PREDICTION OF DISEASE SPREAD PHENOMENA IN LARGE DYNAMIC TOPOLOGY WITH APPLICATION TO MALWARE DETECTION IN AD HOC NETWORKS

A Dissertation

Submitted to the Faculty

of

Purdue University

by

Nadra Guizani

In Partial Fulfillment of the

Requirements for the Degree

of

Doctor of Philosophy

May 2020

Purdue University

West Lafayette, Indiana

THE PURDUE UNIVERSITY GRADUATE SCHOOL
STATEMENT OF DISSERTATION APPROVAL

Dr. Arif Ghafoor, Chair
School of Electrical and Computer Engineering

Dr. Walid G. Aref
Department of Computer Science

Dr. Ali El Gamal
School of Electrical and Computer Engineering

Dr. Ali A. Elghariani
School of Electrical and Computer Engineering

Approved by:

Dimitrios Peroulis, Head of Head of the Graduate Program
School of Electrical and Computer Engineering

To my parents Mohsen Guizani and Saida Garsellaoui, and grandparents Mokhtar and Mubarka Guizani, and Omar and Masouda Garsellaoui.

ACKNOWLEDGMENTS

I would like to thank my advisor Professor Arif Ghafoor for all his support and encouragement over the years; my committee members Professor Walid Aref, Dr. Aly El Gamal, and Dr. Ali Elghariani for their directions and support in completing this dissertation work; the Electrical and Computer Engineering School staff, especially Matt Golden, Karen Jurss, and Michelle Wagner; my colleagues Muhammed Felemban, Amaal Tashkandi, and Badria Alenezi for all the brainstorming and feedback on new and ongoing research ideas. Most importantly, I would like to thank my family: my parents Mohsen Guizani and Saida Garsellaoui for always pushing me to be the best I can be. My siblings Fatma, Maher, Zainab, Sara, and Safa Guizani for always being there for emotional support and family bonding. And last but not least, thanks to my niece Haleema Mohamed for giving me the much needed boost of motivation to complete the writing of this dissertation!

TABLE OF CONTENTS

Page

Page

LIST OF TABLES

LIST OF FIGURES

ABBREVIATIONS

AB	Agent Based
CVSS	Common Vulnerability Scoring System
FSM	finite state machine
GP	Global Population
HMM	Hidden Markov Model
IDS	Intrusion Detection System
IoT	Internet of Things
LSTM	Long-Short Term Memory
MANET	Mobile Ad Hoc Network
NFV	Network Function Virtualization
RNN	Recurrent Neural Network
SB	Spatial Based

ABSTRACT

Guizani Nadra Ph.D., Purdue University, May 2020. Prediction of Disease Spread Phenomena in Large Dynamic Topology with Application to Malware Detection in Ad Hoc Networks. Major Professor: Arif Ghafoor.

Prediction techniques based on data are applied in a broad range of applications such as bioinformatics, disease spread, and mobile intrusion detection, just to name a few. With the rapid emergence of on-line technologies numerous techniques for collecting and storing data for prediction-based analysis have been proposed in the literature. With the growing size of global population, the spread of epidemics is increasing at an alarming rate. Consequently, public and private health care officials are in a dire need of developing technological solutions for managing epidemics. Most of the existing syndromic surveillance and disease detection systems deal with a small portion of a real dataset. From the communication network perspective, the results reported in the literature generally deal with commonly known network topologies. Scalability of a disease detection system is a real challenge when it comes to modeling and predicting disease spread across a large population or large scale networks. In this dissertation, we address this challenge by proposing a hierarchical aggregation approach that classifies a dynamic disease spread phenomena at different scalability levels. Specifically, we present a finite state model (SEIR-FSM) for predicting disease spread, the model manifests itself into three different levels of data aggregation and accordingly makes prediction of disease spread at various scales. We present experimental results of this model for different disease spread behaviors on all levels of granularity. Subsequently, we present a mechanism for mapping the population interaction network model to a wireless mobile network topology. The objective is to analyze the phenomena of malware spread based on vulnerabilities. The goal is to

develop and evaluate a wireless mobile intrusion detection system that uses a Hidden Markov model in connection with the FSM disease spread model (HMM-FSM). Subsequently, we propose a software-based architecture that acts as a network function virtualization (NFV) to combat malware spread in IoT based networks. Taking advantage of the NFV infrastructure's potential to provide new security solutions for IoT environments to combat malware attacks. We propose a scalable and generalized IDS that uses a Recurrent Neural Network Long Short Term Memory (RNN-LSTM) learning model for predicting malware attacks in a timely manner for the NFV to deploy the appropriate countermeasures. The analysis utilizes the susceptible (S), exposed (E), infected (I), and resistant (R) (SEIR) model to capture the dynamics of the spread of the malware attack and subsequently provide a patching mechanism for the network. Our analysis focuses primarily on the feasibility and the performance evaluation of the NFV RNN-LSTM proposed model.

1. INTRODUCTION

Throughout history there has been a number of pandemic waves that have struck the world at large. We can go as far back as the Bubonic plague. Currently in the year 2020, we are dealing with the Novel Coronavirus (COVID-19) which has now become a world wide concern and crisis. Before this virus came the MERS outbreak of 2012, SARS in 2002, Swine Flu of 2009, AID epidemic in the early 1980's, and of course the all to infamous Spanish Flu 1918. All these epidemics have one thing in common, the spatio-temporal network that is the human connection.

Public Health care officials are routinely faced with the challenge of finding an effective way to predict and prevent control epidemics. When dealing with epidemics, quickly developing effective countermeasures is crucial for saving lives. A key factor in this decision making process is to understand the spatio-temporal dynamic of the epidemic [1]. For pre-planning exercises, disease spread models can be developed and analyzed. In essence, such models should capture human-to-human interaction among population as such interaction generally serves as a medium of disease transmission. Interaction within a population can be modeled in terms of social networks. These models can be developed at various levels of population granularity ranging from an individual level up to an aggregated scale representing the total population.

Modeling disease spread data on social networks does not only benefit the longevity of the human race but also provide efficient ways to combat dangerous diseases. These models can be utilized in multiple domains such as data spread in mobile ad hoc networks, power consumption spread in a power grid system, etc [2,3]. In fact, any type of spread that can be defined within the scope of a graph network can benefit from a network based prediction model. Computation and storage of disease spread data for every node in a network is not scalable when it comes to dealing with very large social networks. Also, the computation speed of prediction decreases as the number

of nodes in the network increases. The faster the prediction data is available, the faster the end user can employ countermeasure techniques to stall the spread of the disease.

Likewise, epidemic models can be applied to analyze the spread behavior of viruses in mobile networks. In these networks, each device can be represented as an agent that communicates, either through blue-tooth or wireless connections. The main countermeasure to combat mobile viruses in the network is by deploying software updates/patches. The issue here is that mobile networks do not support global updates, rather by sending updates in turn to specific groups and areas. Utilizing the epidemic models to predict the spread of the virus can help efficiently deliver updates (vaccines) to the groups with more need so that the spread is contained as efficiently as possible. Mobile networks can also be observed at different levels of population granularity, as discussed later in Chapter 3.

1.1 Disease Spread Models

There are three different classes of mathematical approach for modeling infectious disease dynamics: statistical-based methods for epidemic surveillance; mathematical state-space models; and empirical learning-based models.

1.2 Prediction Techniques

Statistical-based methods or prediction techniques (referred to as forecasting) depicts certain events that materialize or occur at any point in time and space. The process of prediction (statistical inference) has been utilized in science, art, finance, and many other fields. Prediction can be performed by any of the several statistical inference approaches. Statistical inference deals with both descriptive prediction of a whole data set or infers properties of an entire population through a sample data set. Large scale social network predictive problems take advantage of both models of inference. Predictive techniques such as Bayesian network, genetic, regression, Hidden

Markov models, Neural Networks algorithms and many more are employed to solve the corresponding problem. This research uses two of the above predictive techniques and discuss how these techniques use disease spread model necessary for both social and mobile networks.

Bayesian network represents the probabilistic relationships between diseases and symptoms. Given a set of symptoms, a network can be used to compute the probabilities of the presence of various diseases. The authors in [4] connect Bayesian estimation to several different social network topologies in terms of learning in the network. Initialization at each node (agent) is chosen from a Gaussian distribution [5]. The research concludes that all the nodes converge to obtain the same conclusion without having direct access to all private information. This process is computationally efficient and depends on the size of the network being analyzed. The process preserves the privacy of individuals while still providing the right conclusion to all nodes in the network.

Bayesian classifier can be incorporated with many other learning algorithms. Authors in [6] utilize a probabilistic neural network (PNN) to create a more efficient Bayesian classifier through the use of a genetic algorithm. In this case the PNN does not require large datasets or the initialization or assumption of certain smoothing parameters. This specific research takes advantage of several prediction (learning) techniques which include: feed-forward neural network (FFN), and support vector machine (SVM) .

Genetic algorithms are applied to an optimization problem to generate a better solution. Genetic algorithms are utilized for solving optimization problems that examine network topology structures [7–9].

The successful implementation of these techniques are based on the type of network which are analyzed. A substantial volume of research focuses on the infection in a population network. Other networks that can be analyzed with these implementations are but not limited to: mobile networks, IoT networks, and data disseminated networks.

1.2.1 Mobile Networks

Many hardware and software mobile technology organizations have been highly concerned with the latest increase in security attacks/breaches that can start in mobile networks (including IoT devices) and extend to other parts of the overall global network. It is important to note that different types of attacks that can threaten the network at both ends. That way, it can be ensured that a virus does not evolve to be a vulnerability for other network types/devices. During the last few years, viruses have been compromising both the mobile network and the mobile device/agent individually. These attacks are expected to increase with the increase of new technologies and the increased usage of mobile devices. Accessing IoT devices over a mobile network is another security risk that can be potentially be introduced to the overall social network.

Throughout the last decade, mobile technologies have become more prominent in our daily life. Daily activities such as social interactions through social media, online shopping, and most importantly transactions of all types (rent, credit card, groceries, food delivery, etc.) have become the social norm. Because of the increase in personal information (name, social security number, birth dates, credit cards, etc.) being saved on mobile technologies, such information has become an increasingly attractive target for online criminals. As a result, hackers are investing in devising more sophisticated attacks that are effective in stealing valuable personal data, and causing disruptions of server, and network functions. In 2016, the mobile threat reported that Android apps containing malware to be three times as many as those that occurred in 2013 and 2014 combined, an increase of 230%. In recent years vulnerabilities on the iOS platform have accounted for the greatest number of mobile vulnerabilities, which is due to the ability to jail-break iPhones thus making it more vulnerable to malware attacks. Also cross over threats have increased drastically given that mobile and web applications are being linked more regularly [10, 11].

From the statistics listed above, this dissertation investigates the development of classification models that predict threats of virus attacks in mobile networks. We use the theory behind epidemic spread models in terms of finite state machine (FSM) implementation on complex networks. This theory can be applied to effectively analyze the issue of information dissemination or virus spread in mobile networks. Mobile human networks (i.e., ad hoc networks composed by devices carried by individuals) can be frequently and temporarily disconnected imitating a behavioral FSM. Figure 3.2 shows an example of a finite state machine for a mobile agent connected to a spatio-temporal network. Spatio-temporal networks are usually seen in terms of the movement of people. The mobile agent in this case is synonymous with a person agent; having the same characteristics of daily movements.

The FSM model would take into consideration the prediction of mobile agents at different points of time in the day, week, or year. At what points in time does a mobile agent become a susceptible agent versus a host agent in terms of different viruses? These predictions provide software stakeholders access to deploy network or mobile updates to the right locations. Creating a more optimized spread of "vaccination" for mobile agents/network. These predictions also provide different stakeholders with statistical information for future research and mobile updates, and network updates (hardware and software). Prediction techniques can be developed through several learning models. They can utilize neural networks, genetic algorithms, Bayesian networks, etc. In this research work, we apply a customized Hidden Markov Model (HMM) to categorize and predict different mobile nodes in a network to their behavioral states. Several categories/classes are developed and estimated using the Baum-Wells estimation maximization technique. The training technique that is developed is generalized to create any HMM-FSM category whether it be for human based diseases or mobile based software viruses. These categories can be added to the preexisting model. HMM is used in particular with networks where the desired nodes' characteristic output can only be observed through one or more hidden pro-

cesses. The characteristics seen do not directly provide the necessary information that the stakeholder requires.

1.3 Malware in IoT

IoT devices are becoming the new norm in our daily life. IoT technology provides solutions to creating a smarter environment that saves time, reduces energy consumption and reduces cost, to name a few. The number of devices connected to emerging IoT applications and infrastructure are expected to reach 50 billion by 2020 [12], giving rise to an enormous amount of valuable data. IoT devices are used across various sectors including ehealth, smart homes and intelligent transportation, for instance. From wearable diabetes machines to the Amazon Alexa, these devices are becoming embedded in the fabric of society. The integration of IoT technology with the network poses enormous security challenges and creates a path in which malicious users can get access to, manipulate, and/or utilize personal and highly classified information.

Due to the increased usage of IoT devices, IoT systems have become a main resource for collecting various types of data such as personal, marketing, and administrative data [13]. IoT networking technologies use a multitude of protocols for data communication and network connectivity. IoT devices can utilize a variety of sensors. Each sensor within an IoT device can utilize different data and communication protocols as shown in Figure 1.1. Protocols within one device are used for interaction between: device to device (D2D), device to server (D2S), and server to server (S2S). In addition to the protocols listed above, Table 1.1 lists several protocols that handle connectivity, data management and communication of IoT devices that must be taken into consideration when gathering or analyzing network traffic data. Each protocol generally uses a different mechanism for processing traffic data dependant on the computing available framework. Computing frameworks include but are not limited to: Fog computing, Edge Computing, Cloud Computing, and Distributed Computing [14].

Table 1.1: List of IoT Protocols.

Communications	**Data Transfer**	**Connectivity**
Constrained Application Protocol (CoAP)	SSI	IPv6
Datagram Transport Layer Security (DTLS)	Websocket	LPWAN
MQ Telemetry Transport (MQTT)	SMCP	Zigbee
Time Synchronized Mesh Protocol (TSMP)	XMPP-IoT	Bluetooth Low Energy
Advanced Message Queuing Protocol (AMQ)	DDS	Z-Wave

Virtualization forms the basis of cloud computing technologies, offering on-demand access to different applications and services by sharing a pool of configurable resources (e.g., networks, servers, and storage). Due to the growing size of IoT networks and the increased data intake that comes with it, a Network Function Virtualization (NFV) system offers the best security solution in terms of combating complexity and creating efficient attack prediction models to process and patch malware attacks [15, 16]. 5G has been the leading communication network for supporting NFV applications [17,18]. Lightweight virtualization has been implemented for security frameworks focusing on the quality of service performance of IoT edge node devices [19] and the efficiency of Security-as-a-Service features for IoT edge devices [20, 21].

Analysis of IoT data poses an important data science question: what is the best data model that can be efficiently processed by an IDS in terms of detecting attack patterns? To analyze and mine such data, efficient and scalable machine learning algorithms are needed. These algorithms must be capable to analyze data for different protocols employed by the IoT framework. In the literature, prominent prediction techniques are classified in two categories: statistical and machine learning. Statistical techniques such as linear regression and Bayesian estimation concentrate on models in terms of their interpretability, precision and uncertainty. The second category includes machine learning (ML) techniques such as neural networks and fuzzy systems. These techniques focus on large scale applications and prediction accu-

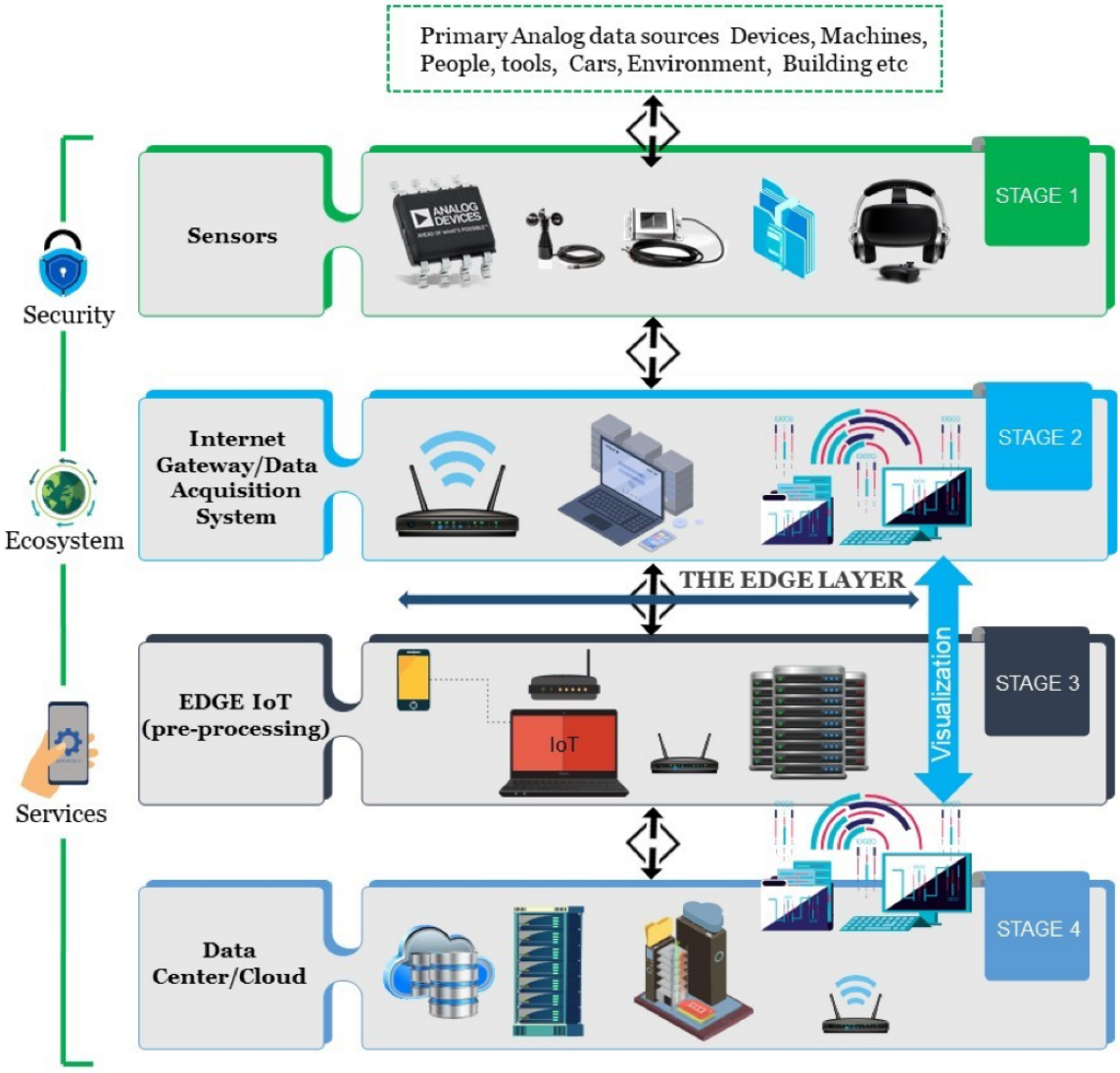

Fig. 1.1.: IoT Architecture [22, 23].

racy [24]. Machine learning algorithms have become the go-to analysis techniques to create better marketing strategies for a business or combat malicious users in high traffic networks. Recently, neural networks have become widely popular due to the easy access to processing power needed to analyze large data sets [25, 26]. The increased access to larger data sets is crucial for the prediction accuracy. Accuracy in prediction is dependant on the type of the neural network used to analyze a specific data set. Image processing, signal processing, and time series estimation applications use a specific type of neural network. As we focus on IoT network security challenges, we propose to use a recurrent neural network time-based formalism. An advantage of neural network and deep learning is to add new features to an already trained model without considerable modifications. Such extensibility helps expand the scope of viruses and malware detected by the model [27]. Specifically, the RNN model has shown to perform superior to other deep learning algorithms when dealing with time-based data sets.

Due to the dense connectivity among IoT devices and with the internet, authors in [28] categorize IoT vulnerabilities based threats into four groups: (1) Denial of Service (DoS) attacks which deny users access to their resources by injecting useless traffic through a network device; (2) Malware attacks that use code injected into the device to access and obtain sensitive information of the user and the system; (3) Data breaches are spoof packets that can retrieve confidential information and communication among users; and (4) Weakening Perimeters in IoT devices which make it easier to breach the security of the network. Different security issues hold higher priority due to the vast range of communication protocols applied to different IoT devices. These threats can be detected by analyzing the network data traffic. Different threats can be associated with different part of the network traffic. Misuse and anomaly intrusion patterns are the generalized versions of threat when analyzing data packets. Misuse intrusion detection is defined as traffic patterns that resemble an intruder's traffic, used in detecting known threats. Anomaly intrusion detection identifies unusual change in traffic patterns as an intrusion. Anomaly is used to find

or compose a set of threats that might be new to the system. When analyzing a new system/network, the use of both techniques is important. Using anomaly detection is necessary to keep threats up to date and the misuse technique is to detect known threats.

1.4 Challenges

The challenges that will be addressed in this dissertation work are:

1. Analysis of epidemiological spread parameters when predicted on differnt levels of data aggregation. For example, the higher global population data aggregation versus the agent-based (tuple) level.

2. Implementation of these epidemiological models to mobile and IoT networks at different levels of data aggregation.

3. Comparison of parameter prediction of the middle layer (state based) to the global population to identify the importance of the spatial aspect of the population definition in different types of social networks.

4. Development of real time intrusion detection system (IDS) for detection and response/recovery of virus spread in mobile ad hoc and IoT Networks.

5. Analysis and comparison of different machine learning algorithms (HMM and Neural Networks) to best develop a real time IDS.

1.5 Organization

This dissertation attempts to answer the challenges listed above through discussion in each chapter. This dissertation covers the following topics: A dynamic system model for a dynamic network that can be generalized to encompass/define different types of information dissemination. The system model includes different levels of an aggregated spatio-temporal social network. Experiments on the dynamic network

is on population networks and mobile ad hoc networks in terms of prediction of disease/virus spread. Chapter 2 delves into the literature that has contributed to topics introduced in Chapter 1. Chapter 3 details the different components of the system model and their connection to each other, discussing the dynamic network model developed to apply a probabilistic SEIR disease spread model to a spatio-temporal population network. It continues to further test these models on different aggregated levels of a population dataset. Chapter 4 examines the similarities between a mobile ad hoc network (MANET) and a population based network. We explore the classification of mobiles in terms of their threat in contracting a virus. Introduces the IDS that is developed and tested in Chapter 5. Chapter 5 introduces and connects an NFV based IDS to an RNN prediction model that is utilized in malware prediction for IoT networks. Answering the challenges 4 and 5. Chapter 6 lists what research tasks will be pursued to continue developing components of the NFV architecture to create an efficient patching malware system for all types of node structure in the network.

2. RELATED WORK

2.1 Disease Spread Models

The effect of network structure with respect to epidemic spread models has been investigated in a variety of ways. Authors in [29] implement these models to analyze and predict viral marketing problems. Concentration has been on analyzing the different network's characteristics (tested 35 different network structures) and the infection seeding strategy (initial infection) of each network. Characteristics of the network were used to analyze as to which one had the most impact on diffusion. The authors have concluded that the size of the network does not have an impact on the diffusion of the disease. Previous research on network characteristics generally focus on network size. The authors in [29] concluded that other researchers should provide other characteristics of network. Average node degree and number of network edges are characteristics that effect the network topology of disease spread in terms of direction and speed of the spread.

SIR model is utilized in [30] to study the effects of clustering on network structures. Clustering is defined as the density within the connection of the network which the authors defined as community structures. Areas of high density with strong connections between nodes creates a strong community structures making the connections among communities weak. If the density of a network is high in a certain area, the connections within that area are stronger than the nodes of the outer laying areas. This density factor is taken into consideration in terms of rate and time of spread when introducing the SIR model. Infection factors are computed with dependence on the interconnections of the community structures. The authors in [30] present an important conclusion, the higher the community structure strength, the lower the number of people that can get infected. Other factors that have an effect on

the network is the seeding strategy. If the initial infection is all in one community structure, then the disease has a higher probability of being contained within that community (since relationships across communities are weak). Another important conclusion; is that the larger the initial infection, the earlier the peak of infection. Which makes it easier for stakeholders to draw conclusions by just concentrating on the very beginning of the infection [31].

The authors in [32] present a variety of methods to ascertain an approximation of any given network, connecting network theory with epidemiological theory. Utilizing methods such as contact or infection tracing to predict the shape and size of the network. In contact tracing only a partial piece of the network is identified since only the neighbors of the infected individual are sought out. But this can be helpful to identify the areas in the network with the highest disease spread. On the other hand, infection tracing uses the initial point of infection to trace the network. This can give more information about partial identification of the network, and can only be used when the initial point of infection is known. The authors also reviewed six computer generated networks and how disease spread is characterized within each network. Subsequently, they have compared their results to the random-mixing model. All networks show a lower initial disease growth than the random-mixing model. The reason is that random networks are characterized by a lack of clustering causing a low early growth rate and low final epidemic size. Lattice networks are generated as a Cartesian product of a complete network graph. That is why lattice networks have high spatial clustering causing them to have a local rapid spread and distribution of infection size mimicking a power law distribution. Scale-free networks have a similar characterization as the lattice networks. An emergent network generated from partnership models can be used to test which network properties, such as number of partnerships, concurrent relationships or network positions, are epidemiologically important [32].

2.2 Network Topology

Authors in [33] take the position of exploring the effect of the Agent Based (AB) and Global Population (GP) models on different types of network structures/topologies. These two models can be applied on different types of social structures. The spread of epidemic as well as the spread of computer viruses is dependent on the underlying network topology. Considering five different network structures, the authors simulated the GP model and compared it to the AB model. The closeness of the GP model to the AB model is measured by a confidence interval percentage. For all the networks simulated, a classic diffusion model (spread initially increases at a high rate and then gradually declines as the time passes) for epidemic spread was used.

In a fully connected network, an infected node can contact every other node in the network minimizing the contact overlap giving the fastest initial growth of a disease. The GP model compares perfectly to the AB simulation due to the fact that, with low clustering, this network experiences the fastest initial disease growth but does not experience early disease burn out staying within the 75 % confidence interval.

The lattice network has maximal clustering (all agents have neighbors), which means that the infected agents repeatedly contact the same agents. After these neighbors are infected, the chance of contacting a susceptible agent declines. Clustering makes it highly unlikely to generate new cases of infected agents. So for the lattice, the epidemic die-out is slightly faster than that of fully connected or random network. Since the latter two go through infecting the whole network whereas the former only spreads to the closest of neighbors.

In a scale free network, one can see a change between the AB and GP models when the disease spread enters a hub of connected nodes. All metrics (diffusion fraction, peak time, and peak prevalence) essentially remain within the 75 percent confidence interval very similar to the random and fully connected graphs. But slightly higher, for the AB simulation for which it is 95 percent. The major difference observed is that in a homogeneous case, the diffusion rate begins at a slow pace as compared to

the GP model whereas in the heterogeneous case, the rate is faster. This is due to the well connected nodes in the heterogeneous case.

A Small World network is where most nodes are not neighbors of one another, rather nodes can be reached from every other node by a small number of hops. In the Small World network, diffusion is even slower than all the previous networks. This is due to the fact that clustering in such networks is significantly higher than the other networks. This only effects the peak time of infection which reduces it slightly in comparison with the GP model. The lattice network also exudes a slow diffusion compared to the GP model. This results in very early burn out rate for the epidemic.

In general, the higher the clustering of a network the earlier the burn out of the spread. Otherwise, there are only a few differences that can be observed among the networks when simulating GP and AB models. The major contributor of the difference in metrics is the high or low clustering of nodes. The heterogeneity and main structure of the network do not have a great impact on the behaviour of the disease spread phenomena.

2.3 Intrusion Detection System (IDS)

Prediction of disease or virus spread in many technology sectors has become a major part of the machine learning research community. Phone and application based companies like Apple, Google, and PayPal face growing security challenges as mobile technologies proliferate. Government entities such as the FCC and others have become weary of dealing with fraudulent cases due to security flaws in applications that house private personal data. Developing threat prediction techniques for networks whether social (for disease spread such as the flu) or mobile (for effectively disseminating updates to avoid security threats) has become an very important research challenge. Further extending this concept towards developing secure networks in IoTs is essential if IoT technology needs to be widely deployed.

A myriad of models exist to detect various attacks mentioned above. Authors in [34] employed a mixture of fuzzy classifier with a genetic attribute selection method to detect cyber attacks for wireless mobile networks. Experimentation to test the fuzzy genetic classifier was performed on the KDD Cup dataset.Researchers have developed a two way mapping technique to classify 25 types of malware families. Support Vector Machines are used to train the classes in the data set used in [35]. Authors in [36] use machine learning to detect the performance differences in Android applications without the need to run the application itself. This can help with the detection of malicious Android applications.

Many researchers have developed models for intrusion detection system (IDS) to detect anomalous behavior in the network. These systems are constructed and deployed at different points of the network. Different characteristics of the network are used to detect misuse attacks such as, buffer overflow, packet exchange, or application system calls. Many researchers prefer to use a probabilistic modeling technique since a users' behavior is non-deterministic. The techniques proposed in this dissertation fall into this category.

In [37], researchers use Hidden Markov Model (HMM) to detect whether TCP network traffic is malicious or normal using the KDD Cup 1999 dataset. The model concentrates on five specific TCP sessions. Authors in [38] use HMM to detect abnormal behavior pattern by analyzing log information. The agents in the mobile network are used as part of the IDS to help mitigate the network intrusions. In [39], researchers apply HMM to develop an IDS for network sensors. They run a small case study to test their algorithm. Authors in [40] create an IDS through the development of a discrete HMM to discover the type of intrusion attack. Their proposed HMM utilizes IP sweeping and use the IP's attributes as the corresponding observation. One of the key attributes is the CVSS score for each attack. This limits the data set for their model. HMM modeling is also used for mobile networks for predicting energy level [41], and combining prevention and authentication techniques [42].

The detection and modeling systems for all the previous research focus on specific attacks and for specific computer network topologies. The test dataset for these systems are limited to small or specific case studies. Our proposed prediction model is general and is applicable to any type of attack. For this purpose is an observable sequence and corresponding hidden states are given to the system to develop the required prediction model. In addition, we not only classify on one HMM class but several classes can be trained and tested.

2.3.1 Hidden Markov Model Definition

There are two stochastic processes of HMM states, observable and hidden states. Observable states are the data points that are given/known to the user. Observable states are usually defined from the characteristics of the node in the network. For example in a social network the node can be a person, with associated demographic and spatio-temporal characteristics. The hidden states are states that cannot be deduced directly. The connection between the observable and hidden is what dictates the hidden state outputs. Considering the same example of a population network, the hidden states would correspond to a disease spread state.

HMMs are used to answer one of three problems: [43]:

1. Given an observation sequence ($O = O_1O_2....O_T$) and a model $\lambda = \{A, B, \pi\}$, compute the probability of the observation sequence for the given model λ.

2. Given an observation sequence ($O = O_1O_2....O_T$) and a model $\lambda = \{A, B, \pi\}$, how can a hidden state sequence ($Q = q_1q_2...q_T$) be chosen.

3. Adjust the model $\lambda = \{A, B, \pi\}$ to optimize and maximize the probability of the observation sequence $[P(O|\lambda)]$ given the model λ.

The general definition of the HMM and the parameters for the proposed model $\lambda = \{A, B, \pi\}$ are listed below. These definitions results in the implementation of the model designed for the mobile network.

- S is the set of hidden states $S = s_1; s_2; ...; s_N$)that a mobile node can transition through. In the proposed model there are four states (N = 4).

- O is the set of observable states ($O = O_1O_2....O_T$) that are the known attributes of each mobile node. T is the end of the observable stream of data for each mobile network

- A is the transition probability matrix that connects the hidden states of set S to each other. A is given by the equation below:

$$A = \{a_{ij}\} = P[q_{t+1} = S_i | q_t = S_j] \tag{2.1}$$

  where q_t is the actual state at time t

- B is an observation probability matrix that connects the observable states O to the hidden states S.

$$B = b_j(k) = P[v_k att | q_t = S_j] \tag{2.2}$$

  Where v_k is the observable state at time t.

- π is the initial distribution vector illustrating the probability of the mobile node initially for each hidden state. Is given as:

$$\pi_i = P[q_1 = S_j] \tag{2.3}$$

2.4 Mobile Viruses

The number of mobile phone users has grown from 38% in 2014 to 50 % in 2018. It is expected to reach 67 % by the end of 2019. The benefits of mobile technologies are ever growing and they come with a plethora of security threats that are evolving and causing interceptions, alterations and disruptions [44].

Mobile viruses over the last ten years have increased with the advancement of technology and interdependence of multiple technologies, such as bluetooth, WiFi, 4G/LTE, etc.

Viruses can get unauthorized access in terms of different methods and intents. In earlier years, these viruses had limited access points. Now with the growing app development, world mobile viruses come in many shapes and sizes. Hackers are using advertising click fraud. Advertisement fraud is one of the most profitable criminal enterprises. Trojan apps have been developed to steal private credentials and passwords for other services, including email; intercept and send messages to infiltrate the owner's contact list.

Since ad hoc networks generally do not have a centralized access point they can be vulnerable. Ad hoc networks are defined as peer-to-peer networks among wireless devices that do not have an access point among them [45]. Due to the decentralized nature of the ad hoc networks Denial of Service (DoS) attacks are easily accomplished by utilizing off-the-shelf equipment. Two things can be accomplished through a DoS attack. First, the sender's transmissions are deferred because the medium is sensed as busy. Second, the receiver's reception is interfered with due to jamming signals. Both of these effects degrade the MANETs network performance significantly.

