telecom

MDPI

Article

Node-Based QoS-Aware Security Framework for Sinkhole Attacks in Mobile Ad-Hoc Networks

Bukohwo Michael Esiefarienrhe *, Thulani Phakathi and Francis Lugayizi

MaSIM, Computer Science and IS Department, North-West University, Mahikeng 2735, South Africa; katshego@gmail.com (T.P.); francis.lugayizi@nwu.ac.za (F.L.)
* Correspondence: 25840525@nwu.ac.za; Tel.: +27-(0)-838650429

Abstract: Most networks strive to provide good security and an acceptable level of performance. Quality of service (QoS) plays an important role in the performance of a network. Mobile ad hoc networks (MANETs) are a decentralized and self-configuring type of wireless network. MANETs are generally challenging and the provision of security and QoS becomes a huge challenge. Many researchers in literature have proposed parallel mechanisms that investigate either security or QoS. This paper presents a security framework that is QoS-aware in MANETs using a network protocol called optimized link state routing protocol (OLSR). Security and QoS targets may not necessarily be similar but this framework seeks to bridge the gap for the provision of an optimal functioning MANET. The framework is evaluated for throughput, jitter, and delay against a sinkhole attack presented in the network. The contributions of this paper are (a) implementation of a sinkhole attack using OLSR, (b) the design and implementation of a lightweight-intrusion detection system using OLSR, and (c) a framework that removes fake routes and bandwidth optimization. The simulation results revealed that the QoS-aware framework increased the performance of the network by more than 70% efficiency in terms of network throughput. Delay and jitter levels were reduced by close to 85% as compared to when the network was under attack.

Keywords: routing protocols; MANETs; sinkhole attack; QoS in MANET

Citation: Esiefarienrhe, B.M.; Phakathi, T.; Lugayizi, F. Node-Based QoS-Aware Security Framework for Sinkhole Attacks in Mobile Ad-Hoc Networks. *Telecom* **2022**, *3*, 407–432. https://doi.org/10.3390/telecom3030022

Academic Editor: Alexandros-Apostolos Boulogeorgos

Received: 10 May 2022
Accepted: 22 June 2022
Published: 29 June 2022

1. Introduction

A mobile ad hoc network (MANET) is a self-organizing and self-healing autonomous system that consists of a loosely connected set of self-configuring nodes connected via a wireless medium. MANETs have become one of the most prevalent areas of research in recent years because of the challenges they pose to the related protocols and are still regarded as an emerging technology that enables network users to interact without any central infrastructure regardless of their geographical location. That is why it is sometimes referred to as an infrastructure-less network [1]. MANETs have been an active area of research for the last few years and their growth is promoted by the urgent need to provide the users with networks that are convenient to use and manage [2]. MANETs are primarily useful in military and other tactical applications such as emergency rescue or exploration missions [3]. Their existence stretches back to the 1970s where they were used in the military, meaning that they are not a new concept, and their commercial success has been due to advances in wireless technology. Standards have been developed for routing in MANETs. The IETF (Internet Engineering Task Force) regulates the new group for MANETs. The IEEE standard 802.11 has contributed to research interest in MANETs [1]. What contributes to their continued growth is the making of smaller and faster devices and this makes MANETs the fastest growing networking area. The mobility of nodes, however, is rapid and unpredictable over time. MANETs, like all wireless networks, are more vulnerable to attacks and weaknesses as compared to wired networks. The limitations of the wireless medium become especially severe in multi-hop networks, where multimedia streams are

subject to the aggregate effect (e.g., packet drop and propagation delay) of each link along with a node-to-node path. The mobility of a node can cause frequent route breaks in MANETs. Updating routes can be time-consuming, thus packets might for shorter periods be lost in bursts since they are sent on non-working routes [4]. The network must be less prone to security attacks to achieve an acceptable QoS as security vulnerabilities reduce the QoS of MANETs.

The biggest challenge in MANETs is the lack of a robust security solution that can protect them from routing attacks. The design solutions needed are not supposed to cause resource constraints such as bandwidth and battery power. The "self-organizing" functionality of MANETs can be their greatest advantage as well as their security weakness as it leaves room open for both passive and active attacks. There are many vulnerabilities poised by MANETs of which some are caused by their dynamic topology, limited power supply, limited bandwidth, and scalability. Existing architectures adopted from wired networks cannot be applied directly to MANETs because of the above vulnerabilities, hence there is need to design a robust security framework that will be solely implemented on MANETs. Any QoS-aware security framework to be developed must seek to achieve security goals, i.e., availability, integrity, anonymity, authentication, and confidentiality. To achieve a secure and QoS-aware network, a customed-designed security protocol must be implemented [4]. The problem with MANETs is that they are not well known for having combined solutions to security and QoS-based because of their dynamic nature. Therefore, this work focuses on sinkhole attacks through protocol modification to achieve good security detail without compromising the QoS in real-time.

The leading contribution of this paper to the existing body of knowledge in MANETs is bridging the gap between security and QoS by addressing the challenges in sinkhole attacks which affect not only the security of the network but also its performance. Therefore, the paper seeks to present a QoS-aware framework that is designed and evaluated to directly deal with sinkhole attacks in MANETs. The work ultimately contributes to QoS improvement of video streaming, security, and network resource optimization. Thus, the framework as presented in this paper seeks to monitor, control, manage, and evaluate QoS without compromising network security.

The remainder of this paper is organized as follows: Section 2 discusses mechanisms that have been implemented in literature specifically addressing sinkhole attacks in MANETs, Section 3 presents the proposed framework in detail including the sinkhole attack implementation and the counter-attack intrusion detection system.

2. Related Literature

In recent years, many schemes have been proposed to enhance the performance and security of MANETs. In these schemes, different attacks that infiltrate the network are presented and their effect on performance. Though, these schemes paid some attention to the individual problems presented against QoS and that of security, none of the schemes focused on addressing the challenges that exist on both MANET requirements. We categorize the schemes into the following categories.

2.1. Cross-Layer Scheme

The authors in [5] proposed a novel cross-layer cooperative scheme for detecting blackhole attacks targeting the QoS of OLSR in VANETs. They implement this using QoS-OLSR, a protocol that relies mainly on MPRs that are responsible for all routing in the network. Blackhole attacks target MPRs where packets are intentionally dropped to cause DoS. The authors proposed two mechanisms that allow the exchange of information between layers. The first scheme utilized information among network and physical layers while the second scheme was linked with the physical, MAC, and network layers to enable reliable and efficient detection of packets in transit. In the physical layer, the detection technique assigned each legitimate user a signature that is multiplied by the message, and each node used a maximum likelihood approach to detect the legitimacy of the message.

In the MAC layer, the detection technique monitored the number of RTS/CTS (request to send/clear to send) requests among neighbor nodes. The cooperative watchdog technique is implemented at the network layer to monitor the packets exchanged among neighbors. The cross-layer technique gave enhanced detection results and minimized the false alarm rate as compared to other state-of-the-art detection schemes. The proposed scheme was implemented for blackhole attacks and there is no evidence of its effectiveness on other attacks in MANETs. Thus, this paper implements an intruder detection system that includes a pathrater and a watchdog between MAC and network layers. At the application layer, the AES algorithm is implemented as an additional security measure.

2.2. Watchdog and Pathrater Schemes

Sharma et al. [6] used an intrusion detection system AODV and a watchdog AODV scheme to simulate a blackhole attack in MANETs. They saw security as a combination of procedures, processes, and systems used to ensure authentication, integrity, access control, nonrepudiation, and confidentiality. Their focus on security was based on the growing demand to support live streaming traffic military and civilian applications. AODV had a decreased performance in simulation with a blackhole attack. Their result shows that a watchdog IDS trust management solution for preventing blackhole attack was better than other proposed solutions. The watchdog had the advantage of listening to the nodes within the network transmission range and constantly monitoring packet delivery amongst the nodes. If a packet is not delivered within a specific time range or is forwarded but now altered by its neighbor, then the neighbor is deemed as misbehaving and the watchdog will declare the node as a gray hole node and exclude it from the path of sending packets [6].

In contrast to the IDS AODV, the authors in [7] proposed an infiltrator detection system for node isolation attacks (DoS) against OLSR. Their solution is based on identifying the false information with a HELLO message using the infiltrator identification system. The performance of the proposed algorithm is tested in terms of network throughput and packet delivery ratio. The proposed algorithm outperformed the standard version of OLSR on the tested parameters. Their algorithm was only tested for a 3 mobile node-scenario and two QoS measures, thus its effectiveness cannot be judged based on 3 nodes analysis in a network scenario. Our research is evaluated in different node density environments to validate its robustness.

Khan et al. [8] proposed a multi-attribute trust framework for MANETs (MATF). The framework was designed to minimize bootstrapping time and also to tackle selective behavior. It was also designed to enhance the security of MANETs by enabling a node to identify and then quarantine malicious nodes from the routing paths. The proposed security framework was not only designed to detect malicious nodes and in time, but also to decrease the number of false positives. They used multi-watchdogs as a measure to achieve this [8]. Their work was based on a reactive routing protocol, AODV, for the detection of wormhole attacks and not OLSR. This aspect of QoS was not a factor in their evaluation process and it is central to this research work.

In their work, Monica et al. [9] analyzed, simulated, and performed a comparative study of three different attacks based on many parameters. These attacks were denial of service (DoS), black hole attacks, and wormhole for wireless ad hoc networks based on performance. Their comparison was based on packet delivery ratio (PDR), end-to-end delay, and throughput. The attacks had different properties, different behavioral patterns of attacks in different environments primarily because of their different designs. Their experiment concluded that the rate of data loss is less in wormhole than compared to the other two attacks. The best way to counterattack is to first know the behavioral patterns and properties of the attack itself, and this is what the authors did. This work, however, was not evaluated for sinkhole attacks.

In their work, Deebak et al. [10] proposed a network routing and intrusion monitoring scheme for IoT-WSNs through a secured routing and intrusion detection models using OLSR and AOMDV protocols. Their scheme works periodically and reactively.

Sekaran et al. [11] developed an AI framework for IoT-based sensor networks that is capable of detecting DOS, misrouting, and identity theft using special space raising devices.

In their research, Raghav et al. [12] developed a linear and static trust-based routing scheme that can detect varieties of attack using sensor nodes.

Table 1 shows a summary of related work, their contribution and limitations of their study in comparison to our research.

Table 1. Related works-contributions and limitations.

Article	Contributions	Limitations
Baiad, et al. [5]	Proposed a novel cross-layer cooperative scheme for detecting blackhole attacks targeting the QoS of OLSR in VANETs. They implemented an IDS that includes a pathrater and a watchdog between MAC and network layers. At the application layer, the AES algorithm is implemented as an additional security measure.	Their techniques focused on blackhole attack, unlike our research, which deals with sinkhole attack. Both studies used a cross-layer approach though our research involves more layers' interactions and more sensor node security was implemented in our research.
Sharma and Mahajan [6]	The authors used an intrusion detection system AODV and a watchdog AODV scheme to simulate a blackhole attack in MANETs and its effects	Although IDS was implemented, it was based on trust network state information. The IDS was not tested for sinkhole attacks but blackhole attacks.
Sahu, Rizvi, and Kapoor [7]	Proposed an infiltrator detection system for node isolation attacks (DoS) against OLSR. The proposed solution is based on identifying the false information with HELLO message using the infiltrator identification system.	Only tested for 3 mobile node-scenario and two QoS measures. Three node-scenario is quite inadequate when a complex network is involved with several nodes. In our research, we experimented with 16, 49, and 100 nodes.
Khan et al. [8]	Proposed a multi-attribute trust framework for MANETs (MATF). The framework was designed to minimize bootstrapping time and also to tackle selective behavior.	The aspect of QoS was not a factor in their evaluation process and it is central to our research work.
Monica et al. [9]	Their work analyzed, simulated, and performed a comparative study of three different attacks, namely, DoS, black hole, and wormhole based on many parameters. The comparison was for their packet delivery ratio (PDR), end-to-end delay, and throughput.	This work was however not evaluated for sinkhole attacks.
Deebak et al. [10]	They developed a secured routing and intrusion detection model using OLSR and AOMDV protocols. Their scheme works periodically and reactively.	Although it is for a complex network, there is absence of node coordination.
Sekaran et al. [11]	The authors developed an AI framework for IoT-based sensor networks that is capable of detecting DOS, misrouting, and identifying theft using special space raising devices.	This nature of the network creates data and network vulnerability and it is not specific to sinkhole attack.
Raghav et al. [12]	They developed a linear and static trust-based routing scheme that can detect varieties of attack using sensor nodes.	Their scheme is incapable of detecting attacks on multi-hop networks and changes in temperatures of sensors nodes.

