

UNIVERSITY OF TECHNOLOGY SYDNEY
Faculty of Engineering and Information Technology

**Modeling and Analysis of Advanced Persistent
Threats in Cyber Space**

by

Xu Wang

A THESIS SUBMITTED
IN PARTIAL FULFILLMENT OF THE
REQUIREMENTS FOR THE DEGREE

Doctor of Philosophy

Sydney, Australia

2020

Certificate of Authorship/Originality

I, Xu Wang declare that this thesis, is submitted in fulfilment of the requirements for the award of Doctor of Philosophy, in the Faculty of Engineering and Information Technology at the University of Technology Sydney. This thesis is wholly my own work unless otherwise reference or acknowledged. In addition, I certify that all information sources and literature used are indicated in the thesis. I certify that the work in this thesis has not been previously submitted for a degree nor has it been submitted as a part of the requirements for other degree except as fully acknowledged within the text. This thesis is the result of a research candidature jointly delivered with Beijing University of Posts and Telecommunications as part of a Collaborative Doctoral Research Degree. This research is supported by the Australian Government Research Training Program.

Production Note:

Signature: Signature removed prior to publication.

Date: 07/01/2020

©Copyright 2020 Xu Wang

Dedication

This thesis is dedicated to my parents.

This stands as a testimony for their endless support and love.

To my supervisors, for the academic guidance.

To my friends, for their encouragement.

Acknowledgements

I have to start by thanking my perfect parents Daofeng Wang and Peizhen Zhang. They bring me to this fantastic world, support me to pursue research, allow me to be myself. Many Thanks!

Throughout the doctoral program, I have received a great deal of support and assistance. I would like to express my sincere gratitude to Prof. Y. Jay Guo for his invaluable support and immense knowledge. My deepest thanks also go to Prof. Ren Ping Liu for all the opportunities he offered and his patience and help. I would like to show my very profound gratitude to Dr. Wei Ni in CSIRO. My research would have been impossible without his support and supervision. I would also like to acknowledge Prof. Xinxin Niu and Associate Prof. Kangfeng Zheng in BUPT. I would also like to thank everyone in GBDTC and UTS who helped me so much.

I must express my sincere thanks to my co-author, Xuan Zha, for the company throughout the whole Ph.D. period from China to Australia. A very special gratitude goes out to my mates Bo Song, Guangsheng Yu, Shangjing Lin, Cheng Qin, Li Duan, Shoulu Hou, Ping Yu, Yixun Hu, Yue Shi, Qingyang Liu, Qingyuan Kuang, Heng Liu, Chao Jiang, and Yongjun Song. It is great having the awesome time with you guys. I would like to thank the wonderful colleagues in UDT. I am also grateful to my family and friends who have supported me along the way.

Xu Wang

Sydney, Australia, 2020

List of Publications

Published Journal Papers

- J-1. **X. Wang**, X. Zha, W. Ni, R. P. Liu, Y. J. Guo, X. Niu and K. Zheng, “Game Theoretic Suppression of Forged Messages in Online Social Networks,” *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 2019, (Chapter 3).
- J-2. **X. Wang**, W. Ni, K. Zheng, R. P. Liu and X. Niu, “Virus Propagation Modeling and Convergence Analysis in Large-scale Networks,” *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 10, pp. 2241-2254, Oct. 2016 (Chapter 4).
- J-3. **X. Wang**, B. Song, W. Ni, R. P. Liu, Y. J. Guo, X. Niu and K. Zheng, “Group-based Susceptible-Infectious-Susceptible Model in Large-Scale Directed Networks,” *Security and Communication Networks*, vol. 2019, Article ID 1657164, 2019 (Chapter 5).
- J-4. **X. Wang**, G. Yu, X. Zha, W. Ni, Y. J. Guo, X. Niu and K. Zheng, “Capacity of Blockchain based Internet-of-Things: Testbed and Analysis,” *Elsevier Internet of Things*, vol. 8, 100109, 2019.
- J-5. B. Song, **X. Wang**, W. Ni, Y. Song, R. P. Liu, G. Jiang and Y. J. Guo, “Reliability Analysis of Large-Scale Adaptive Weighted Networks,” *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 651-665, 2020.
- J-6. **X. Wang**, X. Zha, W. Ni, R. P. Liu, Y. J. Guo, X. Niu and K. Zheng, “Survey on Blockchain for Internet of Things,” *Computer Communications*, vol. 136, pp. 10-29, 2019.
- J-7. X. Zha, W. Ni, **X. Wang**, R. P. Liu, Y. J. Guo, X. Niu and K. Zheng, “The Impact of Link Duration on the Integrity of Distributed Mobile Networks,” *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 9, pp.

2240-2255, Sept. 2018.

- J-8. X. Zha, **X. Wang**, W. Ni, R. P. Liu, Y. J. Guo, X. Niu and K. Zheng, “Blockchain for IoT: The Tradeoff Between Consistency and Capacity,” *Chinese Journal on Internet of Things*, vol.1, no.1, pp.21-33, 2017.
- J-9. Y. Hu, K. Zheng, **X. Wang** and Y. Yang. “WORM-HUNTER: A Worm Guard System using Software-defined Networking,” *KSII Transactions on Internet and Information Systems*, 11, no. 1, 2017.
- J-10. Y. Xu, C. Wu, K. Zheng, **X. Wang**, X. Niu, and T. Lu., “Computing Adaptive Feature Weights with PSO to Improve Android Malware Detection,” *Security and Communication Networks*, vol. 2017, Article ID 3284080, 14 pages, 2017.

Published Conference Papers

- C-1. **X. Wang**, K. Zheng, X. Niu, B. Wu and C. Wu, “Detection of Command and Control in Advanced Persistent Threat based on Independent Access,” *IEEE International Conference on Communications*, 2016, pp. 1-6, (Chapter 6).
- C-2. **X. Wang**, Y. Ping, G. Yu, W. Ni, R. P. Liu and Y. J. Guo, “A High-Performance Hybrid Blockchain System for Traceable IoT Applications,” *International Conference on Network and System Security*, Springer, Cham, 2019: 721-728.
- C-3. **X. Wang**, X. Zha, G. Yu, W. Ni, R. P. Liu, Y. J. Guo, X. Niu and K. Zheng, “Attack and Defence of Ethereum Remote APIs,” *IEEE Globecom Workshops*, 2018.
- C-4. G. Yu, **X. Wang**, X. Zha, J. A. Zhang and R. P. Liu, “An optimized round-robin scheduling of speakers for peers-to-peers-based byzantine faulty tolerance,” *IEEE Globecom Workshops*, 2018.
- C-5. X. Zha, **X. Wang**, W. Ni, R. P. Liu, Y. J. Guo, X. Niu and K. Zheng, “Analytic model on data security in VANETs,” *International Symposium on Communications and Information Technologies (ISCIT)*, Cairns, QLD, 2017, pp. 1-6.

Patents

- P-1. R. P. Liu, **X. Wang**, G. Yu, J. Baird, “A Machine Type Communication System or Device for Recording Supply Chain Information on a Distributed Ledger in a Peer to Peer Network,” 2019901683, filed on 17 May 2019..
- P-2. **X. Wang**, R. P. Liu, X. Zha, G. Yu, “Secure image capture and storage on blockchain,” 2019903089, filed on 23 Aug 2019.

ABSTRACT

Modeling and Analysis of Advanced Persistent Threats in Cyber Space

by

Xu Wang

Advanced Persistent Threat (APT), a professional cyber threat as indicated by its name, has become a type of significant risk in modern society. APT attackers employ various advanced attack technologies to carry out attacks in multiple stages over a long period of time. Due to its complexity, APT research is challenging and incomplete. This thesis proposes a series of models to analyze key processes of APT, i.e., social attack, propagation, and remote control. To be specific, game theoretic models are proposed to describe social network attacks, and epidemic models based on the susceptible-infected-susceptible process are developed to capture the propagation process; machine learning methods are adopted to detect the remote control traffic.

The main contributions of this thesis can be summarized as follows.

- This thesis proposes infinitely repeated games to capture the interactions between a message publisher and the administrator to suppress social attack messages. Critical conditions, under which the publisher can be disincentivized to send any attack messages, are identified. Closed-form expressions are established to give the maximum number of attack messages from an attacker in the absence or presence of misclassification on genuine messages.
- This thesis proposes a new approach to model the propagation of APT across non-trivial networks. A discrete-time absorbing Markov process of epidemic model is first developed based on the adjacency matrix of the network. Asymptotically accurate bounds of the virus extinction rate are derived. We propose

a practical approach for the estimation of the extinction rate in large networks. Our proposal has been proved theoretically and validated via simulations.