Mobile viruses can be categorized to three security goals: confidentiality, integrity, and availability. For confidentiality, examples include information theft, bluebugging, and bluesnarfing. In bluebugging a mobile agent becomes a bugging device. Bluesnarfing lets attackers connect without alerting the owner giving them access to phone book entries, calendars, and even the phone's International Mobile Equipment Identity. Another confidentiality issue that MANETs face is Trojan Apps. This threat has have become wide spread and focuses on stealing banking credentials and passwords for other services such as emails and SMS transations. Integrity comprises of hijacking mobile devices, which includes making phone calls or sending expensive texts. Availability is comprised as a result of Denial of Service attacks (DoS), or draining the battery power. Due to decentralized nature of MANETs, there are many access points in the network that can be used as vulnerable points [46]. DoS attacks can be easily caused by using off-the-shelf equipment [45]. A DoS attack can defer a sender's transmission because the connection is sensed to be busy, due to a jamming signal

at the receiver's end. This degrades the MANETs network performance significantly. Another recently popular availability virus is Crypto-mining which is code disguised as battery saving utilities. This virus can also be embedded in gaming application software and sent as an update to the application. The code can deplete the battery power to mine cryptocurrency.

2.5 Machine Learning and Malware Spread

Previous literature has suggested multiple ways to use machine learning technologies to accurately predict malware and intrusion attacks in the network. Different ML algorithms have been tested and results have been reported in the literature based on the data sets used for experimentation including Android, MANET, and IoT network traffic.

Authors in [47] investigate IoT malware based on OpCode sequences using a long-short term memory (LSTM) structure. Their detection accuracy is 98%. However, this experiment uses a very small ARM-based data set. Authors in [24] estimate the stability of paths in MANETs utilizing the Recurrent Neural Network model. Path stability prediction effectively improves routing in MANETs. The experimentation in [24] is to find the most optimal structure for RNN and it is achieved when there are 5 to 10 hidden layer nodes and input nodes.

Alam *et al.* [48] consider eight well-known data mining algorithms, SVM, KNN, and ANN, just to name a few. Different performance evaluation benchmarks such as processing time and computational requirements are utilized to determine the best algorithm. Several experimental tests have been applied on these data sets. They show that artificial neural network (ANN) and deep learning artificial neural network (DLANN) algorithms performed far better than SVM, KNN, NB and LDA. The researchers conclude that a DLANN algorithm provides the best accuracy. Careful consideration should be given to computational requirements and processing time,

when the goal is to capture hidden knowledge of the IoT device in terms of a real-time scenario [49].

In the last few years, researchers have investigated attacks launched against IoT gateways. IoT gateways are used to aggregate and communicate data from IoT devices to their corresponding servers for data processing. Due to the fact that IoT gateways are usually the connection between the actual device and the internet, they are vulnerable to a plethora of attacks (IP, wireless sensors, network), Denial-of-Service, and Denial-of-Sleep, just to name a few [50, 51]. Authors in [52] propose a sliding window bound mechanism to analyze the packet data through a dense recurrent neural network (DRNN). The DRNN can detect an ongoing attack based on packet information that is presented through the sliding window on a synthesized data set.

Authors in [53] investigate the effectiveness of deep neural networks in terms of the classification of malware in Android applications. They have proposed a sequential convolution neural network (CNN) and compared it to an LSTM and other deep neural network algorithms. They have categorized 8 different malware groups and used a data set of 1016 unique Android APK files. This resulted in a high accuracy for malware detection but a low accuracy for the classification of the malware group. They also have experimented with API sequence length and size of training set to show the change in accuracy of the different MLs. A similar approach has also been proposed by authors in [54, 55]

Nauman *et al.* [56] apply several deep learning models and compare these with the Bayesian machine learning model to predict Android malware. They utilize the DESTN data set to run their experiments. Their main conclusion is that different data set sizes provide different accuracy results for the many algorithms tested. Therefore, one can choose the corresponding machine learning algorithm that administers the greatest accuracy.

Authors in [57] address the challenge of identifying if an Android sample is malicious or legitimate. Through the design of a convolutional neural network, they

experiment on a real-world data set. The accuracy of their test data was in the 75-80% range. Due to their real world sampling, we consider the accuracy produced in their experimentation as a comparison to the experimentation documented in this Chapter.

There is considerable research that connects wireless networks and mobile cloud architecture to service-based architecture [58–61]. However, additional research is needed in terms of connecting the Network Functional Virtualization (NFV) service-based architecture to IoT devices. NFV is a network architecture that creates communication services by building virtual blocks to connect network nodes functions. In general, the 5G network is an important technology for the communication community. So, the usage of IoT devices as stated above is increasing exponentially and security risks are increasing with it. As stated in [62], there is a considerable research need to connect service based network to the communication technology of IoT devices. The research introduces one way of developing that into the IoT infrastructure.

Globally, there is a growing use of mobile components and with that we observe a rise in malware and intrusion to gain access to sensitive information. Exploring previous work gives an insight to numerous challenges that need to be investigated. The first being that the previous work is mostly focused on developing and tweaking algorithms for a specific data set. In many cases, the data sets that are utilized are small. Based on the literature surveyed, we chose to incorporate a RNN-LSTM due to the time sequence dependencies of the threats in IoT network data traffic.

3. AGGREGATED POPULATION MODEL FOR EPIDEMIC SPREAD

The system model we propose is composed of three blocks designated for different stakeholder needs. Each block is connected to the other in terms of how the prediction model is utilized. The full system model is shown in Figure 3.1.

The **Semantic Events block** is split into two categories counting based events and complex spatio-temporal events described below:

- **Counting based events** are predicted at each aggregated level to analyze the effects of the spatial component of prediction. The spatial component is diminished as the aggregation gets coarse. Analyzing certain effects due to aggregation are listed in Challenges 3 and 4 in Section 1.4.

- **Complex Spatio-temporal Events** are detected in the AB model level of aggregation, utilizing either Hidden Markov model (HMM) or the Petri Net graph model. In case HMM, each agent is classified into a specific state (pre-estimated from historical disease data). As for the Petri Net Model overall

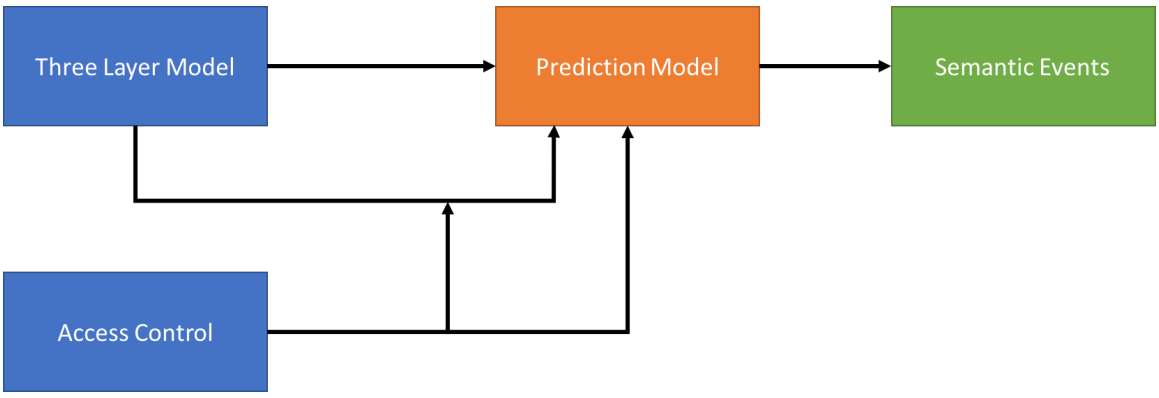

Fig. 3.1.: Aggregated Model System Architecture.

spatio-temporal events are detected. For example, direction of disease, emerging disease, disease decline, etc.

The **Access Control block** is what motivates the three-level aggregation block. Different stake holders of the system have different levels of access to the proposed system, creating a prediction model for semantic events that appear at all access control levels.

3.1 Three Layer Aggregation Model

The **Three Layer Model** is an aggregated structure of the population social interaction graph model. Aggregation a solution is computationally intensive. The three level aggregation model takes care of counting-based semantic events within a specified window of percentage accuracy.

3.1.1 Agent Based Model

Agent Based (AB) modeling has been applied to many different fields, such as traffic analysis, social networking, and economical studies, just to name a few [63,64]. Two types of modeling approaches have been used in previous research work; individual-based model and computational epidemiology model. The AB model allows spatial and temporal aspects of an epidemic visually. A major shortfall of AB model is that it can computationally intensive for graph driven analysis. Due to this, alternate models are presented in terms of granularity thus reducing the computational cost of the simulation.

The AB model can be used to simulate the disease spread. Using Figure 3.2 as a reference to the states, each agent can reside in a state with specific rules that are assigned for the agents to traverse from one state to the next. In our social network structure, population is divided in terms of three daily activities (home, work, transportation). Home activity refers to an agent being at their home and can

only interact with agents assigned to the same location (home). The transportation activity refers to an agent being on public transport and interacting with agents that are assigned the same bus. During the work activity, an agent is at its work location and can only interact with agents in the same location and at the same time. Each activity has a corresponding disease spread rule associated with it. Therefore, agents in different activities have different thresholds for transitioning from state S to state I.

The parameters used in such a model are shown in Tables 3.1 and 3.2. The contact rate is the probability that an individual agent comes in contact with another individual agent. The contact rate of agents is based on which activity they are in and each state of the two connected agents. A susceptible agent can be infected by either an exposed or infected agent. c_{ES} is the contact rate between an exposed and susceptible agents. c_{IS} is the contact rate between infected and susceptible agents. The same subscripts are also used for the infection rate. The infection rate is the probability that an individual is infected based on the activity and state of the other individuals. The heterogeneity factor which is defined as the strength of the interaction link between two agents is listed in Table 3.2. Each activity has a duration which has a uniform distribution. In addition, each agent is randomly assigned a heterogeneity factor from the given uniform distribution. A quick observation of Table 3.2 shows that the home activity has a lower factor than the transportation activity. This is because the close proximity of agents in transportation requires a higher heterogeneity factor. The infection rate, contact rates, and heterogeneity factors are all part of the infection rate equation that directs the transition of the agent from a susceptible state to an exposed state.

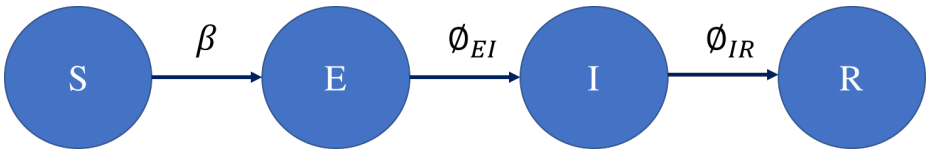

Fig. 3.2.: SEIR Finite State Model.

Table 3.1: Contact Rate for Proposed SEIR Model.

Contact Rate	$\mathbf{c}_{ES}$	$\mathbf{c}_{IS}$
Home	4	1.50
Work	5	1.00
Transport	3	2.00
Infection Rate	$\mathbf{i}_{ES}$	$\mathbf{i}_{IS}$
All Activities	.05	.06

Table 3.2: Heterogeneity Factor of the Proposed SEIR Model.

Heterogeneity Factor	δ
Home	U[0.25-1.00]
Work	U[0.05-0.25]
Transportation	U[1.00-1.50]

3.1.2 Spatial Based Model

The Spatial Based (SB) model can be viewed as an aggregated version of the AB model. The SB model divides the population spatially in a grid like structure. Maciejewski's *et al.* work creates the SB model to analyze the effect of decision measures when implemented for different infectious disease scenarios in Indiana counties [65]. The SB model is unique because it takes into account the distance factor between spatial grids in order to analyze spread of disease in terms of speed and direction. Since the SB model is developed by aggregating the AB model, its contact rates are computed in terms of grouping some of the parameters of the AB model such as age [65]. Each activity in every county has its own infection spread rate ($i_{u,a,t}$) taking the AB model's total transmission rate equation discussed in the previous section into account:

$$i_{u,a,t} = p_u * \rho_u, a * \omega_a. \tag{3.1}$$

where ω_a is the proportion of the population in a target activity (a). $\rho_{u,a}$ is where the AB model comes into play. $\rho_{u,a}$ is the disease rate modifier for activity (a) in a specified area (u) such as a county or town. It is be derived from equation 3.9.

$$\rho_{u,a} = p(x_u|\lambda) = \sum_{i=1}^{p_u} h_{u,a} g(x_i|\mu_{u,a}\Sigma_{u,a}) \tag{3.2}$$

The summation is over the population of a specific area (u) in a specific activity (a), with the Gaussian distribution being in terms of the activity and area identified. h_i is the heterogeneous factor of the specific area and activity. p_u is the number of individuals in that specific area.

3.1.3 Global Population Model

The Global Population (GP) model is a course granularity of the population. The GP model is a generalized version of the AB model that provides more efficient prediction strategy as the GP model takes into account only the infection rate and the contact rate of individuals. It does not take into account any attributes or network structure between individuals [66].

The rate equations for an SIR GP Model can be represented as:

$$\frac{ds}{dt} = -bs(t)i(t) \tag{3.3}$$

$$\frac{di}{dt} = bs(t)i(t) - ki(t) \tag{3.4}$$

$$\frac{dr}{dt} = ki(t) \tag{3.5}$$

where $b = \frac{S}{N}\beta$ is aggregated number of new infected cases.

Taking into account all activities and weighted heterogeneous factor calculating through the following equations:

$$s_n = s_{n-1} - bs_{n-1}(i_{n-1} + e_{n-1})\Delta t \tag{3.6}$$

$$i_n = i_{n-1} + (\varepsilon e_{n-1} - \delta i_{n-1})\Delta t \tag{3.7}$$

$$r_n = r_{n-1} + \delta i_{n-1}\Delta t \tag{3.8}$$

Through the above equations, it can be determined that the AB level model can be abstracted into a SB level model which can be further condensed into the GP level model. Each AB, SB, and GP model can be presented as a distinct finite state machine (FSM) behavioral model as shown in Figure 3.3.

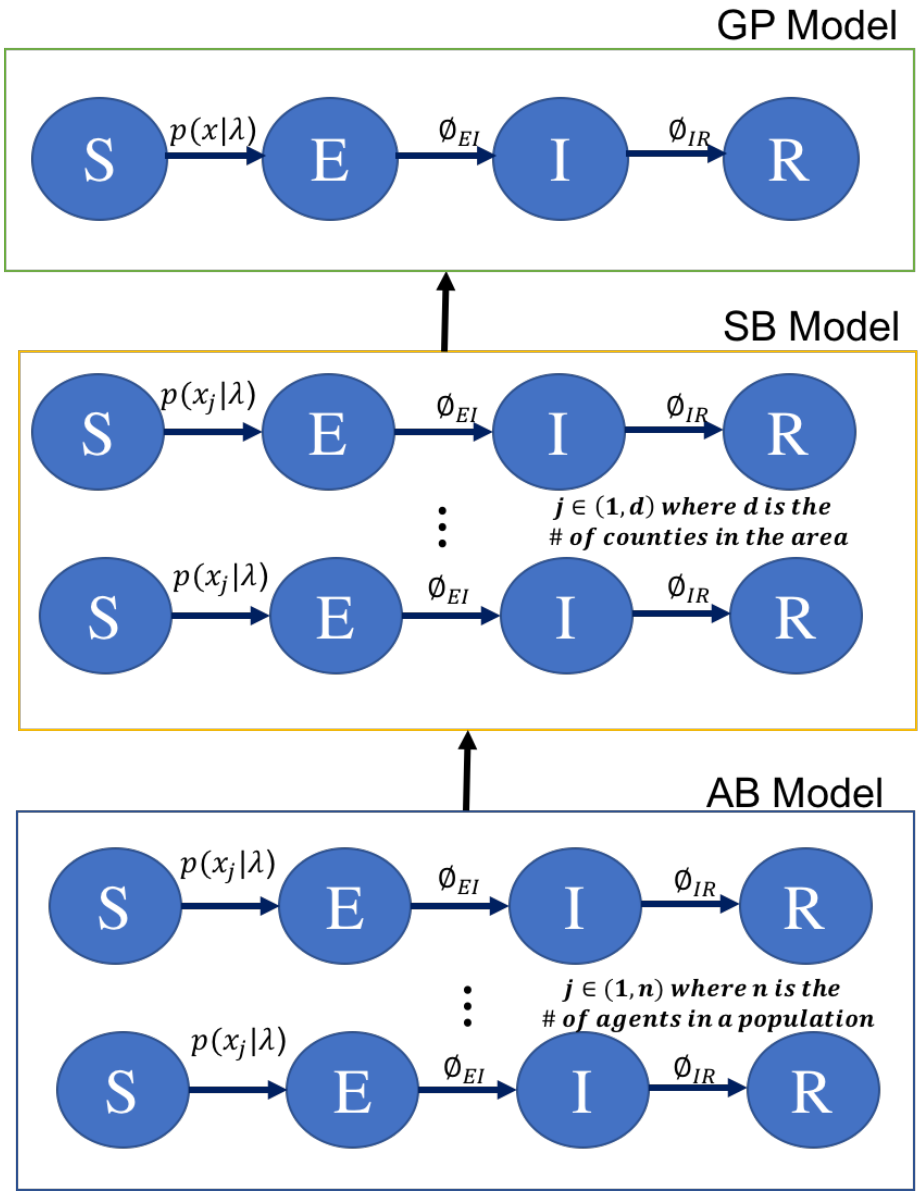

Fig. 3.3.: Aggregated Levels of the Socio-Temporal Network in terms of SEIR-FSM.

3.2 Disease Spread Model Simulation

Mapping disease spread models on network topology has been investigated thoroughly in the literature. But none of this work has dealt with spatio-temporal networks (dynamic networks). For dynamic networks, disease spread models need to be redefined by taking into account the variability in these networks. Creating these redefined disease spread models provides health care stakeholders with a better understanding of how to predict the spread of disease.

Disease spread models are simulated to depict the effect of epidemic under various epidemiological rules. As shown in Figure 3.2, an SEIR/SIR model is represented as a four stage finite state model (FSM) for each individual (represented as an agent) in a general population. In the susceptible state (S), an agent is prone to get infected at any point in time dependent on the spread rules and network dynamic. The second state, exposed state (E), depicts an individual who does not show any symptom of the disease but can infect others. The third state is the infection state (I). After a specified time period, an individual recovering from the infection enters into the recovered state (R). The transition from the I to the R state depends on a combination of time passed since exposure and the number of agents connected/intersected to that specific individual.

The SEIR model can be further classified into probabilistic and deterministic spread models. In a deterministic model, each individual/agent traverses one state to the next in a fixed amount of residency time when all infection criteria are met. The probabilistic models have been developed to more accurately capture the dynamic aspects of an epidemic. In these models, the residency time of each state follows a probabilistic distribution providing a more realistic disease spread model. The main objective of both deterministic and probabilistic SEIR models is to analyze various decision making choices based on the behavior of the disease that encapsulates numerous factors: the maximum infected population, the peak time of the disease, and the final percentage of the infected population [67]. Such analysis provides stakeholders

with a better understanding on how different network structures behave in terms of network connectivity.

The existing disease spread models such Susceptible-Exposed-Infected-Recovered (SEIR) are subsequently incorporated with the social network model to analyze the dynamics of the epidemic under various context. The same models can be used to analyze spread of viruses in communication networks.

The objective here is to present an agent-based spatio-temporal statistical epidemic model (graph data model) depicting interactions among a large scale population. To model this interaction, a unified SEIR model has been developed.

3.3 Developed SEIR-FSM AB Model

In the dynamic social network structure being simulated, an agent has four paths to transfer from the susceptible to the exposed state. A unified SEIR-FSM is developed to take into account an activity based social network. The transmission rate (β) of an agent from the S state to the E state is calculated through a cumulative equation of exposed and infected contacts. First, a Gaussian Mixture Model (GMM) is utilized below to create one unified path from the S state to the E state.

$$p(x|\lambda) = \sum_{i=1}^{M} \omega_i g(x|\mu_i, \Sigma_i) \tag{3.9}$$

The summation equation shows the probability of infection due a number of different activities. M is the number of activities within one process. In our case, there are four activities: home, transportation to work, work place, and transportation towards home. In the AB model, weights for each activity are specific to each individual. Weights can also be dependent on the heterogeneity factor of the given network graph.

Once each agent's probability is calculated through equation 3.9 , then β can be calculated as follows:

$$\beta = \delta_j[i_{ES}E_c + i_{IS}I_c] \tag{3.10}$$

Where E_i is a random variable that follows a Poisson distribution with rate λ_i. Let $E_{i1}, E_{i2}, ..., E_{in}$ be the number of contacts in state E of activity i per susceptible individual. E_c is the sum of the E_i random variables. I_i is a random variable that follows a Poisson distribution with rate λ_i. Let $I_{i1}, I_{i2}, ..., I_{in}$ be the number of contacts in I state of activity i per susceptible individual. I_c is the sum of the I_i random variables. The transmission rate ($beta$) of an individual is then calculated as shown in equation 3.10. This differs from the GP model with respect to the heterogeneity factor (δ_j) for the transmission rate. In the proposed unified SEIR, the transmission rate has to take into account the heterogeneity based on the different activities of the social network. The GP transmission rate contains one uniform distribution among the entire population for the heterogeneity factor. Since the social network changes in time and the contact rate changes due to the heterogeneity factor the GP model does not show these nuanced network characteristics. Note given $I_c, E_c, and \delta_j, \beta$ can be calculated as:

E and I are the number of exposed and infected individuals, respectively. Then parameters are calculated using a Poisson distribution. δ_i is the heterogeneity factor for each activity as listed in Table 3.2.

3.4 Experimentation and Results for a Synthesized Population

In general, the higher the clustering of a network, the earlier the burn out of the spread. Otherwise, there are few differences that can be concluded in the networks when checking at GP and AB model simulations. The major contributor of the difference in metrics is the high or low clustering of nodes. The heterogeneity and main structure of the network do not have a great impact on the outputs of the disease spread.

The AB Model has exact data analysis of all states in any SEIR model that is implemented. The one drawback is that it is computationally intensive. This can hinder the effect of decisions when time is a crucial factor.

As mentioned earlier, the drawback of the SB Model is that the granularity becomes coarser making specific disease queries unattainable. Specifically, the activity based queries that are important in a spatio-temporal graph model. The SB model instead uses spatial distance between counties and does not focus on the individuals' day-to-day activities.

The GP models' lack of granular information is the hindering factor to produce specific output. Certain healthcare decisions can not be made based on specific characteristics of the population. Since the GP equations focus on the overall counting-based semantic events.

Table 3.3: Estimated Disease Parameters (b, k).

	I_0	b	k
Scenario 1	600	1.24	0.24
Scenario 2	153	0.848	0.099
Scenario 3	146	5.8e-07	9.79e-04

Using the AB model simulation data discussed in section 3.1.1 and the GP equations mentioned in section 3.1.3, we analyzed the data from both AB and GP perspectives. The data is from a synthesized population data set developed in [68]. In Figures 3.8 and 3.9 it is seen that the GP model data has a more gradual increase in the infection of individuals and peak infection time is a week later than the AB models' peek infection time. This corresponds to the susceptible graph shown in Figure 3.8 where the rate of decrease in the GP model is smaller than that of the AB model. These results are just a preliminary analysis of the GP model approximation using the Eulers' method of an SEIR model.

The experiments were run on three different scenarios. Scenario 1 implements the disease spread when the initial infection (I_0) = 600, transmission rate (b) = 1.24 and (k) = 0.24. Scenario 2 applies the paramteres listed in Table 3.3 to convey the disease spread of a more densely interconnected network. Scenario 3 represents the

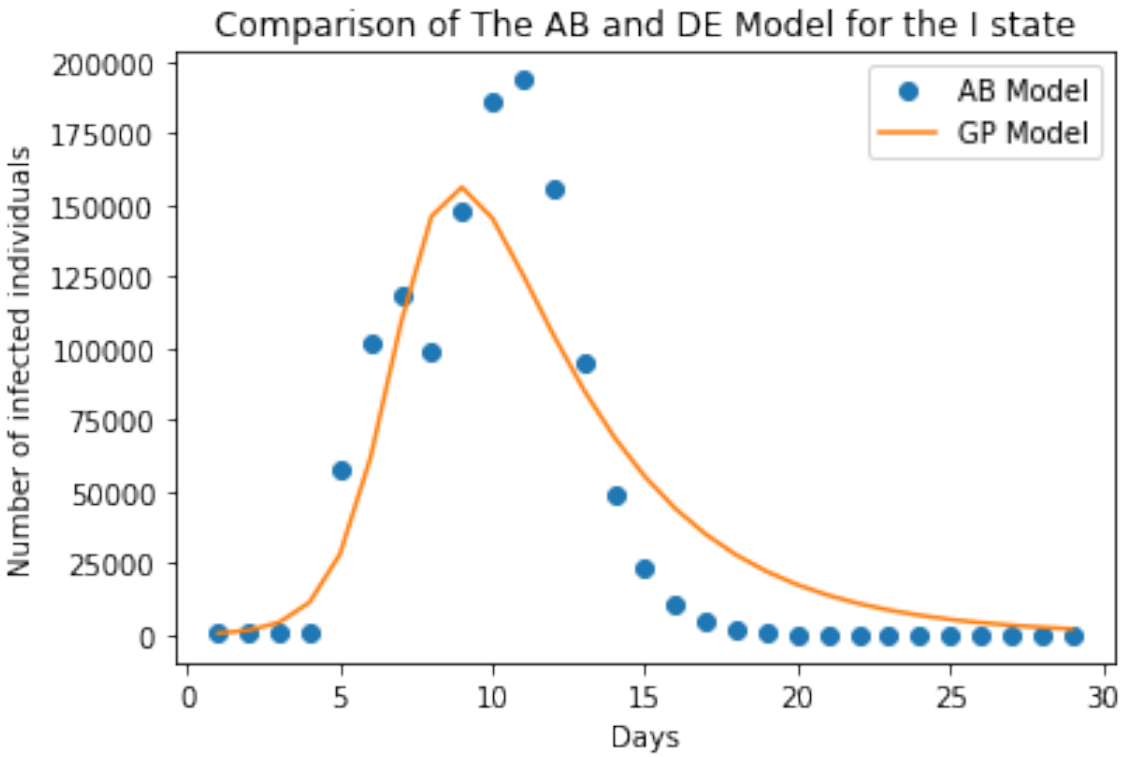

Fig. 3.4.: Number of Individuals in state I in Scenario 1(GP Model vs AB Model).

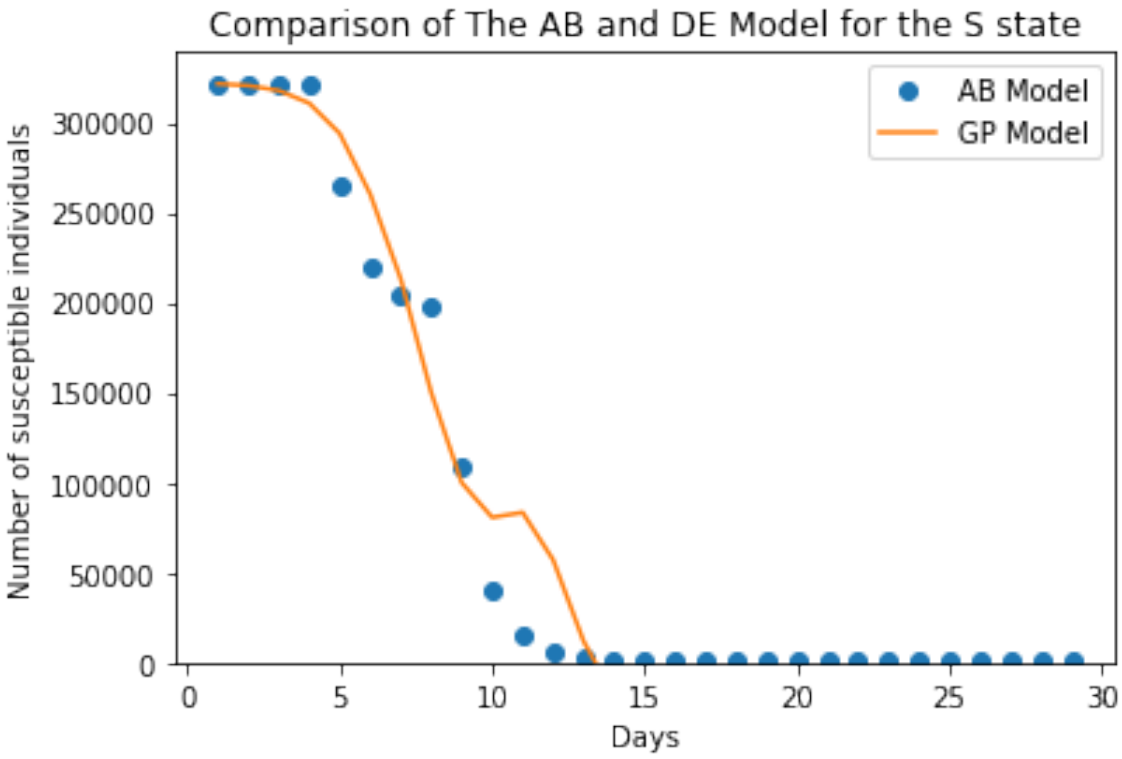

Fig. 3.5.: Number of Individuals in state S in Scenario 1(GP Model vs AB Model).

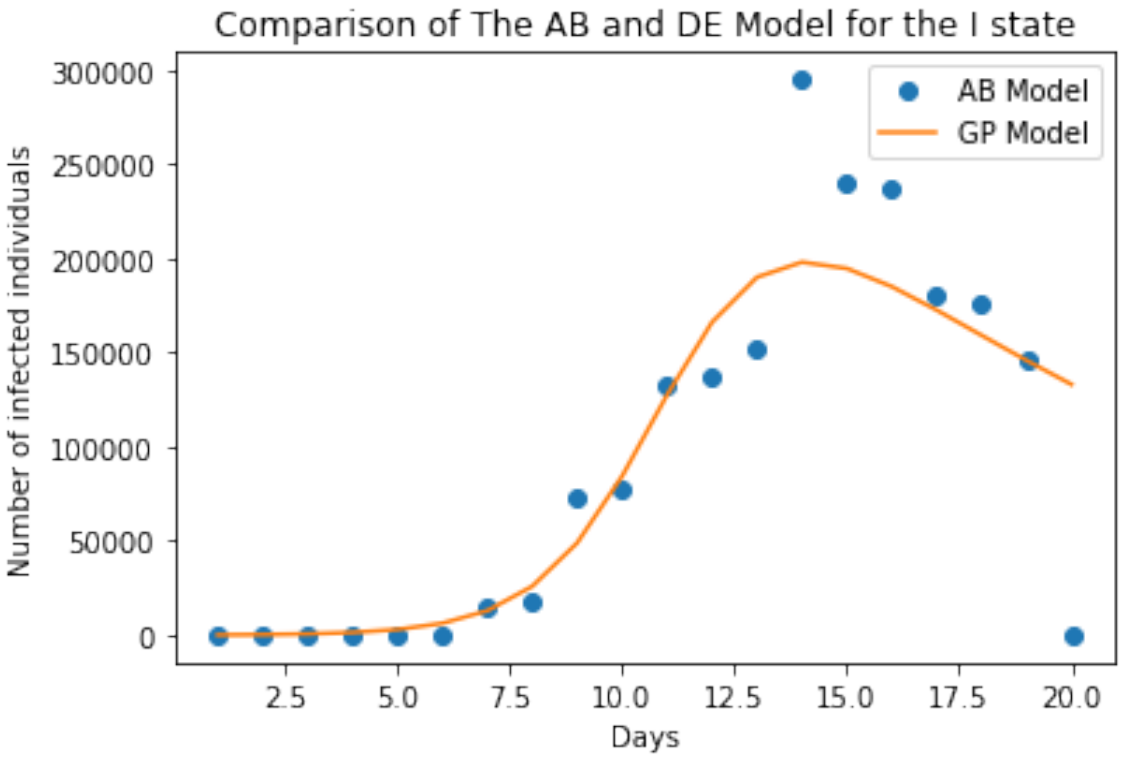

Fig. 3.6.: Number of Individuals in state I in Scenario 2(GP Model vs AB Model).

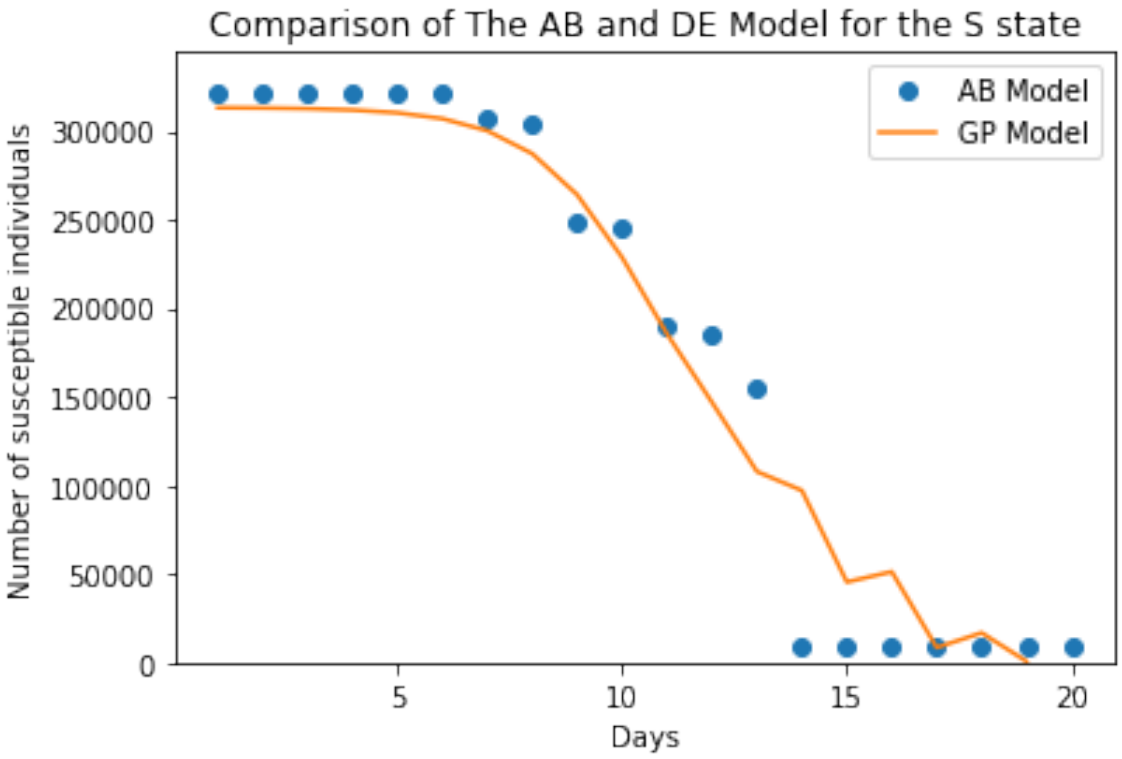

Fig. 3.7.: Number of Individuals in state S in Scenario 2(GP Model vs AB Model).

