Analytical Lifecycle Modeling and Threat Analysis of Botnets

Masood Khosroshahy

A Thesis
In the Department
of
Electrical and Computer Engineering

Presented in Partial Fulfillment of the Requirements
For the Degree of
Doctor of Philosophy

Concordia University
Montréal, Québec, Canada

March 2013

CONCORDIA UNIVERSITY

SCHOOL OF GRADUATE STUDIES

This is to certify that the thesis prepared

By: **Masood Khosroshahy**

Entitled: **Analytical Lifecycle Modeling and Threat Analysis of Botnets**

and submitted in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY (Electrical & Computer Engineering)

complies with the regulations of the University and meets the accepted standards with respect to originality and quality.

Signed by the final examining committee:

_____ Chair
Dr. A. Athienitis

_____ External Examiner
Dr. X. Liu

_____ External to Program
Dr. H. Harutyunyan

_____ Examiner
Dr. F. Khendek

_____ Examiner
Dr. A. Youssef

_____ Thesis Co-Supervisor
Dr. M.K. Mehmet Ali

_____ Thesis Co-Supervisor
Dr. D. Qiu

Approved by _____
Dr. J.X. Zhang, Graduate Program Director

February 26, 2013

_____
Dr. Robin A.L. Drew, Dean
Faculty of Engineering and Computer Science

Abstract

Analytical Lifecycle Modeling and Threat Analysis of Botnets

Masood Khosroshahy, Ph.D.

Concordia University, 2013

Botnet, which is an overlay network of compromised computers built by cybercriminals known as botmasters, is the new phenomenon that has caused deep concerns to the security professionals responsible for governmental, academic, and private sector networks. Botmasters use a plethora of methods to infect network-accessible devices (nodes). The initial malware residing on these nodes then either connects to a central Command & Control (C&C) server or joins a Peer-to-Peer (P2P) botnet. At this point, the nodes can receive the commands of the botmaster and proceed to engage in illicit activities such as Distributed Denial-of-Service (DDoS) attacks and massive e-mail spam campaigns.

Being able to reliably estimate the size of a botnet is an important task which allows the adequate deployment of mitigation strategies against the botnet. In this thesis, we develop analytical models that capture the botnet expansion and size evolution behaviors in sufficient details so as to accomplish this crucial estimation/analysis task. We develop four Continuous-Time Markov Chain (CTMC) botnet models: the first two, SComI and SComF, allow the prediction of initial unhindered botnet expansion in the case of infinite and finite population sizes, respectively. The third model, the SIC model, is a botnet lifecycle model

which accounts for all important node stages and allows botnet size estimates as well as evaluation of botnet mitigation strategies such as disinfections of nodes and attacks on botnet's C&C mechanism. Finally, the fourth model, the SIC-P2P model, is an extension of the SIC model suitable for P2P botnets, allowing fine-grained analysis of mitigation strategies such as index poisoning and sybil attack.

As the convergence of Internet and traditional telecommunication services is underway, the threat of botnets is looming over essential basic communication services. As the last contribution presented in this thesis, we analyze the threat of botnets in the 4G cellular wireless networks. We identify the vulnerability of the air interface, i.e. the Long Term Evolution (LTE), which allows a successful botnet-launched DDoS attack against it. Through simulation using an LTE simulator, we determine the number of botnet nodes per cell that can significantly degrade the service availability of such cellular networks.

Acknowledgements

I would like to thank Profs. Mustafa K. Mehmet Ali and Dongyu Qiu, my Ph.D. supervisors, for trusting me and giving me the opportunity to work on this Ph.D. thesis under their supervision. Further, the completion of this thesis would not have been possible without their significant support and contribution. In our many lively discussions throughout my time at Concordia, I have learned a lot from them which I highly value and carry to my future endeavors.

I would also like to thank my examining committee members throughout my Ph.D. program, from comprehensive exam, to proposal, to seminar, and to final defence, for their time and constructive criticism. My Ph.D. defence was the most wonderful, memorable time of my whole professional life, thanks to my hard work and the support of my supervisors as well as many others.

To all Concordia community, from faculty members, to colleagues, to departmental staff, and to maintenance staff, thanks for your support, dedication, and smiles.

Dedication

To all who keep an open mind, are not afraid of the unknown, and smile.

Contents

List of Figures

List of Abbreviations

C&C Command and Control

CTMC Continuous-time Markov Chain

DHT Distributed Hash Table

DDoS Distributed Denial-of-Service

IMS IP Multimedia Subsystem

ISP Internet Service Provider

ODE Ordinary Differential Equation

P2P Peer-to-Peer

PDE Partial Differential Equation

PGF Probability Generating Function

SComF Susceptible-Compromised-Finite

SComI Susceptible-Compromised-Infinite

SIC Susceptible-Infected-Connected

TCP Transmission Control Protocol

UDP User Datagram Protocol

1. Introduction

1.1. Motivation

Botmasters, the cyber criminals behind botnets, leverage a wide range of methods to infect network-accessible devices, with the majority of the devices being personal computers in homes, businesses, schools, and governments. Once infected, these devices (or *nodes*) form botnets and are remotely controlled by the botmasters for illicit activities such as sending e-mail spam and extortion by threats of launching Distributed Denial-of-Service (DDoS) attacks. In recent years, the number of infected and remotely controlled nodes in each of the major botnets has reached the order of millions. The cumulative processing and bandwidth resources at the disposal of cyber criminals are therefore enough to severely attack any entity or temporarily knock entire countries off the Internet; this has resulted in the designation of botnets as a major security threat.

An important aspect of botnets that needs to be understood and predicted is their size; the bigger the size, the higher the threat level. Availability of accurate mathematical models of population size evolution enables security experts to plan ahead and deploy adequate resources when responding to a growing threat of an emerging botnet. In this thesis, we focus on this aspect of botnets, i.e., their size and the mathematical models thereof. Next, we provide a detailed evidence regarding the very real threat of botnets.

1.1.1. Botnet Threat

Mansfield-Devine [2] reports that there is about 25% chance that an average Windows user gets infected in any given year. The growing trend in recent years has been the rise of botnets as a result of botnet-related malware. Botnets have been used in variety of illicit activities. In what follows, we provide some of the highlights.

Sending e-mail spams to promote often inappropriate products and services is one such illicit activity. As IP addresses of e-mail spam sources can be blacklisted by e-mail providers, botmasters use thousands, even millions, of *nodes* (bots or compromised computers) as sources of e-mail spams; hence, largely avoiding the blacklisting efforts. Bleaken [3] reports that in one single takedown of a rogue ISP which hosted the main infrastructure of few botnets in November 2008, the level of e-mail spams dropped about 80% instantly.

Distributed Denial-of-Service (DDoS) attacks are conducted by flooding web servers and other network services with countless useless requests/packets. A DDoS attack would lead to those servers and services going out of operation, not being able to service the clients. The distributed nature of nodes in the botnet has thus made botnets the perfect platforms from which such DDoS attacks are launched. Mielke and Chen [4] mention that the country of Estonia came under such attack in April 2007 which knocked off critical infrastructure and the media.

Google (Daswani and Stoppelman [5]) has reported another use of botnets: "*low-noise click fraud attack against syndicated search engines*." Click fraud refers to the action of a user who clicks on an advertisement without having any genuine interest about the advertisement content. The user's action leads to the advertiser being unfairly charged and the publisher (which has run the advertisement) getting unfairly paid (Google runs the service that facilitates the transaction between the two). The user who engages in click fraud and the publisher are criminally related. Google reports [5] having detected a botnet of 100,000 nodes which has been designed solely to make automated clicks on advertisements.

Other examples of botnets worth mentioning are Mariposa and Conficker. Mansfield-

Devine [2] notes that the Mariposa botnet has been estimated to have 13 million computers across 190 countries. A working group consisting of anti-malware industry researchers cooperated with the law enforcement authorities in order to make arrests and take down the botnet. This however was quickly followed by a DDoS attack against a key working group member which resulted in temporary crippling of a major ISP.

Conficker (also known as Kido and Downadup) is another major botnet with impressive sophistication in design. Emm [6] (Kaspersky Lab) reports that, as a version of Conficker is using domain names in its Command & Control (C&C) architecture, it generated and registered 50,000 domain names per day and used some of these randomly in order to evade detection and analysis. The size of Conficker is estimated to be between 5 and 10 million nodes. Porras [7] notes how fast Conficker has evolved to combat the countermeasures deployed by the anti-malware industry: five major variants (lettered A through E) from October 2008 to April 2009, having both centralized (domain name based) and Peer-to-Peer-based management and operation.

The latest twist in botnet technology has been the creation of crimeware toolkits such as Zeus. These are toolkits which allow relatively novice criminals to create and operate their very own botnets using a control panel to manage the infected computers and the stolen information. Binsalleeh et al. [8] identify Zeus as the number one botnet threat that has 3.6 million infections in the United States alone and is estimated to be responsible for 44% of the banking-related malware infections.

Bleaken [3] notes that authorities in many countries are worried about botnets becoming the weapon of choice against government and business resources in the next battleground – botnets would be a lot cheaper to construct and operate than conventional weapons. Finally, Bradbury [9] observes that the organized crime behind botnets is hierarchical and modular, with the real minds behind the operation are disguised at the top, evading getting tracked down and apprehended.

1.2. Analytical Botnet Models

In computer science, the term *virus* was first used in late 1980's to refer to a "self-replicating" code intended to do damage. Facing this new phenomenon, Murray [10] was the first who suggested the application of epidemiology for studying the propagation of computer virus. However, it was not until early 1990's that a major work on producing analytical models was done [11]. In the course of the two decades that followed, numerous other analytical models based on the same premises were proposed such as [11, 12, 13, 1]. Before proceeding further, few definitions are due:

Node Stage A node (an arbitrary network-accessible device in the Internet) can be in either of the *stages* defined in the analytical model (e.g., *Susceptible* and *Infected* stages). With time, depending on the model, nodes can usually transition from one stage to another. In this thesis, we use the term *stage* in the context of a node and the term *state* in the context of the whole system to avoid confusion; the terminology of the cited works has been adapted to be compatible with ours. *State* of the system, therefore, is used to indicate the number of nodes that are in each *stage* at any given time.

Lifecycle *Lifecycle* indicates the fact that nodes change stage in the lifetime of the botnet. Botnet refers to the nodes that are in a certain stage. *Botnet lifecycle*, on the other hand, refers to the span of time from the appearance of a botnet to its disappearance. During the botnet lifecycle, state of the system varies as a function of the time.

An analytical botnet model can capture and analyze the expansion and shrinkage of a botnet; hence its *lifecycle*. Analytical botnet models keep track of the number of nodes in each *stage* over time. Each analytical model defines certain possible stages for nodes. Limiting the number of stages to the important ones leads to the development of a tractable model. Further, the model then becomes general enough and can be applied in the study

of majority of botnets, as it does not contain features found rarely in more than one or few botnets.

Analytical models may provide significant benefits in the fight against botnets. When either a new botnet threat emerges or an existing botnet goes into a rapid growth period due to a new infection, then there are two main questions that we would like to have answers to. One of them will be the determination of seriousness of the threat, which requires prediction of the size of the botnet as a function of time. This will let us know the number of nodes that eventually may be compromised. The other will be to determine the appropriate mix of mitigation strategies that need to be deployed to stop the growth of the botnet and possibly reverse it. In both cases, a good analytical model will be helpful if the estimates of its parameters are available. As a result of the growing botnet threat, new organizations are emerging that continuously keep track of botnets and measure their sizes. Thus, it is expected that the estimates of the model parameters will become available so that analytical models may be used to give answers to the above questions.

Methods in the development of analytical models for botnets can be traced back to nearly a century ago with the development of models capturing the spread of disease among humans in the mathematical epidemiology. Beginning from the 1980's to the present date, computer scientists have used epidemiological models, or their adapted versions, in the study of computer virus propagation. Similar analytical models have been proposed for botnets, after their appearance in the previous decade. The models presented in this thesis stand in this latter category, sharing many similar concepts, yet possessing some distinctive features which are not found in earlier models.

As will be explained shortly, we develop four analytical models tailored to botnets, their expansion and evolution behaviors. Each Internet node/host goes through several stages during the lifetime of the botnet. The stages, and the back-and-forth transition between them, associated with an Internet node that can join a botnet are more complex compared to those of an infected computer (node) which remains isolated. These complex node stage

characteristics lead to botnet expansion behavior that cannot be explained or predicted using the available analytical models for computer malware propagation. Further, as shown in the literature review, recent analytical botnet models have not addressed this issue adequately. In this thesis, we intend to fill this gap.

1.2.1. Stochastic vs. Deterministic Modeling

When considering the analytical models, it is important to consider that every analytical model for botnet expansion/lifecycle falls into either of the following two broad categories: deterministic and stochastic. While a deterministic model is easy to develop and analyze, it does not allow some critically important analysis permitted by a stochastic model which is relatively more difficult to construct and analyze. Specifically, the botnet population size is a stochastic process since dynamics of botnet expansion is probabilistic. In the deterministic models, the botnet population size is assumed to be a deterministic variable and the arrivals/departures to/from the population are also assumed to have deterministic values. As a result, the population size as a function of time is governed by an ordinary differential equation which is written in an ad hoc manner. The deterministic models may capture the mean population size accurately, however, this approach neither gives the distribution of the population size nor its higher moments. On the other hand, increasing the number of node stages causes a stochastic model to become intractable far more quickly in comparison to a deterministic model; therefore, when developing a stochastic model, it becomes imperative to limit the number of node stages considered.

1.3. Thesis Organization and Contributions

The contributions of this thesis can be summarized as follows:

- Analysis of botnet expansion and development of mathematical models which can estimate the botnet size at any given time.

- Evaluation of botnet mitigation strategies (attack on botnets and/or disinfection of nodes) through the developed models for botnet size estimation.

- Threat analysis of botnets in 4G cellular wireless networks.

A comprehensive literature review is provided in Chapter 2 in which a thorough background information on botnets is first presented: we start by introducing botnet architectures through some published studies focused on structural analyses and direct measurements. Next, we briefly outline botnet detection methods, countermeasures against botnets, and the methodologies used to measure the size of botnets. This is then followed by a detailed presentation and examination of some simulation and experiments regarding some botnets. Delving into the main theme of this thesis work, a literature review of existing analytical botnet expansion/lifecycle models is presented, concluding with an in-depth analysis of one of the most recent models. In what follows, the contributions are explained with more details while describing the thesis organization.

1.3.1. The SCom Botnet Models

In Chapter 3, we propose the following two Continuous-Time Markov Chain (CTMC)-based models for prediction of the botnet size in the initial phase of botnet lifecycle: SComF for the case of finite number of susceptible nodes (suitable for a botnet expanding in a closed environment such as an administrative domain, or a LAN) and SComI for the case of infinite number of susceptible nodes (suitable for a botnet expanding in the larger Internet). Having access to such models would enable security experts to have reliable size estimates and therefore be able to defend against an emerging botnet with adequate resources. We derive the probability distributions for both models and provide some numerical results as well as a simulation study accompanying the numerical analysis of the SComF model using the GTNetS network simulator.

1.3.2. The SIC Botnet Model

In Chapter 4, we introduce the *Susceptible-Infected-Connected* (SIC) botnet lifecycle model. The SIC model possesses some key improvements over earlier models: (1) keeping track of only key node stages (*Infected* and *Connected*), hence being applicable to a larger set of botnets; and (2) being a CTMC-based model, it takes into account the stochastic nature of population size evolution. The SIC model helps the security experts with the following two key analyses: (1) estimation of the global botnet size during its initial appearance based on local measurements; and (2) comparison of botnet mitigation strategies such as disinfection of nodes and attacks on botnet's Command and Control (C&C) structure. The analysis of the mitigation strategies has been strengthened by the development of an analytical link between the SIC model and the P2P botnet mitigation strategies. Specifically, one can analyze how a random sybil attack on a botnet can be fine-tuned based on the insight drawn from the use of the SIC model. We also show that derived results may be used to model the sudden growth and size fluctuations of real-world botnets.

1.3.3. The SIC-P2P Botnet Model

Among the many underlying architectures powering these overlay networks (i.e., botnets), Distributed Hash Table (DHT)-based Peer-to-Peer (P2P) botnets are perhaps the most resilient, hence the most threatening types. In Chapter 5, we propose the SIC-P2P botnet lifecycle model which is an extension of the previously-introduced SIC model. Being the first analytical model enabling such an analysis, the model allows a detailed evaluation of mitigation strategies such as index poisoning and sybil attack against DHT-based P2P botnets. The SIC-P2P model is also a CTMC-based model and as such, it properly captures the stochastic nature of population size changes. We derive closed-form expressions for the time-dependent means and variances of the number of nodes, among other analytical results, and provide a thorough numerical analysis. Until now, the insight gained from the use of the SIC-P2P model could have been obtained only through large-scale, time-consuming,

and expensive simulations and testbed experiments. The model therefore serves as a rapid, low-cost tool helping security experts evaluate and fine-tune mitigation strategies before deployment.

1.3.4. Botnets in 4G Cellular Networks

Long Term Evolution (LTE) is the main air interface technology for the 4G cellular networks. Voice over LTE is being, or scheduled to be, offered using the packet-switched technologies, rather than using the traditional circuit-switched ones [14]. Further, IP Multimedia Subsystem (IMS) is scheduled to be deployed in the core network with the LTE being the primary cellular access technology. Such a paradigm shift in offering vital telecommunication services has many technological benefits, but it also presents a host of other new challenges, the most important of which, in our view, is the maintenance of security and availability of service in the face of Internet-world security threats such as botnets. The threat of botnets and DDoS attacks due to the all-IP, non-circuit switched architecture, has been reported to be the greatest concern of telecom operators [15]. As botnets have shown to be the platform of choice to launch DDoS attacks in the Internet [16], the aforementioned concern of the operators is very well placed.

Finally, as our last contribution, in Chapter 6, we report a vulnerability of the air interface of 4G cellular networks, the LTE, to DDoS attacks launched from botnets. The attack scenario constitutes of a botmaster instructing the botnet nodes to start sending or downloading dummy data in order to overwhelm the air interface, thereby denying service for voice users. Through simulation using a capable LTE simulator, we determine the number of botnet nodes needed per cell that can effectively render the cellular network unusable.

1.3.5. Conclusions, Future Work, and Appendices

The thesis is concluded in Chapter 7 by providing a summary of the contributions, drawing conclusions, mentioning the produced publications, outlining possible future work, ending

on a note highlighting the significance of the results obtained and the overall contribution of this thesis to knowledge and practice in the domain of computer and telecommunication network security.

This document has three extensive appendices which contain the great portion of derivations of various models in order to remove the tediousness from the main text. Specifically, Appendix A contains some parts of the derivations of the SComI and SComF models as well as the code used in the simulation regarding the SComF model using the GTNetS network simulator. Finally, Appendix B and Appendix C contain some parts of the derivations of the SIC and SIC-P2P models, respectively.

2. Literature Review

In this chapter, we explore all major aspects of botnets, from their construction and architecture, to their detection and countermeasures, ending with describing ways to measure, simulate, and model them. Relevant to the P2P aspect of some of our developed models, we also introduce some analytical studies, simulations, testbed, and real-world experiments dealing with P2P botnets and their mitigation strategies.

2.1. Botnet Design, Construction and Control

2.1.1. Infection Vectors

To create a botnet, the first step is to infect the hosts (personal computers, or *nodes*). In recent major botnets, cyber criminals have been using numerous infection vectors[1] all at the same time which is a major shift from earlier one-method-only attacks (e.g., e-mail attachments). In what follows, some of the infection vectors utilized by botnets are mentioned.

Bailey et al. [17] give the SDBot botnet as an example which utilizes numerous methods such as file sharing sites, peer-to-peer networks, exploits of common Windows vulnerabilities and backdoors which have been left by previous malware. They also note the trend of gaining entry into a system from operating system and low-level services exploits to higher layer web-based methods (e.g., sites exploiting vulnerabilities in Flash player).

[1]The term "vector" is used in biology and computer anti-virus industry to refer to the carrier of virus or means of infection.

P2P file sharing has been an ideal medium to get the malware on victims' computers as it is usually difficult, if not impossible, to inspect the content of the file before its full reception from anonymous sources. Wang et al. [18] observe this exploitation in P2P networks and further note that although recent P2P botnets (to be explained shortly) have tapped into existing P2P networks to recruit nodes, in order not to be bound by the size of the networks, they have used other methods such as e-mail attachments, instant messages and file exchanging simultaneously as well.

Dittrich and Dietrich [19] have examined the Nugache botnet and observed its infection vectors to be a sophisticated mix of direct and indirect propagation methods. The direct propagation methods include: (1) exploiting vulnerabilities in two Windows services which are remotely accessible; (2) when inside a host, using the Windows Address Book to e-mail the malware automatically to the address book entries; and (3) sending instant messages (on AIM and MSN) to potential victims with the content of the message being some random sentences with a link to a site which hosts the malware and can exploit the vulnerabilities of the visiting user. The authors also note an indirect method of propagation which works as follows: Botmasters have created a "trojan horse" out of a freeware video editing application. They have altered the program to include their malware and then re-distributed the software on common freeware sites. Unaware users who download and install the video editing software get also infected as a result.

2.1.2. Construction and Command & Control

Once the malware is on the victim's computer, it tries to connect the computer (the *node*) to the botnet. *Command & Control* (*C&C*) refers to the mechanism that the botmasters use to send commands to the *nodes* in order to control them, i.e., to provide instructions for illicit activities as well as to update/manage the malware residing in the nodes.

C&C mechanism of botnets can be centralized, peer-to-peer, unstructured or a hybrid of these architectures [17, 20, 21]. We will describe the centralized botnets shortly. Peer-

to-Peer botnets and Unstructured/Hybrid botnets are treated in the next sub-sections. It is important to note that a botnet does not necessarily fall into one category or the other. For example, variants of the Conficker botnet (introduced earlier) have evolved from one design to another and the botmasters have kept all these portions operational at the same time.

Like most network technologies, botnets also started with centralized topology by having a C&C server as the focal point of the botnet. Pros of such a centralized topology include the ease and speed of sending commands to nodes. The obvious disadvantage of such a topology is the dependence of the whole botnet on the availability, reachability and performance of the C&C server. To attack a botnet, the first natural step is the takedown of the C&C server which effectively renders the botnet useless, despite the nodes remaining infected. Having a central C&C server as the origin and destination of all C&C traffic leads to higher probability of detection of the server and its subsequent takedown as well. In order to alleviate this weakness, recent centralized botnets use multiple C&C servers as backups (e.g., in the case of Conficker's centralized variant, thousands of random domain names generated daily pointing to the main servers with changing IP addresses.).

Centralized botnets have used Internet Relay Chat (IRC) and Hypertext Transfer Protocol (HTTP) protocols for their C&C traffic. Gu et al. [22] define two styles of communication which result from these protocols: (1) "push" style, using the IRC protocol, based on which commands are pushed to the nodes by the C&C server; and (2) "pull" style, using the HTTP protocol, based on which commands are pulled by the nodes from the C&C server.

Zeidanloo and Manaf [21] attribute the choice of IRC protocol (used for real-time Internet text messaging) as the C&C protocol due to possessing features such as low latency and anonymous communication, capability for group and private communication and simple setup/commands. On the other hand, the choice of HTTP is attributed to its ability to blend with other web traffic and bypass firewalls which usually look for and block the IRC traffic.

2.1.3. Peer-to-Peer Botnets

Mansfield-Devine [2] reports that the biggest shift in botnet design in recent years has been the move from centralized C&C to the distributed Peer-to-Peer (P2P) model which provides resilience against attacks and server takedowns. The prominent examples of such P2P botnets are the Storm botnet [23] and its recent improved version, the Waledac botnet [24]. The biggest advantage of P2P botnets is the fact that neutralizing some nodes (which act both as *nodes* and C&C servers) will not disrupt the botnet operation as a whole; a fact that has been shown in real life to be true by the aforementioned botnet examples.

Wang et al. [18] categorize the P2P botnets into three classes: "parasite", "leeching" and "bot-only". A *parasite* P2P botnet is built within an existing P2P network such as a P2P file sharing network. The botnet can then put in use the existing protocols in the P2P network for its C&C traffic. In a *leeching* P2P botnet, *nodes* have been infected possibly in many ways; however, they join an existing P2P network and use it for their C&C traffic. Finally, the *bot-only* P2P botnet is a botnet that is purely composed of infected *nodes* and the P2P network has been designed from scratch by the botmasters for the specific purpose of routing the C&C traffic.

Among the three classes, *parasite* P2P botnets are easier to construct, but they are less flexible as botmasters have to rely solely on existing P2P protocols. On the other hand, *bot-only* P2P botnets offer more flexibility to botmasters in terms of choices for P2P protocols. *Leeching* and *bot-only* P2P botnets are more vulnerable in the initial phases of botnet expansion as nodes are initially isolated and have to find other nodes to connect to; this process can be interfered with by an attack on the botnet. Finally, *leeching* P2P botnets are stealthier as infected nodes are mixed with other clean nodes in the P2P network.

The aforementioned "pull" and "push" C&C styles for centralized botnets also exist in P2P botnets. In the pull style, nodes actively seek to find the location where the botmaster's commands are and subsequently request the commands. In push style, nodes remain idle until commands are forwarded to them; they in turn forward the commands to other nodes.

As an example, the *pull* style is described as follows: Each piece of content in a Distributed Hash Table (DHT)-based P2P network is saved in association with a *key* which uniquely identifies the location of the content. When a node in a P2P network wants that specific content, it initiates queries for the key and the content is subsequently returned. In a P2P botnet, the keys are defined by the botmaster and all nodes know about them in one way or another (hard-coded or otherwise calculated in the malware). The botmaster stores the commands in the P2P network as content under those predefined keys. The nodes then initiate queries periodically for those keys to find and receive the commands. This mechanism allows the botmaster to remain unidentifiable, all the while capable of communicating their commands to all the nodes in the botnet.

2.1.4. Unstructured and Hybrid Peer-to-Peer Botnets

Apart from the centralized and P2P botnets, Bailey et al. [17] mention that botnets can be completely *unstructured*. Although such a botnet can be easily constructed, it has major performance problems which will likely prevent it from steady growth. In such a botnet, a node knows only about one other node to which it can pass along the commands. Botmaster, and nodes, would need to randomly scan the Internet to find other botnet nodes which leads to high latency in command propagation and subsequent low performance of the botnet. The only advantage of such botnets is that they are extremely resilient to attacks, as the takedown of one node has no effect on the botnet as a whole.

Researchers have recently tried to "outpace" the botmasters by proposing novel botnet designs, implementing both "push" and "pull" styles. They have then analyzed how such botnets can be attacked, in the hope of being prepared for future botnets. Wang et al. [25] have proposed the "Hybrid P2P botnet" in which the existing P2P protocols and architectures have not been reused. In the Hybrid P2P botnet, nodes can forward the commands to their peers as well as proactively query their peers to receive commands. On the other hand, Vogt et al. [26] have proposed the "Super Botnet" in which the botnet is composed

of several smaller centralized botnets, each of which possessing a C&C server that makes the commands available to the nodes using the "pull" mechanism. Further, commands can be "pushed" from one small botnet to the other within the larger botnet. Although *Hybrid P2P botnet* and *Super Botnet* have interesting designs, their performance and effectiveness are unknown in the real world as they are purely hypothetical botnets.

2.2. Botnet Detection

To attack botnets and reduce their threat, we need effective methods to detect them. Detection of botnet's C&C traffic is far from straightforward, as botmasters continuously evolve botnet designs which renders existing detection methods useless. In what follows, we take a brief look at recently proposed methods for botnet detection.

Bailey et al. [17] note the types (or bases) of detection methods as: "signatures", "cooperative behaviors" and "attack behaviors". Wang et al. [27], however, refer to the methods based on the first type as "static" methods and to the ones based on the last two types as "dynamic" methods.

In the category of signature methods, Binkley and Singh [28] have proposed the correlation of the following two sets of data: (1) IP addresses of the hosts that are seen in an IRC channel; and (2) IP addresses from which Internet scanning have been conducted. Common IP addresses between these two sets of data are high-probability candidates for being part of an IRC-based botnet. While maybe effective in some instances, signature-based detection methods are now largely unsuccessful due to the recent use of encryption for C&C traffic.

In the category of "cooperative behavior" methods, three prominent examples are BotHunter [29], BotSniffer [22] and BotMiner [30]. BotHunter [29] identifies a suspected infection event by correlating data from multiple detection systems and comparing the result against known malware propagation processes. BotSniffer [22], and its improved version,

BotMiner [30], have the same foundation: both analyze crowd-like behaviors in the traffic patterns and, in the case of BotMiner, a correlation with the traffic of detected malicious activities is done which results in identification of botnet members. Besides these three examples, methods described by Wang et al. [27] as "anomaly detection" methods roughly fall under this category as well. These latter methods include the one proposed by Gianvecchio et al. [31] which analyzes Internet chat patterns to classify human and bots as well as another one described by Wang et al. [27] themselves which analyzes the P2P traffic in search of peers which initiate periodic queries for the same keys.

Finally, as an "attack behavior"-based method, Brodsky and Brodsky [32] have based their proposal on the simple assumption that botnets tend to do their malicious activity, e.g., sending e-mail spams, in relatively short amount of time. The traffic analysis taking this assumption into account would lead to the detection of the botnet. Like most other detection methods however, this method is not too difficult to evade, e.g., by introducing some randomness in the timing of the activities.

2.3. Countermeasures Against Botnets

After having mentioned some of the detection methods as well as their general categories, in what follows, some methods for attacking botnets (botnet mitigation strategies) are described. Before proceeding further, however, it is important to point out that the ethical aspect of the research on botnets is an important one and has been explored by Dittrich et al. [33]. They note the difficulties of combating botnets in real life as boundaries blur and researchers frequently find themselves having control or needing to take control of machines owned by private citizens and corporations located across many jurisdictions.

Researchers have proposed many methods to attack botnets. However, in the case of non-P2P-based botnets, these methods are largely ad hoc measures which are effective only against the specific botnet under consideration. For P2P botnets, on the other hand,

there are three generally applicable attack methods as follows: (1) *Index poisoning*; (2) *Sybil attack*; and (3) *Eclipse attack*. These methods, however, were originally created and intended for normal P2P networks, as P2P botnets were not around yet. They were, and still are, largely used by corporations fighting illegal distribution of copyrighted content on P2P file sharing networks.

2.3.1. DHT-based P2P Botnets

In a Distributed Hash Table (DHT)-based P2P botnet, nodes find each other, construct their routing tables, and relay the traffic to, or closer to, its intended destination based on normal DHT methods. In a DHT-based P2P botnet like the Storm botnet, with a Kademlia-based [34] DHT implementation, each botnet node has an ID. Each content to be stored in a node on the network gets a unique hash key[2] with the same size (number of bits) as the node IDs; the content is therefore denoted as the pair <hash key, content>. When deciding where to store the content on the network, a distance is calculated between the hash key of the content and the node IDs using the XOR operation. The content is therefore stored on the node whose ID is closest to the hash key. In order to find (lookup) a particular content using its hash key, a search is launched to find nodes whose IDs are close to the hash key of the content; these nodes likely have the content.

The botmaster also relies on the aforementioned methods for the C&C of the botnet; therefore, the decreased efficiency of the C&C mechanism as a result of the attacks translates into an inefficient botnet. In what follows, we explain the aforementioned attack methods briefly using the terminology relevant to P2P botnets.

2.3.2. Index Poisoning, Sybil, and Eclipse Attacks

Index poisoning, introduced by Liang et al. [35], is used for "poisoning" the *index* (address space) of a DHT-based P2P botnets. This method entails injecting bogus content (to replace

[2] *Index*, *key* and *hash key* are terms that are frequently used interchangeably.

the original content which is botmaster's commands) under the same *keys* associated with the original content. After the poisoning, when nodes (bots) of the botnet try to locate the content, they will likely receive the bogus content rather than the botmaster's commands. This leads to the disruption of the C&C mechanism of the botnet.

Sybil attack, first presented by Douceur [36], is an attack method under which numerous *clean* nodes (sybils) are injected into the P2P botnet, posing themselves as "legitimate" botnet nodes. They then try to re-route, block and corrupt the normal C&C traffic flowing in the P2P network, thereby lowering the efficiency of the C&C mechanism of the botnet.

First introduced by Castro et al. [37], an *Eclipse attack* can be carried out by injection of numerous *clean* nodes into the P2P botnet. However, this time, the clean nodes try to surround, or *eclipse*, one or more botnet nodes, thereby controlling all incoming and outgoing C&C traffic of the eclipsed nodes. This method leads to the eclipsed botnet nodes effectively becoming disconnected from the botnet.

Using simulations, Davis et al. [38, 39] have shown the effectiveness of sybil attacks on P2P botnets. On the other hand, Ha et al. [40] have shown that Eclipse attacks are not as effective as index poisoning or sybil attacks against P2P botnets. It is important however to note the limitation of the carried out simulations which is the fact that they have been application-layer-only simulations without realistic, or near-realistic, traffic mix. Considering the simulations carried out across several studies, one can infer that the simulation results obtained are usually open to debate and inconclusive, as simulations of botnets, including attack on botnets, are extremely burdensome and need simplifications on many levels.

2.3.3. Index Poisoning and Sybil Attack: An Analytical View

Later in this thesis, we will use some analytical results with regard to the effect of index poisoning and sybil attack on botnets. As we have just given an overview of these attack methods, it is appropriate to follow up the overview with some analytical results presented

in [18] which will be used later on in Chapter 5.

Wang et al. [18] have provided an analysis which allows the determination of P_s, the probability that a botnet node obtains a real command, based on various parameters of the index poisoning and sybil attacks; these parameters as well as related botnet variables are as follows:

P_s probability that a botnet node obtains a real command.

n botnet size.

n_p number of nodes poisoned in the target zone (target zone is the address space close to the hash key generated by the botmaster from its content, i.e., the botnet's C&C commands).

n_s number of sybil nodes inserted randomly in the network, in the case of random sybil attack, or inserted in the target zone, in the case of targeted sybil attack.

c number of first bits in common with a hash key (used to determine the size of target zone).

n_{tz} number of nodes in the target zone which is determined by n and c as follows: $n_{tz} = \frac{n}{2^c}$.

b number of bits improved per step (i.e., getting closer to the destination) for a lookup for a hash key.

l_{tz} number of steps required in the lookup process to get to the destination within the target zone. The mean value of l_{tz} is $\frac{\text{Log}_2[n_{tz}]}{b}$.

The attacks constitute of intercepting the lookup query made by a botnet node to reach the destination and returning bogus content, instead of the real botmaster C&C commands, to the initiator of the query. To carry out the attacks, first a target zone around the destination is set up with a "radius" that is determined by c. The target zone must be big enough

that at least one step of the lookup process falls within it before reaching the destination. Let x be the probability that a poisoned node (or sybil node) is chosen in a step within the target zone. For the lookup process to be successful, i.e., reaching the destination containing real botmaster C&C commands, no poisoned node (or sybil node) should be chosen in the steps leading to the destination. Therefore, $P_s = (1-x)^{l_{tz}}$.

For an index poisoning attack, $x = \frac{n_p}{n_{tz}}$. With $l_{tz} = \frac{\text{Log}_2[n_{tz}]}{b}$ and $n_{tz} = \frac{n}{2^c}$, the obtained formula is the following:

$$P_s = \left(1 - \frac{2^c n_p}{n}\right)^{\frac{\text{Log}_2[n]-c}{b}} \tag{2.3.1}$$

On the other hand, for the targeted sybil attack, $x = \frac{n_s}{n_s+n_{tz}}$. With $l_{tz} = \frac{\text{Log}_2[n_s+n_{tz}]}{b}$ and $n_{tz} = \frac{n}{2^c}$, the obtained formula is the following:

$$P_s = \left(1 - \frac{n_s}{n_s + \frac{n}{2^c}}\right)^{\frac{\text{Log}_2[n_s+\frac{n}{2^c}]}{b}} \tag{2.3.2}$$

Finally, for the random sybil attack, the target zone is simply the whole network, i.e., $n_{tz} = n$. Therefore, $x = \frac{n_s}{n_s+n}$. With $l_{tz} = \frac{\text{Log}_2[n_s+n]}{b}$, the obtained formula is the following:

$$P_s = \left(1 - \frac{n_s}{n_s + n}\right)^{\frac{\text{Log}_2[n_s+n]}{b}} \tag{2.3.3}$$

2.4. Botnet Measurements

Responding to the growing threat of botnets in the past few years, researchers have paid much attention to measuring the size of botnets in order to determine their threat level. In what follows, we provide a brief overview of these efforts and will conclude the section with some notes regarding the effectiveness and relevance of such measurement studies.

Zhu et al. [41] mention the measurement techniques applicable to centralized, mostly IRC-based, botnets. They note the use of Honeynets [42], in which Honeypots run unpatched (i.e., still vulnerable) versions of Windows operating system in virtualized envi-

ronments. The Honeypots serve to gather data on botnets as botmasters, and the propagating malware, cannot distinguish them from other normal machines[3]. Another reported method is the use of a "drone" to join the IRC server/channel used by the botnet and record the information regarding bot joining, etc. Finally, manipulation of Domain Name System (DNS) record of the C&C server (e.g., IRC server) and redirection of connection requests is another method used to gather data on botnets.

Advent of P2P botnets have further complicated the measurement efforts of the research community, as there is a lack of a central measurement point in such botnets. The commonly used method for measuring the size of a P2P network is the use of a "crawler" which uses "get peers list" type commands to learn about the peers/nodes in the P2P network. However, Kang et al. [43] have shown that such a method misses a large portion of the population, i.e., peers who are behind firewalls and Network Address Translation (NAT) devices. They have proposed a method called Passive P2P Monitor (PPM) which is used for the enumeration of Storm botnet nodes and is based on injection of numerous "routing only" nodes into the P2P botnet; these nodes gather data from the normal C&C traffic flowing in the P2P network.

As the number of measurement studies on botnets has grown, so has the number of discrepancies in the reported results. Rajab et al. [44] attribute the inconsistencies to the used counting techniques and mention "cloning", "temporary migration", and "hidden structures" as issues affecting the accuracy of the size measurements. On the other hand, Kanich et al. [45] attribute such discrepancies to: (1) "temporal dynamics" such as address space reuse; (2) "address aliasing" due to, for example, the presence of NAT devices; (3) activities of other participants such as other researchers, botmaster's competitors, etc.; and (4) bots piggybacking on existing protocols. Finally, regarding botnet size measurement results, Rajab et al. [44] state: "*we hear widely diverging answers. In fact, some may argue, contradictory.*" On the same topic, Kanich et al. [45] state: "*our community should have*

[3]Honeynets can be used to study any type of botnet/malware; their use is not limited to centralized botnets.

considerable skepticism about the veracity of botnet measurement results going forwards."

2.5. General P2P Botnet Simulations

As botnets which have a P2P structure are the focus of some of the work done in this thesis, in this section, we describe some of the important simulation models developed for such botnets.

Using the stochastic Monte Carlo simulation, Wang et al. [46] investigated P2P botnet topologies and studied botnet size and number of peers under worm infection and countermeasures. The study therefore provides some insight with regard to the robustness and effectiveness of the P2P botnet under analysis. The study is limited due to the used simulation environment and as such, it lacks formulas to examine the botnet size, which in general limits any botnet analysis by a third party.

The population size of the Storm botnet [47] has been studied by Ruitenbeek and Sanders [48] through simulation of a Stochastic Activity Network (SAN) model (a variant of stochastic Petri nets). The SAN model and its parameters have been loosely based on the information gathered on the Storm botnet. The SAN models the lifecycle of a node with four stages: *Susceptible*, *InitialBotInfection*, *ConnectedBot*, and *FullyConnectedBot*. It is assumed that the number of nodes in the *Susceptible* stage is *infinite* and the time interval for a node to move from one stage to the next one in the last three stages are exponentially distributed with different parameters (this effectively makes the model very close to an analytical CTMC model). It has been also assumed that the move of a node between stages succeeds with certain probability and unsuccessful nodes are removed from the experiment. Success probabilities may be used to account for the impact of mitigation strategies on the growth of the botnet. The paper presents simulation results for the mean population size of nodes in *FullyConnectedBot* stage as a function of time for different success probabilities between stages. It may be seen that when success probability is one, the botnet grows

exponentially.

As an analytical model further exploring the aforementioned simulation model [48], Kolesnichenko et al. [49] develop a set of ordinary differential equations (ODEs) using mean field theory. This set of ODEs captures the transition of nodes from one stage to another. The numerical solution of this set of ODEs are presented and the focus of the paper is on showing that mean field theory produces the same results much faster compared to the simulations conducted in [48]. Further, this study too shows exponential growth of the number of nodes, as in [48]. The limitation of the method, however, is that the stochastic analysis of the model is not possible as variances/standard deviations cannot be derived using this deterministic approximation.

2.6. DHT-based P2P Botnets and Their Mitigation Strategies: Simulations and Experiments

In this section, we present some simulation models and experiments that focus on DHT details and evaluate attacks such as index poisoning and sybil attack.

Sybil attack against the Storm botnet has been studied using graph theory-based simulations by Davis et al. [38]. The study reveals information such as: (1) the effect of the ratio between the number of sybils and the botnet's growth rate on the level of the disruption of the botnet; (2) how the duration of the sybil attack could affect its effectiveness; and (3) the precise botnet design choices, such as the size of the peer-list of each botnet node, do not seem to have much effect on the sybil attack's effectiveness. The study concludes on the note that packet-level simulations, with more realistic background traffic, delays, and network size are needed to arrive at more definitive conclusions. The authors have extended their work in [39] using a refined graph model that captures more accurately the P2P botnet's search query mechanism. Using the new model, they have then determined that slower-rate sybil injection with random placement of sybils is nearly as effective as

higher-rate injection with targeted placement. Finally, the authors point out that the results are not confined to the Storm botnet and should be applicable to all P2P botnets sharing Storm's core operating principals.

A study of the structure of P2P networks running Kademlia, one of few P2P protocols widely used in practice, has been reported by Ha et al. [40]. The developed simulation testbed incorporates the actual code of a real Kademlia client software to increase the realism of the simulation environment. Many P2P network parameters from a graph-theoretical perspective have been studied using the testbed. They have also investigated countermeasures such as index poisoning and sybil attack. One notable reported result is that, in sybil attack, it is more effective to spread out the IDs (keys) of the sybils than to inject more sybils in a concentrated fashion. With regard to the built testbed, they acknowledge that simulation of a botnet at a realistic scale is computationally prohibitive and cannot be achieved with a single PC within a reasonable amount of time; the built testbed is therefore a 30-machine installation which runs the distributed simulation. However, even with this powerful testbed, in order to have the desired scalability, they had to limit the simulation to IP layer and above and ignore background traffic and IP routing protocols. As the authors of the aforementioned papers [38, 39] pointed out, these simplifications can potentially influence significantly the outcomes, as peers are sensitive towards the propagation delays which determine from which types of nodes (legitimate, poisoned, or sybil) commands are received first.

Apart from the aforementioned simulations and experiment which were examined in depth as examples, there are few other relevant works which are introduced briefly as follows: Holz et al. [23] reported the first instance of attacking a real-world P2P botnet, the Storm botnet. The authors report on having used a sybil attack to infiltrate the botnet in order to enumerate the number of botnet nodes. Further, they report on having successfully executed an index poisoning attack which has disrupted the C&C mechanism of the botnet. A hypothetical P2P botnet with improvements such as having reputation and proof-of-work

systems was simulated and the effectiveness of a sybil attack against the botnet was studied by White et al. [50]. Reputation systems introduce a metric of trust between peers in the botnet and proof-of-work systems are cryptographic puzzles that are time consuming for a client (a sybil that wants to join) to solve, but are quickly verifiable by the server (the targeted botnet node). They found that the reputation system made the botnet more vulnerable to the sybil attack, contrary to the proof-of-work system which made the botnet stronger. They consider the simulation of layers below the application layer unnecessary which might be a point of concern, considering the conclusions made by an aforementioned study [38]. In terms of simulation run time, they indicate that a simulation would take seconds in the case of 1,000 botnet nodes, but that it would grow somewhat exponentially, for example to more than a day in the case of 20,000 botnet nodes. Finally, Calvet et al. [51] reported an at-scale emulation of the Waledac botnet (a recent P2P botnet) in laboratory conditions. The emulated botnet consists of approximately 3,000 nodes and has been made possible using an equivalent of 98 powerful PCs and about 30 virtual machines per PC. Using the testbed, they report the effectiveness of the sybil attack against this P2P botnet by measuring parameters such as e-mail spam output and the connectivity of the botnet. Some of experiment's results, however, are sensitive to how much resource is given to the C&C server which remains an unknown. Further, the authors point out the challenge in the experiment methodology which is the simulation of network characteristics and user behavior and mention that the emulation of the operating environment of the botnet is somewhat simplistic.

While experiments are generally known to produce the most realistic results, the scale of the operations is usually prohibitively large. On the other hand, simulation models may be designed to capture more of the real-world details compared to what analytical models can do; however, the computational cost is generally too high. This is why we have embarked on this research work and developed some analytical botnet models which are presented in the following chapters.

2.7. Analytical Botnet Models

Abstracting away the name of *actors* in the *system* to be modeled, developing analytical models for spread of computer virus, expansion of botnets and disease spread (biology) are similar problems. In the past two decades, researchers have adapted the analytical results from epidemiology [52] to malware propagation and, recently, to botnet lifecycle modeling. We limit, however, the overview in this section to studies regarding botnet population/lifecycle modeling to ensure that the models can be reasonably compared to one another. The only exception would be the first major analytical model for the spread of computer virus [11] which is presented next due to its historical significance.

2.7.1. Analytical Models: The First Major Work

In the development of an analytical model for computer virus propagation, Kephart and White [11](1991) used the common epidemiological approach by ignoring the details of infection inside a single machine *(node)* and considered the node to be in one of few *stages*, e.g., *Infected*, *Uninfected*, *Immune*, etc. Further, another simplification common in epidemiology is the abstraction of the details of viral transmission. A probability per unit time is therefore used based on which an infected node will infect an uninfected/susceptible node (and the same for disinfection). These assumptions and modeling techniques are employed in all subsequent models introduced by other researchers as well as in our own work.

They have formulated a *directed random graph* model in order to study the virus propagation between nodes. An individual machine is a *node* in the graph with directed edges from a given node j to all other nodes that can be infected by the node j. An infection rate is associated with each *edge* and a disinfection rate is associated with each *node*. The model that is considered in this study is the classical epidemiological Susceptible-Infected-Susceptible (SIS) model, i.e., *nodes* can be in either of two stages, *Susceptible* or *Infected* and they can go back and forth between these stages. The authors then proceed to derive

the mean number of infections as a function of time from a deterministic approximation of the described graph.

A random graph of N nodes is constructed with probabilistic decisions regarding the inclusion of edges which can be up to $N(N-1)$. The mean number of edges is therefore $pN(N-1)$, with p being the probability of edge inclusion. Each edge has an associated infection rate β_{jk} (between node j and node k) and the disinfection rate from each node j is δ_j. The deterministic approximation is as follows: they consider all infection and disinfection rates to be identical, ignore the details of graph connectivity (number of nodes is therefore the important value rather than which nodes) and ignore the variation in the number of edges emanating from each node. As a result, the infection rate along each edge is denoted by β and the disinfection rate from each node by δ. Finally, considering N to be large, a continuous quantity i is defined to represent the fraction of *Infected* nodes, i.e., $i \equiv I/N$ (both i and I are functions of time).

As the mean number of edges in the graph is $pN(N-1)$, the mean number of edges emanating from a given *Infected* node is $\bar{b} = p(N-1)$. As the fraction of neighboring nodes that are *Susceptible* is $1-i$, the mean number of *Susceptible* nodes that can be infected by this given *Infected* node is $\bar{b}(1-i)$. They further define β', which is equal to $\beta\bar{b}$, to be the average total rate at which a node attempts to infect its neighboring nodes . The average system-wide rate at which *Infected* nodes infect their neighboring *Susceptible* nodes is therefore $\beta' I(1-i)$. On the other hand, δI is the system-wide rate at which *Infected* nodes are disinfected. The deterministic differential equation describing the time evolution of i is therefore as follows: $\frac{di}{dt} = \beta' i(1-i) - \delta i$. The authors then go on to analyze this derived formula and discuss different scenarios with regard to the possibility of an epidemic.

In a stochastic treatment of the same graph, on top of the approximations made above to derive the deterministic case, they assume that the various probabilities are independent (i.e., there is no correlation between the probability that a node is infected and the probability that its neighbors are infected). In this case, the main resulting equation is a probability

flow equation which is as follows:

$$\frac{dp(I,t)}{dt} = -p(I,t)[I(1-i)\beta' + \delta I] + p(I_+,t)I_+\delta + p(I_-,t)[I_-(1-i_-)\beta'] \qquad (2.7.1)$$

where, $p(I,t)$ denotes the probability distribution of I infected individuals at time t, $I_+ \equiv I+1$, and $I_- \equiv I-1$. For a graph with N nodes, this is a set of $N+1$ coupled linear differential equations which have been numerically solved. While many cases have been discussed and derived, a general probability distribution for the number of infected nodes as a function of time has not been presented. On the other hand, the model/study, has focused on local methods for the spreading of computer virus (namely, computer program sharing with other individuals), as the Internet, with its current form, capabilities, and popularity, appeared much later. Finally, the model studied is, as mentioned, the SIS model, i.e., only two node stages are considered. We will see later on that capturing the full dynamics of botnet size fluctuations will need three node stages.

2.7.2. Stochastic Models

A probability model to estimate the number of machines infected per hour with the Conficker-C botnet has been presented by Weaver [53]; the work includes derivation of the distribution of the number of hourly UDP connection attempts made by an infected host and the conditional distribution of the number of observed hits in the monitored IP space. The model considers the Conficker botnet to be a worm and proposes a one-stage only (Infected stage) model. To model a botnet, however, apart from the healthy/susceptible node stage, one should consider at least two node stages (Infected nodes as well as nodes which subsequently manage to connect to the botnet and receive the C&C traffic). In this sense, the presented model can only partially capture and model the botnet and its real firepower.