- This thesis proposes a group-based propagation model to analyze the propagation process of APT in large-scale networks. The proposed model is efficient and accurate. The network nodes are divided into groups according to their connectivity. A continuous-time Markov susceptible-infectious-susceptible model is developed. The propagation threshold, under which the propagation will eventually stop, is derived based on the spectral radius of the collapsed adjacency matrix. Simulation results validate the model accuracy and the analytical epidemic threshold.
- This thesis proposes a method of traffic feature analysis to detect the remote control traffic of APT. Based on the independent access feature of APT network traffic, concurrent domains in the domain name service are selected to detect APT domains from domain name system records. The proposed traffic features and detection process are then validated using public datasets.

Contents

Certificate	ii
Dedication	iii
Acknowledgments	iv
List of Publications	v
Abstract	viii
List of Figures	xiv
List of Tables	xviii
Abbreviation	xix
1 Introduction	1
1.1 Background	2
1.1.1 APT Attack Case Studies	6
1.1.2 APT Attack Stages	8
1.2 Research Objectives	10
1.3 Thesis Organization	13
2 Literature Review	17
2.1 General APT Models	18
2.2 Social Attack and Defense Models	19
2.3 Propagation Models	22
2.4 Remote Control Models and Detection	25

2.5 Chapter Conclusion	27
3 Social Attacks Modeling and Analysis	28
3.1 Introduction	28
3.2 Network Model	31
3.3 Games of Forged Messages	35
3.3.1 Misclassification-free Infinitely Repeated Game	35
3.3.2 Infinitely Repeated Game with Message Misclassification	41
3.4 Numerical results	45
3.5 Conclusion	53
4 Markov-based Accurate Propagation Model	54
4.1 Introduction	54
4.1.1 Problem Statement	55
4.1.2 Our Contributions	56
4.1.3 Organization and Notations	57
4.2 Discrete-time Markov model for Virus Propagation in Computer Networks	58
4.3 Extrapolation of Markov Epidemic Modeling to Large Networks	64
4.3.1 Virus Extinction Rate	65
4.3.2 Closed-form Bounds of R	69
4.3.3 Epidemic Lifetime	74
4.3.4 Approximate Infected Population and Infection Probability	75
4.4 Simulation and Numerical results	76
4.4.1 Model Validation	78
4.4.2 Large Network Analysis	84

4.5	Summary	90
5	Group-based Fast Propagation Model	91
5.1	Introduction	91
5.2	The Directed Network Model	93
5.3	Group-based Mean-field SIS Model	96
5.4	Simulation and Numerical Results	102
5.5	Summary	108
6	Remote Control Modeling and Detection	109
6.1	Introduction	109
6.2	Remote Control Detection Method	111
6.2.1	Features of Remote Control in APT	111
6.2.2	Remote Control Detection Details	114
6.3	Performance Evaluation	117
6.3.1	Data Set	117
6.3.2	Experiment Details	118
6.3.3	Experiment results	119
6.4	Discussions and Summary	123
7	Future Works	125
7.1	Future Works on Propagation Models	125
7.1.1	New Propagation Processes	125
7.1.2	Propagation in Adaptive Networks	126
7.1.3	Propagation in Multi-layer Networks	126
7.2	Cyber-Attack Detection	127

7.2.1	Detection of New Attacks	127
7.2.2	Artificial Intelligence-based Detection	127
7.3	Blockchain-based Security	128
7.3.1	Introduction of Blockchain	128
7.3.2	Blockchain-based Security Service	129
8	Contributions	130
	Bibliography	132

List of Figures

1.1	The number of reports about active APT attack groups in 2018 [1].	5
1.2	The distribution of victim industries [1].	6
1.3	Lifecycle of APT Attacks.	8
1.4	Research targets, key research point and chapters.	11
3.1	An illustration on the subscription services in OSNs, where there are a network administrator \mathcal{A} , publisher \mathcal{P} , and subscribers (including followers, fans and bots). \mathcal{P} can publish either genuine or forged messages. The subscribers check the messages and give their feedback to \mathcal{A} . The bots are hired by \mathcal{P} to always give positive feedback to \mathcal{A} . Based on the feedback from the subscribers, \mathcal{A} can take either a <i>trust</i> or <i>distrust</i> strategy against \mathcal{P}	33
3.2	The cumulative payoff and the number of forged messages of \mathcal{P} , where $p_1 = 0.03$, $N_r = 90$, $N_n = 10$, $N_t = 10$, $C_1 = 10$, and $C_3 = 10$. $C_2 = 10, 50$ and 120 . The payoff is the average of 5,000 independent simulations.	46
3.3	The visualization of Theorems 1 and 3.	47
3.4	The auxiliary continuous functions $\tilde{g}_i(x)$, indicating the extra payoffs from forged messages with the growth of x , where $N_r = 90$, $N_n = 10$, $N_t = 10$, $C_1 = 10$, $C_3 = 5$, $p_1 = 0.1$, and $p_2 = 0.5$. Two C_2 , i.e., 50 and 80, are considered. Every dot is the average extra payoff of 1,000 independent simulations.	48

3.5	The maximum number of forged messages of a malicious \mathcal{P} with the growth of C_1 , where $N_r = 90$, $N_n = 10$, $C_2 = 80$, $C_3 = 5$, $p_1 = 0.1$, and $p_2 = 0.5$. Two values of N_t , i.e., 0 and 30, are considered.	49
3.6	The maximum number of forged messages of \mathcal{P} , i.e., x_{u_1} and x_{u_2} , by publishing forged messages, where $N_r = 90$, $N_n = 10$, $C_1 = 30$, $C_2 = 50$, $C_3 = 5$, and $p_2 = 0.5$. $N_t = 0$ and 100, are considered.	50
3.7	The maximum extra payoff of \mathcal{P} in the misclassification-free game with the growth of N_t , where $N_r = 90$, $N_n = 10$, $C_2 = 50$, $p_1 = 0.5$. Every dot is an average result of 2,000 independent runs.	51
3.8	The maximum number of forged messages with the growth of the ratio of fans, i.e., $\frac{N_n}{N_r+N_n}$, where $N_r + N_n = 100$, $N_t = 10$, $C_1 = 30$, $C_3 = 5$, $C_2 = 50$ and $p_1 = 0.1$	52
4.1	An example of the BA-2 graph.	59
4.2	Flow chart to construct the transition matrix of the Discrete-time Markov model	61
4.3	A graph demonstrates the problem that which state has a larger $p_{i,0}$. State \mathbf{s}_g contains 1 infected node with 3 susceptible neighbors. On the contrary, state \mathbf{s}_h contains 4 infected nodes with 1 susceptible neighbor. Which one is larger, $p_{g,0}$ or $p_{h,0}$?	70
4.4	Model Validation	79
4.5	Evaluation of the proposed bounds of the virus extinction rate, where BA-2, ring and regular graphs are considered	81

4.6	The actual achievable convergence rate R vs. the convergence rate requirement where complete graphs of N nodes are considered ($N = 2, 4, 6, 8, 10, 11$), $\beta = 0.5$ and δ is adaptively calculated using the upper bound of Theorem 6 under any given convergence rate requirement.	82
4.7	The actually achieved extinction rate versus the target mean extinction rate, where a 5000-node network with topology of complete graph \mathcal{K}_{5000} is considered, and $\beta = 0.5$	85
4.8	The actually achieved virus lifetime versus the target mean virus lifetime, where a 5000-node network with topology of \mathcal{K}_{5000} is considered and $\beta = 0.5$	86
4.9	Virus lifetime versus network connectivity, where a 5000-node network with topology of regular graph is considered, the degree of the regular graph increases from 20% to 100% of the remaining nodes (e.g., d ranges from 1000 to 4999 for 5000-node networks). . . .	87
5.1	An example of networks we considered where 7 nodes are categorized into three groups according to the connection between them.	95
5.2	An example of $[ABC]_{ijk}$ and $[ABC]'_{ijk}$	96
5.3	The growth of infection density where a complete graph with 1000 nodes is considered. $\beta = 0.0005$ and $\delta = 0.1$. The analytical results are obtained based on (5.6) by evenly dividing the nodes into 1, 2, 5, 10, 25 and 50 groups.	103
5.4	The infection density with the growth of time where a complete graph with 1000 nodes is considered. The analytical results are obtained based on (5.6) by evenly dividing the nodes into 5 groups where $\tau = \frac{\beta}{\delta} = 0.002, 0.0015, 0.0011$ and 0.001 , respectively.	104