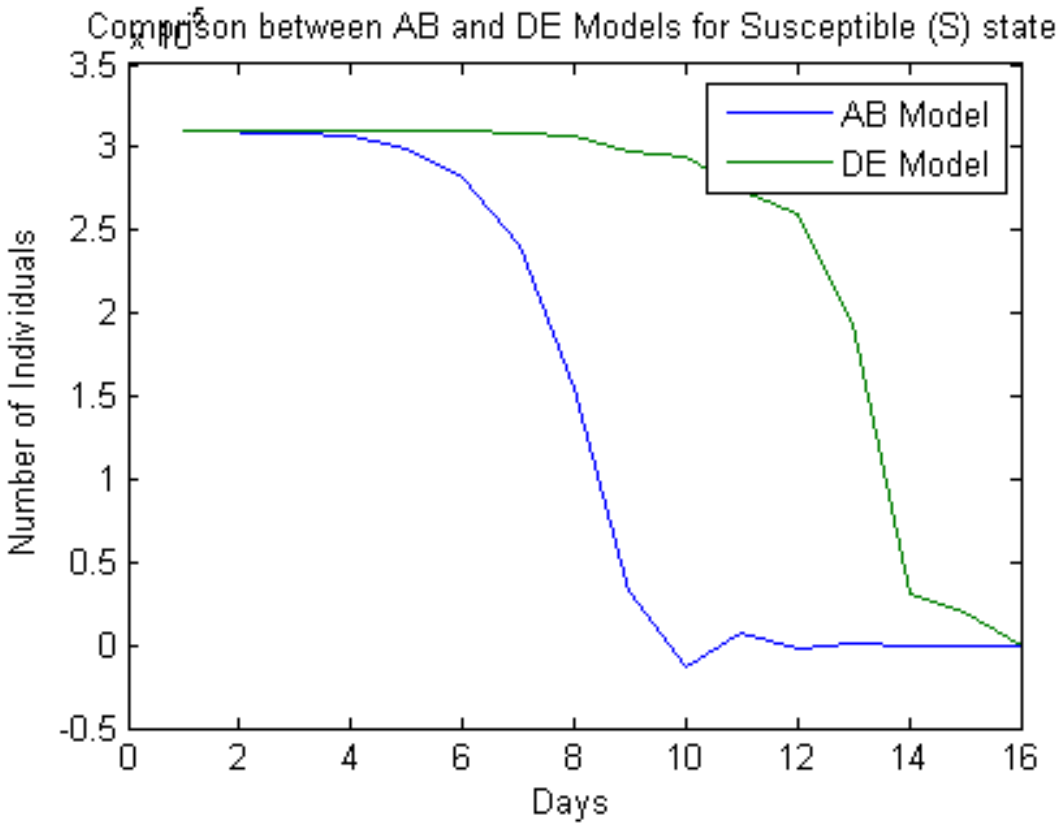

Fig. 3.8.: Number of Individuals in S state (GP Model VS AB Model).

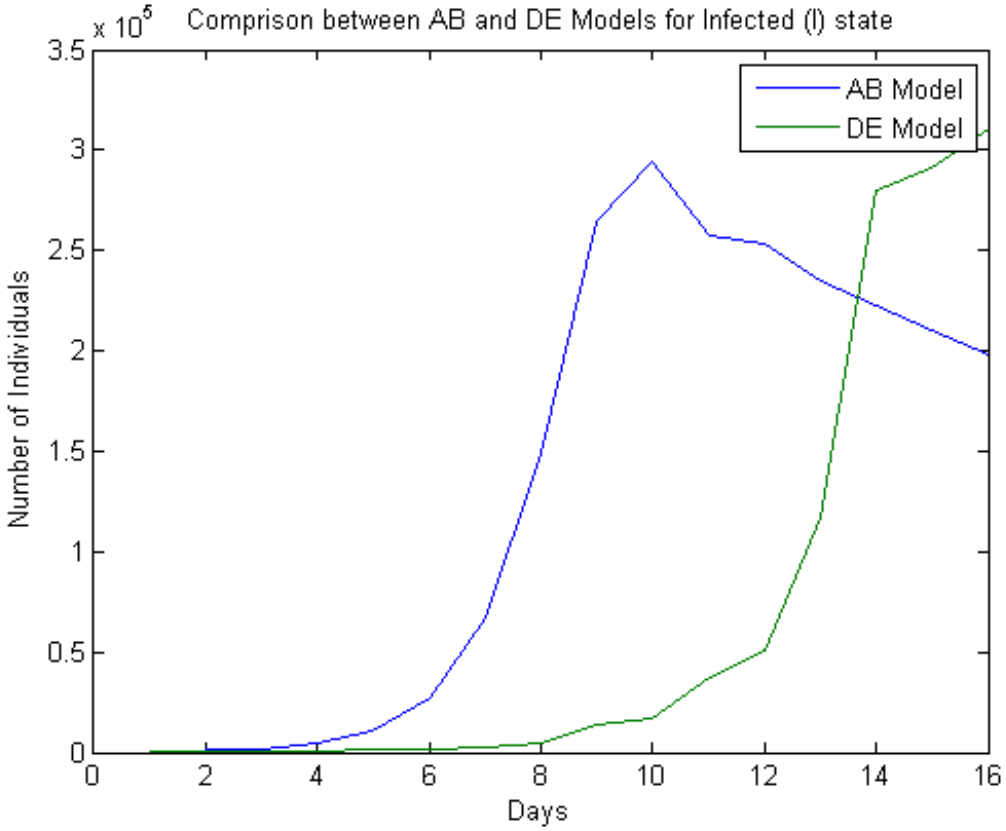

Fig. 3.9.: Number of Individuals in I state (GP Model VS AB Model).

parameters of a network with a smaller initial infection and transmission rates that are very low to convey the spread of disease on a sparse network.

Figures 3.4 and 3.5 correspond to Scenario 1 in Table 3.3. Note that the GP model closely follows the trend of the AB model both for the individuals in state S and state I. Scenario 2 is illustrated in Figures 3.6 and 3.7. This scenario has a smaller initial infected population. The initial infection for Scenario 2 was selected from a less dense area than Scenario 1. Accordingly, Scenario 1 has an early peak time of 10 days, whereas Scenario 2 has a peak time of 13 days. Due to the selection of this initial infection, the estimated transmission rates b and k for Scenario 2 for the GP model are slightly smaller. These results conclude that both the size and the location of the initial infection play an important role in terms of the dynamics of the disease spread.

In Figure 3.8, it can be observed that the GP model curve underestimates the number of susceptible individuals after day 6. This is because, in the AB model, there is a weekend phenomena, where a high increase in susceptible agents come into contact with infected agents. On the other hand, the GP model does not take into account this phenomena. This is also reflected in Figure 3.9, the GP model does not take into account the weekend phenomena creating a late infection peek time. This shows that an AB model is more accurate when simulating time-driven social networks. A GP model can not accurately predict spread when simulating time driven activity based SEIR-FSM.

In figures 3.10 to 3.15, we look into the prediction when access to all agents has been restricted. It shows that direct access control and aggregation decreases the accuracy of prediction even further. This displays the outcome of how the access control block affects the three level aggregated system model.

Our objective was to analyze the dynamics of this model and the impact of the network topology on the behavior of the spread. SEIR model and its multilevel aggregation representation have been applied and results show that the three level

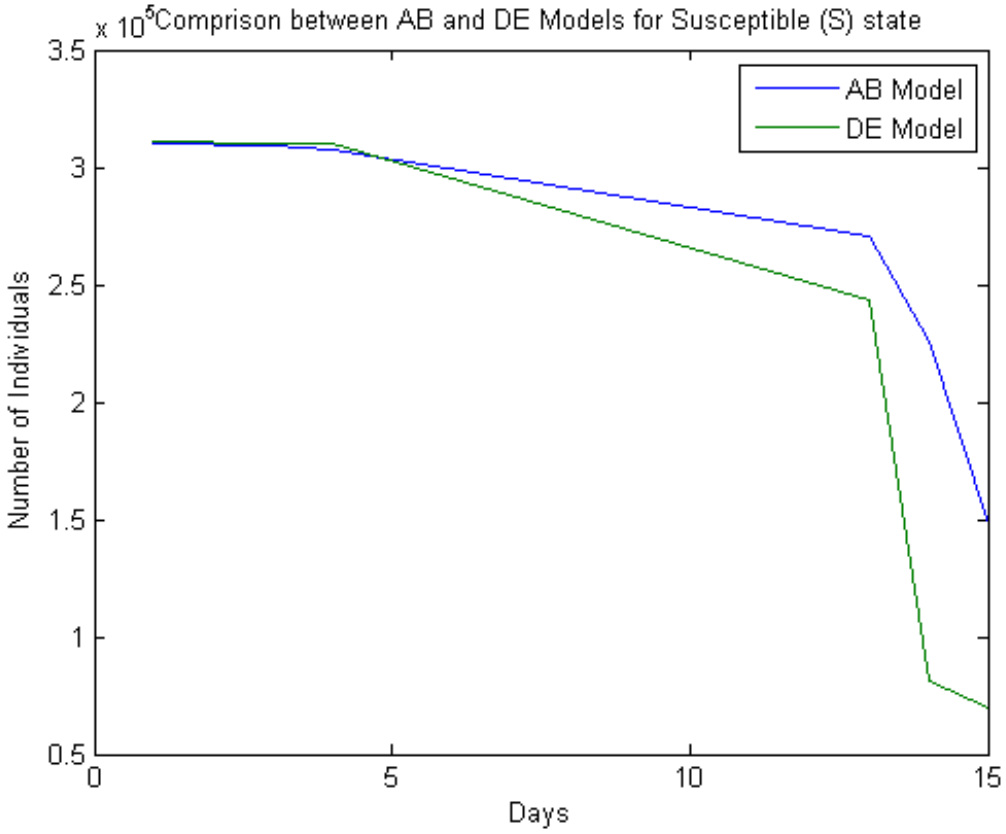

Fig. 3.10.: Number of Individuals in S state (GP Model VS AB Model)-R1.

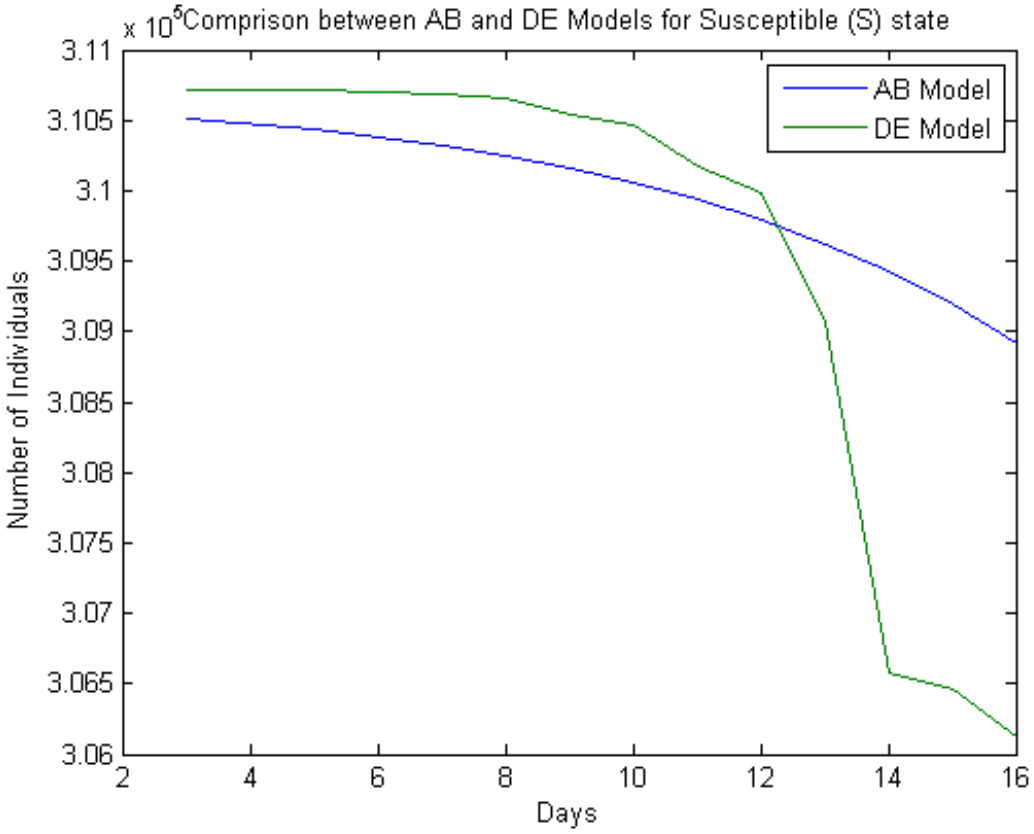

Fig. 3.11.: Number of Individuals in S state (GP Model VS AB Model)-R2.

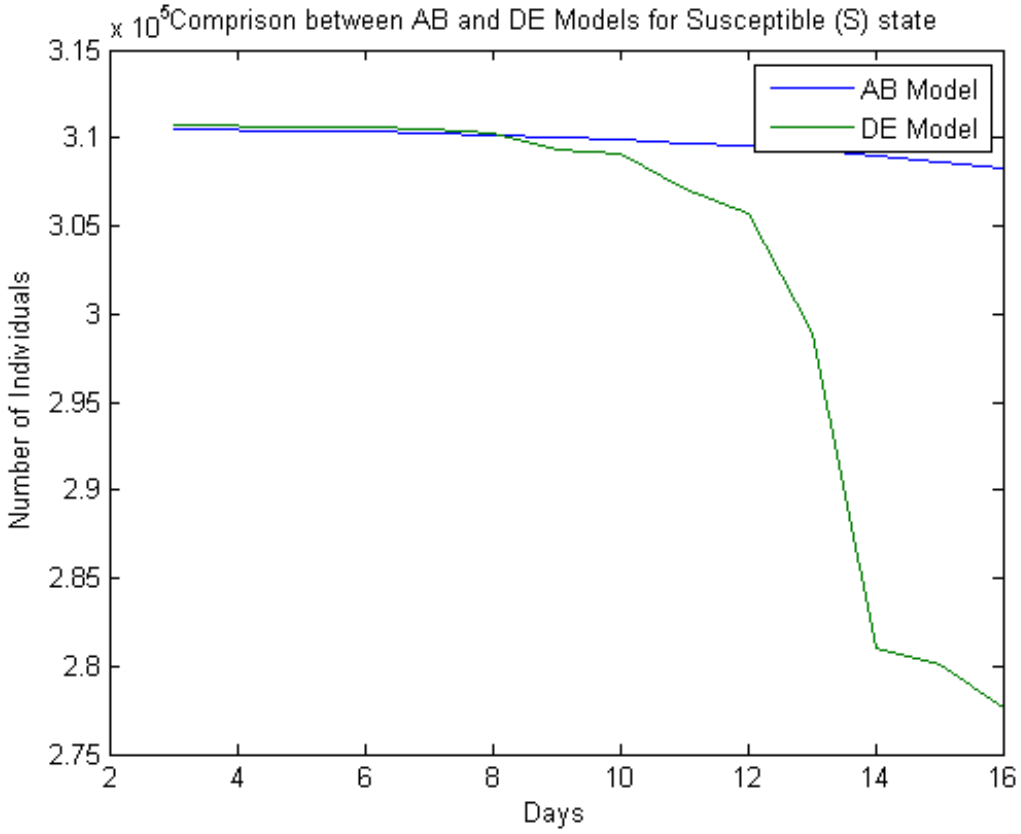

Fig. 3.12.: Number of Individuals in S state (GP Model VS AB Model)-R3.

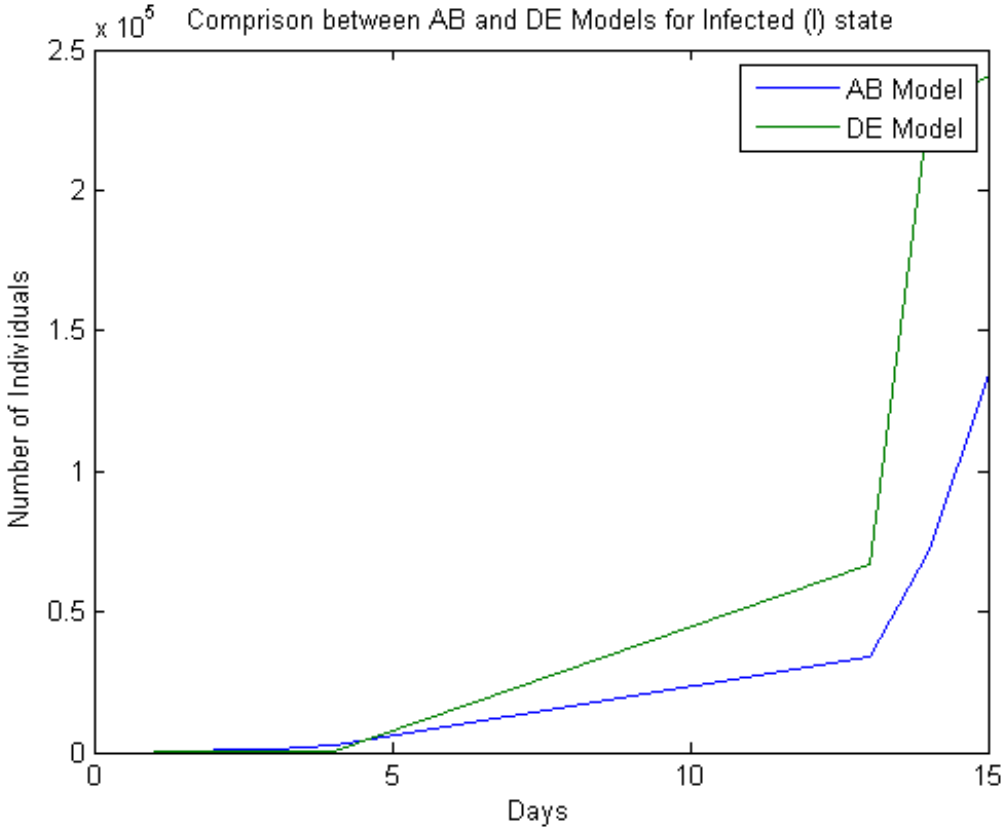

Fig. 3.13.: Number of Individuals in I state (GP Model VS AB Model)-I1.

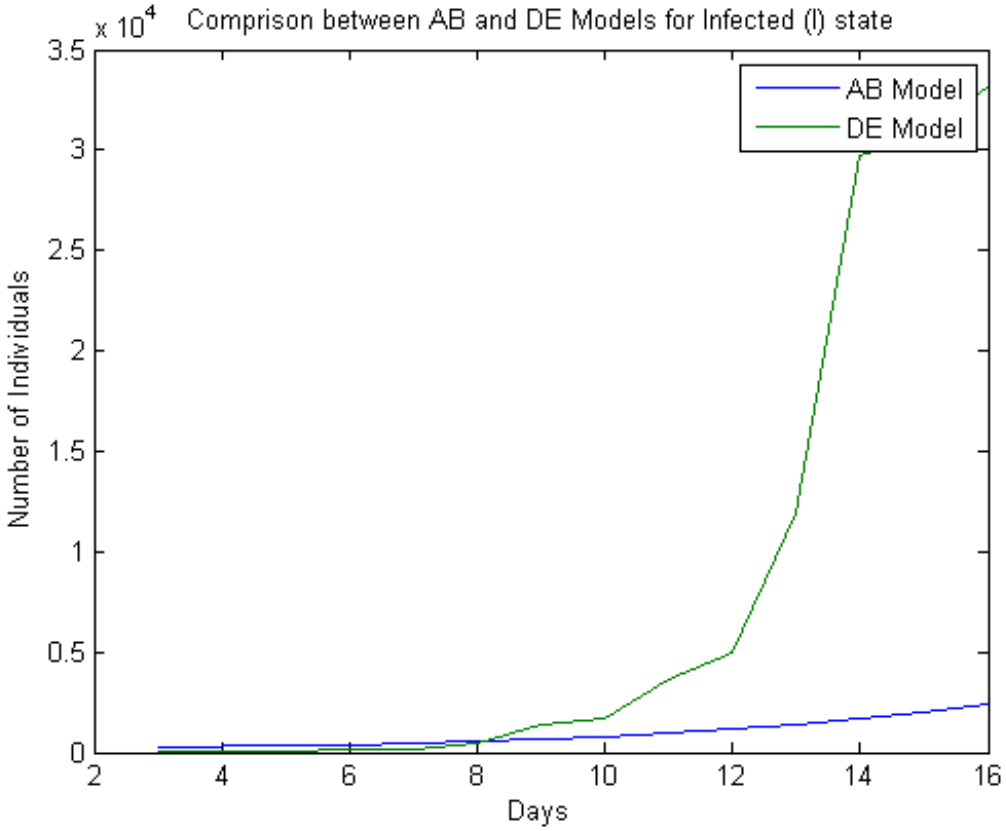

Fig. 3.14.: Number of Individuals in I state (GP Model VS AB Model)-I2.

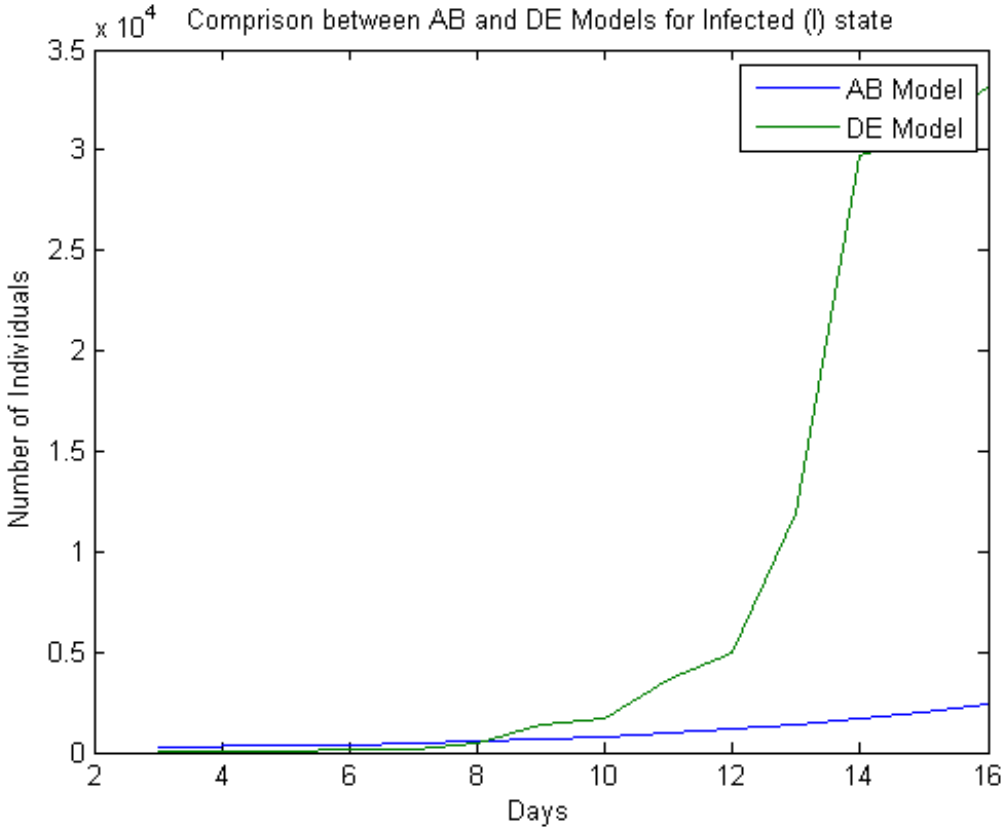

Fig. 3.15.: Number of Individuals in I state (GP Model VS AB Model)-I3.

access model has a varying degree of prediction dependent on what information is provided.

4. APPLICATION OF MANET

A Mobile ad hoc Network (MANET) is a collection of mobile nodes that are connected to each other through wireless links. Any two nodes within the network can communicate directly if they are within a certain range. Nodes that are not in the same range can communicate indirectly through intermediary nodes, as shown in Figure 4.1. Generally, mobile nodes in MANET do not have any centralized controller, such as base stations or mobile switching centers. In addition, there are no restrictions on the nodes to join or leave the network. Therefore, the MANET topology dynamically changes. Note that the underlying communication media can based on different technologies such as bluetooth and WiFi. These characteristics of MANET devices make it possible to effectively apply the epidemic spread model discussed in the previous sections to analyze the issue of information dissemination or virus spread in a MANET. Moreover, the nodes in MANET can be frequently and temporarily disconnected changing the number of neighboring devices. This in turn changes the contact rate among MANET devices. Note, the larger the number of neighboring devices, the higher the contact rate and the higher the possibility of spreading the virus. Musolesi and Mascolo [69] present a middleware that implements epidemic information dissemination techniques for ad hoc networks. They have proposed a primitive middleware for unicast and anycast communication and have used the epidemic model to measure the reliability of these two communication paradigms. They implemented SIR model where an individual (host) is infected if it carries a virus message and susceptible if it does not. In this case, the initial number of virus messages is set to 1 ($I_0 = 1$), for which the transmission rate (β) and removal rate (γ) are computed. β is computed through the corresponding contact and infection rates. γ represents the time it takes for an update message to reach the infected device. In

particular, the infection rate between devices can be predicted at three granularity levels discussed in the previous chapter (Figure 3.3).

The experimental results illustrated in Section 3.4 can be extended to analyze the phenomena of such spread for the MANET. For the AB model, each mobile device can be seen as an agent and the connections among agents can be monitored to detect any virus message. For this model, we characterize the agents based information including the location and time of infection, which is equivalent to the human demographic information presented in Section 3.4. For this information, we utilize Traffic Flow Table (TFT) and a Traffic Information Table (TIT) maintained by each mobile device. These tables are used for two purposes. First, they provide connectivity information among nodes. Second, the TFT can be used to identify infected agents through characteristics such as packet drop, buffer overflow, bandwidth consumption, etc.

SB aggregation method can be used to model and predict virus attacks and to create effective mechanisms to disseminate updates needed to halt the spread of the attacks. Mobile network connections are created through distance metrics. A mobile node sends a virus message within an RF range and the message is received by a neighboring node with a certain probability (p) as discussed in Section 3.4. In other words, p represents the probability with which a mobile device can be exploited depending on the RF range.

The GP model estimates the total number of mobile agents that are infected versus the number that remains susceptible at a global level. As previously stated, the GP model does not provide any specific information about when and where the agents are infected. It provides an estimate of the number of devices that are attacked given some initial number of attacking/infected devices. Based on AB, SB, and GP models a structured security mechanism for MANET devices in presented in Section 4.2.

4.1 Disease Spread Rules in Ad hoc Mobile Networks

In order to apply the epidemic model to MANETs, we utilize the concept of the Common Vulnerability Scoring System (CVSS). The CVSS provides an open framework to characterize device vulnerabilities in terms of different virus attacks. The CVSS has three metric groups which determine the base score in terms of the degree of vulnerability [70]. The virus spread rules between the SEIR States are defined in terms of the basic CVSS metrics. In state S, a device is vulnerable to receive a virus message. A device in state E carries the virus message that can be transmitted to other devices while the device itself is not affected. In state I a device is infected by the virus message. A device in state R implies that the device is recovered after receiving a "clear virus" update message. The CVSS metrics are used to compute the contact rate, infection rate, and heterogeneity factor for the MANET and in turn, the probability of infection for a device to transition from state S to E. The basic CVSS metrics are: the access distance (access vector), number of times contacted (authenticity), and strength of connectivity (access complexity) at which a mobile agent can be reached. The contact rate is generally a function of the access vector and authenticity metrics, that is:
Contact rate $= f$(access vector, authenticity).
High level of authenticity denotes that an agent while trying to connect to another agent needs to go through several steps of authentication. Access complexity corresponds to the heterogeneity factor (δ) of the MANET device which represents the number of devices that can make contact with any given mobile device. Accordingly, the probability of a mobile getting exploited for vulnerability, that corresponds to the transition from state S to E, can be calculated as given in [71]:

$$CVSS_E = \tfrac{1}{10} * 20 * AV * AC * Au \tag{4.1}$$

where AV is the Access Vector, AC is the Access Complexity, and Au is the Authentication level.

Using the above equation and its parameters, the impact of various attack scenarios can be analyzed for MANET. For example, for a virus spread attack scenario in MANET akin to Scenario 1 discussed in Section 3.4, the average transmission rate $b = CVSS_E * 10 = 1.23$ based on the values of $AU = 0.45$, $AV = 0.395$ and $AC = 0.35$, which are taken from the CVSS database [71]. Referring to Figure 3.4, the 322,000 agents be viewed as mobile devices in a MANET which exploits vulnerabilities to attack and "infect" each other at rate b. In this case, the peak number of infected mobile devices can be estimated in day 10 based on the AB model and on day 8 based on the GP model. Accordingly, one can deploy a patching mechanism using the recovery rate (k) by sending a "clear update" message to infected devices in a timely manner. This timely response can avoid the patch/virus race condition [72].

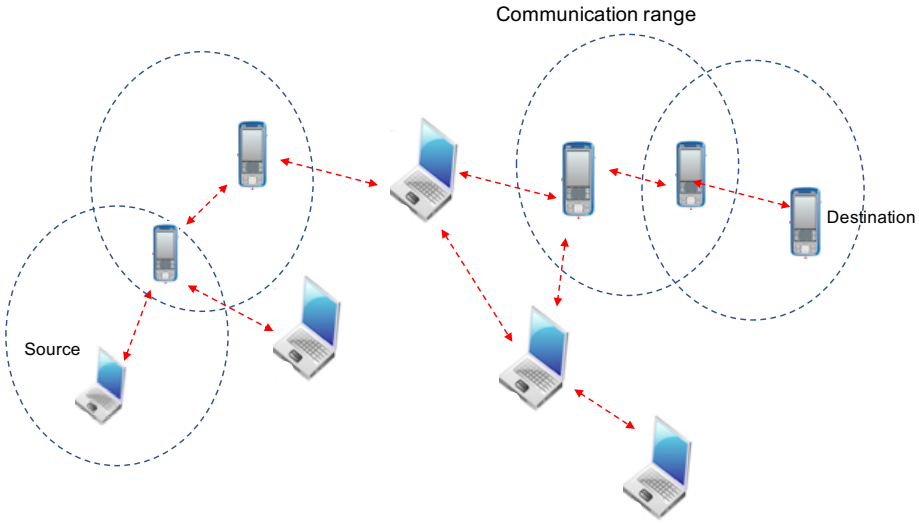

Fig. 4.1.: Example of MANET network.

4.2 Intrusion Detection System Architecture for MANET

Fixed or mobile networks play an increasingly vital role in modern day society, however, the security of these networks is always at risk. Therefore, there is a dire need to develop security assurance technology to protect these systems. Intrusion

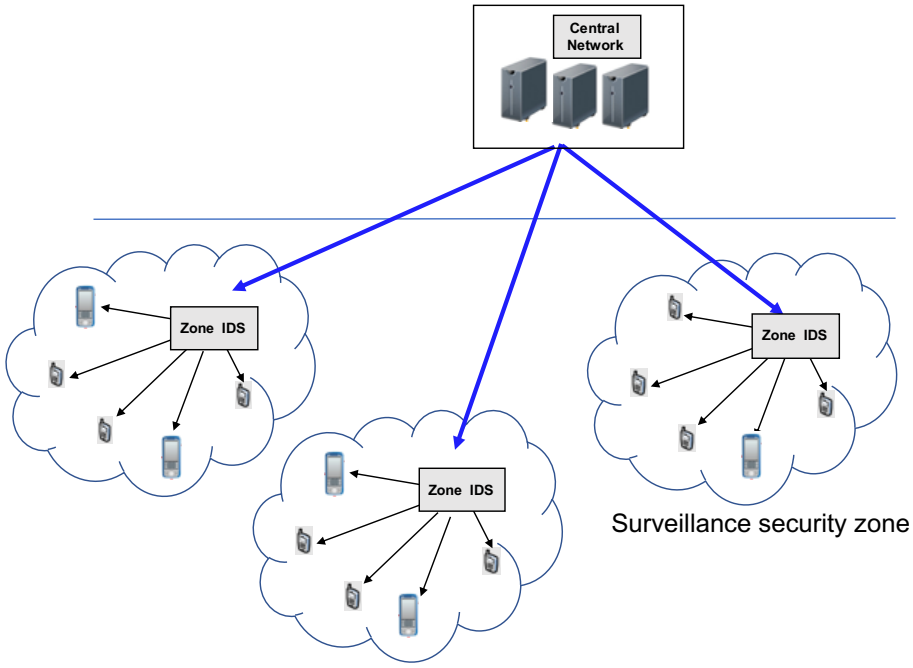

Fig. 4.2.: Semi-distributed hierarchical Security Structure based on IDS surveillance zone deployment in MANET.

Detection System (IDS) is one such technology with the aim to detect security threats and attacks in the network. For this purpose, IDS monitors network traffic and generates relevant alerts when malware and traffic anomalies are detected [46].

