Formal Verification of Internet Worm Propagation Model

By

Misbah Razzaq

NUST201260267MRCMS64012F

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE

REQUIREMENTS FOR THE DEGREE OF MASTER OF SCIENCE

in

Computational Science and Engineering

RESEARCH CENTER FOR MODELING & SIMULATION

(RCMS)

National University of Science and Technology (NUST), Pakistan

Dedication

To my Family,

I could not have done this without all of you

Thank you for your constant support along the way

Declaration

I hereby declare that this submission is my own work and to the best of my knowledge it contains no materials previously published or written by another person, nor material which to a substantial extent has been accepted for the award of any degree or diploma at NUST RCMS or at any other educational institute, except where due acknowledgement has been made in the thesis. Any contribution made to the research by others, with whom I have worked at NUST RCMS or elsewhere, is explicitly acknowledged in the thesis.

Misbah Razzaq

NUST201260267MRCMS64012F

Acknowledgements

I want to start by thanking Allah because he has placed many blessings in my life and has been there for me during the Good and the hard times. He has blessed me with parents who have always been so loving and supportive. I am so proud to be their daughter and I hope I can repay them by being as accomplished as they know I can be and I know I will be. My brothers, Adnan and Munsafeen, have always been there to support me morally and done their utmost to take care of me. I could not have asked for a better family, who have facilitated my desire to learn and succeed in life. May Allah bless them. I would also like to thank my husband, Ali Abbas, who has tried to support me as much as he can. May Allah bless us. I would like to thank my Professor, Dr. Jamil Ahmad, who has constantly been there to support me, advice me and I can honestly say is a great source of inspiration. He has played a significant part in helping me complete my thesis and without his constructive criticism, which I sincerely and highly appreciate, I would have struggled dearly. May Allah grant him continuous success. He is someone I look upto and idolise and I hope to be even half the professor he is in my eyes. I am also thankful to the respected thesis committee members Dr. Mukarram Khan, Mr. Muhammad Tariq Saeed and Mrs. Humaira Salman for providing me with all the support and guidance during my thesis work. They have helped me by giving suggestions to improve my research work.

Abstract

Internet worms are analogous to biological viruses since they can infect a host and have the ability to propagate through a chosen medium. To prevent the spread of a worm or to grasp how to regulate a prevailing worm, compartmental models are commonly used as a means to examine and understand the patterns and mechanisms of a worm spread. However, one of the greatest challenges is that we have failed to produce methods to verify and validate the behavioural properties of a compartmental model. This is why in this study we suggested a framework based on *Petri Nets* and *Model Checking* through which we can meticulously examine and validate these models. We investigated *Susceptible-Exposed-Infectious-Recovered (SEIR)* model and proposed a new model *Susceptible-Exposed-Infectious-Recovered-Delayed-Quarantined (Susceptible/Recovered) (SEIDQR(S/I))* along with hybrid quarantine strategy, which is then constructed and analysed using *Stochastic Petri Nets* and *Continuous Time Markov Chain*. The analysis shows that the hybrid quarantine strategy is extremely effective in reducing the risk of propagating the worm. Through *Model Checking* we gained insight into the functionality of compartmental models. *Model Checking* results validates simulation ones well, which fully support the proposed framework.

Acronyms

SPN	Stochastic Petri Net
CTMC	Continuous Time Markov Chain
SEIR	Susceptible-Exposed-Infectious-Recovered
SEIDQR(S/I)	Susceptible-Exposed-Infectious-Delayed-Recovered (Susceptible/Infectious)
CSL	Continuous Stochastic Logic
CTL	Computation Tree Logic
PCTL	Probabilistic Computation Tree Logic
SEM	Simple Epidemic Model
TFM	Two Factor Model
AAWP	Analytical Active Worm Propagation
KM	Kermack-Mckendrick
LAAWP	Local Analytical Active Worm Propagation

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Since the discovery of computers, security has been one of the major concerns. The first computer virus emerged over the ARPANET in 1970. The spread of viruses was not pandemic in that era because computers were not connected like today. Now a days, every computer is connected with each other via internet. While internet provides many advantages, it also proposes many security threats. The security of the internet or network is referred as network security. There is a slight difference between network security and computer security. Network security is like protecting all entry points of fortress while computer security is protecting individuals inside the fortress [1].

In recent years, worms are attacking internet frequently and have caused major trouble worldwide [2]. The purpose of attacker can be eavesdropping, modification of some data and getting some personal information. Malicious software is used to attack a computer system without the consent of its owner. This term is used to represent different types of hostile and intrusive softwares. Examples of these softwares include viruses, trojan horses, and worms etc. In this thesis, our main focus is on worms [1].

1.1 Internet Worms

A computer worm propagates in a network by copying itself from one host to another and this activity consists of three phases: locating hosts, exploiting loopholes in the system, and execution. Worms and viruses are often used interchangeably, however there is a small difference. Viruses require user intervention in order to infect the system while worms do not require any user intervention for infecting and propagating through the system [3].

Once host is compromised by a worm, it can be used for many despicable activates like launching denial-of-Service attack, sending spam email and downloading harmful programs etc.. Therefore, a mechanism must exist to ensure security of computers on the internet. Worm propagation speeds on a network are directly proportional to the increase in bandwidth. Automatic mitigation is the only solution to stop their propagation because manual countermeasures such as patching infected computers to immunize them, using antivirus software or firewalls, or disconnecting networks are very slow and users do not have the habit of keeping an antivirus software updated. Security patch for SQL Slammer had been released six months before the worm appearance. Despite that, it caused to infect tens of thousands of computers [1].

Intrusion detection techniques have been studied intensively in the past decade. Worm detection techniques are either anomaly-based or signature-based. Anomaly based techniques use statistical features of normal traffic to determine abnormal traffic. Signature based techniques are based on the pattern matching. They can only capture those attacks whose patterns have been stored in their databases. Every technique have its own pros and cons but signature based techniques are good for capturing known worms while

anomaly based techniques are good for identifying unknown behaviours. Hybrid Intrusion Detection System (HIDS) is a combination of signature based and anomaly based in one system. Hybrid Intrusion Detection Techniques are more powerful than other techniques because these can detect old as well as new attacks [4].

1.2 History of Internet Worms

Security threats imposed by internet worms have steadily increased, since the discovery of first internet worm in 1988 viz. Morris Worm. Morris was named after its creator Robert Tappan Morris. This worm caused a lot of damage on the internet and it was launched by MIT on 2nd November 1988. On 3rd November 1988, computers from all over the world were heavily loaded with some unknown processes which made them very slow. Every effort to recover the system was unsuccessful because processes were returning back on the system even after rebooting or cleaning the system. That day is known as "Black Thursday" in the history of the internet. This was the first time people were dealing with internet worms. Creator of this worm said that he created it only to measure the size of the internet [5].

In 1999, the Melissa worm appeared. It was first email worm. When a person opened infected file received through outlook address book, worm checked the contacts of address book and sent copy of itself to first 50 contacts. It spread quickly over the internet. After advent of Melissa worm, email worms have become common. ILOVEYOU and AnnaKornikova are examples of those email worms. Name of these two shows that subject of mail affect the spread of worm [6].

In March 2000, 911 worm spread through windows shares. It was slow au-

tonomous mechanism. It basically searched remotely for C drives and then copied its code into the victim's startup routines. And then when user started its computer next time, the worm became operational. It was slow spreading as compared to other worms because it required booting in order to complete the infection process [6].

Some other notable worms including Code Red and Nimda attacked hundreds of thousands of computers in 2001 [7, 8]. Blaster worm (2003) employed sequential scanning to find its targets [9]. SQL Slammer infected more than ninety percent vulnerable computers within 10 minutes in 2003 [10]. Witty worm was the first world wide spreading worm that damaged infected hosts [11]. Storm worm infected thousands of computers in 2007 [12]. Conficker was detected in November 2008 is the largest known worm since SQL Slammer [13, 14].

The safety and security of the internet has been compromised particularly by worms that exploit zero hour vulnerabilities. The sudden advancement of computer technologies and network applications has become a potential haven for malicious software programs. The propagation behaviour of worms in a system of interacting computers can somewhat be correlated with biological diseases [15, 16].

1.3 Compartmental Models

Epidemiology is the study of distribution and dispersion of the disease. Its primary objective is to trace out those factors which are responsible for the occurrence of disease. A large population is comprising of many individuals with different characteristics. That is why population is divided into different groups of individuals. These subgroups of the population are called com-

partments. Compartmental models are suitable to describe the movement of individual in a system especially for the disease transmission dynamics in an epidemic condition. It must be noted that any individual cannot be in more than one compartment at the same time. However an individual can move from one compartment to another with some transmission rate. For example, if individual transfer from susceptible to infectious compartment then this rate of transfer will be called infection rate. Whereas if individual transfers from infectious to recovered then that rate will be represented by recovery rate. The compartment of the model is referred with some alphabets which represent the structure of the model. For example susceptible, infectious, exposed and recovered etc. [17].

We can apply epidemiological models in Computer Science as well. Computer worms and biological viruses depicts the same behaviour when it comes to the transmission of disease. In computer science we refer to disease as worm and compartments are comprised of computers instead of humans [15].

In compartmental models, traffic is divided into different compartments.For Example, In SIR model population is divided into three different compartments.

- Susceptible: These hosts are vulnerable to worm.
- Infectious: These hosts are infected and can also infect others.
- Remove/Recovered: These hosts are dead/immune and cannot infect others.

Worm propagation or compartmental models are used to understand propagation behaviour in order to develop appropriate defence mechanisms against future attacks [18]. A variety of worm propagation models have been proposed to study the worms spread and effectiveness of number of defensive

strategies.

According to our knowledge, previous studies have not taken into account the effect of time delay in detecting worm and applying countermeasures on susceptible, exposed and infectious hosts. To cope with this delay, delayed state is introduced in the proposed model. To make our results more realistic, we have introduced a transition from recovered to susceptible and infected state because a network can never be worm free as there is always a chance of re-infection. Our proposed model is based on the hybrid intrusion detection, which combine features of both signature based and anomaly based techniques. According to the above description, we have presented a Susceptible-Exposed-Infectious-Recovered-Delayed-Quarantined (Susceptible/Recovered) "SEIDQR(S/I)" worm propagation model by modifying *Susceptible-Exposed-Infectious-Recovered (SEIR)*, to study the behaviour of worm's spread and to analyse the effectiveness of a quarantine strategy through *Model Checking*. This work presents the framework for studying the worm's propagation and validating the compartmental models. We have used *Petri Nets* and *Model Checking* to look closely at the behaviour of the model. *Stochastic Petri Net (SPN)* of the *SEIDQR(S/I)* is constructed and simulated in *Snoopy* Tool. *Continuous Time Markov Chain (CTMC)* of the model is generated in *PRISM* model checker to formally verify the behavioural properties of the model. The *Charlie* tool is used to analyse qualitative properties of the *SPN*. Through *SPN* we have created many experiments and have studied the impact of different parameters and classes on the system. According to our knowledge, framework based on *Petri Nets* and *Model Checking* have not been previously proposed for this purpose. Our work presents the first approach in that direction. This framework offers promising advantages in terms of qualitative and quantitative analysis of compartmental models.

1.4 Contributions

This research is an exploratory study on how worms propagate and how different parameters have effect on their propagation. Worm propagation models are used to understand propagation behaviour in order to develop appropriate defense mechanisms against future attacks. Current worm propagation modelling techniques are based on complex differential equations that rely on unreasonable assumptions and may require long periods of time before getting any results [19]. Using *Petri Nets*, worm propagation models can be developed with relative ease. We can also look closely at the behaviour of the model through *Model Checking* and this offers promising advantages in terms of qualitative and quantitative analyses.

Through *Petri Net* we have created many experiments and have studied the impact of the parameters on the system. Moreover, we have also checked how quarantine of worms affect the worm's speed. We have also performed *Model Checking* in order to validate our results. The main contributions of the current research work include:

- Development of Improved Worm Propagation Model
- Reconstruction of Proposed model using *Stochastic Petri Net*
- Qualitative Analysis of Proposed Model using *Charlie*
- Construction of Continuous Time Markov Chain (CTMC)
- Quantitative Analysis of Proposed Model using *PRISM*
- Quantitative Analysis of Hybrid Quarantine using PRISM

This thesis also provides correct analytical propagation model for worms. This will help network security professionals to clearly understand the worm's dispersion behaviour, containment techniques and other countermeasures.

1.5 Structure of the Thesis

The rest of the thesis has been structured in the following manner: Chapter 2 gives a comprehensive literature review covering the different worm propagation models. In Chapter 3, we covered the approaches used for simulation and analyses of worm propagation model. Chapter 4, gives an overview of the proposed *SEIDQR(S/I)* model and illustrates the *SPN* and *CTMC* of the proposed model. In Chapter 5, we show the simulations using *Snoopy* tool, quantitative analysis using *PRISM* model checker and qualitative analysis using *Charlie* tool. It also shows the analysis of the hybrid quarantine strategy through *PRISM* model checker. We discussed the results in Chapter 6. Conclusions are drawn on the basis of results in Chapter 7.

Chapter 2

Literature Review

This chapter gives an overview of the available related research on worm propagation modelling. Firstly, it explains the target discovery techniques for worms. Then study on worm spreading topology is carried out. Finally, worm propagation modelling approach is discussed

2.1 Worms Target Discovery Mechanisms

There are different target discovery techniques employed by worms. Some of these are discussed in this subsection.

2.1.1 Scanning

It is one of the simplest target discovery techniques and has been employed by very well-known worms like Slammer, Code Red, Code Red II, Sapphire, Blaster, and Witty worm. This technique is based on random selection or sequential selection of vulnerable host [20]. These techniques are explained below.