5.5	The validation of epidemic threshold given by (5.20), where the y -axis is the infection density at $t = 1000$. Three networks with 500 nodes are considered. δ is set to be 0.1. $\alpha = 4, 6, 8$ is used to adjust the number of edges.	106
5.6	The validation of epidemic threshold, where the y -axis is the infected population at $t = 1000$. Different scales of networks ($N=100, 200$ and 300 , respectively) are considered. δ is set to be 0.1.	107
6.1	The number of CODDs of some popular domains	114
6.2	Generative process of the feature vector	117
6.3	Distribution of Average Number	121
6.4	Distribution of the Highest Confidence level	122

List of Tables

3.1	Notations Used in Chapter 3	31
3.2	The payoff matrix to \mathcal{P} per round in the case of misclassification-free game/situations	35
3.3	The payoff matrix of \mathcal{P} per round in the game/situations with misclassification	41
3.4	The confusion matrix for detection results	41
4.1	Notations Used in Chapter 4	58
4.2	The average running time of simulating a single virus propagation process (in seconds), corresponding to Fig. 4.9	89
5.1	Notations Used in Chapter 5	94
6.1	Notations Used in Chapter 6	110
6.2	Remote Control Detection Rules	119
6.3	Confusion Matrix of Classification	120

Abbreviation

AN : Average Number

APT : Advanced Persistent Threat

CIA : Confidentiality, Integrity and Availability

CODD : Concurrent Domains in DNS Records

CPU : Central Processing Unit

DDoS : Distributed Denial of Service

DNS : Domain Name System

HC : Highest Confidence

HTTP : HyperText Transfer Protocol

HTTPS : HyperText Transfer Protocol Secure

IoT : Internet of Things

IP : Internet Protocol

IRC : Internet Relay Chat

LANL : Los Alamos National Laboratory

OSN : Online Social Network

P2P : Peer-to-peer

SI : Susceptible-Infected

SIR : Susceptible-Infected-Recover

SIS : Susceptible-Infected-Susceptible

SLD: Second-Level Domain

TCP : Transmission Control Protocol

TTL : Time to Live

URL : Uniform Resource Locator

Chapter 1

Introduction

The rapidly developing network technology and computer technology are reshaping society. While improving efficiency, these technologies enable cyber attacks to a whole new level. Cyber attacks can be more targeted, professional and persistent than ever before. A new type of cyber attack, i.e., Advanced Persistent Threat (APT), has been proposed and become popular. New theories and models need to be developed to analyze the complex interaction in APT attacks. However, this can be a challenging topic. This is first because APT attacks can be across multiple networks, e.g., social networks and computer networks, and behave in completely different ways. A single model cannot capture complex interactions of APT in different networks. Another challenge is the long-time range and large-scale of APT attacks. Developed models should be able to solve this challenge, typically by simplifying attacks and reducing model complexity.

This thesis models and analyzes key social attack stages, propagation stages, and remote control stages during APT attack process. Based on unique behaviors of different stages, this thesis proposes a series of models to capture the interaction between APT attackers and defenders by employing game theory, Markov theory, virus propagation model, stability analysis, machine learning, etc. Numerical and simulation results are carried out in the proposed models, which help to recover APT attack scenario, evaluate defense effect, and allocate defense resources.

This chapter first introduces the background of network security, including the definition of APT, features of APT, typical APT attack cases and stages of APT.

This chapter then illustrates research topics and key research points, followed by the overview of this thesis.

1.1 Background

The rapid development of information technology enables massive devices and users to be connected [2, 3]. In 2018, more than 4 billion individuals and 17 billion devices can interact with one another in real-time via the Internet. In addition to the traditional computer network, new networks, e.g., mobile network [4], Internet of Things (IoT) network [3], vehicular network [5], industrial network [6] and smart city [7], have been realized to connect various objects and extend the network. The cybersecurity, meanwhile, has become a critical issue. Attacks can damage the Confidentiality, Integrity, and Availability (CIA) of network assets. For example, the Domain Name Systems (DNS) service provider Dyn was under a large-scale Distributed Deny-of-Service (DDoS) attack, where many web services were stopped, e.g., Twitter and Airbnb [8]. The attackers can also launch attacks on physical equipment over the computer network. For example, in 2009, Iran’s nuclear facilities were damaged by the Stuxnet cyber attack [9]. In 2015, Ukraine suffered a large-scale blackout due to a cyber attack on the power grid [10]. As a result, cybersecurity has become a part of national defense. To protect cyber space, many countries have set up specific agencies and released regulations or law against attacks [11].

It is hard for cyberspace to ensure complete cybersecurity. This is first because security vulnerabilities can be found in various modules in cyberspaces, e.g., hardware, software, and protocols. According to the National Vulnerability Database (NVD), more than 100,000 vulnerabilities have been found until 2018 [12]. Besides applying existed vulnerabilities, attackers can also perform vulnerability mining to find unpublished vulnerabilities, with which attackers can efficiently break into target networks and avoid detection. On the other hand, due to the fact that indi-

viduals and devices have been connected by different networks, attackers can reach their targets via the networks and implement attacks. Among the networks, social networks have drawn much attention from attackers. For example, more than 200 million phishing links are blocked by Trend Micro in 2018 [13].

Recently, a type of cyber attack has become popular and is significantly different from traditional network attacks, e.g., worm and DDoS, with features of advanced, targeted and persistent. The terminology Advanced Persistent Threat (APT) was proposed to describe the advanced and sophisticated cyber attacks. In 2006, the name of APT was first used by the US Air Force [14], which is responsible for US national cybersecurity, and then became well known in 2010 because of a popular article analyzing the attack on Google, namely Aurora [15]. Consisting of five phases, Aurora is typically an advanced and sophisticated cyber attack. At the initial phase, attackers collect information of selected employees in targeted companies from online social networks. After that, attackers create a malicious website and send the link to the website to the selected employees. At the third phase, attackers can gain access to the hosts owned by the selected employees. The attackers then build secret remote control channels to the hosts with the Secure Sockets Layer (SSL) protocol. At the last phase, attackers search Gmail data using the credential of the attacked employees and then retrieve the data from the remote control channels.

In 2011, APT was officially defined by the National Institute of Standards and Technology (NIST) as follows [16]:

An adversary that possesses sophisticated levels of expertise and significant resources which allow it to create opportunities to achieve its objectives by using multiple attack vectors (e.g., cyber, physical, and deception). These objectives typically include establishing and extending footholds within the information technology infrastructure of the targeted organizations for purposes of exfiltrating information,

undermining or impeding critical aspects of a mission, program, or organization; or positioning itself to carry out these objectives in the future. The advanced persistent threat: (i) pursues its objectives repeatedly over an extended period of time; (ii) adapts to defenders' efforts to resist it; and (iii) is determined to maintain the level of interaction needed to execute its objectives.

Compared with traditional network attacks, APT has the following outstanding features.

- Targeted

Traditional network attacks attack every reachable computer, e.g., worms. However, only limited devices are under attack in APT. To achieve this, adversaries usually initial attacks through targeted social attacks, where attack payloads are customized according to the targets' profile and then only delivered to the targeted employees. Meanwhile, attackers would carefully control the range of infected nodes to stay inconspicuous.

- Advanced

APT adversaries can attack their targets through multiple networks, e.g., social networks and IoT networks. To be specific, the adversaries usually start cyber intrusions with social attacks. Popular payloads used in social attacks include fishing links, malicious attachments, and watering hole webpages. The adversaries can go beyond computer networks and attack the physical devices via IoT networks. The adversaries can also mine new vulnerabilities according to the intelligence of targets and then customize attack tools.

- Persistent

APT attacks can hide in targeted systems up to hundreds of days before being detected [17], benefited from various antidetection technologies in APT at-

tacks. For example, zero-day exploits, widely used in APT, disable signature-based Intrusion Detection System (IDS). APT attacks also apply encryption technologies or the Virtual Private Network (VPN) technology on the remote control traffic to keep the remote control traffic stay imperceptible. Meanwhile, as a kind of targeted cyber attack, patient adversaries in APT attacks can spend a lot of time to collect intelligence, customize attack plans and upgrade attacks until fulfilling their attack targets.

APT attacks are often labeled with their attack groups to classify the attacks and summarize attack strategies of the attack groups. According to the 360 APT 2018 annual report [1], active APT attack groups and the number reports related their attacks are shown in Figure 1.1.