The deployment of an IDS is an important design aspect in computer networks, and more challenging for MANET because such a network has a dynamically changing topology with no centralized management [46]. Based on the unique characteristics of MANET, one strategy to control spread of virus is to deploy an IDS at each device which is akin to the AB model. In this case, the IDS is part of the mobile device software to detect active intrusion attempts. This software can in turn communicate with a centralized threat management system so that "clear virus" updates can be broad casted to the infected devices. Note, such an AB IDS deployment strategy could be costly in terms of energy and memory constraints. In addition, using AB IDS approach does not scale well for the centralized threat management system and cannot provide effective response to prevent such spread.

In this section, we propose two types of security architectures for monitoring and generating updates for virus spread attacks in a large scale MANET. We use the concept of multiple aggregated levels similar to the concept of SEIR model introduced in Section 3.4 to develop a hierarchical security structure as shown in Figure 4.2 which can have two different architectural designs: fully distributed versus semi-distributed. In a fully distributed architecture, the IDS is deployed at the AB level that communicates alerts to a surveillance zone (SZ) monitor that collects these alerts and in turn reports to a centralized global security monitor. Alternatively, in a semi-distributed architecture the IDS is deployed at the SZ level rather than at the AB level. In this case, an SZ IDS monitors traffic and collects TFT and TIT information to detect infected devices. Consequently, this architecture does not suffer from the power and processing limitations discussed at the AB level.

For both architectures, SZs are identified based on the physical limitations of RF transmission range among ad hoc devices, and the density of these devices in a certain area at a given time. Taking into account both, the network density and

the RF range, this architecture can provide a scalable solution for monitoring the vulnerability and the processing network traffic within each zone. Note that these zones can be separated or overlapped. For both architectures, the SZs report to a centralized manager that is responsible for communicating "clear virus" updates to all other SZs. This message is then distributed to devices in affected SZs.

The centralized security network system represents the aggregation at the GP level, in the sense it aggregates intrusion information from all IDSs. This system is ultimatly in charge of generating and disseminating any global counter measure, such as "clear virus" update message to all or specific SZs that have been "infected". Graphs similar to those shown in Figure 3.4 can be applied to MANET networks to identify threshold levels at which decisions related to "clear virus" update is communicated by the semi-distributed security architecture.

4.3 Probabilistic Virus Spread Model

This section describes the proposed HMM-FSM for categorizing mobile nodes into 1 of the 3 classes that are predicted. The use of the CVSS base score as an observable state is explained and in turn its connection to the four hidden states of a mobile is discussed.

4.3.1 Proposed Model

In the proposed HMM model, the observation states are a continuous sequence of the CVSS scores for a mobile node that changes with time. The time is dependent on when and how the IDS scans for the CVSS scores of each device. In our experimentation, the time is based on a daily calculation of the CVSS score. The CVSS numbers are decided through a combination of common vulnerability characteristics. Having 3 categories each with different levels creates around 1296 combinations for CVSS base score computation. Equation 4.2 calculates the base score given the three major

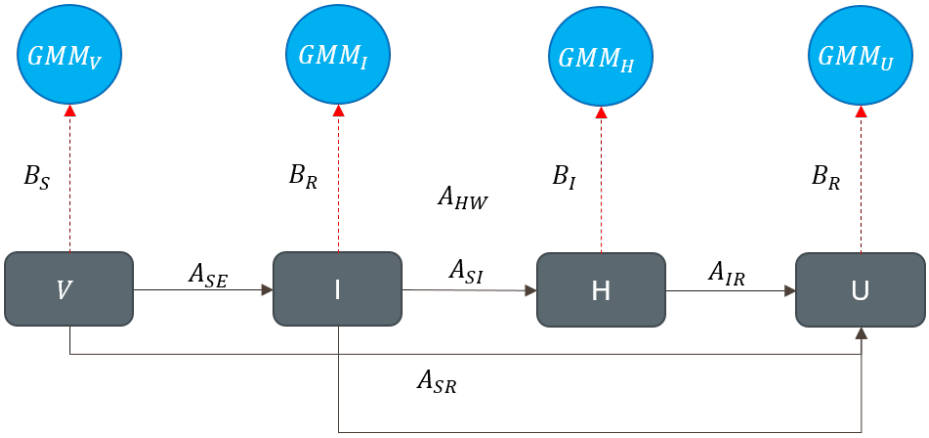

Fig. 4.3.: Mobile Virus Finite State Model.

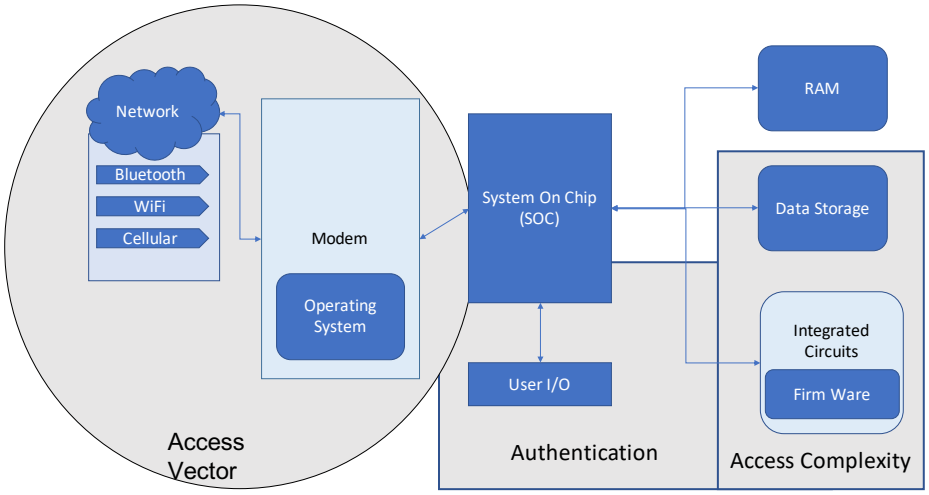

Fig. 4.4.: Mobile Architecture with CVSS metrics.

categories. Each hidden state has a Gaussian distribution probability associated with the CVSS score.

The CVSS has three metric groups that allow for a proper vulnerability score. The base metric group covers the characteristics of a vulnerability that is constant through time and across user environments. The Access Vector (AV), Access Complexity (AC), and Authentication (Au) metrics are the three base metrics that measure the exploitability factor of a mobile device. The AV refers to the level of access the

user requires to exploit the device. The more remote the accessibility, the higher the exploitation of the device. For example, the exploitability factor a LAN connection is lower than that of a wireless connecting device. Looking at it from the perspective of the population disease spread, it would correspond to the location and activity of the node. The AC metric measures the complexity of the attack required to exploit the vulnerability once an attacker has gained access to the target system. For example, consider a buffer overflow in an Internet service: once the target system is located, the attacker can launch and exploit at will. The AC equates to the heterogeneity factor which represents how many other devices (contacts) can be attacked once having access to any given device. The Au metric measures how many times credentials are needed from the attacker to have access to the device. This metric equates to the contact rate in the disease spread equations. The high level of authenticity denotes that an attacker has to go through several authentication steps to gain access and infect the mobile device. The full CVSS score also takes into account the impact factors, these metrics account for the mobile device's impact to itself and the overall network once the exploitability has been a success. The three impact factor are: Confidentiality (C), Integrity (I), and Availability (AI). The Confidentiality impact factor describes the impact on personal and secure data that should only be accessed by the appropriate user. High confidentiality impact increases the vulnerability score of the CVSS. The Integrity factor refers to the veracity of information that has been compromised. The Availability factor increases when the vulnerable device has access to other aspects of the network, such as bandwidth, processor cycles, or disk space. Equations 4.2 and 4.5 are both used to calculate the observation sequences of the MANET devices [71].

$$CVSS_E = \frac{1}{10} * 20 * AV * AC * Au \tag{4.2}$$

$$CVSS_I = 10.41(1 - (1 - ConfImp) * (1 - IntImpact)) \tag{4.3}$$

$$CVSS_B = 0.6 * CVSS_I + 0.4 * CVSS_E \tag{4.4}$$

The research to create the hidden states has been done in terms of what characteristics that are specific to some of the viruses of a mobile node mentioned in section 2.4. The applied hidden states describe a mobile node's behavior completely. Four hidden states that cover the observation states metrics were produced:

1. *Vulnerable*: a mobile node that is vulnerable to attack from any of the observable CVSS components that correspond to that specific mobile phone or technology.

2. *Infected*: Already having the attack administered in the mobile itself

3. *Host*: A mobile node that has the ability to spread the attack through specific CVSS characteristics

4. *Updated*: A mobile node that has been given the "vaccine" so as not to accept any attacks that have been detected in the network.

The emission probability matrix is based off of Table 4.1 where the connection between the observable (CVSS) and the hidden states (disease spread state) is made.

$$ImpactScore = 10.41*(1-(1-ConfImpact)*(1-IntegImpact)*(1-AvailImpact)) \tag{4.5}$$

Seen in the state diagram of Figure 4.3 which also includes the corresponding probability index for the transition and emission matrices that are included in Section 4.4.

The data set as mentioned earlier is a sequence of mobile CVSS base scores. These are populated from a Deter network topology. We train several classes of HMM. Each class of HMM has the same observable and hidden states but each with a corresponding transmission matrix (A) and emissions matrix (B). Each class was trained on 1000 mobile agents, each agent having 1000 sequence of observable states.

Through testing, we can classify each mobile agent to a corresponding HMM class. This is done by utilizing the maximum likelihood method for each class. The observable sequence for each agent is tested for each class and the class with the

Table 4.1: Observable and Hidden Data Connection in Terms of CVSS Base Scores.

Observation States		Hidden States			
		Vulnerable	Infected	Host	Updated
Attack Vector (AV)	Network	3.9	5.2	2.1	1.0
	Adjacent Network	2.8	2.5	3.2	0.7
	Local	2.5	4.6	4.2	0.6
	Physical	0.9	0	3.76	0
User Interface (UI)	None	3.9	0	2.55	-
	Required	2.8	0	0	0.8
Confidentiality Impact (C)	None	0	7.5	1.4	-
	Low	0	6.1	2.3	0.6
	High	0	5.3	0	1.0
Impact Integrity	None	0	5.1	4.9	-
	Low	0	6.5	5.6	0.5
	High	0	7.0	6.7	0.8
Availability Impact (A)	None	1.0	0	4.8	-
	Low	3.0	0	5.6	0.6
	High	6.0	0	6.7	0.8

maximum likelihood is calculated. The class with the largest likelihood is assigned as the mobile agents classification (Figure 4.5). This would reveal the most probable malware attack for that given mobile node.

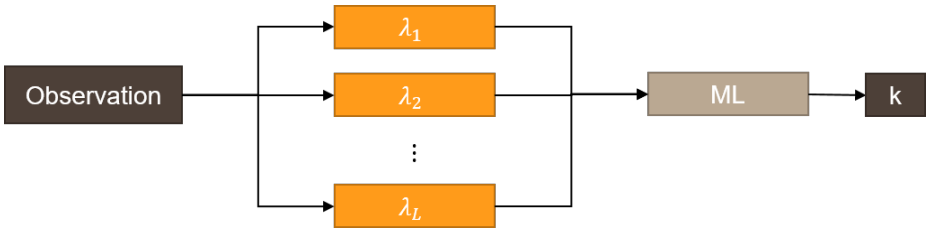

Fig. 4.5.: Categorizing Mobile Agent HMM Class.

4.4 Experimental Results

In this experiment, we estimate three HMM classes. Each class defines a mobile malware attack type. Observable and hidden states for each class are as stated above in Figure 4.3. Two data sets are defined: training and testing. The training data set is used to estimate the transmission and emission matrices for each of the three HMM classes. The testing data set is used to compute the accuracy of these estimations and prediction of hidden states.

Each HMM class is evaluated accordingly, each with its given mobile observable sequences of daily CVSS base scores. Subsequently, each training set was evaluated accordingly generating 3 corresponding HMM classes. All three HMM classes are described as follows:

- **Phishing**(λ_1): attacks devices that might have a high confidentiality impact when the attack is administered.
- **Trojan** (λ_2): A mobile agent is infected but can not be a host to the distribution due to the malware attack being specific to that agent.
- **Worm**(λ_3): A mobile agent is infected and has the ability to distribute malware to other devices.

Table 4.2: The Means for the Emission Probability Matrix for each HMM Class.

HMM Class	**mu (μ)**			
λ_1	3.93575	2.31212	5.53158	3.38396
λ_2	7.31063	2.53849	5.30766	1.13267
λ_3	4.56444	6.73595	7.60372	6.41641

Table 4.3: The Standard Deviation for the Emission Probability Matrix for each HMM Class.

HMM Class	**Sigma (σ)**	
λ_1	0.65326277	0.17731331
λ_2	0.19770494	0.86588532
λ_3	1.18664518	1.19330872

The multiple HMM class strucutre is utilized due to the changing nature of both malware attacks and MANET topology. With this system, as new attacks are introduced then, the system can easily define and add a new HMM class.

4.4.1 Training

The training data set is created for each class of HMM according to the specifications listed above. The initial transmission and emission matrices were created based on Table 4.1. The first HMM class' (λ_1) parameters were estimated as seen in matrix 4.6. TRANS and estTR correspond to the A matrix defined in Section 2.3.1. The emission equation (B matrix) is described in terms of a normal distribution ($X \sim \mathcal{N}(\mu, \sigma^2)$) seen in Tables 4.2 and 4.3.

$$TRANS = \begin{bmatrix} 0.2943 & 0.2278 & 0.4430 & 0.0348 \\ 0.3333 & 0.2207 & 0.4009 & 0.0450 \\ 0.3333 & 0.2104 & 0.4232 & 0.0331 \\ 0.2105 & 0.3421 & 0.3684 & 0.0789 \end{bmatrix} \tag{4.6}$$

After training the original sequence of CVSS base scores that range from 1 to 10 to the four hidden sates, transmission and emission probabilities are then re-estimated by applying the Baum-Welsh algorithm to estimate the most optimized transmission and emission matrices.

$$estTR_{\lambda_1} = \begin{bmatrix} 0.3247 & 0.2170 & 0.4022 & 0.0561 \\ 0.3439 & 0.2087 & 0.3877 & 0.0597 \\ 0.3509 & 0.2103 & 0.3942 & 0.0447 \\ 0.3713 & 0.2156 & 0.3531 & 0.0600 \end{bmatrix} \tag{4.7}$$

$$estTR_{\lambda_2} \begin{bmatrix} 1.00 & 0.00 & 0.00 & 0.00 \\ 0.00 & .0148 & 0.1 & .985 \\ .01 & 0.00 & .98 & 0.00 \\ 0.00 & 0.191 & 0.00239 & .8058 \end{bmatrix} \tag{4.8}$$

Matrix 4.7 corresponds to the the first HMM class (λ_1) that predicts if a mobile agent is going through a virus attack that is not updated.

4.4.2 Testing

The testing phase is to have multiple sequences of CVSS base scores for different mobile users and find the posterior probability for each class. The KLM algorithm then determines which class each mobile agent belongs to and in turn predicts the appropriate hidden state sequence. Figure 4.6 gives a generalized answer to Question 2 (Given an observation sequence ($O = O_1O_2....O_T$) and a model $\lambda = \{A, B, \pi\}$. How can a hidden state sequence ($Q = q_1q_2...q_T$) be chosen.). The hidden states were estimated for a sequence of 3 mobile agents for a span of 300 days that were classified

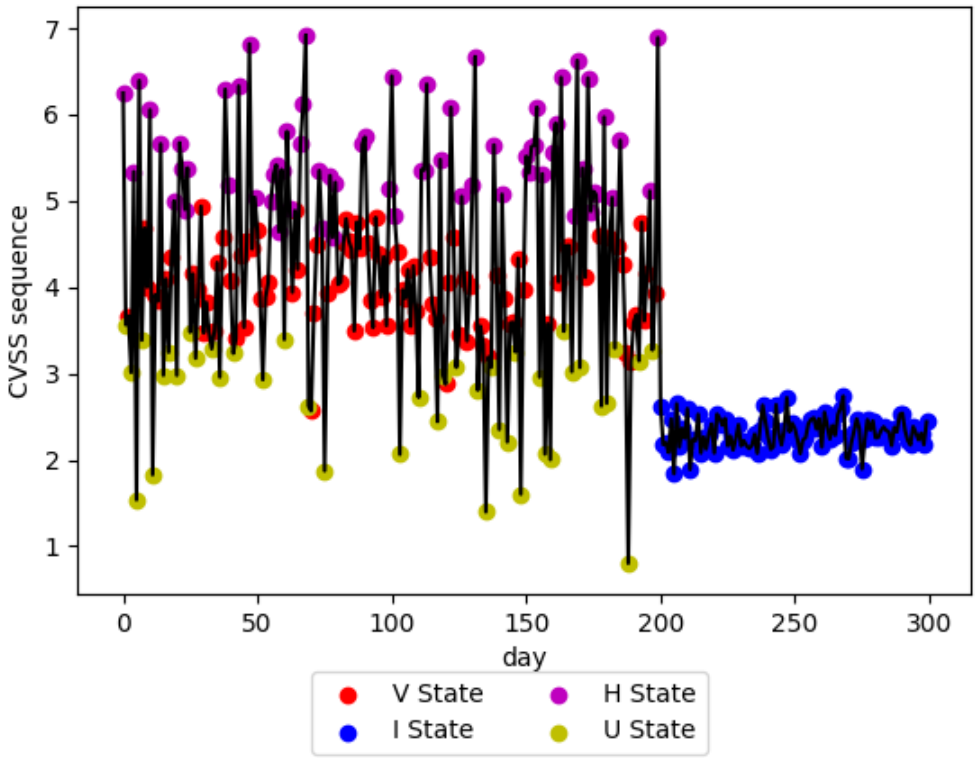

Fig. 4.6.: Prediction of estimated Hidden States for HMM Class 1.

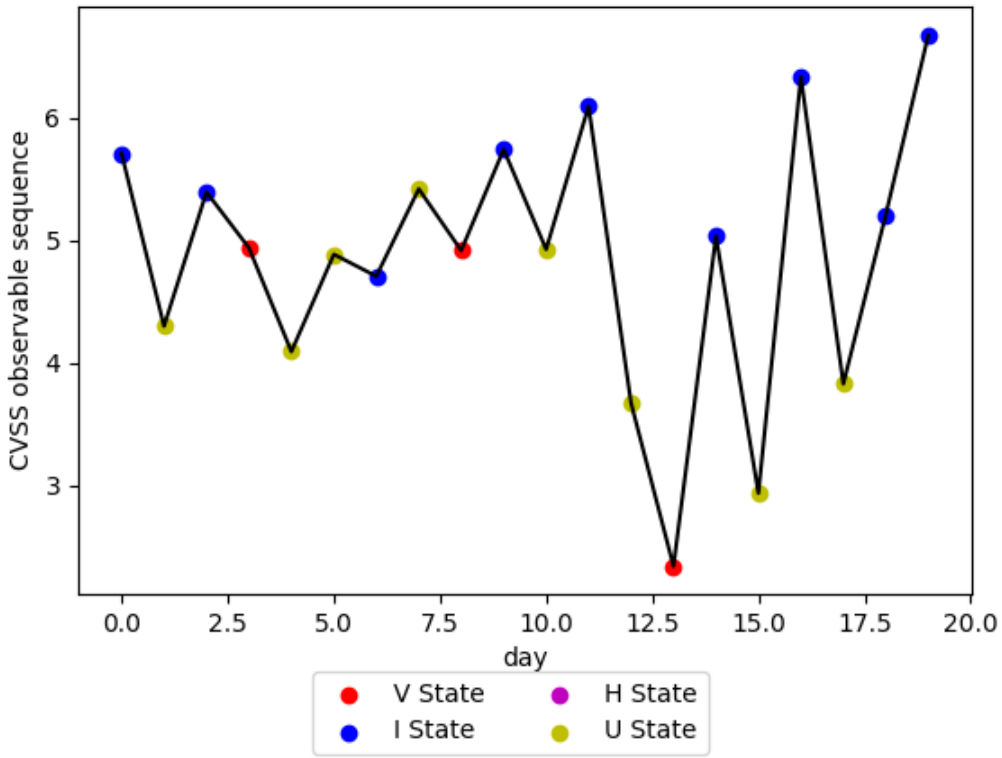

Fig. 4.7.: HMM Class 1 (λ_1) Predicted States for a 20 Day Observable Sequence.

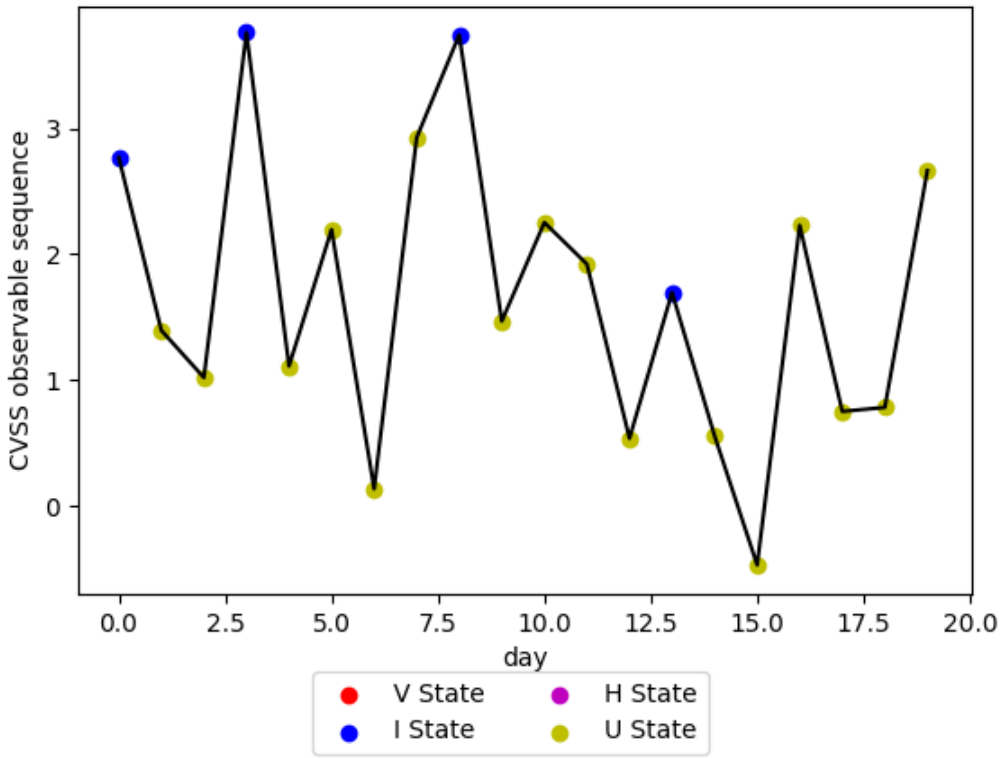

Fig. 4.8.: HMM Class 2 λ_2 Predicted States for a 20 day observable sequence.

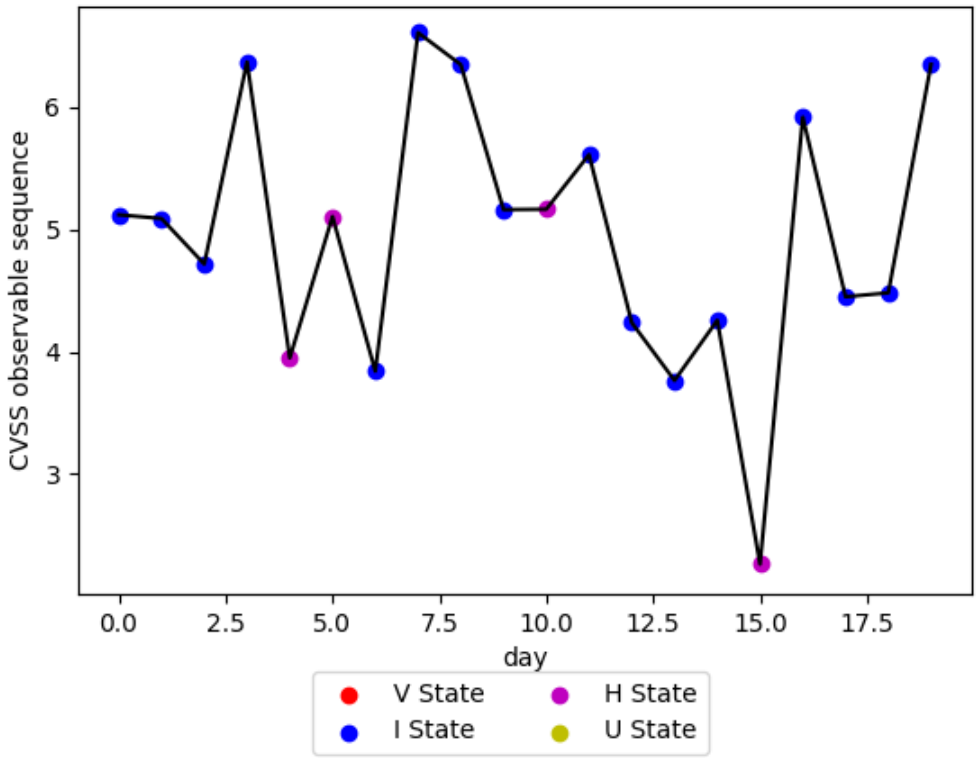

Fig. 4.9.: HMM Class 2 λ_3 Predicted States for a 20 day observable sequence.

in terms of the first HMM class. Due to the large amount of data points another test scenario was run with a 20 day observation sequence that would show each class' output clearly.

After all classes are estimated, a sequence of 20 observed CVSS scores are sent through each HMM. Figure 4.7 depicts the hidden states for the 20 observed sequence points. This mobile agent does not have the ability to become a host to spread the virus. It fluctuates between *infection* and *update* states. In comparison to class 2's sequence observed in Figure 4.8, the mobile device does not return to a vulnerable state. This displays a mobile device that does not receive updates soon enough to ever get back to a vulnerable state. This describes a fast moving virus across the MANET or that the mobile device is in a highly clustered area of the MANET. Figure 4.9 depicts a device that probabilistically is the initial device to receive the virus, this is concluded due to the fact that it starts as an infected device and fluctuates as a host to spread the virus.

Taking these different sequences and running them across the other two classes produces a low posterior probability of that hidden sequence occurring. For example, taking a mobile agent test sequence that is classified for class 1. The probability that it can be categorized to the reaming classes is 27% and below.

This shows that predicting and categorizing attacks can be observed in MANETs in a spread disease model scenario. This prediction model allows different stakeholders of mobile technology to make specific decisions on how to update the mobile network. Specifically, in terms of prioritizing software updates to which mobile agents. An HMM-FSM was developed to characterize five different mobile agent behaviors. These HMM-FSM were estimated and tested using Baum-Welch algorithm for HMMs. The data set was synthesized through a Deter network topology in connection with CVSS base score lists.

5. A NETWORK FUNCTION VIRTUALIZATION SYSTEM FOR DETECTING MALWARE IN LARGE IOT BASED NETWORKS

As stated in Section 1.3, IoT devices have become the norm in day to day usage and the forefront of data collection and subsequently a main target for an increase in malware attacks. With the Iot industry growing so does the number of communication and architecture protocols.

To combat the challenges of malware attacks on IoT, this Chapter addresses three key issues related to IoT security and proposes a machine learning Recurrent Neural Network Long Short Term Memory (RNN-LSTM) model for threat detection. The major issues we are trying to address in this Chapter are as follows:

1. We propose an efficient way to detect the four IoT attacks mentioned above through real-time data sequences by utilizing a RNN-LSTM model.

2. The RNN-LSTM integrates the two pattern intrusion techniques (misuse and anomaly) to create a large scale model for IoT devices. The proposed model provides a building block for developing an efficient distributed Intrusion Detection System (IDS) [73].

3. Subsequently, we propose a distance bound NFV architecture for malware detection. This service based architecture can regulate the computation and scalability of the IDS due to the fact that IoT devices use a multitude of different communication and device protocols. Such diversity can be managed through an NFV service architecture.

Integrating NFV to the above mentioned machine learning algorithm provides a generalized IDS that is utilized to detect not only the presence of malware but also

identify the type of the virus present in the network. Once the malware virus is detected, the virtualized system deploys updates through its distance bound service. Efficient distribution of the updates are achieved through a distance bound NFV service based structure.

In Chapter 5 we introduce the formal definition of the RNN-LSTM and links the input of a real-time traffic data to the output providing information about threat prediction. We then run simulation leading to discussion of these results of an RNN-LSTM based on an IoT real time based data set.

In this chapter, the connection between the predictive system model and the network function virtualization system for an IoT device network as part of a larger communication systems is discussed.

5.1 System Model

A detection or prediction system uses three stages to generate the final result as portrayed in Figure 5.1. Stage one deals with evaluating the data set in terms of available features. In the first stage, the data set is pre-processed, the contents of the data set help determine what type of algorithm should be utilized. Questions that need to be answered about the data set: Is the data set real or synthesized? Is the data set time based? How large is the data set? The answers to these questions are provided in the discussion through the explanation of each part of the prediction model in Figure 5.1. Once the data set is determined, the second layer should pre-process the data to ensure that the machine learning algorithm is capable of attaining the most accurate result [12]. This pre-processing stage can be developed in two ways: feature reduction or data cleaning. Feature reduction is explained in more detail in Section 5.1.1. Data cleaning occurs when parts of the data is missing. To combat this data problem from entering the ML model, the entire entry must be deleted or missing components of the entry filled in with appropriate information. The third level is the ML algorithm itself. The ML level can be one algorithm or a combination

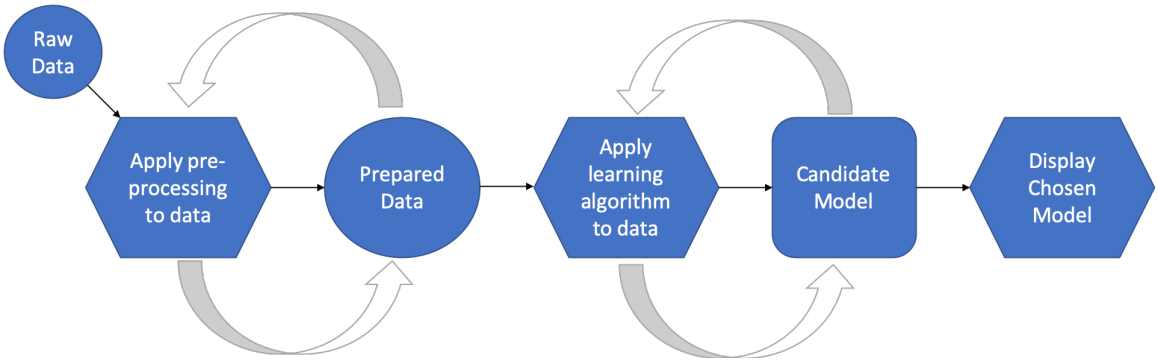

Fig. 5.1.: Proposed deep IoT threat hunting approach.

of several algorithms. Some literature includes the feature reduction step inside the ML algorithm itself.

5.1.1 Stage 2: Pre-processing Data

Before inserting the data into the ML model, there are several steps of cleaning and formatting the data [74]. Data can be missing or inconsistent, and not in the proper format depending on what language used to run the ML model. When looking at the IoT data, it is highly likely that data is inconsistent or missing due to the many protocols and applications, as discussed in Section 1.3.

After cleaning the data, the next step is to distinguish what features are essential in giving the best output to the ML algorithm. This differs depending on what type of algorithm is being applied. For example, since we are applying a time based sequence model, time features are important as part of the input data. Deciding what features are important can be done through domain based knowledge, cross correlation of data, or weighted data. Cross correlation of data is important due to the fact that it gives insight into what data is redundant and proves to be unnecessary in terms of the overall model execution. Also it takes care of removing access processing [75].

Cross correlation was the method chosen to pre-process the specific data set used in the experimentation of the RNN-LSTM model. Experimentation is executed on all

characteristics of the data set and then repeated with the highly correlated features removed. This shows the impact correlated features have on performance time and accuracy for the RNN-LSTM.

5.1.2 Stage 3: Selection of Machine Learning Model: Recurrent Neural Network Model (RNN-LSTM)

During this stage, a suitable machine learning model is chosen for intrusion detection. We use RNN-LSTM model for the reasons mentioned earlier. In this section, we discuss the general structures and characteristics of the RNN model. Subsequently, we provide details about the specific properties of RNN-LSTM in the context of predicting IoT malware. This discussion is given in Section 5.2.

The specific RNN structure that is utilized in the experimentation is discussed in Section 5.2 as a multi-layer long short-term memory (LSTM) neural network. An RNN uses a feedback loop to model a dynamic system, where the hidden state h_t is not only dependent on the current observation x_t, but also relies on the previous hidden state h_{t-1}. More specifically, we can represent h_t as:

$$\tilde{c_t} = tanh\left(w_c\left[h_{t-1}, x_t\right] + b_c\right) \tag{5.1}$$

$$c_t = f_t * c_{t-1} + i_t * \tilde{c} \tag{5.2}$$

$$h_t = f(h_{t-1}, x_t) = o_t + tanh\left(c^t\right) \tag{5.3}$$

where f is a nonlinear mapping [76]. RNN-LSTM can use the history of the sequence to capture information and utilize it for a future state. Capturing the long term history helps in minimizing the vanishing gradient problem that is encountered in an RNN structure [77].

An LSTM cell is depicted in Figure 5.2. The LSTM utilizes three different gates to capture the important characteristics needed for the next cell. The input gate takes

in the history sequence of the previous cell as well as input from the data set. There are three types of RNN; many-to-many, many-to-one, and one-to-many. These types depict the relation between the inputs and outputs of the network. Figure 5.3 shows a RNN-LSTM many-to-many type. The input gate, marked i_t in Figure 5.2 uses a *tanh* or *sigmoid* function to separate the data needed for the learning stage and data can be forgotten (ignored) for input into the next cell. The *forget* gate, marked f_t utilizes the sigmoid function to reset the memory cell to store only those important features that need to be "remembered" for the next memory cell.