Random

In this technique, IP address of the target host is selected randomly. This technique assumes that network is fully connected and probability of infection of every edge is identical. Different strategies have been implemented based on random scanning such as hit list, uniform and routable scanning [20].

Uniform

Worm employing this technique does not have any information about target vulnerable host. It chooses IP address randomly from IPv4 address space. It just needs a good random number generator for selection of the target host. Some very well-known worms employed this technique are Code Red, Code Red II and Slammer [21, 22].

Hit List

It scans and infects all hosts on the hit list, then randomly select host in order to spread in the network. This technique can reduce the time of infection at early stage of worm propagation and was introduced by Staniford et al. [23]. Worms implementing these techniques spread quickly through the whole network. Flash worm is an example of such worms [23].

Routable

This technique is based on selection of routable IP addresses; therefore it does not consider the whole IP address space. A BGP routable worm as BGP routing table is proposed by Zou et al. [24]. Worms following this strategy are slow spreading because of the large payload associated as prefix.

Localized

This strategy is based on selection of nearby vulnerable hosts instead of random selection. It gives preference to the nodes which are closer targets. There are three different categories of localized scanning: 1) Local Preference 2) Local Preference Sequential 3) Selective. In local preference probability of infecting nearby nodes is higher than the far away nodes and this leads to the increase in propagation speed. Code Red II [25] and Blaster [26] are example of such worms. Local preference sequential method selects IP addresses in specific order. First, it selects the starting IP address which is the address selected by the worm and then it scans all IP addresses starting from that point. Worm employing this strategy repeats the propagation sequence thats why their worm propagation speed is very slow [27]. Blaster is an example of the worm employing that strategy [28]. Selective scanning targets particular IP address space. Worms employing this strategy have high propagation speed if target hosts are densely distributed in particular IP address space [27].

2.1.2 Topological

This technique follows a particular structure and uses the information contained in target host. For example a worm received through email first infects the receiver's computer then will send copies to all contacts. Melissa and isomorphic worms such as social network worms, p2p worms and Bluetooth worms are examples of such worms [6, 29, 30, 31, 32]. Worms following this technique have high propagation speed. This method also depends on human factor such as opening of particular email containing worm etc..

2.2 Topologies for Modelling the Worm Propagation

Network topology is an important factor for analysing the dynamics of worm propagation. There are many networks for studying worm propagation. In this subsection we are explaining different types of topologies applied for epidemic modelling in research.

2.2.1 Homogeneous Networks

Any infected node can infect any other node in homogeneous networks. Homogeneous networks are fully connected and every node has the same degree. Homogeneous models are useful for modelling the scan based worms because these do not depend on the topology of the underlying network. Code Red version I and II and slammer fall under the category of homogeneous networks. Differential equations have been used for modelling random scanning worms based on homogeneous networks [33, 34, 35, 36]. AAWP (Analytical Active Worm Propagation) model has been proposed for random scanning worms based on homogeneous network by Chen et al. [36]. Two-factor model has been proposed by Zou et al. [33] to analyse the Code Red worm propagation.

2.2.2 Random Networks

In these networks, link is selected randomly with equal probability. Infected node can infect the distant node quickly because path length is very short. Email worm propagation has been studied on random networks by Zou et al. [37].

2.3 Worm Propagation Models

Worm propagation models are used to understand propagation behaviour in order to develop appropriate defense mechanisms against future attacks. It can be noted that worm propagation models are biologically inspired, e.g. methods used in epidemic disease control. These methods used in epidemiology to represent the spread of infections using either deterministic models, which are suitable for larger systems, or stochastic models, which are suitable for smaller systems. Similarly, these models can be implemented in computer networks to understand worm propagation and their dynamics [18].

2.3.1 The Mathematical Approach

Many models have been developed over the years to represent internet worm propagation such as, Simple Epidemic Model (SEM), Susceptible-Infectious-Susceptible (SIS), Kermack-Mckendrick model (KM model), Two Factor Model (TFM), Analytical Active Worm Propagation (AAWP) model, Susceptible -Exposed-Infected-Recovered (SEIR) etc. [17, 33, 36, 37, 38]. These are either homogeneous models, local preference models or topological models. The distinction between different types of models is based on probabilities of interaction among nodes and structure of the network. Difference between SIS and KM model is that whether an infectious host can become susceptible again or not. In case it cannot then we use KM model and if it can then we use SIS model. Most of the previously proposed models [8, 26, 39] are based on the Kermack-Mckendrick model.

By the use of worm propagation models, Anderson and May have thoroughly explained the behavioural nature of biological diseases and parasites that can lead to the propagation of infectious diseases in human population [40]. By

applying the same method, via using the epidemiological models for disease propagation we can monitor and study the behaviour of worms throughout a network [41]. The SEIRS model presented by Mishra and Saina have latent and temporary periods that identify the propagation of a common worm [42]. Based on the SEIR model, Dong et al. proposed a computer virus propagation model and studied the dynamical behaviour including local asymptotical stability and local Hopf bifurcation of a computer virus model using time delay as a bifurcating parameter [43]. L.-X. Yang and X. Yang, examined the dynamics of the virus propagation, once infected systems are running on the network with positive chance [44]. Under human intervention Gan et al. examined the behaviour of computer virus propagation [45]. Ren et al. gave a new computer model for virus propagation and studied the dynamic behaviour of the model [46]. Quarantine is common and an effective way of containing the worms [10]. The use of quarantine strategy has produced some extraordinary results, successfully regulating diseases [10, 40, 41, 42, 43, 46, 44, 45, 47]. Wang et al. combined both a dynamic quarantine strategy and a vaccination in an epidemic model and referred this new model as SEIQV model [48]. Zou et al. proposed a new model with dynamic quarantine strategy based on two-factor model [10].
All these models are mathematical based on different state transition models and investigate the scan based and topology based worms. These models are discussed in detail in the following subsection.

Homogeneous Scan based Models

These models are based on the assumption that each host has an equal probability of infecting any other host in the network. Homogeneous networks are unstructured which means they do not follow any topology. Code Red

I, II and Slammer are example of scan based worms and they do not follow any topology constraints [8, 22, 49, 50]. Homogeneous models can be further categorized into two types: Discrete Time and Continuous Time.

Discrete Time Models

These models are described by the set of difference equations.

Analytical Active Worm Propagation

Analytical Active Worm Propagation (AAWP) model is a discrete time model and it characterizes the worm propagation based on random scanning strategy. It is based on the assumption that no host will be infected more than once and network is unstructured [26].

Local Analytical Active Worm Propagation

Local Analytical Active Worm Propagation (LAAWP) model is extended form of AAWP model. LAAWP is an example of discrete time model. It is used to analyze the worm propagation based on localized scanning technique. It works on the basis of giving different priority to different nodes. In LAAWP model nodes near infected nodes are highly susceptible to infection as compared to distant nodes [26].

Continuous Time Models

These models are described the set of differential equations.

Simple Epidemic

It is an SI model. In this model whole network is divided into two types of hosts: Susceptible and Infectious. It is also known as Classical Epidemic model. This model contains only one transition from susceptible to infections. Many researchers have used SE model to describe worm propagation for example Code Red [8] and Slammer [22]. Following differential equation describe SE model:

$$\frac{dI(t)}{dt} = \beta I(t)[N - I(t)] \tag{2.1}$$

$I(t)$ represents the number of infectious hosts at time t. β represents the infection rate. N represent the size of the population.

Kermack-McKendrick

Kermack-Mckendrick (KM) model categorizes each host in one of three states: Susceptible, Infectious and Recovered at any instant of time. KM model takes into account the recovery process of the hosts as well [51]. There are two transitions in this model. One is from susceptible to infectious state and other is from infectious to recovered state. Hosts will either stay in susceptible or recovered state forever. The following set of differential equation describe the KM model:

$$\frac{dS(t)}{dt} = -\beta S(t)I(t) \tag{2.2}$$

$$\frac{dI(t)}{dt} = \beta S(t)I(t) - \gamma I(t) \tag{2.3}$$

$$\frac{dR(t)}{dt} = \gamma I(t) \tag{2.4}$$

Here t is the time, $S(t)$ are the number of hosts at time t, $I(t)$ are the number of hosts at time t, $R(t)$ are the number of hosts at time t , β is the infection rate, γ is the removal rate The following equation should hold in the model:

$$N = S(t) + I(t) + R(t) \tag{2.5}$$

where N is the population size.

Susceptible-Infectious-Susceptible

Susceptible-Infectious-Susceptible (SIS) model states that host can go back to the susceptible state again after recovering from infection. The following

set of differential equation describes the SIS model:

$$\frac{dS(t)}{dt} = -\beta S(t)I(t) + \gamma I(t) \tag{2.6}$$

$$\frac{dI(t)}{dt} = \beta S(t)I(t) - \gamma I(t) \tag{2.7}$$

Here t is the time, $S(t)$ are the number of hosts at time t, $I(t)$ are the number of hosts at time t,β is the recovery rate,γ is the removal rate.
The following equation should hold in the model:

$$N = S(t) + I(t) \tag{2.8}$$

where N is the population size.

Two-Factor

Two-Factor (TF) model takes into account the removal of susceptible hosts as well while KM model just consider the removing of infectious hosts. TF model also assumes that infection rate is not constant. Zou et al. [33] introduced the TF model which is basically the extension of KM model with variable infection rate and human countermeasures. TF model can be represented by the following set of differential equations:

$$\frac{dI(t)}{dt} = \beta(t)I(t)[N - I(t) - R(t) - Q(t)] - (\frac{R(t)}{d(t)}) \tag{2.9}$$

$$\frac{dR(t)}{dt} = \gamma I(t) \tag{2.10}$$

Where t is the time, β is the variable infection rate, $I(t)$ are the number of infectious hosts at time t, $R(t)$ number of removed hosts from infectious hosts at time t, $Q(t)$ number of removed hosts from susceptible state at time t and N is the population size.

Susceptible-Exposed-Infectious-Recovered

Susceptible-Exposed-Infectious-Recovered (SEIR) model is one of the various extensions of KM model [15, 52]. In SEIR model an exposed state is introduced between susceptible and infectious class. SEIR model is more realistic than KM model. Hosts experience incubation period when transition from susceptible to exposed class occurs. Hosts in exposed class are infected but not infectious yet. After staying in exposed class for some period of time, hosts move from exposed to infectious class in order to infect other hosts. Conficker worm spread through the network with incubation period of 3.5 h which makes the SEIR model more suitable for studying the worm propagation [53].

Chapter 3

Methodology

In this chapter, we review methods and tools which we are going to use for simulation and performance analysis purpose of our model. This chapter sheds light on the theory behind *Model Checking* and *Petri Nets*. The objective is to help the readers understand the basics of *Model Checking* and *Petri Net* theory and motivate the use of this theory for modelling the worm propagation models.

3.1 Methodology Overview

In the light of literature review, we have selected the SEIR epidemiological model and reconstructed it by introducing delayed state and re-infection probability. The new model was named as *SEIDQR(S/I)*. We developed *Stochastic Petri Net (SPN)* of the proposed mode and simulated using *snoopy* tool. Then we have generated the *Continuous Time Markov Chain (CTMC)* for property verifications. The methodology followed in this study is presented in Figure 3.1 and explained below.

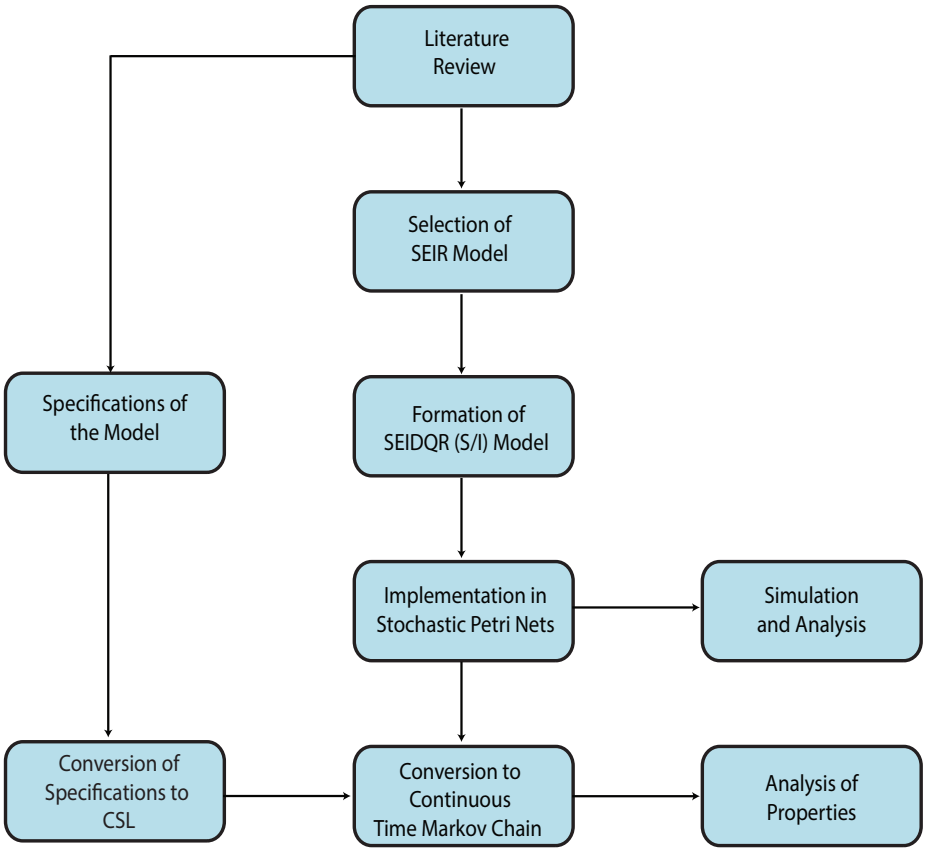

Figure 3.1: Workflow of the Proposed Framework
After reviewing literature, *SEIR* model is selected and a new *SEIDQR(S/I)* is proposed by modifying SEIR model. *SPN* of the proposed model is constructed and analysed in *Snoopy* and *Charlie*, after which the system is converted to *CTMC* and specifications are encoded in *CSL* for quantitative analysis in *PRISM* model checker.