3. Material and Methods

To successfully achieve this work's objectives and the main goal, the following research techniques are applied:

3.1. State-of-the-Art Literature Review

A state-of-the-art literature review was conducted to establish the research gap in MANETs from literature. The methodology used is the systematic review according to Linares-Espinos et al. [13], it is the critical and verifiable summary arising from the various

publications that address the aim of this research done by various researchers globally related to the field of study and various literature was consulted.

3.2. Simulation

The simulation approach involves the creation of an artificial environment within which relevant data can be generated. This is done in a controlled environment, and it permits the observation of a system's dynamic behaviors. One of the purposes of simulation is to perform real decision-making or diagnostic tasks. Decisions taken through analytical formulations are less preferred compared to simulations because they do not provide certainty [14,15]. Network simulators best describe and represent the state of the network. These include nodes, links, switches, hubs, and routers. In modern-day simulators, they are either graphical user interface (GUI) driven or command line interface (CLI) driven. Simulation is mainly used in performance analysis, comparison, or even management and for determining how a network would behave in a real-life situation. The generated result of the simulation helps in identifying the performance attributes of the network.

3.3. Simulation Implemented in this Research

Discrete event simulation and experimentation using Netsim are implemented to adequately characterize variables, corresponding states, and events that change the value of these variable states in some rule-oriented but stochastic manner. The entities are the different components in the proposed framework. A network design [1] that refers to the planning of the implementation of scenarios is also carried out. Figure 1 shows elements of a discrete event simulation as characterized by Law et al. [16].

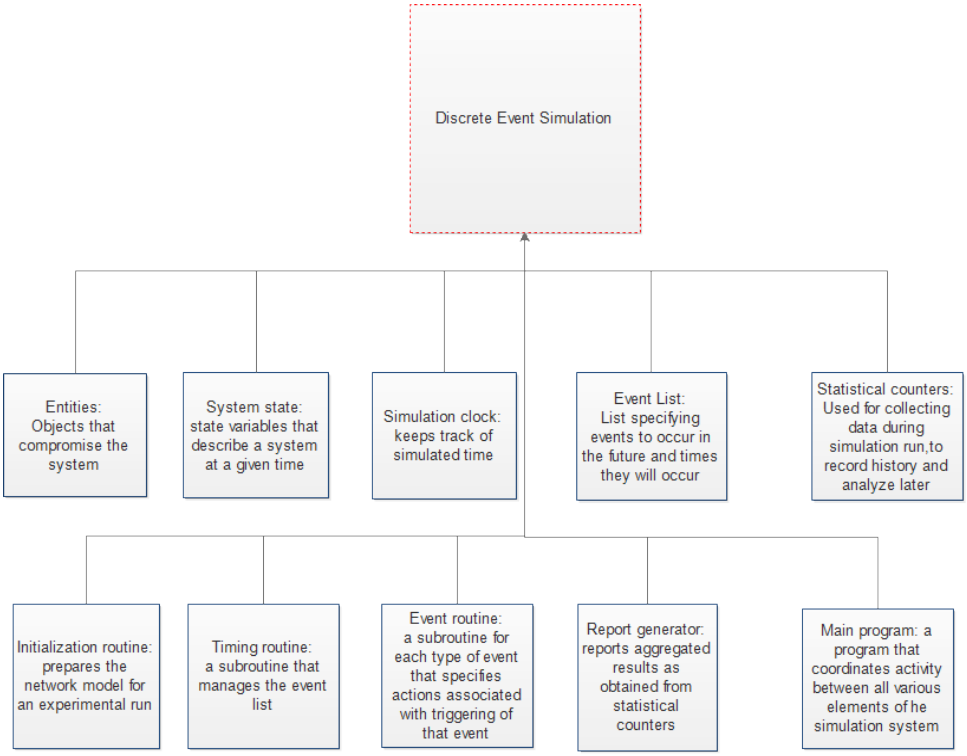

Figure 1. Elements of discrete-event simulation.

3.3.1. Simulation Environment

The simulations are carried out on a Windows 10 Professional operating system (desktop) with 3.40 GHz processor speed, 4.00 GB RAM, and 64-bit operating system.

3.3.2. Software for Simulation and Justification for Selection

NetSim is developed by an Indian organization called TETCOS and was initially developed by BOSON. The connection is made possible by a NetSim cloud server that remotely connects client nodes. Our workstation connects through the server's VPN. NetSim enables users to virtually create a network along with its components, such as devices, links, and applications, etc. One other feature NetSim provides is a graphical editor interface that is used to build models for different network entities [17].

In this study, vehicles were represented by mobile wireless LAN workstations that move randomly. The software is very efficient in the study of application-based protocols. Its potential to generate several traffic types adds another advantage. Its C source code enables us to modify the protocol code by adding features that are unique to this research and thus extend knowledge in the study of sinkhole attacks. The resulting dashboard is easy to read and analyze with graphs and tables. Our target within the tool is its ability to redefine the source code of the protocol for improved performance and security.

4. QoS-Aware Security Framework: Proposed Scheme

The proposed framework follows a cross-layered structure. Cross-layer design in the context of MANETs refers to the sharing of information among the different layers for the efficient use of available network resources and promoting adaptability. The main reason for cross-layer design is to sustain the functionalities found in the original layers and additionally allowing interaction, joint optimization, and coordination of cross-layer protocols. Each layer in the design is characterized by having unique parameters to help them determine the best adaptation rules and control knobs for the current network status. The optimization aspect of the design comes with the customization of variables and constraints from other multiple layers. The design speaks not only towards an isolated problem but also other parts of the system that may encounter unintended effects while the problem is being resolved. This is the reason why security and QoS come as a balanced scale because the complexity of the design might affect other elements of the system and then resulting in poor transmission of packets or a weakened system. The highlight of the model is the intrusion detection system and is embedded in the different layers of the model. The uniqueness of the proposed framework is that it incorporates QoS and security implementation in more than one functional layer.

4.1. Motivation for Framework Design

The key motivation behind the design of the framework and hence this paper, is the rise of new problems in MANETs that includes the possibility of opportunistic communication by malicious devices flooding a network to consume resources thereby causing denial of services to legitimate nodes, stealing information for malicious purposes and diverting network resource flow against the wish of administration. This latter part can be likened to taking over control of a network. This research work introduces new interfaces, such as the lightweight IDS in the cross-layer framework, which is something that would have been otherwise close to impossible within the traditional layered architecture. Another motivation is the introduction of an upward-downward and downward-upward flow within layers as seen in Figure 2. The key design features of the proposed framework are geared towards fostering a method in which the layers can communicate with each other without compromising the quality and integrity of the data. According to Thota [18], the cross-layer design is known for its substantial gain in QoS, output, and efficiency. The traditional layered architecture does not allow direct communication between layers, but with this design, adjacent layers can communicate, and every layer works on its variables. The cross-layer design of the framework also allows new abstractions within the communication infrastructure between layers. Phakathi et al. [19], in a work-in-progress paper, proposed the QoS framework below and this work evaluates its efficacy.

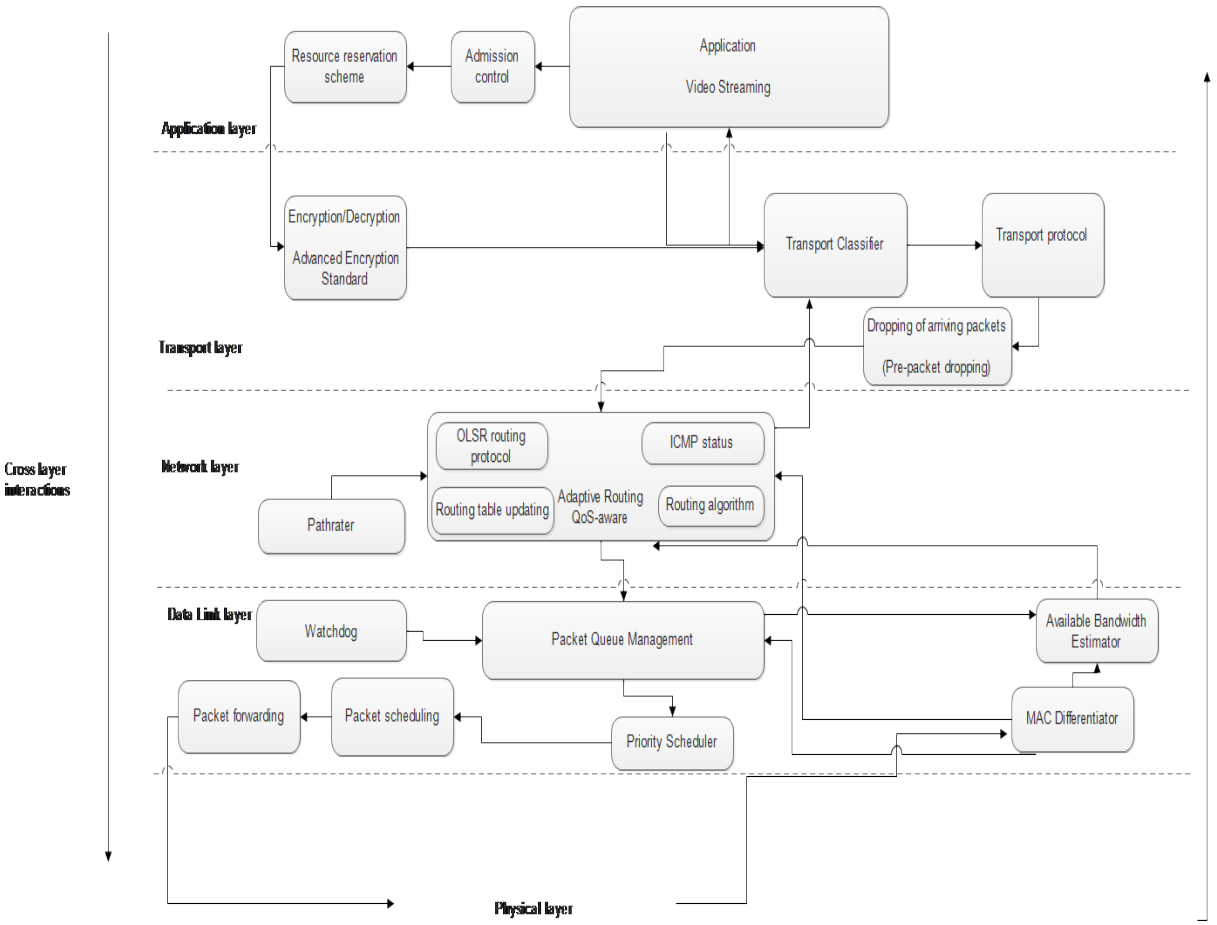

Figure 2. QoS-aware security framework [19].

4.2. Admission Control

The admission control mechanism deals with QoS violations because of false admissions, changes in node neighborhood, node mobility, or signal propagation conditions. This component is responsible for admitting data sessions with QoS requirements that can be satisfied without violating previous sessions. The collection of information about available network resources is done to make admission decisions. The admission control protocol establishes if there are any nodes or node pairs with the necessary resources available. This action is completed during route discovery or even after the routes have been discovered. If no route completes the application's QoS requirements, then that data session is rejected. This is the reason why it is important that during this phase, information about the resources available with neighboring nodes is collected. The key responsibility of this component is the management of data sessions with QoS assurance violations and having those sessions re-admitted in an event when there was an interruption during its packet sending phase and maintenance of state information [18,20].