Li et al. [54] have introduced "genetic mechanism" as the topology construction mechanism of botnets. Through this modeling method, they study "in-degree distribution", short-

est distance, and clustering coefficient of the constructed topology. The study, however, lacks results regarding botnet size and various parameters thereof.

Spread of botnets in social networks has been studied by Li et al. [55]. The model can predict the size of the botnet at any time with the knowledge of social networks' topology and the initial number of infected nodes. The model contains parameters for characterizing user reading habits and the relationship between nodes in the propagation process. Specifically, the probability $P_{i,t}$ has been defined which is the probability that node i is infected at time t and has the following relationship: $P_{i,t} \approx P_{i,t-1} + \sum_{j \neq i} \alpha_{ji} P_{j,t-tw_i}$; where, tw_i is a waiting time of a user who is reacting to a received malware and α_{ji} expresses the relationship that exists between node j and node i. The paper does not give details about how α_{ji} parameters are in fact calculated. From the probabilities, an average of number of infected nodes is then calculated. This developed model is therefore a stochastic analytical model which has been solved and evaluated numerically using the MATLAB package. It seems that the accuracy of the results depends heavily on precise knowledge about multitude of factors regarding social networks which are generally unavailable. Besides the lack of details about some important parts of the model, there is an inherent shortcoming of the modeling approach: botmasters are known to utilize many infection vectors, including social networks, at the same time to maximize the effectiveness of their recruiting campaigns. This model, therefore, may ultimately reveal partial information about a botnet.

Finally, Wang et al. [56] presented a model of worm's propagation probability in a P2P overlay network using a fully-connected graph. This model is limited to small networks, however, as having a square matrix of dimension n, with n being the number of nodes in the network, to define and examine the network topology and botnet size leads to the model being unusable for Internet-scale scenarios.

2.7.3. Deterministic Models

Inspired by epidemic models, there have been several deterministic models proposed in recent years [57, 58, 59, 60, 1] based on ordinary differential equations describing the flow of nodes from one stage to another; these are briefly described as follows: Zou and Cunningham [57] presented a model for the growth of a P2P botnet which is dependent on the number of target hosts that can be infected at any one time. Dagon et al. [58] extended the classic Susceptible-Infectious-Removed (SIR) model by taking into account the diurnal pattern, i.e., the effect of time zones in malware propagation. Using the domain name redirection technique to gather data on the Conficker botnet, Li et al. [59] customized the SIR epidemic model. Xin-liang et al. [60], on the other hand, analyzed the relationship between the number of infected hosts and propagation ratio based on the SIR model, drawing an insight regarding the effects of different propagation ratios on botnet scale and stability.

Detailed Analysis of a Recent Botnet Model: Building on the previous works in theoretical biology and the domain of malware/virus propagation modeling, Ajelli et al. [1](2010) have recently proposed an analytical model which is based on the same premises as the models explained earlier. They have proposed two models in their paper: (1) a model in which *Infected/Active* nodes will never transition back to *Susceptible* stage; and (2) another model, which we consider to be more complete and accurate, in which *Infected/Active* nodes, when disinfected, can be re-infected again; therefore, there is a possible transition from *Infected/Active* stage to *Susceptible* stage. We will describe the second model as follows: the model, like most previous works, is a deterministic model based on ordinary differential equations describing the flow of nodes from one stage to another. With N denoting the total population size of nodes, these are the four stages of nodes: (1) S stage: susceptible nodes that can become infected; (2) I stage: infectious nodes that can infect the susceptible nodes; (3) V stage: infectious nodes that can infect the susceptible nodes on top of being active in botnet's illicit activities (nodes autonomously and probabilistically

Notation	Description
N	total population (nodes in the system)
μ	switching rate between hidden and active
s, S	proportion/number of at–risk–of–infection nodes
i, I	proportion/number of (hidden) infectious nodes
v, V	proportion/number of infectious and spamming nodes
R	number of definitely recovered nodes
β, b	normalized, absolute worm transmission rate
γ, g	normalized, absolute (definitive) recovery rate
p	apportioning coefficient of infected hidden nodes
ρ	rate of temporary recovery

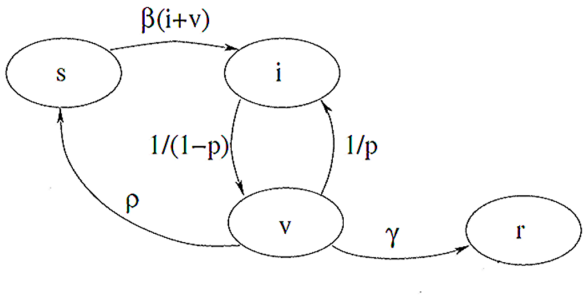


Figure 2.7.1.: Botnet model by Ajelli et al. [1]: model parameters and variables (left) and flow among stages (right). Adapted from [1].

change stage between V and I); and (4) R stage: removed/disinfected nodes that remain immune to all future infection vectors utilized by the botmasters.

The epidemic flow among these stages is depicted in Figure 2.7.1. The same figure also describes all the model variables and parameters. As it is clear from the description of node stages, *Infected* nodes are in one of two possible stages. A fraction p of Infected nodes are in I stage and the rest, fraction $(1-p)$, are in V stage. There is a flow between these two stages $I \rightarrow V$ and $V \rightarrow I$, with normalized rates of $\frac{1}{1-p}$ and $\frac{1}{p}$, respectively. The actual rates of transition between stage I and stage V depend on how often the nodes autonomously decide to change stage between these two stages, with the stage switching rate μ. This means that, on average, the time spent by a node in stage I and stage V is $\frac{1-p}{\mu}$ and $\frac{p}{\mu}$, respectively. Finally, the nodes in stage V can either transition to stage R with the rate γ or transition back to stage S with the rate ρ. Time(τ), all rates and variables are normalized with respect to the switching rate μ and the population size N as follows: $t \equiv \mu\tau$, $s \equiv \frac{S}{N}$, $i \equiv \frac{I}{N}$, $v \equiv \frac{V}{N}$, $\beta \equiv \frac{b}{\mu}$ and $\gamma \equiv \frac{g}{\mu}$. Finally, $N = S(\tau) + I(\tau) + V(\tau) + R(\tau)\ \forall\tau$. The system of equations describing the (normalized) mean values of nodes in each stage is as follows:

$$\begin{cases} \frac{ds(t)}{dt} = & -\beta[i(t)+v(t)]s(t) + \rho v(t) \\ \frac{di(t)}{dt} = & \beta[i(t)+v(t)]s(t) - \frac{1}{1-p}i(t) + \frac{1}{p}v(t) \\ \frac{dv(t)}{dt} = & \frac{1}{1-p}i(t) - \left(\frac{1}{p}+\rho+\gamma\right)v(t) \end{cases}$$

The authors then proceed to present some figures regarding the evolution of variable values, focusing in each case on changing a specific parameter. The above work has several limitations: (1) this is a deterministic model and does not account for the stochastic nature of botnet node population changes; (2) the analysis only leads to the mean number of nodes in different stages of node lifecycle and higher moments cannot be obtained. Further, the results may only be calculated numerically and no closed-form results are obtained for the mean values; (3) in the model, new infections depend on the number of nodes in *Infected* stages (I and V), which is not usually the case in botnets (*Infected* nodes not yet part of the botnet are generally not able to cause new infections, e.g., see [47]); and (4) the developed model embeds the potential characteristics of a botnet used solely for spamming purposes, as I and V stages do not carry much meaning beyond spamming roles; for example, a botmaster launching a DDoS attack instructs all, or most, nodes to "wake up" and attack and nodes do not choose "autonomously" to go to sleep during such operation.

In this chapter, we have provided a literature review of all aspects of research regarding botnets. In the last section, we focused on analytical botnet models and examined in-depth some important works done so far. We have then observed the need to improve the state of the art in this domain. As presented in the following chapters, we have attended to this identified need and have developed analytical models that capture more accurately the main characteristics of botnet population size fluctuations, namely, their stochastic nature as well as dependency on proper selection of node stages and inter-stage transition rates. In the next chapter, we start by presenting two relatively simple stochastic models that are especially suited to predict botnet population size at the onset of botnet expansion. In later chapters, we then move on to more comprehensive, albeit more complicated, analytical botnet models.

3. The SCom Botnet Models

3.1. Introduction

In this chapter, we present two Continuous-Time Markov Chain (CTMC) models of botnet expansion. CTMC models take into account *stochastic* population size changes and the appropriateness of their use has been confirmed [61] for malware propagation which happens under the influence of the same physical processes affecting botnets. Each dimension in the CTMC models represents a node *stage*, with the considered stages being *Susceptible* (i.e., susceptible to be compromised) and *Compromised* (i.e, *Infected* and *Connected* to the botnet). As botmasters use a plethora of methods to infect the nodes, it is reasonable to assume that a node is never in Immune/Removed stage. Further, we do not track the number of *Infected-only* nodes; limiting the number of stages allows the development of tractable CTMC models.

We first model the unhindered growth of botnet when the population size is *infinite*. An infinite population size is a realistic assumption considering the total number of devices that are connected to the Internet today. As we consider **S**usceptible and **Com**promised stages and the population size to be **I**nfinite, we name the model **SComI**. Next, we model the unhindered growth of botnet when the population size is *finite*. The assumption of finite population size makes the model more suitable in case a segment of Internet or a local/wide area network is the environment in which the botnet can expand. As we consider **S**usceptible and **Com**promised stages and the population size to be **F**inite, we name the

model **SComF**.

The chapter is organized as follows: Sections 3.2 and 3.3 present the model, the probability distribution derivation, and some numerical results regarding the SComI model and the SComF model, respectively. Finally, we present the simulation study in Section 3.4.

3.2. SComI: Unhindered Botnet Expansion Model - Infinite Population Size

In this section, we model unhindered growth of the botnet. In this model, a node is either in *Susceptible* stage or *Compromised* stage. Since we assume that the number of nodes in the *Susceptible* stage is infinite, we need to keep track of only the number of nodes in the *Compromised* stage. We therefore define the state of the system to be the number of nodes that are in the botnet (nodes in *Compromised* stage). Our development leads to a solution for the time-dependent probability distribution of the number of nodes in the botnet.

3.2.1. State-transition-rate Diagram

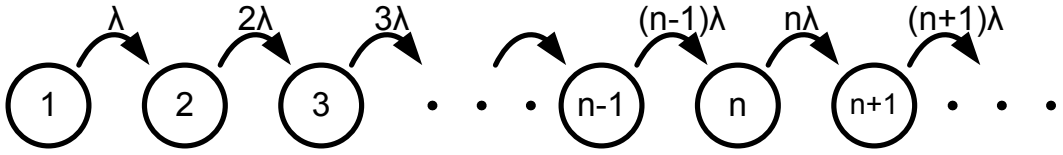


Figure 3.2.1.: SComI botnet model: 1-dimensional CTMC

The state-transition-rate diagram for the SComI model is depicted in Fig. 3.2.1. As initial condition, we assume that the size of the botnet is one. In this model, we consider that each node in the botnet recruits one node (grows the size of the botnet by one) with probability $\lambda\Delta t + o(\Delta t)$ in any Δt interval.

3.2.2. Probability Distribution Derivation

3.2.2.1. Differential-Difference Equations

We let $P_n(t)$ denote the probability that the state of the system will be n at time t. For this pure-birth process, the rate of change of probability at any state is determined by setting it equal to the difference of probability flows into and out of that state as follows:

$$\frac{dP_n(t)}{dt} = (n-1)\lambda P_{n-1}(t) - n\lambda P_n(t) \qquad n \geq 1 \tag{3.2.1}$$

The initial condition is $P_1(0) = 1$.

3.2.2.2. Probability Generating Function

To determine $P_n(t)$, the probability distribution, we first derive the corresponding Probability Generating Function (PGF). For that, we need to start from the aforementioned differential-difference equation. The relationship between $P(z,t)$, the PGF, and $P_n(t)$, the probability distribution is $P(z,t) = \sum_{n=0}^{\infty} P_n(t) z^n$. We can also write $\frac{\partial P(z,t)}{\partial t} = \sum_{n=0}^{\infty} \frac{dP_n(t)}{dt} z^n$ and the initial condition in terms of PGF as $P(z,0) = z$.

We multiply both sides of (3.2.1) by z^n and sum over n to yield:

$$\begin{aligned}
\frac{\partial P(z,t)}{\partial t} &= \sum_{n=1}^{\infty} [(n-1)\lambda P_{n-1}(t) z^n - n\lambda P_n(t) z^n] \\
&= \lambda z^2 \sum_{n=2}^{\infty} (n-1) P_{n-1}(t) z^{n-2} - \lambda z \sum_{n=1}^{\infty} n P_n(t) z^{n-1} \\
&= \lambda z^2 \frac{\partial P(z,t)}{\partial z} - \lambda z \frac{\partial P(z,t)}{\partial z}
\end{aligned}$$

We therefore have:

$$\frac{\partial P(z,t)}{\partial t} + \lambda z(1-z) \frac{\partial P(z,t)}{\partial z} = 0 \tag{3.2.2}$$

We need to solve this first-order Partial Differential Equation (PDE) in order to derive

$P(z,t)$. We use the Method of Characteristics [62] to solve this PDE as detailed in the appendix, Section A.1, which results in the following $P(z,t)$:

$$P(z,t) = \frac{ze^{-\lambda t}}{1-z+ze^{-\lambda t}} \tag{3.2.3}$$

3.2.2.3. Probability Distribution

To get the probability distribution $P_n(t)$, we need to determine the inverse of the PGF in (3.2.3). We use the transform property $A\alpha^n \Leftrightarrow \frac{A}{1-\alpha z}$ [63, p.331] to obtain the following:

$$e^{-\lambda t}(1-e^{-\lambda t})^n \Leftrightarrow \frac{e^{-\lambda t}}{1-(1-e^{-\lambda t})z}$$

And using the property $f_{n-k} \Leftrightarrow z^k F(z)$ (for $k > 0$) [63, p.330], we derive $P_n(t)$, the probability distribution of the number of nodes in the botnet at time t, as follows[1]:

$$P_n(t) = e^{-\lambda t}(1-e^{-\lambda t})^{n-1} \qquad n \geq 1 \tag{3.2.4}$$

3.2.3. Numerical Analysis

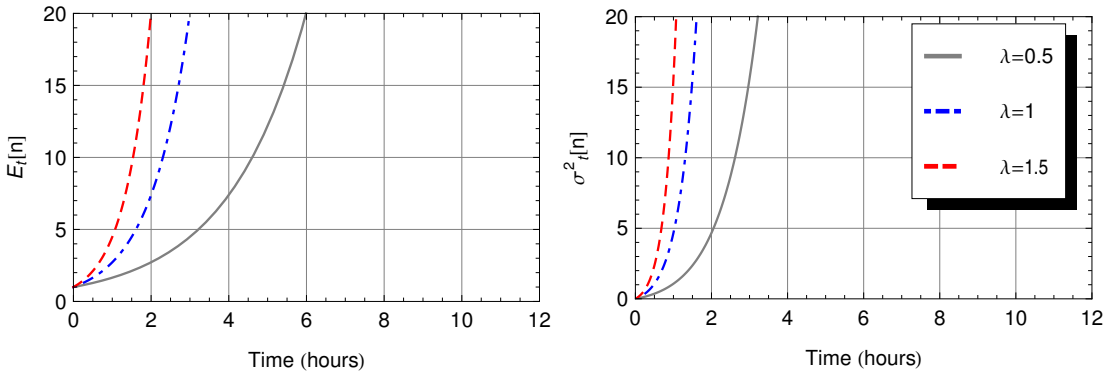


Figure 3.2.2.: SComI model: Mean and variance of botnet size

We now present some numerical results, depicted in Figs. 3.2.2 and 3.2.3, to illustrate

[1]The derived probability distribution is a geometric distribution for a fixed value of t.

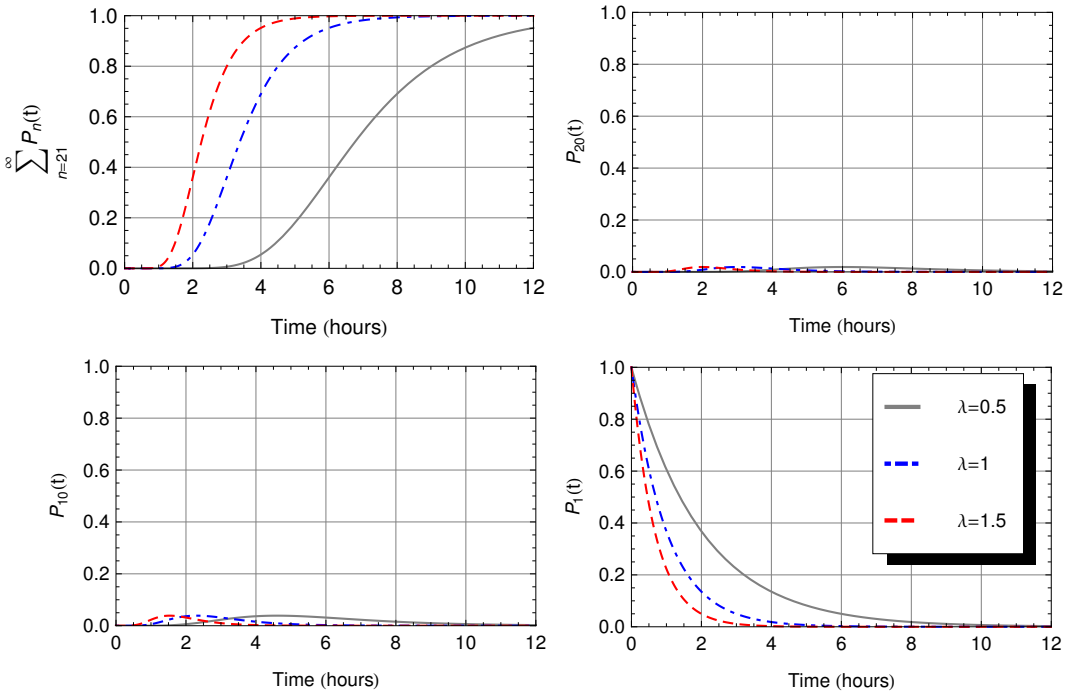


Figure 3.2.3.: SComI model: Time-dependent probability of the number of *Compromised* nodes (botnet size) for $\lambda = 0.5, 1.0, 1.5$

how the derived probability distribution could be used in the study of any particular parameter of interest in the process of botnet expansion. Time-dependent mean and variance have been calculated and drawn, depicted in Fig. 3.2.2, which show how quickly botnet expansion can happen if the botnet is able to expand throughout the Internet. In each figure, we demonstrate the effect of varying values of λ (with the unit of nodes/hour) which provides insight on how this parameter affects the speed of botnet expansion. In the numerical analysis of the SComF model (introduced in the next section), we set the total number of nodes (N) to 20 (N can be set to any arbitrarily large value of interest as well). To have a comparison between the two models, in Fig. 3.2.3, we draw the time-dependent probabilities for several values of N; in the figure, probabilities of botnet size being one (bottom right), ten (bottom left), twenty (top right), or greater than twenty (top left) are depicted.

3.3. SComF: Unhindered Botnet Expansion Model - Finite Population Size

In this section, we model unhindered growth of the botnet with the finite population size assumption. In this model, there is a fixed number of nodes in *Susceptible* stage and these nodes move to *Compromised* stage as time goes by. State of the system is therefore defined to be the number of nodes in the aforementioned stages. Our development leads to a solution for the time-dependent probability distribution of the number of nodes in the botnet (nodes in *Compromised* stage).

3.3.1. State-transition-rate Diagram

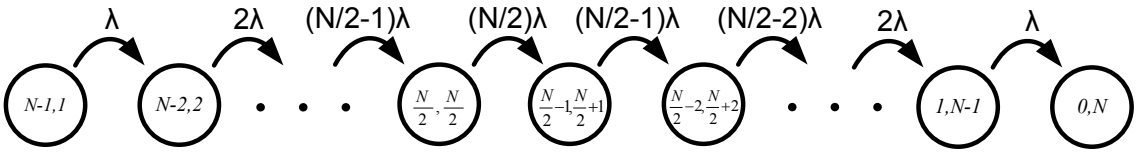


Figure 3.3.1.: SComF botnet expansion model: 2-dimensional CTMC. In the middle, the expansion rate starts to decrease.

The state-transition-rate diagram for the SComF model is depicted in Fig. 3.3.1. As initial condition, we assume that the size of the botnet is one and there are $N-1$ nodes in *Susceptible* stage. A state in the 2-dimensional CTMC is denoted by the duplet (n_0, n_1) (n_0 is the number of nodes in *Susceptible* stage and n_1 is the number of nodes in *Compromised* stage, as indicated in the diagram). Since $n_0 + n_1 = N$, n_0 and n_1 are interdependent. As a result, we can drop one of them. For simplicity of notation, let us denote $P_{n0,n1}(t)$ as $P_{n1}(t)$ by dropping n_0 (i.e., considering n_0 to be the dependent variable). Finally, let us use n instead of n_1, thus $P_{n1}(t)$ is replaced by $P_n(t)$. $P_n(t)$ is therefore the time-dependent probability distribution of the number of nodes in the botnet.

In this model, we consider that each node in the botnet recruits one node (grows the size of the botnet by one) with probability $\lambda\Delta t + o(\Delta t)$ in any Δt interval. The expansion rate

continues to increase up to the point where half of the susceptible population has left this stage. At this point, there are less nodes in *Susceptible* stage in the neighborhood of each node of the botnet; this would lead to a decrease in the expansion rate from that point on. The rate will continue to decrease until all nodes are in *Compromised* stage.

3.3.2. Probability Distribution Derivation

3.3.2.1. Differential-Difference Equations

For this birth process, the equations for the rate of change of probabilities are as follows:

$$\begin{cases} \frac{dP_1(t)}{dt} = -\lambda P_1(t) & n = 1 \\ \frac{dP_n(t)}{dt} = (n-1)\lambda P_{n-1}(t) - n\lambda P_n(t) & 2 \leq n \leq \frac{N}{2} \\ \frac{dP_n(t)}{dt} = (N-n+1)\lambda P_{n-1}(t) - (N-n)\lambda P_n(t) & \frac{N}{2}+1 \leq n \leq N \end{cases} \tag{3.3.1}$$

Initial condition is $P_1(0) = 1$. Without loss of generality, we assume N to be even.

3.3.2.2. Laplace Transform of the Probability Distribution

In order to derive the probability distribution, we first move (3.3.1) to Laplace domain. We let $P_n^*(s)$ denote the Laplace transform of $P_n(t)$ and then use the induction method in order to derive the probability distribution in the Laplace domain as follows (see the appendix, Section A.2 for a detailed derivation):

$$P_n^*(s) = \begin{cases} \frac{1}{s+\lambda} & n = 1 \\ \frac{(n-1)!\lambda^{n-1}}{\prod_{k=1}^{n}(s+k\lambda)} & 2 \leq n \leq \frac{N}{2} \\ \frac{\frac{\frac{N}{2}!}{(N-n)!}\lambda^{(n-\frac{N}{2})}}{\prod_{k=1}^{n-\frac{N}{2}}\left(s+(\frac{N}{2}-k)\lambda\right)} P_{\frac{N}{2}}^*(s) & \frac{N}{2}+1 \leq n \leq N \end{cases} \tag{3.3.2}$$

3.3.2.3. Probability Distribution

In order to derive the probability distribution, we apply the partial fraction expansion method to expressions obtained for $P_n^*(s)$ in order to perform the Laplace inversion; the process is detailed in the appendix, Sub-section A.2.2. Finally, the probability distribution of the number of nodes in the botnet (nodes that are in *Compromised* stage) at time t is determined as follows:

$$P_n(t) = \begin{cases} e^{-\lambda t} & n = 1 \\ \sum_{k=0}^{n-1} \left((-1)^k \binom{n-1}{k} e^{-(k+1)\lambda t} \right) & 2 \le n \le \frac{N}{2} \\ \sum\limits_{\substack{k=1 \\ k \notin [N-n, \frac{N}{2}-1]}}^{\frac{N}{2}} \left(\frac{T_1}{T_2} e^{-k\lambda t} \right) + \sum\limits_{k \in [N-n, \frac{N}{2}-1]}^{\frac{N}{2}-1} \left(\frac{T_1}{T_2} t e^{-k\lambda t} + \left. \frac{d(\frac{T_1}{T_3})}{ds} \right|_{s=-k\lambda} e^{-k\lambda t} \right) & \frac{N}{2}+1 \le n < N \\ \frac{T_1}{T_2|_{k=0}} + \sum_{k=1}^{\frac{N}{2}-1} \left(\frac{T_1}{T_2} t e^{-k\lambda t} + \left. \frac{d(\frac{T_1}{T_3})}{ds} \right|_{s=-k\lambda} e^{-k\lambda t} \right) + \frac{T_1}{T_2|_{k=\frac{N}{2}}} e^{-\frac{N}{2}\lambda t} & n = N \end{cases} \tag{3.3.3}$$

where T_1, T_2 and T_3 are given as follows:

$$\begin{aligned} T_1 &= \frac{\frac{N}{2}!}{(N-n)!} \lambda^{(n-\frac{N}{2})} (\frac{N}{2}-1)! \lambda^{\frac{N}{2}-1} \\ T_2 &= \prod_{\substack{i=1 \\ i \neq k}}^{\frac{N}{2}} (i-k)\lambda \prod_{\substack{j=N-n \\ j \neq k}}^{\frac{N}{2}-1} (j-k)\lambda \\ T_3 &= \prod_{\substack{i=1 \\ i \neq k}}^{\frac{N}{2}} (s+i\lambda) \prod_{\substack{j=N-n \\ j \neq k}}^{\frac{N}{2}-1} (s+j\lambda) \end{aligned}$$

3.3.3. Numerical Analysis

Comparable to the numerical analysis for the SComI model, Fig. 3.3.2 depicts the time-dependent mean and variance of the number of nodes in *Compromised* stage (botnet size) for the SComF model. Fig. 3.3.3 depicts the three respective time-dependent probabilities for the model. Like before, the effect of varying values of λ on the speed of botnet expansion can be observed in the figures. It is also interesting to observe the "saturation effect" as the botnet expands to all nodes, depicted in Fig. 3.3.2.

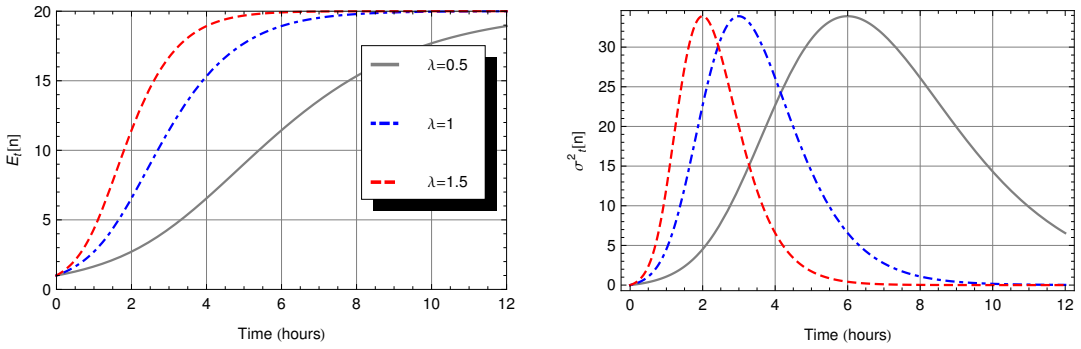


Figure 3.3.2.: SComF model: Mean and variance of botnet size (population size $N = 20$)

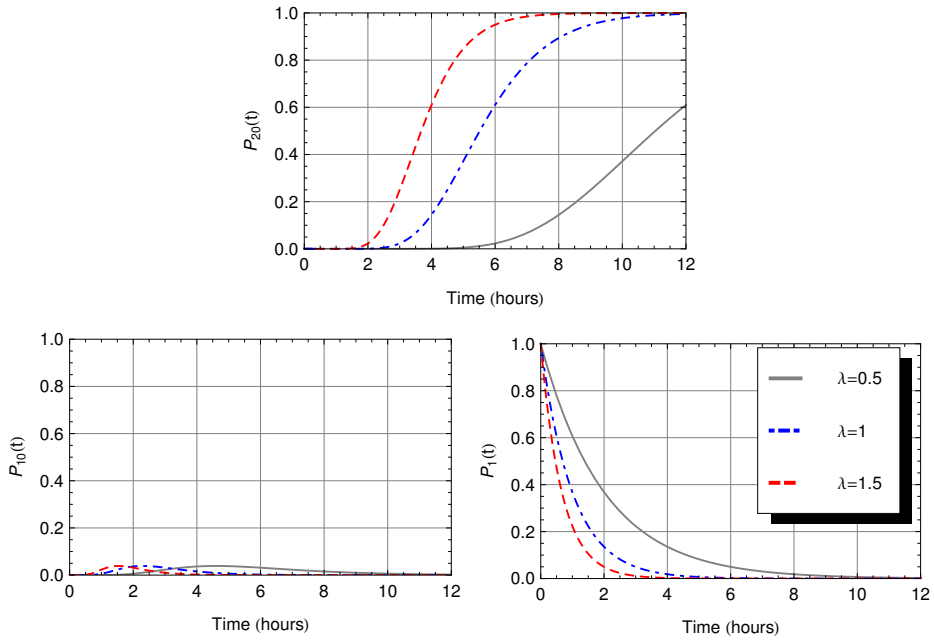


Figure 3.3.3.: SComF model: Time-dependent probabilities of the botnet size (population size $N = 20$)

3.4. Simulation Study

In this section, we provide a brief simulation study which is complementary to the presented numerical analysis. The simulation study concerns the SComF model, as the SComI model does not lend itself to comparison to a simulated network with limited number of nodes. We borrow the term "spearhead" and the notion of "two-stage worm" used in [57] to describe our simulation study. *Spearhead* is the initial infection (e.g., a worm) that spreads in the network after which the infected nodes start to connect to the botnet which is the process of botnet expansion. The botnet expansion therefore happens with a delay compared to the spread of the initial infection. In this simulation study, we examine the spread of the initial infection (the worm spread) and compare it to the determined analytical result for the botnet expansion.

The infection of nodes (e.g., worm spread) is independent from the connection of infected nodes to the botnet. Specifically, there is no relationship between the infection rate and the connection rate (λ), i.e., the latter could be lower or higher than the former. It is, however, interesting to observe how the botnet could expand in relation to the spread of the worm, in case these rates are equal.

The worm models developed in the Georgia Tech Network Simulator (GTNetS) [64] seem to be the most well-developed publicly-available simulation models for worm spread. With these models incorporated, the simulator models the activities of the worms in packet-level detail. In this simulator, a UDP-based worm can be configured using parameters such as worm "scan rate", whereas a TCP-based worm can be configured by setting the number of simultaneous TCP connections that each infectious node can create. We use a UDP-based worm in this simulation in order to be able to have a comparison between λ of the SComF model and the "scan rate" of the worm model[2].

Comparable to the numerical results for the SComF model, we simulated a 20-host topol-

[2]The simulation scenario is a slightly modified version of *wormsim.cc* in GTNetS; see Section A.3 for the code and for a screen shot of the simulated topology.

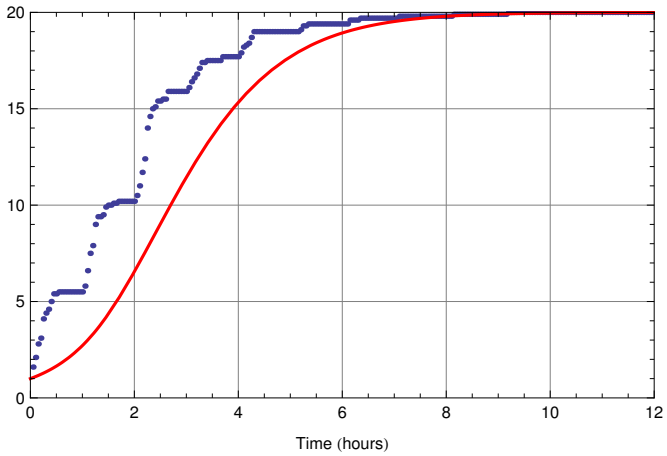


Figure 3.4.1.: Count of infected hosts over time in the simulation in blue dots (scan rate = 1) vs. the botnet expansion curve using the SComF model in red ($\lambda = 1$)

ogy and tracked the number of infected hosts over time. The worm "scan rate" is the number of infectious packets each infected host sends out each hour (here, scan rate is set to one). The number of infected hosts over time (mean of 10 runs) is depicted in Fig. 3.4.1 in blue dots (value reported 20 times each hour). In the same figure, we also show the botnet growth in red, which is the $\lambda = 1$ curve from the left sub-figure of Fig. 3.3.2. We therefore can observe that the botnet growth correctly lags the spread of infection.

The contribution of the work presented in this chapter is twofold: (1) two *stochastic analytical* botnet models, SComI and SComF, that cover both cases of infinite and finite node population sizes ; and (2) the method of examination of the interaction between the botnet expansion and the worm spread using the GTNetS simulator. In order to use the models in the real world, one could consider the following methods when trying to estimate a value for λ: (1) local measurements through Honeynet log analysis [42], for example; and (2) a statistical approach to botnet virulence estimation which has recently been proposed [65]; this latter method improves the reliability of the process of estimating λ.

4. The SIC Botnet Model

4.1. Introduction

In the previous chapter, we introduced the SComI and SComF botnet models which are useful for the initial unhindered botnet expansion phase, as they take into account only two node stages. In this chapter, we increase the complexity of the model one step further by considering three node stages, which allows for the capture of botnet size fluctuations throughout the botnet lifecycle. The stages can be considered to have been created as a result of the split of the *Compromised* stage of the SComI model into two stages of *Infected* and *Connected* of the SIC model which will be explained shortly.

This chapter is organized as follows: in Section 4.2, the SIC model is introduced by describing the Continuous-Time Markov Chain (CTMC) model as well as justifying the modeling assumptions. We then present an extensive performance modeling of the SIC model in Section 4.3. First, the fundamental probability flow equations resulting from the CTMC model are presented. We then proceed to derive the means, variances, and Basic Reproduction Number of the SIC model. Afterwards, we introduce the developed link between the SIC model and mitigation strategies aimed at DHT-based P2P botnets. As a case study, we analyze a random sybil attack on a P2P botnet and examine how the attack can be fine-tuned based on the information provided by the SIC model. Next, we study in Section 4.4 how the results estimated by the SIC model would relate to some of the reported botnet size measurements. Finally, in Section 4.5, numerical results are provided showing

the kinds of insight that can be drawn from the SIC model based on the aforementioned derived analytical results.

4.2. The SIC Model

In this section, we present our botnet lifecycle model and then develop its mathematical representation. We first introduce the model basics and later elaborate on the main assumptions of the model.

4.2.1. Node Stages in the SIC Model

As reported extensively in the literature [17, 18, 48, 66], a node, when infected by a botnet-related malware, goes through multiple stages in the lifetime of the botnet, with the main stages being *Susceptible*, *Infected*, and *Connected*. Here are the definitions of these terms, as used in this chapter (and in the rest of the thesis):

Susceptible (S) A node is considered to be in the Susceptible stage, if it is healthy, whether or not vulnerable. A vulnerable node can be infected through at least one of the possibly many infection vectors (worm scans, e-mail attachments, etc.) deployed simultaneously or sequentially by the botmasters of a single botnet. On the other hand, a node is invulnerable if either it cannot be infected by any infection vector or the address is either unused or unroutable/unreachable. As defined, the Susceptible node population corresponds to the entire population of the Internet. The term Susceptible refers to the fact that until probed, one usually cannot determine whether or not the node is vulnerable. A Susceptible node may either get infected with the small probability p and possibly later become part of the botnet or remain healthy throughout the whole period with the large probability of $1-p$. All nodes are initially considered to be in the Susceptible stage.

Infected (I) The *Infected* stage denotes a stage in which a node has been infected by any of the infection vectors that have been utilized by the botmasters. In this stage, the node usually does not have the full malware code to engage in illicit activities; this is primarily for keeping the payload small. The minimal malware code serves only to connect the node to the botnet and pass the node to the *Connected* stage.

Connected (C) The *Connected* stage refers to the stage when the node is connected to the botnet, can download the full malware code and receive the botmasters' Command & Control (C&C) traffic, and therefore, it is part of the army of bots controlled by the botmasters.

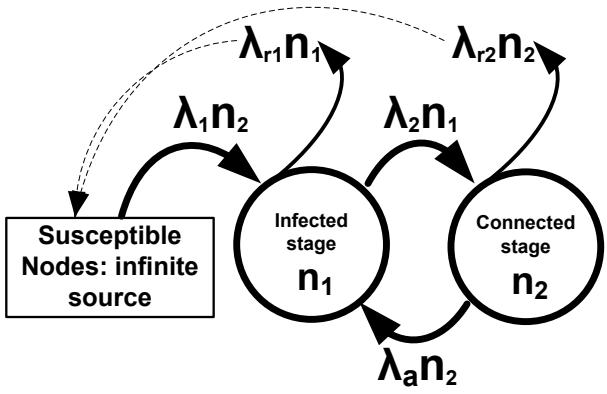


Figure 4.2.1.: SIC model: 2-dimensional birth-death CTMC

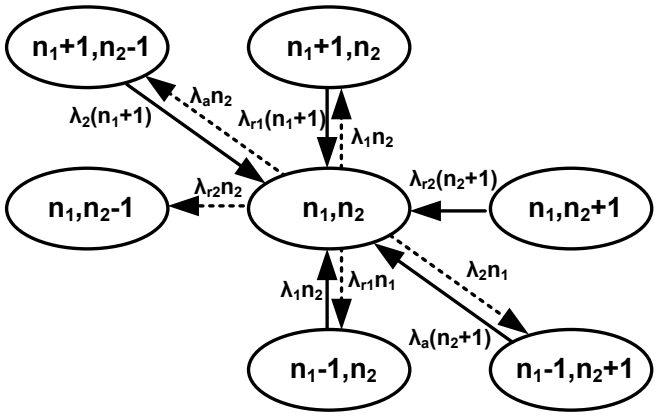


Figure 4.2.2.: SIC model: State-transition-rate diagram

As we model the lifecycle of a node with the aforementioned three stages, the model is referred to as the Susceptible-Infected-Connected (SIC) model. In Fig. 4.2.1, we show the stages of the model and the transitions between the stages. As shown in the figure, we let n_1 and n_2 denote the number of nodes in *Infected* and *Connected* stages, respectively, and the state of the system is represented by the vector (n_1, n_2). In Fig. 4.2.2, we show all the transitions from and to state (n_1, n_2).

In this model, we consider that each node in the botnet (nodes in *Connected* stage) infects one *Susceptible* node (increases n_1 by one) with probability $\lambda_1 \Delta t + o(\Delta t)$ in any Δt interval (cf. Fig. 4.2.1). Thus the time interval for a Connected node to infect a Susceptible node is exponentially distributed with parameter λ_1 and the transition rate between Susceptible and Infected stages is given by $\lambda_1 n_2$. Further, each Infected node can transition to Connected stage (which increases n_2 and decreases n_1) with probability $\lambda_2 \Delta t + o(\Delta t)$ in any Δt interval. Finally, there is a transition rate ($\lambda_a n_2$) from Connected stage to Infected stage. This transition rate represents an attack on the botnet, attacks such as index poisoning and sybil attacks in the case of P2P botnets. Under such attacks, nodes do not transition back to Susceptible stage; they just lose the ability to communicate and might be able to reconnect again (hence the rate from Connected stage back to Infected stage). We further assume the rate of disinfection of nodes which are in Infected stage and Connected stage to be $\lambda_{r1} n_1$ and $\lambda_{r2} n_2$, respectively.

4.2.2. Model Assumptions

In this sub-section, we put forward the reasoning behind the assumptions made in the development of the SIC model. To the best of our knowledge, these assumptions are reasonable mathematically as well as consistent with precedence and evidence from closely-related phenomena such as malware propagation and spread of human disease.

4.2.2.1. CTMC (Exponential probability distributions) Modeling

Continuous-time Markov Chain (CTMC) models are based on the assumption that the time intervals for the transitions of nodes from one stage to the next one are exponentially distributed with different parameters. In this part, we first provide the mathematical basis for the use of exponential distributions and then describe how this assumption is in agreement with precedence.

A. Mathematical Basis: Each attempt to make a node transition from any stage to another stage is a Bernoulli trial with success probability of p. We explain how this concept of Bernoulli trial corresponds to physical reality first for the transition from Susceptible to Infected ($\mathbf{S} \rightarrow \mathbf{I}$) which is the most important transition in the model leading to the exponential growth of the number of nodes in the Infected and Connected stages. At the end, we briefly explain how the same concept of Bernoulli trial also corresponds to physical reality for the rest of the transitions in the SIC model.

Most botnets apply worm-scanning methods to recruit new bots [67]. Further, it has been reported that 66.5% of scan patterns are *uniform* random scanning [68]. To explain the process with a concrete example, we therefore consider *uniform random scanning* as the infection vector used by the botnet node. Using the terminology presented in [69], for a uniform scan worm, η is the average scan rate, i.e., the average number of scans a botnet node sends out per unit of time. Each scan corresponds to an attempt to infect a susceptible node. If the susceptible node is vulnerable to this specific worm scan, then, it will be infected, otherwise the attack will fail and the node will remain healthy. η is therefore equal to m, which is the number of aforementioned Bernoulli trials. The campaign of a single botnet node to infect can then be viewed as a series of Bernoulli trials with few successes/infections among many failures.

The above series of Bernoulli trials has therefore a Binomial distribution with parameters p (success probability) and m (number of Bernoulli trials). A Binomial distribution can be approximated by a Poisson distribution with parameter $\lambda = mp$, when p is small

and m is large [70, pp.111-3]. Note that m is different from n_1 and n_2 which denote the numbers of Infected and Connected nodes, respectively; however, the λ parameter refers to λ_1 indicated in Fig. 4.2.1. The conditions on the values of m and p are consistent with the S $\rightarrow$ I transition, as the success probability is low and the number of trials is large. Therefore, the probability distribution of the number of nodes making the S $\rightarrow$ I transition in unit time period can be approximated by this Poisson distribution. Further, as sum of processes each having a Poisson distribution with parameter λ_1 also has a Poisson distribution, the whole arrivals into the Infected stage (i.e., nodes making the S $\rightarrow$ I transition) due to all botnet nodes then have a Poisson distribution with parameter $\lambda_1 n_2$. From the Poisson distribution, it follows that the time intervals between node arrivals to the Infected stage are exponentially distributed.

As noted at the beginning, we provide a brief explanation regarding how the same concept of Bernoulli trial also corresponds to physical reality for the rest of the transitions in the SIC model as follows:

I $\rightarrow$ C Each Infected node which has the minimal malware code to help it to connect itself to the botnet makes, on average, several attempts to either connect to the central C&C server or find peers in a P2P botnet. As such, we can designate a success probability of p for the successful connection to the botnet for these attempts each of which can be considered a Bernoulli trial.

C $\rightarrow$ I When the botnet is under attack, the effort to disconnect each botnet node can also be considered a Bernoulli trial with a success probability of p which is the probability of disconnection. As botnet mitigation strategies are generally complicated and hard to implement with often limited impact on the botnet, on average, this per-node success probability is small.

I $\rightarrow$ S & C $\rightarrow$ S Similar to the attack on the botnet, each attempt to disinfect a node that is in either stages of Infected or Connected can be considered a Bernoulli trial with a

success probability of p, i.e., the probability of disinfection. As the identification of most nodes as well as the physical access to them are hard, on average, this success probability is small.

With the aforementioned descriptions for the characteristics of all the inter-stage transitions, the CTMC model can be considered a reasonable approximation.

B. Accordance with Precedence: CTMC as a modeling tool in epidemiology has a proven track record [52] that deals with the phenomenon of spread of an element within a susceptible population which has a close resemblance to the spread of malware and the expansion of a botnet. Further, successful use of CTMC models in the study of spread of malware has also been documented [61]. Expansion, and size evolution, of botnets happen under the influence of the same physical processes as the ones affecting the spread of malware; therefore, the use of the same CTMC theory for botnets is a natural extension. To our knowledge, the only case of application of CTMC-like models to botnets is the work of [48] which is a simulation model that has been developed based on the measurement data of the Storm botnet. Finally, in terms of the choice of Poisson distribution for the arrival of nodes into a stage (i.e., the exponentially-distributed inter-arrival times), similar to the SIC model, [53] has also determined this assumption to be reasonable in the study of Conficker-C botnet/worm for the distribution of the number of UDP connection attempts made by an infected host.

4.2.2.2. Node Stages and Transitions

A. Main Node Stages Considered: As described in Sub-section 4.2.1, the main dynamics of botnets can be captured by keeping track of the two main node stages, i.e., *Infected* and *Connected*. On the other hand, as mentioned in Sub-section 1.2.1, the number of stages considered in a stochastic model, and in our CTMC model in particular, must be limited if we are to avoid an intractable model caused by consideration of several node stages. Based on our extensive investigations and considering the prior work done in this field, the optimal

tradeoff has been determined to be the consideration of the aforementioned two node stages (i.e., *Infected* and *Connected*), each being a dimension in the CTMC (hence the number of nodes in each of these two stages is tracked), with *Susceptible* stage having infinite number of nodes (hence the number of *Susceptible* nodes need not be tracked). An infinite susceptible population is a reasonable assumption, since this population corresponds to the population of the entire Internet which is an assumption made also in [48].

B. No Immune/Removed Stage Considered: As botmasters use a plethora of methods to infect (and re-infect) the nodes, it is reasonable to assume that a node is never in Immune (or Removed) stage; therefore, we do not consider this stage in our model. It is important to remember that existence and maintenance of a botnet is independent of any infection vector (e-mail attachments, file sharing sites, worm scans, etc.) used by the botmaster and obtaining immunity against one infection vector still leaves the node susceptible to be re-infected through other infection vectors.

C. Botnet's Footprint vs. Live Population: Using the terminology presented in [44], we emphasize that the SIC model tracks the botnet's "footprint" and not its "live population". As such, effects such as day/night differences and time zones which impact the number of live botnet nodes at any given time, are not taken into account. In the SIC model, *Connected* nodes represent the total number of botnet nodes, i.e., botnet's footprint. On the other hand, it is possible to use the SIC model and take into account the effects of time zones and day/night differences on λ parameters' values as follows depending on the length of the analysis period: (1) if the analysis period is around or less than 24 hours, then, piecewise time-invariant parameters can be used, i.e., we use different sets of values for the λ parameters in each 12-hour analysis period to account for the day/night differences and/or the time zones; and (2) if the analysis period is significantly more than 24 hours, e.g., weekly size variations are important as is the case in the analysis of FourLakeRiders botnet in Section 4.4, then, the variations due to time zones and day/night differences are insignificant and average parameter values will yield accurate results.

D. Accommodating Time-variant Parameters: As presented, the λ parameters are considered constant throughout the analysis period. It is however possible to use the SIC model if these parameters change over time using piecewise time-invariant parameters, i.e., in each piece of the analysis period, we consider the parameters to be constant. The duration of each piece can be decided upon on a case-by-case basis; an example of this kind of analysis, with each piece duration to be a week, is presented in Section 4.4. Another example of this kind of analysis, as suggested in the above point, is to accommodate the effects of time zones and the day/night differences when the analysis period is less than 24 hours. In this case, we can choose a 12-hour analysis period during which we consider the λ parameters to be constant and can set low values for the λ parameters during night time.

As described above, the SIC model and its main assumptions are similar to the model in [48] which has been based on the gathered information about the Storm botnet. These assumptions were further justified mathematically and through comparison to other similar works. As a result, we believe that we have a realistic model, which leads us to two-dimensional Markovian birth-death processes. Using the model, we can study the size evolution of a botnet as well as effectiveness of mitigation strategies by monitoring the number of nodes that are in Infected and Connected stages at any given time.

4.3. Performance Modeling of the SIC Model

In this section, we provide an extensive performance modeling of the SIC model. First, botnet size evolution phases and initial state values for the SIC model are explained. We then proceed to derive the probability flow equations based on the two-dimensional CTMC of the SIC model. These probability flow equations are further reduced to a partial differential equation (PDE) of the probability generating function (PGF). Directly from this PDE, we then derive the mean and variance of the SIC model. Next, the derivation of the Basic Reproduction Number, which is a widely used parameter in epidemiology and the study of

malware propagation, is documented. We conclude this section by deriving a novel analytical result which is a link between the SIC model and the mitigation strategies against Distributed Hash Table (DHT)-based P2P botnets.

4.3.1. Botnet Size Evolution Phases and Initial State Values

A botnet may go through many phases during its lifecycle, where a phase will refer to a period that system parameters (λ_1, λ_2, λ_{r1}, λ_{r2}, and λ_a) remain constant. For example, when a botnet appears for the first time, it will probably experience unhindered expansion as there will not be any active mitigation strategies to counter its growth; thus, λ_{r1}, λ_{r2}, and λ_a will be zero. Typically, the botnet's population will alternate between sawtooth growth period followed by a period of relatively stable population size [71]. The sawtooth growth begins with the release of a new infection; after sometime, it will be reversed with the deployment of new counter measures until an equilibrium is reached. Probably, new equilibrium population will have a size greater than previous equilibrium size. In any phase, the SIC model will apply with the end results of the preceding phase providing the initial conditions (state values) to the next phase.