The following sections explains the Petri Net and Model Checking approach.

3.2 Petri Net

Petri Net is a powerful graphical and mathematical modelling tool. A set of linear algebraic equations can be described by the help of *Petri Net* model. We can also represent many mathematical models depicting behaviour of the system using *Petri Net* tools. This helps in formal analysis of the model to check the behavioural properties of the system. *Petri Nets* are also suitable for depicting natural logical interactions among various parts of the system. This theory originated from Carl Adam's Publications in 1962 [54, 19].

Petri Nets have been acknowledged as a powerful formal verification tool in many theoretical as well as applicative fields. It is used for formal specification of several type of systems like deterministic, parallel, concurrent, distributed, non-deterministic and asynchronous. It has applications in many fields like engineering, business, mathematics, chemistry and manufacturing etc.. Since its acceptance as formal verification framework in 1962, there has been many improvements on the initial design of *Petri Nets*. These improvements adds some additional functionality to the basic *Petri Net* structure [54, 19].

A *Petri Net* is a bipartite directed graph consisting of three types of objects: places, transitions and arcs. One partite consist of places and other consist of transitions. Directed arc are used to connect places with transitions and transitions with places. Graphically, Places are represented by circles and transitions are represented by boxes. A place is called an input place of the transition if there is an arc from that place to a transition. A place is called an output place of the transition if there is an arc from transition to that place. Places are marked with token. These token defines the intitial marking of the *Petri Net* [54, 19].

3.2.1 Petri Net Semantics

The *Petri Net* semantics derive the behaviour of the *Petri Net* and is described by the firing rules, which consist of preconditions and the firing itself. The firing of any transition depends on the tokens. If every place in the input set of a transition contains a number of tokens greater than or equal to the place-transition arc weight then the transition is said to be fireable. Figure 3.2 example explains the firing of a transition. If transition has empty input set of places then transition is always enabled and is said to be a source transition. Figure 3.3 gives the example of a source transition. Firing of a transition consist of removing token from every input place according to the arc weight and adding tokens to every output place of a transition according to the output arc weight. After firing, a new marking of a *Petri Net* is obtained. All these repeated firing sequence of transitions depicts the behaviour of the whole *Petri Net*. A transition without any output place is called sink transition. This transition consumes all tokens and it does not produce any token [54]. Figure 3.4 explains the sink transition.

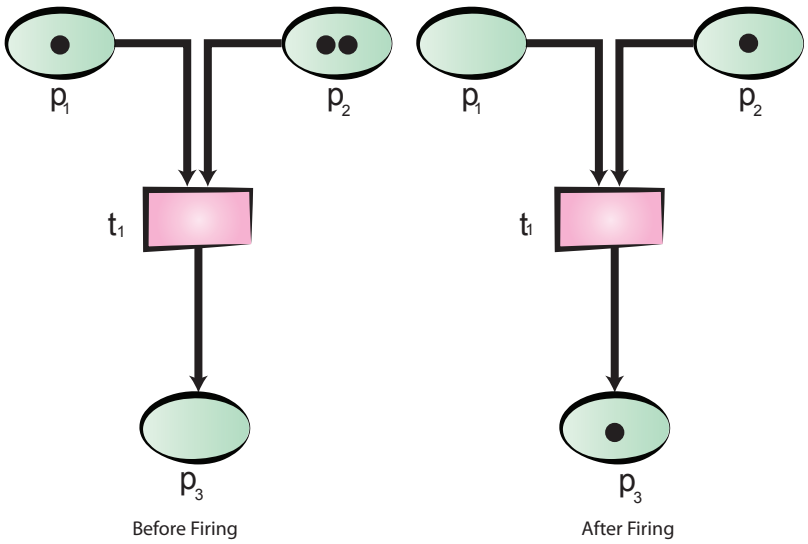

Figure 3.2: Firing Rule
Figure 3.2 shows the firing rule of a transition. When transition t_1 is fired then a token from places p_1 and p_2 is removed and is placed in place p_3.

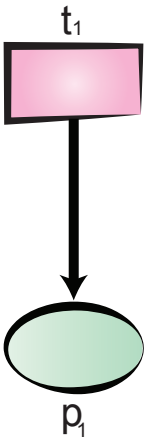

Figure 3.3: Source Transition
Figure 3.3 shows that source transition t_1 with one output place p_1

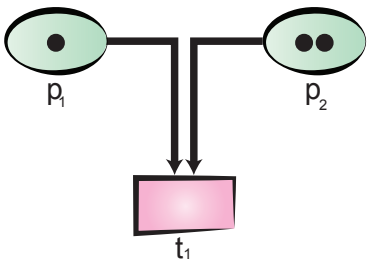

Figure 3.4: Sink Transition
Figure 3.4 shows the sink transition t_1 with two input places p_1 and p_2.

The dynamics of the *Petri Net* are described by the means of a reachability graph. Reachability graph shows the markings reached by the firing of transitions. In other words, reachability graph is basically a state space diagram of the *Petri Net* behaviours. Any marking is reachable from initial marking if there exists a firing sequence from the initial marking of the *Petri Net*. Reachability set of a *Petri Net* consist of all markings which are reachable from the initial marking of a *Petri Net* [54].

Definition of Standard Petri Net

"A *Standard Petri Net* [54] is defined by $5-tuple$ $\langle P, T, F, W, M_0 \rangle$ where;

- P is a finite set of places $\{p_1, p_2, ..., p_m\}$
- T is a finite set of transitions $\{t_1, t_2, ..., t_m\}$
- $F \subseteq (P \times T) \cup (T \times P)$ is a set of arcs
- W is a weight function of arcs
- M_0: is initial marking $P \rightarrow \{0, 1, 2, ...\}$ "

where $P \cap T = \emptyset$ and $P \cup T \neq \emptyset$. The example in Figure 3.5 illustrates the definition of the Standard Petri Net.

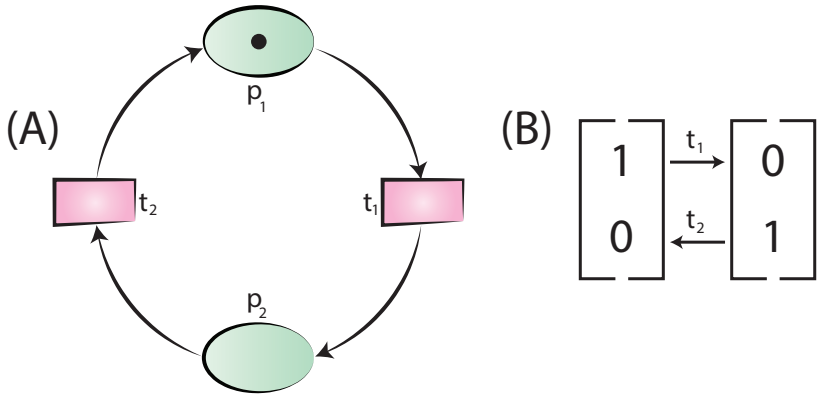

Figure 3.5: Example of a *Standard Petri Net.*
(A) A *Petri Net* consist of a set of places $\{p_1, p_2\}$, set of transitions $\{t_1, t_2\}$ and an initial marking M_0 consisting of one token in place p_1. Initial marking M_0 has an impact on structural properties of the *Petri Net*. In this example, the weight of the arcs is not specified so every arc weighs 1. The enabling degree of a transition is determined by number of times a transition can be fired without depositing a token again to the input place of a transition through self-loop. In case of above example t_1 is 1 enabled and t_2 is 0 enabled from the initial marking M_0.(B) The reachability graph obtained from initial marking M_0 of the *Petri Net*. A reachability graph consist of set of places which can be reached from M_0 and arcs which are labelled with enabled transitions. This graph shows one cycle: $(1,0) \rightarrow (0,1) \rightarrow (1,0)$ and contains no deadlock. To reach marking $M_1 = (0,1)$ from marking $M_0 = (1,0)$, a firing sequence S consist of a transition t_1 once and transition t_2 zero times.

3.2.2 Timed and Stochastic Petri Nets

Petri Net has evolved over time with increase in their functionality. It helps in modelling and analysing more complex systems. In this section, we will explain the Timed and Stochastic *Petri Nets*.

Timed Petri Nets

In a timed *Petri Net* time delays are assigned to transitions. Once a transition is enabled then it is fired after the time delay associated with it [19]. Since in most cases these delays are not deterministic therefore considering them as random variable with more than one outcome will be a better choice.

Stochastic Petri Nets

"A *Stochastic Petri Net (SPN)* [19] is defined by $6-tuple$ $\langle P, T, F, W, M_0, \Omega \rangle$ where P, T, F, W, M_0 are same as described in the definition of the *Standard Petri Net* and Ω represents the function $\Omega : T \rightarrow R$ which assigns rate to a transition $t \in T$ according to the negative exponential distribution function which is a probability distribution that describes the time between events in a stochastic process i.e. a process in which events occur continuously and independently at a constant average rate. It has the property of being memoryless ". It is defined as:

$$\Pr\ [\mathrm{d}_1 \leq \mathrm{t+dt} \mid \mathrm{d}_1 > \mathrm{t}] = \mu.\mathrm{dt}$$

The probability density function (pdf) 'h' and cumulative distribution function (cdf) 'H' of the exponential law [55, ch. 3, p. 110] are given as,

$$\mathrm{h(t)} = \mu.\mathrm{e}^{-\mu t}$$
$$\mathrm{H(t)} = \Pr\ [\mathrm{d}_1 \leq \mathrm{t}] = 1 - \mathrm{e}^{-\mu t}$$

The evolution of Stochastic *Petri Net* is described by a *Continuous Time Markov Chain (CTMC)* where a state represents a marking of the *Petri Net* [19]. This is basically the most important feature of *Stochastic Petri Net* because by analysing *CTMC* we can obtain the probabilities of the states. If priorities are also assigned to the transition in *Stochastic Petri Net* then

it is called *Generalized Stochastic Petri Net*. In Generalized *Stochastic Petri Net*, some transitions are immediate transitions and these transitions are fired with zero time delay [19]. The example in Figure 3.6 illustrates the definition of the *Stochastic Petri Net*.

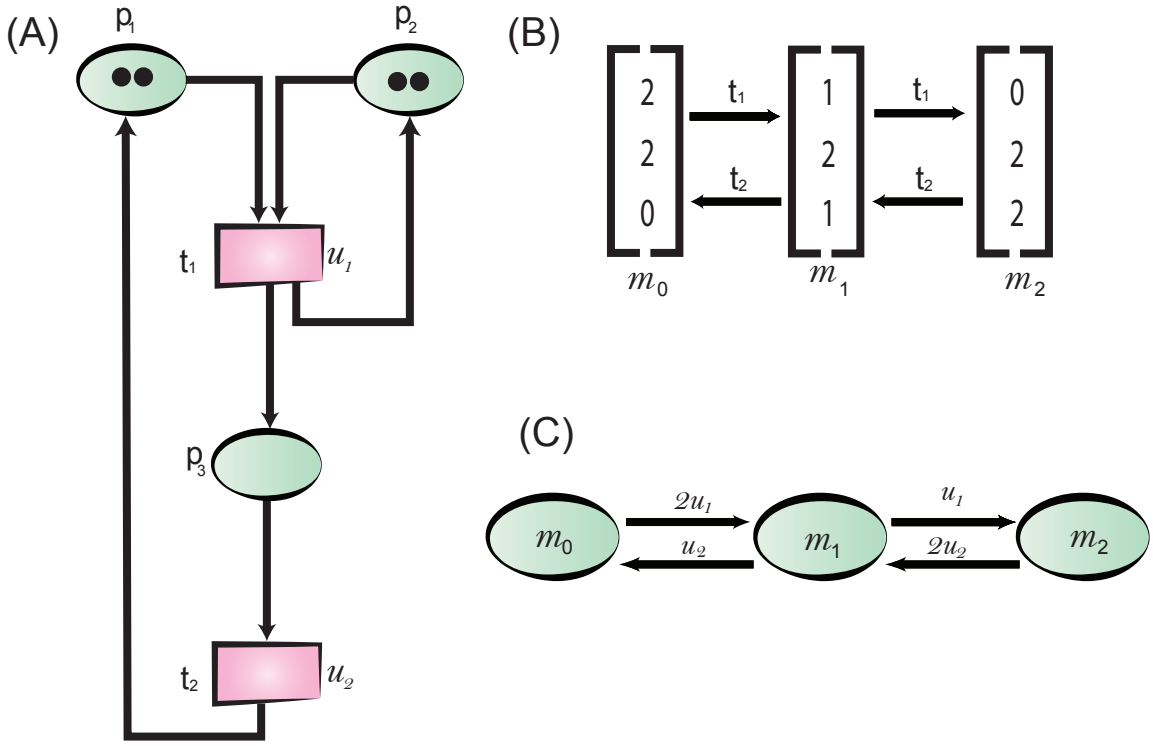

Figure 3.6: Example of a *Stochastic Petri Net*.
(A) The *SPN* consist of a set of places $\{p_1, p_2, p_3\}$, set of transitions $\{t_1, t_2\}$, rates μ_1, μ_2 and an initial marking $M_0 = (2, 2, 0)$. In this example t_1 is 2 enabled and t_2 is 0 enabled from the initial marking M_0.(B) The reachability graph obtained from initial marking M_0 of the *Petri Net*. (C) The *Markov Chain* obtained from the reachability graph in (B). Every reachable marking of the *SPN* is associated with a state of the *Markov Chain* and a transition between states is labelled with the product of the enabling degree and rate.