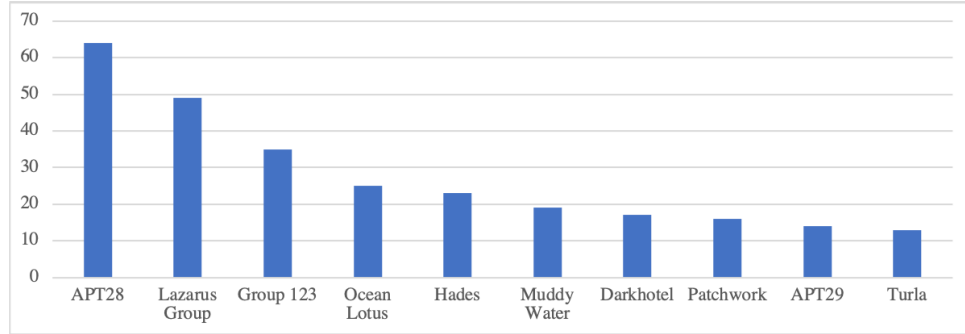


Figure 1.1 : The number of reports about active APT attack groups in 2018 [1].

Driven by clear attack missions, APT attacks can achieve high success rates and cause massive damage. According to 360 APT 2018 annual report [1], the distribution of victim industries is illustrated in Figure 1.2. Large organizations, e.g., government agencies, financial institutions, and diplomatic services, have gained much interest of APT adversaries. The key services to national security, such as military, industry and national defense, have become targets of APT attackers.

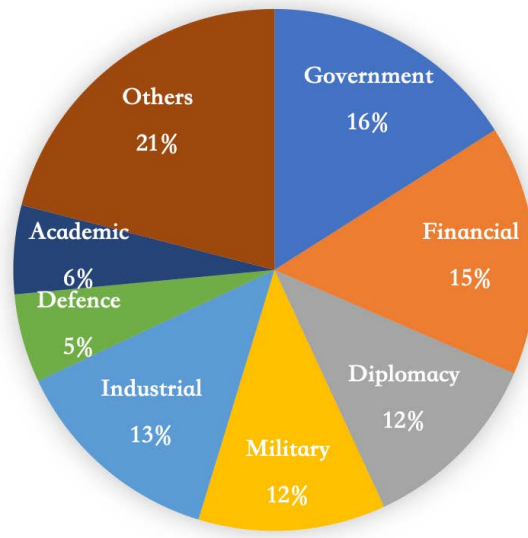


Figure 1.2 : The distribution of victim industries [1].

1.1.1 APT Attack Case Studies

As a kind of targeted attack, every APT attack can be customized according to victim information, attackers' skills and attack plan. Thus, APT attacks can be various on attack details, e.g., attack payloads and targets. Three famous APT attacks are introduced where their common features are summarized, and their unique behaviors are identified.

Stuxnet [9]

In 2009, Iran's nuclear facilities suffered from a complex and targeted attack, namely, the Stuxnet attack, during which Iran's nuclear development was seriously delayed. As a professional cyber attack, Stuxnet applied two zero-day vulnerabilities, spread through physical devices, and attacked Supervisory Control and Data Acquisition (SCADA) systems via computers. As a novel cyber attack, Stuxnet realized the attack on physical devices beyond the limitation of computer networks. To be specific, attackers in Stuxnet sent control commands to the key equipment of the Iranian nuclear factory through a Programmable Logical Controller (PLC) to

overload the equipment and damage them. In this attack, the attackers deployed two remote control centers in two countries [18]. The remote control centers can control infected systems or update the attack program. Once a computer was infected (or attacked), it returned its internal Internet Protocol (IP) address, public IP address, operating system information and other information to the remote control centers. In the later stage of the attack, the Stuxnet virus propagated in the Middle East and resulted in large-scale infection. By 2010, more than 60 thousand computers were infected in Iran [19].

Flame [20]

In 2012, the Flame virus, designed for intelligence gathering, was discovered in countries in the Middle East [21]. Flame attack monitored various communication channels between computer networks and the physical world, typically by using key loggers, screenshots, microphones, and cameras, to collect various intelligence. In the Flame attack, more than 80 remote control centers were found, where most of them realized remote control over HyperText Transfer Protocol (HTTP), HyperText Transfer Protocol Secure (HTTPS) and Secure Shell (SSH). The Flame virus could directly upload the collected information to the remote control centers if the network is available. If the infected hosts lost connection, the Flame Virus could save the collected information in removable storages and then infect other hosts and upload the collected information through the newly infected hosts.

RSA SecurID Attack [22]

In 2011, RSA's SecurID token was hacked by APT attackers. As a result, the companies using SecurID as credentials were compromised and suffered from information leakage. In the initial phase of the APT attack, two phishing emails were sent to RSA's employees. After one of the employees had opened the attachment, a zero-day exploit was triggered resulting that the employee's computer was com-

promised and establish an initial remote control channel [23]. APT adversaries then attacked other hosts in the RSA network and stole a large amount of confidential information including the RSA SecurID token.

1.1.2 APT Attack Stages

It can be concluded from the APT cases and a series of APT reports [24, 9, 25] that APT attacks employ remote control and lateral movement. Besides these two processes, APT attacks also have other common behaviors. According to research [24, 18], the lifecycle of APT attacks can be divided into five stages, namely reconnaissance, initial attack, remote control, lateral movement, and attack. The attack stages and attack behaviors in different stages are shown in Figure 1.3.

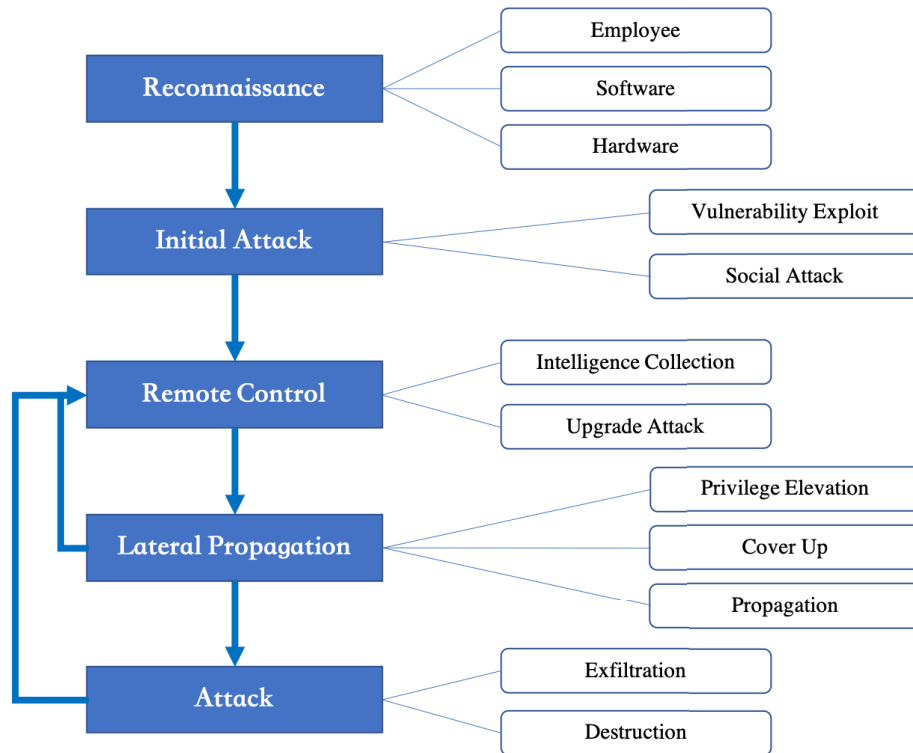


Figure 1.3 : Lifecycle of APT Attacks.

a) Reconnaissance

APT attackers first collect intelligence of their attack targets through multiple channels and then set up profiles for the targets. Attacks can achieve high attack success rates with rich intelligence. The first part of intelligence is about employees including their social activities, life habits, and favorite websites. Employee-related intelligence can be collected from social networks [26]. The intelligence also includes software and hardware in target networks. The hardware information includes network devices, computers, servers, and security devices. The hardware information can be collected by analyzing the supply chain and leaked documents. The software information, e.g., operating systems, network information, network services, and database services, can be detected by various network attacks, such as porting scanning, server scanning and computer viruses [27, 20].

b) Initial Attack

At this stage, APT attackers create attack plan based on the collected intelligence, break through the protection of the target network and invade a small number of nodes in the target network. Vulnerability exploits, including known vulnerability or zero-day vulnerability [28], are popular initial attack technologies, such as the initial attack stage of the Equifax's APT attack in 2017. Social engineering attack, e.g., spearfishing attacks or waterhole attacks [29, 30], is another type of popular attack technology at initial attack stage, such as the initial attacks in the Aurora attack [15] and RSA SecurID leak attacks [23].