4.3. Resource Reservation Scheme

In MANETs, the bandwidth is shared by nodes among neighboring links, and its usage is managed by the routing protocol implemented. The resource reservation scheme (RRS) can be implemented through QoS routing means or through QoS MAC strategies. QoS routing schemes can be proactive or reactive. These are the known routing schemes such as AODV, DSR, and OLSR. The RRS is responsible for providing an acceptable level of QoS for high-priority sessions during routing by reserving resource usage at all immediate nodes along with the source to destination route. This is done in order for applications, such as video or any form of multimedia, to have enough bandwidth during the transmission phase

to satisfy an acceptable level of QoS requirements. The routing session's QoS is dependent on the bandwidth shared among the neighboring links under the same medium and nodes within the interference range of the route. The reason why an RRS is implemented in this research is that state information on existing transmissions is protected within the route. In OLSR, the available resources are detected in advance as part of its proactive routing process, and all state information is constantly updated at the routing table. A secondary route is selected in an event where the primary route cannot satisfy QoS requirements because of link failure or insufficient resources. RRS is implemented to alleviate the impact of latency that comes with the routing process through the transmission of data from any source node because all the possible routes would have been identified proactively [21].

4.4. Transport Protocol

The framework is designed to accommodate both Transmission Control Protocol (TCP) and User Datagram Protocol (UDP) traffic. The video streaming implemented in this work uses UDP as the preferred transport protocol. UDP uses the Internet Protocol (IP) to obtain datagrams from one node to another. It simplifies the need to have an established connection before data transmission. Although TCP is reliable and has assured delivery of packets, it is not favorable for video streaming and live conferencing applications since UDP embraces high performance and allows packets to be dropped instead of trying to process the delayed packets. Uneven delays between sections of a received message are not tolerated by real-time applications. UDP also saves bandwidth usage and is not latency-prone because there is no error checking.

4.5. QoS-Aware Adaptive Routing

The routing protocol, OLSR, facilitates all routing in the network. OLSR works by using a link-state algorithm that constantly works with the routing table. The routing table is flexible to adaptive routing as the topology is dynamic and needs constant updating. The packets will use an optimal path as directed by the MPRs to the destination. Nodes can exchange updates and route table information. Adaptive routing allows the routing path to change over time as the topology in which the nodes operate is ever-changing. Another role-player component in our framework is packet switching. Packet switching is regarded as a higher-level decision-making entity responsible for driving packets from source to destination. The routing protocol, OLSR's flowchart in the proposed framework is shown in Figure 2 where we proposed a few changes towards some of the traditional operations of the protocol to accommodate QoS and increase efficiency in terms of performance. Figure 3 illustrates the flowchart of our modified OLSR protocol. This also includes a proposal of how performance is optimized when there is a malicious node in the network. MPRs still play a critical role in terms of propagating TC messages through to the routing table. An unauthenticated node will be regarded as a malicious node and will be isolated from the network. At first, the node will be sent a fake HELLO message, then selected as an MPR, and then blacklisted at the routing table. After routing, packets are sent to the data link layer where there is packet queue management, MAC differentiator, and available bandwidth estimator. Under packet queue management, there is packet scheduling, priority scheduling, and packet forwarding [22].

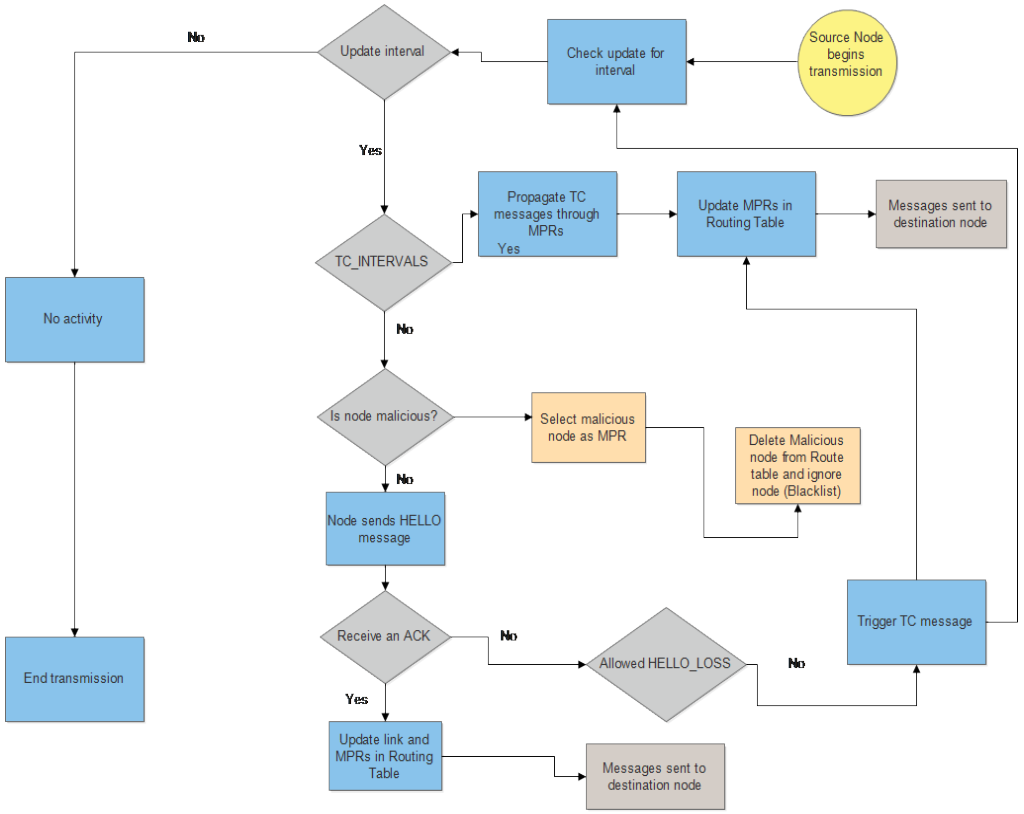

Figure 3. Modified OLSR flow chart [22].

4.6. Packet Queue Management

Admission control is responsible for preventing network overload to guarantee QoS for already admitted sessions and is deployed where a new session is either admitted when resources are available or rejected. Packet queue management is responsible for anticipating signal congestion before occurrence using scheduling means to determine the order of flow of different packets. The packet scheduler is responsible for providing the actual memory used for storing packets and providing a firewall used against malicious nodes whose intent is to selfishly use up network resources. The packet scheduler is responsible for determining the order in which packets are processed at the node level and transmitted over a link. This framework utilizes absolute priority queuing to assist in packet queue management. In absolute priority queuing, the packet scheduler scans and rates queues according to high, medium, and low priority. High priority is most preferred during the forwarding of packets from queues, and this means that packets with lower priority obtain processes as soon as higher priority queues are empty. In terms of resources, packets in high-priority queues will receive the best service, and queues with a lower priority will have to be satisfied with the remaining resources. Once the packet is sent, the scheduler restarts the process again with high priority first. This mechanism is very important in terms of delay of critical data and bandwidth usage for video streaming. The packets are forwarded with minimal delay if the high-priority packets do not exceed the outgoing traffic.

4.7. Sinkhole Attack

A sinkhole attack is orchestrated by modifying Suman's [23] source code. The modified sections are highlighted in all figures and algorithms, respectively. This section contributes to the study of sinkhole attacks and RFC 3626 OLSR as defined by the IETF standards process of 10 November 2008. Sinkhole attack is launched against the network to disorganize the routing flow and modify packet information or even drop packets just to make the network inefficient in terms of performance. This research implements a sinkhole attack on

the OLSR routing protocol. An OLSR MALICIOUS.c file containing a series of functions is added to the ZRP/OLSR project. The malicious node intercepts HELLO PACKETS sent by the source node by disguising itself as an MPR, then adds a fake route entry, and thus flooding the network with fake link-state information that is critical in determining the next hop to the destination. The malicious file added to the ZRP project has the following notable source headers and functions:

4.7.1. fn_NetSim_OLSR_MaliciousNode ()

As shown in Figure 4, the function is placed to check if the defined node is malicious or not. If the node is not malicious, then the function invokes malicious behavior. Any node or device ID within the network can be set as malicious with the function:

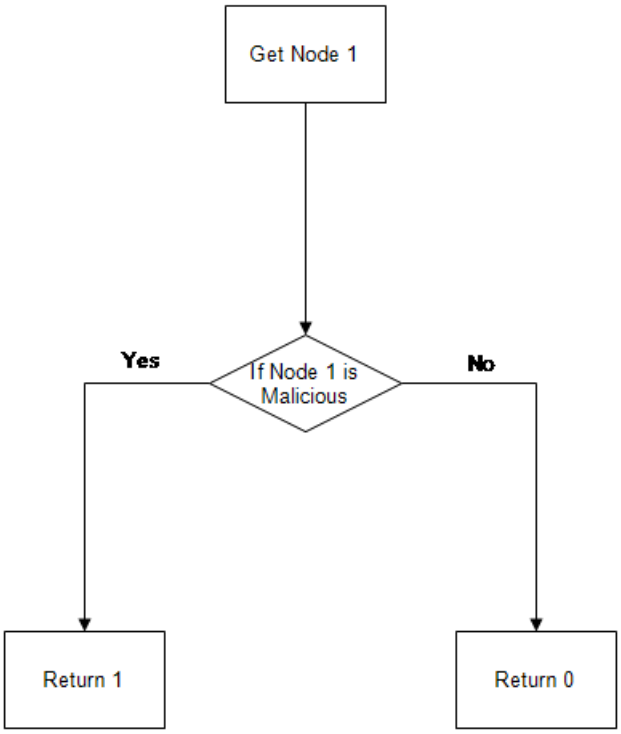

Figure 4. Algorithm to check malicious nodes.

4.7.2. fn_NetSim_Malicious1_OLSR_PopulateMPRSelectorSet ()

In Figure 5, the function MPR selector set of a node, n, is populated by the main addresses of the nodes which have selected n as MPR. The selection of MPRs is sent through by HELLO messages intercepted by the malicious node in the function:

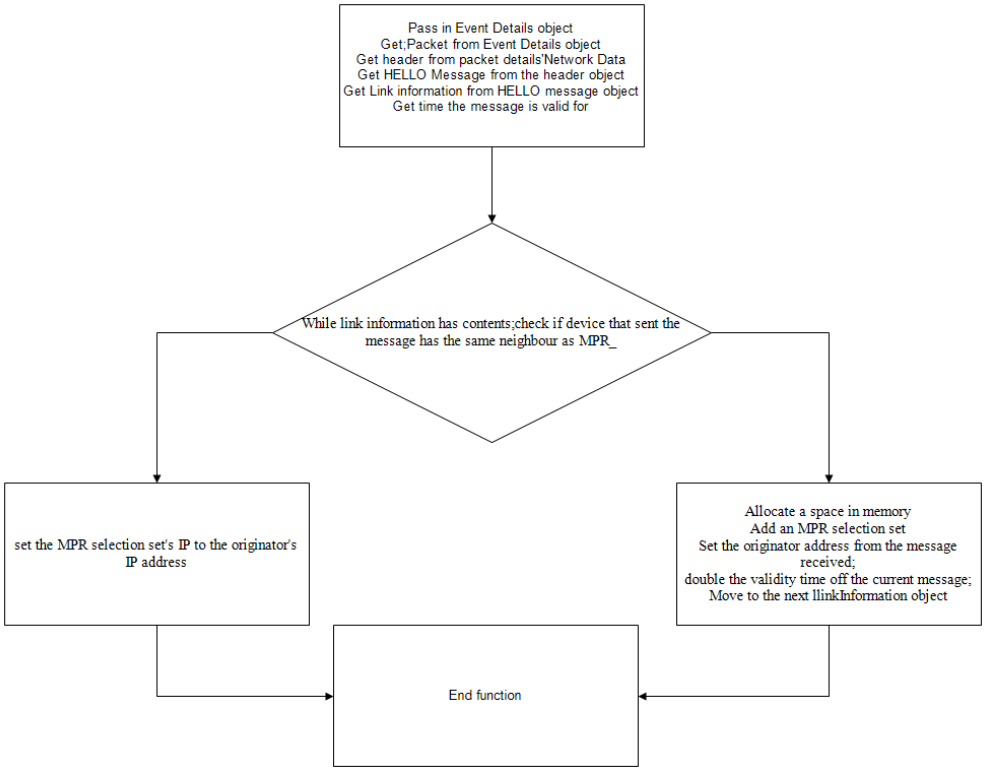

Figure 5. Malicious node infiltrating MPRs.

4.7.3. fn_NetSim_OLSR_MaliciousRouteAddToCache ()

The function as shown in Figure 6 is used by the malicious node that has selected itself as an MPR to add a fake route entry into the route cache before flooding the network with fake link-state information.