There are several behavioural properties of *Petri Nets* [54, 56, 55] and some of these are described below:

- Reachability: This property is used to study dynamic properties of the system. A marking M_k is reachable from initial marking M_0 if there exists a firing sequence from M_0 to M_k.
- Liveness: A live *Petri Net* is a deadlock free *Petri Net* and from any marking, there exists a firing sequence which contain all transitions.
- Reversibility: This property ensures that there will always be a way back to the initial marking M_0 from all reachable markings commencing from M_0.

3.2.3 Qualitative Properties of Petri Net

Petri Net framework allows to check various properties of the system. These properties are divided into two types:

1. Structural
2. Behavioural

Following section explains the structural and behavioural properties which can be analysed through Petri net.

Structural Properties

These properties depends on the overall structure of the *Petri Net*. These are independent of the initial marking of the *Petri Net*. Following are the some structural properties of the *Petri Net* [57]:

- Pure: No self-loops in a *Petri Net*

- Ordinary: Every arc weights 1
- Connected: There is no disconnected part in a net
- Strongly Connected: There must exist a directed path from node a to node b and node b to node a as well

Behavioural Properties

These properties characterize the behaviour of the system and are dependent on the initial marking of the system. Following are some behavioural properties of the system [57]:

- Boundedness
- Liveness
- Reversibility
- Invariance

3.3 Model Checking

Testing or simulation-based system analysis techniques are not as effective as an automatic model-based verification approach [58]. *Model Checking* in particular, is a more powerful tool capable of exploring a whole state space. [59]. A model checker is used to examine whether the model of a system which is specified in some modelling formalism such as *Petri Nets*, meets the requirements of a system which are usually encoded in some temporal logic such as CTL (Computation Tree Logic) or PCTL (Probabilistic Computation Tree Logic) [60]. In order verify largest system, one has to deal with state explosion problem. That is why, it is not best for verifying infinite state

space systems. The size of the system state space affect the performance of *Model Checking* process. In recent years, enormous research work had been done focusing this problem. In the following section, we will explain the *Model Checking* process, the *Probabilistic Model Checking* and *Continuous Stochastic Logic*.

3.3.1 The Model Checking Process

The *Model Checking* process consist of three steps. First of all, design of the system is converted into a formalism which is accepted by *Model Checking* tool. After that, system requirements are converted into logic specifications. And then, automatic verification of the system is conducted. Figure 3.7 explains the *Model Checking* process.

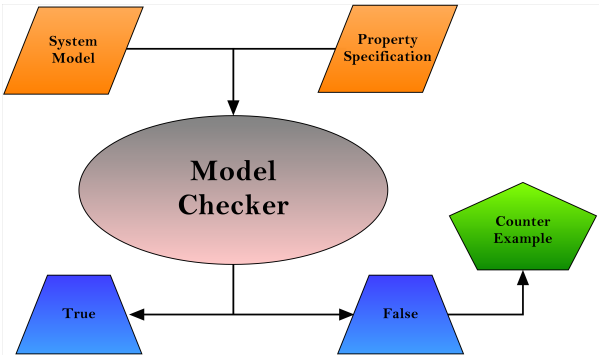

Figure 3.7: *Model Checking* Process
Model checker takes the system model and property specification as input and generates two types of output: (1) true which means property is satisfied (2) false which means property is not satisfied, generating a counter example.

These steps are discussed in detail in the following sections.

System Modelling

In modelling of the underlying system, we design the formalism accepted by a model checker. Formal modelling is a critical step because it owes many

limitations over time and memory. We may have to build the abstraction of the system in order to overcome these limitations. It is not a very simple step to build a model of the system, because we have to include all necessary information and we also have to eliminate all unnecessary details. For example, in case of communication protocols, we may ignore the contents of messages but we focus on the exchange of messages. A state is basically an instantaneous description of the system. The state of the system changes in result of some actions. These action are described by transitions [59].

System Specification

System requirements are specified as properties that the designed model must satisfy. There are different ways of expressing properties. Generally, properties are specified using temporal logic. We can verify many properties [59] like:

- Safety: This property ensures that something bad will never happen.
- Liveliness: This property ensures that something good will eventually happen.

System Verification

System model is specified in model checker. Then state space is being generated for analysis purpose. After that various system properties are checked against that model. If a particular property is not satisfied then a counter example is generated [59].

3.3.2 Probabilistic Model Checking

Probabilistic Model Checking is a variant of a formal verification technique and is used for analysing and modelling a system which shows stochastic behaviours. In *Probabilistic Model Checking*, probabilistic model of the system is checked against various probabilistic properties, to decide whether a model satisfy a particular property or not. There are two types of outputs of a probabilistic model checker: (1) 'True' signifying that the property is satisfied or 'False' signifying the property is dissatisfied. (2) 'Numerical Number' indicating the probability or expected time [61]. The *Model Checking* approach used in this study is probabilistic which is based on *CSL*. *PRISM* model checker is used for this purpose because *PRISM* supports *CTMCs* with *CSL* [61]. There are many probabilistic model checkers and have been used by many users for different purposes. Our work concenters on the PRISM model checker. PRISM supports *CTMCs* for the logic *CSL* and PCTL.

3.3.3 Continuous Stochastic Logic

The temporal logic used in *CTMCs* is *CSL* which is based on both CTL and PCTL [62, 63]. It gives powerful means to specify path based, reward based as well as traditional properties [61]. The syntax of *CSL* is as follows:

$$\phi ::= true \mid a \mid \neg\phi \mid \phi_1 \wedge \phi_2 \mid (P_{\sim p}[\phi_1 \cup^I \phi_2] \mid S_{\sim p}[\phi] \mid R_{\sim r}[F\phi]$$

Where a represents the atomic proposition,$\sim \in \{>,\geq,<,\leq\}$, p=$[0, 1]$, I is an interval of $\mathbb{R}_{\geq 0}$ and r, $t \in \mathbb{R}_{\geq 0}$.

CSL formulae are evaluated over the *CTMC* states. A formula $s \models \phi$ indicates that ϕ is true in state s of *CTMC* model. *CSL* contains all standard operators from propositional logic: true (all states satisfy); atomic proposi-

tions (true in all states labelled with a); negation ($\neg\phi$ is true in all states in which ϕ does not hold) and conjunction ($\phi_1 \wedge \phi_2$) will be true in a state in which both ϕ_1 and ϕ_2 hold). We can drive other standard operators like disjunction and implication from these standard operators.

Furthermore, *CSL* includes probabilistic operators as well like P and S. Both probabilistic operators include probability bound $\sim p$. We write $P_{\sim p}[\Psi]$ to indicate the *CSL* formula Ψ is true in a state s if the probability of *CSL* path formula meets the bound $\sim p$. The S operator is used to verify the steady state behaviour of the *CTMC* in long run.

CSL also has another operator R to calculate those properties which require expected value of reward. $R_{\sim r}[F\phi]$ is used to calculate the accumulated value of the expected reward before a state is reached where ϕ is satisfied. [61, 64].

Semantics of CSL over CTMC

Let $C{=}(S, R, L)$ represents the labelled *CTMC*. For any state s $\in$ S the formula s $\models \phi$ is defined inductively by:

$$
\begin{aligned}
s &\models true \quad \forall \quad s \in S \\
s &\models a \iff a \in L(s) \\
s &\models \neg\phi \iff s \not\models \phi \\
s &\models \phi_1 \wedge \phi_2 \iff s \models \phi_1 \wedge s \models \phi_2 \\
s &\models P_{\sim p}[\psi] \iff prob(s, \psi) \sim p \\
s &\models S_{\sim p}[\phi] \iff \sum_{s' \models \phi} \pi_s(s') \sim p
\end{aligned}
$$

where $Prob(s, \psi) = Pr\{w \in Path(s) \mid w \models \psi\}$. In *CTMC*, probabilistic operators P, S, R can be used to generate the quantitative result by

replacing the bound with =? [61, 64].

3.4 Softwares

As mentioned earlier, *Snoopy* tool is used for *Petri Net* modelling and simulation of the system. *PRISM* model checker is used for analysis of the quantitative properties and quarantine strategy. *Charlie* tool is used for analysis of the qualitative properties of the system.

3.4.1 Snoopy

Snoopy is a tool used to design, animate and simulate *Petri Nets*. *Snoopy* is comprised of many frameworks like stochastic, hybrid and continuous *Petri Nets*. It also allows construction of high level *Petri Nets* like colored *Petri Nets* [57]. The *SPN* of the proposed model is constructed in *Snoopy* tool [57].

3.4.2 PRISM

PRISM is a probabilistic model checker and is used to automatically verify probabilistic systems using continuous stochastic logic. It provides support for three types of probabilistic models: (1) DTMC (2) MDP and (3) *CTMC*. *PRISM* accepts the probabilistic model written in the *PRISM* modelling language, builds the corresponding probabilistic model against it and then uses *CSL* for the verification of the behavioural properties of the model. This research work focuses on the *CTMC* developement of the proposed model in *PRISM* model checker and verification of the properties encoded in *CSL*. [65, 66].

3.4.3 Charlie

Charlie is a software tool that performs analysis of *Petri Nets*. It is used to verify a list of properties of the *Petri Nets* such as reachability, liveness and boundedness etc. This tool is used for the verification as well as validation of the systems in different domains. *Charlie* is used to perform structural, invariants based, reachability graph based and behavioural analysis of the *Petri Nets*. In this paper, we have used *Charlie* to perform qualitative analysis of different properties of the proposed system [67].

Chapter 4

Model Formulation

This chapter explains the new proposed sytem, SPN of the proposed system and finally, *CTMC* of the proposed system.

4.1 SEIDQR(S/I)

Taking into account the worms having exploited zero day vulnerabilities, the host could not be immunized by the usually effective and reliable safety patches. Susceptible hosts initially face an incubation period (exposed) following infection prior to becoming infectious, for this reason people may try placing countermeasures as a precautionary attempt to immunize the exposed and infectious host. However, this can be time consuming and potentially aggravate the harm caused by worms if they are unknown. Due to the existence of the delays, the infectious hosts are put through a temporary state (delayed) before quarantining and recovery. We considered introducing delayed stage prior to the quarantined stage in order to compensate for the delays experienced by the hosts.

It is important to note that there is no permanent immunity in the real net-

work. The only immunity a node achieves in a real network is temporary, for this reason we must observe the real phenomena of possible re-infection. In order to address this issue, a node reverts to the susceptible and infectious compartment again in the proposed model.

Quarantine strategy is dependent on two types of intrusion detection systems, which can be categorized as misuse and anomaly intrusion detection. They both have their pros and cons but misuse intrusion detection systems recognizes the attack behaviour of potential threats with its broad database constructed of known attack behavioural patterns. This is beneficial to an extent but with regards to unknown worms variants, it fails to recognize them as a threat by not having the data of their behavioural patterns. On the contrary, anomaly detection systems are able to recognize abnormal behavioural patterns which help in the detection of unknown worms and their variants. The quarantine strategy proposed in this study is based on both, anomaly and misuse intrusion detection system. Although, anomaly detection technique can detect unknown worms, it is accompanied by false-positive rates and because of that two transitions have been added with rates n_1 and n_2 from susceptible and exposed classes.

According to the above description, we have proposed a *SEIDQR(S/I))* model with hybrid quarantine strategy. Presumably, the host will be in one of the following states: Susceptible state (S_{US}), Exposed state (E_{XP}), Infectious state (I_{NF}), Delayed state (D_{EL}), Quarantined state (Q_{UA}) and Recovered State (R_{EC}) with initial condition $N = S_{US_0} + E_{XP_0} + I_{NF_0} + D_{EL_0} + Q_{UA_0} + R_{EC_0}$. Here, $S_{US}(t)$ are the number of susceptible nodes at time t, $E_{XP}(t)$ are the number of exposed nodes at time t, $I_{NF}(t)$ are the number of infectious nodes at time t, $D_{EL}(t)$ are the number of delayed nodes at time t, $Q_{UA}(t)$ are the number of quarantined nodes at time t, and $R_{EC}(t)$

are the number of recovered nodes at time t. β denotes the infection rate. α is the rate at which exposed hosts become infectious. There are many parameters which will be used throughout the study and they are listed in Table 4.1. Figure 4.1 represents the proposed *SEIDQR(S/I)* model.