c) Remote Control

APT attacks leave backdoors on victims in order to remotely control victims. The remote control can be used to transmit collected intelligence. For example, the Flame attack collects various intelligence and returns it to the attacker for other intrusions through remote control channels [21]. The remote control channel can also be used to control attacks. For example, in the Stuxnet attack, malware on infected

nodes can remain silent or be activated under the control of remote attackers [31]. Remote control traffic is often encrypted and compressed to stay inconspicuous and avoid detection [24].

d) Lateral Propagation

After APT attacks break into target networks, they will spread and attack other nodes in the internal target networks to collect more information and elevate privilege. Popular attack techniques include stealing users' credentials and vulnerability exploits. Some professional APT attacks can develop specific propagation technology to achieve lateral propagation while avoiding detection. For example, the Flame attack can spread itself by hijacking Windows system update process [32]. In target networks, attacks use encryption techniques, zero-day vulnerabilities and other hidden technologies to cover attacks and extend the duration of attacks [30].

e) Attack

APT attacks complete attack missions at this stage. A popular attack target is intelligence property, such as in the Flame attack and RSA SecurID leakage attack. In this type of attack, attackers utilize the remote control channels to collect stolen intelligence properties. APT attacks are also designed to damage target networks and reachable services, e.g., the Stuxnet attack and the APT attack on Ukrainian power grid [9, 33].

1.2 Research Objectives

As a type of cyber threat with complex behavior, long-term harassment, and multidimensional attacks, APT has not been fully studied. This thesis intends to decouple APT attacks and then analyze APT stages. Based on the lifecycle of APT, i.e., reconnaissance, initial attack, remote control, lateral propagation, and attack, a series of models are set up to capture the complicated interactions between attackers

and defenders. The proposed models can help defenders to understand the attack process, evaluate defense strategies, and reduce attack loss. The research targets, key research points and the structure of this thesis are illustrated in Figure 1.4.

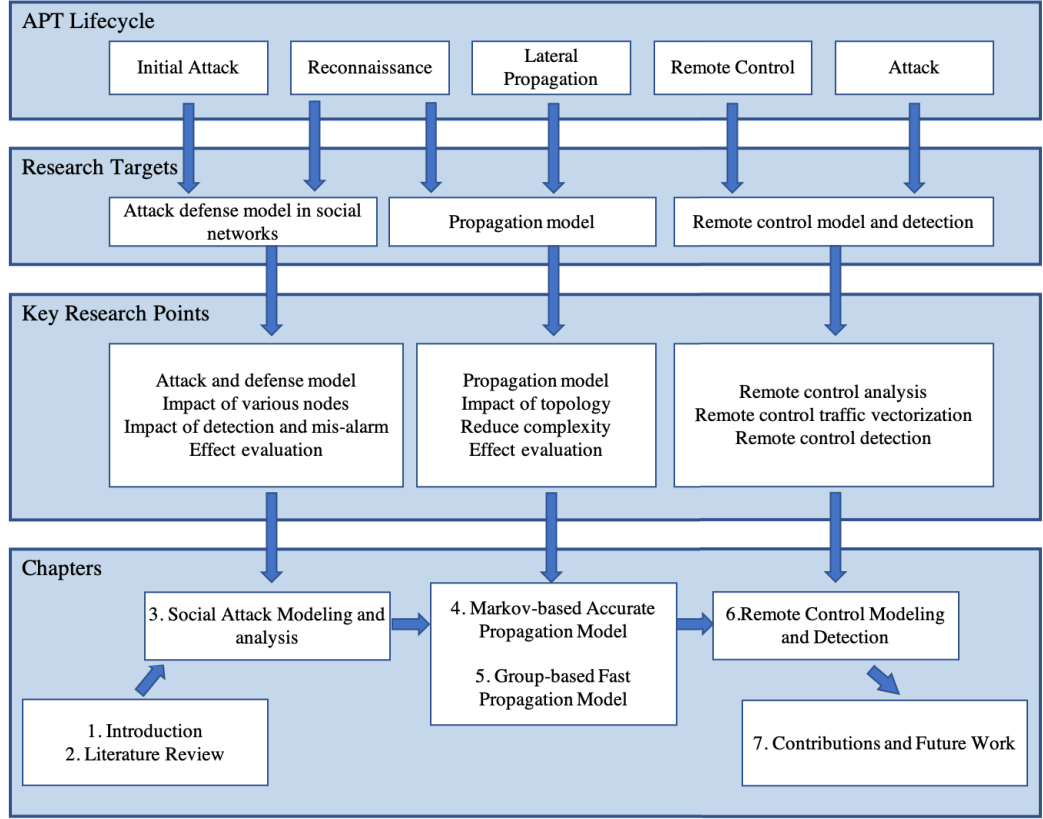


Figure 1.4 : Research targets, key research point and chapters.

At the first stage, APT attackers carry out intelligence gathering [26] and deploy initial intrusion [34] in social networks. The first research target of this thesis is the attack-defense model in online social networks (OSNs). One challenge is that APT attack is a long-term and customized threat where attackers and defenders can update their strategies over an infinite-time horizon according to historical actions. As a result, interactions between attackers and defenders can be sophisticated. Another challenge is that different types of participators may coexist in online social networks, including followers, fans, and bots, reacting distinctly to message pub-

lisher's behaviors. For example, the existence of fans and bots may mislead other nodes in the social network, causing attack messages to be judged as legitimate messages or a reverse misjudgment. The third chapter of this thesis builds infinitely repeated games for the attack-defense process in online social networks. The games can capture behaviors of various participators in the absence and presence of misclassification. Based on the constructed models, numerical results are presented to compare different defense strategies.

APT attacks can perform reconnaissance by employing large-scale malware, such as the Flame attack [20], where the malware can monitor network traffic and steal key data. During this process, malware constantly infects new nodes to collect more information. Another case of propagation in APT occurs in internal networks of victims at the lateral propagation stage. After APT attack has gained an initial foothold in the target's internal network, it keeps on spreading in the internal network and infecting other nodes to gather internal information, elevate its privilege, and prepare for succeeding attacks. The second topic of this thesis is to establish propagation models to capture large-scale infections of APT. In this part, the first research point is to evaluate the impact of network structure on the propagation process. This is because a computer network is with a fixed topology, which has a great impact on the propagation process as indicated by previous studies [35]. Secondly, the proposed model should be able to analyze large-scale and long-term propagation processes. This is because APT can infect massive nodes at the reconnaissance stage. Meanwhile, APT usually targets at large organizations with many hosts. On the other hand, it is difficult to totally remove APT attacks from internal networks because of APT's professional attack skills and long-term dynamic feature. The fourth and fifth chapters of this thesis analyze the propagation process of APT based on virus propagation models and Markov theory. The models can evaluate the propagation process by deducing key indicators, e.g., propagation threshold and

survival time, combining the impact of network topology. The two chapters also propose methods to extend the models to large-scale networks. Numerical results are given to validate the proposed models and extension methods.

APT attackers need to establish remote control channels on the victim nodes in order to maintain long-term and controllable attacks. In popular information leakage attacks, the collected information is also transmitted to remote control centers to complete attack missions. The final research topic is to analyze the remote control traffic in APT attacks and then detect APT by identifying the remote control traffic. The first research point of this topic is to analyze the remote control traffic and identify its unique characteristics which are able to distinguish APT attacks from traditional network attacks such as botnets. This thesis then proposes to study the data processing algorithms to classify the remote control traffic for APT detection. The sixth chapter of this thesis studies the remote control traffic in APT and proposes a DNS traffic vectorization method for APT detection. Finally, the proposed detection method is validated in a public dataset and discussed.

1.3 Thesis Organization

This thesis is based on the research results of 3 journal papers and 1 conference paper. The thesis is organized as follows.

Chapter 1: This chapter gives an overview of this research. It first presents the research background of this thesis, followed by the main motivations and key research points for APT modeling and analysis. This chapter also illustrates the structure of this thesis.

Chapter 2: This chapter presents a comprehensive literature review of the theories related to APT modeling and previous works. Following the lifecycle of APT, the literature review starts with general APT modeling followed by social

attack modeling. This chapter also reviews the propagation theories and models for remote control. This chapter summarizes current research progress and lays the foundation of the reset chapters.