4.3.2. Probability Flow Equations and PDE of PGF

In this section, we determine the probability flow equations and then, the partial differential equation (PDE) of the probability generating function (PGF) describing the system. Let $P_{n_1,n_2}(t)$ denote the probability that the system is in state (n_1, n_2) at time t. We write probability flow equations through inspection from the state-transition-rate diagram given in Fig. 4.2.2 by equating the rate of change of probabilities at any state to the difference between the total input/output flows to/from that state as follows.

$$
\begin{cases}
\dfrac{dP_{n_1,n_2}(t)}{dt} = \lambda_1 n_2 P_{n_1-1,n_2}(t) + \lambda_{r1}(n_1+1)P_{n_1+1,n_2}(t) \\
\qquad +\lambda_{r2}(n_2+1)P_{n_1,n_2+1}(t) + \lambda_2(n_1+1)P_{n_1+1,n_2-1}(t) \\
\qquad +\lambda_a(n_2+1)P_{n_1-1,n_2+1}(t) \\
\qquad -(\lambda_1 n_2 + \lambda_{r1} n_1 + \lambda_{r2} n_2 + \lambda_2 n_1 + \lambda_a n_2)P_{n_1,n_2}(t) \\
\qquad\qquad \big\langle n_1 > 0, n_2 > 0 \big\rangle \, (a) \\
\dfrac{dP_{0,n_2}(t)}{dt} = \lambda_{r1} P_{1,n_2}(t) + \lambda_{r2}(n_2+1)P_{0,n_2+1}(t) + \lambda_2 P_{1,n_2-1}(t) \\
\qquad -(\lambda_1 n_2 + \lambda_{r2} n_2 + \lambda_a n_2)P_{0,n_2}(t) \\
\qquad\qquad \big\langle n_1 = 0, n_2 > 0 \big\rangle \, (b) \\
\dfrac{dP_{n_1,0}(t)}{dt} = \lambda_{r1}(n_1+1)P_{n_1+1,0}(t) + \lambda_{r2} P_{n_1,1}(t) + \lambda_a P_{n_1-1,1}(t) \\
\qquad -(\lambda_{r1} n_1 + \lambda_2 n_1)P_{n_1,0}(t) \\
\qquad\qquad \big\langle n_1 > 0, n_2 = 0 \big\rangle \, (c) \\
\dfrac{dP_{0,0}(t)}{dt} = \lambda_{r1} P_{1,0}(t) + \lambda_{r2} P_{0,1}(t) \\
\qquad\qquad \big\langle n_1 = 0, n_2 = 0 \big\rangle \, (d)
\end{cases}
\tag{4.3.1}
$$

In order to solve (4.3.1) and derive the probability distribution $P_{n_1,n_2}(t)$, a known method is to transform the equations of probability flows to a partial differential equation (PDE) of the probability generating function (PGF) which can be tackled using known methods to solve PDEs. Let us denote $P(z_1,z_2,t)$ as the PGF of the probability distribution $P_{n_1,n_2}(t)$ which is given by $P(z_1,z_2,t) = \sum_{n_1=0}^{\infty}\sum_{n_2=0}^{\infty} P_{n_1,n_2}(t) z_1^{n_1} z_2^{n_2}$.

The initial probability distribution is denoted by $P_{k_1,k_2}(0)$. Here, we assume that the initial number of nodes in each stage is constant (k_1,k_2). Though the initial derivations are conditional, we will suppress the conditions for simplicity in expressing the PDE. This aspect, however, has been fully taken care of in the derivation of means (e.g., see (B.3.5) and (B.3.6)) and later in the derivation of variances.

We multiply each of the equations in (4.3.1) by $z_1^{n_1} z_2^{n_2}$, sum over the respective ranges

of n_1 and n_2, and then add them together. After some simplifications and manipulations (detailed derivation provided in Appendix B.1), we arrive at the following PDE of the PGF:

$$(\lambda_{r1}+\lambda_2 z_2-\lambda_{r1}z_1-\lambda_2 z_1)\frac{\partial P(z_1,z_2,t)}{\partial z_1}$$
$$+(\lambda_1 z_1 z_2+\lambda_{r2}+\lambda_a z_1-\lambda_1 z_2-\lambda_{r2}z_2-\lambda_a z_2)\frac{\partial P(z_1,z_2,t)}{\partial z_2}-\frac{\partial P(z_1,z_2,t)}{\partial t}=0 \quad (4.3.2)$$

Our efforts to solve the preceding PDE, however, have not been successful, as detailed in Appendix B.2. Nonetheless, there are publications reporting new solved cases of Abel/Lienard equations (differential equations encountered in the process of solving the PDE). Thus, it is possible that we may have the solution of the PDE in the near future. We can still obtain from the PDE the moments of botnet population size, as presented next.

4.3.3. Derivation of the Time-dependent Mean and Variance of Botnet Population Size

In this section, we derive the means and variances of the number of nodes in *Infected* stage and *Connected* stage (botnet population size) as a function of time. Let $E_t[n_1]$ and $E_t[n_2]$ denote the mean number of nodes that are in *Infected* and *Connected* stages at time t, respectively, then:

$$E_t[n_1]=\frac{\partial P(z_1,z_2,t)}{\partial z_1}\,|_{z_1=z_2=1}\ ,\ E_t[n_2]=\frac{\partial P(z_1,z_2,t)}{\partial z_2}\,|_{z_1=z_2=1} \quad (4.3.3)$$

We take the derivatives of the PDE given in (4.3.2) with respect to z_1 and z_2, separately. By setting $z_1=z_2=1$ in each resulting equation, we arrive at a set of ODEs of $E_t[n_1]$ and $E_t[n_2]$. To emphasize the time dependency of the means from here on, we will denote $E_t[n_1]$ and $E_t[n_2]$ by $E_1(t)$ and $E_2(t)$, respectively. Note that only the important steps of derivation are provided here; the rest of the steps is in Appendix B.3. After the initial steps outlined

above and detailed in the appendix, we arrive at the following set of ODEs:

$$\begin{cases} \frac{dE_1(t)}{dt} = & (\lambda_1+\lambda_a)E_2(t)-(\lambda_2+\lambda_{r1})E_1(t) \\ \frac{dE_2(t)}{dt} = & \lambda_2 E_1(t)-(\lambda_{r2}+\lambda_a)E_2(t) \end{cases} \tag{4.3.4}$$

We then proceed to derive $E_1(t)$ and $E_2(t)$ from the previous set of ODEs as detailed in Appendix B.3; the final results are as follows:

$$\begin{aligned} E_1(t) = \Bigg[& \exp\left(-\frac{1}{2}t\left(\lambda_{\mathrm{T3}}+\lambda_{\mathrm{T1}}\right)\right)\left(\bar{k_1}\lambda_2\left(-\exp\left(t\lambda_{\mathrm{T3}}\right)\right)\right. \\ & +\left(\bar{k_1}\lambda_a-\bar{k_1}\lambda_{\mathrm{r1}}+\bar{k_1}\lambda_{\mathrm{r2}}+\bar{k_1}\lambda_{\mathrm{T3}}+2\lambda_1\bar{k_2}\right)\exp\left(t\lambda_{\mathrm{T3}}\right) \\ & +2\bar{k_2}\lambda_a\exp\left(t\lambda_{\mathrm{T3}}\right)+\bar{k_1}\lambda_{\mathrm{T3}}+\bar{k_1}\lambda_2-\bar{k_1}\lambda_a+\bar{k_1}\lambda_{\mathrm{r1}} \\ & \left.-\bar{k_1}\lambda_{\mathrm{r2}}-2\lambda_1\bar{k_2}-2\bar{k_2}\lambda_a\right)\Bigg]/\left(2\lambda_{\mathrm{T3}}\right) \end{aligned} \tag{4.3.5}$$

$$\begin{aligned} E_2(t) = \Bigg[& \exp\left(-\frac{1}{2}t\left(\lambda_{\mathrm{T3}}+\lambda_{\mathrm{T1}}\right)\right)\left(2\bar{k_1}\lambda_2\exp\left(t\lambda_{\mathrm{T3}}\right)\right. \\ & +\left(\lambda_2\bar{k_2}-\bar{k_2}\lambda_a+\bar{k_2}\lambda_{\mathrm{r1}}-\bar{k_2}\lambda_{\mathrm{r2}}+\bar{k_2}\lambda_{\mathrm{T3}}\right)\exp\left(t\lambda_{\mathrm{T3}}\right) \\ & -2\bar{k_1}\lambda_2+\bar{k_2}\lambda_{\mathrm{T3}}-\lambda_2\bar{k_2}+\bar{k_2}\lambda_a-\bar{k_2}\lambda_{\mathrm{r1}} \\ & \left.+\bar{k_2}\lambda_{\mathrm{r2}}\right)\Bigg]/\left(2\lambda_{\mathrm{T3}}\right) \end{aligned} \tag{4.3.6}$$

where, $\lambda_{\mathrm{T1}}=\lambda_2+\lambda_a+\lambda_{\mathrm{r1}}+\lambda_{\mathrm{r2}}$, $\lambda_{\mathrm{T2}}=-\lambda_1\lambda_2+\lambda_{\mathrm{r2}}\left(\lambda_2+\lambda_{\mathrm{r1}}\right)+\lambda_a\lambda_{\mathrm{r1}}$, and $\lambda_{\mathrm{T3}}=\sqrt{{\lambda_{\mathrm{T1}}}^2-4\lambda_{\mathrm{T2}}}$.

Next, we describe the derivation of variances, which are given by:

$$\sigma_1^2(t)=E_t[n_1^2]-\left(E_1(t)\right)^2\ ,\ \sigma_2^2(t)=E_t[n_2^2]-\left(E_2(t)\right)^2 \tag{4.3.7}$$

where:

$$E_t[n_1^2] = \frac{\partial^2 P(z_1,z_2,t)}{\partial z_1^2} \mid_{z_1=z_2=1} + \frac{\partial P(z_1,z_2,t)}{\partial z_1} \mid_{z_1=z_2=1}$$
$$E_t[n_2^2] = \frac{\partial^2 P(z_1,z_2,t)}{\partial z_2^2} \mid_{z_1=z_2=1} + \frac{\partial P(z_1,z_2,t)}{\partial z_2} \mid_{z_1=z_2=1} \tag{4.3.8}$$

Let us define:

$$\psi_1(t) \triangleq \frac{\partial^2 P(z_1,z_2,t)}{\partial z_1^2} \mid_{z_1=z_2=1}$$
$$\psi_2(t) \triangleq \frac{\partial^2 P(z_1,z_2,t)}{\partial z_2^2} \mid_{z_1=z_2=1}$$
$$\psi_{12}(t) \triangleq \frac{\partial^2 P(z_1,z_2,t)}{\partial z_1 \partial z_2} \mid_{z_1=z_2=1} \tag{4.3.9}$$

Considering that $E_1(t=0) = \bar{k_1}$ and $E_2(t=0) = \bar{k_2}$, the preceding functions have the following initial values:

$$\psi_1(t=0) = \bar{k_1^2} - \bar{k_1} \ , \ \psi_2(t=0) = \bar{k_2^2} - \bar{k_2} \ , \ \psi_{12}(t=0) = \bar{k_1 k_2} \tag{4.3.10}$$

The variances are then given by:

$$\sigma_1^2(t) = \psi_1(t) + E_1(t) - (E_1(t))^2 \ , \ \sigma_2^2(t) = \psi_2(t) + E_2(t) - (E_2(t))^2 \tag{4.3.11}$$

Next, we take the 2nd derivatives of the PDE in (4.3.2) with respect to z_1 and z_2, separately. Further, we take the derivative of the PDE with respect to z_1 and then with respect to z_2 (see Appendix B.4). By setting $z_1 = z_2 = 1$ in each resulting equation, we arrive at a set of ordinary differential equations, which if written in terms of $\psi_1(t)$, $\psi_2(t)$, and $\psi_{12}(t)$ is, as follows.

$$\begin{cases} \frac{d\psi_1(t)}{dt} = 2(\lambda_1+\lambda_a)\psi_{12}(t) - 2(\lambda_{r1}+\lambda_2)\psi_1(t) \\ \frac{d\psi_2(t)}{dt} = 2\lambda_2\psi_{12}(t) - 2(\lambda_{r2}+\lambda_a)\psi_2(t) \\ \frac{d\psi_{12}(t)}{dt} = -(\lambda_{r1}+\lambda_2+\lambda_{r2}+\lambda_a)\psi_{12}(t) + \lambda_2\psi_1(t) \\ \qquad\qquad + \lambda_1 E_2(t) + (\lambda_1+\lambda_a)\psi_2(t) \end{cases} \tag{4.3.12}$$

Finally, from the preceding set of ODEs, we obtain the variances, as explained in Appendix B.4.

4.3.4. Epidemiological Threshold: Basic Reproduction Number

Basic Reproduction Number[1] (R_0) is a widely used parameter in epidemiology as well as in the study of malware propagation. In the context of botnets, this number is the mean number of infections that any single botnet node can cause among the population of susceptible nodes. The measurement of the mean number is assumed to happen with the presence of mitigation strategies that bring down the number of botnet nodes while the remaining botnet nodes cause new infections. R_0 is calculated based on the rates used in the model. If $R_0 < 1$, the botnet will eventually disappear with probability one. If $R_0 > 1$, however, there is a probability that the botnet size will continue to increase exponentially.

Based on (4.3.4), the Basic Reproduction Number (R_0) can be derived in terms of various SIC model's parameters using the "Next Generation Matrix" method as follows (detailed derivation in Appendix B.7):

$$R_0 = \sqrt{\frac{\lambda_2(\lambda_1+\lambda_a)}{(\lambda_{r2}+\lambda_a)(\lambda_2+\lambda_{r1})}} \tag{4.3.13}$$

[1] In the theoretical epidemiology literature [52], Basic Reproduction Number (R_0) generally refers to the onset of disease spread. Once the epidemic is underway, and especially when control measures (mitigation strategies) are put into effect, other terminologies such as "Control Reproduction Number (R_c)" and "Effective Reproduction Number (R_e)" are used instead to refer to essentially the same threshold parameter. In this thesis, we use the phrase "Basic Reproduction Number (R_0)" in all instances.

4.3.5. P2P Botnet Mitigation Strategies and the SIC Model

As our last analytical result for the SIC model, we present a link between lifecycle (or propagation/population) models and the P2P botnet mitigation strategies. Mitigation strategies aimed at Distributed Hash Table (DHT)-based P2P botnets include sybil, index poisoning, and eclipse attacks. We base the discussion on random sybil attack, however, the process is similar for other attack types.

As described in Section 2.3, *Sybil attack*, first presented in [36], is an attack method under which numerous *clean* nodes (sybils) are injected into the P2P botnet, posing themselves as "legitimate" botnet nodes. They then try to re-route, block, and corrupt the Command & Control (C&C) traffic, thereby lowering the efficiency of the C&C mechanism of the botnet. In a DHT-based P2P botnet, nodes find each other, construct their routing tables, and relay the traffic to, or closer to, its intended destination based on normal DHT methods. The botmaster also relies on the aforementioned methods for the C&C of the botnet; therefore, the decreased efficiency of the C&C mechanism as a result of the sybil attack translates into an inefficient botnet.

Random sybil attack on P2P botnets has been studied in [18]. The derived formula therein can be used to construct a relationship between the number of sybils inserted in the network and $\lambda_a n_2$, the transition rate from *Connected* stage to *Infected* stage (cf. Fig. 4.2.1). The obtained formula for the random sybil attack was mentioned in (2.3.3) which is repeated here:

$$P_s(n_s) = \left(1 - \frac{n_s}{n_s + n}\right)^{\frac{\log_2(n_s+n)}{b}} \tag{4.3.14}$$

Where:

$P_s(n_s)$ The probability that a botnet node successfully obtains the botmaster's commands.

n_s The number of sybils inserted randomly in the network.

b The number of bits improved per step for a lookup (set to a mid-range value of 5 [18]).

n The botnet size. This is the value of n_2 in our model.

We therefore note that $1 - P_s(n_s)$ is the probability that a botnet node is no longer able to receive the commands of the botmaster as a result of the attack on the botnet (insertion of sybils). This probability is therefore equal to $\lambda_a \Delta t$, as the latter is the approximate probability that a botnet node transitions from *Connected* stage to *Infected* stage (i.e., the node gets disconnected). The aforementioned link between lifecycle models and the P2P botnet mitigation strategies is therefore demonstrated using the following formula:

$$\lambda_a \Delta t = 1 - P_s(n_s) \tag{4.3.15}$$

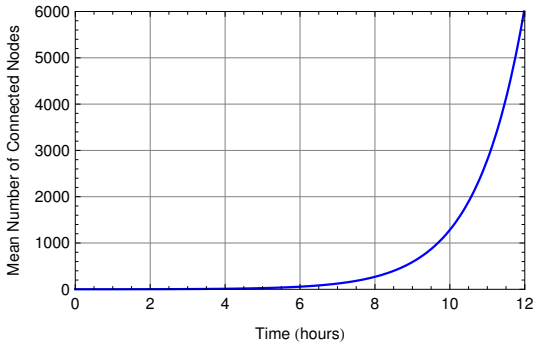
As seen in (4.3.14), $P_s(n_s)$ is a function of n_s. At any instant of time, a change in λ_a (i.e., $\Delta\lambda_a$) is a result of a change in the number of sybils (i.e., Δn_s). Based on (4.3.15), we can then analyze the relationship between the amount of change of λ_a with respect to a change in the number of sybils inserted in the network as follows:

$$\frac{\lambda_a + \Delta\lambda_a}{\lambda_a} = \frac{1 - P_s(n_s + \Delta n_s)}{1 - P_s(n_s)} \tag{4.3.16}$$

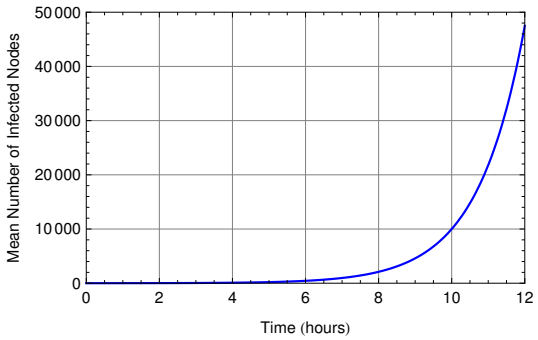
4.4. SIC Model vs. Reported Botnet Measurements

In this section, we show that our results can be used to model the botnets in the real world. Measurements of the size of some botnets have been reported on a weekly basis by Damballa [71]. Assuming that the employed measurement techniques capture correctly the global size of the botnets, in this section, we examine how such measurement results would compare to the results predicted by the SIC model. First, we examine a case of initial unhindered botnet expansion, based on available data from a *Zeus*-based botnet called *GreenAlienRiders*. Next, we will examine a case of deployment of mitigation strategies, based on available data from another *Zeus*-based botnet called *FourLakeRiders*.

GreenAlienRiders is a botnet for which the initial unhindered expansion phase has been



(a) Mean number of *Connected* nodes

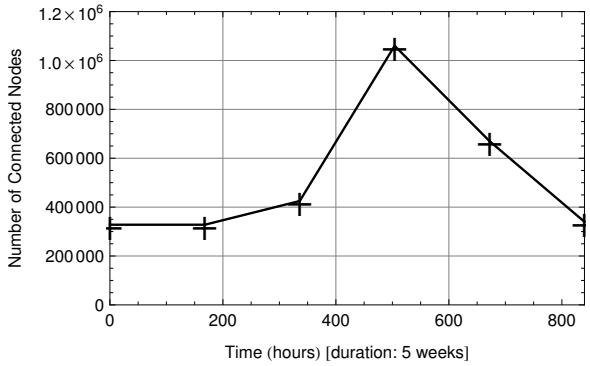


(b) Mean number of *Infected* nodes

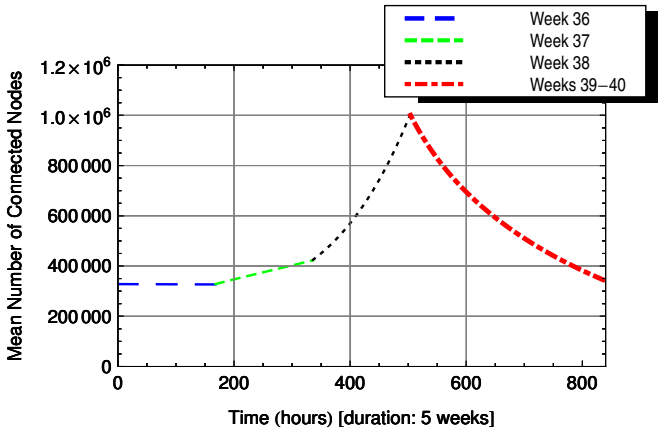
Figure 4.4.1.: GreenAlienRiders (a Zeus-based botnet): initial unhindered botnet expansion estimated using the SIC model.

captured and reported by Damballa [71]. From the Damballa report, it appears that the botnet has reached the size of about 6,000 nodes (i.e., *Connected* nodes) at Hour 12 of its appearance. To reach this size, using the SIC model, we can set $\lambda_1 = 6.85$ and $\lambda_2 = 0.1$ (both nodes/hour). The result is shown in Fig. 4.4.1a. Further, Fig. 4.4.1b shows the SIC model's estimate of the existing *Infected* nodes during this period.

FourLakeRiders, on the other hand, is a botnet for which deployment of mitigation strategies can be analyzed based on a portion of data of the botnet size evolution over time, a 5-week period from Week 36 to Week 40, as captured and reported by Damballa [71]. The data reported for this 5-week period lends itself to an analysis with clear separation of effects of each of the mitigation strategies. The scenario that follows, however, represents one of potentially many possibilities. The reported data on botnet size during this 5-week period is depicted in Fig. 4.4.2a. A scenario that fits this pattern of rise-and-fall is as fol-



(a) Reported weekly botnet size evolution



(b) Botnet size evolution reconstructed using the SIC Model

Figure 4.4.2.: FourLakeRiders (a Zeus-based botnet): botnet mitigation strategies analyzed using the SIC model (time interval: Weeks 36-40).

lows: during Week 36, the botnet size has reached an equilibrium; on one side, the number of *Infected* and *Connected* nodes grow, and on the other side, some mitigation strategies are reducing the number of *Infected* and *Connected* nodes (λ_{r1}, λ_{r2}). During Week 37, the aforementioned mitigation strategies weaken and, during Week 38, they completely disappear, which results in a steep growth of the size of the botnet. During Weeks 39 and 40, all mitigation strategies are employed (λ_{r1}, λ_{r2}, and λ_a), which results in a dramatic reduction in the size of the botnet. The described scenario is depicted in Fig 4.4.2b[2]. The potential

[2]To produce Figs. 4.4.2b and 4.4.3, parameter values have been chosen as follows: During the whole 5-week period, λ_1 and λ_2 are constant and set as follows: $\lambda_1 = 0.042$ and $\lambda_2 = 0.001$. λ_{r1} , λ_{r2} , and λ_a are chosen as follows for each week: Part 1 (Week 36): $\lambda_{r1} = 0.0082$, $\lambda_{r2} = 0.0046$, and $\lambda_a = 0$; Part 2 (Week 37): $\lambda_{r1} = 0.0082$, $\lambda_{r2} = 0.0027$, and $\lambda_a = 0$; Part 3 (Week 38): $\lambda_{r1} = 0$, $\lambda_{r2} = 0$, and $\lambda_a = 0$; Part 4 (Weeks 39-40): $\lambda_{r1} = 0.0082$, $\lambda_{r2} = 0.0046$, and $\lambda_a = 0.0057$. All λ parameters are nodes/hour.

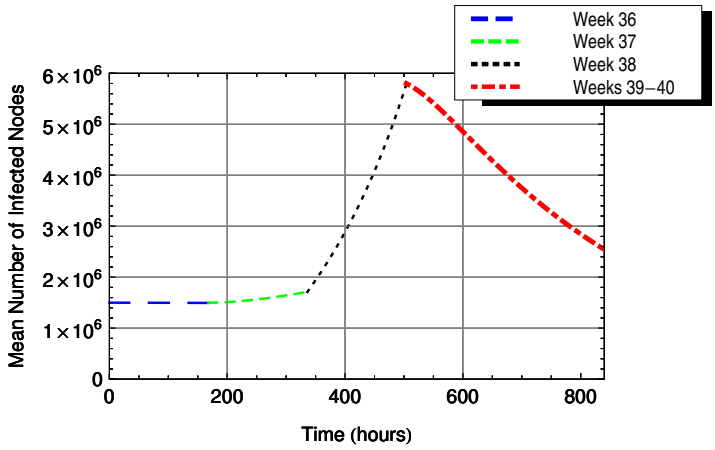


Figure 4.4.3.: FourLakeRiders botnet: size evolution of the number of Infected nodes estimated using the SIC Model.

number of *Infected* nodes are estimated using the SIC model as well, as depicted in Fig. 4.4.3. As may be seen, during both expansion and shrinkage, our results follow quite well the reported data.

4.5. Numerical Analysis

In this section, we present some numerical results to further illustrate the usefulness of the SIC model. First, we briefly introduce some parameter estimation techniques which help with the use of the SIC model. The first set of numerical results are with regard to the analysis of the initial unhindered expansion of a botnet. We then show how the SIC model could help with the evaluation and comparison of mitigation strategies. Botnet size standard deviation and utilization of Basic Reproduction Number are then depicted and examined next. We conclude this section by examining the developed analytical link between the SIC model and the P2P botnet mitigation strategies through an analysis of a random sybil attack on a P2P botnet. Throughout this section, we plot the previously-derived analytical results by assigning values to various parameters (λ_1, λ_2, λ_{r1}, λ_{r2}, and λ_a), all with the unit of nodes/time unit (time unit can be hour, day, week, or any other period). The plotted results

are therefore general as parameter values may be assumed to be nodes per any time unit and then the plotted time-dependent performance measures will be interpreted as functions of that time unit.

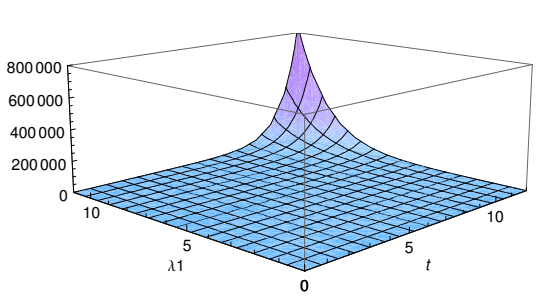
4.5.1. Model's Parameter Estimation Techniques

Using the SIC model, the botnet size estimation problem has been reduced from having to estimate the global size of the botnet to the estimation of the model's parameters (λ_1 and λ_2) which requires only local knowledge. On the other hand, values for λ_{r1}, λ_{r2}, and λ_a depend on the type of disinfection and attack on the botnet; as the mitigation strategies are being conducted by the security experts, they will be able to reliably choose values for these latter parameters.

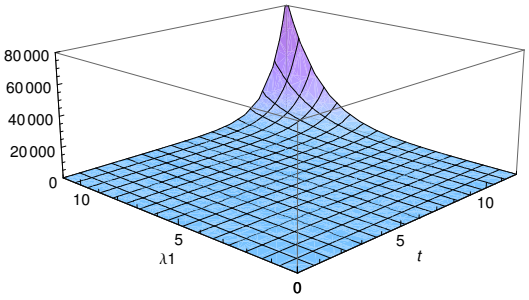
As a starting point, we would suggest a consideration of the following methods when trying to estimate values for λ_1 and λ_2: (1) real botnet size measurements, if available, can be used to estimate the parameter values (as done in Section 4.4); (2) local measurements through Honeynet log analysis [42], for example; and (3) a statistical approach to botnet virulence estimation (vulnerability and infection rates estimation) [65].

4.5.2. Initial Unhindered Botnet Expansion

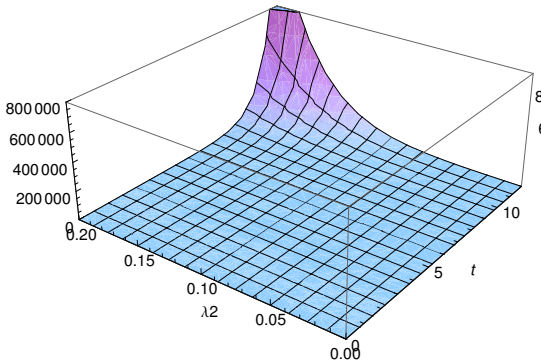
We first examine the unhindered botnet expansion that happens when the botnet first appears. In Fig. 4.5.1, we consider a 12-time-unit period during which the botnet expands. In this initial phase, there is neither any attack on the botnet, nor any removal (disinfection) from *Infected*/*Connected* stages; hence we set $\lambda_{r1} = \lambda_{r2} = \lambda_a = 0$. We choose $\lambda_1 = 7$ and $\lambda_2 = 0.1$ as the center values for these parameters; these values are based on the values derived from the analysis of GreenAlienRiders botnet (cf. Fig. 4.4.1). We then examine how the mean values of the number of nodes in *Infected* stage and *Connected* stage (botnet size) would change over this initial expansion period by varying the parameter values in the following ranges: $0 \leq \lambda_1 \leq 11$ and $0 \leq \lambda_2 \leq 0.2$. In Figs. 4.5.1a and 4.5.1b, we set



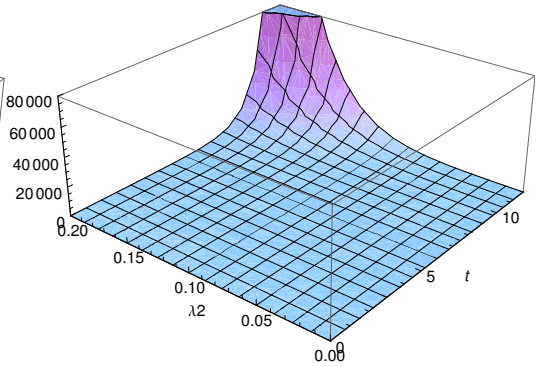
(a) $E_{u1}(t), \lambda_2 = 0.1$



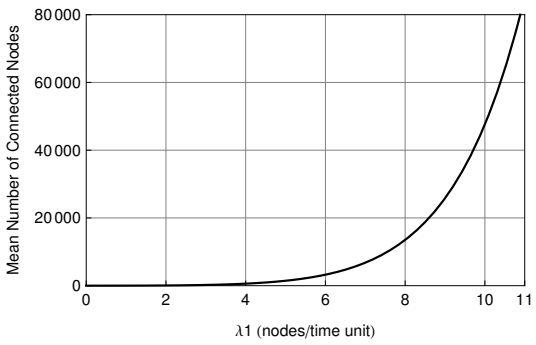
(b) $E_{u2}(t), \lambda_2 = 0.1$



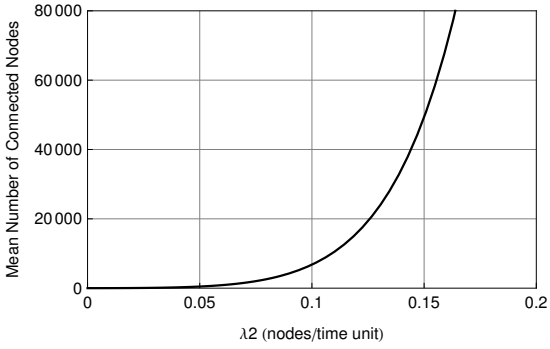
(c) $E_{u1}(t), \lambda_1 = 7$



(d) $E_{u2}(t), \lambda_1 = 7$



(e) $E_{u2}(t = 12), \lambda_2 = 0.1$



(f) $E_{u2}(t = 12), \lambda_1 = 7$

Figure 4.5.1.: SIC model: initial unhindered botnet expansion. Mean number of nodes in *Infected* stage ($E_{u1}(t)$) and *Connected* stage ($E_{u2}(t)$).

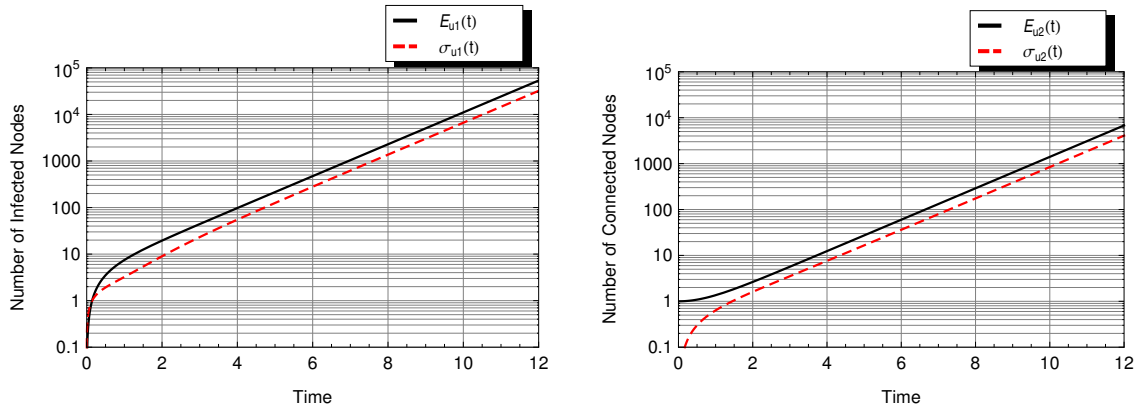
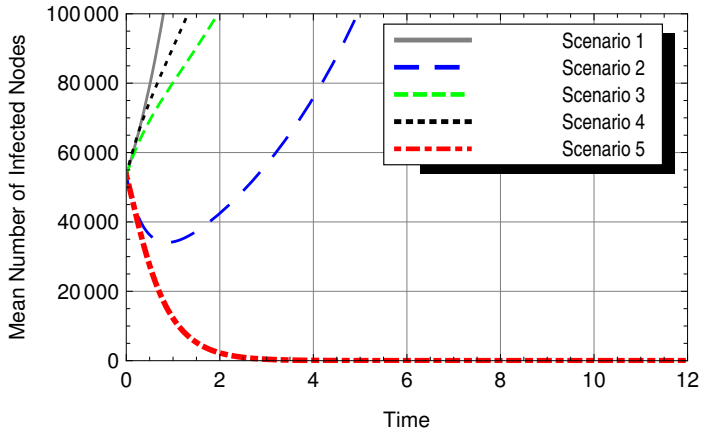


Figure 4.5.2.: SIC model: initial unhindered botnet expansion. Mean and standard deviation of the number of nodes in *Infected* and *Connected* stages.

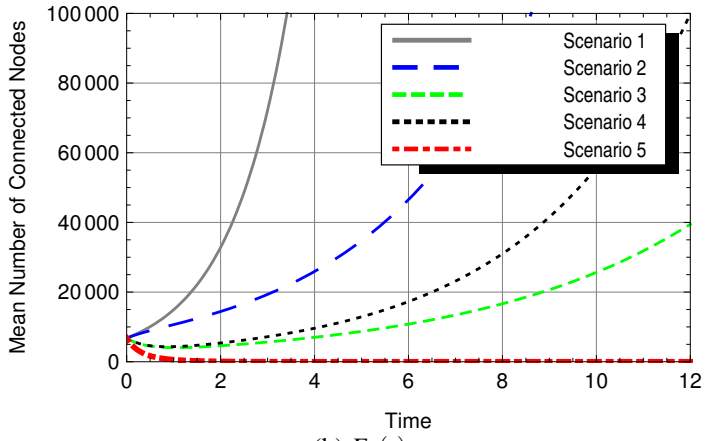
$\lambda_2 = 0.1$ and examine the change of mean values over time by varying λ_1 over $[0, 11]$ (initial state values: $E_{u1}(0) = \bar{k_1} = 0, E_{u2}(0) = \bar{k_2} = 1$; the subscript **u** refers to the **U**nhindered expansion.). In Figs. 4.5.1c and 4.5.1d, on the other hand, we set $\lambda_1 = 7$ and examine the change of mean values over time by varying λ_2 over $[0, 0.2]$. Slicing Figs. 4.5.1b and 4.5.1d at $t = 12$, Figs. 4.5.1e and 4.5.1f closely show how mean numbers would change over the respective ranges of values for λ_1 and λ_2. Finally, Fig. 4.5.2 shows the means along with the standard deviations ($\lambda_1 = 7, \lambda_2 = 0.1$).

4.5.3. Comparison of Mitigation Strategies

One of the main advantages of the SIC model is that it enables the security experts to compare and analyze mitigation strategies *before* deployment. In this sub-section, we study the case where botnet faces attack and/or removal (disinfection) and observe how severe these interventions must be in order to contain or dismantle the botnet. In all scenarios, we assign $\lambda_1 = 7$ and $\lambda_2 = 0.1$; their choice has no bearing on the following analysis regarding λ_{r1}, λ_{r2}, and λ_a. Further, we assume the mean number of *Infected* nodes and *Connected* nodes to be as follows: $E_1(0) = \bar{k_1} = 53484$ and $E_2(0) = \bar{k_2} = 6786$; these values are determined from Fig. 4.5.2 at $t = 12$ when $\lambda_1 = 7$ and $\lambda_2 = 0.1$. We can then proceed to



(a) $E_1(t)$



(b) $E_2(t)$

Figure 4.5.3.: SIC model: comparison of mitigation strategies. Mean number of nodes in *Infected* stage ($E_1(t)$) and *Connected* stage ($E_2(t)$).

analyze how this particular botnet would react to different mitigation strategies.

In Fig. 4.5.3, the solid line depicts the scenario where there are no mitigation strategies and the number of Infected nodes and the botnet size continue to increase. Dotted/dashed lines denote scenarios under which different values chosen for λ_{r1}, λ_{r2}, and λ_a result in different trajectories for the mean[3]. In Fig. 4.5.3a, we observe that the mean eventually goes to zero in only one scenario, i.e., when all three strategies are employed at the same

[3]Initial state values: $E_1(0) = \bar{k_1} = 53484, E_2(0) = \bar{k_2} = 6786$; Parameter values: $\lambda_1 = 7, \lambda_2 = 0.1$; Scenario 1: unhindered expansion ($\lambda_{r1} = 0, \lambda_{r2} = 0, \lambda_a = 0$); Scenario 2: only removal of Infected nodes ($\lambda_{r1} = 2, \lambda_{r2} = 0, \lambda_a = 0$); Scenario 3: only removal of Connected nodes ($\lambda_{r1} = 0, \lambda_{r2} = 2, \lambda_a = 0$); Scenario 4: only attack on botnet ($\lambda_{r1} = 0, \lambda_{r2} = 0, \lambda_a = 2$); Scenario 5: three strategies simultaneously ($\lambda_{r1} = 2, \lambda_{r2} = 2, \lambda_a = 2$). All λ parameters are nodes/time unit.

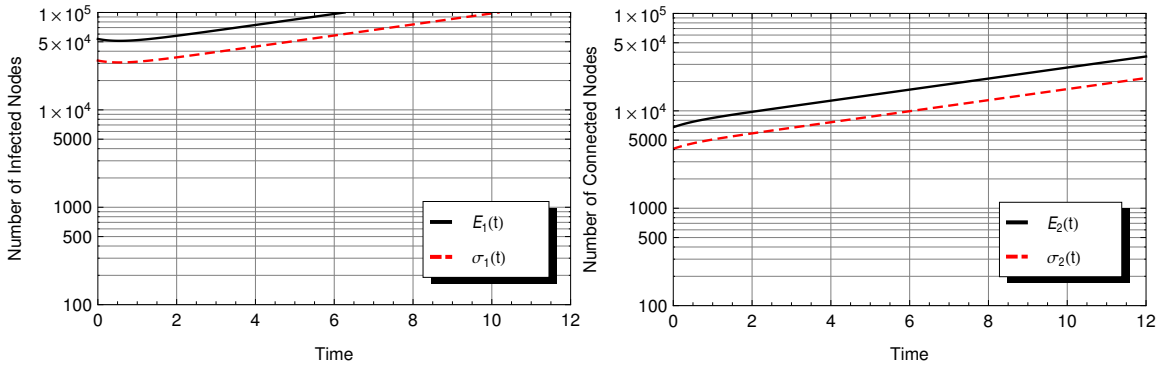
time. Note that a large enough value chosen for λ_{r1} would make the mean number of nodes in *Infected* stage go to zero as well. Fig. 4.5.3b depicts the same scenarios as in Fig. 4.5.3a, but this time, the mean is for the nodes in *Connected* stage (botnet size). In this particular case, we observe that the mean number of nodes in *Connected* stage also eventually goes to zero in only one scenario, i.e., when all three strategies are employed at the same time.

We can therefore state that, all things being equal, removal/disinfection from *Connected* stage (λ_{r2}) has the most effect on containing the size of the botnet (nodes in *Connected* stage). Further, we intuitively deduce that it would be less costly to combat a botnet if we implement all three strategies at the same time, as we can choose moderate disinfection/attack rates. Concentrating on a single strategy (disinfection or attack) would mean that we need to choose a very high rate to achieve a comparable effect. Having to choose a high rate is usually associated with high cost in the real world (e.g., the plan of malware removal from near 100% of computers is either infeasible or extremely costly to implement).

4.5.4. Standard Deviation and Basic Reproduction Number

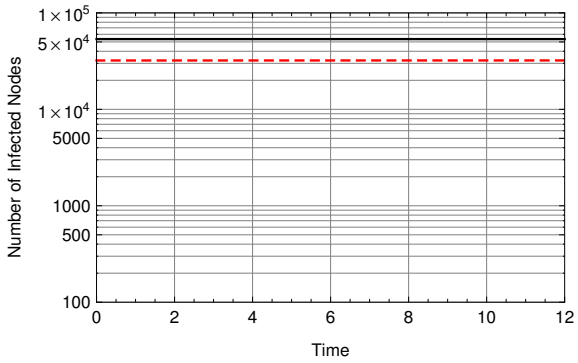
In Fig. 4.5.4, we draw the mean along with the standard deviation in each sub-figure. Since we consider that all mitigation strategies are being implemented, the sub-figures of Fig. 4.5.4 would be comparable to Fig. 4.5.3, as the chosen initial state values (values for $\bar{k_1}$ and $\bar{k_2}$) are the same. Furthermore, in Fig. 4.5.4, we use the derived formula for Basic Reproduction Number (R_0) to choose values for different parameters in a way that leads to the size of the botnet shrinking (bottom sub-figs.), remaining constant (center sub-figs.) or growing (top sub-figs.). To achieve this, we choose sample values for various parameters (except for λ_{r2}) and for R_0; therefore, the value of λ_{r2} would be determined in order to satisfy (4.3.13)[4].

[4]Parameter values are as follows: $\bar{k_1} = 53484$, $\bar{k_2} = 6786$, $\lambda_1 = 7$, $\lambda_2 = 0.1$, and $\lambda_a = 0.2$ for all sub-figures; for bottom sub-figures: $R_0 = 0.8, \lambda_{r1} = 1$, and $\lambda_{r2}(\text{determined}) = 0.8227$; for center sub-figures: $R_0 = 1, \lambda_{r1} = 0.8135$, and $\lambda_{r2}(\text{determined}) = 0.5880$; and for top sub-figures: $R_0 = 1.2, \lambda_{r1} = 1$, and $\lambda_{r2}(\text{determined}) = 0.2545$. All λ parameters are nodes/time unit.

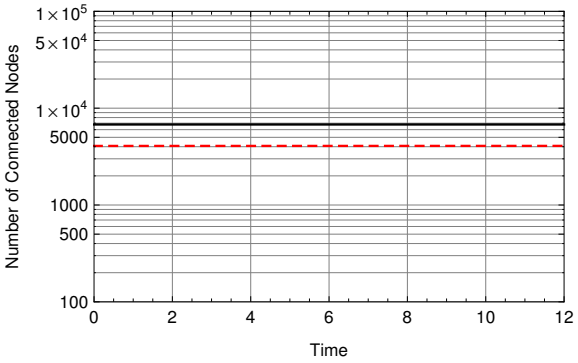


(a) $R_0 = 1.2$

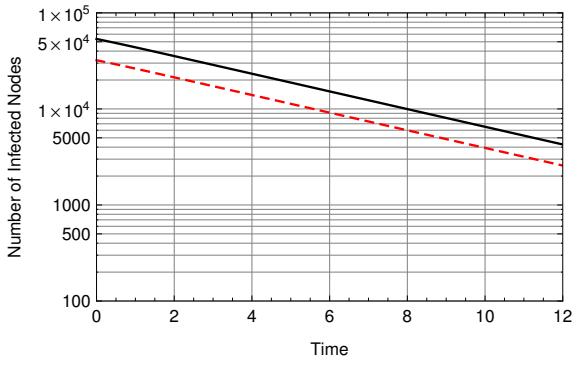
(b) $R_0 = 1.2$



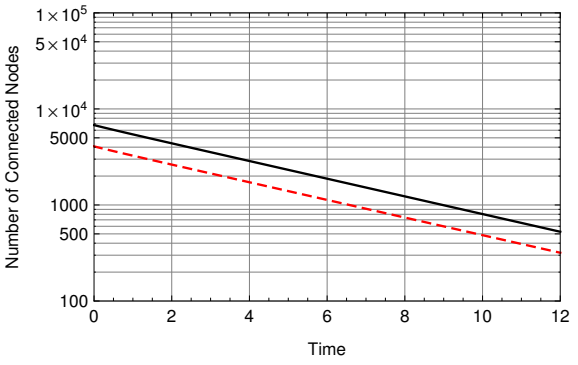
(c) $R_0 = 1.0$



(d) $R_0 = 1.0$



(e) $R_0 = 0.8$



(f) $R_0 = 0.8$

Figure 4.5.4.: SIC model (standard deviation and R_0): number of nodes in *Infected* stage (left sub-figs.) and *Connected* stage (right sub-figs.).

4.5.5. Random Sybil Attack on DHT-based P2P Botnets

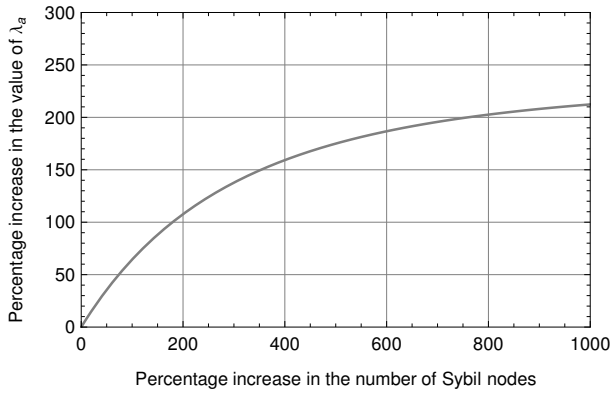


Figure 4.5.5.: SIC model: relationship between the attack rate (λ_a) and the number of sybils

Finally, we provide a numerical analysis of the developed relationship between the SIC model's attack rate (λ_a) and the number of sybils inserted in the P2P botnet. The analysis will be the case of adding the sybils at $t = 0$ in Fig. 4.5.3, assuming an instantaneous effect on the P2P botnet, and examining the situation in the next Δt. The numerical result is derived from (4.3.16) and depicted in Fig. 4.5.5[5]. The figure demonstrates the relationship between the percentage increase in the number of inserted sybils and the resulting percentage increase in the value of λ_a. The demonstrated relationship leads to the following insight: once the sybil attack is underway, the value of n_s is known and the resulting λ_a can be measured. The security expert can then determine, for example, how many sybils should be added in order to arrive at a desired λ_a to have the intended mitigation effect.

As summary, we remind that the contribution of this chapter is twofold: (1) the SIC botnet model which captures the key node stages relevant to botnets; we derived important results such as mean and variance of the number of nodes in different stages based on this model; and (2) development of a link between a botnet lifecycle/propagation/population model (the SIC model) and mitigation strategies aimed at DHT-based P2P botnets; with this analytical link, a security expert would be able to evaluate different mitigation strate-

[5]Initial $n_s = 1000$, $n = \bar{k_2} = 6786$, and $b = 5$. As the size of botnet changes with time, it is necessary to update the respective calculated values in regular intervals to keep a close approximation.

gies (disinfection, sybil attack, index poisoning, etc.) prior to their implementation. The developed link leads to a two-step, or recursive, analysis process: (1) examining the effect of the chosen λ_a on the botnet size based on (4.3.4) for the means; and (2) examining the relationship between a change of λ_a and the associated change in the number of sybils based on (4.3.16). In the next chapter, we integrate these two steps which entails changes to the SIC model itself and leads to an analytical model specific to DHT-based P2P botnets, the SIC-P2P model.

5. The SIC-P2P Botnet Model

5.1. Introduction

The general SIC model was proposed, developed, and analyzed in the previous chapter. In this chapter, we extend the proposed SIC model and customize it so as to be able to thoroughly analyze the Distributed Hash Table (DHT)-based P2P botnets; this extended SIC model is therefore named **SIC-P2P**. The SIC-P2P model is a Continuous-Time Markov Chain (CTMC) model which allows evaluation of botnet mitigation strategies such as disinfections of nodes and attacks on botnet's C&C mechanism. The model, contrary to most earlier deterministic analytical botnet models, is a stochastic model properly capturing the stochastic nature of population size changes. As DHT-based P2P botnets are especially resilient and perhaps pose the most significant threat, we believe the SIC-P2P model may prove to be a valuable asset in the toolkit of security experts who intend to analyze in depth attacks such as index poisoning and sybil attack which are the two most common attacks against P2P networks, including P2P botnets. Until now, the insight gained from the use of the SIC-P2P model could have been obtained only through large-scale, time-consuming, and expensive simulations and testbed experiments.

The chapter is organized as follows: in Section 5.2, we explain in depth the proposed model by examining the CTMC model. Section 5.3 then provides a thorough performance modeling of the SIC-P2P model by first examining the differential-difference probability flow equations resulting from the CTMC model and then moving on to derive closed-form

expressions for the time-dependent means and variances/standard deviations, finishing with the derivation of the Basic Reproduction Number. Section 5.4 then presents the developed analytical link between the SIC-P2P botnet lifecycle model and the real-world attacks against DHT-based P2P botnets; attacks such as index poisoning and sybil attacks. After the analytical results, Section 5.5 then provides some numerical results, shedding light on possible uses of the SIC-P2P model and the kinds of insight that can be drawn.