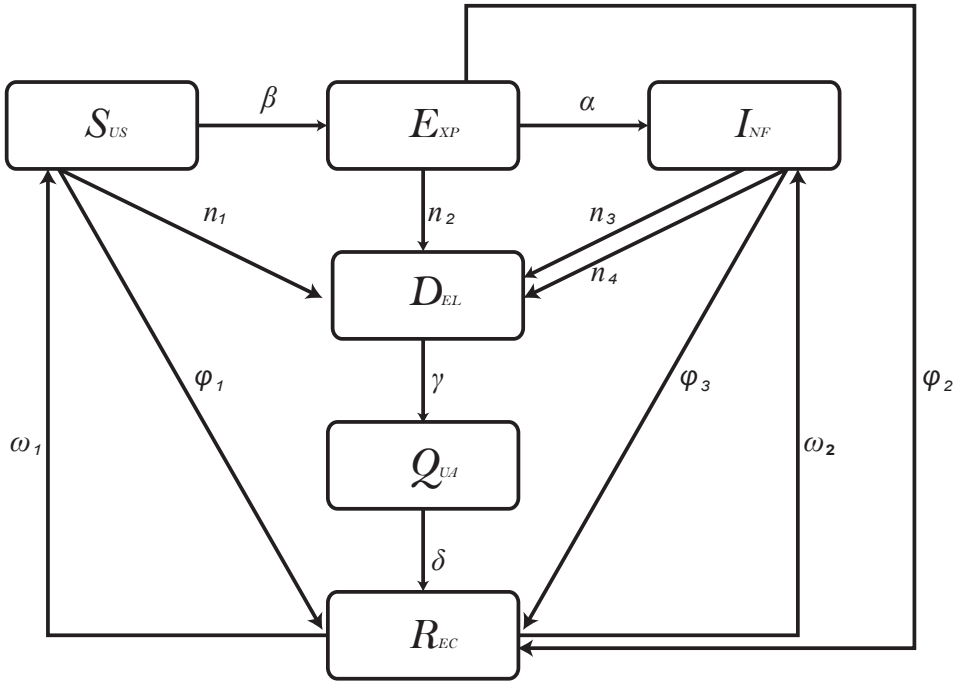

Figure 4.1: The states and the transitions of *SEIDQR(S/I)* model The rectangles represent the compartments and the arrows represent the movement of hosts from one compartment to another. The labels on the rectangles indicate the type of compartment i.e. susceptible, exposed, infectious, delayed, quarantined and recovered. The labels on the arrows indicate the rate of transmission of hosts from one compartment to another.

Notation	Explanation
N	Total size of population
$S_{US}(t)$	Number of susceptible hosts at time t
$E_{XP}(t)$	Number of exposed hosts at time t
$I_{NF}(t)$	Number of infectious at time t
$D_{EL}(t)$	Number of delayed hosts at time t
$Q_{UA}(t)$	Number of quarantined hosts at time t
$R_{EC}(t)$	Number of recovered hosts at time t
β	The rate at which susceptible hosts become exposed
α	The rate at which exposed hosts become infectious
γ	Quarantined rate of delayed hosts
δ	Recovery rate of quarantined hosts
n_1	Delayed rate of susceptible hosts
n_2	Delayed rate of Exposed hosts
n_3	Delayed rate of Infectious hosts with anomaly detection
n_4	Delayed rate of infectious hosts with signature based detection
φ_1	The rate at which susceptible hosts become recovered
φ_2	The rate at which exposed hosts become recovered
φ_3	The rate at which infectious hosts become recovered
ω_1	The rate at which recovered hosts become susceptible
ω_2	The rate at which recovered hosts become infectious

Table 4.1: Notations and Explanation

The population size of the model is equal to constant N at any instant of time and the population is closed.

$S_{US}(t) + E_{XP}(t) + I_{NF}(t) + D_{EL}(t) + Q_{UA}(t) + R_{EC}(t) = N$

4.2 Petri Net Model

We have presented an approach through which we can model network epidemiological systems through *Petri Nets* with relative ease. *Petri Net* modelling of the real system is sometimes called Condition-Event net. *Petri Net* places are used to identify conditions of the system and transitions represent the flow from one condition to another. An event can only occur if all the

conditions are satisfied i.e., input places are marked with sufficient tokens. We are using *SPN* to model our proposed system because we can easily generate *CTMC* though *SPN* as *SPNs* are isomorphic to *CTMC* [19].To model an *SEIDQR(S/I)* as a *SPN*, we need to represent the host population which consist of different compartments. For this purpose, places are used to represent the states or compartments of the system i.e. susceptible, exposed, infectious, delayed, quarantined and recovered. Hosts are represented by the tokens and dynamic part of *SEIDQR(S/I)* is modelled by transitions labelled as $t_1,\ t_2,\ t_3,\ t_4,\ t_5,\ t_6,\ t_7,\ t_8,\ t_9,\ t_{10},\ t_{11},\ t_{12},\ t_{13}$ and these transitions originate the flow of hosts from one state to another with specified rate. Figure 4.2 illustrates the *SPN* of the proposed model.

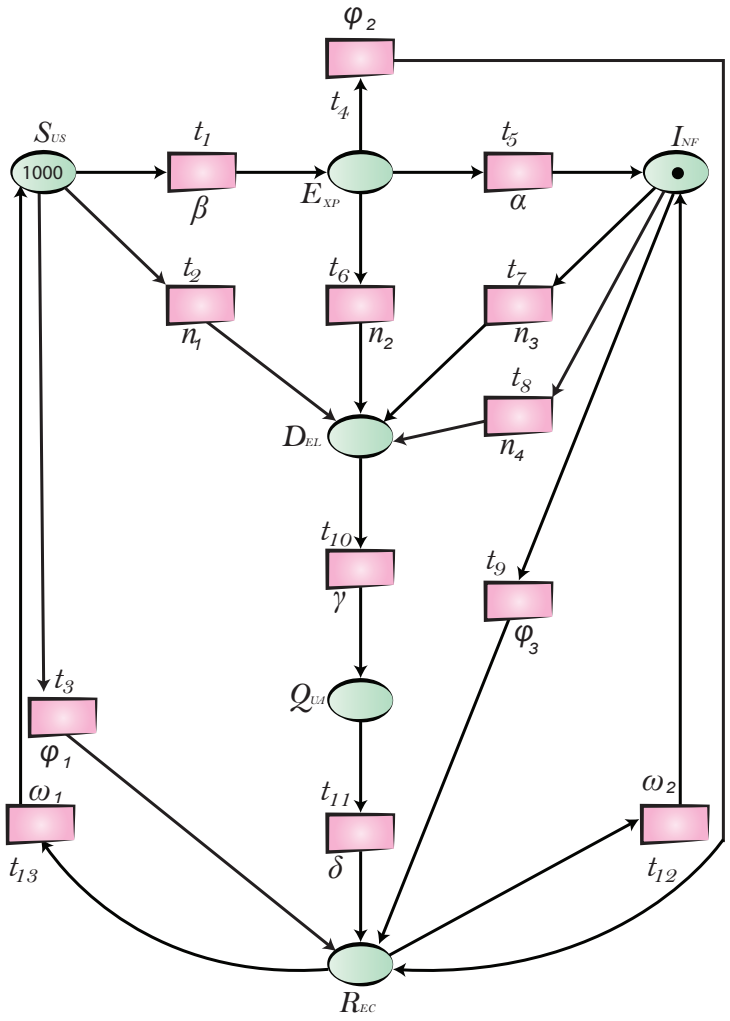

Figure 4.2: The *SPN* of the Proposed model
The *SPN* of the proposed model consists of a set of places $P = \{S_{US},\ E_{XP},\ I_{NF},\ D_{EL},\ Q_{UA},\ R_{EC}\}$ and set of transitions $T = \{t_1,\ t_2,\ t_3,\ t_4,\ t_5,\ t_6,\ t_7,\ t_8,\ t_9,\ t_{10},\ t_{11},\ t_{12},\ t_{13}\}$ and initial marking $M_0 = (1000, 0, 1, 0, 0, 0)$.

4.3 PRISM Model

We have chosen a *Probabilistic Model Checking* approach because it provides both probabilistic analysis and conventional reachability. Through *Probabilistic Model Checking*, we will be able to accumulate accurate answers as compared to approximate solutions obtained through simulation. State space explosion is the only drawback of *Probabilistic Model Checking* [58].

We initiated the model definition, writing keyword "ctmc" in the beginning, to explicitly mention the type of probabilistic model used in this study. Then we mentioned the sequence of possible values upheld by state variables S_{US}, E_{XP}, I_{NF}, D_{EL}, Q_{UA} and R_{EC}. In our case, we choose to let them vary between 0 and upper-limit labelled as Max. It is not necessary to specify upper-limits using parameters, but it helps to maintain the clarity of the model. Then, we labelled the module enclosing the transition rules as "SEIDQRSI". After mentioning state variable names and defining their range within square brackets, keyword "init" is used to define the initial value of the variable. Then, we gave the definition of transition rules β, α, γ, δ, n_1, n_2, n_3, n_4, φ_1, φ_2, φ_3, ω_1, ω_2. The list of transition rules are separated from the list of conditions using a symbol $->$. Then we end the module by writing the keyword "endmodule". We specified a reward structure by writing a reward rule between keywords "rewards" and "endrewards".

Then, we encoded behavioural properties in *CSL* in order to verify against the proposed model. Some of these behavioural properties are listed below:

- Expected number of hosts at any time instant t.
- Probability of reaching the maximum number of infectious hosts

- Invariance principle

CTMC of the Proposed Model is given in the Appendix A.

Chapter 5

Results

In this chapter, first we present the simulation results obtained through the *Snoopy* tool and then the verification results obtained through *PRISM* and *Charlie* tools.

5.1 Simulation Using Snoopy

We present the simulation results using *Snoopy* tool. Initial values of the network are: $N = 1001,\ S_{US}(0) = 1000,\ E_{XP}(0) = 0,\ I_{NF}(0) = 1,\ D_{EL}(0) = 0,\ Q_{UA}(0) = 0,\ R_{EC}(0) = 0,\ \beta = 0.7,\ \alpha = 0.66,\ \gamma = 0.9,\ \delta = 0.8,\ n_1 = 0.01,\ n_2 = 0.01,\ n_3 = 0.01,\ n_4 = 0.02,\ \varphi_1 = 0.001,\ \varphi_2 = 0.005,\ \varphi_3 = 0.004,\ \omega_1 = 0.0001$ and $\omega_2 = 0.0001$.

Figure 5.1 shows the behaviour of all states of the proposed system with respect to time. The results predict that the proposed system is asymptotically stable. Figure 5.1 shows that recovered class has a powerful impact on all other classes of the network. Initially, it can be seen clearly that infection is of a lesser degree but with time it increases gradually. We observe over time that $R_{EC}(t)$ increases whereas $I_{NF}(t)$ decreases. We also notice that we still

have some infected nodes at 100th time unit, which proves our assumption that real network can never be completely free from infection. It also shows the important role of the quarantine strategy. The quarantined computers are kept under observation and treated with anti-virus software.

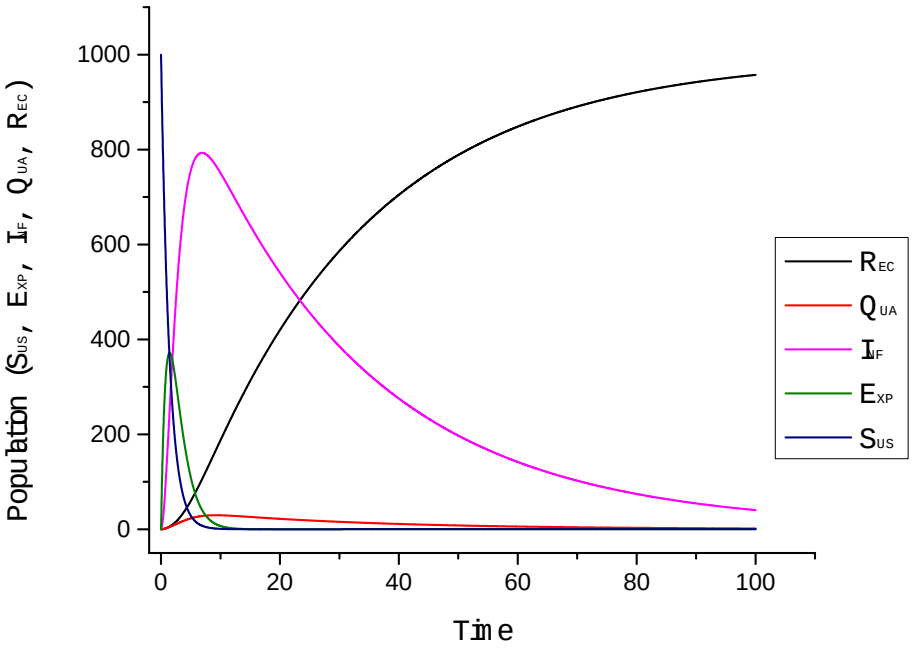

Figure 5.1: Dynamical behaviour of the proposed system

Figure 5.2 shows the result of the proposed model without implementation of the quarantine strategy. We see that infectious hosts are diminishing very slowly and recovery process is slow as well. Figure 5.1 shows the behaviour of the model's entities of the proposed system using quarantine strategy. We see from Figure 5.1 that infectious hosts are diminishing sharply when we applied quarantine strategy.