Chapter 3: This chapter proposes a model to analyze the social attack in APT. To be specific, this chapter carries out the analysis by taking a game theoretic approach, where infinitely repeated games are constructed to capture the interactions between a publisher and a network administrator and suppress forged messages in OSNs. Critical conditions, under which the publisher is disincentivized to publish any forged messages, are identified in the absence and presence of misclassification on genuine messages. Closed-form expressions are established for the maximum number of forged messages that a malicious publisher could publish. Confirmed by the numerical results, the proposed infinitely repeated games reveal that forged messages can be suppressed by improving the payoffs for genuine messages, increasing the cost of bots, and/or reducing the payoffs for forged messages. The increasing detection probability of forged messages or decreasing misclassification probability of genuine messages also has a strong impact on the suppression of forged messages.

Chapter 4: This chapter proposes a discrete-time Markov model based on the epidemic model to accurately capture the lateral propagation of APT. To be specific, this chapter proposes a discrete-time absorbing Markov process to precisely characterize virus propagations. Conducting eigenvalue analysis and Jordan decomposition to the process, we prove that the virus extinction rate, i.e., the rate at which the Markov process converges to a virus-free absorbing state, is bounded. The bounds, depending on the infection and curing probabilities, and the minimum degree of the network topology, have closed forms. We also reveal that the minimum curing probability for a given extinction rate requirement, specified through the upper bound, is independent of the explicit size of the network. As a result, we can interpret the extinction rate requirement of a large network with that of a much

smaller one, evaluate its minimum curing requirement, and achieve simplifications with negligible loss of accuracy. Simulation results corroborate the effectiveness of the interpretation, as well as its analytical accuracy in large networks.

Chapter 5: This chapter further studies the lateral propagation process based on the research in Chapter 4. To be specific, this chapter proposes to group vertices in directed graphs based on connectivity and carries out epidemic spread analysis on the group basis, thereby substantially reducing the modeling complexity while preserving the modeling accuracy. A group-based continuous-time Markov SIS model is developed. The adjacency matrix of the network is also collapsed according to the grouping, to evaluate the Jacobian matrix of the group-based continuous-time Markov model. By adopting the mean-field approximation on the groups of nodes and links, the model complexity is significantly reduced as compared with previous topological epidemic models. An epidemic threshold is deduced based on the spectral radius of the collapsed adjacency matrix. The epidemic threshold is proved to be dependent on network structure and interdependent of the network scale. Simulation results validate the analytical epidemic threshold and confirm the asymptotical accuracy of the proposed epidemic model.

Chapter 6: This chapter analyzes the remote control traffic in APT and proposes a novel method to vectorize the network traffic for APT detection. To be specific, this chapter analyzes the features of remote control in APT and find that the HTTP-based remote control is widely used. Based on the analysis results, this chapter proposes a new feature of remote control traffic, i.e., independent access, to characterize the difference between remote control communications and normal HTTP requests. By applying the independent access feature into DNS records, we implement a novel remote control detection method and validate it on a public dataset. As a new feature of remote control traffic, its advantages and drawbacks are also analyzed.

Chapter 7: This chapter discusses future works, including propagation models, cyber-attack detection and Blockchain-based security.

Chapter 8: This chapter summarizes the research of this thesis and highlights the contributions of this thesis.

Chapter 2

Literature Review

Based on the lifecycle of APT and identified key research points in the first chapter, this chapter summarizes recent research breakthroughs on APT attack models, social network attack and defense models, propagation models and remote control models.

General APT models, used to describe the sophisticated and long-term APT behaviors, are firstly studied in Section 2.1. Such models can be used to integrate APT-related security alarms and events, summarize attack features and predict potential loss caused by APT attack.

Social attack and defense models, used to detect attacks and analyze the interaction between attackers and defenders in APT, are reviewed in Section 2.3. This section studies the latest algorithms to detect attack messages, trustworthiness and game models in social networks.

Propagation models, used to model the long-term interaction between attackers and defenders in APT attacks, are reviewed in Section 2.4. This section studies various epidemic processes, Markov models and mean-field based epidemic models.

Studies in terms of remote control models and detection are reviewed in Section 2.4. Remote control stage can be analyzed to detect APT due to the fact that the stable control link between attackers and victims are essential to APT attacks to upload intelligence and update attacks. This section studies the remote control technology used in APT and beyond. Remote control detection methods and projects are reviewed.

2.1 General APT Models

Multi-stage attack modes were extended to reconstruct APT attack scenarios from various attack events across different networks. The attack tree [36] was extended into an attack pyramid where every plane of the pyramid indicates an attack network, such as a social network and a physical network [37]. The top vertex of the pyramid, corresponding to the root node of an attack tree, represents an attack target. Nodes on different planes collect attack events and are connected by logical operations. Another tool to manage security events is the attack graph [38], which can collect and fully analyze various intelligence, e.g., network topology, vulnerabilities, and host information. Therefore, attack graphs can be used to analyze long-term and multi-stage network intrusion and restore attack scenarios. In [39], the Petri net, developed to describe discrete dynamic systems, was customized from the original triple to a six-tuple. The model defined the elemental components in APT and their connections for the purpose of rebuilding APT attack scenarios. Aiming at building fine APT attack scenarios, these models have to capture massive security events in large-scale networks and store them for a long time. The models are with the challenges of massive states across various networks and high complexity, and therefore can hardly be applied in practical analyses.

Researches were also carried out to analyze interactions between APT attackers and defenders, typically by employing the game theory [40, 41, 42]. In an APT game model, named “FLIPIT”, attackers and defenders could take their strategies and obtain the control of the public resource at any time [40]. Both of their targets were achieving the maximum of resource control while limiting the total action cost. The research evaluated the payoff of attackers and defenders where the players’ movements follow the same independent uniform distribution. Numerical results indicated that a more radical party is more likely to suppress the other one, although

pay a lot of costs. The FLIPIT model helps defenders to deploy effective defense strategies. In [41], APT attackers and defenders were able to perform attack and detection scan at customized intervals. The Nash equilibrium of the game showed that the attacker is more aggressive for a longer detection period, and conservative otherwise. The paper proposed a policy hill-climbing (PHC) based detection scheme to randomize the overall detection and a “hotbooting” algorithm to accelerate the training of PHC detection. Simulation results confirmed that the proposed strategies can achieve better data protection compared with the standard Q-learning strategy. A Colonel Blotto game was proposed to capture the process of APT attackers and defenders competing for storage resources, where the attackers and defenders had limited central processing unit (CPU) time [42]. The Nash equilibrium of the game revealed the numerical impact of the amount of data in storage devices, the number of storage devices, and the number of CPU resources on the game outcome. With the goal of improving data protection level, the paper analyzed the optimization problem of computing resource allocation based on a hill-climbing strategy and Q-learning method. In addition to the game theory, virus propagation models were also used for the APT’s attack-defense interaction modeling and the defense evaluation [43]. Similar to the virus propagation model, each node in the network can be secure or under attack. The dynamic network state was interpreted into a loss function for optimal analysis.

2.2 Social Attack and Defense Models

Techniques have been developed to detect attack/forged messages in social networks, typically by defining features. The techniques can be used by individual users or network operators to classify messages. Egele *et al.* [44] detected malicious accounts in OSNs based on the finding that malicious accounts exhibit consistent behaviors over time, e.g., time of day, source, text, topic, links in messages, direct user

interaction, and proximity. Ruan *et al.* [45] used extroversive behavioral features (i.e., first activity, activity preference, activity sequence, and action latency) and introversive behavioral features (i.e., browsing preference, visit duration, request latency, and browsing sequences) to profile malicious accounts. Chen *et al.* [46] proposed an evolutionary classification algorithm to detect time-varying statistical features of spams, e.g., the number of retweets, by learning detected and manually labeled spams. Yang *et al.* [47] analyzed twitter spammers from the viewpoint of topology and found that spammers have small local clustering coefficients, high betweenness centrality, and small bidirectional link ratios. Neighbor features, such as neighbors' followers, average neighbors' tweets, and followings to median neighbors' followers, were employed to improve the detection of malicious accounts [47]. Review-based features, such as early time frame, rate deviation, the number of first-person pronouns, and the ratio of exclamation sentences, were proposed by Shehnepoor *et al.* [48] in addition to user-based features, to analyze reviews in OSNs (e.g., feedback on a topic or a product). Other review-based features, such as the numbers of views and comments received, ratings, and favorite times, were also used to identify spammers [49]. Linguistic features, such as the structure of sentences and modifiers, were also proposed to analyze reviews [50]. Based on a finding that the majority of collected spams are generated with underlying templates, Tangram [51] divided spams into segments and extracted templates for accurate and fast spam detections.