5.2. The SIC-P2P Model

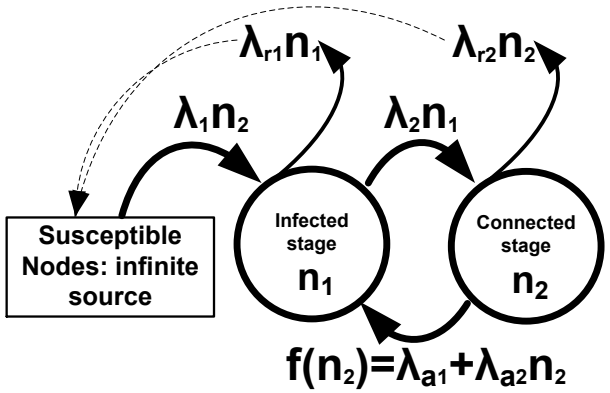


Figure 5.2.1.: SIC-P2P model: Inter-stage-rate diagram

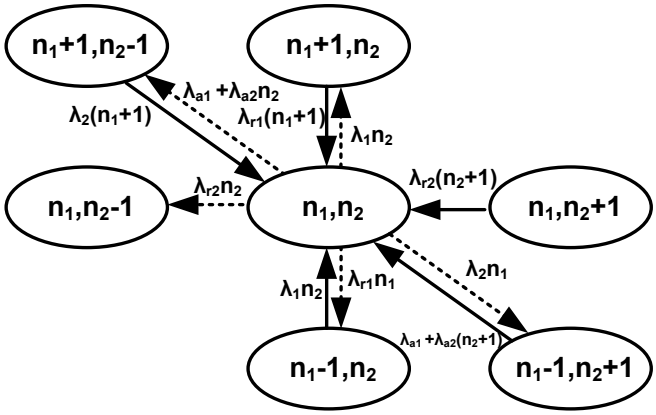


Figure 5.2.2.: SIC-P2P model: State-transition-rate diagram

In this section, we present the SIC-P2P botnet lifecycle model which is an extension of the SIC model. Like the SIC model, the SIC-P2P model accounts for three node stages:

Susceptible, *Infected*, and *Connected*. In Fig. 5.2.1, we show the stages of the model and the transitions between the stages. As shown in the figure, we let n_1 and n_2 denote the number of nodes in *Infected* and *Connected* stages, respectively, and the state of the system is represented by the vector (n_1, n_2). In Fig. 5.2.2, we show all the transitions from and to state (n_1, n_2).

Similar to the SIC model, in the SIC-P2P model, all nodes are initially considered to be *Susceptible*, i.e., they are healthy, but might be *Infected* in the lifetime of the botnet. As botmasters use multitude of methods and infection vectors to infect (and re-infect) the nodes, even after disinfection and/or patching, nodes remain *Susceptible* to infection. In the *Infected* stage, a node is usually not able to engage in illicit activities, as it is not yet part of the botnet and does not receive the botmaster's C&C messages. The initial infection usually only serves to connect the node to the botnet and pass the node to the *Connected* stage. In this latter stage, a node is fully operational as a botnet node and can act on botmaster commands received from either a C&C server or a peer in the P2P botnet.

In the SIC-P2P model, we consider that each node in the botnet (nodes in *Connected* stage) infects one *Susceptible* node (increases n_1 by one) with probability $\lambda_1 \Delta t + o(\Delta t)$ in any Δt interval (cf. Fig. 5.2.1). Thus the time interval for a Connected node to infect a Susceptible node is exponentially distributed with parameter λ_1. The transition rate between Susceptible and Infected stages is therefore given by $\lambda_1 n_2$. Further, each Infected node can transition to Connected stage (which increases n_2 and decreases n_1) with probability $\lambda_2 \Delta t + o(\Delta t)$ in any Δt interval. Finally, there is a transition rate $(\lambda_{a1} + \lambda_{a2} n_2)$ from Connected stage to Infected stage. This transition rate represents an attack on the botnet, attacks such as index poisoning and sybil attacks in the case of P2P botnets. Under such attacks, nodes do not transition back to Susceptible stage; they just lose the ability to communicate and might be able to reconnect again. We further assume the rate of disinfection of nodes which are in Infected stage and Connected stage to be $\lambda_{r1} n_1$ and $\lambda_{r2} n_2$, respectively.

It may be seen that the time interval for the transition of a node from one stage to the

next one is exponentially distributed with different parameters. Further, the model assumes the number of nodes in the *Susceptible* stage to be infinite. This is a reasonable assumption considering the total number of devices that are connected to the Internet today. All the aforementioned assumptions, including the choice of node stages as well as the transition rates between the stages, are similar to the SIC model and were explained extensively and justified earlier. Note that the model's assumptions are consistent with how the Storm botnet actually operated [47] as well as how previous models [48, 49] of the Storm botnet have been designed. As a result, we believe that we have a realistic model, which leads us to two-dimensional Markovian birth-death processes. Using the model, we can study the size evolution of a botnet as well as effectiveness of mitigation strategies by monitoring the number of nodes that are in Infected and Connected stages at any given time.

The node stages as well as the transition rates among stages between the SIC and the SIC-P2P model are the same, except the transition rate from the *Connected* stage to the *Infected* stage (which has changed from $\lambda_a n_2$ in the SIC model to $\lambda_{a1} + \lambda_{a2} n_2$ in the SIC-P2P model). This seemingly minor difference, however, propagates throughout all the steps of derivations and leads to significantly different, more complicated formulas. This more general transition rate leads to the SIC-P2P model being able to capture and analyze attacks on P2P botnets which is the main contribution of this chapter. We will concentrate on DHT-based P2P botnets and incorporate the real-world attacks on such botnets in the lifecycle model. Specifically, we are concerned with determining the values of λ_{a1} and λ_{a2} in the SIC-P2P model. In Section 5.4, we develop a methodology to suggest values for λ_{a1} and λ_{a2} based on known mitigation strategies against DHT-based P2P botnets.

5.3. Performance Modeling

In this section, we provide a comprehensive performance analysis of the SIC-P2P model. We first show the probability flow differential-difference equations set, derived based on

the 2-dimensional CTMC which was shown in the model's inter-stage-rate diagram. In order to solve this equations set, we reduce it to a partial differential equation (PDE) of the probability generating function (PGF). Next, we derive closed-form expressions for the time-dependent means and variances from the aforementioned PDE of the PGF. Finally, we derive the Basic Reproduction Number, a parameter widely used in epidemiology, which helps to find the relationship between various transition rate parameters of the SIC-P2P model in order to achieve a botnet size that is constant over time.

5.3.1. Differential-difference Equations and the PDE

According to the inter-stage-rate diagram depicted in Fig. 5.2.1, for the birth-death process of the SIC-P2P model, the equations for the rate of change of probabilities are as follows:

$$
\begin{cases}
\dfrac{dP_{n_1,n_2}(t)}{dt} = \lambda_1 n_2 P_{n_1-1,n_2}(t) + \lambda_{r1}(n_1+1)P_{n_1+1,n_2}(t) \\
\qquad + \lambda_{r2}(n_2+1)P_{n_1,n_2+1}(t) + \lambda_2(n_1+1)P_{n_1+1,n_2-1}(t) \\
\qquad + [\lambda_{a1} + \lambda_{a2}(n_2+1)]P_{n_1-1,n_2+1}(t) \\
\qquad - (\lambda_1 n_2 + \lambda_{r1} n_1 + \lambda_{r2} n_2 + \lambda_2 n_1 + \lambda_{a1} + \lambda_{a2} n_2)P_{n_1,n_2}(t) \\
\qquad\qquad \langle n_1 > 0, n_2 > 0 \rangle \, (a) \\
\dfrac{dP_{0,n_2}(t)}{dt} = \lambda_{r1} P_{1,n_2}(t) + \lambda_{r2}(n_2+1)P_{0,n_2+1}(t) + \lambda_2 P_{1,n_2-1}(t) \\
\qquad - (\lambda_1 n_2 + \lambda_{r2} n_2 + \lambda_{a1} + \lambda_{a2} n_2)P_{0,n_2}(t) \\
\qquad\qquad \langle n_1 = 0, n_2 > 0 \rangle \, (b) \\
\dfrac{dP_{n_1,0}(t)}{dt} = \lambda_{r1}(n_1+1)P_{n_1+1,0}(t) + \lambda_{r2} P_{n_1,1}(t) \\
\qquad + (\lambda_{a1} + \lambda_{a2})P_{n_1-1,1}(t) - (\lambda_{r1} n_1 + \lambda_2 n_1)P_{n_1,0}(t) \\
\qquad\qquad \langle n_1 > 0, n_2 = 0 \rangle \, (c) \\
\dfrac{dP_{0,0}(t)}{dt} = \lambda_{r1} P_{1,0}(t) + \lambda_{r2} P_{0,1}(t) \\
\qquad\qquad \langle n_1 = 0, n_2 = 0 \rangle \, (d)
\end{cases}
\tag{5.3.1}
$$

The analysis of the SIC-P2P model runs parallel to that of the SIC model presented in the previous chapter. In order to solve (5.3.1) and derive the probability distribution $P_{n_1,n_2}(t)$, as mentioned for the SIC model, a known method is to transform the equations of probability flows to a PDE of the PGF which can be tackled using known methods to solve PDEs. Let us denote $P(z_1,z_2,t)$ as the PGF of the probability distribution $P_{n_1,n_2}(t)$ which is given by $P(z_1,z_2,t) = \sum_{n_1=0}^{\infty}\sum_{n_2=0}^{\infty} P_{n_1,n_2}(t) z_1^{n_1} z_2^{n_2}$.

The initial probability distribution is denoted by $P_{k_1,k_2}(0)$. Here, we assume that the initial number of nodes in each stage is constant (k_1,k_2). Though the initial derivations are conditional, we will suppress the conditions for simplicity in expressing the PDE. Derivations of means and variances, however, properly take this aspect into account.

We multiply each of the equations in (5.3.1) by $z_1^{n_1} z_2^{n_2}$, sum over the respective ranges of n_1 and n_2, and then add them together. After some simplifications and manipulations (see Appendix C.1 for a detailed derivation), we arrive at the following PDE of the PGF:

$$\begin{aligned}
&(\lambda_{r1} + \lambda_2 z_2 - \lambda_{r1} z_1 - \lambda_2 z_1)\frac{\partial P(z_1,z_2,t)}{\partial z_1} \\
+(\lambda_1 z_1 z_2 + \lambda_{r2} + \lambda_{a2} z_1 - \lambda_1 z_2 - \lambda_{r2} z_2 - \lambda_{a2} z_2)&\frac{\partial P(z_1,z_2,t)}{\partial z_2} - \frac{\partial P(z_1,z_2,t)}{\partial t} \\
&= \lambda_{a1}(1-\frac{z_1}{z_2})\left(P(z_1,z_2,t) - \sum_{n_1=0}^{\infty} P_{n_1,0}(t) z_1^{n_1}\right) \qquad (5.3.2)
\end{aligned}$$

The usual use scenario of the SIC-P2P model is when the botnet is already present on the Internet, potentially has a considerable size and, therefore, is considered to be an important threat which needs to be mitigated; hence the evalution of mitigation strategies using the SIC-P2P model. Considering this scenario, we can see that the probability of the botnet size (n_2) being zero, or near zero, is negligible, as the assumption is that the botnet is indeed present and has a considerable size; therefore, $\sum_{n_1=0}^{\infty} P_{n_1,0}(t) \approx 0$.

As $|\sum_{n_1=0}^{\infty} P_{n_1,0}(t) z_1^{n_1}| \leq \sum_{n_1=0}^{\infty} P_{n_1,0}(t)$, for the usual use scenario considered in this chapter,

the PDE of (5.3.2) (its right hand side, in particular) can therefore be reduced as follows:

$$\begin{aligned}(\lambda_{r1}+\lambda_2 z_2-\lambda_{r1}z_1-\lambda_2 z_1)\frac{\partial P(z_1,z_2,t)}{\partial z_1}\\ +(\lambda_1 z_1 z_2+\lambda_{r2}+\lambda_{a2}z_1-\lambda_1 z_2-\lambda_{r2}z_2-\lambda_{a2}z_2)\frac{\partial P(z_1,z_2,t)}{\partial z_2}-\frac{\partial P(z_1,z_2,t)}{\partial t}\\ =\lambda_{a1}(1-\frac{z_1}{z_2})P(z_1,z_2,t)\end{aligned} \tag{5.3.3}$$

The solution of (5.3.3), and in turn the probability distribution of the SIC-P2P model, cannot be obtained at this moment, however, due to the complexity of the equation; it may be obtained in the future as per discussion provided in Appendix B.2. In the meantime, we can obtain from the PDE important results such as the moments of the number of nodes in *Infected* and *Connected* stages, as presented next.

5.3.2. Derivation of Means and Variances

Let $E_t[n_1]$ and $E_t[n_2]$ denote the mean number of nodes that are in *Infected* and *Connected* stages at time t, respectively, then:

$$E_t[n_1]=\frac{\partial P(z_1,z_2,t)}{\partial z_1}\mid_{z_1=z_2=1}, E_t[n_2]=\frac{\partial P(z_1,z_2,t)}{\partial z_2}\mid_{z_1=z_2=1} \tag{5.3.4}$$

We take the derivatives of the PDE given in (5.3.3) with respect to z_1 and z_2, separately. By setting $z_1=z_2=1$ in each resulting equation, we arrive at a set of ODEs of $E_t[n_1]$ and $E_t[n_2]$. To emphasize the time dependency of the means from here on, we will denote $E_t[n_1]$ and $E_t[n_2]$ by $E_1(t)$ and $E_2(t)$, respectively. The details of the aforementioned steps are provided in Appendix C.2; the set of ODEs is as follows:

$$\begin{cases}\frac{dE_1(t)}{dt}= & -(\lambda_2+\lambda_{r1})E_1(t)+(\lambda_1+\lambda_{a2})E_2(t)+\lambda_{a1}\\ \frac{dE_2(t)}{dt}= & \lambda_2 E_1(t)-(\lambda_{r2}+\lambda_{a2})E_2(t)-\lambda_{a1}\end{cases} \tag{5.3.5}$$

$E_1(t)$ and $E_2(t)$ can then be derived as follows (details in Appendix C.2):

$$
\begin{aligned}
E_1(t) = \Big[& \lambda_{T4}\left(2\left(-1+e^{t\lambda_{T1}}\right)\overline{k_2}\left(\lambda_1+\lambda_{a2}\right)\lambda_{T2}\right) \\
& +\lambda_{T4}\overline{k_1}\left(e^{t\lambda_{T1}}\left(-\lambda_2+\lambda_{a2}-\lambda_{r1}+\lambda_{r2}+\lambda_{T1}\right)\right) \\
& +\lambda_{T4}\overline{k_1}\left(\lambda_2-\lambda_{a2}+\lambda_{r1}-\lambda_{r2}+\lambda_{T1}\right)\lambda_{T2} \\
& +\lambda_{T4}\lambda_{a1}e^{t\lambda_{T1}}\left(-\lambda_{r2}^2+\lambda_2\lambda_{r2}-\lambda_{a2}\lambda_{r2}\right) \\
& +\lambda_{T4}\lambda_{a1}e^{t\lambda_{T1}}\left(\lambda_{r1}\lambda_{r2}-\lambda_{T1}\lambda_{r2}+2\lambda_{a2}\lambda_{r1}\right) \\
& +\lambda_{T4}\lambda_{a1}\left(\lambda_{r2}^2-\lambda_2\lambda_{r2}+\lambda_{a2}\lambda_{r2}-\lambda_{r1}\lambda_{r2}\right) \\
& +\lambda_{T4}\lambda_{a1}\left(2e^{\frac{1}{2}t\left(\lambda_{T3}+\lambda_{T1}\right)}\lambda_{T1}\lambda_{r2}-\lambda_{T1}\lambda_{r2}-2\lambda_{a2}\lambda_{r1}\right) \\
& +\lambda_{T4}\lambda_{a1}\lambda_1\left(e^{t\lambda_{T1}}\left(-\lambda_2+\lambda_{a2}+\lambda_{r1}+\lambda_{r2}+\lambda_{T1}\right)\right) \\
& +\lambda_{T4}\lambda_{a1}\lambda_1\left(\lambda_2-\lambda_{a2}-\lambda_{r1}-\lambda_{r2}-2e^{\frac{1}{2}t\left(\lambda_{T3}+\lambda_{T1}\right)}\lambda_{T1}\right) \\
& +\lambda_{T4}\lambda_{a1}\lambda_1\lambda_{T1}\Big] / \left(2\lambda_{T2}\sqrt{{\lambda_{T3}}^2-4\lambda_{T2}}\right)
\end{aligned}
\tag{5.3.6}
$$

$$
\begin{aligned}
E_2(t) = \Big[& \lambda_{T5}\left(2\left(1-e^{t\lambda_{T1}}\right)\overline{k_1}\lambda_2\lambda_{T2}\right) \\
& -\lambda_{T5}\overline{k_2}\lambda_{T2}\left(\left(-1+e^{t\lambda_{T1}}\right)\lambda_{2+\lambda_{a2}-\lambda_{r1}+\lambda_{r2}+\lambda_{T1}}\right) \\
& -\lambda_{T5}\overline{k_2}\lambda_{T2}e^{t\lambda_{T1}}\left(-\lambda_{a2}+\lambda_{r1}-\lambda_{r2}+\lambda_{T1}\right) \\
& -\lambda_{T5}\lambda_{a1}\left(2\left(-1+e^{t\lambda_{T1}}\right)\lambda_1\lambda_2-\lambda_2\lambda_{r1}+\lambda_{a2}\lambda_{r1}\right) \\
& -\lambda_{T5}\lambda_{a1}\left(2\lambda_2\lambda_{r2}-\lambda_{r1}^2+\lambda_{r1}\lambda_{r2}+\lambda_{r1}\lambda_{T1}\right) \\
& +2\lambda_{T5}\lambda_{a1}e^{\frac{1}{2}t\left(\lambda_{T3}+\lambda_{T1}\right)}\lambda_{r1}\lambda_{T1} \\
& -\lambda_{T5}\lambda_{a1}e^{t\lambda_{T1}}\left(\lambda_2\lambda_{r1}-\lambda_{a2}\lambda_{r1}+\lambda_{r1}^2-2\,\lambda_2\lambda_{r2}\right) \\
& -\lambda_{T5}\lambda_{a1}e^{t\lambda_{T1}}\left(-\lambda_{r1}\lambda_{r2}\right) \\
& -\lambda_{T5}\lambda_{a1}e^{t\lambda_{T1}}\left(\lambda_{r1}\lambda_{T1}\right)\Big] / \left(-2\lambda_{T2}\sqrt{{\lambda_{T3}}^2-4\lambda_{T2}}\right)
\end{aligned}
\tag{5.3.7}
$$

where:

$$\begin{aligned}
\lambda_{T1} =& \sqrt{\lambda_2^2 + 4\lambda_1\lambda_2 + 2\left(\lambda_{a2} + \lambda_{r1} - \lambda_{r2}\right)\lambda_2 + \left(\lambda_{a2} - \lambda_{r1} + \lambda_{r2}\right)^2} \\
\lambda_{T2} =& -\lambda_1\lambda_2 + \lambda_{a2}\lambda_{r1} + \left(\lambda_2 + \lambda_{r1}\right)\lambda_{r2} \\
\lambda_{T3} =& \lambda_2 + \lambda_{a2} + \lambda_{r1} + \lambda_{r2} \\
\lambda_{T4} =& \exp\left(-\frac{1}{2}t\left(\lambda_{T3} + \sqrt{{\lambda_{T3}}^2 - 4\lambda_{T2}}\right)\right) \\
\lambda_{T5} =& \exp\left(-\frac{1}{2}t\left(\lambda_{T3} + \lambda_{T1}\right)\right) \\
\overline{k_1}, \overline{k_2} =& E_1(t=0), E_2(t=0), \text{respectively.}
\end{aligned} \tag{5.3.8}$$

Next, we derive the variances from the PDE of the PGF in (5.3.3). We know:

$$\sigma_1^2(t) = E_t[n_1^2] - \left(E_1(t)\right)^2 \ , \ \sigma_2^2(t) = E_t[n_2^2] - \left(E_2(t)\right)^2 \tag{5.3.9}$$

where:

$$\begin{aligned}
E_t[n_1^2] &= \frac{\partial^2 P(z_1, z_2, t)}{\partial z_1^2} \mid_{z_1=z_2=1} + \frac{\partial P(z_1, z_2, t)}{\partial z_1} \mid_{z_1=z_2=1} \\
E_t[n_2^2] &= \frac{\partial^2 P(z_1, z_2, t)}{\partial z_2^2} \mid_{z_1=z_2=1} + \frac{\partial P(z_1, z_2, t)}{\partial z_2} \mid_{z_1=z_2=1}
\end{aligned} \tag{5.3.10}$$

We take the 2nd derivatives of the PDE in (5.3.3) with respect to z_1 and z_2, separately. Further, we take the derivative of the PDE with respect to z_1 and then with respect to z_2. By setting $z_1 = z_2 = 1$ in each resulting equation, we arrive at a set of ordinary differential equations solution of which will lead to closed-form expressions for the variances of the SIC-P2P model. The derivations and obtained expressions are extremely lengthy however; hence, they are provided in Appendix C.3 and [72] instead.

5.3.3. Basic Reproduction Number (R_0)

In this sub-section, we calculate an epidemiological threshold, i.e., the Basic Reproduction Number (R_0), which is a widely used parameter in epidemiology [52] as well as in the study of malware propagation. In the context of botnets, this number is the mean number of infections that any single botnet node can cause among the population of susceptible nodes. If $R_0 < 1$, the botnet will eventually disappear with probability one. If $R_0 > 1$, however, there is a probability that the botnet size will continue to increase exponentially. When $R_0 = 1$, the values of means remain constant as time goes by. Based on the ODEs of means in (5.3.5), R_0 can be written as follows:

$$R_0 = \frac{(\lambda_1 + \lambda_{a2})\bar{k_2} + \lambda_{a1}}{(\lambda_2 + \lambda_{r1})\bar{k_1}} = \frac{\lambda_2 \bar{k_1}}{(\lambda_{r2} + \lambda_{a2})\bar{k_2} + \lambda_{a1}} \tag{5.3.11}$$

Equations (5.3.11) have been derived from (5.3.5) by putting the right-hand-side terms contributing positively in the numerator and putting the right-hand-side terms contributing negatively in the denominator. Assuming that $\bar{k_1}$, $\bar{k_2}$, λ_1, and λ_2 are fixed and given (i.e., the aim is to evaluate different mitigation strategies), then the triplet $\left(\left\langle \lambda_{a1}, \lambda_{a2} \right\rangle, \lambda_{r1}, \lambda_{r2}\right)$ has one degree of freedom, i.e., if, for example, $\left\langle \lambda_{a1}, \lambda_{a2} \right\rangle$ is chosen, λ_{r1} and λ_{r2} will be fixed automatically according to (5.3.11).

The "Next Generation Matrix" method used in deriving R_0 for the SIC model in the previous chapter, as shown in (4.3.13), is not applicable to the SIC-P2P model, since λ_{a1} disappears in the differentiation involved in the method. We have therefore used the above method to derive R_0 for the SIC-P2P model which leads to a formula which is equivalent, albeit unlike, to the one mentioned in (4.3.13). As an example verification test, we can use the parameter values used to generate Figs. 4.5.4c and 4.5.4d which were: $R_0 = 1$, $\bar{k_1} = 53484$, $\bar{k_2} = 6786$, $\lambda_1 = 7$, $\lambda_2 = 0.1$, $\lambda_a = 0.2$, $\lambda_{r1} = 0.8135$, and $\lambda_{r2} = 0.5880$. Using these values for the parameters, by setting $\lambda_{a1} = 0$ and $\lambda_{a2} = \lambda_a$, both equations in (5.3.11) would be satisfied.

5.4. From Real-world Rate of Attack on P2P Botnets to $\lambda_{a1} + \lambda_{a2} n_2$

In this section, we present our novel analytical methodology to relate the attack rate in the SIC-P2P model (i.e., $\lambda_{a1} + \lambda_{a2} n_2$) to previously-derived analytical results by Wang et al. [18] regarding real-world attacks on DHT-based P2P botnets which were described in Subsection 2.3.3. They have provided formulas for index poisoning and sybil attacks that can guide us in choosing values for λ_{a1} and λ_{a2} in our model. Here, we treat the case of index poisoning; random and targeted sybil attacks have similar treatments and are detailed in Section 5.6. The obtained formula for the index poisoning attack was mentioned in (2.3.1) which is repeated here:

$$P_s = \left(1 - \frac{2^c n_p}{n}\right)^{\frac{\text{Log}_2[n]-c}{b}} \tag{5.4.1}$$

where, P_s is the probability that a botnet node obtains a real command; c is the number of first bits in common with a hash key (a configurable attack parameter); n_p is the number of nodes poisoned in the target zone (a configurable attack parameter); b is the number of bits improved per step for a lookup (set to a mid-range value of 5 in our study [18]); and n is the botnet size (this is the value of n_2 in the SIC-P2P model).

5.4.1. Definition of Attack Rate

Let us assume that a botnet node (a node in Connected stage) periodically initiates a new query to receive the new commands of the botmaster. Let us assume that this period is exponentially distributed with mean duration of τ units of time. n_2 denotes the number of nodes in the Connected stage (botnet size). Then, the time interval between consecutive queries generated by all botnet nodes will be exponentially distributed with parameter $\frac{n_2}{\tau}$. P_s denotes the probability that a botnet node will receive a real command. Then, the time interval for the transition of nodes from the Connected to Infected stage will be ex-

ponentially distributed with parameter $(1-P_s)\frac{n_2}{\tau}$. Thus, the total transition rate from the Connected to the Infected stage (i.e., the rate of attack on the botnet or *attack rate*) is given by:

$$\text{Attack Rate} = \frac{(1-P_s)n_2}{\tau} \tag{5.4.2}$$

5.4.2. Attack Rate: Taylor Series Approximation

In the expression for P_s, the only variable (considering the SIC-P2P model) is n (which is n_2, the botnet size). Our goal is to relate the $\lambda_{a1}+\lambda_{a2}n_2$ of the SIC-P2P model to the attack rate defined in (5.4.2). As the formula for P_s is too complex for use in our model, we proceed to obtain the Taylor series of P_s around k, as follows (first 2 terms): $P_s \approx T_1 + (n-k)T_2$, where:

$$\begin{aligned} T_1 = &\quad P_s\big|_{n=k} = \left(1-\frac{2^c n_p}{k}\right)^{\frac{\text{Log}[k]-c\text{Log}[2]}{b\text{Log}[2]}} \\ T_2 = &\quad \frac{d(P_s)}{dn}\Big|_{n=k} = T_1\left(\frac{\text{Log}\left[1-\frac{2^c n_p}{k}\right]}{bk\text{Log}[2]} - \frac{2^c n_p(\text{Log}[k]-c\text{Log}[2])}{bk\text{Log}[2]\,(2^c n_p-k)}\right) \end{aligned} \tag{5.4.3}$$

where k and n are $\bar{k_2}$ and n_2 in the SIC-P2P model, respectively; therefore: $P_s \approx T_1 + T_2(n-k) = T_1 + T_2(n_2-\bar{k_2})$.

Considering (5.4.2), we notice that the attack rate will be quadratic in n_2, i.e., the attack rate will not have the needed linear $\lambda_{a1}+\lambda_{a2}n_2$ form and would need the derivation of a second order PDE of PGF for the SIC-P2P model for which we can derive neither the probability distribution nor the mean/variance. On the other hand, we would need to test whether or not the first 2 terms of the Taylor series approximation are enough with *means* of n_1 and n_2, as a direct test with n_1 and n_2 would need the probability distribution. In order to address both aforementioned issues, let us replace n_2 with $E_2(t)$ and define the Mean Attack Rate as follows:

$$\text{Mean Attack Rate} = \frac{(1-\bar{P}_s)\bar{k_2}}{\tau} \tag{5.4.4}$$

where:

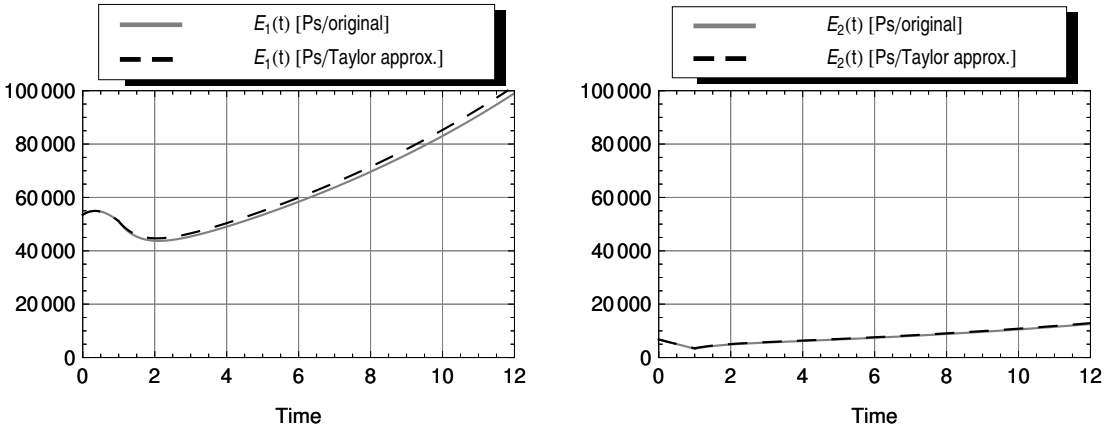
$$\bar{P}_s = \left(1 - \frac{2^c n_p}{E_2(t)}\right)^{\frac{\log_2(E_2(t))-c}{b}} \tag{5.4.5}$$

In (5.4.4), we have also used $(1-\bar{P}_s)\bar{k_2}$ instead of $(1-\bar{P}_s)E_2(t)$ which is an approximation. This latter approximation entails that attack rate can have a linear form rather than a quadratic form. The approximation is acceptable only if the value of $E_2(t)$ does not get too far from $\bar{k_2}$ in any analysis phase. If it does, then we will need to do the analysis in multiple phases, with each phase starting with an updated value for $\bar{k_2}$ which is equal to the last value of $E_2(t)$ in the previous phase (let us assume the following region for the appropriateness of the approximation in each phase: $0.2\bar{k_2} < E_2(t) < 5\bar{k_2}$). The approximation is equivalent to, for example, approximating x^2 with $10x$ as long as x does not get too far from 10. Note that in all figures presented later examining different attack scenarios, we have deliberately chosen attack parameters such that $E_2(t)$ does not get too far from $\bar{k_2}$ in order to avoid having to do the analysis in multiple phases.

We now need to determine whether the first two terms of the the Taylor series of $\bar{P}_s$, i.e., $\bar{P}_s \approx T_1 + T_2(E_2(t) - \bar{k_2})$, are a reasonable approximation. To start the evaluation, we numerically solve the ODEs of means with the expression for $\bar{P}_s$, i.e., (5.4.5), and the defined mean attack rate, i.e., (5.4.4), as follows. The ODEs of means were:

$$\begin{cases} \frac{dE_1(t)}{dt} = & -(\lambda_2 + \lambda_{r1})E_1(t) + (\lambda_1 + \lambda_{a2})E_2(t) + \lambda_{a1} \\ \frac{dE_2(t)}{dt} = & \lambda_2 E_1(t) - (\lambda_{r2} + \lambda_{a2})E_2(t) - \lambda_{a1} \end{cases} \tag{5.4.6}$$

We therefore need to solve:



(a) $E_1(t)$ (b) $E_2(t)$

Figure 5.4.1.: SIC-P2P model: $E_1(t)$ and $E_2(t)$ with the mean rate of attack being $\frac{(1-\bar{P}_s)\bar{k_2}}{\tau}$ (i.e., original P_s) and $\lambda_{a1}+\lambda_{a2}E_2(t)$ (i.e., Taylor approximation of P_s)

$$\begin{cases} \frac{dE_1(t)}{dt} = & -(\lambda_2+\lambda_{r1})E_1(t)+\lambda_1 E_2(t)+\frac{(1-\bar{P}_s)\bar{k_2}}{\tau} \\ \frac{dE_2(t)}{dt} = & \lambda_2 E_1(t)-\lambda_{r2}E_2(t)-\frac{(1-\bar{P}_s)\bar{k_2}}{\tau} \end{cases} \tag{5.4.7}$$

Compared to the ODEs of means in the SIC-P2P model, i.e., (5.4.6), the change is the replacement of $\lambda_{a1}+\lambda_{a2}E_2(t)$ with $\frac{(1-\bar{P}_s)\bar{k_2}}{\tau}$. By choosing some sample values for the parameters of the index poisoning attack, we proceed to numerically solve (5.4.7) and draw $E_1(t)$ and $E_2(t)$, as depicted in Figure 5.4.1 (solid lines). Note that, without loss of any generality, the mean duration of attack on the botnet will be set equal to the mean interarrival time of the queries initiated by botnet nodes to get the botmaster's commands. In this scenario, τ is the first time unit of the analysis period. After $t=\tau$, there is no new attack on the botnet and the botnet continues to expand.

We now investigate how each term of the Taylor series of $\bar{P}_s$ has contributed to the total value of the mean attack rate. With some chosen values for various parameters[1], the values of the terms of the Taylor series of $\bar{P}_s$ are as follows: $T_1 = 0.3544$, $T_2 = 6.503\times10^{-5}$. It therefore seems that using the first 2 terms of the Taylor series is a good approximation; we

[1]Parameter values (nodes/time unit): $\lambda_1=7$, $\lambda_2=0.1$,$\lambda_{r1}=0.7$, and $\lambda_{r2}=0.7$. Index poisoning parameter values: $c=4$, $n_p=190$, $b=5$, and $\tau=1$. Initial state values: $\bar{k_1}=53480$ and $\bar{k_2}=6786$.

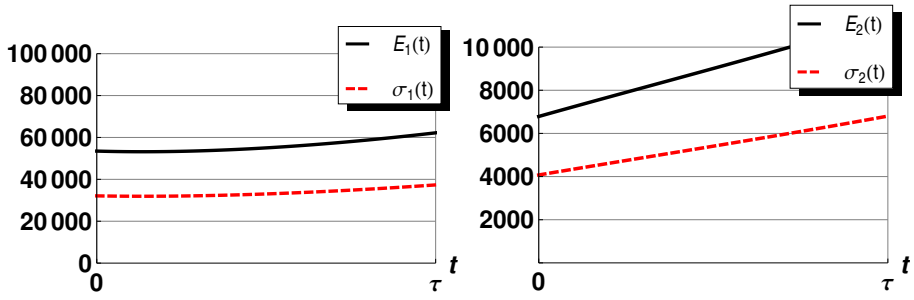
investigate this approximation as follows:

$$\begin{aligned}
\frac{(1-\bar{P}_s)\bar{k_2}}{\tau} &\approx \left(1-T_1-T_2(E_2(t)-\bar{k_2})\right)\frac{\bar{k_2}}{\tau} \\
&\approx \overbrace{\frac{\bar{k_2}}{\tau}\left(1-T_1+T_2\bar{k_2}\right)}^{\lambda_{a1}}\overbrace{-\frac{\bar{k_2}}{\tau}T_2}^{\lambda_{a2}}E_2(t) \qquad (5.4.8) \\
\text{Mean Attack Rate} &= \lambda_{a1}+\lambda_{a2}E_2(t) \qquad (5.4.9)
\end{aligned}$$

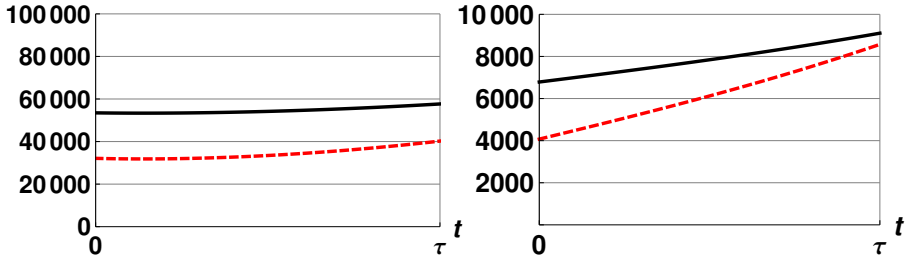
Considering the chosen values for the parameters and the values calculated for T_1 and T_2, the components of the mean rate of attack would be as follows: $\lambda_{a1} = 7375$, $\lambda_{a2} = -0.441$ (the initial rate of attack, i.e., $\lambda_{a1}+\lambda_{a2}\bar{k_2}$, would therefore be 4380 nodes/time unit). As n_p is constant throughout τ, a higher value of $E_2(t)$ means a lower value for the mean attack rate (i.e., the larger the botnet gets, the less effective the previously poisoned nodes become); this fact gets manifested through a negative value for λ_{a2}. In order to confirm that considering only the first 2 terms of the Taylor series of $\bar{P}_s$ is indeed enough, we draw $E_1(t)$ and $E_2(t)$, as depicted in Figure 5.4.1 (dashed lines), using their derived analytical expressions (i.e., with (5.3.6) and (5.3.7)) with the mean attack rate as defined in (5.4.8) by choosing the same sample values as before for the parameters of the index poisoning attack. As is clear from the figure, these two mean rates of attack (i.e., $\frac{(1-\bar{P}_s)\bar{k_2}}{\tau}$ and $\lambda_{a1}+\lambda_{a2}E_2(t)$) have near identical impact on $E_1(t)$ and $E_2(t)$ which shows that the Taylor series approximation (i.e., the first 2 terms of the Taylor series) is a good approximation. Therefore, we can conclude that the SIC-P2P model is a suitable model to study the impact of index poisoning against DHT-based P2P botnets. Finally, note that, as per (5.4.8), both λ_{a1} and λ_{a2} depend on the value of $\bar{k_2}$, among others, and λ_{a1} is not an independent, fixed constant.

5.5. Numerical Analysis

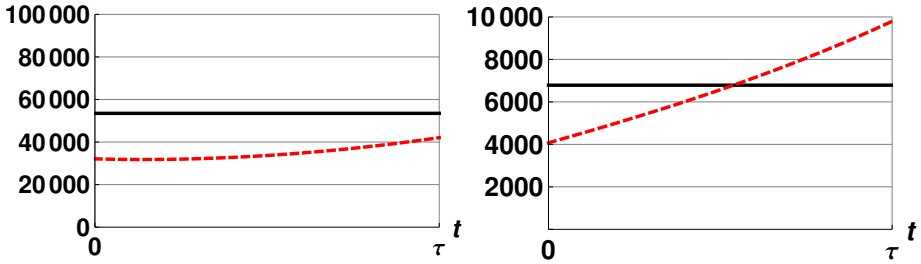
In this section, we provide some numerical analysis based on the analytical results of the SIC-P2P model derived in the previous sections. This numercial analysis sheds light on



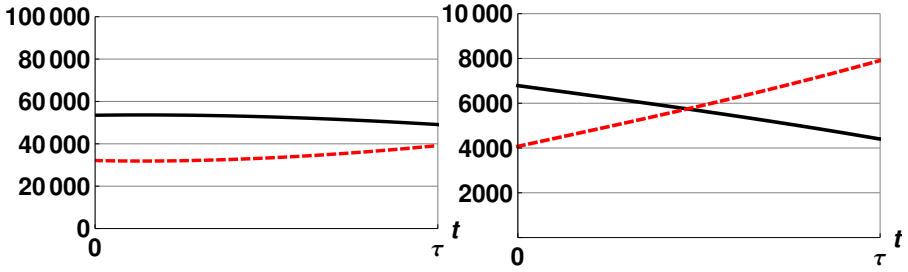
(a) $n_p = 0$ (b) $n_p = 0$



(c) $n_p = 100$ (d) $n_p = 100$



(e) $R_0 = 1$,$n_p = 200$ (f) $R_0 = 1$,$n_p = 200$



(g) $n_p = 400$ (h) $n_p = 400$

Figure 5.5.1.: SIC-P2P model: mean and standard deviation of number of nodes in *Infected* stage (left) and *Connected* stage (right) – with index poisoning.

possible uses of the SIC-P2P model in the real world, i.e., how the model helps security experts evaluate and refine mitigation strategies before deployment. The analysis of the index poisoning attack is depicted in Fig. 5.5.1. Few points on this analysis:

- In all scenarios, we set $\lambda_1 = 7$ and $\lambda_2 = 0.1$, as the focus is on the evaluation of mitigation strategies.
- The initial values at $t = 0$ for means and standard deviations are values at $t = 12$ when the botnet expands unhindered from $\bar{k_1} = 0$ and $\bar{k_2} = 1$.
- All λ parameters are nodes/time unit.
- τ is set to 1 time unit in our analyses.

The examined scenario is as follows: we first choose the following sample values: $n_p = 200$, $c = 4$ and $b = 5$ (values for c and b are some mid-range values [18]); the preceding chosen values determine λ_{a1} and λ_{a2} (i.e., we have: $\lambda_{a1} + \lambda_{a2} n_2 = 7618 - 0.451 n_2$). In order to also demonstrate the use of the Basic Reproduction Number (R_0), we consider the case where $R_0 = 1$. As all other parameters are already determined, λ_{r1} and λ_{r2} are therefore obtained from (5.3.11) (i.e., $\lambda_{r1} = 0.873$ and $\lambda_{r2} = 0.116$). These values of λ_{r1} and λ_{r2} are kept for all scenarios examined in Fig. 5.5.1. The center-bottom sub-figures of Fig. 5.5.1 depict the aforementioned scenario, i.e., when $R_0 = 1$. In the bottom sub-figures of Fig. 5.5.1, the only difference is that we have increased the number of poisoned nodes to 400, i.e., $n_p = 400$ (i.e., we have: $\lambda_{a1} + \lambda_{a2} n_2 = 8020 - 0.188 n_2$). This doubling of the number of the poisoned nodes shows a significant reduction of Infected and botnet nodes at $t = \tau$. In the center-top sub-figures of Fig. 5.5.1, on the other hand, the only difference is that we have decreased the number of poisoned nodes to 100, i.e., $n_p = 100$ (i.e., we have: $\lambda_{a1} + \lambda_{a2} n_2 = 4498 - 0.288 n_2$). This halving of the number of the poisoned nodes shows that the Infected and botnet nodes continue to increase. Finally, the top sub-figures of Fig. 5.5.1 show the scenario where we set $n_p = 0$ (which means $\lambda_{a1} + \lambda_{a2} n_2 = 0$). This scenario

shows the case where there is no attack at all on the botnet and the only mitigation is the disinfection (λ_{r1} and λ_{r2}).

5.6. Sybil Attack

This section concerns the sybil attack (random and targeted) and lays out a treatment similar to what was done for index poisoning in the previous two sections.

5.6.1. Taylor Series Approximation

This sub-section contains the steps equivalent to what was shown in Section 5.4 concerning the index poisoning attack; only formulas and descriptions that are different from the index poisoning case are provided. The obtained formula for the random sybil attack was mentioned in (2.3.3) which is repeated here:

$$P_s = \left(1 - \frac{n_s}{n_s + n}\right)^{\frac{\text{Log}_2[n_s+n]}{b}} \tag{5.6.1}$$

where n_s is the number of sybil nodes inserted randomly in the network (a configurable attack parameter). The rest of the parameters have similar definitions as mentioned for (5.4.1). Further, the Attack Rate (transition rate from *Connected* stage to *Infected* stage) is as defined in (5.4.2).

We proceed to obtain and test the Taylor series of P_s in (5.6.1). The Taylor series of P_s around k, the initial mean botnet size, is as follows (first 2 terms): $P_s \approx T_1 + (n-k)T_2$, where:

$$\begin{aligned} T_1 = &\quad \left(1 - \frac{n_s}{n_s + k}\right)^{\frac{\text{Log}[n_s+k]}{b\text{Log}[2]}} \\ T_2 = &\quad T_1 \frac{\left(k\text{Log}\left[\frac{k}{k+n_s}\right] + n_s\text{Log}\left[k+n_s\right]\right)}{bk\left(k+n_s\right)\text{Log}[2]} \end{aligned} \tag{5.6.2}$$

Likewise, the obtained formula for the targeted sybil attack was mentioned in (2.3.2) which is repeated here:

$$P_s = \left(1 - \frac{n_s}{n_s + \frac{n}{2^c}}\right)^{\frac{\text{Log}_2[n_s + \frac{n}{2^c}]}{b}} \tag{5.6.3}$$

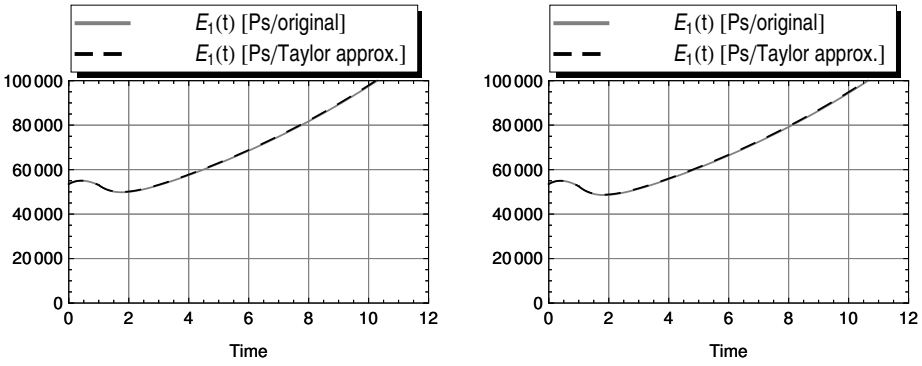
where n_s is the number of sybils inserted in the zone whose address space is close to the hash key used by the botmaster to distribute the botnet's C&C commands. The rest of the parameters have similar definitions as mentioned before. The Taylor series of P_s in (5.6.3) around k, the initial mean botnet size, is $P_s \approx T_1 + (n-k)T_2$, where:

$$\begin{aligned} T_1 = &\left(1 - \frac{n_s}{n_s + \frac{k}{2^c}}\right)^{\frac{\text{Log}\left[n_s + \frac{k}{2^c}\right]}{b\text{Log}[2]}} \\ T_2 = &\; T_1 \frac{\left(2^c n_s \text{Log}\left[2^{-c}k + n_s\right] + k\text{Log}\left[\frac{k}{k + 2^c n_s}\right]\right)}{bk\left(k + 2^c n_s\right)\text{Log}[2]} \end{aligned} \tag{5.6.4}$$

As done for the index poisoning case, we use the approximation of using $E_2(t)$ and $\bar{P}_s$ instead of n_2 and P_s, respectively. We therefore numerically solve the ODEs of means with the expressions for $\bar{P}_s$ and the defined mean attack rate (i.e., with (5.4.4)), as presented in (5.4.7). By choosing some sample values for the parameters of random and targeted sybil attacks, we proceed to numerically solve the ODEs of means in (5.4.7) and draw $E_1(t)$ and $E_2(t)$, as depicted in Fig. 5.6.1[2]. Again, note that in this scenario, τ is the first time unit of the analysis period. After $t = \tau$, there is no new attack on the botnet and the botnet continues to expand.

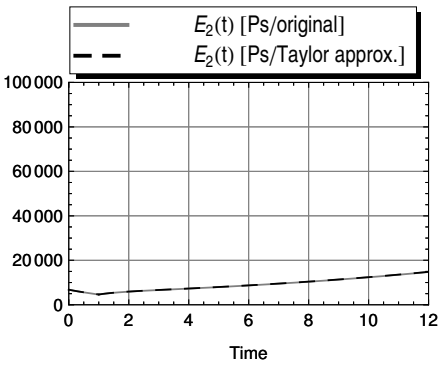
Considering the chosen values for the parameters, the components of the mean rate of attack (i.e., (5.4.8)) would be as follows: $\lambda_{a1} = 5195$ and $\lambda_{a2} = -0.273$ (initial rate of attack, i.e., $\lambda_{a1} + \lambda_{a2}\bar{k_2} = 3337$ nodes/time unit) in the case of random sybil attack and they would be as follows: $\lambda_{a1} = 5410$ and $\lambda_{a2} = -0.255$ (initial rate of attack, i.e., $\lambda_{a1} + \lambda_{a2}\bar{k_2} = 3674$ nodes/time unit) in the case of targeted sybil attack. In order to confirm that

[2]Parameter values (nodes/time unit): $\lambda_1 = 7$, $\lambda_2 = 0.1$, $\lambda_{r1} = 0.7$, and $\lambda_{r2} = 0.7$. Random sybil attack parameter values: $n_s = 2000$, and $b = 5$. Targeted sybil attack parameter values: $n_s = 220$, $b = 5$, and $c = 4$. τ is 1 time unit. Initial state values (number of nodes): $\bar{k_1} = 53480$ and $\bar{k_2} = 6786$.

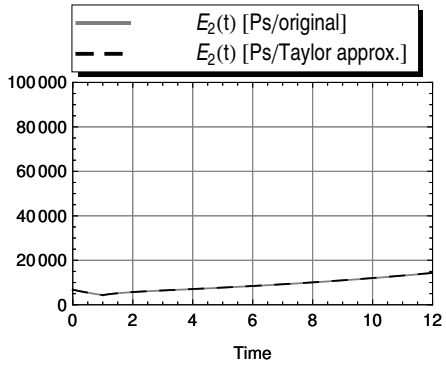


(a) Random sybil attack, $E_1(t)$

(b) Targeted sybil attack, $E_1(t)$



(c) Random sybil attack, $E_2(t)$



(d) Targeted sybil attack, $E_2(t)$

Figure 5.6.1.: SIC-P2P model: $E_1(t)$ and $E_2(t)$ with the mean rate of attack being $\frac{(1-\bar{P}_s)\bar{k}_2}{\tau}$ (i.e., original P_s) and $\lambda_{a1}+\lambda_{a2}E_2(t)$ (i.e., Taylor approximation of P_s).

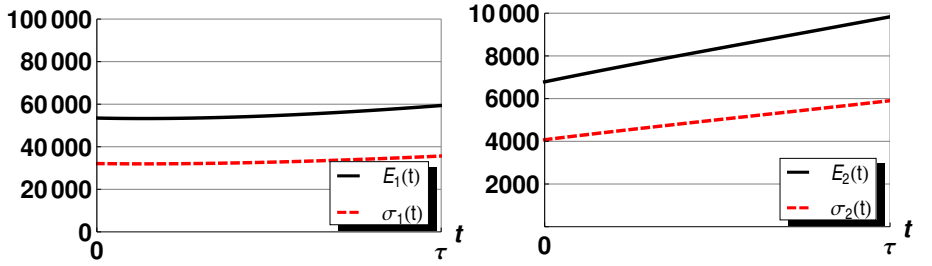
considering only the first 2 terms of the Taylor series of P_s is indeed enough, in Fig. 5.6.1, we draw $E_1(t)$ and $E_2(t)$, as derived analytically in (5.3.6) and (5.3.7), with the mean attack rate as defined in (5.4.8) and with the same sample values as before for the parameters of the random and targeted sybil attacks.

As is clear from Fig. 5.6.1, these two mean rates of attack (i.e., $\frac{(1-\bar{P}_s)\bar{k}_2}{\tau}$ and $\lambda_{a1}+\lambda_{a2}E_2(t)$) have near identical impact on $E_1(t)$ and $E_2(t)$. Therefore, we can conclude that the SIC-P2P model with $\lambda_{a1}+\lambda_{a2}n_2$ as the transition rate from *Connected* stage to *Infected* stage is also a suitable model to study the impact of both random and targeted sybil attacks, as mitigation strategies, against DHT-based P2P botnets.