As depicted in Figure 5.3, back propagation is used to create a learning environment that minimizes the prediction error. The forward propagate mechanism permeates through each layer in order to predict the weight values for the input provided as shown by the black arrows in Figure 5.3. Subsequently, the error is calculated by comparing the predicted output with the ground truth. This error is then back-propagated through the neurons (cells) and the weight of each cell (W_{hh}) is updated. Such update is represented by the red arrows in Figure 5.3.

In our specific RNN-LSTM model, three feed-forward layers are used. The reason for choosing LSTM is to emphasize the importance of temporal characteristics of the network traffic data. The flexibility of the RNN-LSTM types (many-to-one, many-to-many) is important in terms of designing an IDS for detecting a wide variety of attacks. We first present a many-to-one RNN-LSTM structure in Section 5.2. Subsequently, we extend our analysis to obtain the results for the many-to-many structure of the proposed model.

5.2 Experimental Results

In the following section, we present the data set of an IoT decentralized network and several experiments' runs to create an RNN-LSTM that produces the greatest accuracy. We utilize the KERAS python library to build our RNN-LSTM.

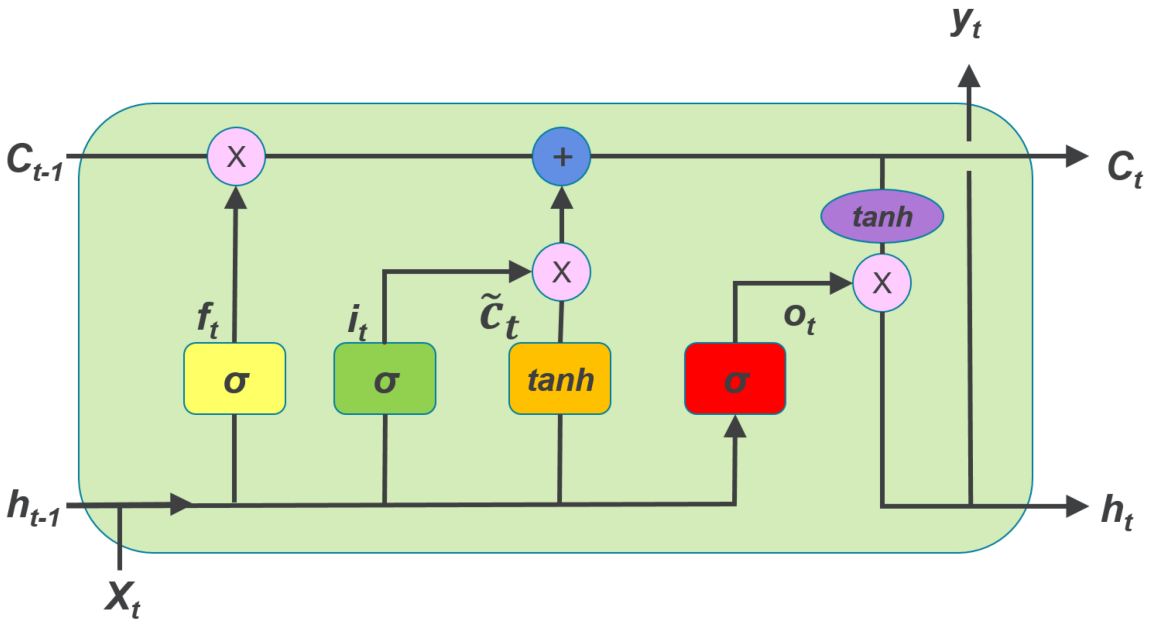

Fig. 5.2.: LSTM Cell.

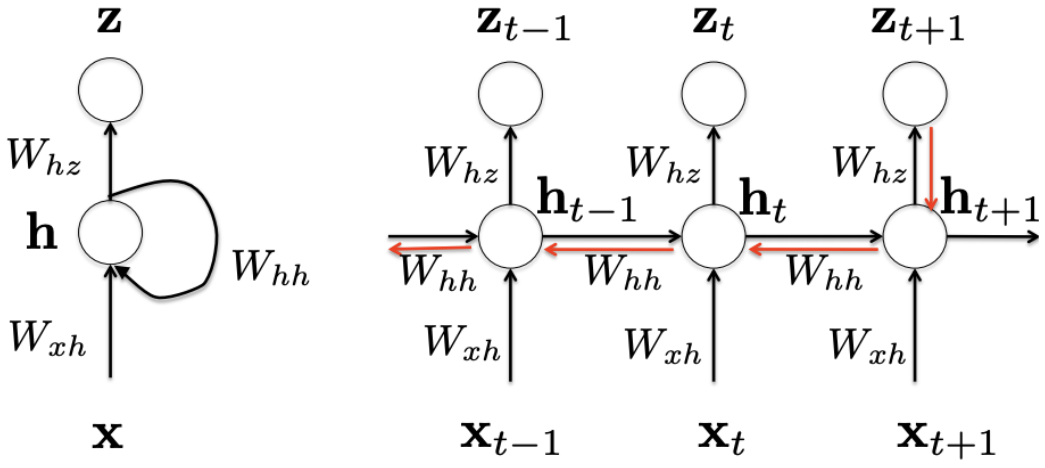

Fig. 5.3.: RNN Model.

```
Epoch 1/10
82332/82332 [==============================] - 10s 127us/sample - loss: 6.8908 - acc: 0.5506
Epoch 2/10
82332/82332 [==============================] - 10s 121us/sample - loss: 6.8908 - acc: 0.5506
Epoch 3/10
82332/82332 [==============================] - 10s 121us/sample - loss: 6.8908 - acc: 0.5506
Epoch 4/10
82332/82332 [==============================] - 10s 123us/sample - loss: 6.8908 - acc: 0.5506
Epoch 5/10
82332/82332 [==============================] - 10s 124us/sample - loss: 6.8908 - acc: 0.5506
Epoch 6/10
82332/82332 [==============================] - 9s 114us/sample - loss: 6.8908 - acc: 0.5506
Epoch 7/10
82332/82332 [==============================] - 10s 124us/sample - loss: 6.8908 - acc: 0.5506
Epoch 8/10
82332/82332 [==============================] - 10s 121us/sample - loss: 6.8908 - acc: 0.5506
Epoch 9/10
82332/82332 [==============================] - 10s 119us/sample - loss: 6.8908 - acc: 0.5506
Epoch 10/10
82332/82332 [==============================] - 11s 134us/sample - loss: 6.8908 - acc: 0.5506
```

Fig. 5.4.: Trained RNN-LSTM with Input on All Features.

5.2.1 BoT-IoT Data Set

Most of the approaches mentioned in the literature discussed in Section 2.5 employ either Android based applications or LAN based networks data sets to test their machine learning algorithms. All these data sets have specific network characteristics as mentioned in Section 2.5. These characteristics are used as assumptions when developing the corresponding machine learning algorithm.

Our objective is to develop a ML algorithm that meets the needs of most if not all the IoT devices currently available for several applications. These include LAN, MANET, and home based applications. For the evaluation of the proposed algorithm, we utilize the BoT-IoT data set [78]. This data set is composed of over two million records. In addition, it is comprised of nine types of attacks; namely, Fuzzers, Analysis, Backdoors, DoS, Exploits, Generic, Reconnaissance, Shellcode and Worms. The Argus and Bro-IDS tools are used and twelve algorithms are developed to generate a total of 49 features with the class label [79]. Some features are source and destination IP addresses, round trip packet time, and the service that a packet is sent through. The primary reason for choosing this data set is due to its detailed characteristics. These characteristics, as mentioned above, can be found in all the network

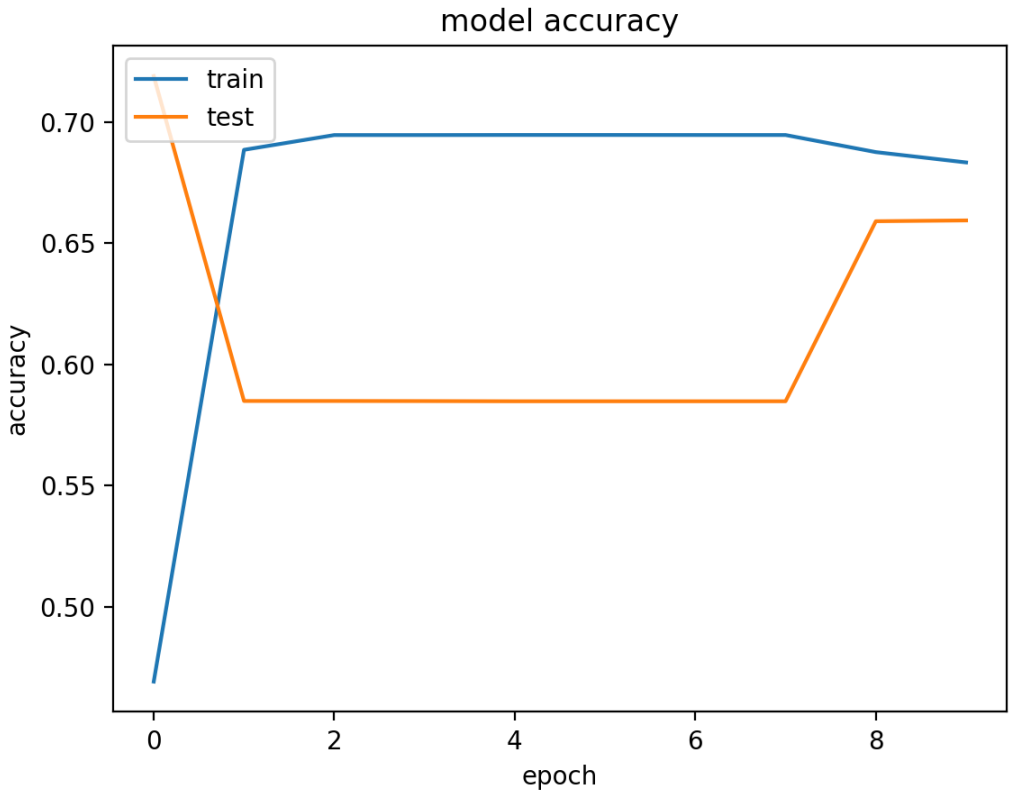

Fig. 5.5.: Training/Testing on 10 epochs

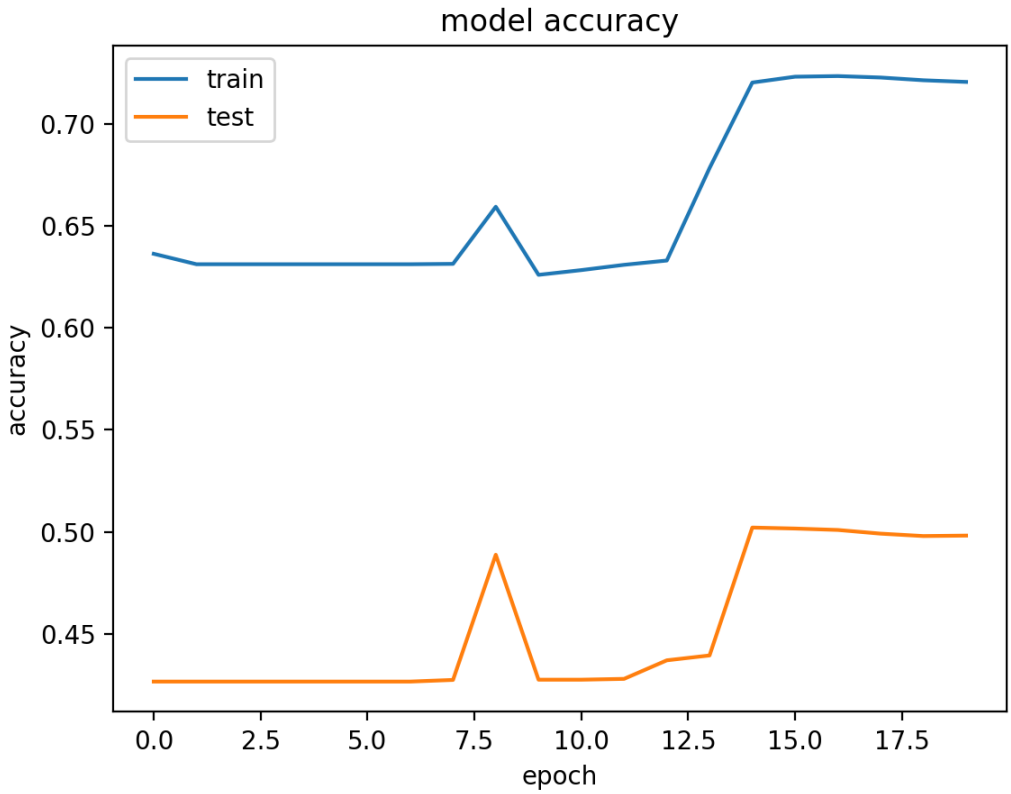

Fig. 5.6.: Training/Testing on 20 epochs.

	dur	spkts	dpkts	sbytes	dbytes	rate	sttl
dur	1.000000	0.280239	0.217507	0.225432	0.172492	-0.118031	-0.000986
spkts	0.280239	1.000000	0.369554	0.965750	0.198324	-0.068249	-0.092536
dpkts	0.217507	0.369554	1.000000	0.175834	0.976419	-0.083173	-0.163830
sbytes	0.225432	0.965750	0.175834	1.000000	0.010036	-0.025102	-0.017866
dbytes	0.172492	0.198324	0.976419	0.010036	1.000000	-0.047978	-0.114537
rate	-0.118031	-0.068249	-0.083173	-0.025102	-0.047978	1.000000	0.388155
sttl	-0.000986	-0.092536	-0.163830	-0.017866	-0.114537	0.388155	1.000000
dttl	0.090048	0.054601	0.036483	0.049891	0.012537	-0.453913	-0.033338
sload	-0.076343	-0.044194	-0.054145	-0.015228	-0.031266	0.550104	0.252901
dload	-0.047032	0.074440	0.133835	-0.006428	0.100923	-0.138441	-0.386224
sloss	0.240113	0.973644	0.189060	0.995027	0.014561	-0.040139	-0.038088
dloss	0.171182	0.198683	0.981506	0.007091	0.997109	-0.062073	-0.137737
sinpkt	0.079840	-0.014501	-0.017141	-0.005399	-0.010201	-0.065681	-0.179270
dinpkt	0.150801	-0.003309	-0.007181	-0.001432	-0.007266	-0.052206	-0.006154
sjit	0.146599	-0.002407	-0.003862	-0.002675	-0.005182	-0.061961	0.030062
djit	0.165418	0.010481	0.034276	-0.003050	0.029201	-0.081591	-0.084072
swin	0.083990	0.111828	0.141478	0.041450	0.083015	-0.534075	-0.370458
stcpb	0.044482	0.083647	0.110259	0.028712	0.064456	-0.428528	-0.305700

Fig. 5.7.: Correlation Matrix.

devices. Additionally, the data set provides a more accurate labeling mechanism for data packet as a malware attack, thus making it easier to identify the source of the data packet which is important in terms of utilizing the information for the distance based NFV patching model for the IDS, discussed in Section 5.3.

The training set for the proposed RNN-LSTM is composed of 175,341 records and the testing set contains 82,332 records comprised of the different types of attacks and normal data packets.

5.2.2 Experiments and Performance Results

The experiment is focused on the many-to-one RNN-LSTM model. In other words, we focus on predicting whether a data packet is malicious or normal. The experiment consists of all the features of the data set, 44 features in total. The 44^{th} feature corresponds to labeling the specific type of malware attacks. The 45^{th} feature is used as the label which corresponds to 1 if the packet is malicious and 0 for a normal packet. Figure 5.4 displays the accuracy and loss for each epoch when the training data set is used by the proposed the RNN-LSTM with the parameters listed in Table 5.1. The results of this experiment are depicted in Figure 5.4. The results show that

after 10 epochs, the model predicts with an accuracy of 55%. This low accuracy is due to the presence of highly correlated features and the use of epoch and batch size that did not fit with the input to the learning model. Additionally, the number of neurons used at each level of the neural network was not optimal for the input of this structure of RNN-LSTM. We modify the experiment of the RNN-LSTM design parameters and study the effect of change in epoch size, batch size, and neuron layer size. Figure 5.6 shows a higher prediction when the batch size is increased to a value of 20 and the input layer neurons are decreased to 8. For these new model parameters, accuracy for the trained and tested data sets for both 10 and 20 epochs is observed. However, again due to the high correlation, the prediction accuracy only reached to 75% for the trained data set and to a low value of 52% for the testing data set.

This low accuracy is potentially due to high correlation as we investigate with further experiments. Figure 5.7 presents the correlation matrix of the original feature set. The correlation matrix provides an insight into the features that are highly correlated to one another. This observation is important due to the fact that highly correlated data skews the RNN-LSTM learning mechanism to pure labeling of the data points to one class. In order to remove the skewing effect of the correlated data, we identify the highly correlated features. These features are excluded from the input to the training of the RNN-LSTM. In Figure 5.7, the small subset of correlated numbers shown in red rectangles represents high correlation. In essence, the third experiment connects the RNN-LSTM to an uncorrelated set of data generated by excluding the highly correlated data. Consequently, the total number of features were decreased to 29 for input to the proposed RNN-LSTM. Furthermore, the RNN-LSTM layers were modified to create a 3-layer model that balanced the number of neurons in each layer. The prediction accuracy increased to 93% for training and for testing; the accuracy jumped to 87% as seen in Figures 5.8 and 5.9. The testing data set consists of around 90,000 data points. Table 5.2 is the confusion matrix that presents the false positives and negatives of the model. Consistent to testing accuracy of the model, 87% of the labels were predicted correctly. Decreasing the number of features not only increases

Table 5.1: RNN Parameters.

Batch Size	64
Epoch	30
Optimizer	rmsprop

Table 5.2: Confusion Matrix for the IoT test Dataset.

Actual/Predicted	Threat	Non-Threat
Threat	43,831	18,786
Non-Threat	8,417	19,637

the accuracy but also creates a shorter training sequence per epoch. The computation time of the RNN-LSTM learning decreased, as each epoch was completed in 3 seconds versus the 10 seconds taken in the first experiment. We started experimenting at 20 epochs as seen in Figure 5.8. Figure 5.9 shows the accuracy at 120 epochs where the highest accuracy for testing at around epoch 40. The time aspect is an important performance criteria for IDS since the proposed model can be deployed by the IDS in a real world network where the damage containment race in terms of patching against malware spread is crucial. The faster the training of the RNN-LSTM, the faster the deployment of updates as well as the containment of the malware spread.

The BoT-IoT Dataset includes 10 different categories of network attacks as listed in Section 5.2.1. We train and test our RNN-LSTM on identifying the specific viruses. This new experiment designed and tested a many-to-many RNN model, gave prediction of 10 different malware categories. Figure 5.10 and 5.11 again show that accuracy of the RNN-LSTM in terms of testing is on the average of 85%. As we increase the epochs of the learning set, the testing prediction tends to decrease. This level of categorization can tremendously enhance the benefits provided by IDS when it is integrated with a response mechanism such as communicating patches to the infected

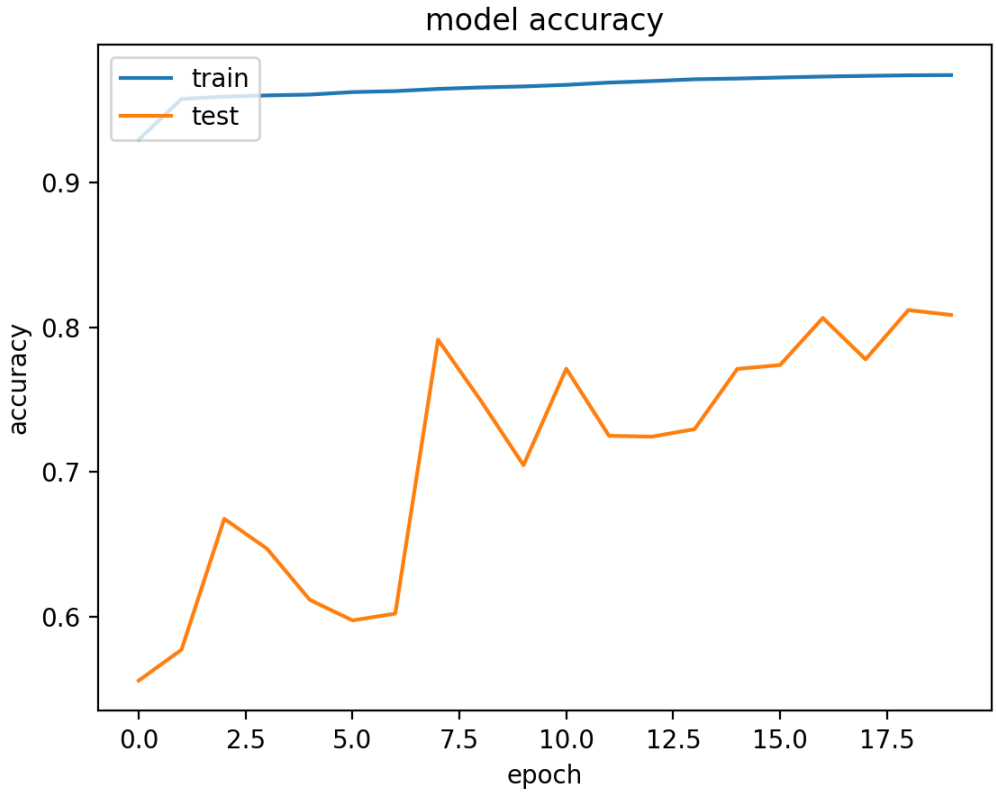

Fig. 5.8.: Accuracy of RNN-LSTM Model on the uncorrelated feature set on 20 epochs.

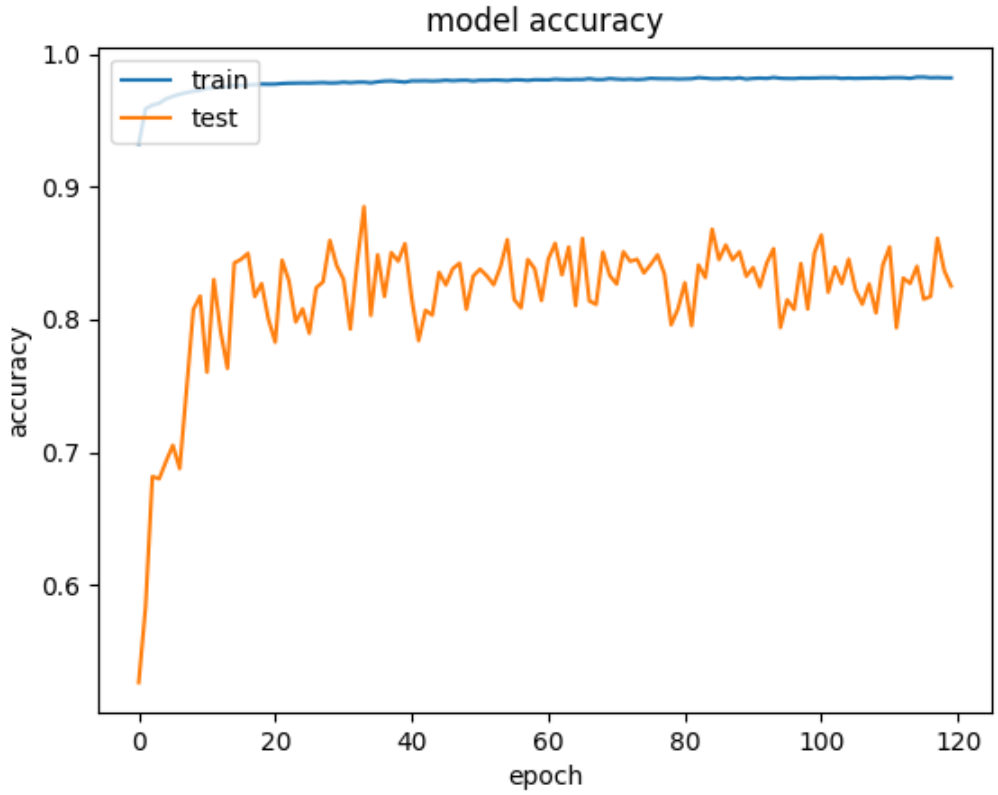

Fig. 5.9.: Accuracy of RNN-LSTM Model on the uncorrelated feature set on 120 epochs.

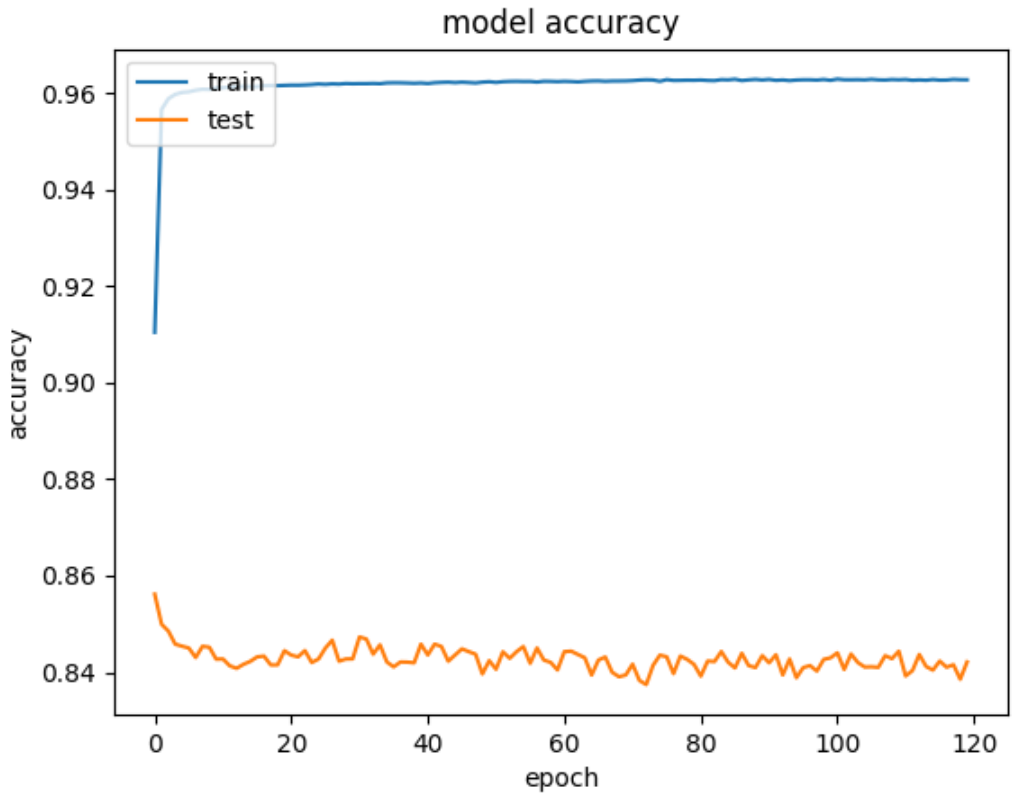

Fig. 5.10.: Accuracy of RNN-LSTM Model of the 10 specified Attacks on 120 epochs.

IoT devices. In the next section, we propose an NFV patching model. Once the IDS identifies a specific type of malware attack, the patching mechanism can deploy patches specific to that attack. This focused patching mechanism can be highly efficient as it can incur less network resources and can decrease the patching time.

5.3 NFV Patching Model

Based on the semi-distributed IDS architecture proposed in [73], we present a *distance bound patching system* that pinpoints the devices requiring patching. This not only decreases the computational time for each patch but can also greatly decrease the usage of network resources. Figure 5.12 illustrates how the patch is distributed

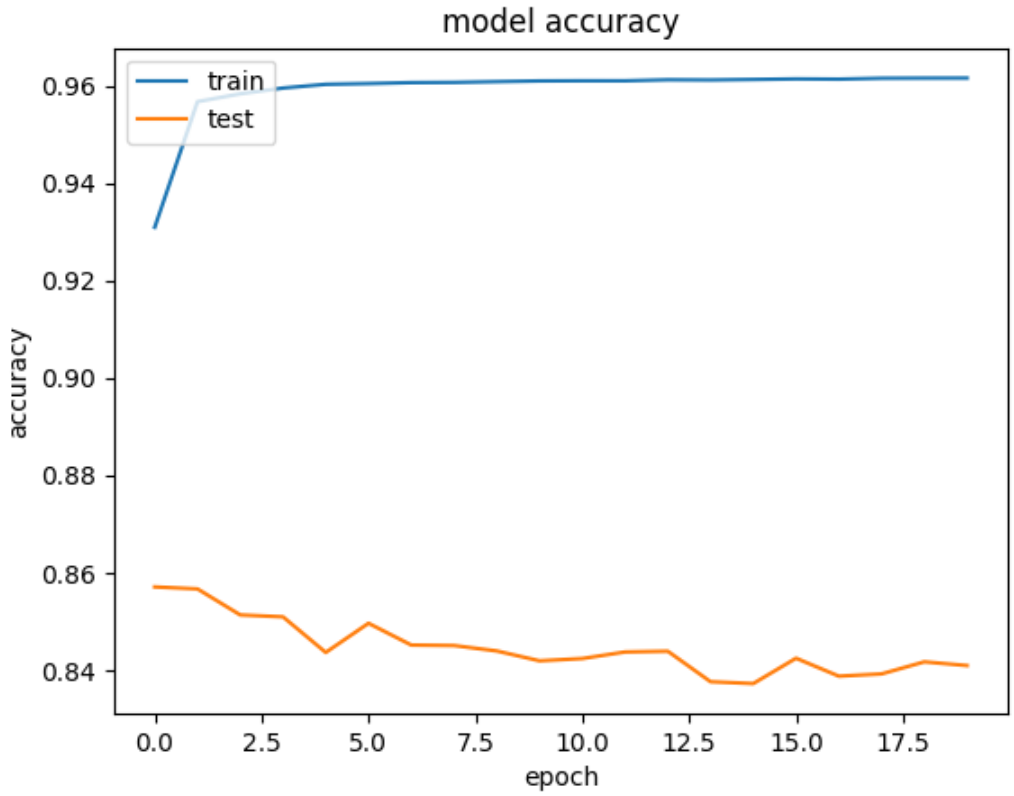

Fig. 5.11.: Accuracy of RNN-LSTM Model of the 10 specified Attacks on 20 epochs.

based on a distance bounded area. For this purpose, we use surveillance zone's control monitoring mechanism. These surveillance zones are designated as a patch delivery site. The patching system utilizes a push delivery concept. In practice, a push delivery patch is accomplished by the central network sending out the patch which is usually not scalable [72]. The distance bounded patch system solves the scalability issue and can also help with allocating resources in the right areas.

The surveillance zones are connected to a central network controller that houses the RNN-LSTM learning model. As packets are collected within a surveillance zone, the RNN-LSTM continues to learn through every batch. As the proposed RNN-LSTM keeps learning at each epoch, we can time the epochs to pull packets from the network. In case there is a new type of attack, it is added as part of the classification set of the RNN-LSTM. If an attack is triggered from a specific packet in the network, a patch can be delivered to a surveillance zone that controls all devices within a bounded distance of where the initial attack happened. The process is represented in Figure 5.12 by the purple and orange ovals on the network. This specific structure was made to solve the scalability and efficiency issues we have with decentralized networks. In this discussion, the network is purposefully vague, due to the fact that this complete IDS can be implemented in an IoT network (Figure 1.1, MANET or other device based networks that do not include a specified centralized system.) Such virtualization is manifested by a cloud information based environment.

In essence, NFV virtualizes surveillance zones in order to perform various network security-oriented functions seen as monitoring of traffic among devices for the malware/attack detection, and deploying a patch distribution-based response mechanism within a surveillance zone. Due to the distance bound virtualization structure of surveillance zones manifested by the NFV, once devices enter a specific surveillance zone, the device is registered and mapped to a specific identification group maintained by the NFV for that surveillance zone. Virtualizing the group-bound patching mechanism creates a seamless flow of traffic across the network. The virtualization of the patching mechanism eliminates the need to search for an update or change protocols

across IoT devices. Note that in Figure 1.1 each application in the IoT infrastructure can have several protocols that it utilizes. The virtualization at the surveillance zone level allows for heterogeneity across the network. When the surveillance zone receives a patch packet from the central network, it does the work of making it compatible with all protocols of the IoT devices in that zone. In a worst case malware attack, the network would require several resources to determine the application and protocols being used by these devices. Virtualization at the surveillance zone level and implementation of NFV concept at the gateway level can reduce the complexity of resource management. The gateways of the IoT accept a patch update from the surveillance zone. This helps with relieving some of the network resources to efficiently complete other network tasks.

For example, protocols in an Android might differ from an Apple phone. If the NFV is not available to different patches, it would need to be administered to cover all protocols in the network. With the NFV structure, one patch is sent and the surveillance zone creates compatible versions for each device that is mapped to it. Additionally, an IoT device would not have to accept certain traffic packets due to the fact that it needs to apply search and update its functionalities. This increases the security component of the network.

This architecture can be evaluated based on several performance criteria such as: Number of alerts versus the number of patches in the network, time of attack detection, time of patch deployment, malware containment time, malware die-out time, patch packet loss, number of surveillance zones, size of the surveillance zone versus the density of devices managed. These metrics can present a direct assessment of the RNN-LSTM and the NFV patch dependent structure.