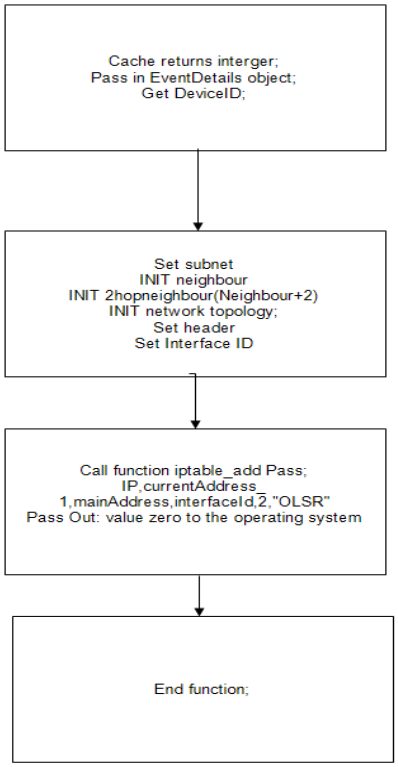

Figure 6. Malicious route added to the cache.

4.7.4. fn_NetSim_OLSR_MaliciousProcessSourceRouteOption ()

The function in Figure 7 is used to drop packets instead of the node forwarding them to the next-hop:

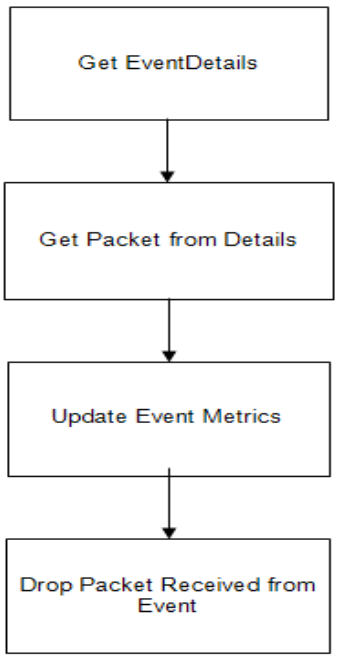

Figure 7. Malicious node dropping packets.

4.8. Intrusion Detection System

An intrusion detection system is implemented by modifying Suman's [23] source code to detect and remove a sinkhole attack from the network. The modified sections are explained and highlighted in all figures. According to Kumat and Venugopalan [24], intrusion detection is the process of capturing and analyzing significant events in a system for the possible presence of an intrusion. Intrusion detection systems aim to prevent unauthorized access to the network and responding to the malicious activity that may be caused by the intruder. Researchers have proposed various approaches to implementing an IDS based on wired networks which are quite different because MANET is a wireless platform.

The implemented standalone intrusion detection system is customized to function in the two layers of the TCP/IP model, namely, the network and data link layer running the OLSR routing protocol. It uses two mechanisms to achieve its goal and is intended to work for UDP traffic because TCP requires ACKs from the destination. The watchdog analyzes every node that is in transmission proximity. It compares two variables (threshold value and counter) before detecting that the node is showing malicious behavior. If a node withholds a packet, the watchdog algorithm counts the downtime for the delay of the node to forward the message and compares this with the threshold value. The delay must be less than the threshold value for the node to be considered cooperative or else the node will be considered malicious. The watchdog will also check the packet itself for alterations and possible modifications. If any node only accepts the watchdog node and does not forward, then the watchdog declares it as a sinkhole node and isolates that node from the path of packet transmission. A sinkhole node has an infinite ratio of receiving and sending data packets [25].

4.8.1. Watchdog

The watchdog mechanism is implemented in the WLAN IEEE802_11 project folder and is a key method for the detection of intrusions in the network. It is considered simplistic

and lightweight because it works with less overhead by using embedded local information handling capabilities. In this code (WATCHDOG_TIME) a watchdog timer is set for 2 s. The nodes wait for the duration of this timer to detect if the next-hop node is malicious or not by checking if it broadcasts the packet further to the destination node. The failure threshold is set at 5. The failure threshold is the number of times a packet is resent before declaring it as malicious.

4.8.2. Watchdog Code Flow

Once the _NETSIM_WATCHDOG_ is defined, the watchdog timer begins. When a packet is sent to a neighboring hop, the current node checks the watchdog timer duration to verify if the packet is forwarded to its destination. If the node is malicious, then it will not forward the packets it receives until the watchdog timer in the node expires. Once a packet is forwarded to the next-hop node, the current node checks for watchdog timer duration if the packet is getting forwarded further to the destination node or not. The packet is then sent until the watchdog timer expires and the failure threshold is reached. There is a counter that measures the frequency at which the watchdog timer expires and once the counter's value equates to the failure threshold then the next hop is marked by the current node as malicious. As highlighted in the QoS framework, malicious nodes are blacklisted and removed to maintain the integrity of the network. The function add_to_blacklist (NETSIM_ID, NETSIM_IPAddress) is responsible for adding the intruder's IP address to the blacklist. The function find_ip_in_blacklist (NETSIM_ID, NETSIM_IPAddress) returns true if the node is marked as blacklisted. The function verify_route_reply (NETSIM_ID, OLSR_HELLO_PACKET*) verifies the IP address obtained from the control packet to check if it is blacklisted or not. If the reply is not from the malicious node, then it returns true. The function "blacklist_found (NETSIM_ID, NETSIM_ID)" deletes the node's route entry from the cache when it is blacklisted [23].

4.9. Pathrater

The pathrater's function is to validate routes as shown in Figure 8. It runs in the network layer hence it is added to the pathrater project folder. In an event where a HELLO packet is processed, the function verifies the response in the route cache to check for the blacklisted node. If a blacklisted node is found, that route entry is deleted from the cache. The function "add_to_blacklist (NETSIM_ID, NETSIM_IPAddress)" is responsible for adding the IP address of the sinkhole node to a blacklist and it is noted as a possible intruder. The function "find_ip_in_blacklist (NETSIM_ID, NETSIM_IPAddress)" returns true if the suspected node is blacklisted. The function "add_to_blacklist (NETSIM_ID, NETSIM_IPAddress)" verifies the IP address obtained from the OLSR control packet considering whether it is blacklisted or not. It returns true if the OLSR packet is not from the sinkhole node. The function "blacklist_found (NETSIM_ID, NETSIM_ID)" removes the sinkhole node's route entry from the cache.

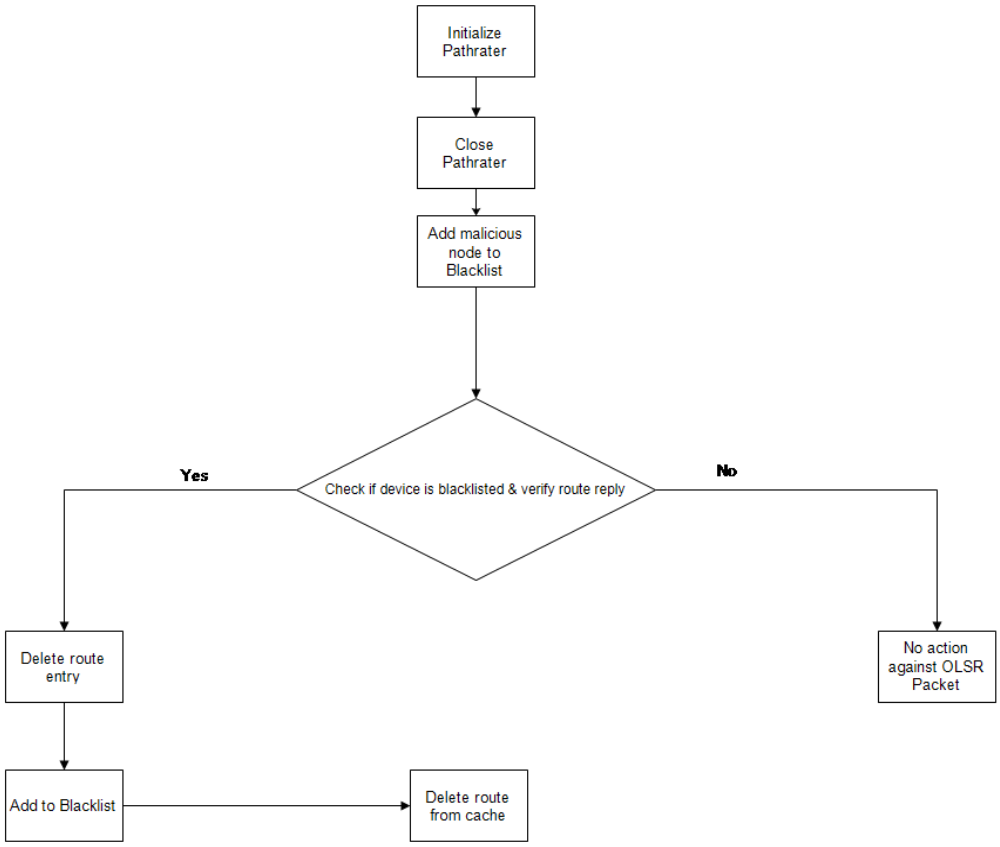

Figure 8. Pathrater algorithm.

4.10. Simulation Parameters

Table 2 shows the simulation environment that was prepared to create different scenarios that were used to obtain the best possible results. The parameters are set up to represent low, medium, and high density (fixed size with varying nodes) traffic. The sensors' capacity in terms of transmit distances, power sustainability, and speed of transmission are considered in setting these parameters.

Table 2. Simulation parameters.

Parameters	Value(s)
Simulator	NetSim Standard v12.1
Application protocols	OLSR
Grid length	1000 m × 1000 m
Simulation time	100 s
Traffic type	Video
QoS class	rtPS (real-time polling service)
Node movement model	Random waypoint
Trajectory	Random
Transport protocol	UDP
Speed	50 km/h
Refresh interval	2 s
Encryption algorithm	Advanced Encryption Standard
Node density	16, 49, 100

QoS Measures

The QoS measures used to ensure that quality is optimized in the network are throughput, delay, and jitter.

Throughput: It represents the number of bits forwarded from the MAC to higher layers in all nodes in the network; it is measured in bits per second. The throughput may also be referred to as the average number of packets successfully transmitted or received per second. This work focused on the application throughput which is the total user data sent to the intended destination per second. The formula to calculate throughput is given as:

$$\text{Throughput} = \text{Pd}/\text{T} \tag{1}$$

where Pd is the number of packets delivered and T is the time in seconds.

Delay: This is normally the time taken for one packet to be transmitted from the source node to the destination node. This performance metric evaluates the routing protocol's effectiveness in the use of network resources. Delay may be caused by several obstacles including transfer time, buffering during discovery latency, interference queue, and propagation. The formula to calculate delay is given as:

$$\text{Delay} = \text{Trx} - \text{Tst} \tag{2}$$

where Trx is the time the packet is received and Tst is the time the packet is sent.

Jitter: This is the variation in time between route changes and data packets arriving. The variation may be caused by internal sources, such as data transmission errors, the presence of a malicious node, and network congestion. It usually affects the audio quality of the video if its level is high. The formula to calculate jitter is given as:

$$\text{Jitter} = \text{Dt} - \text{Dp} \tag{3}$$

where Dt is the transmission delay of the current packet and Dp is the transmission delay of the previous packet.