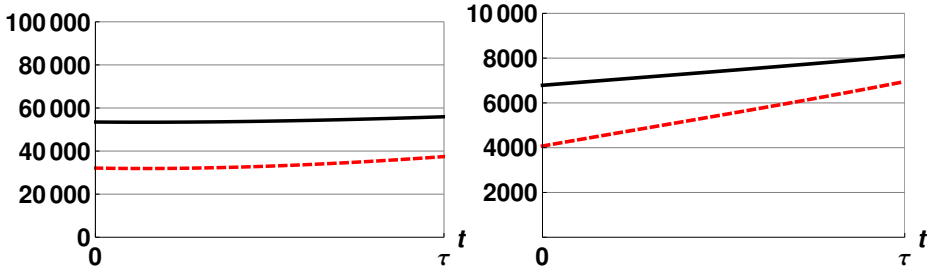
5.6.2. Numerical Analysis

The numerical analysis of a targeted sybil attack is not necessary, as the results and the values for different parameters are very similar to what was depicted in Fig. 5.5.1. This similarity in the results is due to the closeness in the structure of expressions of P_s in (5.6.3) and (5.4.1); in the interpretation of the results, n_p should be replaced by n_s which is the number of sybils inserted in the target zone. The numerical analysis of the random sybil attack, on the other hand, is depicted in Fig. 5.6.2. The examined scenario is similar to what was described for the index poisoning attack and is as follows: we first choose the following sample values: $n_s = 2000$ and $b = 5$; the preceding chosen values determine λ_{a1} and λ_{a2} (i.e., we have: $\lambda_{a1} + \lambda_{a2} n_2 = 5195 - 0.273 n_2$). In order to also demonstrate the use of the Basic Reproduction Number (R_0), we again consider the case where $R_0 = 1$. As all other parameters are already determined, λ_{r1} and λ_{r2} are therefore obtained from (5.3.11) (i.e., $\lambda_{r1} = 0.850$ and $\lambda_{r2} = 0.296$). These values of λ_{r1} and λ_{r2} are kept for all scenarios examined in Fig. 5.6.2. The center-bottom sub-figures of Fig. 5.6.2 depict the aforementioned scenario, i.e., when $R_0 = 1$.

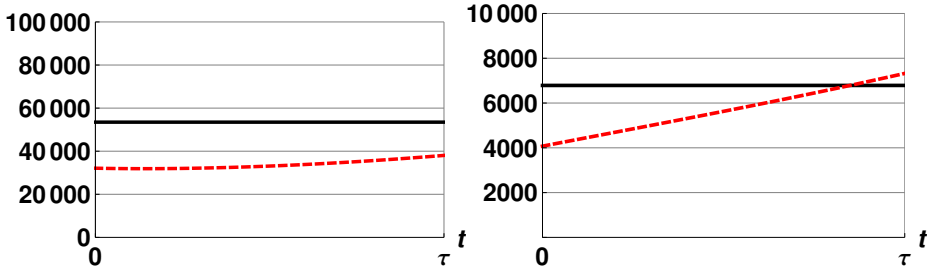
In the bottom sub-figures of Fig. 5.6.2, the only difference is that we have increased the number of inserted sybils to 10000, i.e., $n_s = 10000$ (i.e., we have: $\lambda_{a1} + \lambda_{a2} n_2 = 7088 - 0.123 n_2$). This increase in the number of sybils shows a significant reduction of Infected and botnet nodes at $t = \tau$. In the center-top sub-figures of Fig. 5.6.2, on the other hand, the only difference is that we have decreased the number of sybils to 1000, i.e., $n_s = 1000$ (i.e., we have: $\lambda_{a1} + \lambda_{a2} n_2 = 3444 - 0.208 n_2$). This halving of the number of sybils shows that the Infected and botnet nodes continue to increase. Finally, the top sub-figures of Fig. 5.6.2 show the scenario where we set $n_s = 0$ (which means $\lambda_{a1} + \lambda_{a2} n_2 = 0$). This scenario shows the case where there is no attack at all on the botnet and the only mitigation is the disinfection (λ_{r1} and λ_{r2}).



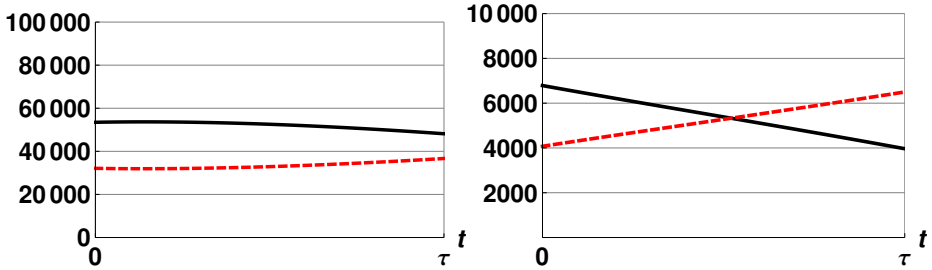
(a) $n_s = 0$ (b) $n_s = 0$



(c) $n_s = 1000$ (d) $n_s = 1000$



(e) $R_0 = 1$,$n_s = 2000$ (f) $R_0 = 1$,$n_s = 2000$



(g) $n_s = 10000$ (h) $n_s = 10000$

Figure 5.6.2.: SIC-P2P model: mean and standard deviation of number of nodes in *Infected* stage (left) and *Connected* stage (right) – with sybil nodes.

6. Botnets in 4G Cellular Networks

6.1. Introduction and Background

In this last chapter, we report and analyze a vulnerability of the air interface of 4G cellular networks, the Long Term Evolution (LTE), to DDoS attacks launched from botnets. The chapter is organized as follows: in this section, the iKee.B botnet is first introduced which was a botnet designed for cellular networks. Some hypothetical cellular botnet designs are explored afterwards which are few examples showing the growing interest of researchers in exploring and understanding potential cellular botnets. We then examine a study which quantified the threat of botnets in 2G/3G cellular networks by focusing on attacking a core network element. In Section 6.2, we explain the contribution of this chapter, i.e., identification and evaluation of botnet threat against the air interface in 4G networks. Finally, in Section 6.3, we describe the simulation scenario that we have used in order to quantify the threat against the 4G air interface; the section ends with the simulation results giving some indication regarding the needed botnet size in order to seriously impact the service availability.

6.1.1. Meet iKee.B Botnet

Botnets are already operating in cellular networks and the threat is not a hypothetical one. An example of a cellular botnet is iKee.B which was released in November 2009 and targeted iPhone users in several countries in Europe and Australia; the incident and the client

code have been examined in [73]. It is reported that iKee.B was the successor to iKee.A; the latter was released shortly beforehand in Australia and managed to infect an estimated 21,000 iPhone users within a week. iKee.B was spotted in Europe, however, and was the more capable version of the malware, possessing many of the important features of modern-day botnets: ability to self-propagate, carrying of malicious payload, and the functionality to connect to the C&C server to receive updates and commands from the botmaster.

6.1.2. Studies on Hypothetical Cellular Botnets

Recognizing the threat of botnets in cellular networks, many researchers have recently started to examine how botnets can be constructed in such networks: In [74], it has been shown how the Session Initiation Protocol (SIP), which is used in IMS for service delivery, can be abused to conceal botnet traffic which would prevent the detection of a botnet in 4G cellular networks. An example of in-the-lab cellular botnet is the implementation and evaluation of an iPhone-based mobile botnet [75]. In this work, authors report having tested a P2P-based and an SMS-based C&C mechanisms for the botnet and conclude that a hybrid approach of SMS- and HTTP-based C&C mechanism is the most threatening structure due to the difficulties in detection that it presents. Similar to the research that has been done for several years now for botnets operating in the wired Internet world, these studies provide ideas on how to develop mitigation strategies for botnets operating in the wireless/cellular networks.

6.1.3. Botnet-launched DDoS Attacks in 2G/3G Networks

Regarding the threat of botnets in 2G/3G cellular networks, the work done in [76] is perhaps the most detailed, with convincing and worrying results. In the study, a DDoS attack on the Home Location Register (HLR) has been tested. A DDoS attack that can successfully overload the HLR would make the network unusable for the clients. In order to carry out the attack, they have chosen to test various signaling requests sent by handsets; these

requests must be processed by the HLR. After several tests, they have determined that, in 2G/3G networks, "Insert/Delete Call Forwarding" is the most demanding request that can be sent by the handsets that needs to be processed by the HLR. The DDoS attack then constitutes of handsets simultaneously sending such requests to the HLR. To determine whether or not the generated traffic overloads the HLR, a several-computer testbed has been built using a traffic generating and benchmarking suite. The result of the tests is the number of botnet nodes needed by the botmaster in each HLR service area in order to overload the HLR. Depending on the traffic condition and the capacity of the HLR system, the numbers range from 11,750 to 141,000 botnet nodes (infected phones). With the assumption of one million users serviced by each HLR, the paper concludes, the needed infection rate would be between 1.2% and 14.1%.

6.2. DDoS Attacks Against the Air Interface of 4G Cellular Networks

The contribution of this chapter is the identification and evaluation of the threat of attacking the air interface which always has limited capacity due to the limited frequency spectrum available for such networks. 4G cellular networks, with their packet-switched mechanism and support of rich multimedia applications, are particularly vulnerable to such DDoS attacks that can be launched at will from a botnet operating in the cellular network. We therefore consider the air interface as the *main target* of the DDoS attack; the attack scenario would be as follows: when the botmasters create a botnet using smartphones, they can either activate all botnet nodes using a command or pre-program the nodes to wake up at a certain time. At the moment of attack, all botnet nodes can either start downloading a large file (a YouTube video, for example) to create congestion on the downlink or send dummy data to an arbitrary destination to create congestion on the uplink. The effect of the congestion is that most clients would no longer be able to effectively use the cellular

network which is the same effect predicted in [76] caused through overloading an HLR in 2G/3G networks. In the next section, we determine the needed botnet nodes per cell through a simulation study done using a highly capable open-source LTE simulator. The knowledge about existence of such a threat and how it might impact the availability of service, especially in emergency situations where the system already operates at capacity, is important and may lead to the designing of countermeasures by the operators.

6.3. Impact of Botnets: A Simulation Study

6.3.1. Simulation Scenario

In order to do performance evaluations of the LTE air interface, one of the best options is to use the LTE-Sim simulator [77]. As a feature-rich simulator, LTE-Sim has everything we need to assess the congestion caused by the DDoS attack on the air interface: it has an implementation of the physical layer, radio resource schedulers, applications (Voice over IP [VoIP], video, etc.), and a full protocol stack. In our scenario, VoIP uses a G.729 voice codec and the voice flow alternates between On and Off periods to model the natural silences in human conversation [77]. The duration of On periods is exponentially distributed with a mean value of 3 secs. On the other hand, the Off periods have a truncated exponential distribution with an average of 3 secs. and an upper limit of 6.9 secs. During the Off period, the sending rate is zero, as a voice activity detector is assumed to be present. During the On period, the source sends with the rate of 8 Kbps (20 bytes every 20 msec.). Finally, the video flow uses realistic video trace files of type H.264 Foreman sequence with a bit rate of 242 Kbps [77].

The whole evaluation of the botnet-launched DDoS attack takes place when the system operates at or near capacity, as we are concerned with times when the system is already under pressure to service many users due to an emergency; the assumption is that the cellular system has already been planned and deployed to be able to deal with a hypothetical

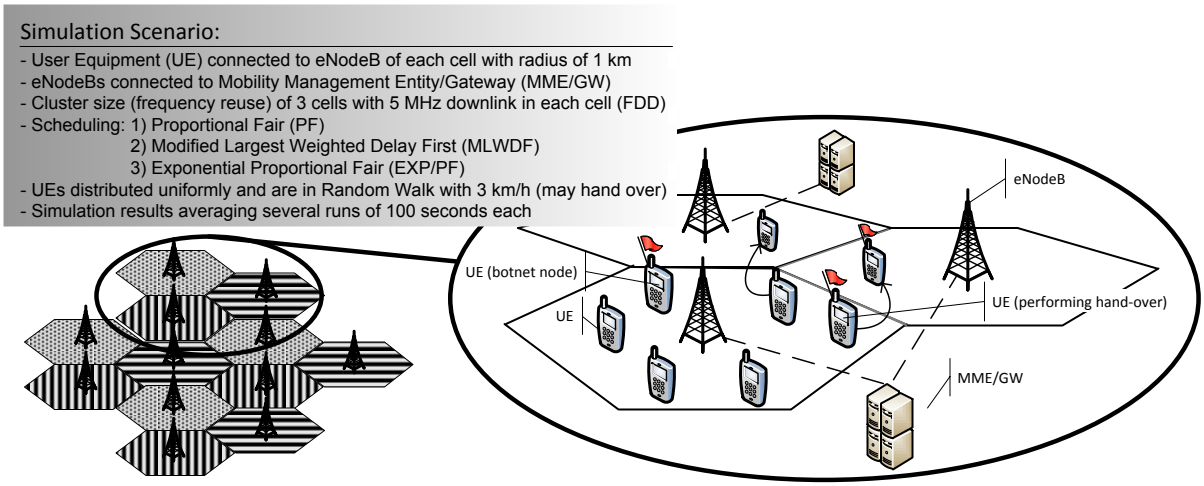


Figure 6.3.1.: Simulation Scenario: User Equipments (UEs) carrying VoIP sessions as well as botnet nodes starting dummy video sessions both moving around

emergency situation. When operating at capacity, the botnet attack is launched and we observe the effect of such an attack and determine the needed botnet nodes per cell in order to effectively deny service to users.

The simulated scenario is depicted in Fig. 6.3.1. We consider that there is a botnet built by a botmaster that can be activated to launch a DDoS attack against the air interface. The total botnet size is equal to the number of botnet nodes per cell times the number of cells. In each simulation run, there are a number of botnet nodes that are configured to download video simultaneously and there are other normal nodes (users) in the simulation that would be using VoIP in the meantime. We then examine the relationship between the number of botnet nodes and the level of degradation of service quality for the VoIP users.

In this work, due to simulator limitations, we only examine the case that all calls can pass through any call admission control module that might be present. It is noteworthy that even the presence of any such module does not diminish the threat posed by botnets, as the module cannot differentiate between any VoIP/video calls initiated by the botnet and the legitimate VoIP calls. The botnet therefore still manages to decrease the available capacity of service considerably.

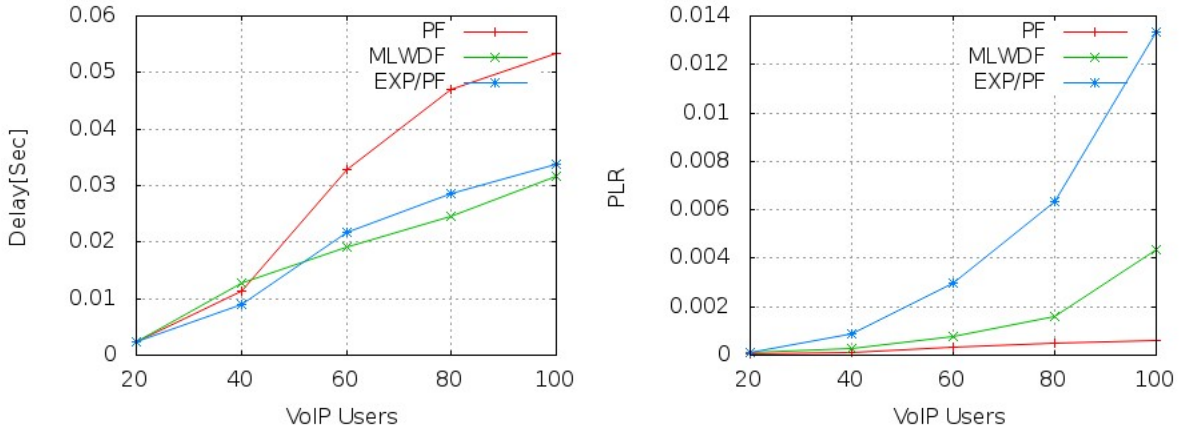
As we are examining an extreme case in terms of number of User Equipments (UEs)

and the fact that LTE-Sim is a particularly detailed simulator, each simulation run of 100 seconds takes 24 hours to complete in a powerful PC. In order to reduce the time needed for each run to 24 hours, however, we had to simulate a rather small network size of one cluster of 3 cells, with each cell having 5 MHz allocated downlink bandwidth. The cell radius is 1 km and users move around with a pedestrian speed of 3 km/h according to the Random Walk mobility model [78]; the UEs may hand over to the eNodeB of the neighboring cell due to changing power reception levels. As will be shown later on, with the aforementioned configuration for the cellular network, each cell/eNodeB has a capacity of servicing 100 VoIP users simultaneously; above this threshold, the quality of service starts to drop below industry standards. While examining the botnet-launched DDoS attack when the system operates at this maximum capacity, we also compare the performance of the main three downlink schedulers: (1) Proportional Fair (PF); (2) Modified Largest Weighted Delay First (MLWDF); and (3) Exponential Proportional Fair (EXP/PF). The details of these schedulers are beyond the scope of this thesis and it suffices to mention that MLWDF and EXP/PF are designed to deal with real-time flows, while PF treats every flow the same. One key difference would therefore be that MLWDF and EXP/PF erase packets belonging to real-time flows from the queue if those packets cannot be sent within a reasonable delay; this is done to avoid wasting bandwidth.

6.3.2. Determining the Cell Capacity

We first determine the cell capacity in the set up cellular network by increasing the number of VoIP users and monitoring each VoIP user's delay and packet loss ratio (PLR); see Fig. 6.3.2. In each sub-figure, results are reported for all the downlink schedulers. Considering the VoIP quality metrics, we see that the cell capacity is around 100 simultaneous VoIP users; these metrics will be elaborated on shortly.

We now need to determine the average number of subscribers that are present in a cell which has the capacity of serving 100 simultaneous VoIP users. For this, we turn to reports



(a) Delay

(b) PLR

Figure 6.3.2.: Determining cell capacity; increasing number of VoIP users until Delay and Packet Loss Ratio (PLR) reach maximum acceptable levels.

on user behavior regarding average monthly phone conversations. According to [79], a typical mobile phone user talks 761.5 minutes on average per month; the daily average would therefore be about 25 minutes. Assuming that most phone conversations happen in the period from 8 a.m. to 10 p.m. (i.e., a 14-hour slot in each 24 hours), we have: $25 \text{ mins.}/(14 \times 60 \text{ mins.}) \approx 3\%$. We can then state that each subscriber is actively using the system resources (i.e., it becomes one of the 100 simultaneous VoIP users) about 3% of the time each day[1]. Therefore, we can now consider that the maximum number of simultaneous VoIP users is about 3% of the average number of subscribers that are present in each cell. As our set up network had the capacity of serving 100 simultaneous VoIP users, then the average number of subscribers at this capacity would be about 3,300. This 1-to-33 relationship between the number of simultaneously-active users and the average number of subscribers has also been considered to be a reasonable estimate in real-world LTE deployments by [80].

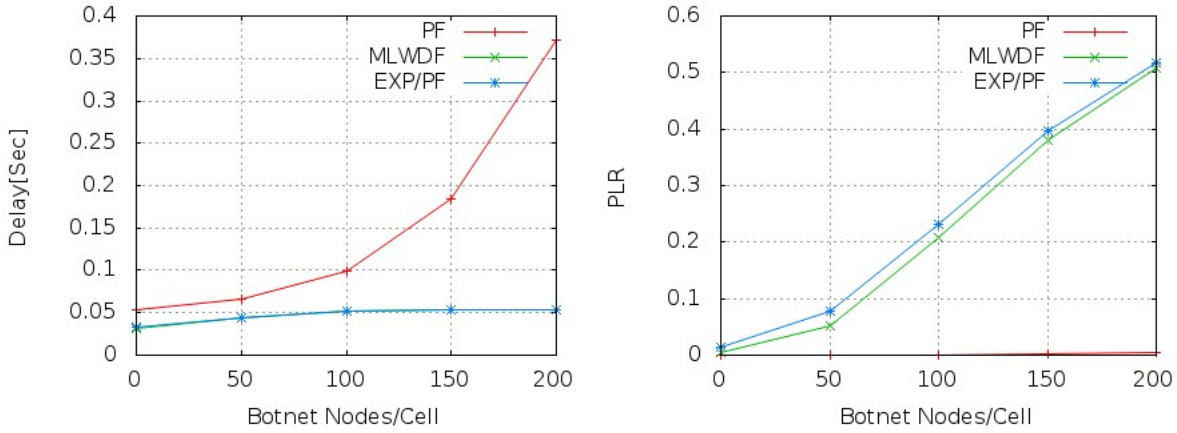
[1]Note that assuming a complete 24-hour slot, instead of the 14-hour slot, will lead to a percentage lower than 3%; hence, the needed percentage of botnet spread among subscribers of each cell to cause an outage, as will be determined later one, will be even lower.

6.3.3. VoIP Quality Metrics

Before proceeding further, we briefly introduce key VoIP quality metrics, as these will be needed in order to understand the observed degradation of service. Mean Opinion Score (MOS) is a subjective measurement of voice quality described in ITU recommendation P.800. MOS value for a voice communication ranges from 1 (impossible to communicate) to 5 (very satisfied). The two main VoIP quality metrics affecting MOS are packet loss and delay. Mouth-to-ear (one-way) delay is usually considered acceptable as long as it is below 150 msec. We, however, measure the packet delays only on the air interface which must be significantly less than 150 msec in order to satisfy the mouth-to-ear threshold. On the other hand, as we use the common G.729 voice codec in the simulator, we refer to results reported in [81] to point out that packet loss for this codec (with replacing the lost packet by the repetition method) leads to MOS going down from 4.3 (with 1% packet loss) to 2.8 (with 20% packet loss); the decrease is near linear and no results are reported for above 20% loss, as this would be completely unacceptable. Nonetheless, as MOS has degraded in a linear fashion from 1% packet loss to 20% packet loss, we can expect that at 50% packet loss, we will have a MOS value of 1 (impossible to communicate).

6.3.4. Determining the Effect of Botnet-launched DDoS Attack

At the maximum capacity, i.e., while 100 VoIP users are being served, we now start adding an increasing number of botnet nodes which download video while the VoIP users continue their sessions. The effect of botnet nodes on the VoIP quality of those 100 users is reported in Fig. 6.3.3. It can be seen that while the PF scheduler keeps the PLR near acceptable levels, the delay becomes increasingly large which hinders a proper phone conversation. Note that the shown delays are only the air interface delays which must be a small fraction of the acceptable mouth-to-ear delay of 150 msec. On the other hand, MLWDF and EXP/PF schedulers are designed for real-time flows and as such, drop many packets due to large delays to save bandwidth, as those delayed packets are no longer useful. These two



(a) Delay

(b) PLR

Figure 6.3.3.: Determining the effect of botnet-launched DDoS attack on 100 VoIP users that share resources with an increasing number of botnet nodes.

schedulers hence show an acceptable level of delay, however, the PLR reaches 20% with 100 botnet nodes and reaches 50% with 200 botnet nodes. Note that 100 botnet nodes and 200 botnet nodes represent a 3% infection rate and a 6% infection rate, respectively, among the subscribers in each cell.

We therefore observe that a botnet that has spread to only 3% of subscribers is capable of lowering the voice quality from 4.3 to 2.8 in Mean Opinion Score (MOS) scale of 1 to 5 for scheduling strategies designed for real-time flows. On the other hand, a botnet that has managed to spread to 6% of subscribers can cause a MOS value of 1, i.e., a complete outage. Contrasting these percentages with up to 14.1% botnet spread needed in 2G/3G networks, reported in Section 6.1.3, 4G/LTE networks seem to be much more vulnerable to botnet-launched DDoS attacks. The threat identified and the reported results could inspire the implementation of new mechanisms to ensure the security and availability of vital telecommunication services.

7. Conclusions and Future Work

There is a lack of appropriate analytical models on botnets in the literature. The prior work on botnets mostly consists of either deterministic analytical or simulation-based models. The deterministic models have the drawback of treating the botnet size as a deterministic variable, which neglects the stochastic nature of the evolution of botnets. These models only lead to determination of the mean botnet population size and not to the probability distribution of size or its higher moments. Further, the existing models determine the mean botnet size numerically and they have not obtained closed-form expressions. On the other hand, simulation-based models can be designed to capture the details of botnet lifecycle, but their results cannot be easily replicated or used by others.

7.1. Summary of Contributions and Conclusions

The work and the contribution of this thesis were presented in four chapters: Chapter 3 through Chapter 6. Here, we mention the main results, conclusions, and contributions.

In Chapter 3, two stochastic botnet models, SComI and SComF, were presented which cover both cases of infinite and finite node population sizes. As both models account for the most important node stages (i.e., *Susceptible* and *Compromised*), they are useful and sufficient models for the prediction and analysis of *initial unhindered botnet expansion*. For each of the two models, we derived the probability distribution of botnet size which allows a comprehensive analysis of botnet expansion phenomenon. These two models are

especially useful when not enough information is available on detailed operations of an emerging botnet; hence preventing the use of a three-node-stage model which needs more parameters to be known for it to be used. Using the developed models, the botnet size estimation problem has been reduced from having to estimate the global size of the botnet to the estimation of the model's parameter (λ) which requires only local knowledge.

In Chapter 4, we developed the SIC stochastic analytical model that captures the dynamics of a botnet's lifecycle. We have modeled the lifecycle of a node in the system with three stages referred to as *Susceptible*, *Infected*, and *Connected*. Considering three stages for the nodes would allow the proper incorporation of mitigation strategies, i.e., disinfection of nodes and attacks on botnet's C&C infrastructure, in the model. Therefore, the model would be applicable not only to *initial unhindered expansion*, but also to the later phases; hence, it can capture the full botnet lifecycle. We have modeled the system using a two-dimensional Markov process and derived a partial differential equation for the joint distribution of the number of nodes in each stage (*Infected* and *Connected*). Though this equation could not be solved, we were able to obtain closed-form expressions for the time dependent mean and variance/standard deviation of the population size in each stage. We also presented a successful attempt at explaining the size fluctuations of a real-world botnet using the SIC model. This success raised our confidence level with regard to the accuracy of the SIC model and its applicability in the real world.

A comparison between the values of mean and standard deviation in the figures regarding the SIC model suggests that the value of standard deviation is consistently about half of the mean value in all cases. This has implications with regard to how mean values should be interpreted in the real world. Inclusion of standard deviation in our analysis helps put the mean in its proper context; the higher the standard deviation gets, the less should be the importance of the precise value of the mean. For example, if a critical decision with regard to the deployment of costly mitigation strategies needs to be made when the predicted mean value of the botnet size crosses over an important threshold, it may be appropriate to delay

this decision until the predicated mean value of the botnet size crosses over the threshold plus the value of the standard deviation. This way, we would be more confident about the actual size of the botnet and the necessity of the deployment of the costly mitigation strategies.

In Chapter 5, we proposed and developed the SIC-P2P model which is an extension of the SIC model. Being the first analytical botnet model capable of doing so, the SIC-P2P model creates a direct analytical link between botnet size and different parameters of DHT-based P2P botnet mitigation strategies. This analytical link allows the planning and fine-tuning of mitigation strategies such as index poisoning and sybil attack with the granularity that was never possible before. Using the SIC-P2P model, we can examine the effect of increasing the number of sybils, in the case of sybil attack, or the number of poisoned nodes, in the case of index poisoning, on the botnet size. We are therefore able to have a more controlled approach in dealing with a botnet, rather than the existing approach which is just accelerating the mitigation strategies, at a potentially high cost, while being in the dark and hoping for the best.

For the SIC-P2P model, we derived closed-form expressions for the time-dependent means and variances of the number of nodes in *Infected* and *Connected* stages. Like the ones provided for the SIC model, the numerical results accompanying the analysis of the SIC-P2P model present potential use cases of the model and how P2P botnet mitigation strategies can be fine-tuned. As DHT-based P2P botnets have been among the most successful botnets from the botmasters' perspective, it is likely that their core operating principles are carried over to many future generations of botnets. The SIC-P2P model may then prove to be a valuable asset for a long time to come for the security professionals.

Having developed several analytical botnet models in previous chapters, in Chapter 6, we focused on how botnets could affect cellular networks. The trend in recent years has been the full integration of Internet services in the cellular networks, accommodating the huge demand for Internet access through smartphones. An obvious research item for us was

the examination of Internet security threats, and botnets in particular, that can now impact cellular wireless networks. We therefore examined how a 4G network using the LTE air interface could be impacted by botnets. We subsequently identified a potentially devastating threat against the LTE cellular networks, i.e., the launch of a DDoS attack against the air interface which is simple to implement and does not require inside knowledge about core network elements. Through the simulations, we determined that a botnet that has spread to only 6% of subscribers can effectively cause an outage in cellular services, particularly in peak hours and especially in emergency situations. We hope the exposed vulnerability and results shown would lead to the implementation of mechanisms to eliminate such a threat.

7.2. Publications

Matching the four chapters, Chapter 3 through Chapter 6, contributions of this thesis are presented in four papers, the first two of which have already been peer-reviewed and accepted for publication:

- "*SComF and SComI Botnet Models: The Cases of Initial Unhindered Botnet Expansion*", 25th Annual Canadian Conference on Electrical and Computer Engineering (CCECE12), Montreal, Canada, April 29-May 2, 2012
- "*The SIC Botnet Lifecycle Model: A Step Beyond Traditional Epidemiological Models*", Accepted paper to appear in Computer Networks (Elsevier), Special Issue on Botnet Activity: Analysis, Detection and Shutdown, DOI: 10.1016/j.comnet.2012.07.020
- "*SIC-P2P: A Lifecycle Model for the Evaluation of Mitigation Strategies Against P2P Botnets*", Submitted.
- "*Botnets in 4G Cellular Networks: Platforms to Launch DDoS Attacks Against the Air Interface*", Submitted.

7.3. Future Work

Considering the work done in this thesis, one could consider the following two aspects as suitable candidates for future work:

- Derivation of the probability distributions for the SIC and SIC-P2P models. The considerable efforts made to derive the probability distributions for these models were not successful due to certain cases of differential equations remaining unsolved as of now. The most likely path that could lead to a solution is documented in Section B.2, as mentioned earlier. By monitoring developments in this branch of mathematics, one could ultimately obtain closed-form solutions for the probability distributions at some point in the future. In the meantime, the derivation of means and variances done in this thesis can serve adequately the practical needs of the security research community.

- Extension of the work done with regard to the threat of botnet in 4G cellular networks in two directions:

    1. In light of the threat exposed, one could investigate the potential mitigation techniques in order to reduce and possibly eliminate the threat of the air interface falling victim to a DDoS attack. One possible mitigation technique would be the enhancement of the call admission control module so that the module is able to screen, detect, and prevent rogue calls;

    2. Following the work reported in [76] and described earlier, it is certainly relevant to determine how a botnet could attack a core network element in 4G systems. The equivalent of a Home Location Register (HLR) in 2G/3G networks is the Home Subscriber Server (HSS) of IMS in 4G networks. This research work could be carried out as soon as IMS implementations reach maturity and hardware specifications of HSS are known in order to test HSS's capacity and overload thresholds.

7.4. Concluding Remarks

Through an assessment of the recent simulation studies and testbed experiments, we noticed that both these methods are very time-consuming, need extensive resources, and involve some simplifications. Analytical models may achieve a sufficient level of realism, comparable to the levels achieved by simulation models and testbed experiments, significantly faster and less expensive; they are therefore accessible to a larger group of security researchers.

Earlier analytical botnet models, however, are mostly based on the simple path of reusing models initially developed in the context of epidemiology and malware propagation. Botnets, though, possess key differentiating characteristics. They merit a closer scrutiny and a ground-up approach to model development; this is the path taken in this thesis. Emergence of botnets, especially the resilient P2P botnets, as formidable threats on the Internet, and potentially within other Internet-enabled telecommunication infrastructure, motivated us to develop analytical models that properly capture the stochastic nature of population size changes and can help security professionals assess both the threat and the deployed mitigation strategies. With the development of several models each providing an invaluable, unique insight, this task has been accomplished.

Bibliography

[1] M. Ajelli, R. L. Cigno, and A. Montresor, "Modeling botnets and epidemic malware," in *Proc. IEEE Int'l Communications Conference (ICC)*, 2010, pp. 1–5.

[2] S. Mansfield-Devine, "Battle of the botnets," *Network Security*, vol. 2010, no. 5, pp. 4 – 6, 2010.

[3] D. Bleaken, "Botwars: the fight against criminal cyber networks," *Computer Fraud & Security*, vol. 2010, no. 5, pp. 17 – 19, 2010.

[4] C. J. Mielke and H. Chen, "Botnets, and the cybercriminal underground," in *Proc. IEEE Int. Conf. Intelligence and Security Informatics (ISI)*, 2008, pp. 206–211.

[5] N. Daswani and M. Stoppelman, "The anatomy of clickbot.a," in *Proc. First Workshop on Hot Topics in Understanding Botnets*. Berkeley, CA, USA: USENIX Association, 2007.

[6] D. Emm, "The kido botnet: Back to the future," in *Global Security, Safety, and Sustainability*, ser. Communications in Computer and Information Science, H. Jahankhani, A. G. Hessami, and F. Hsu, Eds. Springer Berlin Heidelberg, 2009, vol. 45, pp. 191–194.

[7] P. Porras, "Inside risks: Reflections on conficker," *Commun. ACM*, vol. 52, pp. 23–24, Oct. 2009.

[8] H. Binsalleeh, T. Ormerod, A. Boukhtouta, P. Sinha, A. Youssef, M. Debbabi, and L. Wang, "On the analysis of the zeus botnet crimeware toolkit," in *Proc. Eighth Annual Int Privacy Security and Trust (PST) Conf*, 2010, pp. 31–38.

[9] D. Bradbury, "Digging up the hacking underground," *Infosecurity*, vol. 7, no. 5, pp. 14 – 17, 2010.

[10] W. H. Murray, "The application of epidemiology to computer viruses," *Computers & Security*, vol. 7, no. 2, pp. 139 – 145, 1988.

[11] J. O. Kephart and S. R. White, "Directed-graph epidemiological models of computer viruses," in *Proc. IEEE Computer Society Symp Research in Security and Privacy*, 1991, pp. 343–359.

[12] G. Serazzi and S. Zanero, "Computer virus propagation models," in *Performance Tools and Applications to Networked Systems*, ser. Lecture Notes in Computer Science, M. C. Calzarossa and E. Gelenbe, Eds. Springer Berlin Heidelberg, 2004, vol. 2965, pp. 26–50.

[13] S. Fei, L. Zhaowen, and M. Yan, "A survey of internet worm propagation models," in *Proc. 2nd IEEE Int. Conf. Broadband Network & Multimedia Technology (IC-BNMT)*, 2009, pp. 453–457.

[14] "Voice and video calling over lte," Ericsson, White paper 284 23-3163 Uen, Feb. 2012. [Online]. Available: http://www.ericsson.com/res/docs/whitepapers/WP-Voice-Video-Calling-LTE.pdf

[15] A. Berger, I. Gojmerac, and O. Jung, "Internet security meets the ip multimedia subsystem: an overview," *Security Comm. Networks*, vol. 3, no. 2-3, pp. 185–206, 2010.

[16] C. Elliott, "Botnets: To what extent are they a threat to information security?" *Infor-*

mation Security Technical Report, vol. 15, no. 3, pp. 79 – 103, 2010 (computer crime - a 2011 update).

[17] M. Bailey, E. Cooke, F. Jahanian, Y. Xu, and M. Karir, "A survey of botnet technology and defenses," in *Proc. Cybersecurity Applications & Technology Conference for Homeland Security (CATCH)*. Washington, DC, USA: IEEE Computer Society, 2009, pp. 299–304.

[18] P. Wang, L. Wu, B. Aslam, and C. Zou, "A systematic study on peer-to-peer botnets," in *Proc. 18th Internatonal Conference on Computer Communications and Networks (ICCCN)*, Aug. 2009, pp. 1 –8.

[19] D. Dittrich and S. Dietrich, "P2p as botnet command and control: A deeper insight," in *Proc. 3rd Int. Conf. Malicious and Unwanted Software (MALWARE)*, 2008, pp. 41–48.

[20] J. Leonard, S. Xu, and R. Sandhu, "A framework for understanding botnets," in *Proc. Int. Conf. Availability, Reliability and Security (ARES '09)*, 2009, pp. 917–922.

[21] H. R. Zeidanloo and A. A. Manaf, "Botnet command and control mechanisms," in *Proc. Second Int. Conf. Computer and Electrical Engineering (ICCEE)*, vol. 1, 2009, pp. 564–568.

[22] G. Gu, J. Zhang, and W. Lee, "Botsniffer: Detecting botnet command and control channels in network traffic," in *Proc. 15th Annual Network and Distributed System Security Symposium (NDSS'08)*, San Diego, CA, USA, Feb. 10-13 2008.

[23] T. Holz, M. Steiner, F. Dahl, E. Biersack, and F. Freiling, "Measurements and mitigation of peer-to-peer-based botnets: a case study on storm worm," in *Proc. 1st Usenix Workshop on Large-Scale Exploits and Emergent Threats*. Berkeley, CA, USA: USENIX Association, 2008, pp. 9:1–9:9.

[24] G. Sinclair, C. Nunnery, and B. B.-H. Kang, "The waledac protocol: The how and why," in *Proc. 4th Int Malicious and Unwanted Software (MALWARE) Conf*, 2009, pp. 69–77.

[25] P. Wang, S. Sparks, and C. C. Zou, "An advanced hybrid peer-to-peer botnet," in *Proc. First Workshop on Hot Topics in Understanding Botnets*. Berkeley, CA, USA: USENIX Association, 2007.

[26] R. Vogt, J. Aycock, and M. Jacobson, "Army of botnets," in *Proc. 14th Network and Distributed System Security Symp. (NDSS)*, Feb. 2007.

[27] P. Wang, B. Aslam, and C. C. Zou, "Peer-to-peer botnets," in *Handbook of Information and Communication Security*, P. Stavroulakis and M. Stamp, Eds. Springer Berlin Heidelberg, 2010, pp. 335–350.

[28] J. R. Binkley and S. Singh, "An algorithm for anomaly-based botnet detection," in *Proc. 2nd conference on Steps to Reducing Unwanted Traffic on the Internet*, vol. 2. Berkeley, CA, USA: USENIX Association, 2006.

[29] G. Gu, P. Porras, V. Yegneswaran, M. Fong, and W. Lee, "Bothunter: detecting malware infection through ids-driven dialog correlation," in *Proc. 16th USENIX Security Symposium on USENIX Security Symposium*. Berkeley, CA, USA: USENIX Association, 2007, pp. 12:1–12:16.

[30] G. Gu, R. Perdisci, J. Zhang, and W. Lee, "Botminer: clustering analysis of network traffic for protocol- and structure-independent botnet detection," in *Proc. 17th conference on Security symposium*. Berkeley, CA, USA: USENIX Association, 2008, pp. 139–154.

[31] S. Gianvecchio, M. Xie, Z. Wu, and H. Wang, "Measurement and classification of humans and bots in internet chat," in *Proc. 17th conference on Security symposium*. Berkeley, CA, USA: USENIX Association, 2008, pp. 155–169.

[32] A. Brodsky and D. Brodsky, "A distributed content independent method for spam detection," in *Proc. First Workshop on Hot Topics in Understanding Botnets*. Berkeley, CA, USA: USENIX Association, 2007.

[33] D. Dittrich, F. Leder, and T. Werner, "A case study in ethical decision making regarding remote mitigation of botnets," in *Financial Cryptography and Data Security*, ser. Lecture Notes in Computer Science, R. Sion, R. Curtmola, S. Dietrich, A. Kiayias, J. Miret, K. Sako, and F. Sebe, Eds. Springer Berlin / Heidelberg, 2010, vol. 6054, pp. 216–230.

[34] P. Maymounkov and D. Mazières, "Kademlia: A peer-to-peer information system based on the xor metric," in *Revised Papers from the First International Workshop on Peer-to-Peer Systems*, ser. IPTPS '01. London, UK: Springer-Verlag, 2002, pp. 53–65.

[35] J. Liang, N. Naoumov, and K. W. Ross, "The index poisoning attack in p2p file sharing systems," in *Proc. 25th IEEE Int. Conf. Computer Communications (INFOCOM)*, 2006, pp. 1–12.

[36] J. R. Douceur, "The sybil attack," in *Revised Papers from the First International Workshop on Peer-to-Peer Systems*, ser. IPTPS '01. London, UK: Springer-Verlag, 2002, pp. 251–260.

[37] M. Castro, P. Druschel, A. Ganesh, A. Rowstron, and D. S. Wallach, "Secure routing for structured peer-to-peer overlay networks," *SIGOPS Oper. Syst. Rev.*, vol. 36, pp. 299–314, Dec. 2002.

[38] C. Davis, J. Fernandez, S. Neville, and J. McHugh, "Sybil attacks as a mitigation strategy against the storm botnet," in *Proc. 3rd International Conference on Malicious and Unwanted Software (MALWARE)*, Oct. 2008, pp. 32 –40.

[39] C. Davis, J. Fernandez, and S. Neville, "Optimising sybil attacks against p2p-based botnets," in *Proc. 4th International Conference on Malicious and Unwanted Software (MALWARE)*, Oct. 2009, pp. 78 –87.

[40] D. Ha, G. Yan, S. Eidenbenz, and H. Ngo, "On the effectiveness of structural detection and defense against p2p-based botnets," in *Proc. IEEE/IFIP International Conference on Dependable Systems Networks (DSN '09)*, Jul. 2009, pp. 297 –306.

[41] Z. Zhu, G. Lu, Y. Chen, Z. J. Fu, P. Roberts, and K. Han, "Botnet research survey," in *Proc. 32nd Annual IEEE Int. Computer Software and Applications (COMPSAC)*, 2008, pp. 967–972.

[42] The honeynet project. [Online]. Available: http://www.honeynet.org/

[43] B. B. Kang, E. Chan-Tin, C. P. Lee, J. Tyra, H. J. Kang, C. Nunnery, Z. Wadler, G. Sinclair, N. Hopper, D. Dagon, and Y. Kim, "Towards complete node enumeration in a peer-to-peer botnet," in *Proc. 4th International Symposium on Information, Computer, and Communications Security*, ser. ASIACCS '09. New York, NY, USA: ACM, 2009, pp. 23–34.

[44] M. A. Rajab, J. Zarfoss, F. Monrose, and A. Terzis, "My botnet is bigger than yours (maybe, better than yours): why size estimates remain challenging," in *Proc. First Workshop on Hot Topics in Understanding Botnets*. Berkeley, CA, USA: USENIX Association, 2007.

[45] C. Kanich, K. Levchenko, B. Enright, G. M. Voelker, and S. Savage, "The heisenbot uncertainty problem: challenges in separating bots from chaff," in *Proc. 1st Usenix Workshop on Large-Scale Exploits and Emergent Threats*. Berkeley, CA, USA: USENIX Association, 2008, pp. 10:1–10:9.

[46] Q. Wang, Z. Chen, C. Chen, and N. Pissinou, "On the robustness of the botnet topol-

ogy formed by worm infection," in *Proc. IEEE Global Telecommunications Conf. (GLOBECOM)*, 2010, pp. 1–6.

[47] P. Porras, H. Saidi, and V. Yegneswaran, "A multi-perspective analysis of the storm (peacomm) worm," Computer Science Laboratory, SRI International, CSL Technical Note, Oct. 2007. [Online]. Available: http://www.cyber-ta.org/pubs/StormWorm/

[48] E. V. Ruitenbeek and W. H. Sanders, "Modeling peer-to-peer botnets," in *Proc. Fifth International Conference on Quantitative Evaluation of Systems (QEST)*. Washington, DC, USA: IEEE Computer Society, 2008, pp. 307–316.

[49] A. Kolesnichenko, A. Remke, P.-T. de Boer, and B. Haverkort, "Comparison of the mean-field approach and simulation in a peer-to-peer botnet case study," in *Computer Performance Engineering*, ser. Lecture Notes in Computer Science, N. Thomas, Ed. Springer Berlin / Heidelberg, 2011, vol. 6977, pp. 133–147.

[50] A. White, A. Tickle, and A. Clark, "Overcoming reputation and proof-of-work systems in botnets," in *Proc. 4th Int Network and System Security (NSS) Conf*, 2010, pp. 120–127.

[51] J. Calvet, C. R. Davis, J. M. Fernandez, J.-Y. Marion, P.-L. St-Onge, W. Guizani, P.-M. Bureau, and A. Somayaji, "The case for in-the-lab botnet experimentation: creating and taking down a 3000-node botnet," in *Proc. 26th Annual Computer Security Applications Conference*, ser. ACSAC. New York, NY, USA: ACM, 2010, pp. 141–150.

[52] F. Brauer, P. van den Driessche, and J. Wu, Eds., *Mathematical Epidemiology*. Springer-Verlag Berlin Heidelberg, 2008.

[53] R. Weaver, "A probabilistic population study of the conficker-c botnet," in *Passive and Active Measurement*, ser. Lecture Notes in Computer Science. Springer Berlin / Heidelberg, 2010, vol. 6032, pp. 181–190.

[54] X. Li, H. Duan, W. Liu, and J. Wu, "The growing model of botnets," in *Proc. Int Green Circuits and Systems (ICGCS) Conf*, 2010, pp. 414–419.

[55] S. Li, X. Yun, Z. Hao, X. Cui, and Y. Wang, "A propagation model for social engineering botnets in social networks," in *Proc. 12th Int Parallel and Distributed Computing, Applications and Technologies (PDCAT) Conf*, 2011, pp. 423–426.

[56] Y. Wang, S. Wen, W. Zhou, W. Zhou, and Y. Xiang, "The probability model of peer-to-peer botnet propagation," in *Algorithms and Architectures for Parallel Processing*, ser. Lecture Notes in Computer Science, Y. Xiang, A. Cuzzocrea, M. Hobbs, and W. Zhou, Eds. Springer Berlin / Heidelberg, 2011, vol. 7016, pp. 470–480.

[57] C. C. Zou and R. Cunningham, "Honeypot-aware advanced botnet construction and maintenance," in *Proc. Int. Conf. Dependable Systems and Networks (DSN)*, 2006, pp. 199–208.

[58] D. Dagon, C. Zou, and W. Lee, "Modeling botnet propagation using time zones," in *Proc. 13th Network and Distributed System Security Symposium (NDSS)*, 2006.

[59] R. Li, L. Gan, and Y. Jia, "Propagation model for botnet based on conficker monitoring," in *Proc. Second Int Information Science and Engineering (ISISE) Symp*, 2009, pp. 185–190.

[60] W. Xin-liang, C. Lu-Ying, L. Fang, and L. Zhen-ming, "Analysis and modeling of the botnet propagation characteristics," in *Proc. 6th Int Wireless Comm. Netw. & Mobile Comp. (WiCOM) Conf*, 2010, pp. 1–4.

[61] H. Okamura, H. Kobayashi, and T. Dohi, "Markovian modeling and analysis of internet worm propagation," in *Proc. 16th IEEE Int. Symp. Software Reliability Engineering (ISSRE)*, 2005.

[62] D. Zwillinger, *Handbook of Differential Equations, 3rd Ed.* Academic Press, 1997.

[63] L. Kleinrock, *Queueing Systems - Volume I: Theory*. Wiley-Interscience, 1975.

[64] G. E. Riley, M. L. Sharif, and W. Lee, "Simulating internet worms," in *Proc. IEEE Computer Society's 12th Annual Int. Symp. Modeling, Analysis, and Simulation of Computer and Telecommunications Systems (MASCOTS)*, 2004, pp. 268–274.

[65] J. Rrushi, E. Mokhtari, and A. A. Ghorbani, "A statistical approach to botnet virulence estimation," in *Proc. 6th ACM Symp. on Info., Comp. & Comm. Sec.*, ser. ASIACCS, 2011, pp. 508–512.

[66] J. B. Grizzard, V. Sharma, C. Nunnery, B. B. Kang, and D. Dagon, "Peer-to-peer botnets: overview and case study," in *Proc. First Workshop on Hot Topics in Understanding Botnets (HotBots'07)*. Berkeley, CA, USA: USENIX Association, 2007.

[67] Q. Wang, Z. Chen, and C. Chen, "Characterizing internet worm infection structure," in *Proc. 4th USENIX conference on Large-scale exploits and emergent threats*, ser. LEET. Berkeley, CA, USA: USENIX Association, 2011.

[68] Z. Li, A. Goyal, Y. Chen, and V. Paxson, "Automating analysis of large-scale botnet probing events," in *Proc. 4th International Symposium on Information, Computer, and Communications Security*, ser. ASIACCS '09. New York, NY, USA: ACM, 2009, pp. 11–22.

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

[70] A. Papoulis and S. U. Pillai, *Probability, Random Variables and Stochastic Processes*, 4th ed. McGraw-Hill, 2002.

[71] "Top 10 botnet threat report - 2010," Damballa Inc., Tech. Rep., 2011. [Online]. Available: http://www.damballa.com/downloads/r_pubs/Damballa_2010_Top_10_Botnets_Report.pdf

[72] M. Khosroshahy, M. K. Mehmet-Ali, and D. Qiu. (2012, Mar.) Sic-p2p: A lifecycle model for the evaluation of mitigation strategies against p2p botnets (accompanying tech report: Mathematica derivations). [Online]. Available: http://www.masoodkh.com/files/papers/SIC/SIC-P2P-TechReport.pdf

[73] P. A. Porras, H. Saidi, and V. Yegneswaran, "An analysis of the ikee.b iphone botnet," *MobiSec, Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, Springer*, vol. 47, pp. 141–152, 2010.

[74] A. Berger and M. Hefeeda, "Exploiting sip for botnet communication," in *Proc. 5th IEEE Workshop Secure Network Protocols*, 2009, pp. 31–36.

[75] C. Mulliner and J.-P. Seifert, "Rise of the ibots: Owning a telco network," in *Proc. 5th International Conference on Malicious and Unwanted Software (MALWARE)*, Oct. 2010, pp. 71 –80.

[76] P. Traynor, M. Lin, M. Ongtang, V. Rao, T. Jaeger, P. McDaniel, and T. La Porta, "On cellular botnets: measuring the impact of malicious devices on a cellular network core," in *Proc. 16th ACM conference on Computer and communications security*, ser. CCS. NY, USA: ACM, 2009, pp. 223–234.