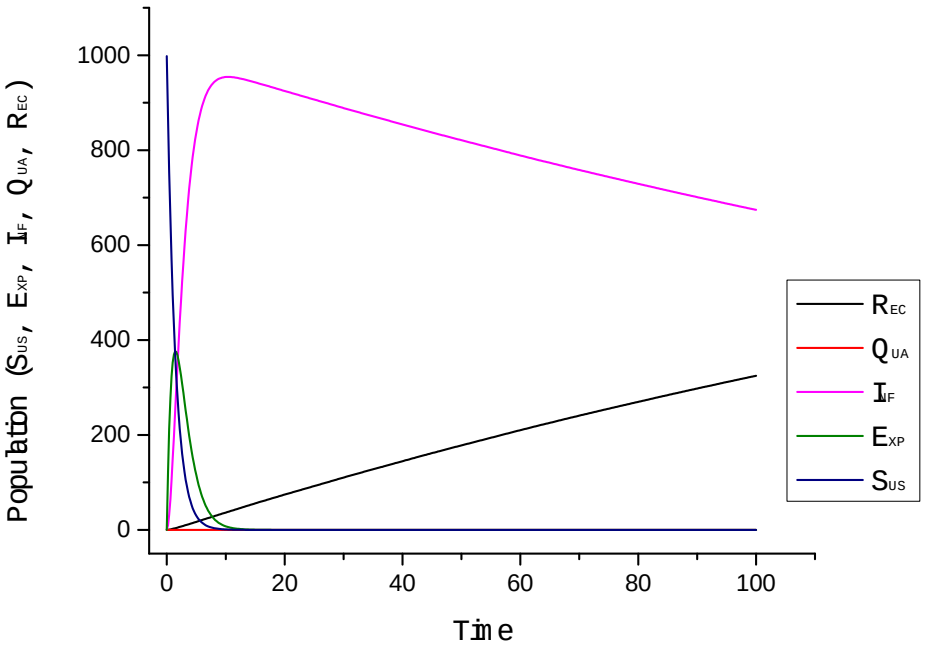

Figure 5.2: Dynamical behaviour of the proposed system without Quarantine

Figure 5.3 shows the results of susceptible, exposed and infectious classes with respect to the quarantine class. We observe in this Figure that nodes from these classes (Susceptible, Exposed and Infectious) are recovering quickly with application of the quarantine strategy.

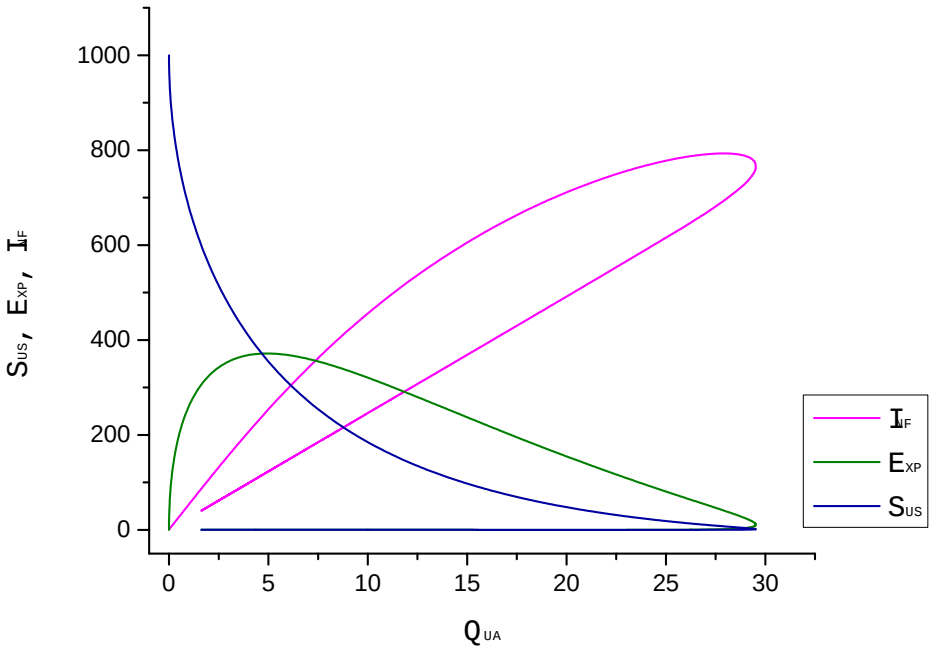

Figure 5.3: Quarantine effect on different compartments

Figure 5.4 shows the behaviour of infectious class with and without quarantine implementation. We observe that when hosts are infected then quarantine is very effective solution.

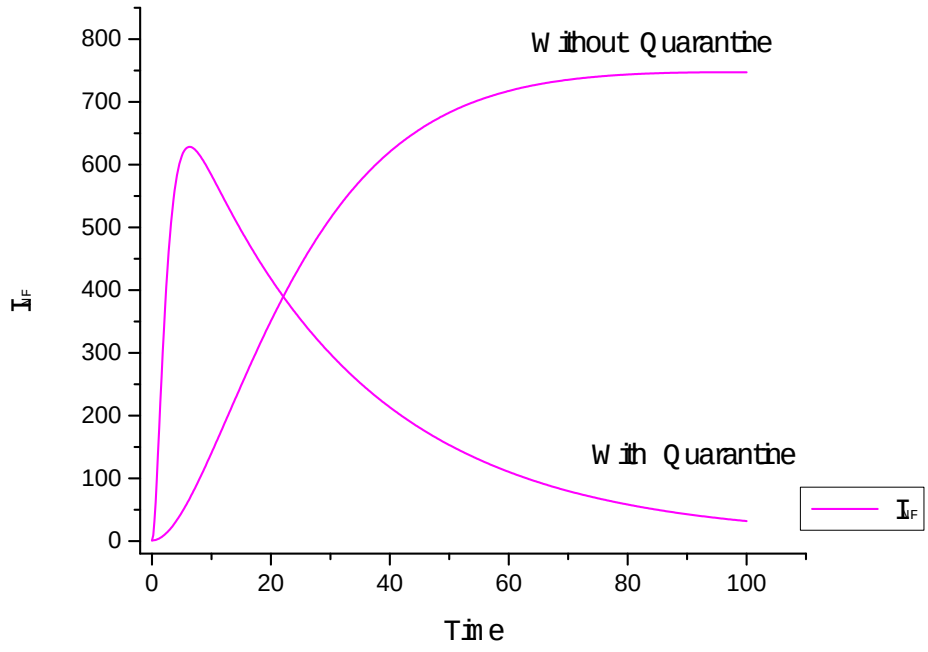

Figure 5.4: Dynamic behaviour of infectious class with and without quarantine

Figure 5.5 shows the relationship between recovered and susceptible compartments. We can observe decrease in susceptible nodes when recovered nodes are increasing. It shows that as time passes, recovered hosts increases gradually. It means susceptibility towards worm decreases with time.

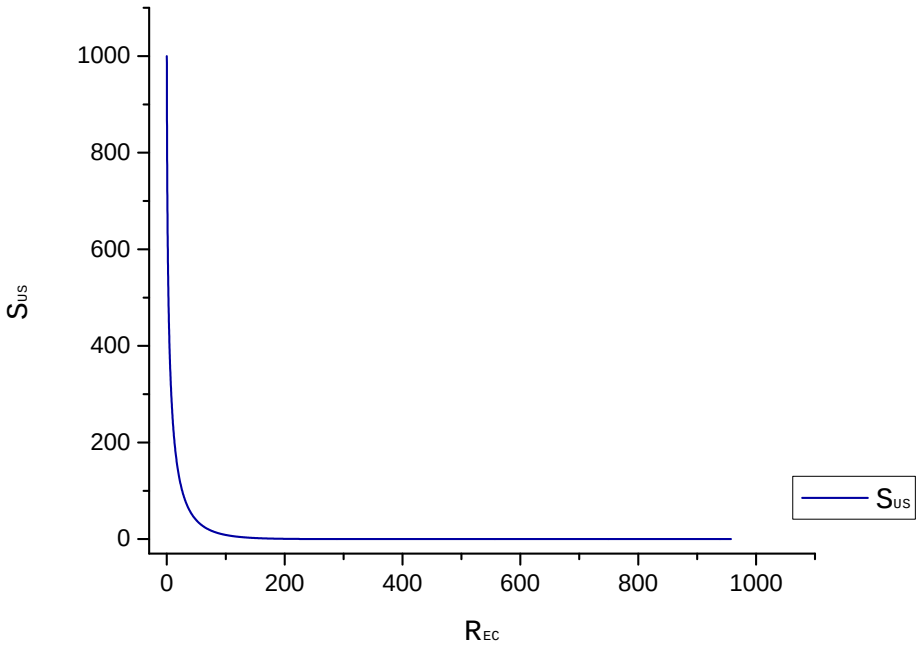

Figure 5.5: Behaviour of susceptible versus recovered compartment

For next analysis, all parameters are same except these: $\gamma = 0.01$, $\delta = 0.1$, $n_1 = 0.001$, $n_2 = 0.001$, $n_3 = 0.001$, $n_4 = 0.002$.

We see in Figure 5.6 that the worm is spreading quickly in the whole network and nodes are becoming infectious with high pace when infection rate is higher than the recovery rate.

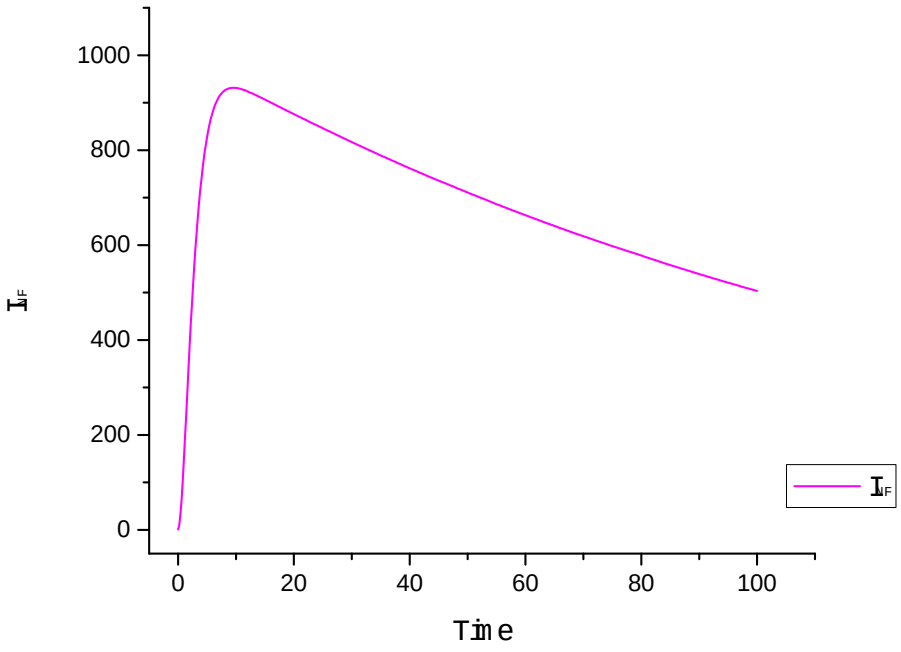

Figure 5.6: Behaviour of infectious compartment when infection rate is greater than the recovery rate

5.2 Verification Results

This section contains the verification results of the proposed system.

5.2.1 Verification of Quantitative Properties using PRISM

In the context of several applications, formal methods have been used to understand and characterize the behaviour of population models. *Probabilistic Model Checking* is a fully automated formal method for verifying quantitative properties of systems that exhibit stochastic behaviour. It is based on the exhaustive searching of the state space. We can check various be-

havioural properties regarding time and probabilities through *Probabilistic Model Checking* [61].

For quantitative analysis, we have developed a *CTMC* in *PRISM* model checker. The logic used is *Continuous Stochastic Logic (CSL)* which is an extension of Probabilistic Computation Tree Logic (PCTL). PCTL itself is a variant of CTL (Computation Tree Logic) where the path quantifiers (A and E) are replaced by a probabilistic operator (P) [61]. We can also use the keyword "filter" to customize the *PRISM* properties. Filters are represented using the following form:

$filter(op,\ prop,\ states)$, where *op* denotes the operation we want to perform, *prop* denote the property we want to verify and *states* is boolean valued expression representing all those states over which we are verifying the specified property using filter [68].

For analysis purpose, the parameters in the experiments are: $N = 10,\ S_{US}(0) = 9,\ E_{XP}(0) = 0,\ I_{NF}(0) = 1,\ D_{EL}(0) = 0,\ Q_{UA}(0) = 0,\ R_{EC}(0) = 0,\ \beta = 0.7,\ \alpha = 0.66,\ \gamma = 0.9,\ \delta = 0.8,\ n_1 = 0.01,\ n_2 = 0.01,\ n_3 = 0.02,\ n_4 = 0.03,\ \varphi_1 = 0.001,\ \varphi_2 = 0.005,\ \varphi_3 = 0.004,\ \omega_1 = 0.001$ and $\omega_2 = 0.001$.

We verified the behavioural properties in *PRISM*. We have checked a list of behavioural properties and some of those are illustrated below:

- $P =?\quad [F\quad I_{NF} = 0]$

  This formula inquires, "what is the probability that infection will be eradicated eventually?". This property is verified with probability 1. It means that retreat of the infection is unavoidable. The probability of reaching the state where infected individuals are 0 is 1.

- $P =?\quad [F\quad R_{EC} > S_{US}]$

  This represents the probability that recovered individuals will be greater than susceptible individuals. In other words, it means that most of the

population is infected at certain points of time. This property is verified with probability 1 in the proposed model which means that most of the population first became infected at certain times and then eventually recovered.

- $P =?\quad [F \leq 10 \quad I_{NF} = N/2]$

  It shows the probability that half of the population will be infected within 10 time units. The results shows that there is a 99 percent chance of infection spreading in half of the population within 10 time units.

- $P =?\quad [F \leq 10 \quad I_{NF} \leq S_{US}]$

  This property inquires, "what is the probability that infected nodes will exceed the susceptible nodes within 10 time units?". This property shows how the worm is spreading in the initial period of time. There is a 99 percent chance that infected individuals will exceed the susceptible within 10 time units.

- $P =?\quad [F \leq 10 \quad I_{NF} = N]$

  It represents the probability that the whole network will be infected within 10 time units. There is only a 12 percent chance of the infection spreading in every computer within this time.

- $P \geq 1 \quad [G \quad (S_{US} + E_{XP} + I_{NF} + D_{EL} + Q_{UA} + R_{EC}) = N]$

  This property represents the very important principle "Invariance". It represent the probability that sum of all nodes will be equal to the size of the population. This should be the case in our proposed model because the population can never be negative. Since our model is based on closed population, this property should globally hold and the result shows that it is true in the proposed model.