5. Results and Discussion

5.1. Simulation

Performance evaluation under low-density nodes scenario (16 nodes) The results shown in Figure 9 represent the network topology with 16 nodes having two malicious nodes added to the network. The three metrics used in the performance analysis are hereby discussed:

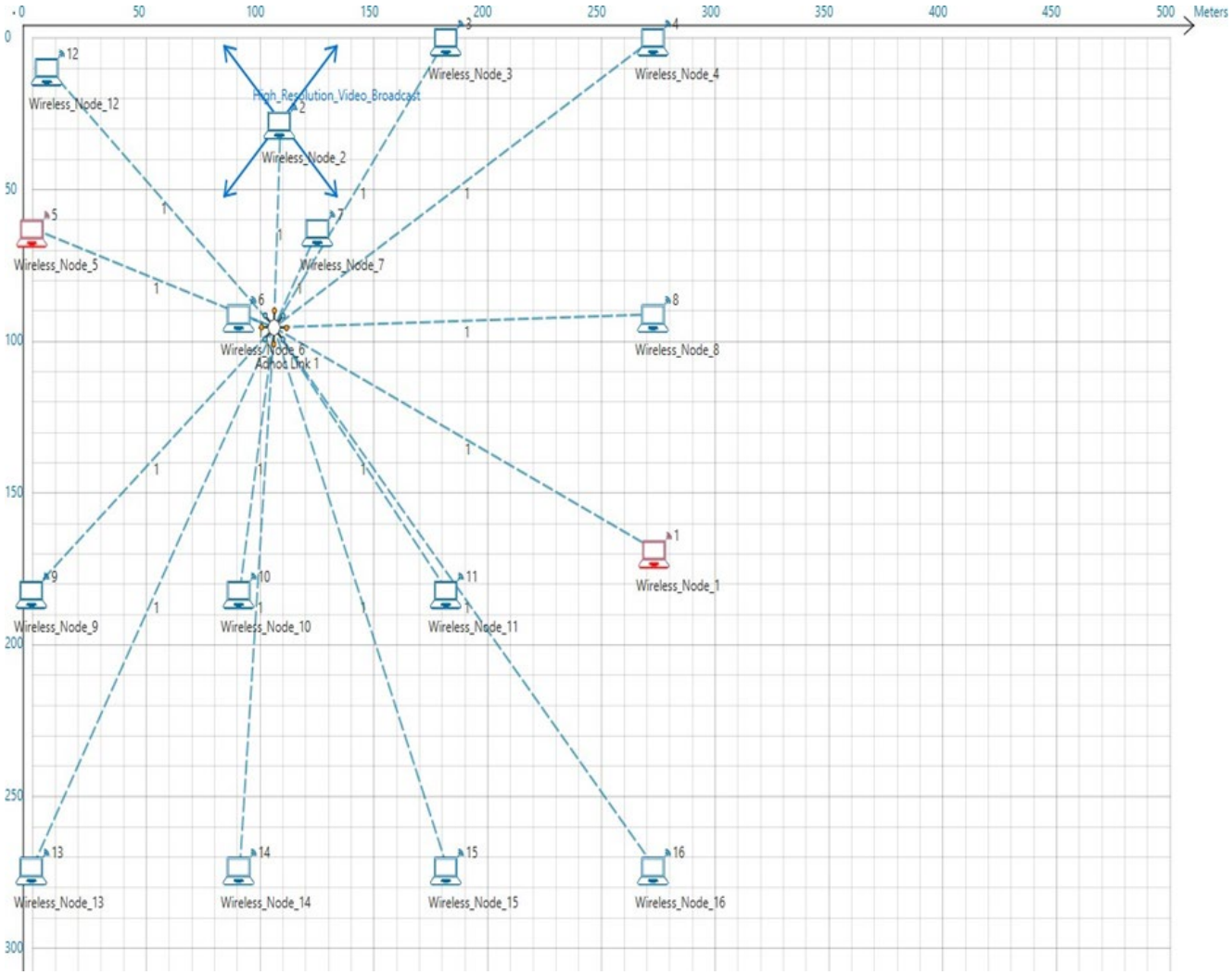

Figure 9. Network model of 16 nodes with two sinkhole nodes added.

5.1.1. Throughput Performance Analysis

Figure 10 shows the throughput for both sinkhole and framework scenarios. The y-axis shows the throughput in megabits/s (Mbps) and x-axis shows the time in milliseconds (ms). In the sinkhole scenario, there was a 2000 ms lag time at which the application starts. The next 14,000 ms showed a positive hike up to 0.74 Mbps in throughput followed by a steady increase until the end of the simulation, unlike the framework scenario which grew with time. The highest network throughput achieved by the framework was 0.77 Mbps at 68,806.37 ms of simulation time. Although the framework scenario's throughput grew with time, it was not by a big margin, nevertheless it was better than that of the malicious scenario from 2000 ms moving forward.

5.1.2. Delay Performance Analysis

The proposed QoS framework had the lowest recorded value in terms of network delay, and as expected, the malicious scenario produced the highest level of delay for any 16-node scenario as shown in Figure 11. The delay in the sinkhole scenario was caused by the time the attack exceeds the failure threshold value before forwarding the packets to the rest of the network. The framework significantly reversed this action by a significant margin meaning that the intrusion was identified, blacklisted, and alternative routes that guarantee an acceptable level of QoS were chosen. In the framework scenario, the first node had a delay of 12,369.96 microseconds. Node 3 has an elevated delay value of 13,739.96 microseconds.

The delay slightly fluctuated between 13,663.037 and 13,739.96 microseconds for nodes 7 and 12. The rest of the nodes maintained a constant delay of 12,017.41 microseconds. The delay was relatively low as compared to that of the sinkhole scenario.

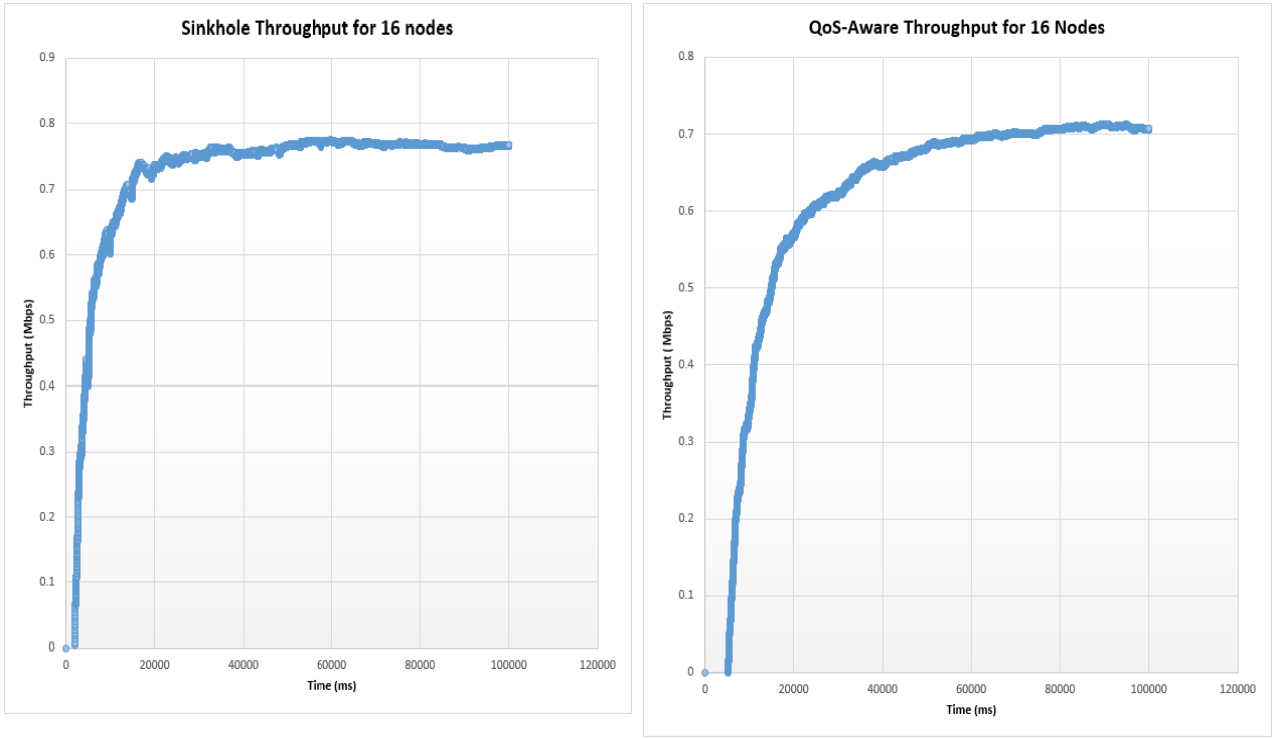

Figure 10. Throughput for sinkhole and framework scenarios.

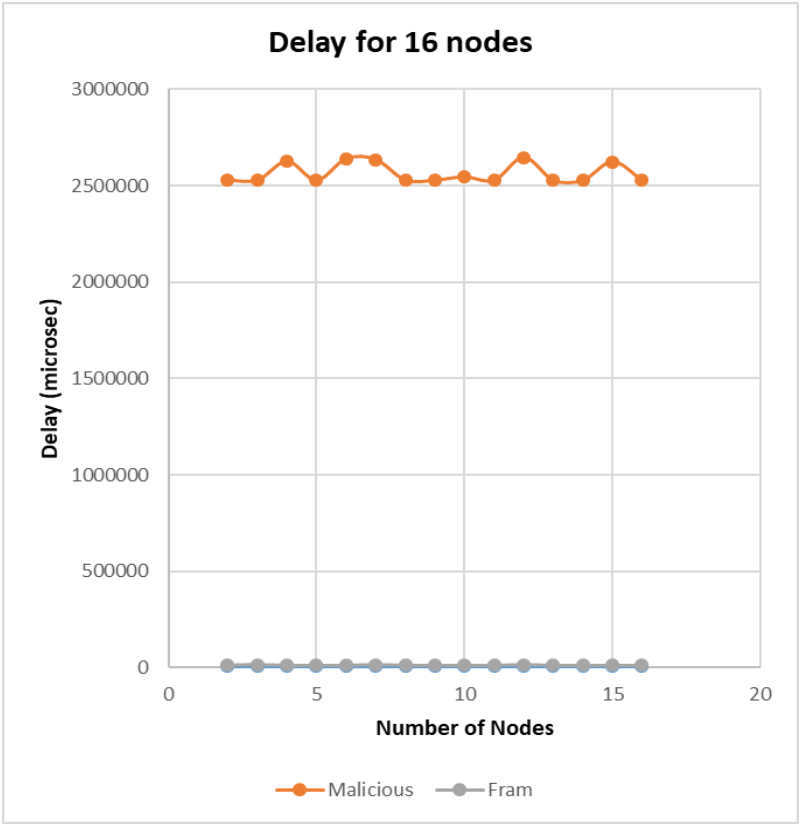

Figure 11. Delay for 16 nodes.

5.1.3. Jitter Performance Analysis

The proposed QoS framework outperformed the malicious scenario again in terms of jitter recording an average of about 6500 microseconds as compared to the malicious scenario which had margins of about 35,000 microseconds. In the sinkhole scenario, the first node had a delay of 26,543.9 microseconds. Nodes 2–5 showed a constant jitter of 26,468.72 microseconds but node 6 and node 12 both had 26,400.87 microseconds. This was an expected margin based on the malicious activity within this broadcast session. As shown in Figure 12, the framework significantly lowered the levels of jitter among all the nodes. The framework proved to be effective in low-density node scenarios.

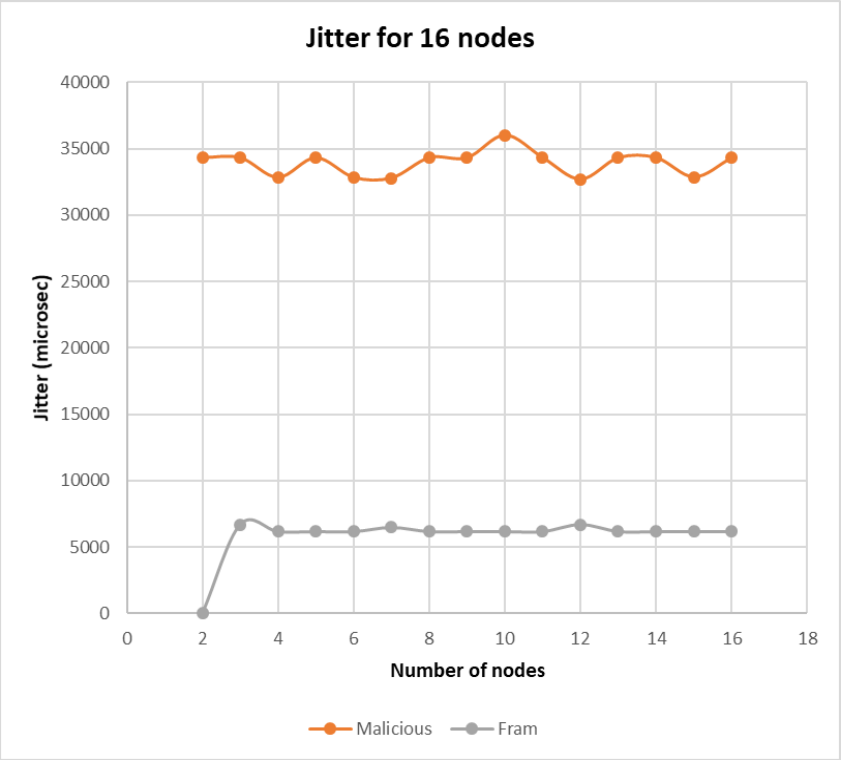

Figure 12. Jitter for 16 nodes.