[77] G. Piro, L. A. Grieco, G. Boggia, F. Capozzi, and P. Camarda, "Simulating lte cellular systems: An open-source framework," *IEEE Transactions on Vehicular Technology*, vol. 60, no. 2, pp. 498–513, 2011.

[78] T. Camp, J. Boleng, and V. Davies, "A survey of mobility models for ad hoc network research," *Wireless Communications and Mobile Computing*, vol. 2, no. 5, pp. 483–502, 2002.

[79] "Mobile usage data : The voice results compiled from one year (april 2009-march 2010) of 60,000 mobile subscribers," The Nielsen Company, Online Report,

Aug. 2010. [Online]. Available: http://blog.nielsen.com/nielsenwire/online_mobile/african-americans-women-and-southerners-talk-and-text-the-most-in-the-u-s/

[80] N. Etminani (Core Network Lead at Nokia Siemens Networks), Private communication, Nov. 15 2012.

[81] L. Ding and R. Goubran, "Speech quality prediction in voip using the extended e-model," in *Proc. IEEE Global Telecommunications Conference (GLOBECOM)*, vol. 7, Dec. 2003, pp. 3974 – 3978.

[82] A. D. Polyanin and V. F. Zaitsev, *Handbook of exact solutions for ordinary differential equations*, 2nd ed. Chapman & Hall/CRC, 2003.

[83] P. L. Sachdev, *A compendium on nonlinear ordinary differential equations*. John Wiley & Sons, Inc., 1997.

[84] S. Kondratenya and E. Prolisko, "The existence and the form of solutions of lienard equations with a moving algebraic singularity," *Differential Equations*, vol. 9, pp. 198–201, 1973.

[85] M. Khosroshahy, M. K. Mehmet-Ali, and D. Qiu. (2012, Feb.) The sic botnet lifecycle model: A step beyond traditional epidemiological models (accompanying tech report: Mathematica derivations). [Online]. Available: http://www.masoodkh.com/files/papers/SIC/SIC-TechReport.pdf

A. The SCom Models

A.1. The SComI Model Derivations

Here, we provide the details of solving (3.2.2) on page 36. We define the auxiliary variable s which represents the scaled distance along a characteristic curve. Based on the Method of Characteristics [62], z, t and therefore $P(z,t)$ are effectively all functions of s. We therefore can write the following equations based on (3.2.2):

$$\begin{cases} \frac{\partial t}{\partial s} = 1 \\ \frac{\partial z}{\partial s} = \lambda z(1-z) \\ \frac{dP(z,t)}{ds} = 0 \end{cases} \tag{A.1.1}$$

With the initial condition $P(z,0) = z$, we therefore have:

$$\begin{cases} t(s=0) = 0 \\ z(s=0) = i_1 \\ P(s=0) = i_1 \end{cases} \tag{A.1.2}$$

From the previous equations, we have $t(s,i_1) = s$ and $P(s,i_1) = i_1$ (i_1 is an unknown parameter.). To solve $\frac{\partial z}{\partial s} = \lambda z(1-z)$, we proceed as follows:

$$\begin{aligned} \frac{\partial z}{\partial s} &= \lambda z(1-z) \\ z' &= \lambda z(1-z) \\ z' - \lambda z &= -\lambda z^2 \qquad \text{(A.1.3)} \\ \frac{z'}{z^2} - \frac{\lambda}{z} &= -\lambda \qquad \text{(A.1.4)} \end{aligned}$$

Eq. (A.1.3) is a Bernoulli differential equation [62] which can be solved through taking some steps including the use of Integrating Factors method (the Integrating Factor being $e^{\lambda s}$) as follows: we define $u(s) \triangleq \frac{1}{z(s)}$; therefore $u'(s) = \frac{-z'(s)}{z^2(s)}$. Eq. (A.1.4) can therefore be written as: $-u'(s) - \lambda u(s) = -\lambda$ or:

$$u'(s) + \overbrace{\lambda}^{G(s)} u(s) = \overbrace{\lambda}^{Q(s)} \qquad \text{(A.1.5)}$$

Solution of (A.1.5) can be found through the method of Integrating Factors as follows: Integrating Factor $= W(s) = \exp(\int^s G(m)dm) = \exp(\lambda s) = e^{\lambda s}$. We multiply (A.1.5) by the calculated Integrating Factor and derive the solution as follows:

$$\begin{aligned} e^{\lambda s}u'(s) + e^{\lambda s}\lambda u(s) &= e^{\lambda s}\lambda \\ \frac{d\left(e^{\lambda s}u(s)\right)}{ds} &= e^{\lambda s}\lambda \\ e^{\lambda s}u(s) &= e^{\lambda s} + C \\ u(s) &= 1 + Ce^{-\lambda s} \qquad \text{(A.1.6)} \end{aligned}$$

As $u(s) \triangleq \frac{1}{z(s)}$, from (A.1.6), we finally derive $z(s)$ as follows:

$$z(s) = \frac{1}{1 + Ce^{-\lambda s}} \qquad \text{(A.1.7)}$$

Using the initial condition in (A.1.2), we derive the constant C in (A.1.7): $C = \frac{1}{i_1} - 1$. To summarize, we have:

$$\begin{cases} t = s \\ z = \frac{1}{1+(\frac{1}{i_1}-1)e^{-\lambda s}} \\ P = i_1 \end{cases} \tag{A.1.8}$$

From (A.1.8), we can eliminate i_1 and s and derive P in terms of z and t as shown in (3.2.3) on page 37.

A.2. The SComF Model Derivations

A.2.1. Laplace Transform of the Probability Distribution

Taking (3.3.1) on page 40 to Laplace domain, we have:

$$\begin{aligned} sP_1^*(s) - P_1(0) &= -\lambda P_1^*(s) \\ P_1^*(s) &= \frac{1}{s+\lambda} \qquad n = 1 \\ sP_n^*(s) - P_n(0) &= (n-1)\lambda P_{n-1}^*(s) - n\lambda P_n^*(s) \qquad 2 \le n \le \frac{N}{2} \end{aligned}$$

Since $P_n(0) = 0$ for $n > 1$, we can derive:

$$P_n^*(s) = \frac{(n-1)\lambda}{s+n\lambda} P_{n-1}^*(s) \qquad 2 \le n \le \frac{N}{2} \tag{A.2.1}$$

Again, since $P_n(0) = 0$ for $n > 1$, we proceed as follows for $\frac{N}{2} + 1 \le n \le N$:

$$sP_n^*(s) - P_n(0) = (N-n+1)\lambda P_{n-1}^*(s) - (N-n)\lambda P_n^*(s)$$

$$P_n^*(s) = \frac{(N-n+1)\lambda}{s+(N-n)\lambda} P_{n-1}^*(s) \qquad \frac{N}{2} + 1 \le n \le N \tag{A.2.2}$$

To summarize, we have:

$$P_n^*(s) = \begin{cases} \frac{1}{s+\lambda} & n = 1 \\ \frac{(n-1)\lambda}{s+n\lambda} P_{n-1}^*(s) & 2 \le n \le \frac{N}{2} \\ \frac{(N-n+1)\lambda}{s+(N-n)\lambda} P_{n-1}^*(s) & \frac{N}{2}+1 \le n \le N \end{cases} \tag{A.2.3}$$

Using the induction method, from (A.2.3), we can recursively determine $P_n^*(s)$ as follows: for $2 \le n \le 5$ (assuming that $N = 10$), we can write:

$$\begin{aligned} P_2^*(s) &= \frac{\lambda}{s+2\lambda} P_1^*(s) = \frac{\lambda}{(s+\lambda)(s+2\lambda)} \\ P_3^*(s) &= \frac{2\lambda}{s+3\lambda} P_2^*(s) = \frac{2\lambda^2}{(s+\lambda)(s+2\lambda)(s+3\lambda)} \\ P_4^*(s) &= \frac{3\lambda}{s+4\lambda} P_3^*(s) = \frac{6\lambda^3}{(s+\lambda)(s+2\lambda)(s+3\lambda)(s+4\lambda)} \\ P_5^*(s) &= \frac{4\lambda}{s+5\lambda} P_4^*(s) = \frac{24\lambda^4}{(s+\lambda)(s+2\lambda)(s+3\lambda)(s+4\lambda)(s+5\lambda)} \end{aligned}$$

For $\frac{N}{2}+1 \le n \le N$, we proceed as follows:

$$\begin{aligned} P_{\frac{N}{2}+1}^*(s) &= \frac{\frac{N}{2}\lambda}{s+(\frac{N}{2}-1)\lambda} P_{\frac{N}{2}}^*(s) \\ P_{\frac{N}{2}+2}^*(s) &= \frac{(\frac{N}{2}-1)\lambda}{s+(\frac{N}{2}-2)\lambda} P_{\frac{N}{2}+1}^*(s) = \frac{\frac{N}{2}(\frac{N}{2}-1)\lambda^2}{\left(s+(\frac{N}{2}-1)\lambda\right)\left(s+(\frac{N}{2}-2)\lambda\right)} P_{\frac{N}{2}}^*(s) \\ P_{\frac{N}{2}+3}^*(s) &= \frac{(\frac{N}{2}-2)\lambda}{s+(\frac{N}{2}-3)\lambda} P_{\frac{N}{2}+2}^*(s) = \frac{\frac{N}{2}(\frac{N}{2}-1)(\frac{N}{2}-2)\lambda^3}{\left(s+(\frac{N}{2}-1)\lambda\right)\left(s+(\frac{N}{2}-2)\lambda\right)\left(s+(\frac{N}{2}-3)\lambda\right)} P_{\frac{N}{2}}^*(s) \end{aligned}$$

From the structure of the preceding expressions, $P_n^*(s)$ has the general form shown in (3.3.2) on page 40.

A.2.2. Probability Distribution

We apply the partial fraction expansion method to the expressions obtained for $P_n^*(s)$ in order to perform the Laplace inversion to derive $P_n(t)$. From (3.3.2) on page 40 and the previously derived expressions for $P_n^*(s)$, we can write (assuming that $N = 10$):

$$
\begin{aligned}
P_2^*(s) &= \tfrac{1}{s+\lambda} + \tfrac{-1}{s+2\lambda} \\
P_3^*(s) &= \tfrac{1}{s+\lambda} + \tfrac{-2}{s+2\lambda} + \tfrac{1}{s+3\lambda} \\
P_4^*(s) &= \tfrac{1}{s+\lambda} + \tfrac{-3}{s+2\lambda} + \tfrac{3}{s+3\lambda} + \tfrac{-1}{s+4\lambda} \\
P_5^*(s) &= \tfrac{1}{s+\lambda} + \tfrac{-4}{s+2\lambda} + \tfrac{6}{s+3\lambda} + \tfrac{-4}{s+4\lambda} + \tfrac{1}{s+5\lambda} \\
P_6^*(s) &= \tfrac{\frac{5}{3}}{s+\lambda} + \tfrac{-10}{s+2\lambda} + \tfrac{30}{s+3\lambda} + \tfrac{-20\lambda}{(s+4\lambda)^2} + \tfrac{-\frac{50}{3}}{s+4\lambda} + \tfrac{-5}{s+5\lambda} \\
P_7^*(s) &= \tfrac{\frac{10}{3}}{s+\lambda} + \tfrac{-40}{s+2\lambda} + \tfrac{120\lambda}{(s+3\lambda)^2} + \tfrac{-120}{s+3\lambda} + \tfrac{80\lambda}{(s+4\lambda)^2} + \tfrac{\frac{440}{3}}{s+4\lambda} + \tfrac{10}{s+5\lambda} \\
P_8^*(s) &= \tfrac{10}{s+\lambda} + \tfrac{-120\lambda}{(s+2\lambda)^2} + \tfrac{280}{s+2\lambda} + \tfrac{-360\lambda}{(s+3\lambda)^2} + \tfrac{0}{s+3\lambda} + \tfrac{-120\lambda}{(s+4\lambda)^2} + \tfrac{-280}{s+4\lambda} + \tfrac{-10}{s+5\lambda} \\
P_9^*(s) &= \tfrac{20\lambda}{(s+\lambda)^2} + \tfrac{\frac{-235}{3}}{s+\lambda} + \tfrac{240\lambda}{(s+2\lambda)^2} + \tfrac{-320}{s+2\lambda} + \tfrac{360\lambda}{(s+3\lambda)^2} + \tfrac{180}{s+3\lambda} + \tfrac{80\lambda}{(s+4\lambda)^2} + \tfrac{\frac{640}{3}}{s+4\lambda} + \tfrac{5}{s+5\lambda} \\
P_{10}^*(s) &= \tfrac{1}{s} + \tfrac{-20\lambda}{(s+\lambda)^2} + \tfrac{\frac{175}{3}}{s+\lambda} + \tfrac{-120\lambda}{(s+2\lambda)^2} + \tfrac{100}{s+2\lambda} + \tfrac{-120\lambda}{(s+3\lambda)^2} + \tfrac{-100}{s+3\lambda} + \tfrac{-20\lambda}{(s+4\lambda)^2} + \tfrac{-\frac{175}{3}}{s+4\lambda} + \tfrac{-1}{s+5\lambda}
\end{aligned}
$$

An an example for the more complicated case of $\frac{N}{2}+1 \leq n \leq N$, the expression for $P_6^*(s)$ has been derived as follows:

$$
\begin{aligned}
P_6^*(s) &= \frac{5\lambda}{s+4\lambda} P_5^*(s) \\
&= \frac{5\lambda \times 4! \times \lambda^4}{(s+5\lambda)(s+4\lambda)^2(s+3\lambda)(s+2\lambda)(s+\lambda)} \\
&= \frac{A}{s+\lambda} + \frac{B}{s+2\lambda} + \frac{C}{s+3\lambda} + \frac{D}{(s+4\lambda)^2} + \frac{E}{s+4\lambda} + \frac{F}{s+5\lambda}
\end{aligned}
$$

$$
\begin{aligned}
A &= \frac{5\lambda \times 4! \times \lambda^4}{(4\lambda)(3\lambda)^2(2\lambda)(\lambda)} = \frac{5}{3} \\
B &= \frac{5\lambda \times 4! \times \lambda^4}{(3\lambda)(2\lambda)^2(\lambda)(-\lambda)} = -10 \\
C &= \frac{5\lambda \times 4! \times \lambda^4}{(2\lambda)(\lambda)^2(-\lambda)(-2\lambda)} = 30 \\
D &= \frac{5\lambda \times 4! \times \lambda^4}{(\lambda)(-\lambda)(-2\lambda)(-3\lambda)} = -20\lambda \\
E &= \frac{d}{ds}\left(\frac{5\lambda \times 4! \times \lambda^4}{(s+5\lambda)(s+3\lambda)(s+2\lambda)(s+\lambda)}\right)\Bigg|_{s=-4\lambda} = -\frac{50}{3} \\
F &= \frac{5\lambda \times 4! \times \lambda^4}{(-\lambda)^2(-2\lambda)(-3\lambda)(-4\lambda)} = -5
\end{aligned}
$$

From the preceding expressions, $P_n^*(s)$ has the following general form:

$$
P_n^*(s) = \begin{cases}
\frac{1}{s+\lambda} & n = 1 \\
\sum_{k=0}^{n-1}\left((-1)^k\binom{n-1}{k}\frac{1}{s+(k+1)\lambda}\right) & 2 \le n \le \frac{N}{2} \\
\sum\limits_{\substack{k=1 \\ k\notin[N-n,\frac{N}{2}-1]}}^{\frac{N}{2}}\left(\frac{T_1}{T_2}\frac{1}{s+k\lambda}\right) + \sum\limits_{k\in[N-n,\frac{N}{2}-1]}^{\frac{N}{2}-1}\left(\frac{T_1}{T_2}\frac{1}{(s+k\lambda)^2} + \frac{d(\frac{T_1}{T_3})}{ds}\Big|_{s=-k\lambda}\frac{1}{s+k\lambda}\right) & \frac{N}{2}+1 \le n < N \\
\frac{T_1}{T_2|_{k=0}}\frac{1}{s} + \sum_{k=1}^{\frac{N}{2}-1}\left(\frac{T_1}{T_2}\frac{1}{(s+k\lambda)^2} + \frac{d(\frac{T_1}{T_3})}{ds}\Big|_{s=-k\lambda}\frac{1}{s+k\lambda}\right) + \frac{T_1}{T_2|_{k=\frac{N}{2}}}\frac{1}{s+\frac{N}{2}\lambda} & n = N
\end{cases}
\tag{A.2.4}
$$

where T_1, T_2 and T_3 are given as follows:

$$
\begin{aligned}
T_1 &= \frac{\frac{N}{2}!}{(N-n)!}\lambda^{(n-\frac{N}{2})}(\frac{N}{2}-1)!\lambda^{\frac{N}{2}-1} \\
T_2 &= \prod_{\substack{i=1 \\ i\neq k}}^{\frac{N}{2}}(i-k)\lambda \prod_{\substack{j=N-n \\ j\neq k}}^{\frac{N}{2}-1}(j-k)\lambda \\
T_3 &= \prod_{\substack{i=1 \\ i\neq k}}^{\frac{N}{2}}(s+i\lambda) \prod_{\substack{j=N-n \\ j\neq k}}^{\frac{N}{2}-1}(s+j\lambda)
\end{aligned}
$$

The Laplace inversion of (A.2.4) gives the probability distribution as shown in (3.3.3) on page 41.

A.3. The SComF Model: Simulation Using the GTNetS simulator

A.3.1. Simulated Topology

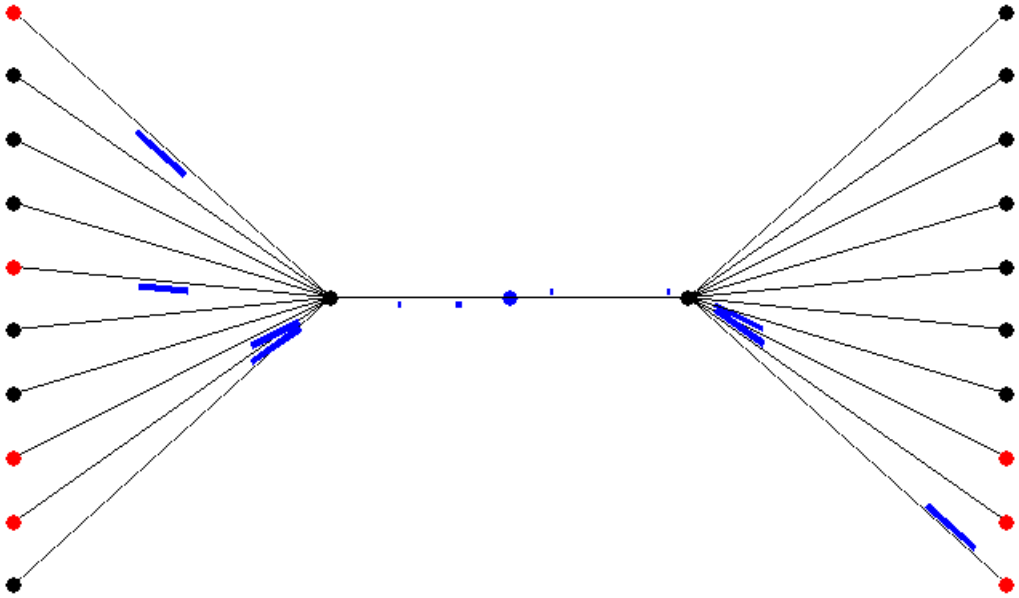


Figure A.3.1.: Worm spread in the 20-host simulated topology in GTNetS

A.3.2. Simulation Scenario

```
1  // wormsim.cc used for the SComF paper by Masood Khosroshahy
2  // Changes compared to the original wormsim.cc are noted by the term "SComF".
3  // Changes made only correct few problems; they do not affect
4  // the simulation scenario (except the number of hosts in the network).
5
6  // Worm Demo for GTNetS
7  // Supports command level arguments and log files.
8  // Monirul I Sharif, Georgia Institute of Technology
9
10 #include <iostream>
11 #include <stdio.h>
12 #include <string>
13 #ifndef WIN32
14 #include <sys/time.h>
15 #else
16 #include <time.h>
17 #include <winsock.h>
18 #endif
19 #include "validation.h"
20
21 //#define DEBUG_MASK 0x01
22
23 #include "debug.h"
24 #include "common-defs.h"
25 #include "simulator.h"
26 #include "node.h"
27 #include "dumbbell.h"
28 #include "tcp-tahoe.h"
29 #include "randomtree.h"
30 #include "tree.h"
31 #include "ratetimeparse.h"
32 #include "linkp2p.h"
33 #include "routing.h"
34 #include "routing-static.h"
35 #include "args.h"
36 #include "bfs.h"
37
38 #include "wormtcp.h"
39 #include "wormudp.h"
40 #include "wormtargetvector.h"
41 #include "wormhelper.h"
42
43 #ifdef HAVE_QT
44 #include <qnamespace.h>
45 #endif
46
47
48 // ------------ Define worm types    ---------------
49 #define TCPWORMTYPE  1
50 #define UDPWORMTYPE  2
51 #define WORMTYPE     UDPWORMTYPE
52
53 // ------------ Define the topology  ---------------
54 //SComF commented out:
55 //#define TREES        4
56 #define TREES        2
57 //SComF commented out:
58 //#define DEPTH        4
59 #define DEPTH        2
60 //SComF commented out:
61 //#define FANOUT       8
62 #define FANOUT       10
63 //SComF commented out:
64 //#define TREELEGPROB  0.85
65 #define TREELEGPROB  1
66
67 // SComF:
68 // The above changes to the topology is such that we will ALWAYS
```

```
69 // have 20 real hosts in the network (if not altered at the command line).
70 // This is to make sure we can set N=20 for SComF.
71
72 #define LINKBW       "10Mb"
73 #define HLINKBW      "10Mb"
74 #define BLINKBW      "100Mb"
75
76 // change the following if you want to have
77 #define DEPTH_FS   0
78 #define FANOUT_FS  0
79
80 // ------------ Define Addressing   ----------------
81 #define BASEIP    "10.0.0.0"
82
83 // ------------ Worm parameters -----------------------
84 #define VULNERABILITY  1.0
85 #define THREADS        3
86 #define SCANRATE       100
87 #define SCANRANGE      0
88 #define PAYLOAD        1000
89
90 // ------------ File names ----------------------------
91 #define DEFFILENAME "wormsimdata"
92
93
94 // ----------- Simulation settings -------------------
95 #define SIMTIME        10.0
96 #define SIMINTERVAL    0.05
97 #define SIMSHOW        0
98 #define SIMRANDOM      1
99
100 using namespace std;
101
102 // SComF: commented out due to compilation problem
103 // (not important: used only for the calculation of simulation run time)
104 //#ifdef WIN32
105 //void          gettimeofday(struct timeval *tv,int dummy)
106 //{
107 //SYSTEMTIME SystemTime;
108 //GetSystemTime(&SystemTime);
109 //tv->tv_sec = SystemTime.wSecond ;
110 //tv->tv_usec = SystemTime.wMilliseconds ;
111 //}
112 //#endif
113
114
115 FILE *tracefile, *infofile;
116 Count_t wormtype;
117
118
119 // SComF: commented out due to compilation problem
120 // (not important: used only for the calculation of simulation run time)
121
122 // returns time since January 1, 1970 in useconds
123 //unsigned long long int get_time()
124 //{
125 //  struct timeval tp;
126 //  unsigned long long int result;
127 //  gettimeofday(&tp, NULL);
128 //
129 //  result = tp.tv_sec;
130 //  result = (result * 1000000L) + tp.tv_usec;
131
132 //  return result;
133 //}
134
135 double in_seconds(unsigned long long int time)
136 {
```

```
137   return ((double)time)/1000000.0;
138 }
139
140
141 // Progress hook
142 static void Progress(Time_t now)
143 {
144   int infected;
145   if (wormtype == UDPWORMTYPE)
146     infected = WormUDP::TotalInfected();
147   else
148     infected = WormTCP::TotalInfected();
149
150   fprintf(tracefile, "%6.2lf %6ld\n", Simulator::Now(), infected);
151   fflush(tracefile);
152
153   cout
154     << "Prog " << Simulator::Now()
155     << " infected "<<infected
156     << " mem " << Simulator::instance->ReportMemoryUsageMB() << "MB"
157     << " bfs calls " << bfs_calls
158     << " bfs avg searches " << (float)bfs_nodessearched/bfs_calls
159     << endl;
160
161   bfs_calls = 0;
162   bfs_nodessearched = 0;
163 }
164
165
166 int main(int argc, char** argv)
167 {
168   Validation::Init(argc, argv);
169   string filename, tracefilename, infofilename;
170   Simulator s;
171   Count_t nd,nf,nt,ndfs, nffs;
172   string hlinkbw, linkbw, blinkbw, baseip;
173   Count_t threads, scanrange, scanrate;
174   Size_t payload;
175   double vulnerability, treelegprob;
176   double simtime, siminterval;
177   unsigned long long int start_time, end_time;
178   Count_t showsim,randomsim;
179
180   cout << "Worm Toolkit Demo on GTNetS  Version 2.0"<<endl;
181   cout << "CoC G.Tech, (c)2003"<<endl<<endl;
182
183   if (argc==1) {
184 // SComF correction:
185 //    cout << "Usage:  wormtcpsim <option1=...> <option2=...> ..."<<endl;
186     cout << "Usage:  GTNetS.exe <option1=...> <option2=...> ..."<<endl;
187     cout << "       Options:"<<endl;
188 // SComF correction:
189 //    cout << "            wormtype   : type of worm 1=UDP or 2=TCP"<<endl<<endl;
190     cout << "            wormtype   : type of worm 1=TCP or 2=UDP"<<endl<<endl;
191     cout << "            tracefile  : name portion of trace files "<<endl<<endl;
192     cout << "            trees      : number of trees"<<endl;
193     cout << "            depth      : depth of each tree"<<endl;
194     cout << "            fanout     : maximum fanout"<<endl;
195     cout << "            depthfs    : depth of first stage in tree"<<endl;
196     cout << "            fanoutfs   : maximum fanout in first stage of tree"<<endl;
197     cout << "            treelegprob: probability that a leg will be created"<<endl;
198     cout << "            linkbw     : bandwidth of links in the tree"<<endl;
199     cout << "            hlinkbw    : bandwidth of links in first stage"<<endl;
200     cout << "            blinkbw    : backbone link bandwidth"<<endl;
201     cout << "            nbaseip    : base IP address of nodes in network"<<endl;
202     cout << "            baseip     : base ip address of worm scan"<<endl;
203     cout << "            scanrange  : IP scanrange of worm (0 scans only the network↙
       space)"<<endl;
```

```
204     cout << "             threads    : (for  TCP) number of parallel TCP connection
        each TCP worm makes"<<endl;
205     cout << "             scanrate   : (for  UDP) number of packets per second"<<endl
        ;
206 // SComF: "vulnerability" added to command line help:
207     cout << "             vulnerability   : The probability that a host is vulnerable
         to the worm attack"<<endl;
208     cout << "             simtime    : the total time of the simulation"<<endl;
209     cout << "             siminterval: the interval for producing trace information"<
        <endl<<endl;
210
211     cout << "Please provide atleast one argument to begin"<<endl<<endl;
212     return 0;
213   }
214
215   Arg("tracefile", filename, DEFFILENAME);
216
217   Arg("wormtype", wormtype, WORMTYPE);
218
219   Arg("trees", nt, TREES);
220   Arg("depth", nd, DEPTH);
221   Arg("fanout", nf, FANOUT);
222   Arg("depthfs", ndfs, DEPTH_FS);
223   Arg("fanoutfs", nffs, FANOUT_FS);
224   Arg("linkbw", linkbw, LINKBW);
225   Arg("hlinkbw", hlinkbw, HLINKBW);
226   Arg("blinkbw", blinkbw, BLINKBW);
227   Arg("baseip", baseip, BASEIP);
228   Arg("scanrange", scanrange, SCANRANGE);
229   Arg("threads", threads, THREADS);
230   Arg("scanrate", scanrate, SCANRATE);
231   Arg("payload", payload, PAYLOAD);
232   Arg("vulnerability", vulnerability, VULNERABILITY);
233   Arg("treelegprob", treelegprob, TREELEGPROB);
234   Arg("simtime", simtime, SIMTIME);
235   Arg("siminterval", siminterval, SIMINTERVAL);
236   Arg("show", showsim, SIMSHOW);
237   Arg("random", randomsim, SIMRANDOM);
238
239   Arg::ProcessArgs(argc, argv);
240
241   // Set the random number generated
242   //  Random::GlobalSeed(30,40,450,60,20,30);
243   if (!randomsim) {
244     Random::GlobalSeed(31731,44543,425345,19367,48201,72333);
245   }
246
247   tracefilename = filename+".dat";
248   infofilename = filename+".inf";
249   cout << "Data:"<< tracefilename <<", Info:"<< infofilename<<endl;
250
251   tracefile = fopen(tracefilename.c_str(), "wt");
252   infofile = fopen(infofilename.c_str(), "wt");
253   // Calculate number of HOST IP's
254
255   Count_t rangeTotalHosts = AddressRangeOfRandomTreeNetworks(nt, nd, nf, ndfs, nffs);
256
257   if (scanrange==0)
258     scanrange = rangeTotalHosts;
259
260   // Now let us set the worm application defaults
261   if (wormtype==UDPWORMTYPE) {
262     WormUDP::SetBaseIP(IPAddr(baseip));
263     WormUDP::SetVulnerability(vulnerability);
264     WormUDP::SetPayloadLength(payload);
265     WormUDP::SetScanRate(scanrate);
266     WormUDP::SetTargetVector(WTVUniform(scanrange));
267   } else {
```

```
268     WormTCP::SetBaseIP(IPAddr(baseip));
269     WormTCP::SetVulnerability(vulnerability);
270     WormTCP::SetPayloadLength(payload);
271     WormTCP::SetConnections(threads);
272     WormTCP::SetTargetVector(WTVUniform(scanrange));
273   }
274
275   // Set node shape to a circle for animation
276   Node::DefaultShape(Node::CIRCLE);
277
278   // Trace* tr = Trace::Instance(); // Get a pointer to global trace object
279   //  tr->Open("testworm.txt");
280   //  TCP::LogFlagsText(true);       // Log TCP flags in text mode
281   //  IPV4::Instance()->SetTrace(Trace::ENABLED);
282
283   Linkp2p lk(Rate(linkbw.c_str()), Time("20ms"));
284   Linkp2p blk(Rate(blinkbw.c_str()), Time("20ms"));
285   Linkp2p hlk(Rate(hlinkbw.c_str()), Time("20ms"));
286
287   if (wormtype == UDPWORMTYPE)
288     CreateRandomTreeNetworksWithWorms(nt, nd, nf, lk, ndfs, nffs,
289                       hlk, treelegprob, blk, IPAddr(baseip), WormUDP());
290   else
291     CreateRandomTreeNetworksWithWorms(nt, nd, nf, lk, ndfs, nffs,
292                       hlk, treelegprob, blk, IPAddr(baseip), WormTCP());
293
294   // Specify animation
295
296   int TotalInstances, TotalVulnerable;
297
298   if (wormtype == UDPWORMTYPE) {
299     TotalInstances = WormUDP::TotalInstances();
300     TotalVulnerable = WormUDP::TotalVulnerable();
301   } else {
302     TotalInstances = WormTCP::TotalInstances();
303     TotalVulnerable = WormTCP::TotalVulnerable();
304   }
305
306   cout<<"-------------------------------------------"<<endl;
307   cout<<"Topology used: trees="<<nt<<" depth="<<nd<<" fanout="<<nf<<endl;
308   cout<<"Link bandwidths "<<linkbw<<", "<<hlinkbw<<", "<<blinkbw<<endl;
309
310   if (wormtype==UDPWORMTYPE) {
311     cout<<"Worm type : UDP"<<endl;
312     cout<<"Scan rate "<<scanrate<<endl;
313   } else {
314     cout<<"Worm type : TCP"<<endl;
315     cout<<"TCP Connections "<<threads<<endl<<endl;
316   }
317
318   cout<<"Worm scans from "<<baseip<<" scan range "<<scanrange<<endl;
319   cout<<endl;
320   cout<<"Total Possible Hosts  : "<<rangeTotalHosts<<endl;
321   cout<<"Total Real Hosts      : "<<TotalInstances<<endl;
322   cout<<"Total Vulnerable hosts: "<<TotalVulnerable<<endl;
323
324
325   fprintf(infofile, "Topology used: trees=%ld depth=%ld fanout=%ld fs_depth=%ld ↙
        fs_fanout=%ld\n",
326       nt, nd, nf, ndfs, nffs);
327
328   fprintf(infofile, "Link bandwidths %s, %s, %s\n", linkbw.c_str(), hlinkbw.c_str(), ↙
        blinkbw.c_str());
329
330   if (wormtype==UDPWORMTYPE) {
331     fprintf(infofile, "UDP worm.\n");
332     fprintf(infofile, "UDP Scanrate    :%ld\n", scanrate);
333   } else {
```

```
334     fprintf(infofile, "TCP worm.\n");
335     fprintf(infofile, "TCP Connections :%ld\n", threads);
336   }
337   fprintf(infofile, "Payload length  :%ld\n", payload);
338   fprintf(infofile, "Worm scan range :%ld\n", scanrange);
339   fprintf(infofile, "Possible hosts  :%ld\n", rangeTotalHosts);
340   fprintf(infofile, "Real hosts      :%ld\n", TotalInstances);
341   fprintf(infofile, "Vulnerable hosts:%ld\n", TotalVulnerable);
342 // SComF: "vulnerability" added:
343   fprintf(infofile, "Vulnerability:%lf\n", vulnerability);
344
345   s.ProgressHook(Progress);
346
347   if (showsim && !Validation::noAnimation) {
348     s.StartAnimation(0, true);
349     s.AnimationUpdateInterval(Time("10us")); // 10us initial update rate
350   }
351
352   s.Progress(siminterval);
353   s.StopAt(simtime);
354
355 // SComF: commented out due to compilation problem (not important)
356 //  start_time = get_time();
357
358   s.Run();
359
360 // SComF: commented out due to compilation problem (not important)
361 //  end_time = get_time();
362
363 // SComF: changed due to compilation problem (not important)
364 // fprintf(infofile, "Total run time :%0.6lf seconds\n", in_seconds(end_time-
      start_time));
365
366   fclose(infofile);
367   fclose(tracefile);
368 }
369
```

B. The SIC Model

B.1. Deriving a PDE from the Differential-Difference Equations

We can write (4.3.1.a) on page 55 as follows:

$$\sum_{n_1=1}^{\infty}\sum_{n_2=1}^{\infty}\frac{dP_{n_1,n_2}(t)}{dt}z_1^{n_1}z_2^{n_2} = \tag{B.1.1}$$
$$\sum_{n_1=1}^{\infty}\sum_{n_2=1}^{\infty}\lambda_1 n_2 P_{n_1-1,n_2}(t)z_1^{n_1}z_2^{n_2}+\sum_{n_1=1}^{\infty}\sum_{n_2=1}^{\infty}\lambda_{r1}(n_1+1)P_{n_1+1,n_2}(t)z_1^{n_1}z_2^{n_2}$$
$$+\sum_{n_1=1}^{\infty}\sum_{n_2=1}^{\infty}\lambda_{r2}(n_2+1)P_{n_1,n_2+1}(t)z_1^{n_1}z_2^{n_2}+\sum_{n_1=1}^{\infty}\sum_{n_2=1}^{\infty}\lambda_2(n_1+1)P_{n_1+1,n_2-1}(t)z_1^{n_1}z_2^{n_2}$$
$$+\sum_{n_1=1}^{\infty}\sum_{n_2=1}^{\infty}\lambda_a(n_2+1)P_{n_1-1,n_2+1}(t)z_1^{n_1}z_2^{n_2}$$
$$-\sum_{n_1=1}^{\infty}\sum_{n_2=1}^{\infty}(\lambda_1 n_2+\lambda_{r1}n_1+\lambda_{r2}n_2+\lambda_2 n_1+\lambda_a n_2)P_{n_1,n_2}(t)z_1^{n_1}z_2^{n_2}$$

And write (4.3.1.b) as follows:

$$\sum_{n_2=1}^{\infty}\frac{dP_{0,n_2}(t)}{dt}z_2^{n_2} = \tag{B.1.2}$$
$$\sum_{n_2=1}^{\infty}\lambda_{r1}P_{1,n_2}(t)z_2^{n_2}+\sum_{n_2=1}^{\infty}\lambda_{r2}(n_2+1)P_{0,n_2+1}(t)z_2^{n_2}+$$
$$\sum_{n_2=1}^{\infty}\lambda_2 P_{1,n_2-1}(t)z_2^{n_2}-\sum_{n_2=1}^{\infty}(\lambda_1 n_2+\lambda_{r2}n_2+\lambda_a n_2)P_{0,n_2}(t)z_2^{n_2}$$

Finally, we write (4.3.1.c) as follows:

$$\sum_{n_1=1}^{\infty} \frac{dP_{n_1,0}(t)}{dt} z_1^{n_1} = \tag{B.1.3}$$
$$\sum_{n_1=1}^{\infty} \lambda_{r1}(n_1+1)P_{n_1+1,0}(t)z_1^{n_1} + \sum_{n_1=1}^{\infty} \lambda_{r2}P_{n_1,1}(t)z_1^{n_1}$$
$$+ \sum_{n_1=1}^{\infty} \lambda_a P_{n_1-1,1}(t)z_1^{n_1} - \sum_{n_1=1}^{\infty} (\lambda_{r1}n_1 + \lambda_2 n_1)P_{n_1,0}(t)z_1^{n_1}$$

We now add together (B.1.1), (B.1.2), (B.1.3), and (4.3.1.d). Here is the result:

$$\frac{\partial P(z_1,z_2,t)}{\partial t} =$$
$$\sum_{n_1=1}^{\infty}\sum_{n_2=1}^{\infty} \lambda_1 n_2 P_{n_1-1,n_2}(t)z_1^{n_1}z_2^{n_2} \tag{B.1.4}$$
$$+ \sum_{n_1=0}^{\infty}\sum_{n_2=0}^{\infty} \lambda_{r1}(n_1+1)P_{n_1+1,n_2}(t)z_1^{n_1}z_2^{n_2} \tag{B.1.5}$$
$$+ \sum_{n_1=0}^{\infty}\sum_{n_2=0}^{\infty} \lambda_{r2}(n_2+1)P_{n_1,n_2+1}(t)z_1^{n_1}z_2^{n_2} \tag{B.1.6}$$
$$+ \sum_{n_1=0}^{\infty}\sum_{n_2=1}^{\infty} \lambda_2(n_1+1)P_{n_1+1,n_2-1}(t)z_1^{n_1}z_2^{n_2} \tag{B.1.7}$$
$$+ \sum_{n_1=1}^{\infty}\sum_{n_2=0}^{\infty} \lambda_a(n_2+1)P_{n_1-1,n_2+1}(t)z_1^{n_1}z_2^{n_2} \tag{B.1.8}$$
$$- \sum_{n_1=0}^{\infty}\sum_{n_2=0}^{\infty} (\lambda_1 n_2 + \lambda_{r1}n_1 + \lambda_{r2}n_2 + \lambda_2 n_1 + \lambda_a n_2)P_{n_1,n_2}(t)z_1^{n_1}z_2^{n_2} \tag{B.1.9}$$

We write (B.1.4) as follows:

$$\sum_{n_1=1}^{\infty}\sum_{n_2=1}^{\infty} \lambda_1 n_2 P_{n_1-1,n_2}(t)z_1^{n_1}z_2^{n_2} \tag{B.1.10}$$
$$= \lambda_1 z_1 \sum_{n_1=1}^{\infty}\sum_{n_2=1}^{\infty} n_2 P_{n_1-1,n_2}(t)z_1^{n_1-1}z_2^{n_2}$$
$$= \lambda_1 z_1 z_2 \sum_{n_1=0}^{\infty}\sum_{n_2=0}^{\infty} n_2 P_{n_1,n_2}(t)z_1^{n_1}z_2^{n_2-1}$$
$$= \lambda_1 z_1 z_2 \frac{\partial P(z_1,z_2,t)}{\partial z_2}$$

And (B.1.5) as follows:

$$\begin{aligned}
&\sum_{n_1=0}^{\infty}\sum_{n_2=0}^{\infty}\lambda_{r1}(n_1+1)P_{n_1+1,n_2}(t)z_1^{n_1}z_2^{n_2} \qquad \text{(B.1.11)}\\
&=\sum_{n_1=0}^{\infty}\sum_{n_2=0}^{\infty}\lambda_{r1}\frac{n_1+1}{z_1}P_{n_1+1,n_2}(t)z_1^{n_1+1}z_2^{n_2}\\
&=\sum_{n_1=1}^{\infty}\sum_{n_2=0}^{\infty}\lambda_{r1}n_1P_{n_1,n_2}(t)z_1^{n_1-1}z_2^{n_2}\\
&=\lambda_{r1}\frac{\partial P(z_1,z_2,t)}{\partial z_1}
\end{aligned}$$

And (B.1.6) as follows:

$$\begin{aligned}
&\sum_{n_1=0}^{\infty}\sum_{n_2=0}^{\infty}\lambda_{r2}(n_2+1)P_{n_1,n_2+1}(t)z_1^{n_1}z_2^{n_2} \qquad \text{(B.1.12)}\\
&=\sum_{n_1=0}^{\infty}\sum_{n_2=0}^{\infty}\lambda_{r2}\frac{n_2+1}{z_2}P_{n_1,n_2+1}(t)z_1^{n_1}z_2^{n_2+1}\\
&=\sum_{n_1=0}^{\infty}\sum_{n_2=1}^{\infty}\lambda_{r2}n_2P_{n_1,n_2}(t)z_1^{n_1}z_2^{n_2-1}\\
&=\lambda_{r2}\frac{\partial P(z_1,z_2,t)}{\partial z_2}
\end{aligned}$$

And (B.1.7) as follows:

$$\begin{aligned}
&\sum_{n_1=0}^{\infty}\sum_{n_2=1}^{\infty}\lambda_2(n_1+1)P_{n_1+1,n_2-1}(t)z_1^{n_1}z_2^{n_2} \qquad \text{(B.1.13)}\\
&=\sum_{n_1=0}^{\infty}\sum_{n_2=1}^{\infty}\lambda_2(n_1+1)\frac{z_2}{z_1}P_{n_1+1,n_2-1}(t)z_1^{n_1+1}z_2^{n_2-1}\\
&=\sum_{n_1=1}^{\infty}\sum_{n_2=0}^{\infty}\lambda_2n_1\frac{z_2}{z_1}P_{n_1,n_2}(t)z_1^{n_1}z_2^{n_2}\\
&=\lambda_2z_2\sum_{n_1=0}^{\infty}\sum_{n_2=0}^{\infty}n_1P_{n_1,n_2}(t)z_1^{n_1-1}z_2^{n_2}=\lambda_2z_2\frac{\partial P(z_1,z_2,t)}{\partial z_1}
\end{aligned}$$

And (B.1.8) as follows:

$$\begin{aligned}
&\sum_{n_1=1}^{\infty}\sum_{n_2=0}^{\infty}\lambda_a(n_2+1)P_{n_1-1,n_2+1}(t)z_1^{n_1}z_2^{n_2} \qquad \text{(B.1.14)}\\
&=\sum_{n_1=1}^{\infty}\sum_{n_2=0}^{\infty}\lambda_a(n_2+1)\frac{z_1}{z_2}P_{n_1-1,n_2+1}(t)z_1^{n_1-1}z_2^{n_2+1}\\
&=\sum_{n_1=0}^{\infty}\sum_{n_2=1}^{\infty}\lambda_a n_2\frac{z_1}{z_2}P_{n_1,n_2}(t)z_1^{n_1}z_2^{n_2}\\
&=\lambda_a z_1\sum_{n_1=0}^{\infty}\sum_{n_2=0}^{\infty}n_2P_{n_1,n_2}(t)z_1^{n_1}z_2^{n_2-1}=\lambda_a z_1\frac{\partial P(z_1,z_2,t)}{\partial z_2}
\end{aligned}$$

Finally, (B.1.9) as follows:

$$\begin{aligned}
&\sum_{n_1=0}^{\infty}\sum_{n_2=0}^{\infty}(\lambda_1 n_2+\lambda_{r1}n_1+\lambda_{r2}n_2+\lambda_2 n_1+\lambda_a n_2)P_{n_1,n_2}(t)z_1^{n_1}z_2^{n_2}\\
&=\sum_{n_1=0}^{\infty}\sum_{n_2=0}^{\infty}(\lambda_{r1}+\lambda_2)n_1P_{n_1,n_2}(t)z_1^{n_1}z_2^{n_2}\\
&+\sum_{n_1=0}^{\infty}\sum_{n_2=0}^{\infty}(\lambda_1+\lambda_{r2}+\lambda_a)n_2P_{n_1,n_2}(t)z_1^{n_1}z_2^{n_2}\\
&=(\lambda_{r1}+\lambda_2)z_1\sum_{n_1=0}^{\infty}\sum_{n_2=0}^{\infty}n_1P_{n_1,n_2}(t)z_1^{n_1-1}z_2^{n_2}\\
&+(\lambda_1+\lambda_{r2}+\lambda_a)z_2\sum_{n_1=0}^{\infty}\sum_{n_2=0}^{\infty}n_2P_{n_1,n_2}(t)z_1^{n_1}z_2^{n_2-1}\\
&=(\lambda_{r1}+\lambda_2)z_1\frac{\partial P(z_1,z_2,t)}{\partial z_1}+(\lambda_1+\lambda_{r2}+\lambda_a)z_2\frac{\partial P(z_1,z_2,t)}{\partial z_2} \qquad \text{(B.1.15)}
\end{aligned}$$

Replacing (B.1.4) through (B.1.9) with the ones derived in (B.1.10) through (B.1.15), after simplification, we arrive at (4.3.2) on page 56.

B.2. Attempt to Solve the PDE Using Method of Characteristics

We describe our efforts to solve the partial differential equation (4.3.2) on page 56 describing the system. Following the Method of Characteristics [62, p.432] to solve PDEs, based on (4.3.2), we can write:

$$\begin{cases} \frac{\partial t}{\partial s} = -1 & (a) \\ \frac{dP}{ds} = 0 & (b) \\ \frac{\partial z_1}{\partial s} = \lambda_{r1} + \lambda_2 z_2 - \lambda_{r1} z_1 - \lambda_2 z_1 & (c) \\ \frac{\partial z_2}{\partial s} = \lambda_1 z_1 z_2 + \lambda_{r2} + \lambda_a z_1 - \lambda_1 z_2 - \lambda_{r2} z_2 - \lambda_a z_2 & (d) \end{cases} \tag{B.2.1}$$

where s is a parametric variable and $P = P(z_1, z_2, t)$ is the PGF. With the initial condition $P(z_1, z_2, 0) = z_1^{k_1} z_2^{k_2}$, we therefore have:

$$\begin{cases} t(s=0) = 0 & (a) \\ z_1(s=0) = i_1 & (b) \\ z_2(s=0) = i_2 & (c) \\ P(s=0) = i_1^{k_1} i_2^{k_2} & (d) \end{cases} \tag{B.2.2}$$

From (B.2.1.a) and (B.2.2.a), we have:

$$t = -s \tag{B.2.3}$$

Likewise, from (B.2.1.b) and (B.2.2.d), we have:

$$P = \left(i_1(z_1, z_2, t)\right)^{k_1} \left(i_2(z_1, z_2, t)\right)^{k_2} \tag{B.2.4}$$

Equations (B.2.1.c) and (B.2.1.d) are "non-separable", i.e., we cannot derive z_1 and z_2 from 1st order ordinary differential equations (ODEs). We therefore proceed as follows: from (B.2.1.c), we derive z_2:

$$z_2 = \frac{1}{\lambda_2}[\frac{dz_1}{ds} + (\lambda_{r1} + \lambda_2)z_1 - \lambda_{r1}] \tag{B.2.5}$$

Replacing z_2 in (B.2.1.d) with the expression given in (B.2.5), after some simplifications, we can write (B.2.1.d) as follows:

$$\begin{aligned}\frac{d^2z_1}{ds^2} + (\lambda_{r1} + \lambda_2 + \lambda_1 + \lambda_{r2} + \lambda_a)\frac{dz_1}{ds} - \lambda_1 z_1 \frac{dz_1}{ds} - \lambda_1(\lambda_{r1} + \lambda_2)z_1^2 \\ + ((\lambda_1 + \lambda_{r2} + \lambda_a)(\lambda_{r1} + \lambda_2) + \lambda_1\lambda_{r1} - \lambda_a\lambda_2)\, z_1 \\ -(\lambda_1 + \lambda_{r2} + \lambda_a)\lambda_{r1} - \lambda_{r2}\lambda_2 = 0\end{aligned} \tag{B.2.6}$$

Equation (B.2.6) has the form of a second order Lienard equation [82] given below:

$$\frac{d^2z_1}{ds^2} + (A + Bz_1)\frac{dz_1}{ds} + Cz_1^2 + Dz_1 + E = 0 \tag{B.2.7}$$

Equation (B.2.7) is not in the form of solvable cases presented in [82, Sec.2.2.3-2], [83, pp.204-5], and [84]. As a further attempt to solve (B.2.7), we have used the following substitution suggested in [82, Sec.2.2.3-1]:

$$w = \frac{dz_1}{ds} \quad , \quad \frac{d^2z_1}{ds^2} = w'_s = w'_{z_1}\frac{dz_1}{ds} = w'_{z_1}w \tag{B.2.8}$$

The above substitution transformed (B.2.7) into an Abel equation of the 2nd kind given below:

$$ww'_{z_1} + (A + Bz_1)w + Cz_1^2 + Dz_1 + E = 0 \tag{B.2.9}$$

Equation (B.2.9) is also not among the solvable cases presented in [82, Sec.1.3.3-2].