Interactions among users have been exploited to improve the accuracy of identifying forged messages and misbehaved publishers and penalize the publishers to inhibit forged messages. A subjective trustworthiness model and an objective model were proposed in [52], where trustworthiness was evaluated based on local knowledge and global information, respectively. The trustworthiness of nodes can be judged by network administrators. For example, a trusted user was defined as the

one which has never posted any spam tweet and has posted at least five confident ham tweets [53]. Priority can be given to the trusted publishers [52, 54, 55]. Sirivianos *et al.* [56] proposed a collaborative spam filtering system, where feedback from different reporters was jointly assessed based on the trustworthiness of the reporters. Apart from manual complaints, the feedback can be automated reports, e.g., from honeypots [57].

A popular yet harsh technique of regulating misbehaved publishers was to block any forged messages and their publishers [58, 59, 60]. The solution would be too harsh and less effective in the case where genuine messages can be misclassified due to biased reviews [56] or unintentional mistakes can be made at the publishers, such as granting permission to malicious applications [53]. Another increasingly popular, soft technique of regulating the publishers' behaviors is to have service providers (or administrators) send risk alerts on forged messages without blocking the contents and their publishers, as done in Facebook [61]. Given the alerts, all subscribers can by their own decision to distrust and reject the messages, thereby reducing the payoff to the publisher of the messages and disincentivizing the publisher from misbehaving. However, the technique has yet to be appropriately analyzed in the literature, due to sophisticated interactions between the publisher, administrator, and subscribers, as well as potentially different types of coexisting subscribers.

Game theory has been employed to model the interactions in OSNs [62, 63, 64]. In a game between publishers and administrators [62], the publishers can send more spam messages to gain higher payoff, which increases conversely the successful detection of the spam messages at the administrators. The detected malicious publishers can be added into blacklists and quarantined. In [63], a Stackelberg-type game was set up to capture viral product design and customer satisfaction and optimize product adoption in OSNs. Abbass *et al.* [64] applied game theory to analyze the trustworthiness of N players in an OSN, revealing that the society with no untrust-

worthy individuals would yield the maximum wealth. However, game theory has not been used to model and analyze a more sophisticated scenario, as the one discussed in this thesis, where there can be multiple types of users with different views on (forged) messages and messages can even be misclassified by mistake.

To the best of our knowledge, none of the existing game-theoretic analyses are able to capture the increasingly popular risk alert technique in OSNs [61]. An understanding of the long-term effect of this technique on the regulation of the publisher's behaviors is important to the design and configuration of the technique though.

2.3 Propagation Models

In APTs, attackers first launch initial attacks aiming at targeted nodes and gain footholds. The attackers then attack/infect other nodes in target networks to collect more intelligence and deliver succeeding attacks. Attack techniques used in the propagation stage include network scanning, exploit exploitation and other attacks. As a kind of professional attack, APTs can be difficult to be completely removed from the targeted network [28]. The nodes in the targeted network will be in the transition of being attacked and repaired for a long time.

The virus propagation model in computer networks can be used to characterize the lateral propagation process in APT [65]. Some classic virus propagation models only support one-way state transition, such as the Susceptible-Infected (SI) model and Susceptible-Infected-Recovered (SIR) model. Once a node is infected, it cannot be transferred to the susceptible state anymore. There are also models that allow looping state transition, such as the Susceptible-Infected-Susceptible (SIS) model, where nodes can transfer between susceptible and infected states. Thus, the SIS model can be used to model the long-term interaction in APT. Based on the SIS model, this thesis interprets the lateral mobility of APTs and the ability of defenders

into the infection probability and the curing probability in the virus infection model.

The epidemic model, originally developed to characterize the spread of biological infectious diseases in [66], was first extended to computer networks in [67], where the network topology was omitted (i.e., which is equivalent to assuming a network topology of a complete graph). In the model, the infection probability of each network element solely depends on the infected population. The curing probabilities of the elements are independent and identically distributed, as those for biological diseases. Such model was used for mobile Bluetooth networks [68, 69], and wireless sensor networks [70]. Kephart et al. [71] and Vojnovic et al. [72] proposed a new “warning” state at every node, apart from the “infected” and “uninfected” states. The probability of a node switching to the “warning” state depends on the population of the warned nodes and is independent of the network topology.

Statistic topology models, e.g., the scale-free graph, were later adopted to study the statistics of computer virus spread. Pastor-Satorras et al. [73] derived the probability of a node being infected as a function of the expected number of its infected neighbors. The number was calculated, following the power-law distribution of the neighbors in scale-free graphs. The probability was used in [74] to derive the threshold required for a virus to exponentially die out. Zou et al. [75] used the scale-free graph to model the propagation of Internet email worms. In practice, different topologies, e.g., star and ring graphs, can have very different virus propagation and cure properties. Unfortunately, Li et al. [76] showed that analyses on scale-free graphs are inaccurate for specific topologies.

Markov processes were adopted to model virus propagations in specific network topologies, where every Markov state collects the binary status, “infected” or “cured”, of every node in the network [35, 77, 78]. Van Mieghem et al. [35] developed a continuous-time absorbing Markov epidemic model, where the state space is

2^N in an N -node network. The second-largest eigenvalue of the Markov transition rate matrix, which determines the speed of convergence to the absorbing virus-free state, was approximated and numerically calculated using curve fitting, due to the high complexity of eigenvalues decomposition. There were attempts to reduce the state space for specific topologies, e.g., complete and star graphs, by exploiting graph isomorphism [77, 78]. However, the complexity of the Markov epidemic model is still high and prohibitive to large network analyses.

There were efforts to decompose the Markov models to reduce complexity and improve scalability. In [35], the continuous-time 2^N -state Markov epidemic model was decoupled to N small Markov processes described by N differential equations, one per node, by taking moment closure approximations. The N small Markov processes were assumed to be uncorrelated and described by N separate differential equations. Mean-field approximations were carried out to each differential equation. Specifically, the probability of a node being infected, which depends on the number of its infected neighbors, were approximated to depending on the mean number of the infected neighbors. Chakrabarti et al. [79, 80] applied the independence assumption and mean-field approximation to discrete-time Markov epidemic models, which were later extended to analyze the propagations of email malware [81] and social network worms [82]. The model was extended to multi-layer networks [83], where a network can be visualized as several independent logical networks (each infected by a different virus), and later generalized to the case where the logic networks are correlated due to the competition among the viruses at a node [84], and the case each node can have different infection and curing rates [85]. Unfortunately, the independence assumption may not withstand in practice due to the interconnections between the nodes. As a consequence, the effective infection rate is overestimated [35, Sec. IV-A]. Moreover, the mean-field approximations can cause inaccuracy in the models, especially when N is small to medium [35].

The statistic topology models are generally based on undirected networks. However, there is an intrinsic directionality in the propagation in specific types of dynamics, e.g., infectious disease spreading [86] and information transmission [87]. Directed networks, sets of vertices and a collection of directed edges that connect pairs of ordered vertices, are useful to represent specific transmissions with intrinsic directionality in the propagation [88]. Meyers et al. [89] employed the percolation theory to predict disease transmission through semi-directed contact networks, where edges may be directed or undirected and found that the probability of an epidemic and the expected fraction of a population infected during an epidemic can be different in semi-directed networks, in contrast to the routine assumption that these two quantities are equal. Li et al. [90] defined the directionality ξ as the percentage of unidirectional links and found that the lower bound of the epidemic threshold increases with a growing ξ , implying that the directionality hinders the propagation of epidemic processes. In [91], Khanafer et al. studied the stability of an SIS N -intertwined Markov model over arbitrary directed network topologies and showed that when the basic reproduction number is greater than one, the epidemic state is locally exponentially stable, and when the network is not initialized at the disease-free state, the epidemic state is globally asymptotically stable.

2.4 Remote Control Models and Detection

Besides APT, the remote control has been widely used in botnets as a fundamental technology with two popular remote control structures, i.e., the centralized structure and the distributed structure [92]. The centralized remote control employs the master-slave structure where a single control center can send commands to many under control nodes. The centralized remote control is usually realized over popular network protocols, e.g., IRC and HTTP. In the distributed command and control structure, the controlled nodes play the same role and form a peer-to-peer network

in which the remote commands propagate.

The identification of remote control traffic is key to the detection and control of botnets. The infected nodes in the distributed command and control system behave similarly with one another, which can be used to model the control traffic and detect botnets [93]. On the other hand, the centralized remote control traffic has outstanding features on domains, information of infected nodes, time to live (TTL), etc. Anomaly detection algorithms have been widely used in the detection of remote control traffic. By combining the TCP anomaly detection and IRC anomaly detection, [94] proposed to detect IRC-based botnets. [95] found that the infected nodes in botnet share similar group behavior and applied the finding on DNS records to detect infected nodes. Many botnet detection systems have been developed and deployed on the public Internet. For example, Botsniffer [96] and BotMiner [97] revealed that relativity can be found in the remote control traffic from live data streams. Popular similarities include the same source addresses, same destination addresses, package sizes, and network speed. The two systems detect botnets by employing the features.