- $R =?\quad [F \quad I_{NF} = 0]$

  This property inquires, "what is the expected time for a network to be eventually infection free?". The expected time for the extinction of the worm is 58 time units.

- $filter(forall, \quad P \geq 1 \quad [F \quad R_{EC} = N])$

  This states that we will eventually reach a state, initiating from any reachable state, where all hosts are recovered with probability 1. This property is true in the proposed model.

- $S =?\quad [I_{NF} = 0]$

  This property represents the long run probability (steady state) of the worm's extinction from the network. The result obtained through this property shows that there is a 70 percent chance of the worm being neutralized. This shows that network is never going to be infection free, there will always be a chance of re-infection. *Model Checking* results successfully validates the simulation results.

These behavioural properties allow us to know if behaviour of *SEIDQR(S/I)* model is stable and valid at every instance of time.

5.2.2 Verification of Hybrid Quarantine Strategy using PRISM

In order to analyze the effectiveness of the quarantine strategy through *Model Checking*, the parameters in the experiments are same as defined above.
Table 5.1 summarises the results of *Model Checking* with and without quarantine strategy based on these parameters. *Model Checking* validates the use of quarantine strategy in worm containment. We have compared results in

three scenarios. In the first scenario, we have checked the probability of the whole network becoming infectious within 10 time units. It shows a huge difference in results with and without quarantine implementation. When we have implemented quarantine strategy it shows that there is only a 12 percent chance of spreading the worm in the whole population within 10 time units but without quarantine it spreads rapidly. Without applying the quarantine strategy there is a 74 percent chance of the infection spreading throughout the whole population within 10 time units. In second scenario, we have inquired, "What are the chances that half of the population will be infected within 10 time units?". In that particular case probability is equal in both cases with and without quarantine strategy. In third scenario, we have investigated the chances of the network becoming infection free within 100 time units. In case of the quarantine strategy, there is a 91 percent chance that whole network will be infection free within the specified time limit. In case of without the quarantine strategy, there is a 0 percent chance that whole network will be infection free within 100 time units.

Property	**Probabilities With Quarantine**	**Probabilities Without Quarantine**
$P =?\ [F \leq 10\ \ I_{NF} = N]$	0.1207	0.7412
$P =?\ [F \leq 10\ \ I_{NF} = N/2]$	0.9999	0.9999
$P =?\ [F \leq 100\ \ I_{NF} = 0]$	0.9131	0.0029

Table 5.1: Results with and without Quarantine for sample size 10

Table 5.2 summarises the results of *Model Checking* with and without quarantine strategy based on the above mentioned parameters except that population size is 26 now, susceptible hosts are 25 and infectious host is 1. In the first study, we have checked the probability of the worm infecting the whole population within 10 time units. The probability of the infection spreading across the whole population after implementing the quarantine strategy was

minimal, on the contrary, without the use of the quarantine strategy the probability of infection increased by 40 percent. In the second experiment, we discovered that the probability of the infection spreading to half of the population was 0.9999 within 10 time units, with or without the implementation of the quarantine strategy. In the third experiment, we have discovered that with the implementation of the quarantine strategy, almost 69 percent of the population was infection free within 100 time units but the percentage was considerably less without quarantine.

Property	**Probabilities With Quarantine**	**Probabilities Without Quarantine**
$P =? \ [F \leq 10 \ \ I_{NF} = N]$	0.0013	0.3972
$P =? \ [F \leq 10 \ \ I_{NF} = N/2]$	0.9999	0.9999
$P =? \ [F \leq 100 \ \ I_{NF} = 0]$	0.6849	0.0016

Table 5.2: Results with and without Quarantine for sample size 26

This analysis indicates the significance of implementing mitigation techniques in the initial stages of an infection which otherwise would be incredibly difficult to control and may cause severe harm to the majority of the hosts and this will be very costly. The results in Table 5.1 and 5.2 shows the importance of implementing the hybrid quarantine strategy. It also enlightens the fact that within a larger population the speed of a worm's spread gradually decelerates with time. The results reported in Table 5.1 and 5.2 also indicates that smaller population recovered quickly as compared to the larger population.

5.2.3 Verification of Qualitative Properties using Charlie

The reachability graph is strongly connected which means that the graph is homogeneous. Each pair of markings are reachable from one another. Since the proposed system (compartmental model) is homogeneous therefore, the reachability graph is homogeneous. This reachability graph indicates that all markings of the model always end up in a cycle which also states that the system is deadlock free, reversible and live. This model ensures the invariance principle which means that population will never be negative. This property can be ensured by P-invariant and is verifiable in our model. Figure 5.7 shows the reachability graph of the *SPN*.

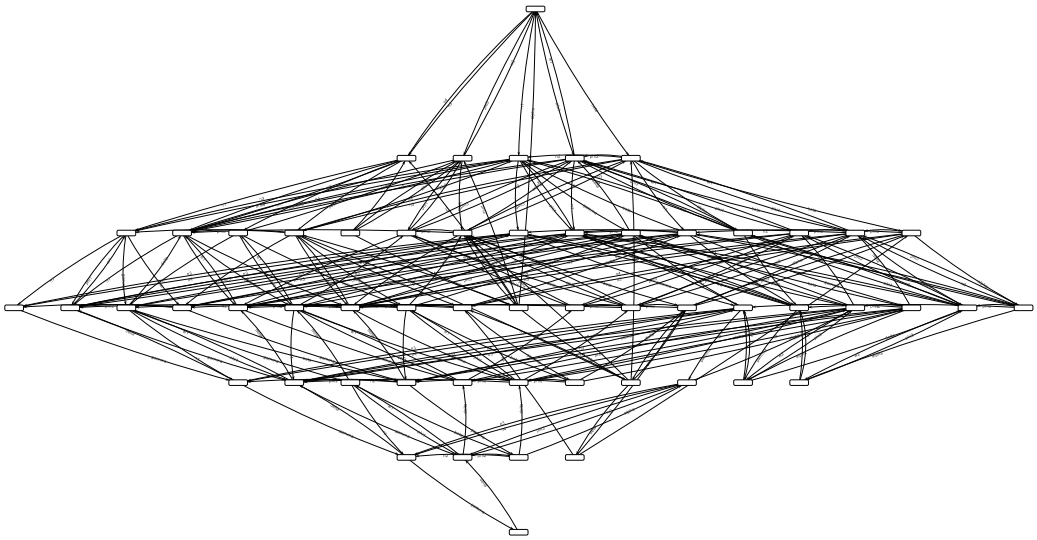

Figure 5.7: Reachability Graph
Figure 5.7 shows the reachability graph consisting a total of 56 unique markings and 273 transitions with initial marking M_0=$(2, 0, 1, 0, 0, 0)$.

Chapter 6

Discussion

In this work, we have proposed a new model called *SEIDQR(S/I)* model which was based on the delays experienced by the hosts before worm containment and possible reinfection probability. We presented modelling of a compartmental model via *Petri Net* approach. Moreover, in order to study the dynamic behaviour of the proposed model we used two types of analysis techniques: *Simulation* and *Probabilistic Model Checking*.

It is important to model and study the behaviour of the system before resulting to deployment. Sometimes, models are developed after deployment as well in order to study the dynamics of the systems. The domain of communication systems are full of queries regarding cost and efficiency etc.. In order to resolve these problems it requires the construction and study of analytical and simulation models before the development and deployment of the systems. It is necessary to develop models for qualitative and quantitative understanding of the systems. The nature of traffic between communication systems is unpredictable and therefore, it is typical to develop a stochastic model to represent such systems [19]. Therefore, in this study we have used *SPN* to construct models. Mathematical models have been conventionally

used for analysis of worm propagation. However, these models rely on unreasonable assumptions and may require long periods of time before getting any results [19]. In this work, we have used the *SPN* to model worm propagation. Then, *CTMC* of the proposed model is constructed in *PRISM* model checker.

Our results are comparable to Mishra and Tyagi [69] where dynamical behaviour is achieved after quarantining 50 nodes. Our model encapsulated a population of 1000 nodes and generated results more accurately by quarantining 30 nodes while retaining similar behavioural curves, as that of Mishra and Tyagi [69].

The analysis of the *CTMC* strongly validates the results obtained through simulation of the *SPN*, which makes the proposed framework valid for application in the field of epidemiology. The analysis obtained through *Model Checking* also supports assumption that real network is never infection free. There is always a chance of possible re-infection. We have already showed this through simulation of the *SPN*. We verified the structural property of the *SPN* through *Charlie* tool. Since proposed model is homogeneous, it should be strongly connected and this was confirmed through *Charlie* tool. We also verified some behavioural properties of the *SPN* such as reachability, liveliness, reversibility and deadlock freeness through *Charlie* tool.

A major disadvantage of the modelling approach is the computational cost of the method. In order to evaluate quantitative properties via the *Probabilistic Model Checking* method, it requires a reasonable amount of time (in hours). To overcome this computational obstacle we had to limit our sample size to the maximum of 26 hosts. A possible solution to this problem is to use approximate *Model Checking* [70]. However, *Probabilistic Model Checking* offers promising outcomes in the analysis of dynamics of compartmental

models and therefore, it is worth further investigation.

Chapter 7

Conclusion

In this thesis, we have proposed a framework to formally verify and validate the compartmental models via *Model Checking* and simulation. The proposed framework can be applied to any epidemiological compartmental network model. Beginning with the development of a *SEIDQR(S/I)* model in *Petri Nets* and *PRISM*, we were able to get insight into how *SEIDQR(S/I)* works. On the basis of proposed methodology, we were able to simulate as well as investigate the *SEIDQR(S/I)* model through queries encoded in *CSL*. By varying different parameters of the proposed model, we verified its behavioural properties. Using this approach, we checked certain situations through these properties such as when the worm's infection will be at its peak point, its duration and so on and so forth. The *Petri Net* approach described here have allowed us to perform modelling of the system easily and quickly as compared to other analysis methods. According to this work, we have come to the conclusion that quarantine strategies are extremely effective in reducing the risk of propagating the worm and, in fact, have an outstanding effect in regulating it. *Probabilistic Model Checking* allowed us to explore many behavioural properties of our model. The proposed approach is applicable

for both understanding worm propagation and developing more challenging worm defence strategies. There are many factors that effect worm propagation such as delay, bandwidth and activity of device in the network, which cannot be neglected and will be taken into consideration in our future works.

Appendices

Appendix A

Supplementary Content

A.1 CTMC

```
ctmc                        //Specifying the type of probabilistic
                            model


const int Max;              //Upper limit
const int N;                //Population size
const double v_beta;        //The rate at which susceptible hosts
                            become exposed
const double v_alpha;       //The rate at which exposed hosts
                            become infectious
const double v_n1;          //Delayed rate of susceptible hosts
const double v_n2;          //Delayed rate of Exposed hosts
const double v_n3;          //Delayed rate of Infectious hosts with
                            anomaly detection
const double v_n4;          //Delayed rate of infectious hosts with
                            signature based detection
```

```
const double v_phi1;      //The rate at which susceptible hosts
                          become recovered
const double v_phi2;      //The rate at which exposed hosts
                          become recovered
const double v_phi3;      //The rate at which infectious hosts
                          become recovered
const double v_omega1;    //The rate at which recovered hosts
                          become susceptible
const double v_omega2;    //The rate at which recovered hosts
                          become infectious
const double v_gamma;     //Quarantined rate of delayed hosts
const double v_delta;     //Recovery rate of quarantined hosts


module  SEIDQRSI          //Name of the module


//Definition of State Variables
DEL:  [ 0..Max ]  init 0;      //Delayed Hosts
INF:  [ 0..Max ]  init 1;      //Infectious Hosts
QUA:  [ 0..Max ]  init 0;      //Quarantined Hosts
REC:  [ 0..Max ]  init 0;      //Recovered Hosts
SUS:  [ 0..Max ]  init N-1;    //Susceptible Hosts
EXP:  [ 0..Max ]  init 0;      //Exposed Hosts
```

```
//Definition of Transition Rules
//If guards are met then state variable will be updated by the
specified rate.
[n4]
(Inf > 0) & (Del < Max )-> (v_n4) * Inf :
(Del' = Del + 1) & (Inf' = Inf - 1);
[omega2]
(Rec > 0) & (Inf < Max )-> (v_omega2) * Rec :
(Inf' = Inf + 1) & (Rec' = Rec - 1);
[phi2]
(Exp > 0) & (Rec < Max )-> (v_phi2) * Exp :
(Exp' = Exp - 1) & (Rec' = Rec + 1);
[phi3]
(Inf > 0) & (Rec < Max )-> (v_phi3) * Inf :
(Inf' = Inf - 1) & (Rec' = Rec + 1);
[n3]
(Inf > 0) & (Del < Max )-> (v_n3) * Inf :
(Del' = Del + 1) & (Inf' = Inf - 1);
[n1]
(Sus > 0) & (Del < Max )-> (v_n1) * Sus :
(Del' = Del + 1) & (Sus' = Sus - 1);
[omega1]
(Rec > 0) & (Sus < Max )-> (v_omega1) * Rec :
(Rec' = Rec -1) & (Sus' = Sus + 1);
```

```
[phi1]
(Sus > 0) & (Rec < Max )-> (v_phi1) * Sus :
(Rec' = Rec + 1) & (Sus' = Sus - 1);
[delta]
(Qua > 0) & (Rec < Max )-> (v_delta) * Qua :
(Qua' = Qua - 1) & (Rec' = Rec + 1);
[gamma]
(Del > 0) & (Qua < Max )-> (v_gamma) * Del :
(Del' = Del - 1) & (Qua' = Qua + 1);
[n2]
(Exp > 0) & (Del < Max )-> (v_n2) * Exp :
(Del' = Del + 1) & (Exp' = Exp - 1);
[alpha]
(Exp > 0) & (Inf < Max )-> (v_alpha) * Exp :
(Exp' = Exp - 1) & (Inf' = Inf + 1);
[beta]
(Sus > 0) & (Exp < Max )-> (v_beta) * Sus :
(Exp' = Exp + 1) & (Sus' = Sus - 1);

endmodule       //end of module

//Definition of the reward
rewards "time"
 true : 1;      //assign reward 1 to each state
endrewards
//end of reward
```

Bibliography

[1] J. Jung, *Real-time detection of malicious network activity using stochastic models*. PhD thesis, Citeseer, 2006.