The NFV distance bound patch model is developed to boost whether it be in terms of increasing or decreasing the aforementioned performance criteria. The NFV distance bound patch model and the RNN-LSTM only work hand in hand if there is a balance in terms of detection and patching. This proposed NFV architecture at the surveillance zone level can be evaluated with respect to the race condition which

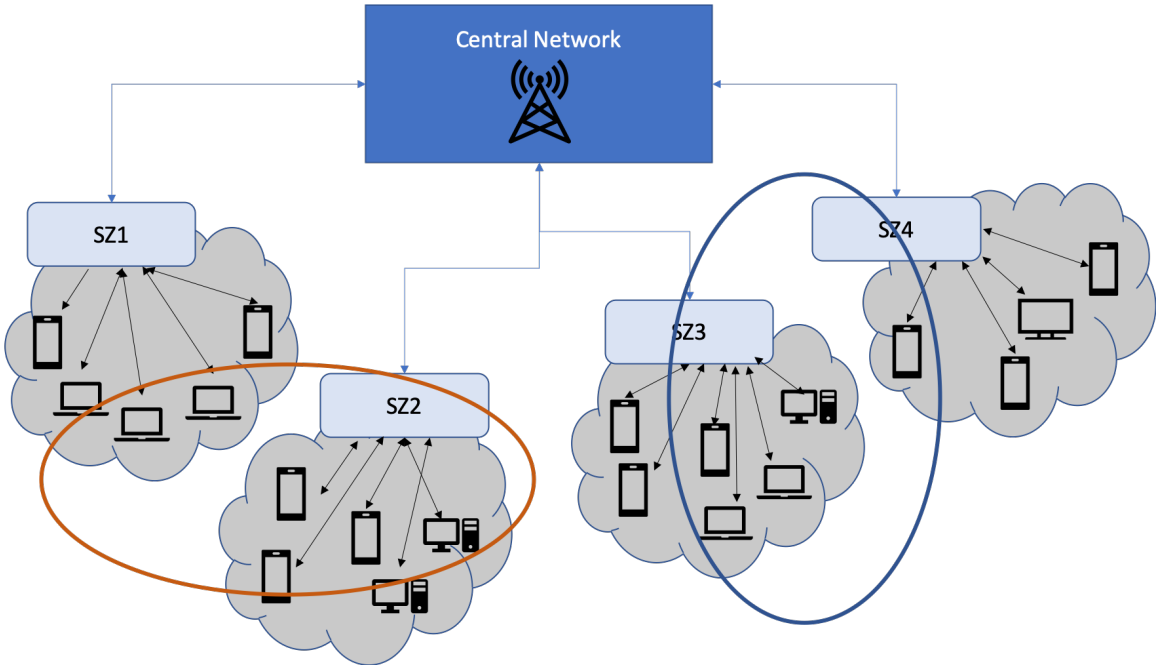

Fig. 5.12.: NFV Distance Bound Patch System Architecture.

entails comparing the time it takes to predict an attack to the number of devices already infected. Subsequently, we need to compare the time it takes to clear the infected devices which indicates the die-out time of the malware attack.

5.3.1 Malware Spread Model Definitions

The proposed architecture connecting the RNN model and the NFV distance bound patch model structure is analyzed by inserting a malware attack to the IoT network. To present the proper analysis, we present results through the susceptible (S), exposed (E), infected (I), and resistant (R) (SEIR) disease (malware) spread model experimentation. The malware spread model presents the percentage of nodes at the four given malware states. In our case, the four states are described below [80]:

1. *Vulnerable (S)*: a node that is vulnerable to attacks.

2. *Infected (I)*: the node is already infected with the malware attack.

3. *Host (E)*: a node that has the ability to spread the malware attack.

4. *Updated (R)*: a node that has been given the "patch" so as not to accept any malware attacks that have been detected in the network.

Each parameter listed in the previous section is analysed through the SEIR definition of spread in terms of malware attacks. The SEIR spread analysis employs three different parameters:

1. Infection probability parameter (β): the probability that a node in the network gets infected due to the malware attack.

2. Incubation period parameter (α): the time it takes for a node to stay in the infected or exposed state.

3. Removal probability parameter (γ): the probability that a node becomes immune to the attack.

With the parameters listed above and the number of nodes in the network, the SEIR model produces the number of nodes in each malware attack state at any given interval of time through the computation of differential equations [73]. The parameters are modified as the RNN-LSTM runs to predict and identify different malware attacks on the IoT network. With the parameters computed, the structure and initial infection of the network also have a bearing on the output of the SEIR model.

5.4 NFV Performance Evaluation

In this section, we present the SEIR model evaluation on several performance metrics: scalability of the network, increased number of malware attacks, surveillance zone node density, and interconnectivity between surveillance zone nodes. Increasing the number of nodes in the network reveals the ability of scalability on the overall network and its capability to contain the spread of a malware attack. The density

Table 5.3: SEIR parameters For NFV Distance Bound Network Architecture.

Figure	$\beta(\%)$	$\gamma(\%)$	i(0) (%)	Number of Nodes	Number of Surveillance Zones
5.21	1.0	0.5	1	1000	5
5.20	5.0	1.0	1	1000	5
5.14	1.0	1.0	5	1000	5
5.15	1.0	1.0	1	10000	50
5.16	1.0	1.0	1	100000	50

thresholds for each surveillance zone is calculated and analyzed based on the number of nodes in the overall network.

The NFV distance bound patch network undergoes several SEIR model tests to analyze these metrics. Tests include the manipulation of SEIR parameters dependent on the initial infection, and the analyzed malware attack. Different SIER parameters convey the ability of the RNN-LSTM to indicate the right malware attack and therefore deploy the appropriate patch for the malware attack. Table 5.3 lists all the model parameters of the IoT network and the SEIR model parameters that have been calculated utilizing the speed of worm and DoS attacks [72]. Additionally, the number of surveillance zones has been listed with a uniform distribution of the nodes for each network listed.

To analyze the malware spread accurately, the NFV distance bound patch model is represented as a combination of two graph models: a balanced tree graph and Erdos-Renyi graph. The balanced tree metrics manipulate the density threshold for each surveillance zone. Whereas the Erdos-Renyi is in charge of the number of nodes in the graph and the interconnectivity of nodes between surveillance zones. A sample graph structure is shown in Figure 5.13.

In the following sections, each metric is explained and accompanied by its corresponding SEIR analysis graphs.

5.4.1 Scalability

The scalability problem is tested through the analysis of three different sized networks. Figures 5.14, 5.15, and 5.16 are for networks of node sizes 1000, 10,000, and 100,000, respectively. The scalability of the NFV distance based architecture is confirmed when the peak number of infections stays constant between 50% and 65% even when the number of nodes is increased to 100,000. This is due to the fact that the surveillance zones are 2% of the total nodes in the network for both 1000 and 100,000. The peak infection rate for 10,000 (Figure 5.15 is higher than

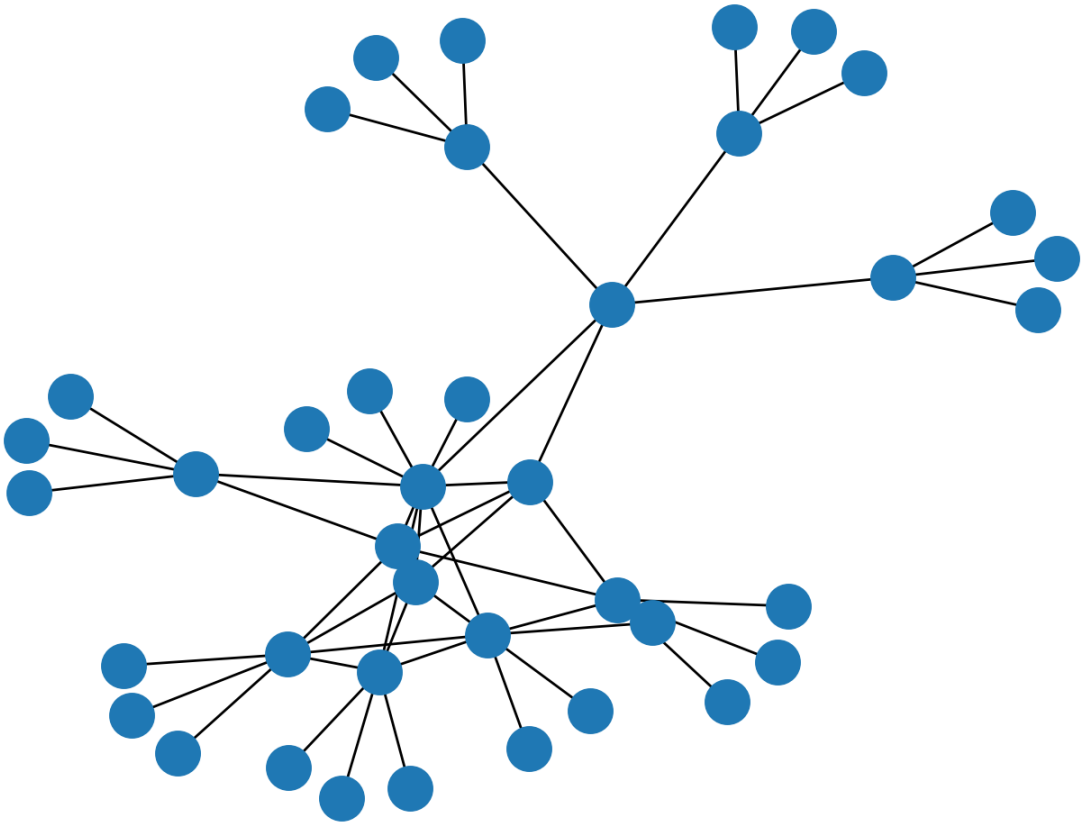

Fig. 5.13.: Sample NFV ditance bound patch graph.

both peaks in populations of Figures 5.14 and 5.16 but corresponds to an output of an earlier die-out rate. This matches the hypothesis that the higher the peak of infection the faster the infection die-out time. Additionally, despite the high rate of infection the number of nodes for all three populations and that of hosts are low. This is due to the distribution of nodes among surveillance zones designed by the distance bound architecture. The number of hosts of infection, does not increase past 25% of the node population for all three population sizes. Another topic that can be addressed is: does the density of surveillance zones have any bearing on the scalability in maintaining the networks security from malware attacks? Management and redistribution of surveillance zones due to high network traffic can be seen as an overhead that might hinder the distribution of patching.

We also experimented on the scalability of the network if the infection rate is to increase due to different types of malware that can be introduced to the system. Answering the question how would the NFV model respond if there was a sudden malware attack? As shown in Figures 5.17, 5.18, and 5.19, the SEIR output with infection rates at 8, 10, and 50 %, respectively, we can conclude that even with the infection rate (*beta*) being at 50%, the peak infected nodes stay below 60%. This shows that the NFV distance bound patch model is able to contain the malware spread to within a specific surveillance zone. The results signify that the design of a distributed NFV system quarantines the malware attack to a specific area, this also creates a more efficient network in terms of when and where the patch is administered.

5.4.2 Multiple Malware

The RNN component of the overall architecture is to determine if the network is capable of patching multiple malware attacks at any given time. In the following, we run and discuss tests to see whether this is achievable or not.

Figures 5.20 and 5.21 present the SEIR output for a 1000 nodes network with a different infection probability due to the RNN output of two different malware attacks

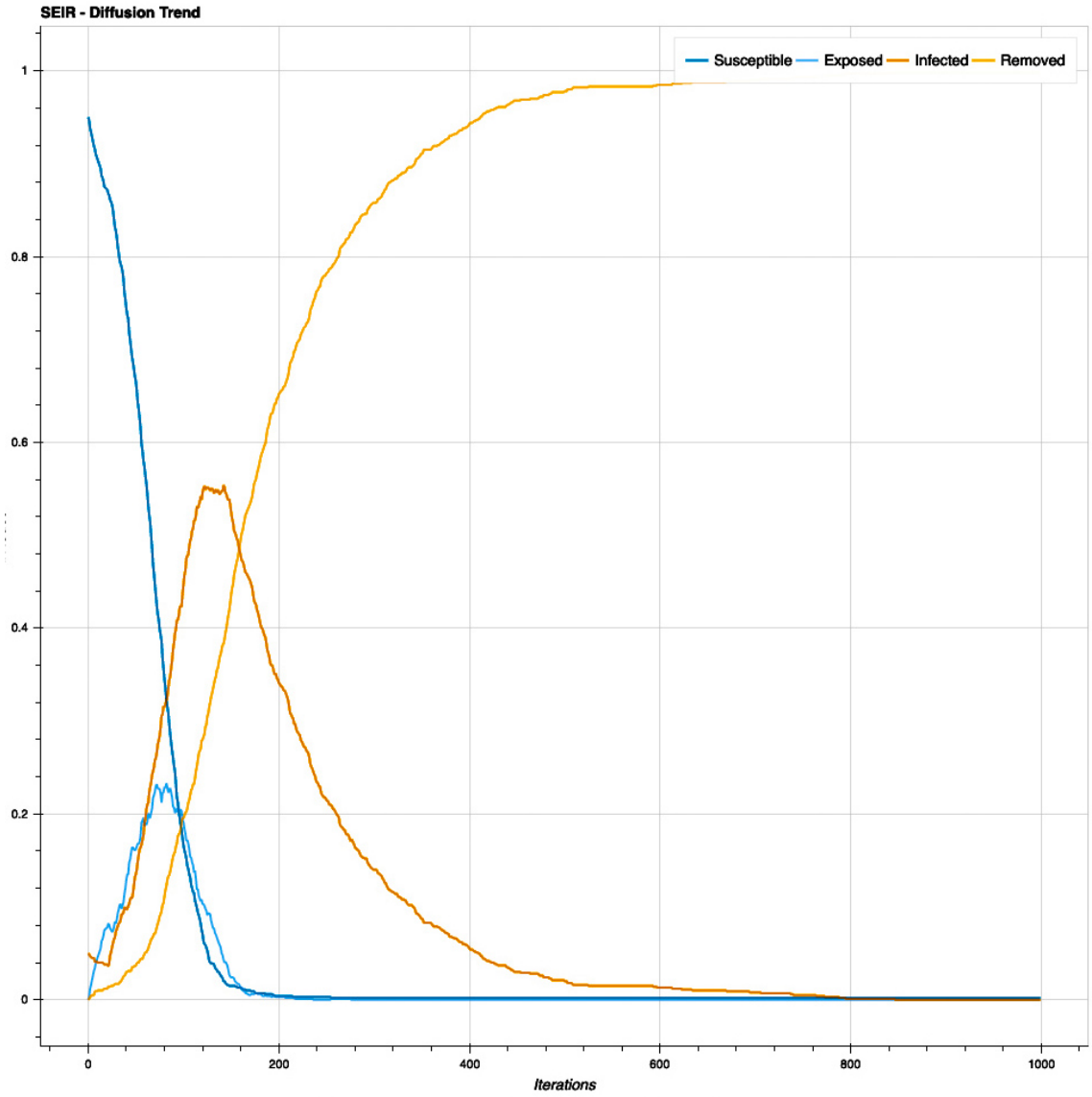

Fig. 5.14.: SEIR Graph with N = 1000.

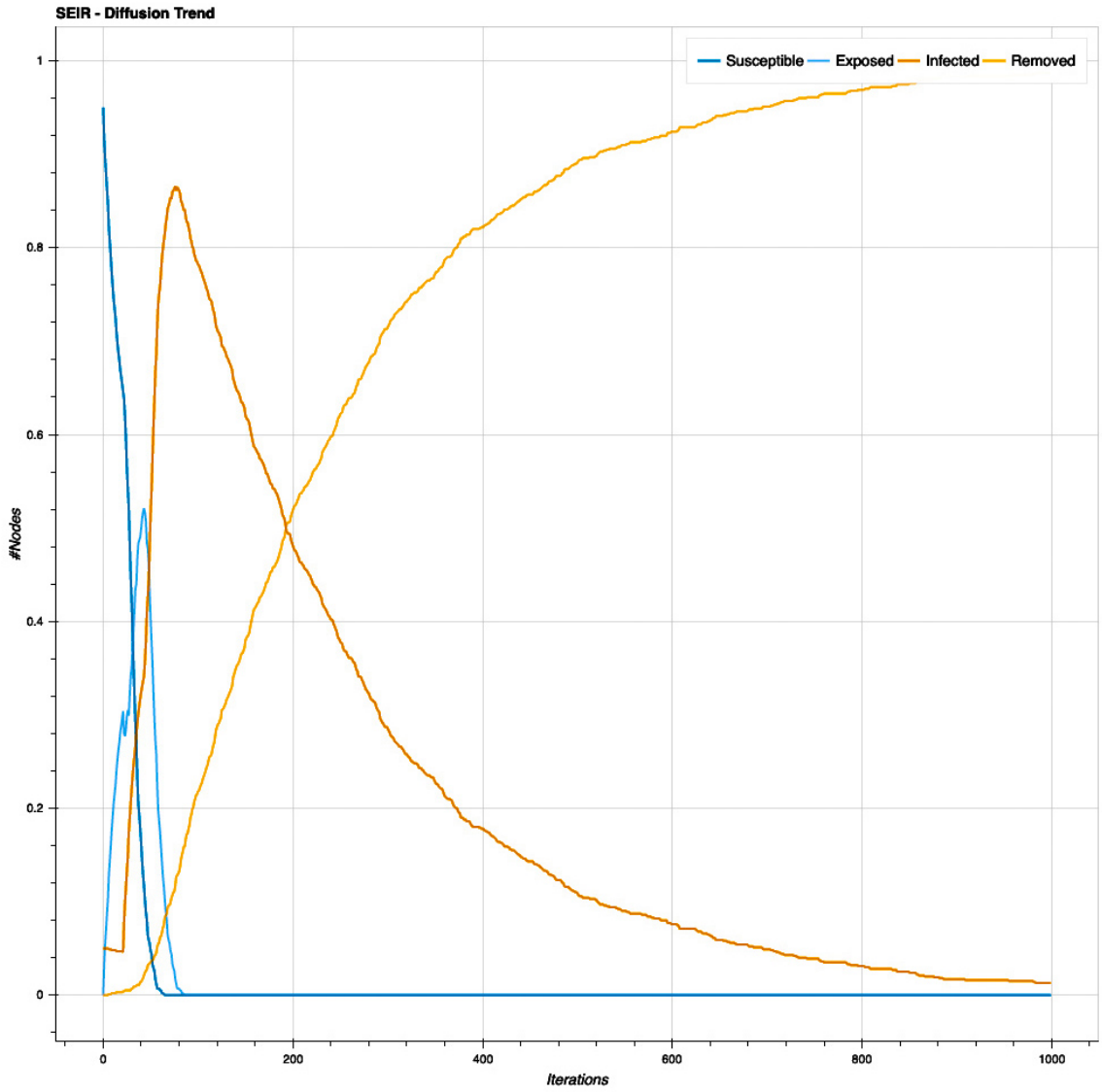

Fig. 5.15.: SEIR Graph with N = 10,000.

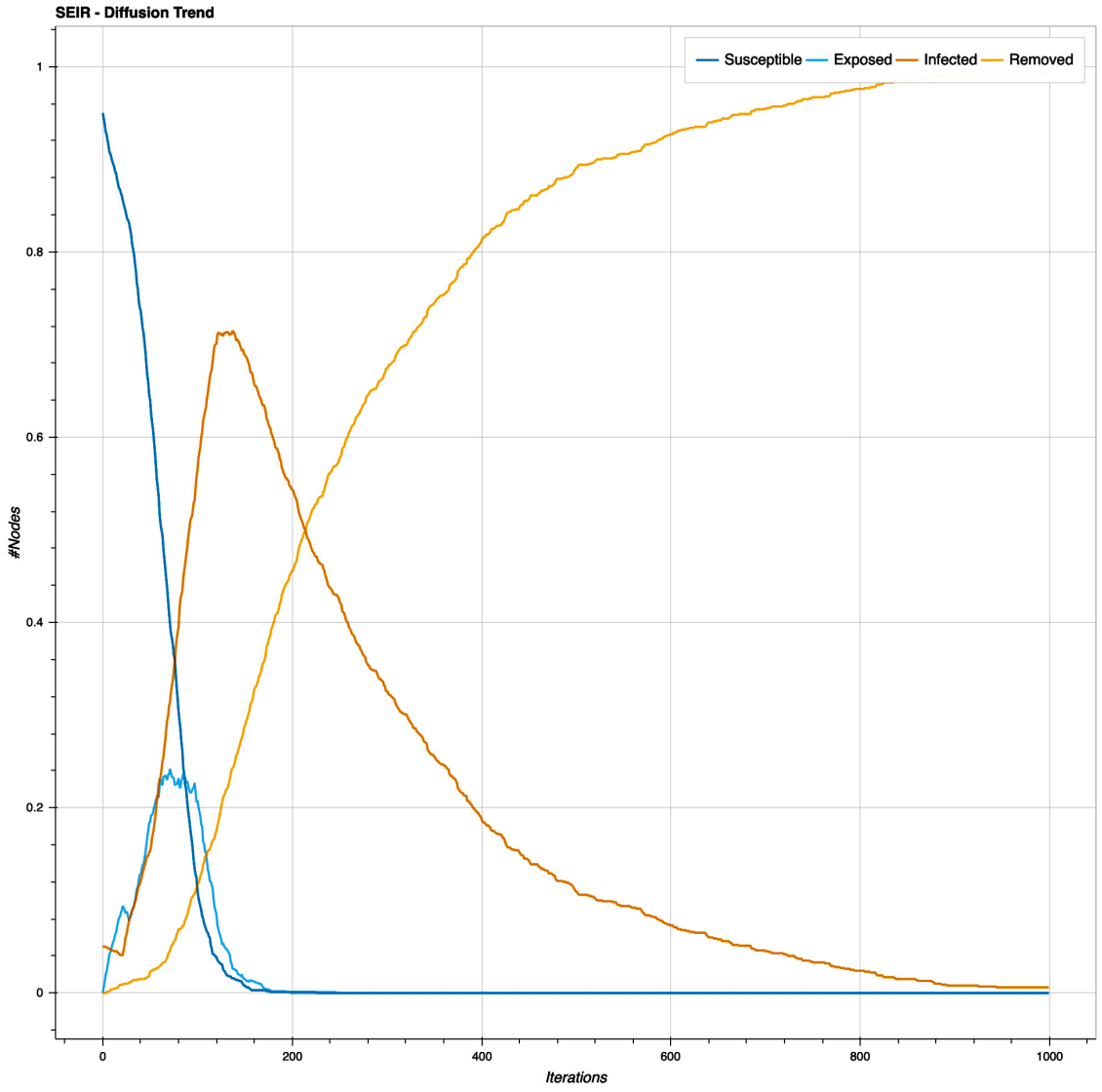

Fig. 5.16.: SEIR Graph with N = 100,000.

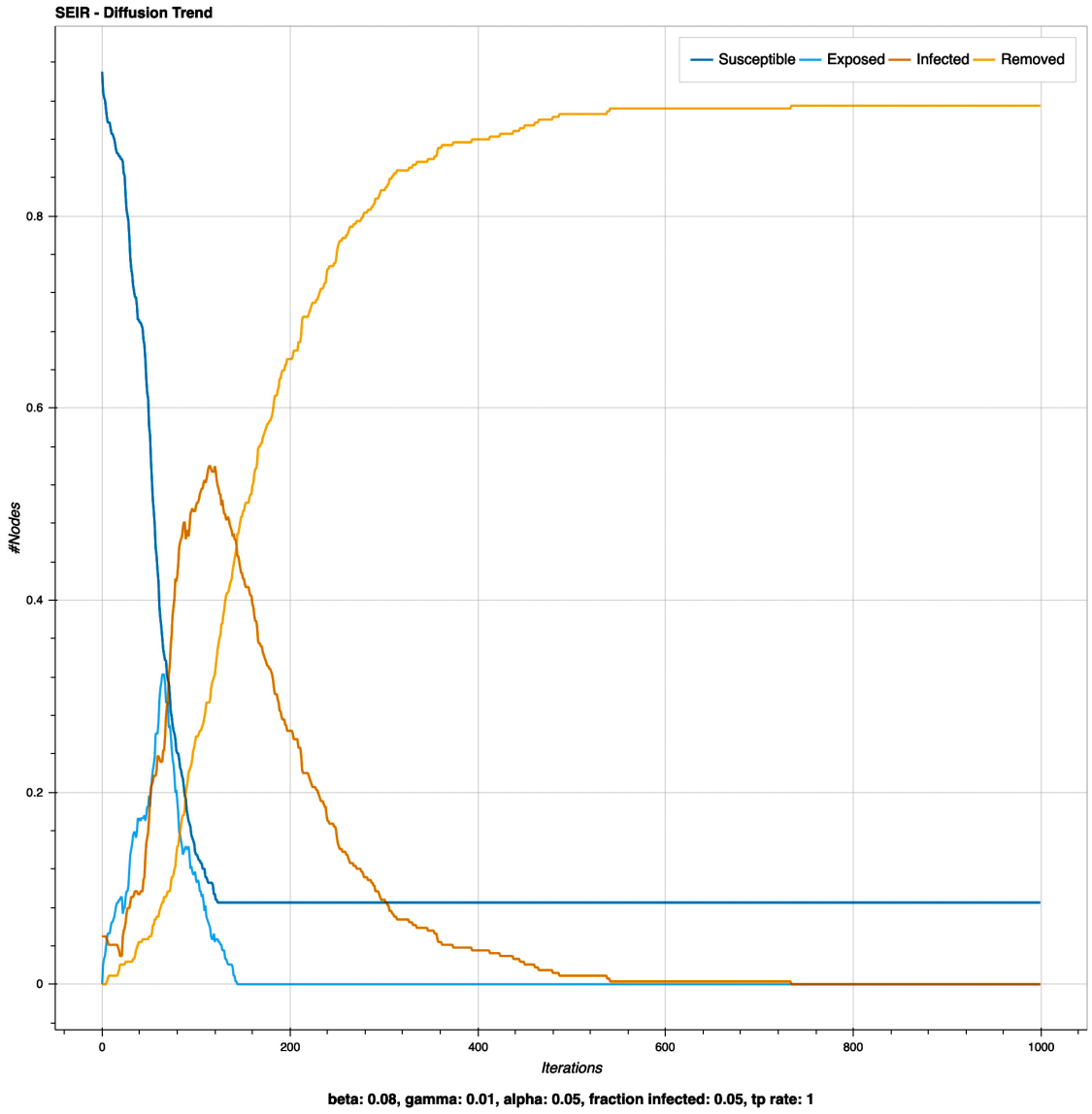

Fig. 5.17.: NFV SEIR output with infection rate at 8%.

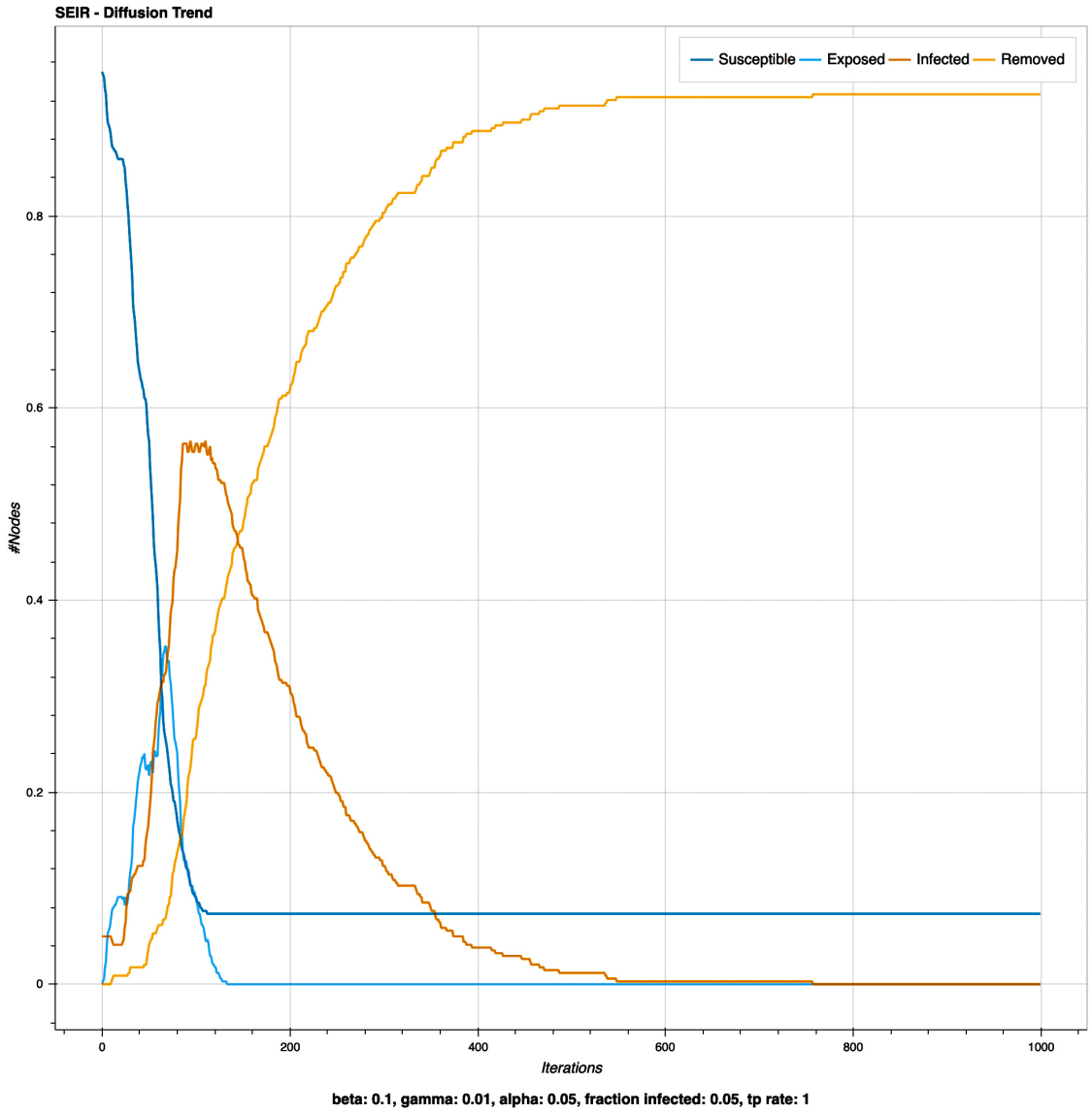

Fig. 5.18.: NFV SEIR output with infection rate at 10%.

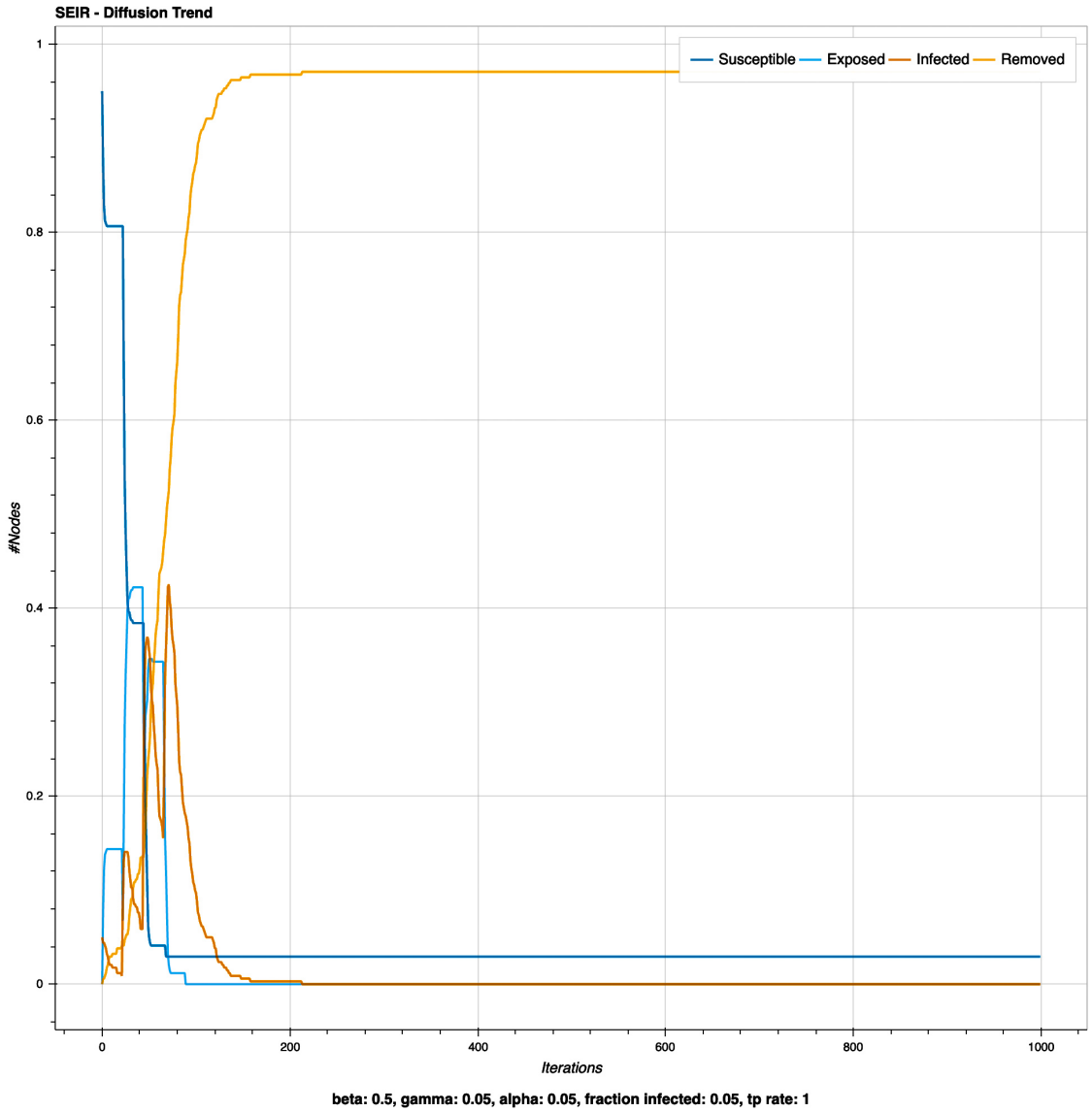

Fig. 5.19.: NFV SEIR output with infection rate at 50%.

(e.g., worm and DoS attack). Figure 5.20 represents the network malware spread when the RNN is only aware of one attack. The percentage of nodes infected is high and has a slow die-out. The number of hosts in the network is also above 40% which is higher than all previous experimental results. When introducing the RNN model into the NFV infrastructure, the host (E) parameter drops from 5% to 1% due to the RNN detecting the second malware in the system. This is exhibited in Figure 5.21, due to this phenomena both the percentage of hosts and the percentage of infected individuals are lowered again back down to less than 30% and 60%, respectively. One thing to point out is that the die-out time for Figure 5.21 is slower than of Figure 5.20. In Figure 5.21, the peak time of infections is reached early on at around the one hundred and twentieth (128th) iteration whereas in Figure 5.20 it is reached after the one hundred and sixtieth (160th) iteration. This slow decline is a result of two different patches being pushed to the network nodes.