5.2. Simulation 2: Performance Evaluation under Average Density Nodes Scenario (49 Nodes)

The second simulation results showed the performance of the network with 49 nodes exposed to a sinkhole attack and the proposed framework. The proposed framework still outperformed the malicious node scenarios based on the given throughput, but the result was not satisfactory as there were high margins in terms of delay and jitter. Figure 13 shows the implemented network model with 49 mobile nodes and two marked malicious nodes. The experiment was run 10 times for validation leading to results shown in Figures 14–16.

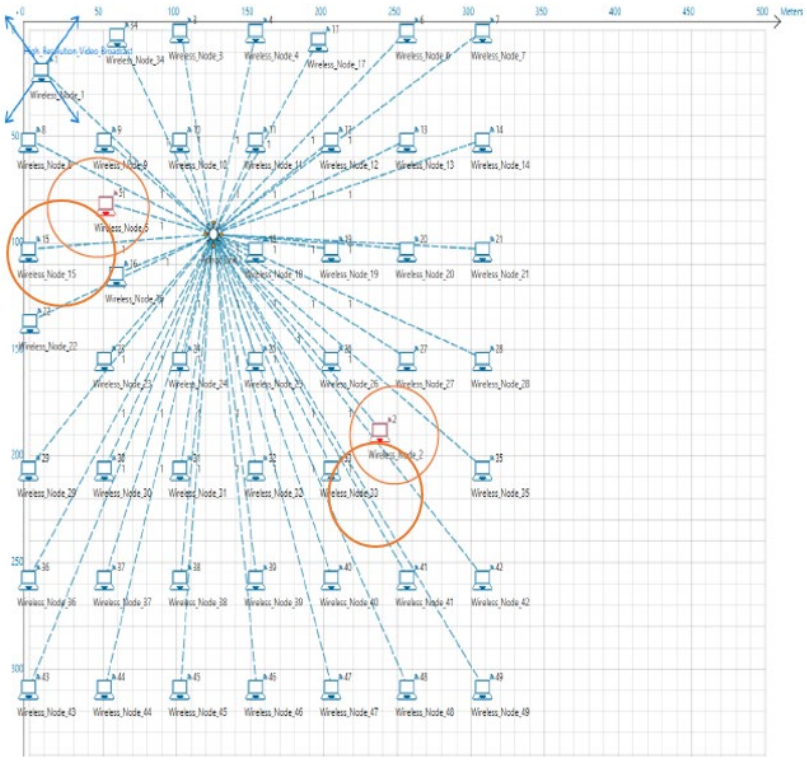

Figure 13. Network model of 49 nodes with two sinkhole nodes added.

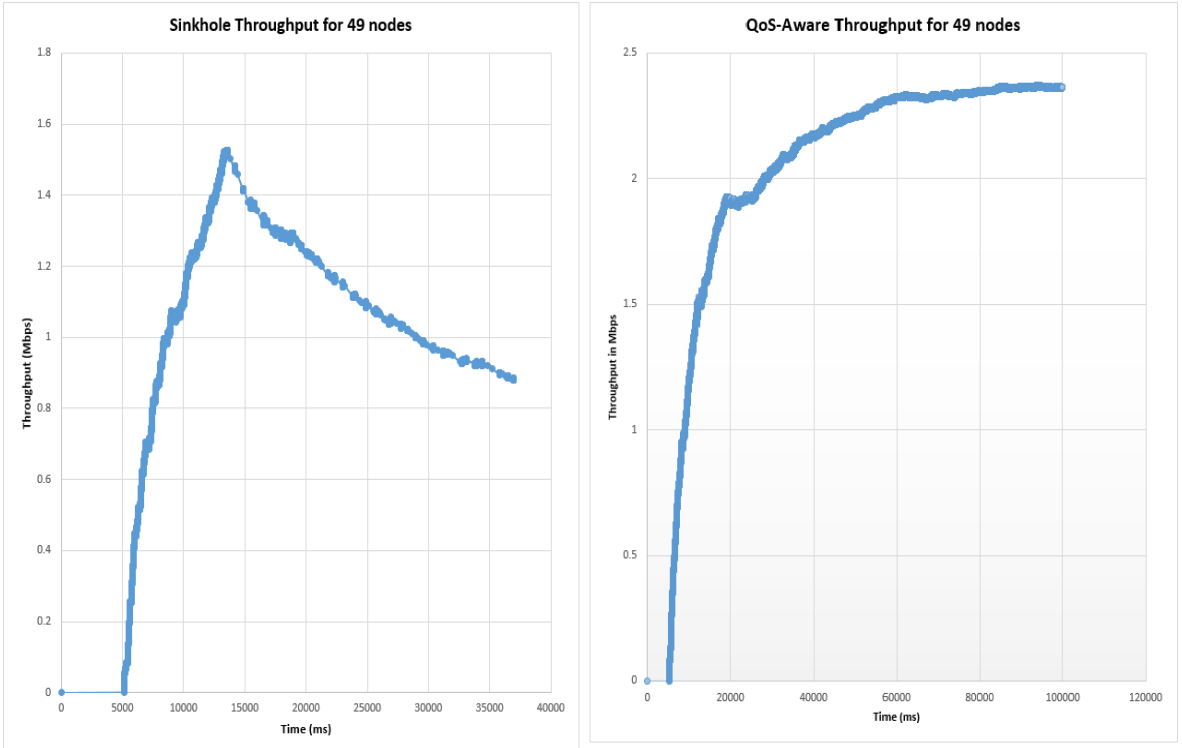

Figure 14. Throughput for sinkhole and framework scenarios.

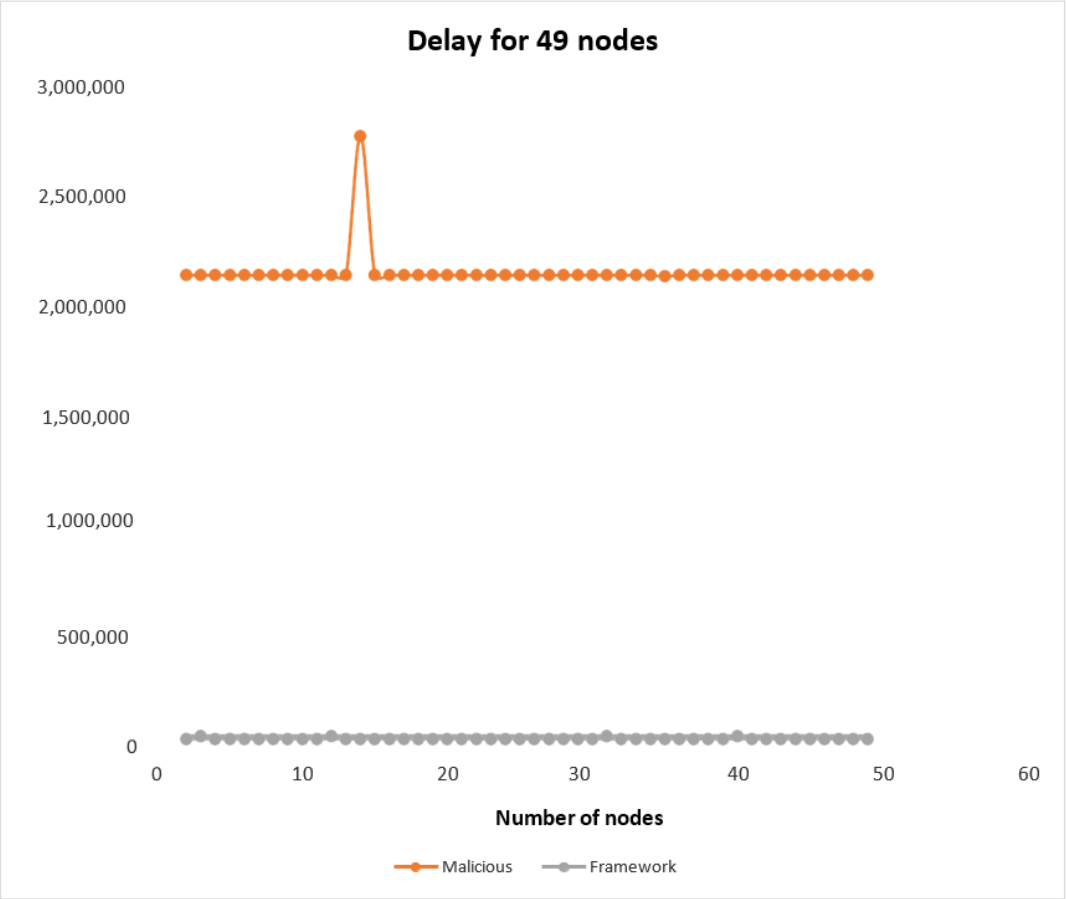

Figure 15. Delay for 49 nodes.

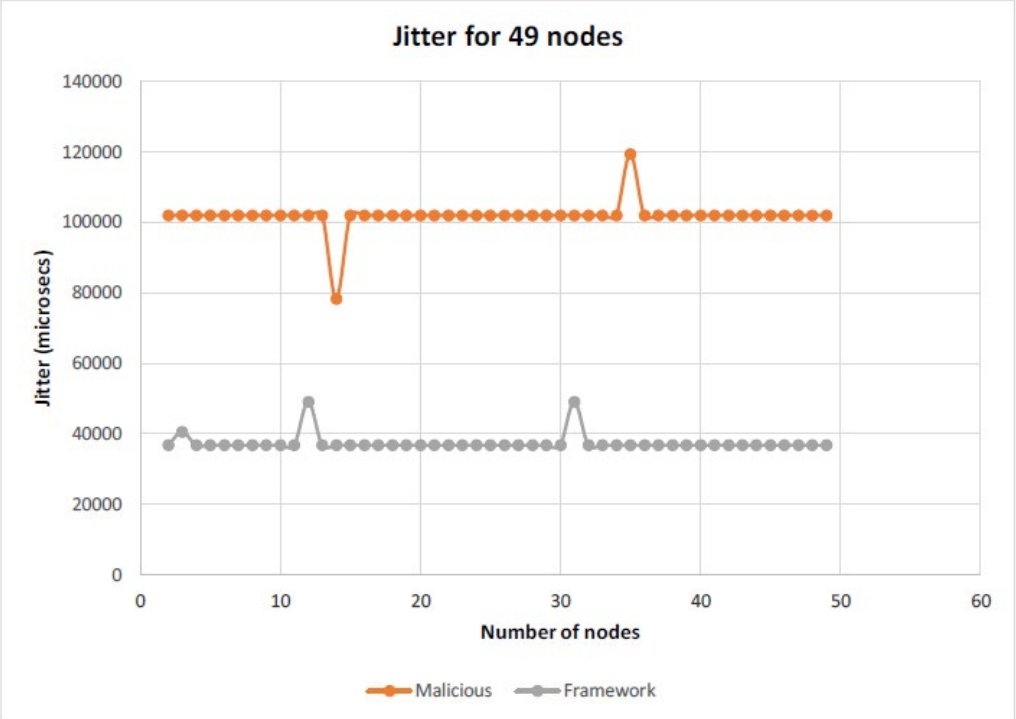

Figure 16. Jitter for 49 nodes.

5.2.1. Throughput Performance Analysis

Figure 14 shows the throughput for both sinkhole and framework scenarios as the first 5000 ms were set to be the time at which the application starts. In the sinkhole scenario, the next 8833 milliseconds showed a positive hike to 1.522 Mbps in throughput followed by a steady decrease to 0.88 Mbps. The highest network throughput achieved is 1.525 Mbps at 13,604.24 ms of simulation time. The scenario's output significantly dropped after 10,000 milliseconds. The framework on the other hand had a positive hike in figures throughout with a steady increase until the end of the simulation. The highest network throughput achieved was 2.37 Mbps at 94,282.78 ms. The framework scenario's throughput grew over time, unlike the sinkhole scenario which dropped performance just after a tenth of the simulation time.

