) and Congress on Cybermatics (Cybermatics) 2021 (pp. 65-72). IEEE.
[60] Sahay R, Geethakumari G, Mitra B, Thejas V. Exponential smoothing based approach for detection of blackhole attacks in IoT. In international conference on advanced networks and telecommunications systems 2018 (pp. 1-6). IEEE.
[61] Alghamdi R, Bellaiche M. A cascaded federated deep learning based framework for detecting wormhole attacks in IoT networks. Computers & Security. 2023; 125:103014.

[62] Pu C, Choo KK. Lightweight Sybil attack detection in IoT based on bloom filter and physical unclonable function. Computers & Security. 2022; 113:102541.
[63] Srinivas T, Manivannan SS. Black hole and selective forwarding attack detection and prevention in IoT in health care sector: hybrid meta-heuristic-based shortest path routing. Journal of Ambient Intelligence and Smart Environments. 2021; 13(2):133-56.
[64] Hariri A, Giannelos N, Arief B. Selective forwarding attack on iot home security kits. In computer security: ESORICS international workshops, CyberICPS, SECPRE, SPOSE, and ADIoT, Luxembourg city, 2020 (pp. 360-73). Springer International Publishing.
[65] Saikia B, Majumder S. Analysis of performance vulnerability of MAC scheduling algorithms due to SYN flood attack in 5G NR mmWave. International Journal of Advanced Technology and Engineering Exploration. 2021; 8(82):1102-19.

Abhijit Biswas is currently serving as an Assistant Professor in the Department of CSE, TSSOT, Assam University since October 2009. He is an alumnus of NIT Bhopal and earned his M.Tech and PhD degrees from NERIST in the years 2009 and 2018, respectively. His research interests include Network Routing, Network-On-Chip Topologies, and Data Science.
Email: abhijit.btcs06@gmail.com

Rabinder Kumar Prasad is presently an Assistant Professor in the Department of Computer Science and Engineering at D.U.I.E.T., Dibrugarh University. He holds a B.E. degree from Jorhat Engineering College, an M.Tech from Tezpur University, and a Ph.D. from Dibrugarh University. His research area broadly encompasses Data Analysis, Machine Learning Techniques, and Computer Networks.
Email: rkp@dibru.ac.in

Abhijit Boruah obtained his Bachelor's degree in Computer Science and Engineering, Master's degree in Information Technology from Tezpur University's School of Engineering, and completed his PhD in Computer Science from Dibrugarh University in 2023. He has been serving as an Assistant Professor at the Department of Computer Science and Engineering, Dibrugarh University Institute of Engineering and Technology since 2012. His research focuses on Robot Kinematics, Ontology Design, Machine Learning and Computer Networks.
Email: abhijit.boruah@dibru.ac.in

Sudipta Majumder completed his B.Tech in Computer Science & Engineering from NERIST in 2009. He has done his M.Tech degree in IT and PhD degree in Network Security from the Department of Electronics and Communication Engineering. His research interest is IoT, Peer to Peer Networks, Wireless Networks and Network Security. Currently, he is working as a senior assistant professor at CSE department, DUIET, Dibrugarh University. He is also a life member of the Computer Society of India.
Email: sudipta2020@dibru.ac.in

Appendix I

S. No.	Abbreviation	Description
1	5G	Fifth Generation
2	ACK	Acknowledgement
3	AODV	Ad-hoc On-Demand Distance Vector
4	ARP	Address Resolution Protocol
5	BGP	Border Gateway Protocol
6	DDoS	Distributed Denial of Service
7	DNS	Domain Name System
8	DoS	Denial-of-Service
9	DSR	Dynamic Source Routing
10	ID	Identifier
11	IEEE	Institute of Electrical and Electronics Engineers
12	IP	Internet Protocol
13	IPv6	Internet Protocol Version 6
14	IQR	Interquartile Range
15	IoT	Internet of Things
16	ISS	Initial Sequence Number
17	LoWPAN	Low Power Personal Area Network
18	MAC	Media Access Control
19	MANETs	Mobile ad-hoc Networks
20	MITM	Man-in-the-Middle
21	MQTT	Message Queuing Telemetry Transport
22	OSPF	Open Shortest Path First
23	O-QPSK	Offset Quadrature Phase-Shift Keying
24	PDR	Packet Delivery Ratio
25	RPL	Routing Protocol for Low-Power and Lossy Networks
26	RREQ	Route Request
27	RREP	Route Reply
28	RF	Radio Frequency
29	SDN	Software-Defined Networking
30	SPO_2	Oxygen Saturation
31	SQL	Structured Query Language
32	SYN	Synchronization
33	TCB	TCP Control Block
34	TCP	Transmission Control Protocol
35	UDP	User Datagram Protocol
36	VANET	Vehicular Ad hoc Networks
37	WOA	Web-Oriented Architecture
38	XML	eXtensible Markup Language