The NFV component is where this problem is solved. The NFV gives the ability to administer one patch at the network level and each node uses the virtualization to create a patch that is specific to its protocol. Structuring the network in this manner ensures that more nodes get the patch faster and decreasing the die-out time.

5.4.3 Density and Interconnectivity of Surveillance Zones

The most important factor in the distance bound model is both the density of surveillance zones and the interconnectivity of nodes between surveillance zone nodes. These two factors answer the main dissertation question: can we design a structure that decreases scalabilty and computational limitations. To test this theory two different experiments were run.

The first being how does interconnectivity affect the malware state of each node in the network, specifically if there is an increase in malware (meaning the infection rate *beta* increases)? In theory, the higher the infection the higher the peak infection, but interchangeably due to the sparsity of interconnectivity of nodes, there should

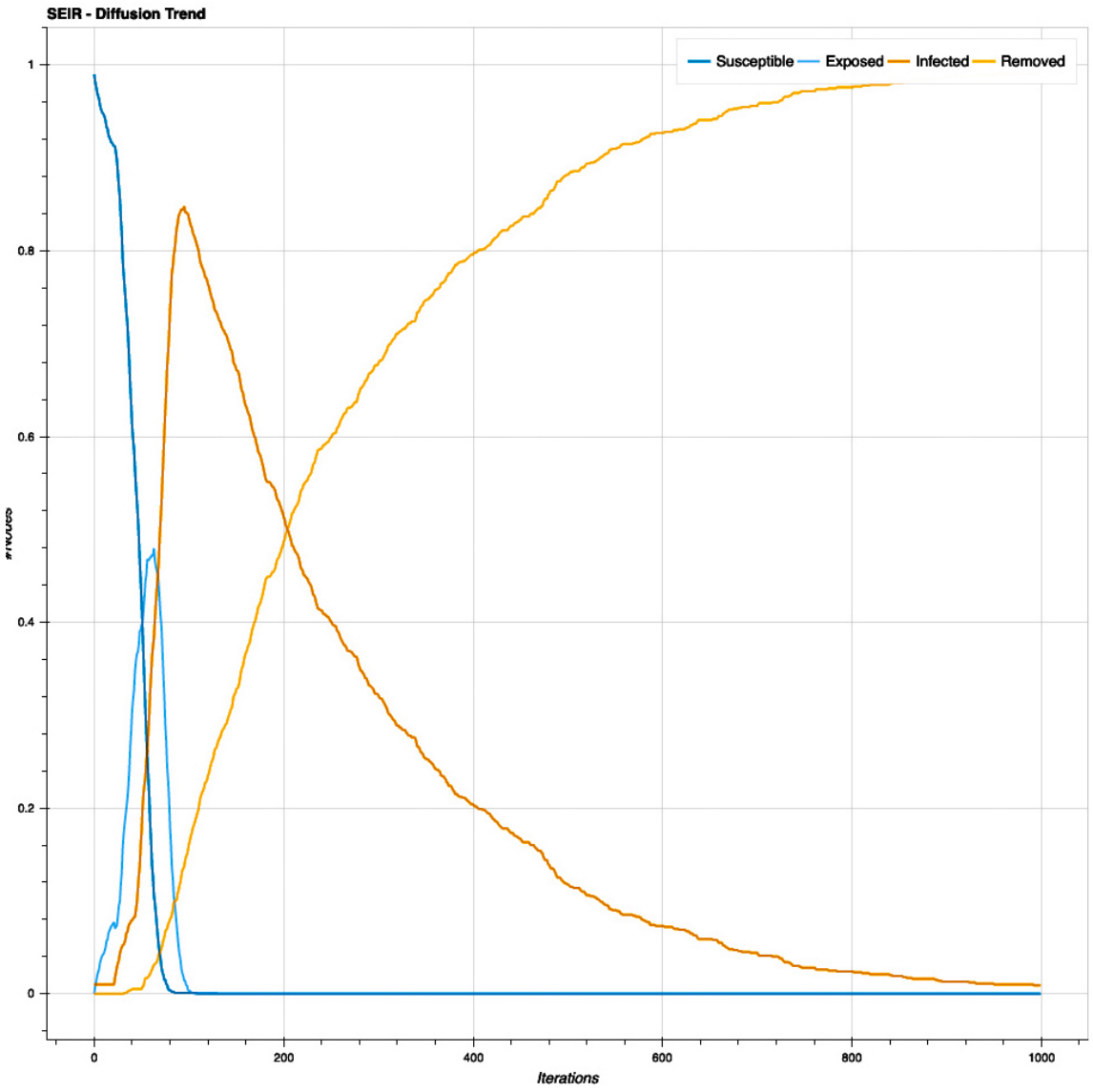

Fig. 5.20.: SEIR Graph with N = 1000.

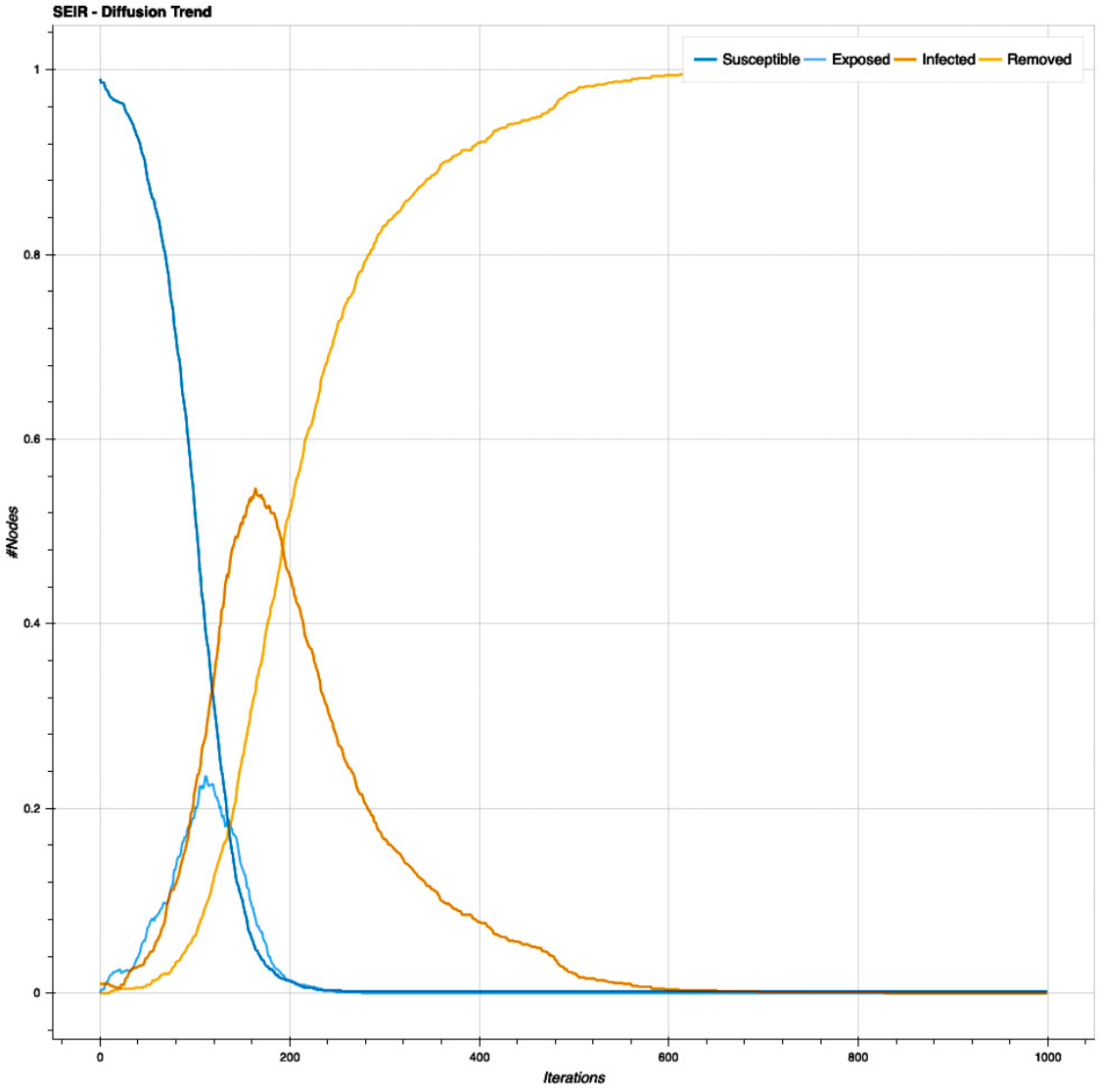

Fig. 5.21.: SEIR Graph with N = 1000.

not be a large infected population. We increase the infection rate (*beta*) to 10% in Figure 5.22 versus 5 % in Figure 5.23 on a sparse interconnected network. A sparse interconnected network in the distance bound case means that our distance bounds for each surveillance zone has limited overlap. Now the results show that for Figure 5.22, even though the peak infected percentage is higher, the number of hosts is still contained to under 40% and the die-out is faster than that of Figure 5.23 due to the sparse network.

To increase the interconnectivity between the nodes with different surveillance zone points, we increase the distance bound patch threshold of the model. This not only sends patches from the surveillance zones but also from other nodes that already have the path available to them. This in turn helps with the scalability of increasing the density to each surveillance zone. Figure 5.24 represents the output percentage of nodes in each state at different points of time in the network. Comparing Figure 5.23 and 5.24, it can be concluded that the increase of nodes' interconnectivity not only increases the number of infected nodes but also increases the number of hosts to the virus.

Overall, when changing the distance bound threshold, the surveillance zone density responds accordingly. All models show that the percentage of infected nodes does not exceed the 60% threshold. Some are recorded as low as 25% infected nodes as seen in Figure 5.25. In this case, the number of surveillance zones were increased to 2% of the total number of nodes in the network.

Overall, we can conclude that the NFV distance bound structure administers and helps with the quarantine of multiple malware attacks. The NFV distance bound structure has the ability to be modified in terms of density and interconnectivity of surveillance zones, to adapt to the scalability needs of the IoT network.

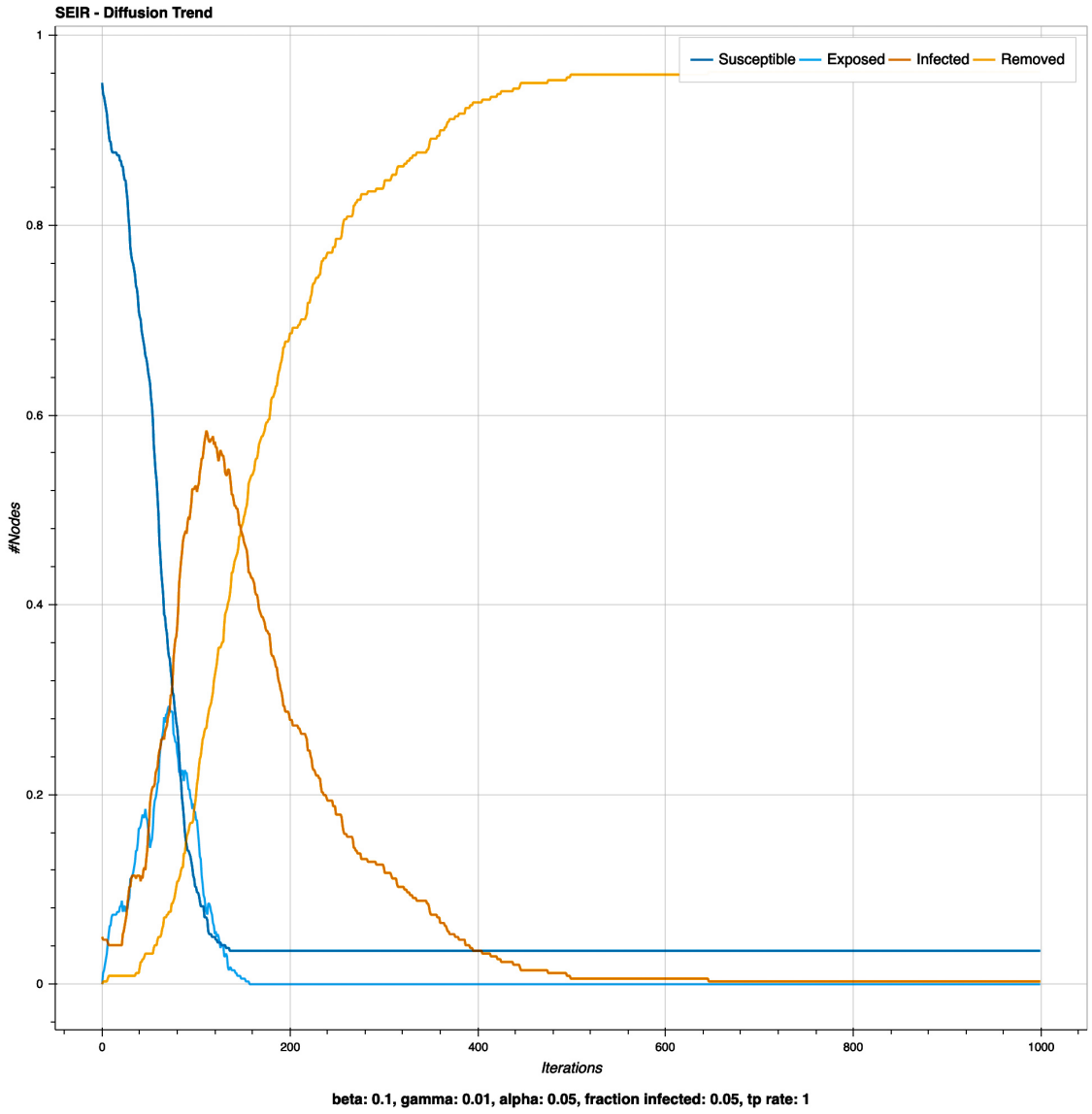

Fig. 5.22.: SEIR output with sparse interconnectivity – high infection rate.

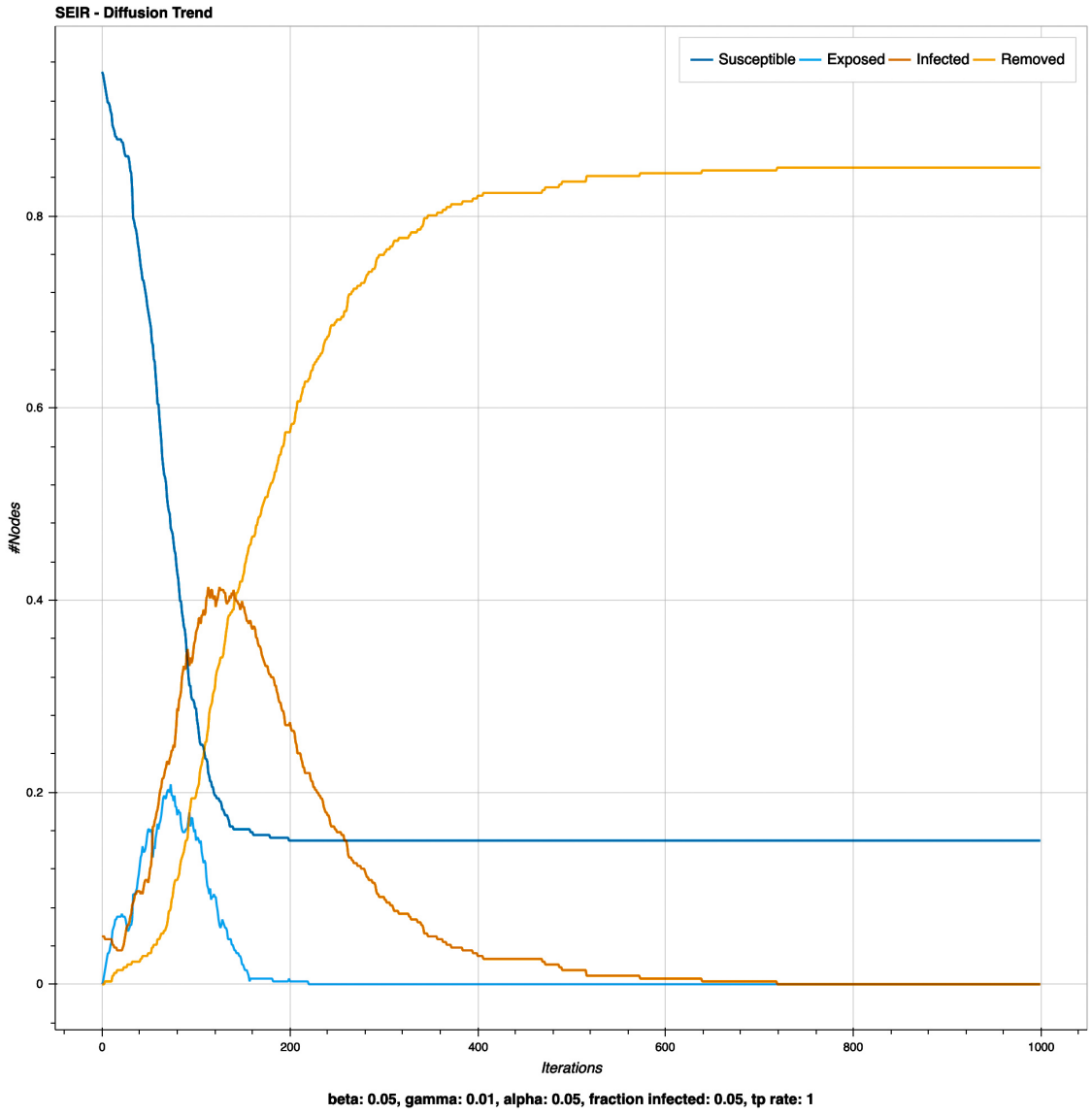

Fig. 5.23.: SEIR output with sparse interconnectivity – low infection rate.

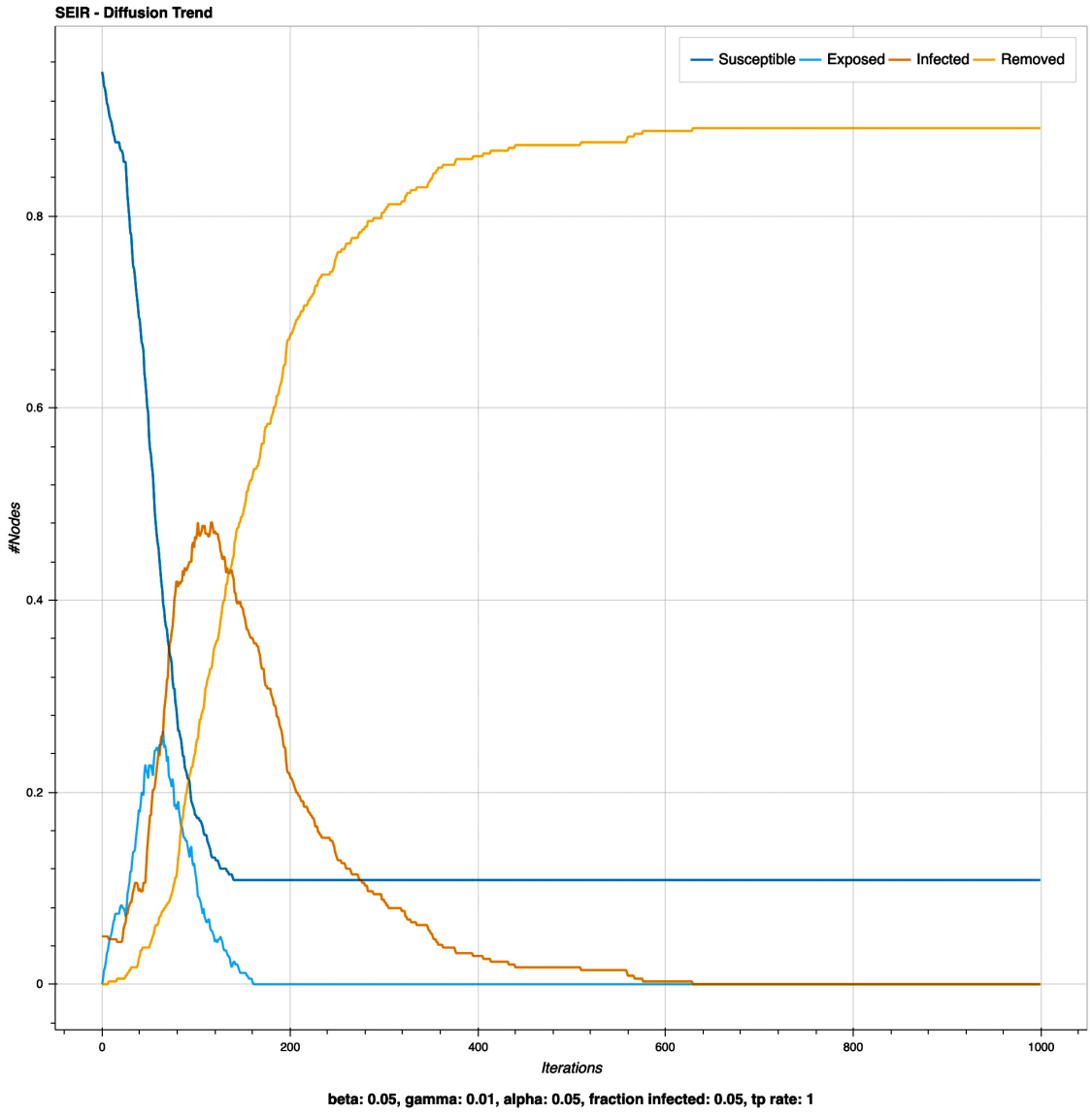

Fig. 5.24.: SEIR output with increase in interconnectivity.

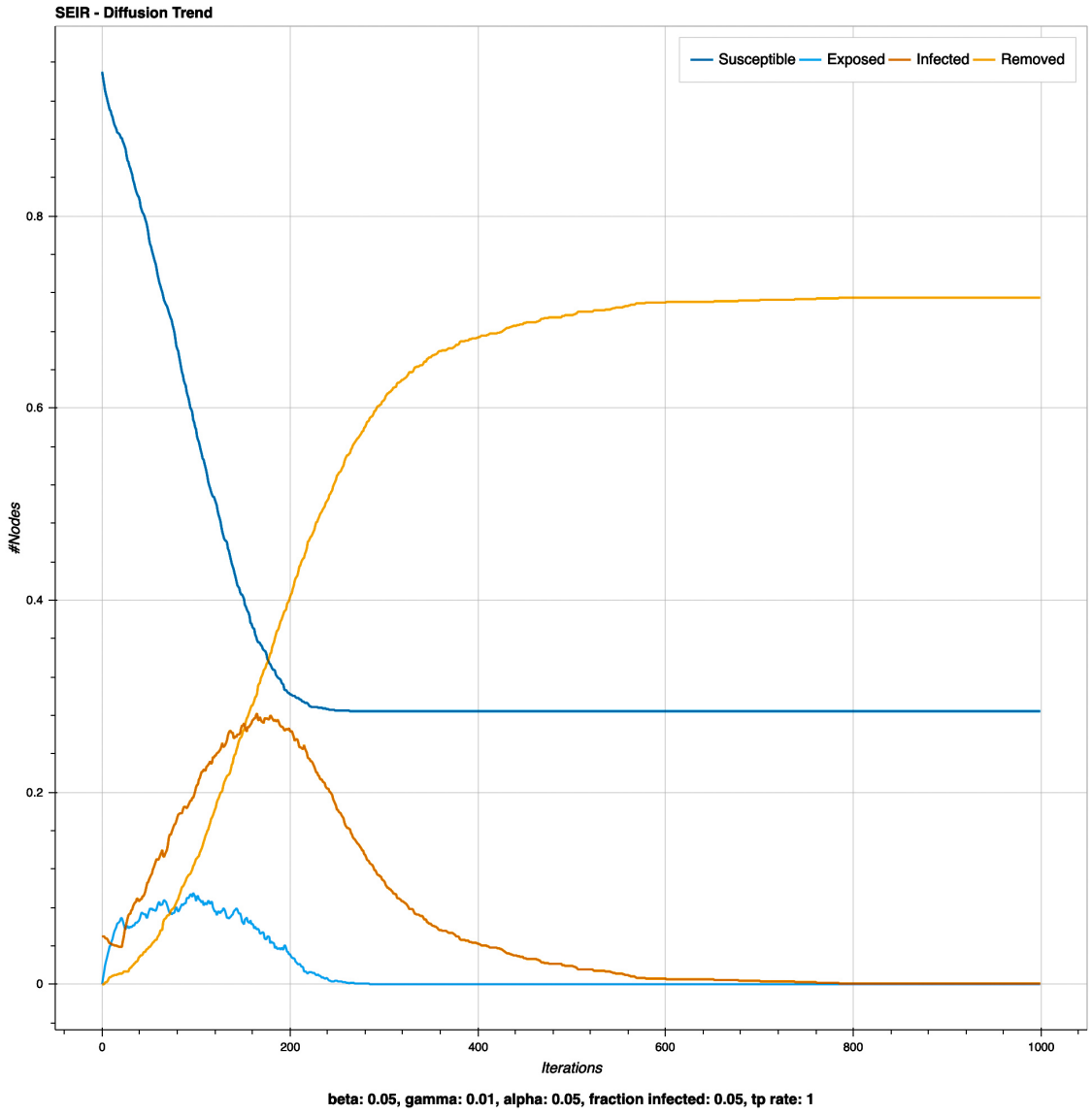

Fig. 5.25.: SEIR output with increase in SZs.

5.5 Performance Evaluation Comparison

To assess the performance of the proposed effectiveness of the NFV surveillance zone architecture, we compare its capability of containing the spread of the malware attack to another hierarchical architecture proposed in previous research. The architecture that is chosen is the hierarchical subnet patching system introduced in [72]. The subnet system architecture is a two-level hierarchy that is divided into superhosts and subnets. Superhosts also are connected to mobile devices as well as to the subnets. A patch is disseminated to the superhosts prior to making it available to their subnets. In the subnet system, there are also overlay connections that exist between subnets that are connected to the same superhost. The NFV architecture and subnet system both use a partitioning mechanism in their network connection structure.

To simulate the two architectures as closely as possible, a combination of network topologies are joined. The NFV architecture is the union of two graph structures consisting of: Gaussian random graph and a 4-3 balanced tree [81] seen in Figure 5.26. The subnet system is represented through the combination of a full 3-ary tree and a random lobster graph [81] as shown in Figure 5.27. Identical malware spread scenarios are run on both the structures. Three metrics are collected which include: (1) the peak percentage of the number of infected nodes, (2) the peak infection percentage of the number of host nodes in the network, and (3) the die-out time of infected nodes. Infected nodes are devices that are compromised due to the malware attack. Host nodes are devices that have the ability to spread the malware to other devices. Die-out time is the time it takes for the malware attack to be eliminated from the network. These three metrics assess if a partitioning strategy for a decentralized IoT network can have any benefit when it comes to detecting and patching a malware attack in a large-scale system.

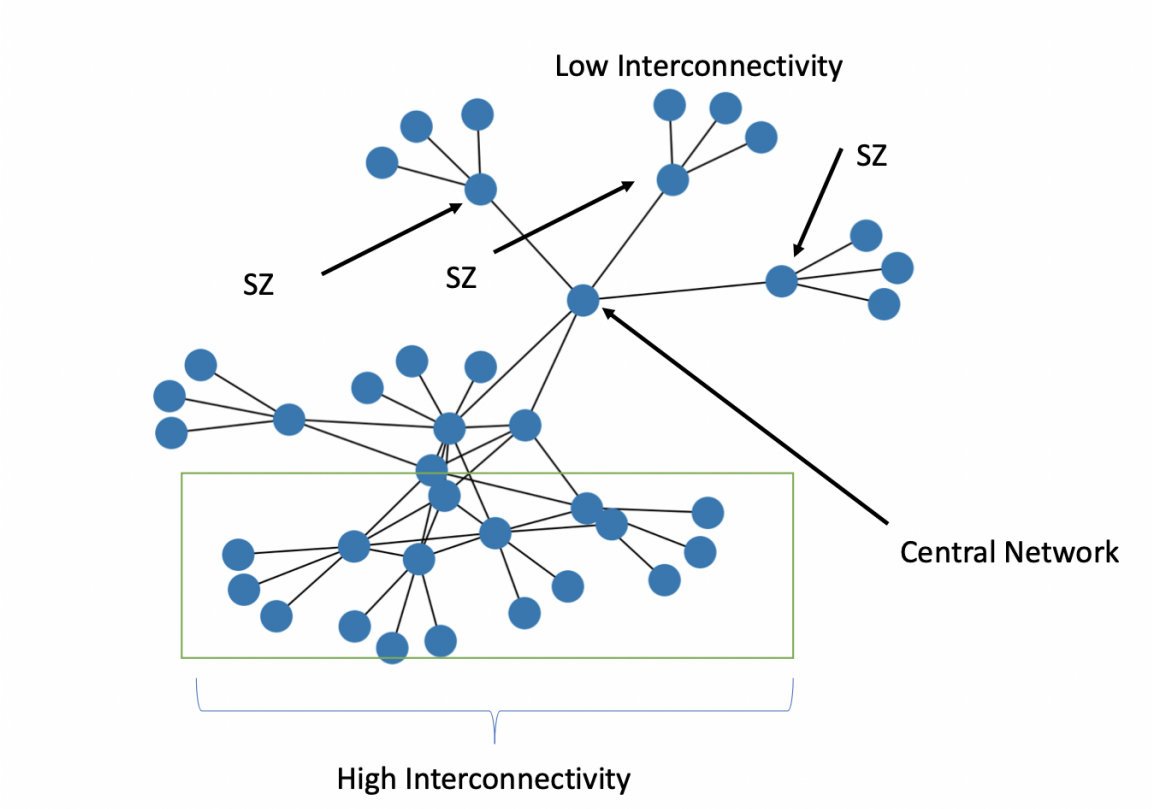

Fig. 5.26.: NFV Graph Architecture.

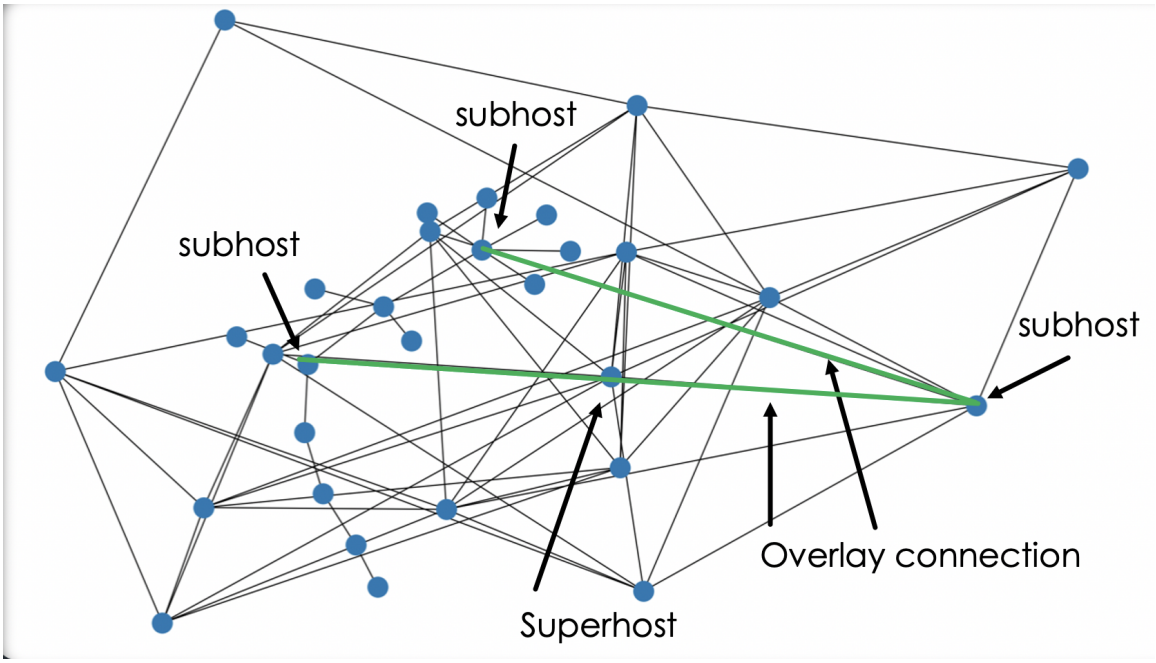

Fig. 5.27.: Hierarchical Subnet Patching Infrastructure.

5.5.1 Experimental Set Up

These selected metrics are not only used to compare the two architectures, but they can also provide information about the scalability aspect of the architecture. Scalability is tested in terms of the number of nodes in the given network with respect to three metrics mentioned above. The SEIR parameters introduced in Chapter 3 are used for this analysis and these parameters are described as follows:

- Infection probability (β)— the probability that the mobile device becomes infected due to a malware attack.
- Incubation period (α) – the time it takes for a mobile device that gets infected with the malware to become a host. A host node is a mobile device that has the ability to spread the malware attack to other devices connected to the same network.
- Removal probability (γ) — the probability that the mobile device receives the patch and becomes a recovered node which is a mobile device that is updated and does not have the malware attack.

For the experiment, four malware scenarios are created by changing the above three parameters. For scenario 1 parameters are: $\beta = 8\%$, $\alpha = 5\%$, and $\gamma = 0.5\%$. For scenario 2 parameters are: $\beta = 10\%$, $\alpha = 5\%$, and $\gamma = 0.5\%$. For scenario 3 parameters are: $\beta = 50\%$, $\alpha = 5\%$, and $\gamma = 0.5\%$. For scenario 4 parameters are: $\beta = 80\%$, $\alpha = 5\%$, and $\gamma = 0.5\%$. In order to analyze different types of malware that have different speed, we change the infection rate in each scenario. The initial attack is set to randomly choose 0.5% of the network nodes. Each scenario is tested for the network size that ranges from ten thousand nodes to one million nodes.