[2] D. Christoffersen and B. J. Mauland, "Worm detection using honeypots," 2006.

[3] P. P. Reddy and P. R. Reddy, "Modeling the spread of malware in computer networks," *Month*, 2009.

[4] Y. Tang and S. Chen, "Defending against internet worms: A signature-based approach," in *INFOCOM 2005. 24th Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings IEEE*, vol. 2, pp. 1384–1394, IEEE, 2005.

[5] M. Feily, A. Shahrestani, and S. Ramadass, "A survey of botnet and botnet detection," in *Emerging Security Information, Systems and Technologies, 2009. SECURWARE'09. Third International Conference on*, pp. 268–273, IEEE, 2009.

[6] N. Weaver, "A brief history of the worm." `http://www.symantec.com/connect/articles/brief-history-worm/`. Updated November 26, 2011.

[7] P. Li, M. Salour, and X. Su, "A survey of internet worm detection and containment," *Communications Surveys & Tutorials, IEEE*, vol. 10, no. 1, pp. 20–35, 2008.

[8] C. C. Zou, W. Gong, and D. Towsley, "Code red worm propagation modeling and analysis," in *Proceedings of the 9th ACM conference on Computer and communications security*, pp. 138–147, ACM, 2002.

[9] M. Bailey, E. Cooke, F. Jahanian, and D. Watson, "The blaster worm: Then and now," *Security & Privacy, IEEE*, vol. 3, no. 4, pp. 26–31, 2005.

[10] C. C. Zou, W. Gong, and D. Towsley, "Worm propagation modeling and analysis under dynamic quarantine defense," in *Proceedings of the 2003 ACM workshop on Rapid malcode*, pp. 51–60, ACM, 2003.

[11] C. Shannon and D. Moore, "The spread of the witty worm," *Security & Privacy, IEEE*, vol. 2, no. 4, pp. 46–50, 2004.

[12] N. M. Mukamurenzi, "Storm worm: A p2p botnet," 2008.

[13] S. Shin and G. Gu, "Conficker and beyond: a large-scale empirical study," in *Proceedings of the 26th Annual Computer Security Applications Conference*, pp. 151–160, ACM, 2010.

[14] P. Kabiri and A. A. Ghorbani, "Research on intrusion detection and response: A survey.," *IJ Network Security*, vol. 1, no. 2, pp. 84–102, 2005.

[15] R. M. Anderson and R. M. May, *Infectious diseases of humans*, vol. 1. Oxford university press Oxford, 1991.

[16] Q. Zhu, X. Yang, and J. Ren, "Modeling and analysis of the spread of computer virus," *Communications in Nonlinear Science and Numerical Simulation*, vol. 17, no. 12, pp. 5117–5124, 2012.

[17] M. Roberts and J. Heesterbeek, "Mathematical models in epidemiology," *Encyclopedia of Life Support Systems (EOLSS)*, 2003.

[18] S. Misslinger, "Internet worm propagation," *Technische University Munchen*, 2003.

[19] A. A. Bhat, *Stochastic Petri Net Models of Service Availability in a PBNM System for Mobile Ad Hoc Networks*. PhD thesis, Virginia Polytechnic Institute and State University, 2004.

[20] N. Weaver, V. Paxson, S. Staniford, and R. Cunningham, "A taxonomy of computer worms," in *Proceedings of the 2003 ACM workshop on Rapid malcode*, pp. 11–18, ACM, 2003.

[21] D. Moore, C. Shannon, *et al.*, "Code-red: a case study on the spread and victims of an internet worm," in *Proceedings of the 2nd ACM SIGCOMM Workshop on Internet measurment*, pp. 273–284, ACM, 2002.

[22] D. Moore, V. Paxson, S. Savage, C. Shannon, S. Staniford, and N. Weaver, "Inside the slammer worm," *IEEE Security & Privacy*, vol. 1, no. 4, pp. 33–39, 2003.

[23] S. Staniford, V. Paxson, N. Weaver, *et al.*, "How to own the internet in your spare time.," in *USENIX Security Symposium*, pp. 149–167, 2002.

[24] C. C. Zou, D. Towsley, W. Gong, and S. Cai, "Routing worm: A fast, selective attack worm based on ip address information," in *Proceedings*

of the 19th Workshop on Principles of Advanced and Distributed Simulation, pp. 199–206, IEEE Computer Society, 2005.

[25] Y. Wang, S. Wen, S. Cesare, W. Zhou, and Y. Xiang, "The microcosmic model of worm propagation," *The Computer Journal*, vol. 54, no. 10, pp. 1700–1720, 2011.

[26] Z. Chen, L. Gao, and K. Kwiaty, "Modeling the spread of active worms," in *INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications. IEEE Societies*, vol. 3, pp. 1890–1900, IEEE, 2003.

[27] C. C. Zou, D. Towsley, and W. Gong, "On the performance of internet worm scanning strategies," *Performance Evaluation*, vol. 63, no. 7, pp. 700–723, 2006.

[28] D. Copley, R. Hassell, B. Jack, K. Lynn, R. Permeh, and D. Soeder, "Analysis: Blaster worm. eeye digital security research," 2003.

[29] R. W. Thommes and M. Coates, "Epidemiological modelling of peer-to-peer viruses and pollution.," in *INFOCOM*, vol. 6, pp. 1–12, 2006.

[30] X. Fan and Y. Xiang, "Modeling the propagation of peer-to-peer worms," *Future generation computer systems*, vol. 26, no. 8, pp. 1433–1443, 2010.

[31] W. Fan and K. Yeung, "Online social networksparadise of computer viruses," *Physica A: Statistical Mechanics and its Applications*, vol. 390, no. 2, pp. 189–197, 2011.

[32] G. Yan and S. Eidenbenz, "Modeling propagation dynamics of bluetooth worms (extended version)," *Mobile Computing, IEEE Transactions on*, vol. 8, no. 3, pp. 353–368, 2009.

[33] A.-L. Barabási and R. Albert, "Emergence of scaling in random networks," *science*, vol. 286, no. 5439, pp. 509–512, 1999.

[34] C. Pillers Dobler, "Mathematical statistics: Basic ideas and selected topics," *The American Statistician*, vol. 56, no. 4, pp. 332–332, 2002.

[35] D. Brumley, L.-H. Liu, P. Poosankam, and D. Song, "Design space and analysis of worm defense strategies," in *Proceedings of the 2006 ACM Symposium on Information, computer and communications security*, pp. 125–137, ACM, 2006.

[36] Z. Chen and C. Ji, "Optimal worm-scanning method using vulnerable-host distributions," *International Journal of Security and Networks*, vol. 2, no. 1, pp. 71–80, 2007.

[37] M. Boguá, R. Pastor-Satorras, and A. Vespignani, "Epidemic spreading in complex networks with degree correlations," in *Statistical mechanics of complex networks*, pp. 127–147, Springer, 2003.

[38] R. Pastor-Satorras and A. Vespignani, "Epidemic spreading in scale-free networks," *Physical review letters*, vol. 86, no. 14, p. 3200, 2001.

[39] C. C. Zou, W. Gong, D. Towsley, and L. Gao, "The monitoring and early detection of internet worms," *IEEE/ACM Transactions on Networking (TON)*, vol. 13, no. 5, pp. 961–974, 2005.

[40] R. M. Anderson, R. M. May, *et al.*, "Population biology of infectious diseases.," in *[Report of the Dahlem Workshop, Berlin, 14th-19th March 1982].*, Springer-Verlag, 1982.

[41] B. K. Mishra and D. Saini, "Mathematical models on computer viruses," *Applied Mathematics and Computation*, vol. 187, no. 2, pp. 929–936, 2007.

[42] 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.

[43] T. Dong, X. Liao, and H. Li, "Stability and hopf bifurcation in a computer virus model with multistate antivirus," in *Abstract and Applied Analysis*, vol. 2012, Hindawi Publishing Corporation, 2012.

[44] L.-X. Yang and X. Yang, "Propagation behavior of virus codes in the situation that infected computers are connected to the internet with positive probability," *Discrete Dynamics in Nature and Society*, vol. 2012, 2012.

[45] C. Gan, X. Yang, W. Liu, Q. Zhu, and X. Zhang, "Propagation of computer virus under human intervention: a dynamical model," *Discrete Dynamics in Nature and Society*, vol. 2012, 2012.

[46] J. Ren, X. Yang, Q. Zhu, L.-X. Yang, and C. Zhang, "A novel computer virus model and its dynamics," *Nonlinear Analysis: Real World Applications*, vol. 13, no. 1, pp. 376–384, 2012.

[47] Y. Yao, L. Guo, H. Guo, G. Yu, F.-x. Gao, and X.-j. Tong, "Pulse quarantine strategy of internet worm propagation: modeling and analysis," *Computers & Electrical Engineering*, vol. 38, no. 5, pp. 1047–1061, 2012.

[48] F. Wang, Y. Zhang, C. Wang, J. Ma, and S. Moon, "Stability analysis of a seiqv epidemic model for rapid spreading worms," *Computers & Security*, vol. 29, no. 4, pp. 410–418, 2010.

[49] D. Moore, "Caida analysis of code-red," *The Cooperate Association for Internet Data Analysis*, 2001.

[50] D. Moore and C. Shannon, "The spread of the code-red worm (crv2)," 2001.

[51] M. J. Keeling and P. Rohani, *Modeling infectious diseases in humans and animals*. Princeton University Press, 2008.

[52] J. Giesecke *et al.*, *Modern infectious disease epidemiology.* Edward Arnold (Publisher) Ltd., 1994.

[53] O. A. Toutonji, S.-M. Yoo, and M. Park, "Stability analysis of veisv propagation modeling for network worm attack," *Applied Mathematical Modelling*, vol. 36, no. 6, pp. 2751–2761, 2012.

[54] T. Murata, "Petri nets: Properties, analysis and applications," *Proceedings of the IEEE*, vol. 77, no. 4, pp. 541–580, 1989.

[55] R. David and H. Alla, *Discrete, continuous, and hybrid Petri nets*. Springer Science & Business Media, 2010.

[56] N. Lu, *Power system modeling using Petri nets*. PhD thesis, Rensselaer Polytechnic Institute, 2002.

[57] M. Heiner, M. Herajy, F. Liu, C. Rohr, and M. Schwarick, "Snoopy–a unifying petri net tool," in *Application and Theory of Petri Nets*, pp. 398–407, Springer, 2012.

[58] M. Kwiatkowska, G. Norman, and D. Parker, "Quantitative analysis with the probabilistic model checker prism," *Electronic Notes in Theoretical Computer Science*, vol. 153, no. 2, pp. 5–31, 2006.

[59] C. Baier, J.-P. Katoen, *et al.*, *Principles of model checking*, vol. 26202649. MIT press Cambridge, 2008.

[60] E. M. Clarke, O. Grumberg, and D. Peled, *Model checking*. MIT press, 1999.

[61] M. Kwiatkowska, G. Norman, and D. Parker, "Probabilistic model checking for systems biology," 2011.

[62] A. Aziz, K. Sanwal, V. Singhal, and R. Brayton, "Model-checking continuous-time markov chains," *ACM Transactions on Computational Logic (TOCL)*, vol. 1, no. 1, pp. 162–170, 2000.

[63] A. Aziz, K. Sanwal, V. Singhal, and R. Brayton, "Verifying continuous time markov chains," in *Computer Aided Verification*, pp. 269–276, Springer, 1996.

[64] M. Kwiatkowska, G. Norman, and D. Parker, *Symbolic Systems Biology*, ch. Probabilistic Model Checking for Systems Biology, pp. 31–59. Jones and Bartlett, 2010.

[65] M. K. G. N. D. Parker, "Prism 2.0: A tool for probabilistic model checking," 2004.

[66] A. Hinton, M. Kwiatkowska, G. Norman, and D. Parker, "Prism: A tool for automatic verification of probabilistic systems," in *Tools and Algorithms for the Construction and Analysis of Systems*, pp. 441–444, Springer, 2006.

[67] M. Heiner, R. Donaldson, and D. Gilbert, "Petri nets for systems biology," *Symbolic Systems Biology: Theory and Methods. Jones and Bartlett Publishers, Inc., USA (in Press, 2010)*, 2010.

[68] U. o. O. Department of Computer Science, "Property specification." `http://www.prismmodelchecker.org/manual/PropertySpecification/AllOnOnePage/`. Accessed May 20, 2015.

[69] B. K. Mishra and I. Tyagi, "Defending against malicious threats in wireless sensor network: A mathematical model," *International Journal of Information Technology and Computer Science (IJITCS)*, vol. 6, no. 3, p. 12, 2014.

[70] J.-P. Katoen, "Abstraction of probabilistic systems," in *Formal Modeling and Analysis of Timed Systems*, pp. 1–3, Springer, 2007.