B.3. Derivation of Means from the PDE of the PGF

We take the derivative of (4.3.2) on page 56 with respect to z_1 as follows:

$$\begin{aligned}(-\lambda_{r1}-\lambda_2)\frac{\partial P(z_1,z_2,t)}{\partial z_1}\\ +(\lambda_{r1}+\lambda_2 z_2-\lambda_{r1}z_1-\lambda_2 z_1)\frac{\partial^2 P(z_1,z_2,t)}{\partial z_1^2}+(\lambda_1 z_2+\lambda_a)\frac{\partial P(z_1,z_2,t)}{\partial z_2}\\ +(\lambda_1 z_1 z_2+\lambda_{r2}+\lambda_a z_1-\lambda_1 z_2-\lambda_{r2}z_2-\lambda_a z_2)\frac{\partial^2 P(z_1,z_2,t)}{\partial z_2 \partial z_1}-\frac{\partial^2 P(z_1,z_2,t)}{\partial t \partial z_1}=0\end{aligned} \quad \text{(B.3.1)}$$

Setting $z_1 = z_2 = 1$ in (B.3.1) gives us the following equation:

$$\frac{dE_1(t)}{dt}+(\lambda_2+\lambda_{r1})E_1(t)-(\lambda_1+\lambda_a)E_2(t)=0 \quad \text{(B.3.2)}$$

We then take the derivative of (4.3.2) with respect to z_2 as follows:

$$\begin{aligned}\lambda_2\frac{\partial P(z_1,z_2,t)}{\partial z_1}+(\lambda_{r1}+\lambda_2 z_2-\lambda_{r1}z_1-\lambda_2 z_1)\frac{\partial^2 P(z_1,z_2,t)}{\partial z_2 \partial z_1}\\ +(\lambda_1 z_1-\lambda_1-\lambda_{r2}-\lambda_a)\frac{\partial P(z_1,z_2,t)}{\partial z_2}\\ +(\lambda_1 z_1 z_2+\lambda_{r2}+\lambda_a z_1-\lambda_1 z_2-\lambda_{r2}z_2-\lambda_a z_2)\frac{\partial^2 P(z_1,z_2,t)}{\partial z_2^2}-\frac{\partial^2 P(z_1,z_2,t)}{\partial t \partial z_2}=0\end{aligned} \quad \text{(B.3.3)}$$

Setting $z_1 = z_2 = 1$ in (B.3.3) gives us the following equation:

$$\frac{dE_2(t)}{dt}-\lambda_2 E_1(t)+(\lambda_{r2}+\lambda_a)E_2(t)=0 \quad \text{(B.3.4)}$$

Re-arranging (B.3.2) and (B.3.4) gives us (4.3.4).

Taking (4.3.4) to Laplace domain, we can write:

$$\begin{cases} sE_1(s \mid k_1,k_2)-k_1+(\lambda_{r1}+\lambda_2)E_1(s \mid k_1,k_2)-(\lambda_1+\lambda_a)E_2(s \mid k_1,k_2)=0 \\ sE_2(s \mid k_1,k_2)-k_2-\lambda_2 E_1(s \mid k_1,k_2)+(\lambda_{r2}+\lambda_a)E_2(s \mid k_1,k_2)=0 \end{cases} \quad \text{(B.3.5)}$$

where k_1 and k_2 are values of n_1 and n_2 at $t = 0$, respectively. Note that k_1 and k_2 are variables themselves and their means are obtained as follows:

$$\begin{cases} \bar{k_1} = \sum_{k_1=0}^{\infty} \sum_{k_2=0}^{\infty} k_1 P_{k_1,k_2}(t=0) \\ \bar{k_2} = \sum_{k_1=0}^{\infty} \sum_{k_2=0}^{\infty} k_2 P_{k_1,k_2}(t=0) \end{cases} \tag{B.3.6}$$

$\bar{k_1}$ and $\bar{k_2}$ are therefore the values of the means at $t = 0$. We then proceed to uncondition (B.3.5), i.e., we take $\sum_{k_1=0}^{\infty} \sum_{k_2=0}^{\infty} \{X\} P_{k_1,k_2}(t=0)$, with X being each element of the equation set. After simplification, we have:

$$\begin{cases} sE_1(s) - \bar{k_1} + (\lambda_{r1} + \lambda_2)E_1(s) - (\lambda_1 + \lambda_a)E_2(s) = 0 \\ sE_2(s) - \bar{k_2} - \lambda_2 E_1(s) + (\lambda_{r2} + \lambda_a)E_2(s) = 0 \end{cases} \tag{B.3.7}$$

$E_1(s)$ and $E_2(s)$ are then obtained as follows:

$$E_1(s) = \frac{\bar{k_1}s + \bar{k_2}(\lambda_1 + \lambda_a) + \bar{k_1}(\lambda_{r2} + \lambda_a)}{s^2 + (\lambda_{r2} + \lambda_a + \lambda_{r1} + \lambda_2)s + (\lambda_{r2} + \lambda_a)(\lambda_{r1} + \lambda_2) - \lambda_2(\lambda_1 + \lambda_a)} \tag{B.3.8}$$

$$E_2(s) = \frac{s + \lambda_2 + \lambda_{r1}}{\lambda_1 + \lambda_a} E_1(s) - \frac{\bar{k_1}}{\lambda_1 + \lambda_a} \tag{B.3.9}$$

Finally, the inverse Laplace of $E_1(s)$ and $E_2(s)$ are obtained as shown in (4.3.5) and (4.3.6) on page 57. Note that another method to derive the differential equations of means is documented in Section B.6.

B.4. Derivation of Variances from the PDE of the PGF

Taking the derivative of (B.3.1) with respect to z_1 (i.e., deriving the 2nd derivative of (4.3.2) on page 56 with respect to z_1), we have:

$$\begin{aligned}
(-\lambda_{r1}-\lambda_2)\frac{\partial^2 P(z_1,z_2,t)}{\partial z_1^2}+(-\lambda_{r1}-\lambda_2)\frac{\partial^2 P(z_1,z_2,t)}{\partial z_1^2} \\
+(\lambda_{r1}+\lambda_2 z_2-\lambda_{r1}z_1-\lambda_2 z_1)\frac{\partial^3 P(z_1,z_2,t)}{\partial z_1^3}+(\lambda_1 z_2+\lambda_a)\frac{\partial^2 P(z_1,z_2,t)}{\partial z_2 \partial z_1} \\
+(\lambda_1 z_2+\lambda_a)\frac{\partial^2 P(z_1,z_2,t)}{\partial z_2 \partial z_1} \\
+(\lambda_1 z_1 z_2+\lambda_{r2}+\lambda_a z_1-\lambda_1 z_2-\lambda_{r2}z_2-\lambda_a z_2)\frac{\partial^3 P(z_1,z_2,t)}{\partial z_2 \partial z_1^2} \\
-\frac{\partial^3 P(z_1,z_2,t)}{\partial t \partial z_1^2}=0 \qquad \text{(B.4.1)}
\end{aligned}$$

Likewise, taking the derivative of (B.3.3) with respect to z_2 (i.e., deriving the 2nd derivative of (4.3.2) with respect to z_2), we have:

$$\begin{aligned}
\lambda_2\frac{\partial^2 P(z_1,z_2,t)}{\partial z_1 \partial z_2}+\lambda_2\frac{\partial^2 P(z_1,z_2,t)}{\partial z_1 \partial z_2} \\
+(\lambda_{r1}+\lambda_2 z_2-\lambda_{r1}z_1-\lambda_2 z_1)\frac{\partial^3 P(z_1,z_2,t)}{\partial z_2^2 \partial z_1} \\
+(\lambda_1 z_1-\lambda_1-\lambda_{r2}-\lambda_a)\frac{\partial^2 P(z_1,z_2,t)}{\partial z_2^2} \\
+(\lambda_1 z_1-\lambda_1-\lambda_{r2}-\lambda_a)\frac{\partial^2 P(z_1,z_2,t)}{\partial z_2^2} \\
+(\lambda_1 z_1 z_2+\lambda_{r2}+\lambda_a z_1-\lambda_1 z_2-\lambda_{r2}z_2-\lambda_a z_2)\frac{\partial^3 P(z_1,z_2,t)}{\partial z_2^3} \\
-\frac{\partial^3 P(z_1,z_2,t)}{\partial t \partial z_2^2}=0 \qquad \text{(B.4.2)}
\end{aligned}$$

Finally, taking the derivative of (B.3.1) with respect to z_2, we have:

$$\begin{aligned}
&(-\lambda_{r1}-\lambda_2)\frac{\partial^2 P(z_1,z_2,t)}{\partial z_1 \partial z_2}+\lambda_2\frac{\partial^2 P(z_1,z_2,t)}{\partial z_1^2}\\
&+(\lambda_{r1}+\lambda_2 z_2-\lambda_{r1}z_1-\lambda_2 z_1)\frac{\partial^3 P(z_1,z_2,t)}{\partial z_1^2 \partial z_2}\\
&+\lambda_1\frac{\partial P(z_1,z_2,t)}{\partial z_2}+(\lambda_1 z_2+\lambda_a)\frac{\partial^2 P(z_1,z_2,t)}{\partial z_2^2}\\
&+(\lambda_1 z_1-\lambda_1-\lambda_{r2}-\lambda_a)\frac{\partial^2 P(z_1,z_2,t)}{\partial z_2 \partial z_1}\\
&+(\lambda_1 z_1 z_2+\lambda_{r2}+\lambda_a z_1-\lambda_1 z_2-\lambda_{r2}z_2-\lambda_a z_2)\frac{\partial^3 P(z_1,z_2,t)}{\partial z_2^2 \partial z_1}\\
&-\frac{\partial^3 P(z_1,z_2,t)}{\partial t \partial z_1 \partial z_2}=0
\end{aligned} \tag{B.4.3}$$

Setting $z_1 = z_2 = 1$ in (B.4.1), (B.4.2), and (B.4.3) gives us (4.3.12).

In (4.3.12), we have three ODEs and three variables ($\psi_1(t)$, $\psi_{12}(t)$, and $\psi_2(t)$); therefore, we can find a unique solution by solving this system of linear ODEs. Taking (4.3.12) to Laplace domain, we have:

$$\begin{cases}
s\psi_1(s\mid k_1,k_2)-k_1^2+k_1 = 2(\lambda_1+\lambda_a)\psi_{12}(s\mid k_1,k_2)-2(\lambda_{r1}+\lambda_2)\psi_1(s\mid k_1,k_2)\\
s\psi_2(s\mid k_1,k_2)-k_2^2+k_2 = 2\lambda_2\psi_{12}(s\mid k_1,k_2)-2(\lambda_{r2}+\lambda_a)\psi_2(s\mid k_1,k_2)\\
s\psi_{12}(s\mid k_1,k_2)-k_1k_2 = -(\lambda_{r1}+\lambda_2+\lambda_{r2}+\lambda_a)\psi_{12}(s\mid k_1,k_2)+\lambda_2\psi_1(s\mid k_1,k_2)\\
\qquad\qquad +\lambda_1 E_2(s\mid k_1,k_2)+(\lambda_1+\lambda_a)\psi_2(s\mid k_1,k_2)
\end{cases} \tag{B.4.4}$$

Like before, we then proceed to uncondition (B.4.4). After simplification, we have:

$$\begin{cases}
s\psi_1(s)-\bar{k_1^2}+\bar{k_1} = 2(\lambda_1+\lambda_a)\psi_{12}(s)-2(\lambda_{r1}+\lambda_2)\psi_1(s)\\
s\psi_2(s)-\bar{k_2^2}+\bar{k_2} = 2\lambda_2\psi_{12}(s)-2(\lambda_{r2}+\lambda_a)\psi_2(s)\\
s\psi_{12}(s)-\bar{k_1k_2} = -(\lambda_{r1}+\lambda_2+\lambda_{r2}+\lambda_a)\psi_{12}(s)+\lambda_2\psi_1(s)\\
\qquad\qquad +\lambda_1 E_2(s)+(\lambda_1+\lambda_a)\psi_2(s)
\end{cases} \tag{B.4.5}$$

The solution of (B.4.5) (i.e., the expressions for $\psi_1(s)$ and $\psi_2(s)$) is provided in Section B.5. On the other hand, the expressions for $\sigma_1^2(t)$ and $\sigma_2^2(t)$ are extremely lengthy; hence, they are provided in [85] instead. Note that another method to derive the variances is documented in Section B.6.

B.5. ψ Formulas (Laplace Domain)

$$\psi_1(s) =$$
$$\frac{\left(\overline{k_2}-\overline{k_2^2}\right)(\lambda_1+\lambda_a)}{\lambda_2\,(s+2\,(\lambda_a+\lambda_{r2}))}-\frac{\lambda_1\left(\overline{k_1}\,\lambda_2+\overline{k_2}\,(s+\lambda_2+\lambda_{r1})\right)+\overline{k_1}\,\overline{k_2}\,(s^2-\lambda_1\,\lambda_2+\lambda_a\,\lambda_{r1}+\lambda_2\,\lambda_{r2}+\lambda_{r1}\,\lambda_{r2}+s\,(\lambda_2+\lambda_a+\lambda_{r1}+\lambda_{r2}))}{\lambda_2\,(s^2-\lambda_1\,\lambda_2+\lambda_a\,\lambda_{r1}+\lambda_2\,\lambda_{r2}+\lambda_{r1}\,\lambda_{r2}+s\,(\lambda_2+\lambda_a+\lambda_{r1}+\lambda_{r2}))}-$$
$$\left(\left(-\frac{s+\lambda_2+\lambda_a+\lambda_{r1}+\lambda_{r2}}{\lambda_2}+\frac{2\,(\lambda_1+\lambda_a)}{s+2\,(\lambda_a+\lambda_{r2})}\right)\right.$$
$$\left(-\left(\overline{k_2}-\overline{k_2^2}\right)(\lambda_1+\lambda_a)\,(s+2\,(\lambda_2+\lambda_{r1}))+(s+2\,(\lambda_a+\lambda_{r2}))\left(\left(-\overline{k_1}+\overline{k_1^2}\right)\lambda_2+\left((s+2\,(\lambda_2+\lambda_{r1}))\right.\right.\right.$$
$$\left.\left.\left(\lambda_1\left(\overline{k_1}\,\lambda_2+\overline{k_2}\,(s+\lambda_2+\lambda_{r1})\right)+\overline{k_1}\,\overline{k_2}\,(s^2-\lambda_1\,\lambda_2+\lambda_a\,\lambda_{r1}+\lambda_2\,\lambda_{r2}+\lambda_{r1}\,\lambda_{r2}+s\,(\lambda_2+\lambda_a+\lambda_{r1}+\lambda_{r2}))\right)\right)\Big/$$
$$\left.\left.(s^2-\lambda_1\,\lambda_2+\lambda_a\,\lambda_{r1}+\lambda_2\,\lambda_{r2}+\lambda_{r1}\,\lambda_{r2}+s\,(\lambda_2+\lambda_a+\lambda_{r1}+\lambda_{r2}))\right)\right)\Big/$$
$$\left((s+\lambda_2+\lambda_a+\lambda_{r1}+\lambda_{r2})\,(s^2+2\,s\,(\lambda_2+\lambda_a+\lambda_{r1}+\lambda_{r2})+4\,(-\lambda_1\,\lambda_2+\lambda_a\,\lambda_{r1}+(\lambda_2+\lambda_{r1})\,\lambda_{r2}))\right)$$

$$\psi_{12}(s) =$$
$$\left(-\left(\overline{k_2}-\overline{k_2^2}\right)(\lambda_1+\lambda_a)\,(s+2\,(\lambda_2+\lambda_{r1}))+(s+2\,(\lambda_a+\lambda_{r2}))\left(\left(-\overline{k_1}+\overline{k_1^2}\right)\lambda_2+\left((s+2\,(\lambda_2+\lambda_{r1}))\left(\lambda_1\left(\overline{k_1}\,\lambda_2+\overline{k_2}\,(s+\lambda_2+\lambda_{r1})\right)+\right.\right.\right.$$
$$\left.\left.\left.\overline{k_1}\,\overline{k_2}\,(s^2-\lambda_1\,\lambda_2+\lambda_a\,\lambda_{r1}+\lambda_2\,\lambda_{r2}+\lambda_{r1}\,\lambda_{r2}+s\,(\lambda_2+\lambda_a+\lambda_{r1}+\lambda_{r2}))\right)\right)\Big/$$
$$\left.\left.(s^2-\lambda_1\,\lambda_2+\lambda_a\,\lambda_{r1}+\lambda_2\,\lambda_{r2}+\lambda_{r1}\,\lambda_{r2}+s\,(\lambda_2+\lambda_a+\lambda_{r1}+\lambda_{r2}))\right)\right)\Big/$$
$$\left((s+\lambda_2+\lambda_a+\lambda_{r1}+\lambda_{r2})\,(s^2+2\,s\,(\lambda_2+\lambda_a+\lambda_{r1}+\lambda_{r2})+4\,(-\lambda_1\,\lambda_2+\lambda_a\,\lambda_{r1}+(\lambda_2+\lambda_{r1})\,\lambda_{r2}))\right)$$

$$\psi_2(s)=\frac{1}{s+2\,(\lambda_a+\lambda_{r2})}$$
$$\left(-\overline{k_2}+\overline{k_2^2}+\left(2\,\lambda_2\left(-\left(\overline{k_2}-\overline{k_2^2}\right)(\lambda_1+\lambda_a)\,(s+2\,(\lambda_2+\lambda_{r1}))+(s+2\,(\lambda_a+\lambda_{r2}))\left(\left(-\overline{k_1}+\overline{k_1^2}\right)\lambda_2+\left((s+2\,(\lambda_2+\lambda_{r1}))\right.\right.\right.\right.$$
$$\left.\left.\left.\left.\left(\lambda_1\left(\overline{k_1}\,\lambda_2+\overline{k_2}\,(s+\lambda_2+\lambda_{r1})\right)+\overline{k_1}\,\overline{k_2}\,(s^2-\lambda_1\,\lambda_2+\lambda_a\,\lambda_{r1}+\lambda_2\,\lambda_{r2}+\lambda_{r1}\,\lambda_{r2}+s\,(\lambda_2+\lambda_a+\lambda_{r1}+\right.\right.\right.\right.\right.$$
$$\left.\left.\left.\left.\left.\lambda_{r2}))\right)\right)\Big/(s^2-\lambda_1\,\lambda_2+\lambda_a\,\lambda_{r1}+\lambda_2\,\lambda_{r2}+\lambda_{r1}\,\lambda_{r2}+s\,(\lambda_2+\lambda_a+\lambda_{r1}+\lambda_{r2}))\right)\right)\Big/$$
$$\left.\left.\left((s+\lambda_2+\lambda_a+\lambda_{r1}+\lambda_{r2})\,(s^2+2\,s\,(\lambda_2+\lambda_a+\lambda_{r1}+\lambda_{r2})+4\,(-\lambda_1\,\lambda_2+\lambda_a\,\lambda_{r1}+(\lambda_2+\lambda_{r1})\,\lambda_{r2}))\right)\right)$$

B.6. Direct Derivation of Means and Variances from Probability Flow Differential-difference Equations

B.6.1. Derivation of Means

We derive the means directly from the probability flow differential-difference equations of (4.3.1) on page 55.

B.6.1.1. Deriving $E_1(t)$

As $E_1(t) = E_t[n_1] = \sum_{n_1=0}^{\infty}\sum_{n_2=0}^{\infty} n_1 P_{n_1,n_2}(t)$, we proceed as follows: we multiply (4.3.1.a,c) by n_1 and sum over n_1 and n_2:

$$\begin{aligned}
&\sum_{n_1=1}^{\infty}\sum_{n_2=1}^{\infty} n_1 \frac{dP_{n_1,n_2}(t)}{dt} = \\
&\sum_{n_1=1}^{\infty}\sum_{n_2=1}^{\infty} n_1\lambda_1 n_2 P_{n_1-1,n_2}(t) + \sum_{n_1=1}^{\infty}\sum_{n_2=1}^{\infty} n_1\lambda_{r1}(n_1+1)P_{n_1+1,n_2}(t) + \\
&\sum_{n_1=1}^{\infty}\sum_{n_2=1}^{\infty} n_1\lambda_{r2}(n_2+1)P_{n_1,n_2+1}(t) + \sum_{n_1=1}^{\infty}\sum_{n_2=1}^{\infty} n_1\lambda_2(n_1+1)P_{n_1+1,n_2-1}(t) + \\
&\sum_{n_1=1}^{\infty}\sum_{n_2=1}^{\infty} n_1\lambda_a(n_2+1)P_{n_1-1,n_2+1}(t) - \\
&\sum_{n_1=1}^{\infty}\sum_{n_2=1}^{\infty} n_1(\lambda_1 n_2 + \lambda_{r1}n_1 + \lambda_{r2}n_2 + \lambda_2 n_1 + \lambda_a n_2)P_{n_1,n_2}(t)
\end{aligned} \tag{B.6.1}$$

$$\begin{aligned}
\sum_{n_1=1}^{\infty} n_1 \frac{dP_{n_1,0}(t)}{dt} &= \sum_{n_1=1}^{\infty} n_1\lambda_{r1}(n_1+1)P_{n_1+1,0}(t) + \sum_{n_1=1}^{\infty} n_1\lambda_{r2}P_{n_1,1}(t) + \\
&\quad \sum_{n_1=1}^{\infty} n_1\lambda_a P_{n_1-1,1}(t) - \sum_{n_1=1}^{\infty} n_1(\lambda_{r1}n_1 + \lambda_2 n_1)P_{n_1,0}(t)
\end{aligned} \tag{B.6.2}$$

Adding the previous two equations together, after some manipulations, we have:

$$\begin{aligned}\frac{dE_1(t)}{dt} &= \lambda_1 E_t[n_1 n_2] + \lambda_1 E_2(t) + \lambda_{r1} E_t[n_1^2] - \lambda_{r1} E_1(t) \\ &\quad + \lambda_{r2} E_t[n_1 n_2] + \lambda_2 E_t[n_1^2] - \lambda_2 E_1(t) \\ &\quad + \lambda_a E_t[n_1 n_2] + \lambda_a E_2(t) \\ &\quad - (\lambda_{r1} + \lambda_2) E_t[n_1^2] - (\lambda_1 + \lambda_{r2} + \lambda_a) E_t[n_1 n_2]\end{aligned} \tag{B.6.3}$$

The preceding equation simplifies to the following:

$$\frac{dE_1(t)}{dt} = (\lambda_1 + \lambda_a) E_2(t) - (\lambda_{r1} + \lambda_2) E_1(t) \tag{B.6.4}$$

Equation (B.6.4) is the same as (B.3.2). The equation for $\frac{dE_2(t)}{dt}$ can also be derived following a similar procedure. Therefore, this method is also a feasible method to derive the equations for means.

B.6.2. Derivation of Variances

Here, we derive the variances directly from the probability flow differential-difference equations of (4.3.1). We know:

$$\sigma_1^2(t) = E_t[n_1^2] - (E_1(t))^2 \tag{B.6.5}$$

We derived $E_1(t)$ and $E_2(t)$ earlier[1]; we now need to derive $E_t[n_1^2]$ and $E_t[n_2^2]$.

[1] $E_1(t) \equiv E_t[n_1]$ and $E_2(t) \equiv E_t[n_2]$.

B.6.2.1. 2nd Moment DEs

Derivation of $\frac{dE_t[n_1^2]}{dt}$: As $E_t[n_1^2] = \sum_{n_1=0}^{\infty}\sum_{n_2=0}^{\infty} n_1^2 P_{n_1,n_2}(t)$, we proceed as follows: we multiply (4.3.1.a,c) by n_1^2 and sum over n_1 and n_2:

$$\begin{aligned}
&\sum_{n_1=1}^{\infty}\sum_{n_2=1}^{\infty} n_1^2 \frac{dP_{n_1,n_2}(t)}{dt} = \\
&\sum_{n_1=1}^{\infty}\sum_{n_2=1}^{\infty} n_1^2 \lambda_1 n_2 P_{n_1-1,n_2}(t) + \sum_{n_1=1}^{\infty}\sum_{n_2=1}^{\infty} n_1^2 \lambda_{r1}(n_1+1) P_{n_1+1,n_2}(t) + \\
&\sum_{n_1=1}^{\infty}\sum_{n_2=1}^{\infty} n_1^2 \lambda_{r2}(n_2+1) P_{n_1,n_2+1}(t) + \sum_{n_1=1}^{\infty}\sum_{n_2=1}^{\infty} n_1^2 \lambda_2(n_1+1) P_{n_1+1,n_2-1}(t) + \\
&\sum_{n_1=1}^{\infty}\sum_{n_2=1}^{\infty} n_1^2 \lambda_a(n_2+1) P_{n_1-1,n_2+1}(t) - \\
&\sum_{n_1=1}^{\infty}\sum_{n_2=1}^{\infty} n_1^2 (\lambda_1 n_2 + \lambda_{r1} n_1 + \lambda_{r2} n_2 + \lambda_2 n_1 + \lambda_a n_2) P_{n_1,n_2}(t)
\end{aligned} \tag{B.6.6}$$

$$\begin{aligned}
\sum_{n_1=1}^{\infty} n_1^2 \frac{dP_{n_1,0}(t)}{dt} &= \sum_{n_1=1}^{\infty} n_1^2 \lambda_{r1}(n_1+1) P_{n_1+1,0}(t) + \sum_{n_1=1}^{\infty} n_1^2 \lambda_{r2} P_{n_1,1}(t) + \\
&\quad \sum_{n_1=1}^{\infty} n_1^2 \lambda_a P_{n_1-1,1}(t) - \sum_{n_1=1}^{\infty} n_1^2 (\lambda_{r1} n_1 + \lambda_2 n_1) P_{n_1,0}(t)
\end{aligned} \tag{B.6.7}$$

Adding the previous two equations together, after some manipulations (see Sub-section B.6.3), we have:

$$\begin{aligned}
\frac{dE_t[n_1^2]}{dt} &= \lambda_1 E_t[n_1^2 n_2] + 2\lambda_1 E_t[n_1 n_2] + \lambda_1 E_2(t) \\
&\quad + \lambda_{r1} E_t[n_1^3] - 2\lambda_{r1} E_t[n_1^2] + \lambda_{r1} E_1(t) + \lambda_{r2} E_t[n_1^2 n_2] \\
&\quad + \lambda_2 E_t[n_1^3] - 2\lambda_2 E_t[n_1^2] + \lambda_2 E_1(t) \\
&\quad + \lambda_a E_t[n_1^2 n_2] + 2\lambda_a E_t[n_1 n_2] + \lambda_a E_2(t) \\
&\quad - (\lambda_1 + \lambda_{r2} + \lambda_a) E_t[n_1^2 n_2] - (\lambda_{r1} + \lambda_2) E_t[n_1^3]
\end{aligned} \tag{B.6.8}$$

The preceding equation simplifies to the following:

$$\frac{dE_t[n_1^2]}{dt} = 2(\lambda_1+\lambda_a)E_t[n_1n_2] - 2(\lambda_{r1}+\lambda_2)E_t[n_1^2] + (\lambda_{r1}+\lambda_2)E_1(t) + (\lambda_1+\lambda_a)E_2(t) \tag{B.6.9}$$

Derivation of $\frac{dE_t[n_1n_2]}{dt}$: As $E_t[n_1n_2] = \sum_{n_1=0}^{\infty}\sum_{n_2=0}^{\infty} n_1n_2P_{n_1,n_2}(t)$, we proceed as follows: we multiply (4.3.1.a) by n_1n_2 and sum over n_1 and n_2:

$$\begin{aligned}
&\sum_{n_1=1}^{\infty}\sum_{n_2=1}^{\infty} n_1n_2\frac{dP_{n_1,n_2}(t)}{dt} = \\
&\sum_{n_1=1}^{\infty}\sum_{n_2=1}^{\infty} n_1n_2\lambda_1n_2P_{n_1-1,n_2}(t) + \sum_{n_1=1}^{\infty}\sum_{n_2=1}^{\infty} n_1n_2\lambda_{r1}(n_1+1)P_{n_1+1,n_2}(t) + \\
&\sum_{n_1=1}^{\infty}\sum_{n_2=1}^{\infty} n_1n_2\lambda_{r2}(n_2+1)P_{n_1,n_2+1}(t) + \sum_{n_1=1}^{\infty}\sum_{n_2=1}^{\infty} n_1n_2\lambda_2(n_1+1)P_{n_1+1,n_2-1}(t) + \\
&\sum_{n_1=1}^{\infty}\sum_{n_2=1}^{\infty} n_1n_2\lambda_a(n_2+1)P_{n_1-1,n_2+1}(t) - \\
&\sum_{n_1=1}^{\infty}\sum_{n_2=1}^{\infty} n_1n_2(\lambda_1n_2+\lambda_{r1}n_1+\lambda_{r2}n_2+\lambda_2n_1+\lambda_an_2)P_{n_1,n_2}(t)
\end{aligned} \tag{B.6.10}$$

After some manipulations (see Sub-section B.6.3), we have:

$$\begin{aligned}
\frac{dE_t[n_1n_2]}{dt} &= \lambda_1E_t[n_1n_2^2] + \lambda_1E_t[n_2^2] + \lambda_{r1}E_t[n_1^2n_2] - \lambda_{r1}E_t[n_1n_2] \\
&+\lambda_{r2}E_t[n_1n_2^2] - \lambda_{r2}E_t[n_1n_2] + \lambda_2\left(E_t[n_1^2n_2] + E_t[n_1^2] - E_t[n_1n_2] - E_1(t)\right) \\
&+\lambda_a\left(E_t[n_1n_2^2] - E_t[n_1n_2] + E_t[n_2^2] - E_2(t)\right) \\
&-(\lambda_{r1}+\lambda_2)E_t[n_1^2n_2] - (\lambda_1+\lambda_{r2}+\lambda_a)E_t[n_1n_2^2]
\end{aligned} \tag{B.6.11}$$

The preceding equation simplifies to the following:

$$\frac{dE_t[n_1n_2]}{dt} = \lambda_2E_t[n_1^2] + (\lambda_1+\lambda_a)E_t[n_2^2] - (\lambda_{r1}+\lambda_{r2}+\lambda_2+\lambda_a)E_t[n_1n_2] - \lambda_2E_1(t) - \lambda_aE_2(t) \tag{B.6.12}$$

Derivation of $\frac{dE_t[n_2^2]}{dt}$: As $E_t[n_2^2] = \sum_{n_1=0}^{\infty}\sum_{n_2=0}^{\infty} n_2^2 P_{n_1,n_2}(t)$, we proceed as follows: we multiply (4.3.1.a,b) by n_2^2 and sum over n_1 and n_2:

$$\begin{aligned}
&\sum_{n_1=1}^{\infty}\sum_{n_2=1}^{\infty} n_2^2 \frac{dP_{n_1,n_2}(t)}{dt} = \\
&\sum_{n_1=1}^{\infty}\sum_{n_2=1}^{\infty} n_2^2 \lambda_1 n_2 P_{n_1-1,n_2}(t) + \sum_{n_1=1}^{\infty}\sum_{n_2=1}^{\infty} n_2^2 \lambda_{r1}(n_1+1) P_{n_1+1,n_2}(t) + \\
&\sum_{n_1=1}^{\infty}\sum_{n_2=1}^{\infty} n_2^2 \lambda_{r2}(n_2+1) P_{n_1,n_2+1}(t) + \sum_{n_1=1}^{\infty}\sum_{n_2=1}^{\infty} n_2^2 \lambda_2(n_1+1) P_{n_1+1,n_2-1}(t) + \\
&\sum_{n_1=1}^{\infty}\sum_{n_2=1}^{\infty} n_2^2 \lambda_a(n_2+1) P_{n_1-1,n_2+1}(t) - \\
&\sum_{n_1=1}^{\infty}\sum_{n_2=1}^{\infty} n_2^2 (\lambda_1 n_2 + \lambda_{r1} n_1 + \lambda_{r2} n_2 + \lambda_2 n_1 + \lambda_a n_2) P_{n_1,n_2}(t)
\end{aligned} \tag{B.6.13}$$

$$\begin{aligned}
\sum_{n_2=1}^{\infty} n_2^2 \frac{dP_{0,n_2}(t)}{dt} &= \sum_{n_2=1}^{\infty} n_2^2 \lambda_{r1} P_{1,n_2}(t) + \sum_{n_2=1}^{\infty} n_2^2 \lambda_{r2}(n_2+1) P_{0,n_2+1}(t) \\
&+ \sum_{n_2=1}^{\infty} n_2^2 \lambda_2 P_{1,n_2-1}(t) - \sum_{n_2=1}^{\infty} n_2^2 (\lambda_1 n_2 + \lambda_{r2} n_2 + \lambda_a n_2) P_{0,n_2}(t)
\end{aligned} \tag{B.6.14}$$

Adding the previous two equations together, after some manipulations (see Sub-section B.6.3), we have:

$$\begin{aligned}
\frac{dE_t[n_2^2]}{dt} &= \lambda_1 E_t[n_2^3] + \lambda_{r1} E_t[n_1 n_2^2] \\
&+ \lambda_{r2}\left(E_t[n_2^3] - 2E_t[n_2^2] + E_2(t)\right) + \lambda_2\left(E_t[n_1 n_2^2] + 2E_t[n_1 n_2] + E_1(t)\right) \\
&+ \lambda_a\left(E_t[n_2^3] - 2E_t[n_2^2] + E_2(t)\right) \\
&- (\lambda_{r1} + \lambda_2) E_t[n_1 n_2^2] - (\lambda_1 + \lambda_{r2} + \lambda_a) E_t[n_2^3]
\end{aligned} \tag{B.6.15}$$

The preceding equation simplifies to the following:

$$\frac{dE_t[n_2^2]}{dt} = 2\lambda_2 E_t[n_1 n_2] - 2(\lambda_{r2}+\lambda_a)E_t[n_2^2] + \lambda_2 E_1(t) + (\lambda_{r2}+\lambda_a)E_2(t) \tag{B.6.16}$$

Taking (B.6.9), (B.6.12) and (B.6.16) to Laplace domain, we can arrive at the solutions as done in Section B.4.

B.6.3. Derivation of 2nd Moment DEs

B.6.3.1. Derivation of $\frac{dE_t[n_1^2]}{dt}$

Here, we provide the detailed derivation of the expression for $\frac{dE_t[n_1^2]}{dt}$ (B.6.8) from (B.6.6) and (B.6.7). Here is the result of addition of (B.6.6) and (B.6.7):

$$\sum_{n_1=1}^{\infty}\sum_{n_2=0}^{\infty} n_1^2 \frac{dP_{n_1,n_2}(t)}{dt} =$$

$$\sum_{n_1=1}^{\infty}\sum_{n_2=1}^{\infty} n_1^2 \lambda_1 n_2 P_{n_1-1,n_2}(t) \tag{B.6.17}$$

$$+ \sum_{n_1=1}^{\infty}\sum_{n_2=0}^{\infty} n_1^2 \lambda_{r1}(n_1+1) P_{n_1+1,n_2}(t) \tag{B.6.18}$$

$$+ \sum_{n_1=1}^{\infty}\sum_{n_2=0}^{\infty} n_1^2 \lambda_{r2}(n_2+1) P_{n_1,n_2+1}(t) \tag{B.6.19}$$

$$+ \sum_{n_1=1}^{\infty}\sum_{n_2=1}^{\infty} n_1^2 \lambda_2(n_1+1) P_{n_1+1,n_2-1}(t) \tag{B.6.20}$$

$$+ \sum_{n_1=1}^{\infty}\sum_{n_2=0}^{\infty} n_1^2 \lambda_a(n_2+1) P_{n_1-1,n_2+1}(t) \tag{B.6.21}$$

$$- \sum_{n_1=1}^{\infty}\sum_{n_2=0}^{\infty} n_1^2 (\lambda_1 n_2 + \lambda_{r1} n_1 + \lambda_{r2} n_2 + \lambda_2 n_1 + \lambda_a n_2) P_{n_1,n_2}(t) \tag{B.6.22}$$

We write (B.6.17) as follows:

$$\begin{aligned} \sum_{n_1=1}^{\infty}\sum_{n_2=1}^{\infty} n_1^2 \lambda_1 n_2 P_{n_1-1,n_2}(t) &= \sum_{n_1=0}^{\infty}\sum_{n_2=1}^{\infty} \lambda_1 (n_1+1)^2 n_2 P_{n_1,n_2}(t) \\ &= \lambda_1 E_t[n_1^2 n_2] + 2\lambda_1 E_t[n_1 n_2] + \lambda_1 E_2(t) \end{aligned} \tag{B.6.23}$$

And (B.6.18) as follows:

$$\begin{aligned}\sum_{n_1=1}^{\infty}\sum_{n_2=0}^{\infty} n_1^2\lambda_{r1}(n_1+1)P_{n_1+1,n_2}(t) &= \sum_{n_1=2}^{\infty}\sum_{n_2=0}^{\infty}\lambda_{r1}(n_1-1)^2 n_1 P_{n_1,n_2}(t)\\ &= \lambda_{r1}\sum_{n_1=0}^{\infty}\sum_{n_2=0}^{\infty} n_1(n_1-1)^2 P_{n_1,n_2}(t)\\ &= \lambda_{r1}E_t[n_1^3]-2\lambda_{r1}E_t[n_1^2]+\lambda_{r1}E_1(t) \quad \text{(B.6.24)}\end{aligned}$$

And (B.6.19) as follows:

$$\sum_{n_1=1}^{\infty}\sum_{n_2=0}^{\infty} n_1^2\lambda_{r2}(n_2+1)P_{n_1,n_2+1}(t) = \sum_{n_1=1}^{\infty}\sum_{n_2=1}^{\infty}\lambda_{r2}n_1^2 n_2 P_{n_1,n_2}(t) = \lambda_{r2}E_t[n_1^2 n_2] \quad \text{(B.6.25)}$$

And (B.6.20) as follows:

$$\begin{aligned}\sum_{n_1=1}^{\infty}\sum_{n_2=1}^{\infty} n_1^2\lambda_2(n_1+1)P_{n_1+1,n_2-1}(t) &= \sum_{n_1=2}^{\infty}\sum_{n_2=0}^{\infty}(n_1-1)^2\lambda_2 n_1 P_{n_1,n_2}(t)\\ &= \lambda_2\sum_{n_1=0}^{\infty}\sum_{n_2=0}^{\infty}(n_1-1)^2 n_1 P_{n_1,n_2}(t)\\ &= \lambda_2 E_t[n_1^3]-2\lambda_2 E_t[n_1^2]+\lambda_2 E_1(t) \quad \text{(B.6.26)}\end{aligned}$$

And (B.6.21) as follows:

$$\begin{aligned}&\sum_{n_1=1}^{\infty}\sum_{n_2=0}^{\infty} n_1^2\lambda_a(n_2+1)P_{n_1-1,n_2+1}(t)\\ =\ &\sum_{n_1=0}^{\infty}\sum_{n_2=1}^{\infty}(n_1+1)^2\lambda_a n_2 P_{n_1,n_2}(t)\\ =\ &\lambda_a E_t[n_1^2 n_2]+2\lambda_a E_t[n_1 n_2]+\lambda_a E_2(t) \quad \text{(B.6.27)}\end{aligned}$$

Finally, (B.6.22) as follows:

$$\begin{aligned}&-\sum_{n_1=1}^{\infty}\sum_{n_2=0}^{\infty} n_1^2(\lambda_1 n_2+\lambda_{r1}n_1+\lambda_{r2}n_2+\lambda_2 n_1+\lambda_a n_2)P_{n_1,n_2}(t)\\ =\ &-(\lambda_1+\lambda_{r2}+\lambda_a)E_t[n_1^2 n_2]-(\lambda_{r1}+\lambda_2)E_t[n_1^3] \quad \text{(B.6.28)}\end{aligned}$$

Replacing (B.6.17) through (B.6.22) with the ones derived in (B.6.23) through (B.6.28), we arrive at (B.6.8).

B.6.3.2. Derivation of $\frac{dE_t[n_1 n_2]}{dt}$

In this section, we provide the detailed derivation of the expression for $\frac{dE_t[n_1 n_2]}{dt}$ (B.6.11) from (B.6.10). Here is (B.6.10) again:

$$\sum_{n_1=1}^{\infty}\sum_{n_2=1}^{\infty} n_1 n_2 \frac{dP_{n_1,n_2}(t)}{dt} =$$

$$\sum_{n_1=1}^{\infty}\sum_{n_2=1}^{\infty} n_1 n_2 \lambda_1 n_2 P_{n_1-1,n_2}(t) \tag{B.6.29}$$

$$+\sum_{n_1=1}^{\infty}\sum_{n_2=1}^{\infty} n_1 n_2 \lambda_{r1}(n_1+1) P_{n_1+1,n_2}(t) \tag{B.6.30}$$

$$+\sum_{n_1=1}^{\infty}\sum_{n_2=1}^{\infty} n_1 n_2 \lambda_{r2}(n_2+1) P_{n_1,n_2+1}(t) \tag{B.6.31}$$

$$+\sum_{n_1=1}^{\infty}\sum_{n_2=1}^{\infty} n_1 n_2 \lambda_2(n_1+1) P_{n_1+1,n_2-1}(t) \tag{B.6.32}$$

$$+\sum_{n_1=1}^{\infty}\sum_{n_2=1}^{\infty} n_1 n_2 \lambda_a(n_2+1) P_{n_1-1,n_2+1}(t) \tag{B.6.33}$$

$$-\sum_{n_1=1}^{\infty}\sum_{n_2=1}^{\infty} n_1 n_2 (\lambda_1 n_2 + \lambda_{r1} n_1 + \lambda_{r2} n_2 + \lambda_2 n_1 + \lambda_a n_2) P_{n_1,n_2}(t) \tag{B.6.34}$$

We write (B.6.29) as follows:

$$\begin{aligned}\sum_{n_1=1}^{\infty}\sum_{n_2=1}^{\infty} n_1 n_2 \lambda_1 n_2 P_{n_1-1,n_2}(t) &= \sum_{n_1=0}^{\infty}\sum_{n_2=1}^{\infty} \lambda_1 (n_1+1) n_2^2 P_{n_1,n_2}(t) \\ &= \lambda_1 E_t[n_1 n_2^2] + \lambda_1 E_t[n_2^2]\end{aligned} \tag{B.6.35}$$

And (B.6.30) as follows:

$$\begin{aligned}\sum_{n_1=1}^{\infty}\sum_{n_2=1}^{\infty} n_1 n_2 \lambda_{r1}(n_1+1) P_{n_1+1,n_2}(t) &= \sum_{n_1=2}^{\infty}\sum_{n_2=1}^{\infty} \lambda_{r1}(n_1-1) n_1 n_2 P_{n_1,n_2}(t) \\ = \sum_{n_1=0}^{\infty}\sum_{n_2=0}^{\infty} \lambda_{r1}(n_1^2 n_2 - n_1 n_2) P_{n_1,n_2}(t) &= \lambda_{r1} E_t[n_1^2 n_2] - \lambda_{r1} E_t[n_1 n_2]\end{aligned} \tag{B.6.36}$$

And (B.6.31) as follows:

$$\begin{aligned}\sum_{n_1=1}^{\infty}\sum_{n_2=1}^{\infty} n_1 n_2 \lambda_{r2}(n_2+1)P_{n_1,n_2+1}(t) &= \sum_{n_1=1}^{\infty}\sum_{n_2=2}^{\infty} \lambda_{r2} n_1 (n_2-1) n_2 P_{n_1,n_2}(t) \\ &= \lambda_{r2}\sum_{n_1=0}^{\infty}\sum_{n_2=0}^{\infty}(n_1 n_2^2 - n_1 n_2)P_{n_1,n_2}(t) \\ &= \lambda_{r2}E_t[n_1 n_2^2] - \lambda_{r2}E_t[n_1 n_2] \end{aligned} \tag{B.6.37}$$

And (B.6.32) as follows:

$$\begin{aligned}&\sum_{n_1=1}^{\infty}\sum_{n_2=1}^{\infty} n_1 n_2 \lambda_2 (n_1+1)P_{n_1+1,n_2-1}(t) \\ =\ &\sum_{n_1=2}^{\infty}\sum_{n_2=0}^{\infty}(n_1-1)(n_2+1)\lambda_2 n_1 P_{n_1,n_2}(t) \\ =\ &\lambda_2\sum_{n_1=0}^{\infty}\sum_{n_2=0}^{\infty}(n_1^2 n_2 + n_1^2 - n_1 n_2 - n_1)P_{n_1,n_2}(t) \\ =\ &\lambda_2\left(E_t[n_1^2 n_2] + E_t[n_1^2] - E_t[n_1 n_2] - E_1(t)\right) \end{aligned} \tag{B.6.38}$$

And (B.6.33) as follows:

$$\begin{aligned}&\sum_{n_1=1}^{\infty}\sum_{n_2=1}^{\infty} n_1 n_2 \lambda_a (n_2+1)P_{n_1-1,n_2+1}(t) \\ =\ &\sum_{n_1=0}^{\infty}\sum_{n_2=2}^{\infty}(n_1+1)(n_2-1)\lambda_a n_2 P_{n_1,n_2}(t) \\ =\ &\lambda_a\sum_{n_1=0}^{\infty}\sum_{n_2=0}^{\infty}(n_1 n_2^2 - n_1 n_2 + n_2^2 - n_2)P_{n_1,n_2}(t) \\ =\ &\lambda_a\left(E_t[n_1 n_2^2] - E_t[n_1 n_2] + E_t[n_2^2] - E_2(t)\right) \end{aligned} \tag{B.6.39}$$

Finally, (B.6.34) as follows:

$$\begin{aligned}&-\sum_{n_1=1}^{\infty}\sum_{n_2=1}^{\infty} n_1 n_2(\lambda_1 n_2 + \lambda_{r1} n_1 + \lambda_{r2} n_2 + \lambda_2 n_1 + \lambda_a n_2)P_{n_1,n_2}(t) \\ =\ &-(\lambda_{r1}+\lambda_2)E_t[n_1^2 n_2] - (\lambda_1 + \lambda_{r2} + \lambda_a)E_t[n_1 n_2^2] \end{aligned} \tag{B.6.40}$$

Replacing (B.6.29) through (B.6.34) with the ones derived in (B.6.35) through (B.6.40), we arrive at (B.6.11).

B.6.3.3. Derivation of $\frac{dE_t[n_2^2]}{dt}$

Here, we provide the detailed derivation of the expression for $\frac{dE_t[n_2^2]}{dt}$ (B.6.15) from (B.6.13) and (B.6.14). Here is the result of addition of (B.6.13) and (B.6.14):

$$\sum_{n_1=0}^{\infty}\sum_{n_2=1}^{\infty} n_2^2 \frac{dP_{n_1,n_2}(t)}{dt} =$$

$$\sum_{n_1=1}^{\infty}\sum_{n_2=1}^{\infty} n_2^2 \lambda_1 n_2 P_{n_1-1,n_2}(t) \tag{B.6.41}$$

$$+ \sum_{n_1=0}^{\infty}\sum_{n_2=1}^{\infty} n_2^2 \lambda_{r1}(n_1+1) P_{n_1+1,n_2}(t) \tag{B.6.42}$$

$$+ \sum_{n_1=0}^{\infty}\sum_{n_2=1}^{\infty} n_2^2 \lambda_{r2}(n_2+1) P_{n_1,n_2+1}(t) \tag{B.6.43}$$

$$+ \sum_{n_1=0}^{\infty}\sum_{n_2=1}^{\infty} n_2^2 \lambda_2(n_1+1) P_{n_1+1,n_2-1}(t) \tag{B.6.44}$$

$$+ \sum_{n_1=1}^{\infty}\sum_{n_2=1}^{\infty} n_2^2 \lambda_a(n_2+1) P_{n_1-1,n_2+1}(t) \tag{B.6.45}$$

$$- \sum_{n_1=0}^{\infty}\sum_{n_2=1}^{\infty} n_2^2 (\lambda_1 n_2 + \lambda_{r1} n_1 + \lambda_{r2} n_2 + \lambda_2 n_1 + \lambda_a n_2) P_{n_1,n_2}(t) \tag{B.6.46}$$

We write (B.6.41) as follows:

$$\sum_{n_1=1}^{\infty}\sum_{n_2=1}^{\infty} n_2^2 \lambda_1 n_2 P_{n_1-1,n_2}(t) = \sum_{n_1=0}^{\infty}\sum_{n_2=0}^{\infty} \lambda_1 n_2^3 P_{n_1,n_2}(t) = \lambda_1 E_t[n_2^3] \tag{B.6.47}$$

And (B.6.42) as follows:

$$\sum_{n_1=0}^{\infty}\sum_{n_2=1}^{\infty} n_2^2 \lambda_{r1}(n_1+1) P_{n_1+1,n_2}(t) = \sum_{n_1=1}^{\infty}\sum_{n_2=1}^{\infty} \lambda_{r1} n_2^2 n_1 P_{n_1,n_2}(t) = \lambda_{r1} E_t[n_1 n_2^2] \tag{B.6.48}$$

And (B.6.43) as follows:

$$\begin{aligned}\sum_{n_1=0}^{\infty}\sum_{n_2=1}^{\infty} n_2^2\lambda_{r2}(n_2+1)P_{n_1,n_2+1}(t) &= \sum_{n_1=0}^{\infty}\sum_{n_2=2}^{\infty}\lambda_{r2}(n_2-1)^2 n_2 P_{n_1,n_2}(t)\\ &= \lambda_{r2}\sum_{n_1=0}^{\infty}\sum_{n_2=0}^{\infty}(n_2^3-2n_2^2+n_2)P_{n_1,n_2}(t)\\ &= \lambda_{r2}\left(E_t[n_2^3]-2E_t[n_2^2]+E_2(t)\right) \qquad \text{(B.6.49)}\end{aligned}$$

And (B.6.44) as follows:

$$\begin{aligned}\sum_{n_1=0}^{\infty}\sum_{n_2=1}^{\infty} n_2^2\lambda_{2}(n_1+1)P_{n_1+1,n_2-1}(t) &= \sum_{n_1=1}^{\infty}\sum_{n_2=0}^{\infty}\lambda_{2}(n_2+1)^2 n_1 P_{n_1,n_2}(t)\\ &= \lambda_2\left(E_t[n_1n_2^2]+2E_t[n_1n_2]+E_1(t)\right) \qquad \text{(B.6.50)}\end{aligned}$$

And (B.6.45) as follows:

$$\begin{aligned}\sum_{n_1=1}^{\infty}\sum_{n_2=1}^{\infty} n_2^2\lambda_{a}(n_2+1)P_{n_1-1,n_2+1}(t) &= \sum_{n_1=0}^{\infty}\sum_{n_2=2}^{\infty}\lambda_{a}(n_2-1)^2 n_2 P_{n_1,n_2}(t)\\ &= \lambda_a\sum_{n_1=0}^{\infty}\sum_{n_2=0}^{\infty}(n_2^3-2n_2^2+n_2)P_{n_1,n_2}(t)\\ &= \lambda_a\left(E_t[n_2^3]-2E_t[n_2^2]+E_2(t)\right) \qquad \text{(B.6.51)}\end{aligned}$$

Finally, (B.6.46) as follows:

$$\begin{aligned}&-\sum_{n_1=0}^{\infty}\sum_{n_2=1}^{\infty} n_2^2(\lambda_1 n_2+\lambda_{r1}n_1+\lambda_{r2}n_2+\lambda_2 n_1+\lambda_a n_2)P_{n_1,n_2}(t)\\ =\ &-(\lambda_{r1}+\lambda_2)E_t[n_1n_2^2]-(\lambda_1+\lambda_{r2}+\lambda_a)E_t[n_2^3] \qquad \text{(B.6.52)}\end{aligned}$$

Replacing (B.6.41) through (B.6.46) with the ones derived in (B.6.47) through (B.6.52), we arrive at (B.6.15).