5.2.2. Delay Performance Analysis

The results of the scenarios show a relatively high delay value for the framework as compared to the other scenarios in Figure 15. This margin is proportional to the throughput of the network which is higher than the other scenario. The presented result proved that the employed practices were very effective. In the malicious scenario, all the other nodes show a constant delay of 2,137,687.396 microseconds except node 14, which has a delay of 2,771,719.28 microseconds. The margin shown in Figure 15 is extremely high compared to the framework whose delay for almost all the nodes was around 40,000 microseconds. The delay increases as many nodes are included in the network, but the framework was able to sustain the network performance and provide an acceptable level of QoS. The two scenarios have an opposite outlook in terms of their delay propagation. The growth in the delay is, however, normal as the network grows but the optimization of OLSR and the cross-layer interaction between the network, physical, and MAC layers play a huge role in making the network more delay-sensitive and efficient.

5.2.3. Discussion for Jitter

The results of the scenarios in Figure 16 show a rise in jitter levels for the two scenarios as compared to when there were 49 nodes in the network. The malicious scenario had extremely higher levels of jitter than that of the proposed framework. This result is strongly attributed to the framework's key components, such as admission control, resource reservation, and the IDS. In the normal scenario, the first two nodes show a steady hike in jitter from 0 until it reaches 101,938.83 microseconds. The 11 nodes maintain a constant level of 101,938.83 microseconds but node 14 dropped to 78,282.15 microseconds. The rest of the nodes maintain this level until the end of the application. The highest recorded jitter level remains at 101,938.83 microseconds which is fairly normal based on the number of nodes in the network. This scenario was not part of the comparison between the sinkhole and framework scenarios but rather acted as a litmus to the two scenarios. In the malicious scenario, the first two nodes show a steady hike in jitter from 0 until it reaches 119,335.24 microseconds. The rest of the nodes maintain this level until the end of the application except node 14, which has a jitter level of 78,282.16 microseconds. The framework scenario outperformed the sinkhole scenario by reducing the jitter levels to almost four times lower than when the sinkhole nodes were in action. The framework's route selection mechanism is well equipped to deal with various changes in the route or queuing network. The prioritization of packets by the framework further helped in reducing network congestion.

5.3. Simulation 3: Performance Evaluation under High-Density Nodes Scenario (100 Nodes)

The proposed framework still outperforms the malicious node scenarios based on the stated performance metrics. There were high margins of delay and jitter for the malicious scenario relative to the density of the nodes due to lack of timely access to resources. Figure 17 presents the implemented network model of 100 uniformly placed mobile nodes with sinkhole nodes present.

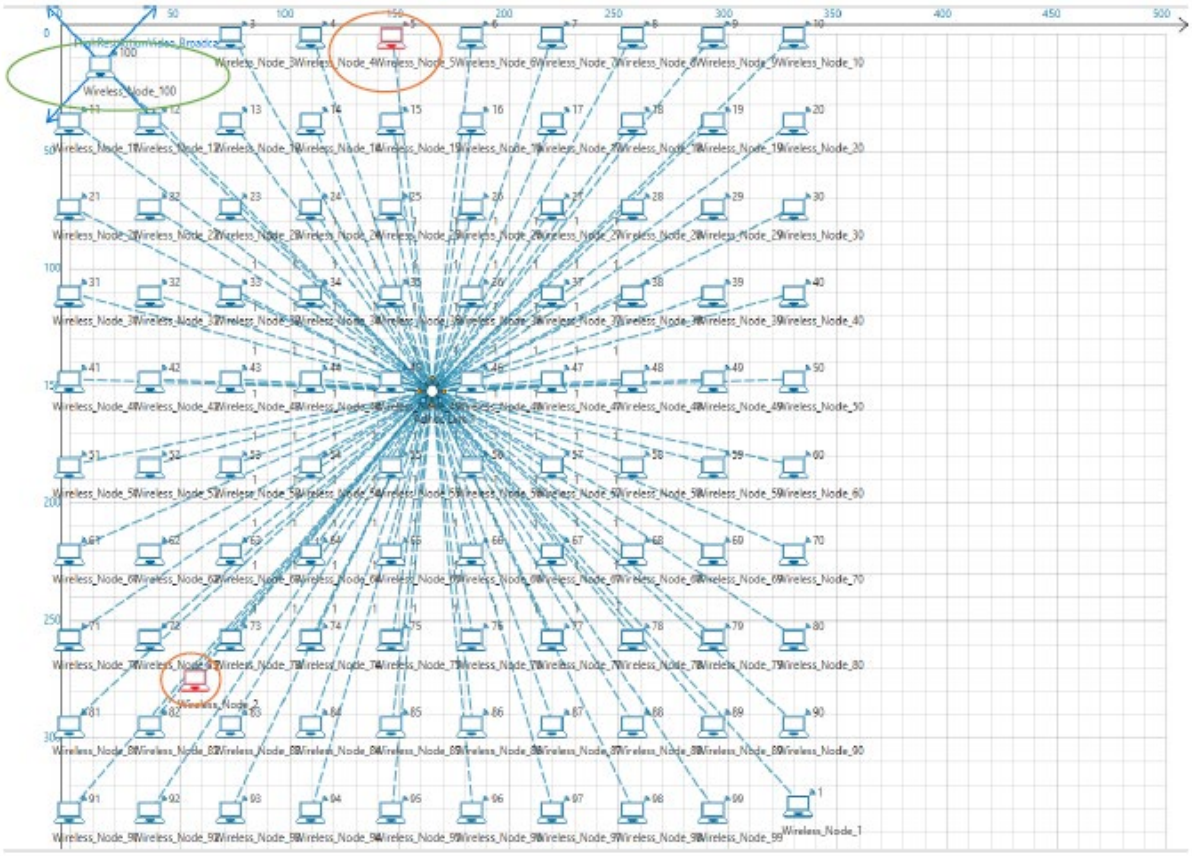

Figure 17. Network model of 100 nodes with two sinkhole nodes added.

5.3.1. Throughput Performance Analysis

Figure 18 shows the throughput of 100 nodes as the first 5000 milliseconds represent the time at which the application starts. In the malicious scenario, the next 4972 milliseconds showed a positive hike to 1.78 megabits/s in throughput followed by a steady decrease to 0.391 megabits/s at 68,851.81 milliseconds. The framework scenario showed a positive hike to about 3.68 megabits/s in throughput followed by a steady increase to 4.79 megabits/s at 99,727.63 milliseconds. This is the highest recorded throughput throughout all the simulations conducted despite the high node density. The high jitter levels contribute to the late arrival of packets and low levels of throughput in the sinkhole scenario. The high node density environment means that many nodes compete for access to resources and the unavailability of queuing mechanisms would mean higher traffic congestion in the network. This does not mean that having fewer nodes in the network might be the solution as literature reveals that OLSR performs better in high node densities. The detection and isolation of the sinkhole attack in the network is the best solution to network performance as done with the framework scenario. The QoS mechanism implemented optimizes the use of the available bandwidth and traffic prioritization.

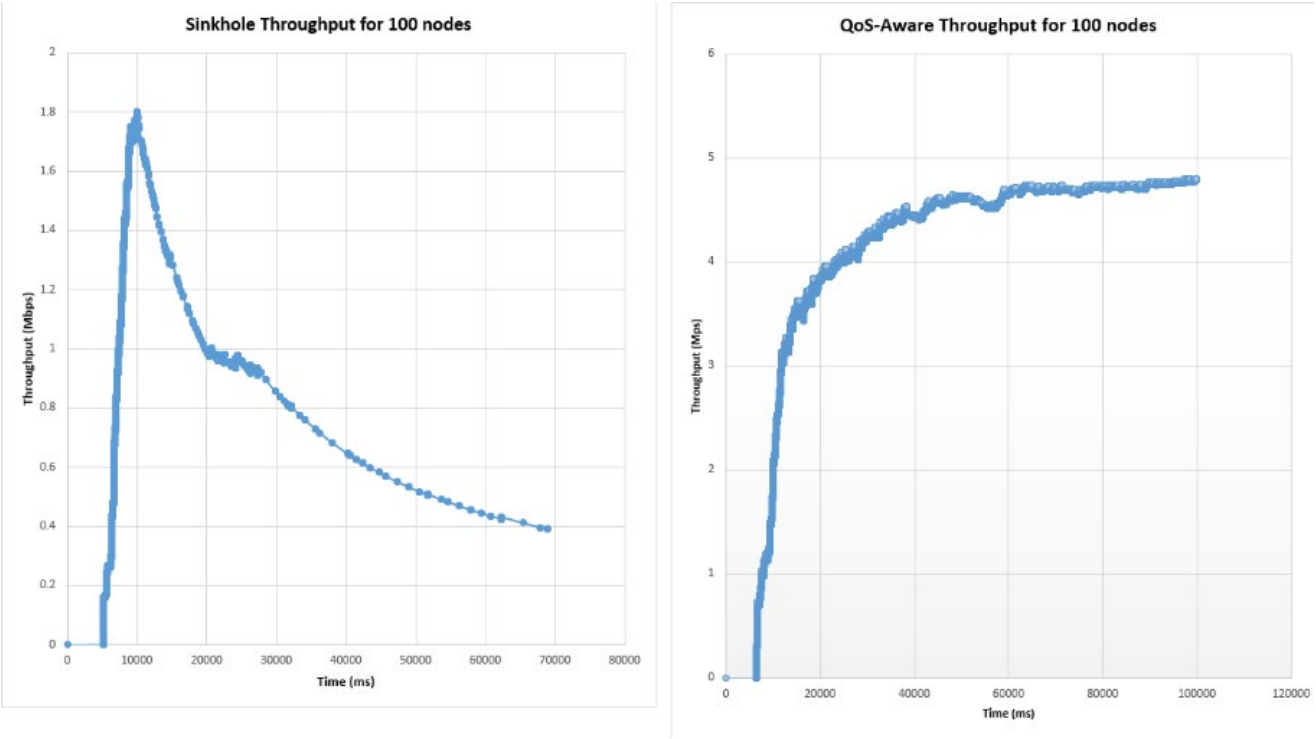

Figure 18. Throughput for sinkhole and framework scenarios.

5.3.2. Delay Performance Analysis

The results of the scenarios ijn Figure 19 show a high delay margin for the malicious scenario as compared to the framework scenario. In the sinkhole scenario, the highest recorded delay value was 6,152,112.929 microseconds for node 63 as it was the first computed node, such as nodes 29 and 53. These levels are too high as compared to the framework scenario. The framework scenario at node 1 had a delay of 581,591.68 microseconds. All the other nodes produce a constant delay of 429,470.72 microseconds. This was a significant drop in delay compared to the other scenario. The effectiveness of the framework is clearly shown in the result presented in Figure 19 where the delay was reduced to about four times as compared to that of the sinkhole scenario. The framework aimed to maximize QoS offering and this is evident as it outperforms the normal scenario as well. It is important to note that the delay caused by the malicious nodes was improved significantly by the framework implementing the modified OLSR protocol.

5.3.3. Jitter Performance Analysis

The results of the scenarios in Figure 20 show a high jitter margin for the malicious scenario as compared to the other scenarios. It was eight times higher than that of the framework. The jitter levels rise from 309,406.55 to a constant 796,236.23 microseconds and this result is less desirable from the malicious scenario because it has an even lower throughput to that effect. In the sinkhole attack scenario, the first 28 nodes show a constant level of 796,236.267 microseconds. Node 29 is on 309,406.55 microseconds. There is a continuous drop of jitter levels from 796,236.267 microseconds for three other nodes in the network. The drop fluctuates between 297,602.65 and 305,562.99 microseconds. This is a relatively high level of jitter, but it is not surprising as the network is under an active sinkhole attack. The projected result from the malicious scenario was caused by the late access to enough bandwidth as it is shared among the different nodes. The density of the nodes in the network further affected this result but the framework has an even lower margin because there was no malicious behavior. The sinkhole attack in the other scenario maximizes bandwidth usage unnecessarily in certain nodes and thus starving others.