The partitioning of the network nodes is set up to have 2% of the network connected to one surveillance node. Similarly, for the subnet system, 2% of the network is connected to one subhost. In addition, the overlay percentage which is the number of subhosts that are connected to each other is set to 0.33%. In the results that are

shown in Figures 5.28, 5.29, and 5.30, the NFV architecture is represented as SZ and the subnet system is represented as RM.

Metric 1: Peak Host Population (E_p)

The first metric we analyze is the host population percentage (E_p) which is represented as the percentage of population in sate E of the SEIR model. This is an important metric due to the fact that the host nodes have the ability to spread the attack. As shown in Figure 5.28, the RM architecture outperforms the SZ architecture for the case of a small network size, for example around 5000 nodes. However, when the number of nodes in the network increases, it can be noticed that the SZ infrastructure yields lower percentages of both infected and host nodes. The host population percentage (E_p) has a direct effect on the population of infected nodes (I_p) in the network. An important observation is that spread seems to decrease for the SZ and not for the RM network as the size of the network increases. A possible reason for this decrease in the case of RM is due to the overlay connections present in this network. Subsequently, it increases the number of nodes that can be susceptible to attack. Consequently, in this scenario the partitioning of the network does not provide any advantage rather it becomes a hindrance to the performance. As the number of subhosts in the network increases, the number of overlays between subhosts also increases. When a subhost receives the malware attack, the overlays make the network more susceptible to the attack, this causes performance degradation. In the NFV based partitioning, nodes connected to the same SZ are only connected to the nodes that are within the SZ bound. This implicitly provides an attack containment mechanism. In other words, if a node, in a SZ is infected with a malware, that node is not only contained within the specific SZ but also all nodes within the same SZ have the patch administered prior to the rest of the network. This "partition-driven containment" mechanism can address the scalability aspect in the sense that the percentage of the hosts remains constant. This can be observed for the case of one

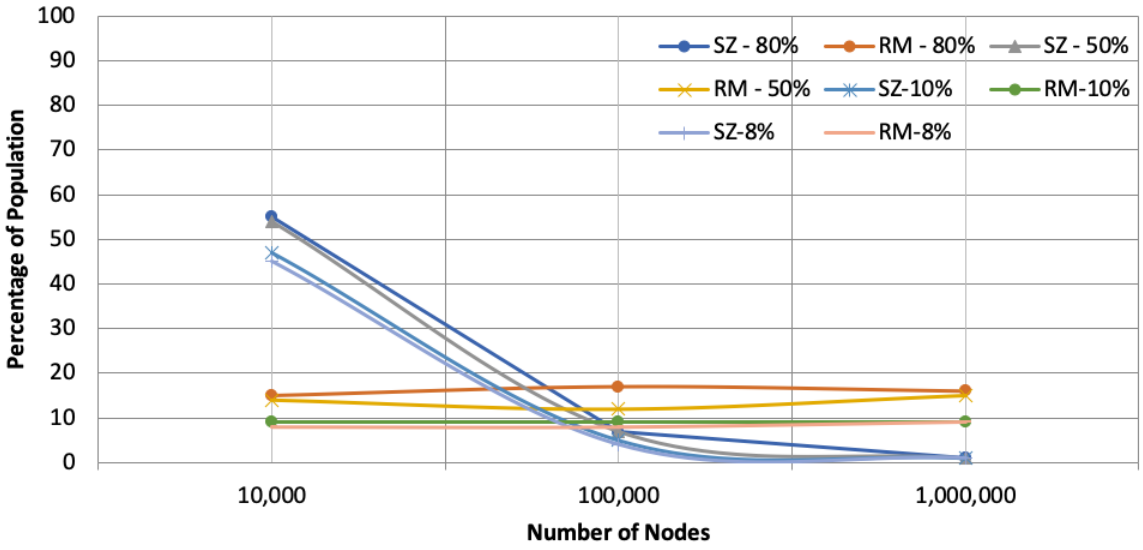

Fig. 5.28.: Host Peak Comparison.

million nodes where the percentage of hosts in the network remains rather constant due to the 2% partitioning study.

Metric 2: Peak Infected Population (I_p)

In connection with E_p we also examine the peak infected population (I_p) which is represented as the percentage of population in state I of SEIR model. This is to determine if there is any correlation between the two parameters. If the latter is true, it further confirms that the partitioning of the NFV structure is better due to there being no overlays between the SZs. Figure 5.29 depicts the peak infection percentage (I_p) for different network sizes at different infection rates (β). From this Figure, it can be observed that this percentage increases as the number of nodes increases for the RM structure. For the partitioning of the SZs to be 2% of the total population, we notice that the peak infected population percentage (I_p) decreases as the number of nodes in the network increases. The reason is due to the RM subhost interconnections that is always increasing with the size of the network. However in the

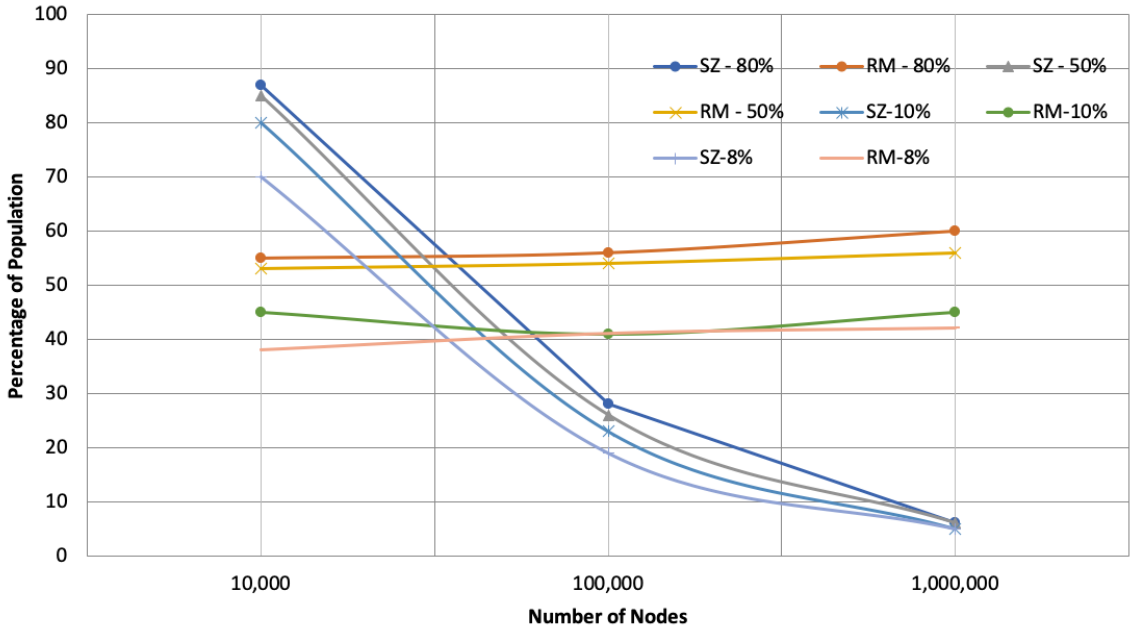

Fig. 5.29.: Infection Peak Comparison.

case of SZ partitioning, the number of direct interconnections between nodes reduces due to having a bound for each SZ. Not only do we observe a decrease in I_p, but even with the increase of the infection rate (β). I_p stays constant for the SZ structure. In other words even when the malware attack is aggressive/contagious the partitioning keeps the attack contained.

Metric 3: Infection Die-Out Rate (I_t)

Another important metric to analyze and discuss is how long does the malware attack stay in the network. This gives us an idea if the patching mechanism distributes patches in a timely manner. For this propose, we define the third metric which is the infection die-out rate (I_t) which is plotted in Figure 5.30. These results also give us an insight on the containment of the malware attack in the network as a result of the partitioning strategy. Both the RM system and the SZ system are analyzed with the same four malware attack scenarios mentioned above. The trend seen in

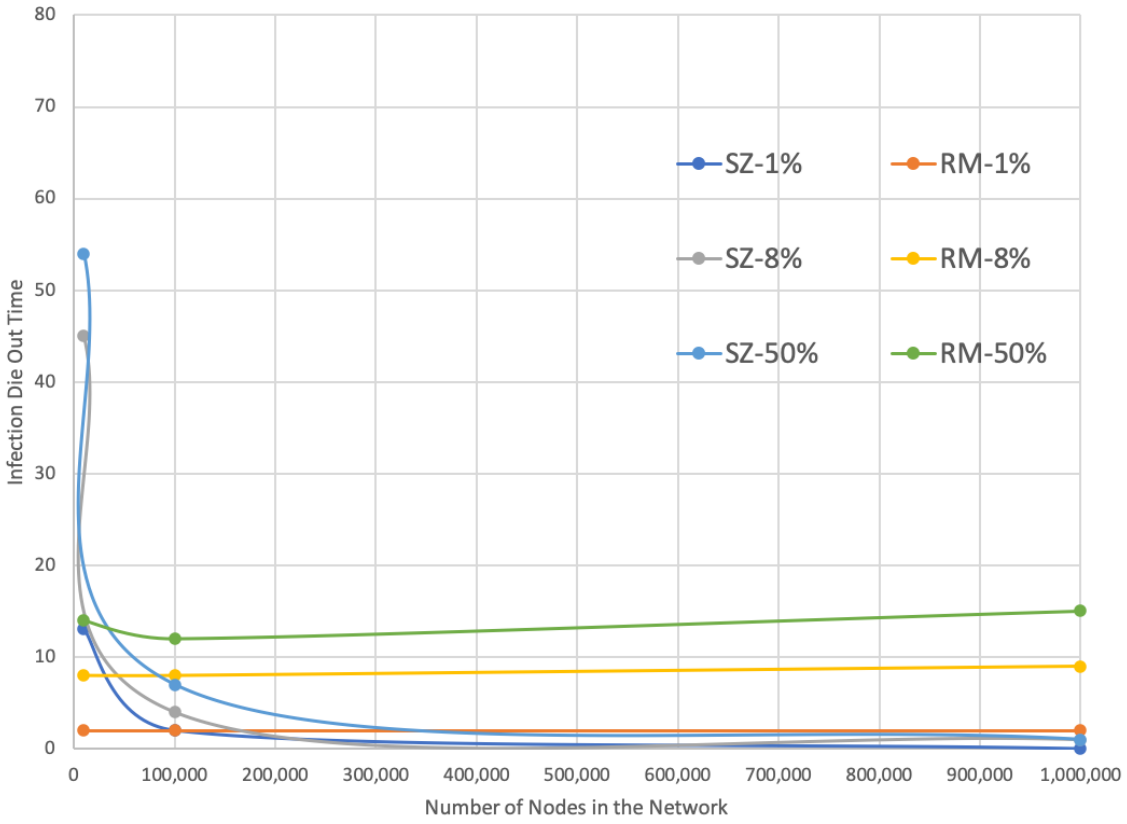

Fig. 5.30.: Infection Die-Out Comparison.

Figure 5.30 is similar to the trend results that are noted in the other two metrics mentioned above. For example, with a small network size, the die-out time for RM is substantially less than that of the SZ infrastructure. However, when the infection probability (β) and the number of nodes increase, the SZ outperforms the RM as the network grows in size.

In order to compare the impact of distributed versus non-distributed systems, we create the scenario of the NFV structure with only one SZ. In other words, there is no partition present in the network (the SZ density is at 100%). As seen from the results listed in Table 5.4, the SZ exceeds in terms of metric performance in comparison to the 100% structure. When the SZ is at 100%, then the performance is extremely poor in comparison with the partition study of 2% in terms of both E_p and I_p. We

Table 5.4: Comparison of the 2% partition (SZ) versus 100% SZ (RW) structure for all Three Metrics

Infection Probability (β)	SZ(I_p)	RW(I_p)	SZ(E_p)	RW(E_p)	SZ(I_t)	RW(I_t)
1%	48	92	8	83	900	1200
5%	64	96	28	84	800	1200
10%	68	98	32	86	600	1200
50%	70	98	42	86	600	1000

test this special case against the 2% partition study. Given an aggressive malware attack (high infection probability), the partitioning gives way for the malware attack to not spread to other parts of the network and the patch produces a shorter I_t. This supports our hypothesis that partitioning is in fact a better option in terms of containing and patching the malware attack.

5.5.2 Summary of Contribution

In conclusion, the NFV architecture is more advantageous infrastructure when it comes to dealing with large-scale networks. The subnet system is a better choice when the network is not large (under 10,000 nodes). What is causing the performance difference is the strict delineation of partitioning in the NFV bound system. Propagation of the malware does not happen between the SZ gateways. Whereas, in the subnet system, if a subhost comes under a malware attack, can spread through the overlay connections of the sub hosts. Another advantage of the NFV infrastructure is that it can administer selective patching to multiple malware attacks due to the virtualization function.

6. CONCLUSION AND FUTURE RESEARCH PLAN

In this chapter, a summary of contributions will be listed in Section 6.1 as well as future research plans in Section 6.2.

6.1 Summary of Contributions

Chapter 1 and 2 give the introduction and the literature review. In Chapter 3, we addressed the challenges of scalability in terms of predicting events in a network. The first step was to develop an agent based socio-temporal population network to be used for disease spread analysis. Creating an activity level data network gave us the ability to create a disease spread model that correctly describes the population network. The second step was to create an SEIR-FSM disease spread model that can accommodate the prediction on three levels of an aggregated socio-temporal human network. The disease spread model was tested on three different aggregated levels of the socio temporal network to compare the accuracy of prediction in terms of disease spread.

In Chapter 4, we introduced mobile ad hoc networks and its applicable connections to the socio-temporal population network. The SEIR-FSM and its multilevel aggregation representation have been applied to MANETs to understand the dynamics associated with the virus spread in these networks. Subsequently, we introduced two security architectures for monitoring and responding to viruses for MANETs. Subsequently, a generalized classification mobile network model is introduced. Expanding on the SEIR model in Chapter 3, each HMM class is a corresponding SEIR-FSM. A mobile agent can be classified in one of five classes depending on the type of threat the mobile has on the network. The classification model was developed in this way for future use, if a slightly different threat appears (i.e., a new HMM class), the overar-

ching model does not have to change. This new threat can be attached as a new class in the generalized classification model. This specific technique can be seen as another solution to the scalability of the ever growing mobile industry and its accompanying threats to privacy.

In Chapter 5, we tackled the analysis of the security architecture introduced in Chapter 4. We defined the architecture as a decentralized network function virtualization (NFV) distance bound patching system. This network structure not only provides a scalable infrastructure for sending out patches to stop malware attack spread, but also provides a system in which different mobile components that utilize different communication and security protocols accept the same patch. The SEIR-FSM model was utilized to compare peak infection time, die-out rate, and other metrics on the changing topology of the decentralized NFV architecture.

6.2 Future Work

As future research work, the following research can be pursued: First possible research direction is the development and analysis of the proposed practical IDS to solve the race condition problem not only in the IoT devices/networks but but also to generalize for all networks. The overarching challenge for this work was to solve the limitation of scalability and computation, that has yet to be answered on the NFV level. That is the next challenge that we would like to address. In addition, can we construct an NFV structure in itself able to handle a patch system that can be sent to multiple standards across millions of devices?

Other questions yet to be answered are:

1. Does aggregation of IoT data help or hinder the race condition between the patch and the virus?

2. Does the NFV itself hold too much overhead that will hinder the race condition between patch and virus?

3. How to create a smooth transition from a hierarchical subnet patching system to an NFV distance bound system?

4. Investigate the optimal percentage for surveillance zone densities on any given network size.

5. Improve the analysis by using different analytical schemes.

6. Further analyze the scalability of the mobile network as it grows and its accompanying threats to privacy.

LIST OF REFERENCES

LIST OF REFERENCES

[1] A. L. Lloyd and V. A. Jansen, "Spatiotemporal dynamics of epidemics: synchrony in metapopulation models," *Mathematical biosciences*, vol. 188, no. 1-2, pp. 1–16, 2004.

[2] S. Agarwal, R. H. Katz, S. V. Krishnamurthy, and S. K. Dao, "Distributed power control in ad-hoc wireless networks," in *12th IEEE International Symposium on Personal, Indoor and Mobile Radio Communications. PIMRC 2001. Proceedings (Cat. No. 01TH8598)*, vol. 2, pp. F–F, IEEE, 2001.

[3] C. Rudin, D. Waltz, R. N. Anderson, A. Boulanger, A. Salleb-Aouissi, M. Chow, H. Dutta, P. N. Gross, B. Huang, S. Ierome, *et al.*, "Machine learning for the new york city power grid," *IEEE transactions on pattern analysis and machine intelligence*, vol. 34, no. 2, pp. 328–345, 2011.

[4] E. Mossel, N. Olsman, and O. Tamuz, "Efficient bayesian learning in social networks with gaussian estimators," in *Communication, Control, and Computing (Allerton), 2016 54th Annual Allerton Conference on*, pp. 425–432, IEEE, 2016.

[5] C. E. Rasmussen, "Gaussian processes in machine learning," in *Summer School on Machine Learning*, pp. 63–71, Springer, 2003.

[6] R. Miguez, M. Georgiopoulos, and A. Kaylani, "G-pnn: A genetically engineered probabilistic neural network," *Nonlinear Analysis: Theory, Methods & Applications*, vol. 73, no. 6, pp. 1783–1791, 2010.

[7] D. Turgut, S. K. Das, R. Elmasri, and B. Turgut, "Optimizing clustering algorithm in mobile ad hoc networks using genetic algorithmic approach," in *Global Telecommunications Conference, 2002. GLOBECOM'02. IEEE*, vol. 1, pp. 62–66, IEEE, 2002.

[8] M. Lahiri and M. Cebrian, "The genetic algorithm as a general diffusion model for social networks," in *Twenty-Fourth AAAI Conference on Artificial Intelligence*, 2010.

[9] D. Mantzaris, G. Anastassopoulos, and A. Adamopoulos, "Genetic algorithm pruning of probabilistic neural networks in medical disease estimation," *Neural Networks*, vol. 24, no. 8, pp. 831–835, 2011.

[10] P. Wood, B. Nahorney, K. Chandrasekar, S. Wallace, and K. Haley, "Internet security threat report," Apr 2016.

[11] M. Fossi, G. Egan, K. Haley, E. Johnson, T. Mack, T. Adams, J. Blackbird, M. K. Low, D. Mazurek, D. McKinney, *et al.*, "Symantec internet security threat report trends for 2010," *Volume XVI*, 2011.

[12] M. S. Mahdavinejad, M. Rezvan, M. Barekatain, P. Adibi, P. Barnaghi, and A. P. Sheth, "Machine learning for internet of things data analysis: A survey," *Digital Communications and Networks*, vol. 4, no. 3, pp. 161–175, 2018.

[13] B. Nguyen and L. Simkin, "The internet of things (iot) and marketing: the state of play, future trends and the implications for marketing," 2017.

[14] C. Perera, A. Zaslavsky, P. Christen, and D. Georgakopoulos, "Context aware computing for the internet of things: A survey," *IEEE communications surveys & tutorials*, vol. 16, no. 1, pp. 414–454, 2013.

[15] M. Ojo, D. Adami, and S. Giordano, "A sdn-iot architecture with nfv implementation," in *2016 IEEE Globecom Workshops (GC Wkshps)*, pp. 1–6, IEEE, 2016.

[16] I. Farris, T. Taleb, Y. Khettab, and J. Song, "A survey on emerging sdn and nfv security mechanisms for iot systems," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 1, pp. 812–837, 2018.

[17] M. Condoluci and T. Mahmoodi, "Softwarization and virtualization in 5g mobile networks: Benefits, trends and challenges," *Computer Networks*, vol. 146, pp. 65–84, 2018.

[18] C. Benzaid and T. Taleb, "Zsm security: Threat surface and best practices," *IEEE Network*, 2020.

[19] A. Boudi, I. Farris, M. Bagaa, and T. Taleb, "Lightweight virtualization based security framework for network edge," in *2018 IEEE Conference on Standards for Communications and Networking (CSCN)*, pp. 1–6, IEEE, 2018.

[20] A. Boudi, I. Farris, M. Bagaa, and T. Taleb, "Assessing lightweight virtualization for security-as-a-service at the network edge," *IEICE Transactions on Communications*, vol. 102, no. 5, pp. 970–977, 2019.

[21] P. Porambage, J. Okwuibe, M. Liyanage, M. Ylianttila, and T. Taleb, "Survey on multi-access edge computing for internet of things realization," *IEEE Communications Surveys & Tutorials*, vol. 20, no. 4, pp. 2961–2991, 2018.

[22] "Stages of "internet of things" architecture: Marlabs," Sep 2019.

[23] A. Grizhnevich, "Iot architecture: building blocks and how they work," Apr 2019.

[24] H. Kaaniche and F. Kamoun, "Mobility prediction in wireless ad hoc networks using neural networks," *arXiv preprint arXiv:1004.4610*, 2010.

[25] I. Guyon, S. Gunn, A. Ben-Hur, and G. Dror, "Result analysis of the nips 2003 feature selection challenge," in *Advances in neural information processing systems*, pp. 545–552, 2005.

[26] I. A. Gheyas and L. S. Smith, "Feature subset selection in large dimensionality domains," *Pattern recognition*, vol. 43, no. 1, pp. 5–13, 2010.

[27] A. Singla and E. Bertino, "How deep learning is making information security more intelligent," *IEEE Security & Privacy*, vol. 17, no. 3, pp. 56–65, 2019.

[28] E. Hodo, X. Bellekens, A. Hamilton, P.-L. Dubouilh, E. Iorkyase, C. Tachtatzis, and R. Atkinson, "Threat analysis of iot networks using artificial neural network intrusion detection system," in *2016 International Symposium on Networks, Computers and Communications (ISNCC)*, pp. 1–6, IEEE, 2016.

[29] M. Opuszko, F.-s.-u. Jena, J. Ruhland, and F.-s.-u. Jena, "Impact of the Network Structure on the SIR Model Spreading Phenomena in Online Networks," no. c, pp. 22–28, 2013.

[30] D. Xu, "Modeling and Control of Dynamic Network SIR Based on Community Structure *," 2014.

[31] S. Eubank, V. Kumar, M. V. Marathe, A. Srinivasan, and N. Wang, "Structural and algorithmic aspects of massive social networks," in *Proceedings of the fifteenth annual ACM-SIAM symposium on Discrete algorithms*, pp. 718–727, Society for Industrial and Applied Mathematics, 2004.

[32] M. J. Keeling and K. T. Eames, "Networks and epidemic models," *Journal of The Royal Society Interface*, vol. 2, no. 4, pp. 295–307, 2005.

[33] H. Rahmandad and J. Sterman, "Heterogeneity and Network Structure in the Dynamics of Diffusion: Comparing Agent-Based and Differential Equation Models," *Management Science*, vol. 54, no. 5, pp. 998–1014, 2008.

[34] E.-S. M. El-Alfy and F. N. Al-Obeidat, "Detecting cyber-attacks on wireless mobile networks using multicriterion fuzzy classifier with genetic attribute selection," *Mobile Information Systems*, vol. 2015, 2015.

[35] A. Bose, X. Hu, K. G. Shin, and T. Park, "Behavioral detection of malware on mobile handsets," in *Proceedings of the 6th international conference on Mobile systems, applications, and services*, pp. 225–238, ACM, 2008.

[36] A. Shabtai, Y. Fledel, and Y. Elovici, "Automated static code analysis for classifying android applications using machine learning," in *Computational Intelligence and Security (CIS), 2010 International Conference on*, pp. 329–333, IEEE, 2010.

[37] J. C. Badajena and C. Rout, "Incorporating hidden markov model into anomaly detection technique for network intrusion detection," *International Journal of Computer Applications*, vol. 53, no. 11, 2012.

[38] D.-h. Lee, D.-y. Kim, and J.-i. Jung, "Mobile agent based intrusion detection system adopting hidden markov model," in *International Conference on Computational Science and Its Applications*, pp. 122–130, Springer, 2007.

[39] K. Haslum, M. E. Moe, and S. J. Knapskog, "Real-time intrusion prevention and security analysis of networks using hmms," in *Local Computer Networks, 2008. LCN 2008. 33rd IEEE Conference on*, pp. 927–934, IEEE, 2008.

[40] P. Holgado, V. A. Villagra, and L. Vazquez, "Real-time multistep attack prediction based on hidden markov models," *IEEE Transactions on Dependable and Secure Computing*, 2017.

[41] P. Hu, Z. Zhou, Q. Liu, and F. Li, "The hmm-based modeling for the energy level prediction in wireless sensor networks," in *Industrial Electronics and Applications, 2007. ICIEA 2007. 2nd IEEE Conference on*, pp. 2253–2258, IEEE, 2007.

[42] S. Bu, F. R. Yu, X. P. Liu, P. Mason, and H. Tang, "Distributed combined authentication and intrusion detection with data fusion in high-security mobile ad hoc networks," *IEEE transactions on vehicular technology*, vol. 60, no. 3, pp. 1025–1036, 2011.

[43] L. R. Rabiner, "A tutorial on hidden markov models and selected applications in speech recognition," *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257–286, 1989.

[44] M.-k. Choi, R. J. Robles, C.-h. Hong, and T.-h. Kim, "Wireless network security: Vulnerabilities, threats and countermeasures," *International Journal of Multimedia and Ubiquitous Engineering*, vol. 3, no. 3, pp. 77–86, 2008.

[45] K. Pelechrinis, M. Iliofotou, and S. V. Krishnamurthy, "Denial of service attacks in wireless networks: The case of jammers," *IEEE Communications surveys & tutorials*, vol. 13, no. 2, pp. 245–257, 2011.

[46] G. S. Dhillon, "Vulnerabilities & attacks in mobile adhoc networks (manet)," *International Journal of Advanced Research in Computer Science*, vol. 8, no. 4, 2017.

[47] H. HaddadPajouh, A. Dehghantanha, R. Khayami, and K.-K. R. Choo, "A deep recurrent neural network based approach for internet of things malware threat hunting," *Future Generation Computer Systems*, vol. 85, pp. 88–96, 2018.

[48] F. Alam, R. Mehmood, I. Katib, and A. Albeshri, "Analysis of eight data mining algorithms for smarter internet of things (iot)," *Procedia Computer Science*, vol. 98, pp. 437–442, 2016.

[49] L. Xiao, X. Wan, X. Lu, Y. Zhang, and D. Wu, "Iot security techniques based on machine learning," *arXiv preprint arXiv:1801.06275*, 2018.

[50] H. Li, K. Ota, and M. Dong, "Learning iot in edge: Deep learning for the internet of things with edge computing," *IEEE Network*, vol. 32, no. 1, pp. 96–101, 2018.

[51] P. Bull, R. Austin, E. Popov, M. Sharma, and R. Watson, "Flow based security for iot devices using an sdn gateway," in *2016 IEEE 4th International Conference on Future Internet of Things and Cloud (FiCloud)*, pp. 157–163, IEEE, 2016.

[52] E. Gelenbe and Y. Yin, "Deep learning with dense random neural networks," in *International Conference on Man–Machine Interactions*, pp. 3–18, Springer, 2017.

[53] R. Nix and J. Zhang, "Classification of android apps and malware using deep neural networks," in *2017 International joint conference on neural networks (IJCNN)*, pp. 1871–1878, IEEE, 2017.

[54] Y. Meidan, M. Bohadana, A. Shabtai, J. D. Guarnizo, M. Ochoa, N. O. Tippenhauer, and Y. Elovici, "Profiliot: a machine learning approach for iot device identification based on network traffic analysis," in *Proceedings of the symposium on applied computing*, pp. 506–509, ACM, 2017.

[55] J. Canedo and A. Skjellum, "Using machine learning to secure iot systems," in *2016 14th Annual Conference on Privacy, Security and Trust (PST)*, pp. 219–222, IEEE, 2016.

[56] M. Nauman, T. A. Tanveer, S. Khan, and T. A. Syed, "Deep neural architectures for large scale android malware analysis," *Cluster Computing*, vol. 21, no. 1, pp. 569–588, 2018.

[57] F. Martinelli, F. Marulli, and F. Mercaldo, "Evaluating convolutional neural network for effective mobile malware detection," *Procedia computer science*, vol. 112, pp. 2372–2381, 2017.

[58] X. Hu, X. Li, E. C.-H. Ngai, V. C. Leung, and P. Kruchten, "Multidimensional context-aware social network architecture for mobile crowdsensing," *IEEE Communications Magazine*, vol. 52, no. 6, pp. 78–87, 2014.

[59] Z. Sanaei, S. Abolfazli, A. Gani, and M. Shiraz, "Sami: Service-based arbitrated multi-tier infrastructure for mobile cloud computing," in *2012 1st IEEE International Conference on Communications in China Workshops (ICCC)*, pp. 14–19, IEEE, 2012.

[60] F. C. Delicato, P. F. Pires, L. Pinnez, L. Fernando, and L. Da Costa, "A flexible web service based architecture for wireless sensor networks," in *23rd International Conference on Distributed Computing Systems Workshops, 2003. Proceedings.*, pp. 730–735, IEEE, 2003.

[61] A. V. Bakre and T. Nishida, "Switch based network architecture for ip multicast and integrated services," Dec. 30 2003. US Patent 6,671,276.

[62] K. K. Patel, S. M. Patel, *et al.*, "Internet of things-iot: definition, characteristics, architecture, enabling technologies, application & future challenges," *International journal of engineering science and computing*, vol. 6, no. 5, 2016.

[63] L. Perez and S. Dragicevic, "An agent-based approach for modeling dynamics of contagious disease spread," *International journal of health geographics*, vol. 8, no. 1, p. 50, 2009.

[64] E. Teweldemedhin, T. Marwala, and C. Mueller, "Agent-based modelling: a case study in hiv epidemic," in *Hybrid Intelligent Systems, 2004. HIS'04. Fourth International Conference on*, pp. 154–159, IEEE, 2004.

[65] R. Maciejewski, P. Livengood, S. Rudolph, T. F. Collins, D. S. Ebert, R. T. Brigantic, C. D. Corley, G. A. Muller, and S. W. Sanders, "A pandemic influenza modeling and visualization tool," *Journal of Visual Languages & Computing*, vol. 22, no. 4, pp. 268–278, 2011.

[66] H. V. D. Parunak, R. Savit, and R. L. Riolo, "Agent-based modeling vs. equation-based modeling: A case study and users' guide," in *International Workshop on Multi-Agent Systems and Agent-Based Simulation*, pp. 10–25, Springer, 1998.

[67] C. L. Barrett, K. R. Bisset, S. G. Eubank, X. Feng, and M. V. Marathe, "Episimdemics: an efficient algorithm for simulating the spread of infectious disease over large realistic social networks," in *High Performance Computing, Networking, Storage and Analysis, 2008. SC 2008. International Conference for*, pp. 1–12, IEEE, 2008.

[68] M. Sahar, *A framework for synthesizing agent-based heterogeneous population model for epidemic simulation.* PhD thesis, Purdue University, 2014.

[69] M. Musolesi and C. Mascolo, "Controlled epidemic-style dissemination middleware for mobile ad hoc networks," in *Mobile and Ubiquitous Systems-Workshops, 2006. 3rd Annual International Conference on*, pp. 1–9, IEEE, 2006.

[70] P. Mell, K. Scarfone, and S. Romanosky, "A complete guide to the common vulnerability scoring system (cvss), version 2.0, forum of incident response and security teams," 2007.

[71] P. Mell, K. Scarfone, and S. Romanosky, "A complete guide to the common vulnerability scoring system version 2.0," in *Published by FIRST-forum of incident response and security teams*, vol. 1, p. 23, 2007.

[72] M. Vojnovic and A. J. Ganesh, "On the race of worms, alerts, and patches," *IEEE/ACM Transactions on Networking*, vol. 16, no. 5, pp. 1066–1079, 2008.

[73] N. Guizani, A. Elghariani, J. Kobes, and A. Ghafoor, "Effects of social network structure on epidemic disease spread dynamics with application to ad hoc networks," *IEEE Network*, vol. 33, no. 3, pp. 139–145, 2019.

[74] E. Rahm and H. H. Do, "Data cleaning: Problems and current approaches," *IEEE Data Eng. Bull.*, vol. 23, no. 4, pp. 3–13, 2000.

[75] M. A. Hall, "Correlation-based feature selection of discrete and numeric class machine learning," 2000.

[76] G. Chen, "A gentle tutorial of recurrent neural network with error backpropagation," *arXiv preprint arXiv:1610.02583*, 2016.

[77] N. L. D. Khoa, K. Sakakibara, and I. Nishikawa, "Stock price forecasting using back propagation neural networks with time and profit based adjusted weight factors," in *2006 SICE-ICASE International Joint Conference*, pp. 5484–5488, IEEE, 2006.

[78] N. Koroniotis, N. Moustafa, E. Sitnikova, and B. Turnbull, "Towards the development of realistic botnet dataset in the internet of things for network forensic analytics: Bot-iot dataset," *Future Generation Computer Systems*, vol. 100, pp. 779–796, 2019.

[79] N. Moustafa and J. Slay, "Unsw-nb15: a comprehensive data set for network intrusion detection systems (unsw-nb15 network data set)," in *2015 military communications and information systems conference (MilCIS)*, pp. 1–6, IEEE, 2015.

[80] B. K. Mishra and D. K. Saini, "Seirs epidemic model with delay for transmission of malicious objects in computer network," *Applied mathematics and computation*, vol. 188, no. 2, pp. 1476–1482, 2007.

[81] A. Hagberg, P. Swart, and D. S Chult, "Exploring network structure, dynamics, and function using networkx," tech. rep., Los Alamos National Lab.(LANL), Los Alamos, NM (United States), 2008.

VITA

VITA

Nadra Guizani is a Clinical Assistant Professor at Washington State University's Voiland College of Engineering and Architecture. Her research interests include: machine learning, mobile networking, large data analysis, and prediction techniques. She is an active member of both the Women in Engineering program and the Computing Research Association.