B.7. Calculation of Basic Reproduction Number

B.7.1. "Next Generation Matrix" Method: Introduction

In this section, we introduce the "Next Generation Matrix" method to calculate the Basic Reproduction Number (R_0). The introduction is based on the information provided in [52, pp.160-165], albeit some aspects including the terminology and variable names have been adapted to the current document.

Let f_i denote the rate at which secondary infections increase the number of nodes in the ith stage. On the other hand, v_i would denote the rate at which the number of nodes in the ith stage decreases. We then define F and V (two square matrices) with the following entries:

$$F_{ij} = \left.\frac{\partial f_i}{\partial E_j(t)}\right|_{t=0} \qquad V_{ij} = \left.\frac{\partial v_i}{\partial E_j(t)}\right|_{t=0} \tag{B.7.1}$$

We can then write the following equation:

$$\begin{bmatrix} \frac{dE_1(t)}{dt} \\ \frac{dE_2(t)}{dt} \end{bmatrix} = (F-V) \times \begin{bmatrix} E_1(t) \\ E_2(t) \end{bmatrix} \tag{B.7.2}$$

The mean time a hypothetical node spends in a stage can be expressed by the integral $\int_0^\infty E^*(t)dt$, where $E^*(t)$ is the solution to (B.7.2) with $F = 0$; the solution to this equation would be: $E(t)^* = e^{-Vt}$. Therefore, we have: $\int_0^\infty e^{-Vt}dt = V^{-1}$. The (i, j) entry of V^{-1} can be interpreted as the mean time a node, which is intially in stage j, will spend in stage i. On the other hand, the (i, j) entry of F is the rate of arrival of nodes to stage i caused by a node in stage j. Therefore, the mean of number of secondary infections caused by a hypothetical infective node is given by $\int_0^\infty Fe^{-Vt}dt = FV^{-1}$. The matrix $K = FV^{-1}$ is referred to as the "Next Generation Matrix" for the system. The (i, j) entry of K is the mean number of nodes in stage i produced by nodes which were intially in stage j. The largest eigenvalue of the

matrix K is defined as R_0, the Basic Reproduction Number. The associated eigenvector can be interpreted as the distribution of nodes in the stages that produces the greatest number (i.e., R_0) of secondary infections per generation.

B.7.2. R_0 Calculation

Based on the steps of the "Next Generation Matrix" method, we proceed as follows: From SIC model's differential equations for means (i.e., (4.3.4) on page 57), we extract the f and v matrices:

$$f = \begin{bmatrix} (\lambda_1 + \lambda_a)E_2(t) \\ \lambda_2 E_1(t) \end{bmatrix} \qquad v = \begin{bmatrix} (\lambda_2 + \lambda_{r1})E_1(t) \\ (\lambda_{r2} + \lambda_a)E_2(t) \end{bmatrix} \tag{B.7.3}$$

F and V matrices would be therefore as follows:

$$F = \begin{bmatrix} 0 & \lambda_1 + \lambda_a \\ \lambda_2 & 0 \end{bmatrix} \qquad V = \begin{bmatrix} \lambda_2 + \lambda_{r1} & 0 \\ 0 & \lambda_{r2} + \lambda_a \end{bmatrix} \tag{B.7.4}$$

The next generation matrix (K) would be as follows:

$$\begin{aligned} K &= F \times V^{-1} \\ &= \begin{bmatrix} 0 & \lambda_1 + \lambda_a \\ \lambda_2 & 0 \end{bmatrix} \times \frac{1}{(\lambda_2 + \lambda_{r1})(\lambda_{r2} + \lambda_a)} \begin{bmatrix} \lambda_{r2} + \lambda_a & 0 \\ 0 & \lambda_2 + \lambda_{r1} \end{bmatrix} \\ &= \frac{1}{(\lambda_2 + \lambda_{r1})(\lambda_{r2} + \lambda_a)} \begin{bmatrix} 0 & (\lambda_1 + \lambda_a)(\lambda_2 + \lambda_{r1}) \\ \lambda_2(\lambda_{r2} + \lambda_a) & 0 \end{bmatrix} \\ K &= \begin{bmatrix} 0 & \frac{\lambda_1 + \lambda_a}{\lambda_{r2} + \lambda_a} \\ \frac{\lambda_2}{\lambda_2 + \lambda_{r1}} & 0 \end{bmatrix} \end{aligned} \tag{B.7.5}$$

To derive R_0, we proceed as follows:

$$\det(K - R_0 \times I) = 0 \tag{B.7.6}$$

where I is an identity matrix. We therefore have:

$$\begin{aligned} \det \begin{bmatrix} -R_0 & \frac{\lambda_1+\lambda_a}{\lambda_{r2}+\lambda_a} \\ \frac{\lambda_2}{\lambda_2+\lambda_{r1}} & -R_0 \end{bmatrix} &= 0 \\ R_0^2 - \frac{\lambda_1+\lambda_a}{\lambda_{r2}+\lambda_a} \times \frac{\lambda_2}{\lambda_2+\lambda_{r1}} &= 0 \end{aligned}$$

Basic Reproduction Number (R_0) is therefore derived as noted in (4.3.13) on page 59.

C. The SIC-P2P Model

C.1. Deriving a PDE from the Differential-Difference Equations

In this appendix, we provide the detailed derivation of the PDE of the PGF in (5.3.2) on page 78 from the differential-difference equations in (5.3.1). After summing over n_1 and n_2 and adding together (5.3.1.a), (5.3.1.b), (5.3.1.c) and (5.3.1.d), we have:

$$\frac{\partial P(z_1,z_2,t)}{\partial t} =$$

$$\sum_{n_1=1}^{\infty}\sum_{n_2=1}^{\infty}\lambda_1 n_2 P_{n_1-1,n_2}(t) z_1^{n_1} z_2^{n_2} \tag{C.1.1}$$

$$+\sum_{n_1=0}^{\infty}\sum_{n_2=0}^{\infty}\lambda_{r1}(n_1+1) P_{n_1+1,n_2}(t) z_1^{n_1} z_2^{n_2} \tag{C.1.2}$$

$$+\sum_{n_1=0}^{\infty}\sum_{n_2=0}^{\infty}\lambda_{r2}(n_2+1) P_{n_1,n_2+1}(t) z_1^{n_1} z_2^{n_2} \tag{C.1.3}$$

$$+\sum_{n_1=0}^{\infty}\sum_{n_2=1}^{\infty}\lambda_2(n_1+1) P_{n_1+1,n_2-1}(t) z_1^{n_1} z_2^{n_2} \tag{C.1.4}$$

$$+\sum_{n_1=1}^{\infty}\sum_{n_2=0}^{\infty}[\lambda_{a1}+\lambda_{a2}(n_2+1)] P_{n_1-1,n_2+1}(t) z_1^{n_1} z_2^{n_2} \tag{C.1.5}$$

$$-\sum_{n_1=0}^{\infty}\sum_{n_2=0}^{\infty}(\lambda_1 n_2+\lambda_{r1} n_1+\lambda_{r2} n_2+\lambda_2 n_1+\lambda_{a2} n_2) P_{n_1,n_2}(t) z_1^{n_1} z_2^{n_2} \tag{C.1.6}$$

$$-\sum_{n_1=0}^{\infty}\sum_{n_2=1}^{\infty}\lambda_{a1} P_{n_1,n_2}(t) z_1^{n_1} z_2^{n_2} \tag{C.1.7}$$

We write (C.1.1) as follows:

$$\begin{aligned}
&\sum_{n_1=1}^{\infty}\sum_{n_2=1}^{\infty}\lambda_1 n_2 P_{n_1-1,n_2}(t) z_1^{n_1} z_2^{n_2} \qquad \text{(C.1.8)}\\
&= \lambda_1 z_1 \sum_{n_1=1}^{\infty}\sum_{n_2=1}^{\infty} n_2 P_{n_1-1,n_2}(t) z_1^{n_1-1} z_2^{n_2}\\
&= \lambda_1 z_1 z_2 \sum_{n_1=0}^{\infty}\sum_{n_2=0}^{\infty} n_2 P_{n_1,n_2}(t) z_1^{n_1} z_2^{n_2-1} = \lambda_1 z_1 z_2 \frac{\partial P(z_1,z_2,t)}{\partial z_2}
\end{aligned}$$

And (C.1.2) as follows:

$$\begin{aligned}
&\sum_{n_1=0}^{\infty}\sum_{n_2=0}^{\infty}\lambda_{r1}(n_1+1) P_{n_1+1,n_2}(t) z_1^{n_1} z_2^{n_2} \qquad \text{(C.1.9)}\\
&= \sum_{n_1=0}^{\infty}\sum_{n_2=0}^{\infty}\lambda_{r1}\frac{n_1+1}{z_1} P_{n_1+1,n_2}(t) z_1^{n_1+1} z_2^{n_2}\\
&= \sum_{n_1=1}^{\infty}\sum_{n_2=0}^{\infty}\lambda_{r1} n_1 P_{n_1,n_2}(t) z_1^{n_1-1} z_2^{n_2} = \lambda_{r1}\frac{\partial P(z_1,z_2,t)}{\partial z_1}
\end{aligned}$$

And (C.1.3) as follows:

$$\begin{aligned}
&\sum_{n_1=0}^{\infty}\sum_{n_2=0}^{\infty}\lambda_{r2}(n_2+1) P_{n_1,n_2+1}(t) z_1^{n_1} z_2^{n_2} \qquad \text{(C.1.10)}\\
&= \sum_{n_1=0}^{\infty}\sum_{n_2=0}^{\infty}\lambda_{r2}\frac{n_2+1}{z_2} P_{n_1,n_2+1}(t) z_1^{n_1} z_2^{n_2+1}\\
&= \sum_{n_1=0}^{\infty}\sum_{n_2=1}^{\infty}\lambda_{r2} n_2 P_{n_1,n_2}(t) z_1^{n_1} z_2^{n_2-1} = \lambda_{r2}\frac{\partial P(z_1,z_2,t)}{\partial z_2}
\end{aligned}$$

And (C.1.4) as follows:

$$\begin{aligned}
&\sum_{n_1=0}^{\infty}\sum_{n_2=1}^{\infty}\lambda_2(n_1+1) P_{n_1+1,n_2-1}(t) z_1^{n_1} z_2^{n_2} \qquad \text{(C.1.11)}\\
&= \sum_{n_1=0}^{\infty}\sum_{n_2=1}^{\infty}\lambda_2(n_1+1)\frac{z_2}{z_1} P_{n_1+1,n_2-1}(t) z_1^{n_1+1} z_2^{n_2-1} = \sum_{n_1=1}^{\infty}\sum_{n_2=0}^{\infty}\lambda_2 n_1\frac{z_2}{z_1} P_{n_1,n_2}(t) z_1^{n_1} z_2^{n_2}\\
&= \lambda_2 z_2 \sum_{n_1=0}^{\infty}\sum_{n_2=0}^{\infty} n_1 P_{n_1,n_2}(t) z_1^{n_1-1} z_2^{n_2} = \lambda_2 z_2 \frac{\partial P(z_1,z_2,t)}{\partial z_1}
\end{aligned}$$

And (C.1.5) as follows:

$$\begin{aligned}
&\sum_{n_1=1}^{\infty}\sum_{n_2=0}^{\infty}[\lambda_{a1}+\lambda_{a2}(n_2+1)]P_{n_1-1,n_2+1}(t)z_1^{n_1}z_2^{n_2} \qquad \text{(C.1.12)}\\
&=\sum_{n_1=1}^{\infty}\sum_{n_2=0}^{\infty}[\lambda_{a1}+\lambda_{a2}(n_2+1)]\frac{z_1}{z_2}P_{n_1-1,n_2+1}(t)z_1^{n_1-1}z_2^{n_2+1}\\
&=\sum_{n_1=0}^{\infty}\sum_{n_2=1}^{\infty}(\lambda_{a1}+\lambda_{a2}n_2)\frac{z_1}{z_2}P_{n_1,n_2}(t)z_1^{n_1}z_2^{n_2}\\
&=\lambda_{a1}\frac{z_1}{z_2}\sum_{n_1=0}^{\infty}\sum_{n_2=1}^{\infty}P_{n_1,n_2}(t)z_1^{n_1}z_2^{n_2}+\lambda_{a2}z_1\sum_{n_1=0}^{\infty}\sum_{n_2=0}^{\infty}n_2P_{n_1,n_2}(t)z_1^{n_1}z_2^{n_2-1}\\
&=\lambda_{a1}\frac{z_1}{z_2}\left(P(z_1,z_2,t)-\sum_{n_1=0}^{\infty}P_{n_1,0}(t)z_1^{n_1}\right)+\lambda_{a2}z_1\frac{\partial P(z_1,z_2,t)}{\partial z_2}
\end{aligned}$$

And (C.1.6) as follows:

$$\begin{aligned}
&-\sum_{n_1=0}^{\infty}\sum_{n_2=0}^{\infty}(\lambda_1 n_2+\lambda_{r1}n_1+\lambda_{r2}n_2+\lambda_2 n_1+\lambda_{a2}n_2)P_{n_1,n_2}(t)z_1^{n_1}z_2^{n_2} \qquad \text{(C.1.13)}\\
&=-\sum_{n_1=0}^{\infty}\sum_{n_2=0}^{\infty}(\lambda_{r1}+\lambda_2)n_1P_{n_1,n_2}(t)z_1^{n_1}z_2^{n_2}-\sum_{n_1=0}^{\infty}\sum_{n_2=0}^{\infty}(\lambda_1+\lambda_{r2}+\lambda_{a2})n_2P_{n_1,n_2}(t)z_1^{n_1}z_2^{n_2}\\
&=-(\lambda_{r1}+\lambda_2)z_1\sum_{n_1=0}^{\infty}\sum_{n_2=0}^{\infty}n_1P_{n_1,n_2}(t)z_1^{n_1-1}z_2^{n_2}\\
&\quad-(\lambda_1+\lambda_{r2}+\lambda_{a2})z_2\sum_{n_1=0}^{\infty}\sum_{n_2=0}^{\infty}n_2P_{n_1,n_2}(t)z_1^{n_1}z_2^{n_2-1}\\
&=-(\lambda_{r1}+\lambda_2)z_1\frac{\partial P(z_1,z_2,t)}{\partial z_1}-(\lambda_1+\lambda_{r2}+\lambda_{a2})z_2\frac{\partial P(z_1,z_2,t)}{\partial z_2}
\end{aligned}$$

Finally, (C.1.7) as follows:

$$-\sum_{n_1=0}^{\infty}\sum_{n_2=1}^{\infty}\lambda_{a1}P_{n_1,n_2}(t)z_1^{n_1}z_2^{n_2}=-\lambda_{a1}\left(P(z_1,z_2,t)-\sum_{n_1=0}^{\infty}P_{n_1,0}(t)z_1^{n_1}\right) \qquad \text{(C.1.14)}$$

Replacing (C.1.1) through (C.1.7) with the terms derived in (C.1.8) through (C.1.14), after simplification, we arrive at (5.3.2).

C.2. Derivation of Means from the PDE of the PGF

We take the derivative of (5.3.3) on page 79 with respect to z_1 as follows:

$$\begin{aligned}(-\lambda_{r1}-\lambda_2)\frac{\partial P(z_1,z_2,t)}{\partial z_1}+(\lambda_{r1}+\lambda_2 z_2-\lambda_{r1}z_1-\lambda_2 z_1)\frac{\partial^2 P(z_1,z_2,t)}{\partial z_1^2}\\ +(\lambda_1 z_2+\lambda_{a2})\frac{\partial P(z_1,z_2,t)}{\partial z_2}\\ +(\lambda_1 z_1 z_2+\lambda_{r2}+\lambda_{a2}z_1-\lambda_1 z_2-\lambda_{r2}z_2-\lambda_{a2}z_2)\frac{\partial^2 P(z_1,z_2,t)}{\partial z_2 \partial z_1}-\frac{\partial^2 P(z_1,z_2,t)}{\partial t \partial z_1}\\ =\lambda_{a1}(-\frac{1}{z_2})P(z_1,z_2,t)+\lambda_{a1}(1-\frac{z_1}{z_2})\frac{\partial P(z_1,z_2,t)}{\partial z_1} \quad \text{(C.2.1)}\end{aligned}$$

Setting $z_1 = z_2 = 1$ in (C.2.1) gives us the following equation:

$$-(\lambda_2+\lambda_{r1})E_1(t)+(\lambda_1+\lambda_{a2})E_2(t)-\frac{dE_1(t)}{dt}=-\lambda_{a1} \quad \text{(C.2.2)}$$

We then take the derivative of (5.3.3) with respect to z_2:

$$\begin{aligned}\lambda_2\frac{\partial P(z_1,z_2,t)}{\partial z_1}+(\lambda_{r1}+\lambda_2 z_2-\lambda_{r1}z_1-\lambda_2 z_1)\frac{\partial^2 P(z_1,z_2,t)}{\partial z_2 \partial z_1}\\ +(\lambda_1 z_1-\lambda_1-\lambda_{r2}-\lambda_{a2})\frac{\partial P(z_1,z_2,t)}{\partial z_2}\\ +(\lambda_1 z_1 z_2+\lambda_{r2}+\lambda_{a2}z_1-\lambda_1 z_2-\lambda_{r2}z_2-\lambda_{a2}z_2)\frac{\partial^2 P(z_1,z_2,t)}{\partial z_2^2}-\frac{\partial^2 P(z_1,z_2,t)}{\partial t \partial z_2}\\ =\lambda_{a1}\frac{z_1}{z_2^2}P(z_1,z_2,t)+\lambda_{a1}(1-\frac{z_1}{z_2})\frac{\partial P(z_1,z_2,t)}{\partial z_2} \quad \text{(C.2.3)}\end{aligned}$$

Setting $z_1 = z_2 = 1$ in (C.2.3) gives us the following equation:

$$\lambda_2 E_1(t)-(\lambda_{r2}+\lambda_{a2})E_2(t)-\frac{dE_2(t)}{dt}=\lambda_{a1} \quad \text{(C.2.4)}$$

Re-arranging (C.2.2) and (C.2.4), we arrive at (5.3.5) on page 79.

Taking (5.3.5) to Laplace domain, we can write:

$$\begin{cases} sE_1(s \mid k_1,k_2) - k_1 = & -(\lambda_2+\lambda_{r1})E_1(s \mid k_1,k_2) \\ & +(\lambda_1+\lambda_{a2})E_2(s \mid k_1,k_2) + \frac{\lambda_{a1}}{s} \\ sE_2(s \mid k_1,k_2) - k_2 = & \lambda_2 E_1(s \mid k_1,k_2) \\ & -(\lambda_{r2}+\lambda_{a2})E_2(s \mid k_1,k_2) - \frac{\lambda_{a1}}{s} \end{cases} \tag{C.2.5}$$

where k_1 and k_2 are values of n_1 and n_2 at $t = 0$, respectively. Note that k_1 and k_2 are variables themselves and their means are obtained as follows:

$$\begin{cases} \bar{k_1} = \sum_{k_1=0}^{\infty}\sum_{k_2=0}^{\infty} k_1 P_{k_1,k_2}(t=0) \\ \bar{k_2} = \sum_{k_1=0}^{\infty}\sum_{k_2=0}^{\infty} k_2 P_{k_1,k_2}(t=0) \end{cases} \tag{C.2.6}$$

$\bar{k_1}$ and $\bar{k_2}$ are therefore the values of the means at $t = 0$. We then proceed to uncondition (C.2.5), i.e., we take $\sum_{k_1=0}^{\infty}\sum_{k_2=0}^{\infty}\{X\}P_{k_1,k_2}(t=0)$, with X being each element of the equation set. After simplification, we have:

$$\begin{cases} sE_1(s) - \bar{k_1} = -(\lambda_2+\lambda_{r1})E_1(s) + (\lambda_1+\lambda_{a2})E_2(s) + \frac{\lambda_{a1}}{s} \\ sE_2(s) - \bar{k_2} = \lambda_2 E_1(s) - (\lambda_{r2}+\lambda_{a2})E_2(s) - \frac{\lambda_{a1}}{s} \end{cases} \tag{C.2.7}$$

$E_1(s)$ and $E_2(s)$ are then obtained as follows:

$$\begin{cases} E_1(s) = \dfrac{\overline{k_1}s\left(\lambda_{a2}+\lambda_{r2}+s\right) + \overline{k_2}s\left(\lambda_1+\lambda_{a2}\right) + \lambda_{a1}\left(-\lambda_1+\lambda_{r2}+s\right)}{s\left(-\left(\lambda_1\lambda_2\right)+\lambda_2\lambda_{r2}+\lambda_{a2}\lambda_{r1}+\lambda_{r1}\lambda_{r2}+s^2+s\left(\lambda_2+\lambda_{a2}+\lambda_{r1}+\lambda_{r2}\right)\right)} \\ \\ E_2(s) = \dfrac{\overline{k_1}s\lambda_2 - \lambda_{a1}\left(s+\lambda_{r1}\right) + \overline{k_2}s\left(s+\lambda_2+\lambda_{r1}\right)}{s\left(s^2-\lambda_1\lambda_2+\lambda_{a2}\lambda_{r1}+\lambda_2\lambda_{r2}+\lambda_{r1}\lambda_{r2}+s\left(\lambda_2+\lambda_{a2}+\lambda_{r1}+\lambda_{r2}\right)\right)} \end{cases} \tag{C.2.8}$$

Finally, the inverse Laplace of $E_1(s)$ and $E_2(s)$ are obtained as shown in (5.3.6) and (5.3.7) on page 80.

C.3. Derivation of Variances from the PDE of the PGF

Let us define:

$$\begin{aligned}\psi_1(t) &\triangleq \frac{\partial^2 P(z_1,z_2,t)}{\partial z_1^2}\,|_{z_1=z_2=1}\\ \psi_2(t) &\triangleq \frac{\partial^2 P(z_1,z_2,t)}{\partial z_2^2}\,|_{z_1=z_2=1}\\ \psi_{12}(t) &\triangleq \frac{\partial^2 P(z_1,z_2,t)}{\partial z_1 \partial z_2}\,|_{z_1=z_2=1}\end{aligned} \tag{C.3.1}$$

The variances are then given by:

$$\sigma_1^2(t) = \psi_1(t) + E_1(t) - (E_1(t))^2\ ,\ \sigma_2^2(t) = \psi_2(t) + E_2(t) - (E_2(t))^2 \tag{C.3.2}$$

Considering that $E_1(t=0) = \bar{k_1}$ and $E_2(t=0) = \bar{k_2}$, the functions in (C.3.1) have the following initial values:

$$\psi_1(t=0) = \bar{k_1^2} - \bar{k_1},\ \psi_2(t=0) = \bar{k_2^2} - \bar{k_2},\ \psi_{12}(t=0) = \bar{k_1}\bar{k_2} \tag{C.3.3}$$

Taking the derivative of (C.2.1) with respect to z_1 (i.e., deriving the 2nd derivative of (5.3.3) on page 79 with respect to z_1), we have:

$$\begin{aligned}&(-\lambda_{r1}-\lambda_2)\frac{\partial^2 P(z_1,z_2,t)}{\partial z_1^2} + (-\lambda_{r1}-\lambda_2)\frac{\partial^2 P(z_1,z_2,t)}{\partial z_1^2}\\ &+(\lambda_{r1}+\lambda_2 z_2 - \lambda_{r1} z_1 - \lambda_2 z_1)\frac{\partial^3 P(z_1,z_2,t)}{\partial z_1^3}\\ &+(\lambda_1 z_2 + \lambda_{a2})\frac{\partial^2 P(z_1,z_2,t)}{\partial z_2 \partial z_1} + (\lambda_1 z_2 + \lambda_{a2})\frac{\partial^2 P(z_1,z_2,t)}{\partial z_2 \partial z_1}\\ &+(\lambda_1 z_1 z_2 + \lambda_{r2} + \lambda_{a2} z_1 - \lambda_1 z_2 - \lambda_{r2} z_2 - \lambda_{a2} z_2)\frac{\partial^3 P(z_1,z_2,t)}{\partial z_2 \partial z_1^2} - \frac{\partial^3 P(z_1,z_2,t)}{\partial t \partial z_1^2}\\ &= -\frac{\lambda_{a1}}{z_2}\frac{\partial P(z_1,z_2,t)}{\partial z_1} - \frac{\lambda_{a1}}{z_2}\frac{\partial P(z_1,z_2,t)}{\partial z_1} + \lambda_{a1}(1-\frac{z_1}{z_2})\frac{\partial^2 P(z_1,z_2,t)}{\partial z_1^2}\end{aligned} \tag{C.3.4}$$

Setting $z_1 = z_2 = 1$ in the preceding equation leads to the following equation:

$$2(\lambda_1 + \lambda_{a2})\psi_{12}(t) - 2(\lambda_{r1} + \lambda_2)\psi_1(t) - \frac{d\psi_1(t)}{dt} = -2\lambda_{a1}E_1(t) \tag{C.3.5}$$

Likewise, taking the derivative of (C.2.3) with respect to z_2 (i.e., deriving the 2nd derivative of (5.3.3) with respect to z_2), we have:

$$\begin{aligned}
&\lambda_2\frac{\partial^2 P(z_1,z_2,t)}{\partial z_1 \partial z_2} + \lambda_2\frac{\partial^2 P(z_1,z_2,t)}{\partial z_1 \partial z_2} + (\lambda_{r1} + \lambda_2 z_2 - \lambda_{r1}z_1 - \lambda_2 z_1)\frac{\partial^3 P(z_1,z_2,t)}{\partial z_2^2 \partial z_1} \\
&+(\lambda_1 z_1 - \lambda_1 - \lambda_{r2} - \lambda_{a2})\frac{\partial^2 P(z_1,z_2,t)}{\partial z_2^2} + (\lambda_1 z_1 - \lambda_1 - \lambda_{r2} - \lambda_{a2})\frac{\partial^2 P(z_1,z_2,t)}{\partial z_2^2} \\
&+(\lambda_1 z_1 z_2 + \lambda_{r2} + \lambda_{a2}z_1 - \lambda_1 z_2 - \lambda_{r2}z_2 - \lambda_{a2}z_2)\frac{\partial^3 P(z_1,z_2,t)}{\partial z_2^3} - \frac{\partial^3 P(z_1,z_2,t)}{\partial t \partial z_2^2} \\
&= -2\lambda_{a1}\frac{z_1}{z_2^3}P(z_1,z_2,t) + \lambda_{a1}\frac{z_1}{z_2^2}\frac{\partial P(z_1,z_2,t)}{\partial z_2} + \lambda_{a1}\frac{z_1}{z_2^2}\frac{\partial P(z_1,z_2,t)}{\partial z_2} + \lambda_{a1}(1-\tfrac{z_1}{z_2})\tfrac{\partial^2 P(z_1,z_2,t)}{\partial z_2^2}
\end{aligned} \tag{C.3.6}$$

Setting $z_1 = z_2 = 1$ in the preceding equation leads to the following equation:

$$2\lambda_2\psi_{12}(t) - 2(\lambda_{r2} + \lambda_{a2})\psi_2(t) - \frac{d\psi_2(t)}{dt} = -2\lambda_{a1} + 2\lambda_{a1}E_2(t) \tag{C.3.7}$$

Finally, taking the derivative of (C.2.1) with respect to z_2, we have:

$$\begin{aligned}
&(-\lambda_{r1} - \lambda_2)\frac{\partial^2 P(z_1,z_2,t)}{\partial z_1 \partial z_2} + \lambda_2\frac{\partial^2 P(z_1,z_2,t)}{\partial z_1^2} \\
&+(\lambda_{r1} + \lambda_2 z_2 - \lambda_{r1}z_1 - \lambda_2 z_1)\frac{\partial^3 P(z_1,z_2,t)}{\partial z_1^2 \partial z_2} + \lambda_1\frac{\partial P(z_1,z_2,t)}{\partial z_2} \\
&+(\lambda_1 z_2 + \lambda_{a2})\frac{\partial^2 P(z_1,z_2,t)}{\partial z_2^2} + (\lambda_1 z_1 - \lambda_1 - \lambda_{r2} - \lambda_{a2})\frac{\partial^2 P(z_1,z_2,t)}{\partial z_2 \partial z_1} \\
&+(\lambda_1 z_1 z_2 + \lambda_{r2} + \lambda_{a2}z_1 - \lambda_1 z_2 - \lambda_{r2}z_2 - \lambda_{a2}z_2)\frac{\partial^3 P(z_1,z_2,t)}{\partial z_2^2 \partial z_1} - \frac{\partial^3 P(z_1,z_2,t)}{\partial t \partial z_1 \partial z_2} \\
&= \frac{\lambda_{a1}}{z_2^2}P(z_1,z_2,t) - \frac{\lambda_{a1}}{z_2}\frac{\partial P(z_1,z_2,t)}{\partial z_2} + \lambda_{a1}\frac{z_1}{z_2^2}\frac{\partial P(z_1,z_2,t)}{\partial z_1} + \lambda_{a1}(1-\frac{z_1}{z_2})\tfrac{\partial^2 P(z_1,z_2,t)}{\partial z_1 \partial z_2}
\end{aligned} \tag{C.3.8}$$

Setting $z_1 = z_2 = 1$ in the preceding equation leads to the following equation:

$$\begin{aligned}&-(\lambda_{r1}+\lambda_2+\lambda_{r2}+\lambda_{a2})\psi_{12}(t)+\lambda_2\psi_1(t)+\lambda_1 E_2(t)+(\lambda_1+\lambda_{a2})\psi_2(t)-\frac{d\psi_{12}(t)}{dt}\\&=\lambda_{a1}-\lambda_{a1}E_2(t)+\lambda_{a1}E_1(t)\end{aligned} \tag{C.3.9}$$

Re-arranging (C.3.5), (C.3.7), and (C.3.9) (if written in terms of $\psi_1(t)$, $\psi_2(t)$, and $\psi_{12}(t)$) gives us (C.3.10):

$$\begin{cases}\frac{d\psi_1(t)}{dt} =2(\lambda_1+\lambda_{a2})\psi_{12}(t)-2(\lambda_{r1}+\lambda_2)\psi_1(t)+2\lambda_{a1}E_1(t)\\ \frac{d\psi_2(t)}{dt} =2\lambda_2\psi_{12}(t)-2(\lambda_{r2}+\lambda_{a2})\psi_2(t)+2\lambda_{a1}-2\lambda_{a1}E_2(t)\\ \frac{d\psi_{12}(t)}{dt} =-(\lambda_{r1}+\lambda_2+\lambda_{r2}+\lambda_{a2})\psi_{12}(t)+\lambda_2\psi_1(t)\\ \qquad\qquad +\lambda_1E_2(t)+(\lambda_1+\lambda_{a2})\psi_2(t)-\lambda_{a1}+\lambda_{a1}E_2(t)-\lambda_{a1}E_1(t)\end{cases} \tag{C.3.10}$$

We have three ODEs and three variables ($\psi_1(t)$, $\psi_{12}(t)$ and $\psi_2(t)$); therefore, we can find a unique solution by solving this system of linear ODEs. Taking the preceding equations to Laplace domain, we have:

$$\begin{cases}s\psi_1(s\mid k_1,k_2)-k_1^2+k_1 =2(\lambda_1+\lambda_{a2})\psi_{12}(s\mid k_1,k_2)\\ \qquad\qquad -2(\lambda_{r1}+\lambda_2)\psi_1(s\mid k_1,k_2)+2\lambda_{a1}E_1(s\mid k_1,k_2)\\ s\psi_2(s\mid k_1,k_2)-k_2^2+k_2 =2\lambda_2\psi_{12}(s\mid k_1,k_2)-2(\lambda_{r2}+\lambda_{a2})\psi_2(s\mid k_1,k_2)\\ \qquad\qquad +\frac{2\lambda_{a1}}{s}-2\lambda_{a1}E_2(s\mid k_1,k_2)\\ s\psi_{12}(s\mid k_1,k_2)-k_1k_2 =-(\lambda_{r1}+\lambda_2+\lambda_{r2}+\lambda_{a2})\psi_{12}(s\mid k_1,k_2)+\lambda_2\psi_1(s\mid k_1,k_2)\\ \qquad\qquad +\lambda_1E_2(s\mid k_1,k_2)+(\lambda_1+\lambda_{a2})\psi_2(s\mid k_1,k_2)\\ \qquad\qquad -\frac{\lambda_{a1}}{s}+\lambda_{a1}E_2(s\mid k_1,k_2)-\lambda_{a1}E_1(s\mid k_1,k_2)\end{cases} \tag{C.3.11}$$

Like before, we then proceed to uncondition (C.3.11). After simplification, we have:

$$\begin{cases} s\psi_1(s) - \bar{k_1^2} + \bar{k_1} = 2(\lambda_1 + \lambda_{a2})\psi_{12}(s) - 2(\lambda_{r1} + \lambda_2)\psi_1(s) + 2\lambda_{a1}E_1(s) \\ s\psi_2(s) - \bar{k_2^2} + \bar{k_2} = 2\lambda_2\psi_{12}(s) - 2(\lambda_{r2} + \lambda_{a2})\psi_2(s) + \dfrac{2\lambda_{a1}}{s} - 2\lambda_{a1}E_2(s) \\ s\psi_{12}(s) - k_1\bar{k}_2 = -(\lambda_{r1} + \lambda_2 + \lambda_{r2} + \lambda_{a2})\psi_{12}(s) + \lambda_2\psi_1(s) \\ \qquad\qquad + \lambda_1 E_2(s) + (\lambda_1 + \lambda_{a2})\psi_2(s) - \dfrac{\lambda_{a1}}{s} + \lambda_{a1}E_2(s) - \lambda_{a1}E_1(s) \end{cases} \tag{C.3.12}$$

The solution of (C.3.12) (i.e., the expressions for $\psi_1(s)$, $\psi_2(s)$, and $\psi_{12}(s)$) is provided in Sections C.4 through C.6. On the other hand, the expressions for $\psi_1(t)$, $\psi_2(t)$, and $\psi_{12}(t)$ (which are components of $\sigma_1^2(t)$ and $\sigma_2^2(t)$ according to (C.3.2)) are extremely lengthy; hence, they are provided in [72] instead.

C.4. SIC-P2P Model: $\psi_1(s)$ Formula

$$\psi_1(s) = -\frac{(-\lambda_1-\lambda_{a2})\left(\overline{k_2}-\overline{k_2^2}-\frac{2\lambda_{a1}}{s}+\frac{2\lambda_{a1}\left(s\overline{k_1}\lambda_2-\lambda_{a1}(s+\lambda_{r1})+s\overline{k_2}(s+\lambda_2+\lambda_{r1})\right)}{s\left(s^2+(\lambda_2+\lambda_{a2}+\lambda_{r1}+\lambda_{r2})s-\lambda_1\lambda_2+\lambda_{a2}\lambda_{r1}+\lambda_2\lambda_{r2}+\lambda_{r1}\lambda_{r2}\right)}\right)}{\lambda_2(s+2(\lambda_{a2}+\lambda_{r2}))} -$$
$$\frac{1}{\lambda_2}\left(\overline{k_1k_2}-\frac{\lambda_{a1}}{s}+\frac{\lambda_1\left(s\overline{k_1}\lambda_2-\lambda_{a1}(s+\lambda_{r1})+s\overline{k_2}(s+\lambda_2+\lambda_{r1})\right)}{s(s^2+(\lambda_2+\lambda_{a2}+\lambda_{r1}+\lambda_{r2})s-\lambda_1\lambda_2+\lambda_{a2}\lambda_{r1}+\lambda_2\lambda_{r2}+\lambda_{r1}\lambda_{r2})}+\right.$$
$$\frac{\lambda_{a1}\left(s\overline{k_1}\lambda_2-\lambda_{a1}(s+\lambda_{r1})+s\overline{k_2}(s+\lambda_2+\lambda_{r1})\right)}{s(s^2+(\lambda_2+\lambda_{a2}+\lambda_{r1}+\lambda_{r2})s-\lambda_1\lambda_2+\lambda_{a2}\lambda_{r1}+\lambda_2\lambda_{r2}+\lambda_{r1}\lambda_{r2})}-$$
$$\left.\frac{\lambda_{a1}\left(s\overline{k_2}(\lambda_1+\lambda_{a2})+\lambda_{a1}(s-\lambda_1+\lambda_{r2})+s\overline{k_1}(s+\lambda_{a2}+\lambda_{r2})\right)}{s(s^2+(\lambda_2+\lambda_{a2}+\lambda_{r1}+\lambda_{r2})s-\lambda_1\lambda_2+\lambda_{a2}\lambda_{r1}+\lambda_2\lambda_{r2}+\lambda_{r1}\lambda_{r2})}\right)-$$
$$\left(\left(\frac{2(\lambda_1+\lambda_{a2})}{s+2(\lambda_{a2}+\lambda_{r2})}-\frac{s+\lambda_2+\lambda_{a2}+\lambda_{r1}+\lambda_{r2}}{\lambda_2}\right)\left((-\lambda_1-\lambda_{a2})(s+2(\lambda_2+\lambda_{r1}))\right.\right.$$
$$\left(\overline{k_2}-\overline{k_2^2}-\frac{2\lambda_{a1}}{s}+\frac{2\lambda_{a1}\left(s\overline{k_1}\lambda_2-\lambda_{a1}(s+\lambda_{r1})+s\overline{k_2}(s+\lambda_2+\lambda_{r1})\right)}{s(s^2+(\lambda_2+\lambda_{a2}+\lambda_{r1}+\lambda_{r2})s-\lambda_1\lambda_2+\lambda_{a2}\lambda_{r1}+\lambda_2\lambda_{r2}+\lambda_{r1}\lambda_{r2})}\right)+(s+2(\lambda_{a2}+\lambda_{r2}))$$
$$\left(-(s+2(\lambda_2+\lambda_{r1}))\left(-\overline{k_1k_2}+\frac{\lambda_{a1}}{s}-\frac{\lambda_1\left(s\overline{k_1}\lambda_2-\lambda_{a1}(s+\lambda_{r1})+s\overline{k_2}(s+\lambda_2+\lambda_{r1})\right)}{s(s^2+(\lambda_2+\lambda_{a2}+\lambda_{r1}+\lambda_{r2})s-\lambda_1\lambda_2+\lambda_{a2}\lambda_{r1}+\lambda_2\lambda_{r2}+\lambda_{r1}\lambda_{r2})}-\right.\right.$$
$$\frac{\lambda_{a1}\left(s\overline{k_1}\lambda_2-\lambda_{a1}(s+\lambda_{r1})+s\overline{k_2}(s+\lambda_2+\lambda_{r1})\right)}{s(s^2+(\lambda_2+\lambda_{a2}+\lambda_{r1}+\lambda_{r2})s-\lambda_1\lambda_2+\lambda_{a2}\lambda_{r1}+\lambda_2\lambda_{r2}+\lambda_{r1}\lambda_{r2})}+$$
$$\left.\left.\frac{\lambda_{a1}\left(s\overline{k_2}(\lambda_1+\lambda_{a2})+\lambda_{a1}(s-\lambda_1+\lambda_{r2})+s\overline{k_1}(s+\lambda_{a2}+\lambda_{r2})\right)}{s(s^2+(\lambda_2+\lambda_{a2}+\lambda_{r1}+\lambda_{r2})s-\lambda_1\lambda_2+\lambda_{a2}\lambda_{r1}+\lambda_2\lambda_{r2}+\lambda_{r1}\lambda_{r2})}\right)-\right.$$
$$\left.\left.\lambda_2\left(\overline{k_1}-\overline{k_1^2}-\frac{2\lambda_{a1}\left(s\overline{k_2}(\lambda_1+\lambda_{a2})+\lambda_{a1}(s-\lambda_1+\lambda_{r2})+s\overline{k_1}(s+\lambda_{a2}+\lambda_{r2})\right)}{s(s^2+(\lambda_2+\lambda_{a2}+\lambda_{r1}+\lambda_{r2})s-\lambda_1\lambda_2+\lambda_{a2}\lambda_{r1}+\lambda_2\lambda_{r2}+\lambda_{r1}\lambda_{r2})}\right)\right)\right)\Big/$$
$$\left((s+\lambda_2+\lambda_{a2}+\lambda_{r1}+\lambda_{r2})\left(s^2+2(\lambda_2+\lambda_{a2}+\lambda_{r1}+\lambda_{r2})s+4(-\lambda_1\lambda_2+\lambda_{a2}\lambda_{r1}+(\lambda_2+\lambda_{r1})\lambda_{r2})\right)\right)$$

C.5. SIC-P2P Model: $\psi_{12}(s)$ Formula

$$\psi_{12}(s) = \Bigg((-\lambda_{a2} - \lambda_1)\,(2\,(\lambda_2 + \lambda_{r1}) + s) \Bigg(\frac{2\,\lambda_{a1}\left(s\,\overline{k_2}\,(\lambda_2 + \lambda_{r1} + s) + \lambda_2\, s\,\overline{k_1} - \lambda_{a1}\,(\lambda_{r1} + s)\right)}{s\,(\lambda_{a2}\,\lambda_{r1} + s\,(\lambda_{a2} + \lambda_2 + \lambda_{r1} + \lambda_{r2}) - \lambda_1\,\lambda_2 + \lambda_{r1}\,\lambda_{r2} + \lambda_2\,\lambda_{r2} + s^2)} + \overline{k_2} - \overline{k_2^2} - \frac{2\,\lambda_{a1}}{s} \Bigg) +$$

$$(2\,(\lambda_{a2} + \lambda_{r2}) + s) \Bigg(-(2\,(\lambda_2 + \lambda_{r1}) + s) \Bigg(-\frac{\lambda_1\left(s\,\overline{k_2}\,(\lambda_2 + \lambda_{r1} + s) + \lambda_2\, s\,\overline{k_1} - \lambda_{a1}\,(\lambda_{r1} + s)\right)}{s\,(\lambda_{a2}\,\lambda_{r1} + s\,(\lambda_{a2} + \lambda_2 + \lambda_{r1} + \lambda_{r2}) - \lambda_1\,\lambda_2 + \lambda_{r1}\,\lambda_{r2} + \lambda_2\,\lambda_{r2} + s^2)} -$$

$$\frac{\lambda_{a1}\left(s\,\overline{k_2}\,(\lambda_2 + \lambda_{r1} + s) + \lambda_2\, s\,\overline{k_1} - \lambda_{a1}\,(\lambda_{r1} + s)\right)}{s\,(\lambda_{a2}\,\lambda_{r1} + s\,(\lambda_{a2} + \lambda_2 + \lambda_{r1} + \lambda_{r2}) - \lambda_1\,\lambda_2 + \lambda_{r1}\,\lambda_{r2} + \lambda_2\,\lambda_{r2} + s^2)} +$$

$$\frac{\lambda_{a1}\left(s\,\overline{k_1}\,(\lambda_{a2} + \lambda_{r2} + s) + s\,\overline{k_2}\,(\lambda_{a2} + \lambda_1) + \lambda_{a1}\,(-\lambda_1 + \lambda_{r2} + s)\right)}{s\,(\lambda_{a2}\,\lambda_{r1} + s\,(\lambda_{a2} + \lambda_2 + \lambda_{r1} + \lambda_{r2}) - \lambda_1\,\lambda_2 + \lambda_{r1}\,\lambda_{r2} + \lambda_2\,\lambda_{r2} + s^2)} - \overline{k_1\,k_2} + \frac{\lambda_{a1}}{s} \Bigg) -$$

$$\lambda_2 \Bigg(-\frac{2\,\lambda_{a1}\left(s\,\overline{k_1}\,(\lambda_{a2} + \lambda_{r2} + s) + s\,\overline{k_2}\,(\lambda_{a2} + \lambda_1) + \lambda_{a1}\,(-\lambda_1 + \lambda_{r2} + s)\right)}{s\,(\lambda_{a2}\,\lambda_{r1} + s\,(\lambda_{a2} + \lambda_2 + \lambda_{r1} + \lambda_{r2}) - \lambda_1\,\lambda_2 + \lambda_{r1}\,\lambda_{r2} + \lambda_2\,\lambda_{r2} + s^2)} + \overline{k_1} - \overline{k_1^2} \Bigg) \Bigg) \Bigg) \Bigg/$$

$$\left((\lambda_{a2} + \lambda_2 + \lambda_{r1} + \lambda_{r2} + s)\left(4\,(\lambda_{a2}\,\lambda_{r1} - \lambda_1\,\lambda_2 + (\lambda_2 + \lambda_{r1})\,\lambda_{r2}) + 2\,s\,(\lambda_{a2} + \lambda_2 + \lambda_{r1} + \lambda_{r2}) + s^2\right)\right)$$

C.6. SIC-P2P Model: $\psi_2(s)$ Formula

$$\psi_2(s) = \frac{1}{s + 2(\lambda_{a2} + \lambda_{r2})}\left(-\overline{k_2} + \overline{k_2^2} + \frac{2\lambda_{a1}}{s} + \left(2\lambda_2\left((-\lambda_1 - \lambda_{a2})(s + 2(\lambda_2 + \lambda_{r1}))\right.\right.\right.$$
$$\left(\overline{k_2} - \overline{k_2^2} - \frac{2\lambda_{a1}}{s} + \frac{2\lambda_{a1}\left(s\,\overline{k_1}\,\lambda_2 - \lambda_{a1}(s + \lambda_{r1}) + s\,\overline{k_2}(s + \lambda_2 + \lambda_{r1})\right)}{s(s^2 + (\lambda_2 + \lambda_{a2} + \lambda_{r1} + \lambda_{r2})s - \lambda_1\lambda_2 + \lambda_{a2}\lambda_{r1} + \lambda_2\lambda_{r2} + \lambda_{r1}\lambda_{r2})}\right) + (s + 2(\lambda_{a2} + \lambda_{r2}))$$
$$\left(-(s + 2(\lambda_2 + \lambda_{r1}))\left(-\overline{k_1\,k_2} + \frac{\lambda_{a1}}{s} - \frac{\lambda_1\left(s\,\overline{k_1}\,\lambda_2 - \lambda_{a1}(s + \lambda_{r1}) + s\,\overline{k_2}(s + \lambda_2 + \lambda_{r1})\right)}{s(s^2 + (\lambda_2 + \lambda_{a2} + \lambda_{r1} + \lambda_{r2})s - \lambda_1\lambda_2 + \lambda_{a2}\lambda_{r1} + \lambda_2\lambda_{r2} + \lambda_{r1}\lambda_{r2})} - \right.\right.$$
$$\frac{\lambda_{a1}\left(s\,\overline{k_1}\,\lambda_2 - \lambda_{a1}(s + \lambda_{r1}) + s\,\overline{k_2}(s + \lambda_2 + \lambda_{r1})\right)}{s(s^2 + (\lambda_2 + \lambda_{a2} + \lambda_{r1} + \lambda_{r2})s - \lambda_1\lambda_2 + \lambda_{a2}\lambda_{r1} + \lambda_2\lambda_{r2} + \lambda_{r1}\lambda_{r2})} +$$
$$\left.\frac{\lambda_{a1}\left(s\,\overline{k_2}(\lambda_1 + \lambda_{a2}) + \lambda_{a1}(s - \lambda_1 + \lambda_{r2}) + s\,\overline{k_1}(s + \lambda_{a2} + \lambda_{r2})\right)}{s(s^2 + (\lambda_2 + \lambda_{a2} + \lambda_{r1} + \lambda_{r2})s - \lambda_1\lambda_2 + \lambda_{a2}\lambda_{r1} + \lambda_2\lambda_{r2} + \lambda_{r1}\lambda_{r2})}\right) -$$
$$\left.\left.\left.\lambda_2\left(\overline{k_1} - \overline{k_1^2} - \frac{2\lambda_{a1}\left(s\,\overline{k_2}(\lambda_1 + \lambda_{a2}) + \lambda_{a1}(s - \lambda_1 + \lambda_{r2}) + s\,\overline{k_1}(s + \lambda_{a2} + \lambda_{r2})\right)}{s(s^2 + (\lambda_2 + \lambda_{a2} + \lambda_{r1} + \lambda_{r2})s - \lambda_1\lambda_2 + \lambda_{a2}\lambda_{r1} + \lambda_2\lambda_{r2} + \lambda_{r1}\lambda_{r2})}\right)\right)\right)\right)\Big/$$
$$\left((s + \lambda_2 + \lambda_{a2} + \lambda_{r1} + \lambda_{r2})\left(s^2 + 2(\lambda_2 + \lambda_{a2} + \lambda_{r1} + \lambda_{r2})s + 4(-\lambda_1\lambda_2 + \lambda_{a2}\lambda_{r1} + (\lambda_2 + \lambda_{r1})\lambda_{r2})\right) -$$
$$\left.\frac{2\lambda_{a1}\left(s\,\overline{k_1}\,\lambda_2 - \lambda_{a1}(s + \lambda_{r1}) + s\,\overline{k_2}(s + \lambda_2 + \lambda_{r1})\right)}{s(s^2 + (\lambda_2 + \lambda_{a2} + \lambda_{r1} + \lambda_{r2})s - \lambda_1\lambda_2 + \lambda_{a2}\lambda_{r1} + \lambda_2\lambda_{r2} + \lambda_{r1}\lambda_{r2})}\right)$$