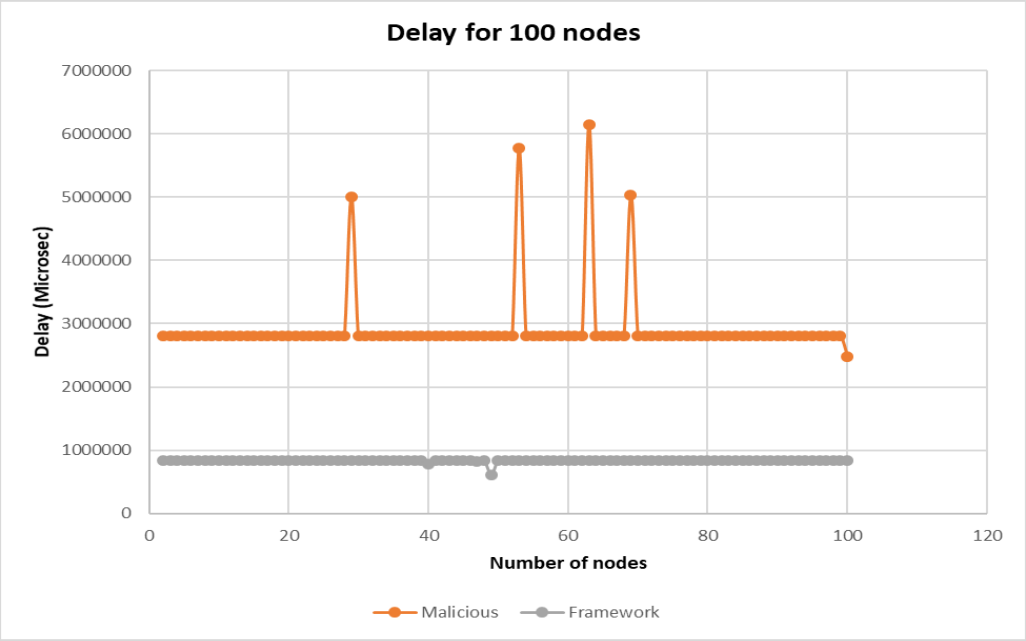

Figure 19. Delay for 100 nodes.

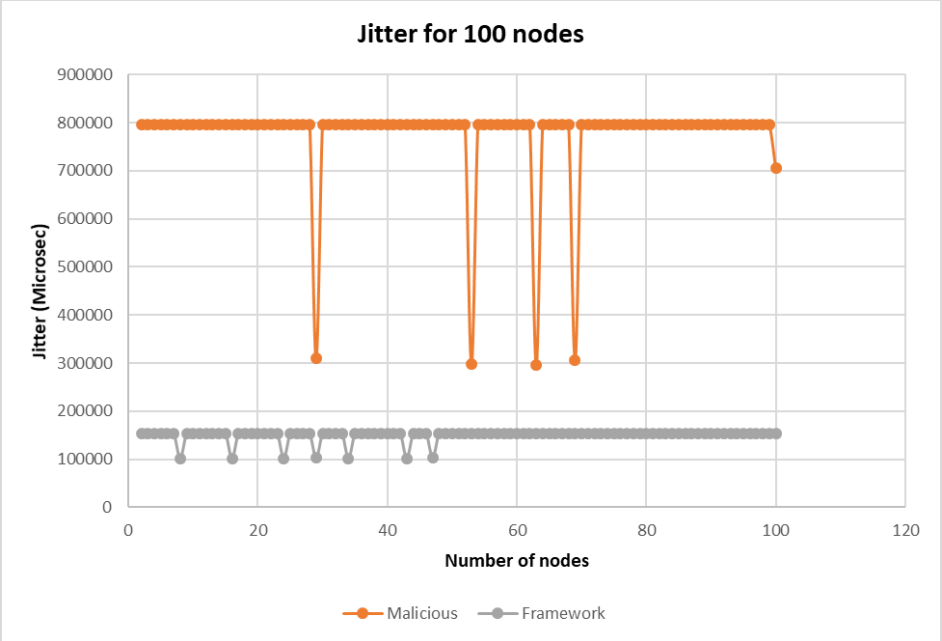

Figure 20. Jitter for 100 nodes.

6. Conclusions

In this paper, we presented and discussed a security framework from the perspective of QoS by using MANET routing protocol, OLSR. The framework aimed to address sinkhole attacks during OLSR routing and to maximize network performance. The architecture of the QoS-aware security framework was presented, and its unique set of components explained with the aim of QoS delivery in video streaming applications. The functions within the framework were explained under their respective interconnected layers. Most approaches to network security do not consider the aspect of QoS, hence our contribution to knowledge by developing a QoS-centric framework that will not only manage the security aspect of the network but also the QoS delivery capacity of the network. The implemented lightweight IDS enhances performance as it is customized to work with the routing process

of OLSR in the MAC and network layers. A custom flow chart of OLSR was also presented as part of our contribution to QoS-aware adaptive routing. The evaluation results showed the effectiveness of the proposed framework and its strengths. In real life situations, the three scenarios can be likened to three environments (banks, companies or institutions, vehicular highway, smart cities etc.) that are low, medium, and highly populated within a given space size (say 1000 m $\times$ 1000 m grid in our experiment). The sensor nodes are computers, smart vehicles, smart houses, etc., which communicate to share secret and vital information. The malicious nodes are intruders, hackers, phishers, etc., of the same devices wanting to break into the system by masquerading as a genuine member of the system. Our framework scenario is set up in such a way that irrespective of the environment and density, any masquerading devices are detected and denied access to the network. In all cases of performance measurement, our framework has a higher throughput, reduction in delay to transmit packets, and less jitters enabling sent signals (packets) to be received without distortions. As future work, the ultimate intention is to further customize our framework for high-speed nodes, VANETs, and their routing protocols. We will also consider the battery power of each sensor node and how to manage them for efficient power utilization.

Author Contributions: Conceptualization, B.M.E., T.P., and F.L.; methodology, B.M.E. and T.P; software, T.P.; validation, B.M.E., T.P., and F.L.; formal analysis, B.M.E. and T.P.; investigation, T.P.; resources, B.M.E., T.P., and F.L.; data curation, B.M.E. and T.P.; writing—original draft preparation, T.P.; writing—review and editing, B.M.E. and F.L.; visualization, B.M.E. and F.L.; supervision, B.M.E. and F.L.; project administration, B.M.E.; funding acquisition, B.M.E. and F.L. All authors have read and agreed to the published version of the manuscript.

Funding: The APC was funded by North-West University.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data used in the simulation are contained in this article.

Acknowledgments: The authors are grateful for the support from FNAS Postgraduate office, MaSIM research entity of the North-West University, and our partners at TETCOS, India who provided the testbed for the practical and evaluation of the proposed framework.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Goyal, P.; Parmar, V.; Rishi, R. Manet: Vulnerabilities, challenges, attacks, application. *IJCEM Int. J. Comput. Eng. Manag.* **2011**, *11*, 32–37.
2. Islam, M.S.; Riaz, A.; Tariqu, M. Performance analysis of the routing protocols for video streaming over mobile ad hoc networks. *Int. J. Comput. Netw. Commun.* **2012**, *4*, 133–150.
3. Delgado, G.D.; Frías, V.C.; Igartua, M.A. Video-streaming transmission with qos over cross-layered ad hoc networks. In Proceedings of the 2006 International Conference on Software in Telecommunications and Computer Networks, Split, Croatia, 29 September–1 October 2006; pp. 102–106.
4. Lindeberg, M.; Kristiansen, S.; Plagemann, T.; Goebel, V. Challenges and techniques for video streaming over mobile ad hoc networks. *Multimed. Syst.* **2011**, *17*, 51–82. [CrossRef]
5. Baiad, R.; Alhussein, O.; Otrok, H.; Muhaidat, S. Novel cross layer detection schemes to detect blackhole attack against QoS-OLSR protocol in VANET. *Veh. Commun.* **2016**, *5*, 9–17. [CrossRef]
6. Sharma, S.; Mahajan, M. Security mechanisms for mitigating multiple black hole attack in Manets. *IJISE Int. J. Innov. Sci. Eng. Technol.* **2015**, *2*, 582–588.
7. Sahu, Y.; Rizvi, M.; Kapoor, R. Intruder detection mechanism against DoS attack on OLSR. In Proceedings of the 2016 Fifth International Conference on Eco-friendly Computing and Communication Systems (ICECCS), Bhopal, India, 8–9 December 2016; pp. 99–103.
8. Khan, M.S.; Khan, M.I.; Khalid, O.; Azim, M.; Javaid, N. MATF: A multi-attribute trust framework for MANETs. *EURASIP J. Wirel. Commun. Netw.* **2016**, *2016*, 197. [CrossRef]
9. Monica, L.L. Evaluation of attacks using different parameters based on their performance. *Evaluation* **2018**, *1*, 106–110.
10. Deebak, B.D.; Al-Turjman, F. A hybrid secure routing and monitoring mechanism in IoT-based wireless sensor networks. *Ad Hoc Netw.* **2020**, *97*, 102022.

11. Sekaran, R.; Goddumarri, S.N.; Kallam, S.; Ramachandran, M.; Patan, R.; Gupta, D. 5G integrated spectrum selection and spectrum access using AI-based frame work for IoT based sensor networks. *Comput. Netw.* **2021**, *186*, 107649. [CrossRef]
12. Raghav, R.S.; Thirugnansambandam, K.; Anguraj, D.K. Beeware routing scheme for detecting network layer attacks in wireless sensor networks. *Wirel. Pers. Commun.* **2020**, *112*, 2439–2459. [CrossRef]
13. Linares-Espinós, E.; Hernández, V.; Domínguez-Escrig, J.; Fernández-Pello, S.; Hevia, V.; Mayor, J.; Padilla-Fernández, B.; Ribal, M.J. Metodología de una revisión sistemática. *Actas Urológicas Españolas* **2018**, *42*, 499–506. [CrossRef] [PubMed]
14. Dooley, K. Simulation research methods. *Companion Organ.* **2002**, 829–848.
15. Kothari, C.R. *Research Methodology: Methods and Techniques*; New Age International: New Delhi, India, 2004.
16. Law, A.; Kelton, W. *Simulation Modeling and Analysis*; McGraw-Hill Company: New York, NY, USA, 1982.
17. Saifuddin, K.M.; Ali, A.J.B.; Ahmed, A.S.; Alam, S.S.; Ahmad, A.S. Watchdog and pathrater based intrusion detection system for MANET. In Proceedings of the 2018 4th International Conference on Electrical Engineering and Information & Communication Technology (iCEEiCT), Dhaka, Bangladesh, 13–15 September 2018; pp. 168–173.
18. Appandairaj, P.; Kannan, K. Software-defined multilayered admission control for quality of service assurance in mobile ad-hoc networks. *Wirel. Commun. Mob. Comput.* **2020**, *2020*, 2989751. [CrossRef]
19. Phakathi, T.; Lugayizi, F.; Esiefarienrhe, M. Quality of service-aware security framework for mobile ad hoc networks using optimized link state routing protocol. *arXiv* **2020**, arXiv:2010.01852.
20. Moad, D.; Djahel, S.; Naït-Abdesselam, F. Improving the quality of service routing in OLSR protocol. In Proceedings of the 2012 International Conference on Communications and Information Technology (ICCIT), Hammamet, Tunisia, 26–28 June 2012; pp. 314–319.
21. Yu, X.; Navaratnam, P.; Moessner, K. Resource reservation schemes for IEEE 802.11-based wireless networks: A survey. *IEEE Commun. Surv. Tutor.* **2012**, *15*, 11–29.
22. Hou, C.; Xu, Z.; Jia, W.-K.; Cai, J.; Li, H. Improving aerial image transmission quality using trajectory-aided OLSR in flying ad hoc networks. *EURASIP J. Wirel. Commun. Netw.* **2020**, *2020*, 140. [CrossRef]
23. Suman, S.K. File Exchange. 2020. Available online: https://www.tetcos.com/file-exchange.html (accessed on 11 March 2021).
24. Kumar, D.A.; Venugopalan, S. Intrusion detection systems: A review. *Int. J. Adv. Res. Comput. Sci.* **2017**, *8*. [CrossRef]
25. Singh, G.; Cheema, K.A.; Kapoor, N. Performance evaluation of routing protocol in internet of things using netsim. *Int. J. Adv. Res. Comput. Sci.* **2017**, *8*, 856–859.