The remote control (or command and control) can be modeled and then detected based on time features and spatial features because the attackers need to control groups of attackers in a long period. The topology between internal hosts and external domains was used to find out potential victims [98]. Hierarchy clustering algorithms were developed to detect APT attacks and attacked internal hosts based on the connection records [99]. Based on the APT attack stages, [100] constructed a graph from DNS records for the detection of remote control centers in APT. These APT detection researches focus on the remote control stage of APT because the remote control lasts the whole APT period is possible to be detected than other attack stages.

However, the remote control in APT is different from the command and control in botnets due to the fact that APT attacks stay undercover and do not generate enormous traffic like botnets. As a result, new detection methods need to be developed based on the control behavior of the remote control in APT.

2.5 Chapter Conclusion

This chapter reviewed literature related to this thesis and laid a solid foundation for the following chapters. To be specific, this chapter summarized and compared the latest research progress about APT models, i.e., general APT models, attack-defense models in social networks, lateral propagation models and remote control models and detection. This chapter also identified the gap between research targets and existing researches and pointed out key research points of this thesis.

Chapter 3

Social Attacks Modeling and Analysis

APT attacks often start with information gathering, especially on social networks. Once an APT target is identified, a malicious payload can be constructed and then delivered to the target through social networks in order to break into the target network. Popular social network attacks include fishing attack [101], spearfishing attack [29], and waterhole attack [30]. Users in social networks are sensitive to attack messages at different levels. By collecting enough information of targeted users on social networks, attackers can build specific attack messages within the interest of attack targets to trap the targeted users. Attackers can also hire bots or collaborate with other collusive attackers [102, 103] to further improve success attack rates and evade detection and punishment. Under this background, this chapter proposes game theory based models to analyze attack-defense interactions in social networks.

3.1 Introduction

Online social network (OSN), enabling users to interact with each other through the Internet, dramatically reshapes the way people are connected and has a strong impact on the public decision-making [104]. OSNs, e.g., Facebook and Twitter, have been playing important roles even in politics and commerce [105]. Other OSNs, such as Yelp and TripAdvisor, collect reviews from consumers and can have a direct influence on product sales [106].

With no verification on user-generated content, OSNs are vulnerable to forged messages that intentionally mislead or deceive the recipients. There are a variety of

forged messages, such as email spams [107], fake views, e.g., in Amazon [108, 48], and diffusing rumors [109, 110]. The publishers of forged messages can use social bots [102] or employ water army [103] to propagate forged messages and avoid sanction. Users in OSNs are susceptible to forged messages to different extents [111] and can be sensitive to the content of the messages [112].

Behavioral and linguistic features have been used by individual subscribers to judge the genuineness and forgery of a message [44, 50, 113]. Many subscribers can feed back their judgments to help correctly classify the message, penalize the publication of forged messages, and inhibit forgery of messages. For instance, Yelp filters reviews and encourages users to report inappropriate reviews [114]. Facebook enables users to provide feedback on any posts which are potentially forged, in addition to independent third-party verification [61]. Once identified, forged posts and/or their publishers can be blocked, which could be too harsh and inadequate in the case where there is misclassification of genuine messages or unintentional mistakes of publishers [56, 53].

An increasingly popular approach to regulating the publisher's behaviors is to have service providers (or administrators) send risk alerts to all users on disputable contents without blocking the contents and their publishers, as done in Facebook [61]. However, the effect of risk alerts on the inhibition of forged messages has yet to be properly analyzed and understood. One challenge is that the publisher can interact with the subscribers over an infinite time horizon, and change its strategies over time to potentially mislead the subscribers. The interactions can be sophisticated. Another challenge is that different types of subscribers may coexist in OSNs, including followers, fans and bots, reacting distinctly to the publisher's behaviors.

The motivation of this chapter is to provide new quantitative understanding on

the effect of risk alerts on the inhibition of forged messages in the presence of multiple types of subscribers and possible miss-detection and false alarm of forged messages. We propose a new game theoretic approach for modeling and analyzing the proliferation of forged messages in OSNs, where there are a network administrator, a message publisher, and message subscribers (including followers, fans and bots). Infinitely repeated games are constructed to characterize the interactions between the publisher, administrator, and subscribers; and quantify the payoffs of the publisher, first in the absence of misclassification on genuine messages and then in the presence of misclassification. This chapter also identifies critical conditions, under which the broadcast of forged messages can be disincentivized and forged messages can be suppressed in OSNs. The contributions of the chapter can be summarized as follows.

- We propose infinitely repeated games to capture the interactions between a publisher and the administrator to suppress forged messages in OSNs;
- Critical conditions, under which the publisher can be disincentivized to send any forged messages, are identified in the absence and presence of misclassification on genuine messages;
- Closed-form expressions are established for the maximum number of forged messages of a malicious publisher in the absence and presence of misclassification on genuine messages.

Validated by numerical results, our analysis indicates that forged messages can be suppressed by improving the payoffs for genuine messages, increasing the cost of bots, and/or reducing the payoffs for forged messages. The increasing detection probability of forged messages or decreasing misclassification probability of genuine messages can also benefit the game theoretic suppression of forged messages.

The rest of this chapter is organized as follows. The proposed social network model and infinitely repeated games of forged messages are presented in Section 3.2. Sufficient conditions, under which the publisher is disincentivized from publishing forged messages, are established, and closed-form expressions for the maximum number of forged messages which can be published before the publisher is disincentivized are derived in Section 3.3. The proposed model, as well as the policies to suppress forged messages, are numerically validated in Section 3.4, followed by conclusions in Section 3.5.

Notations used in the chapter are as listed in Table 3.1.

Table 3.1 : Notations Used in Chapter 3

Notation	Description
\mathcal{P}	A message publisher
\mathcal{A}	The network administrator
N_r	The number of followers
N_n	The number of fans
N_t	The number of hired bots
p_1	The miss-detection probability on a forged message
p_2	The POD on a genuine message
q_1	The probability on which a forged message is misjudged
q_1	The probability on which a genuine message is correctly deduced
C_1	The unit payoff from a genuine message
C_2	The unit payoff from a forged message
C_3	The cost of a bot

3.2 Network Model

We consider subscription services of OSNs, as shown in Fig. 3.1, where a publisher, denoted by \mathcal{P} , publishes messages to its subscribers. \mathcal{P} can either publish genuine messages, referred to as the *benign* strategy; or forged messages, referred to as the *malicious* strategy. Genuine messages can benefit the OSN overall, while forged messages can potentially increase \mathcal{P} 's payoff. Typical forged messages include rumor, commercial advertisement, and biased reviews. We consider the case that \mathcal{P} can publish an infinite number of messages. At every round (one message per round), \mathcal{P} takes either the *benign* or *malicious* strategy.

The subscribers consist of N_r followers which feed back objective judgements of \mathcal{P} 's messages to an administrator \mathcal{A} , and N_n fans which provide subjective (persistently positive) feedback in favor of \mathcal{P} . Let p_1 denote the miss-detection probability at which a follower incorrectly gives positive feedback on a forged message; and p_2 denote the probability of detection (POD) at which a follower correctly gives positive feedback on a genuine message. $(1 - p_2)$ is the probability of false alarm. The followers and fans feed back independently to \mathcal{A} . Moreover, \mathcal{P} can also hire N_t bots which always give positive feedback to mislead \mathcal{A} at a cost of \mathcal{P} .

Based on the feedback received, the administrator \mathcal{A} can take either a *trust* or *distrust* strategy to encourage \mathcal{P} to publish genuine messages or disincentivize \mathcal{P} from sending forged messages, respectively. When taking the *trust* strategy, \mathcal{A} confirms the genuineness of a message and advises all the subscribers to accept the message. When taking the *distrust* strategy, \mathcal{A} sends risk alerts to notify the subscribers of a potentially forged message. The followers follow \mathcal{A} 's advices, while the fans always trust \mathcal{P} (at no cost of \mathcal{P} , as opposed to the bots). The *distrust* strategy reduces \mathcal{P} 's payoff, and can penalize and suppress forged messages.

The payoff that \mathcal{P} can receive per message is reasonably assumed to be linear to the number of subscribers who trust the message. The payoff is contributed by

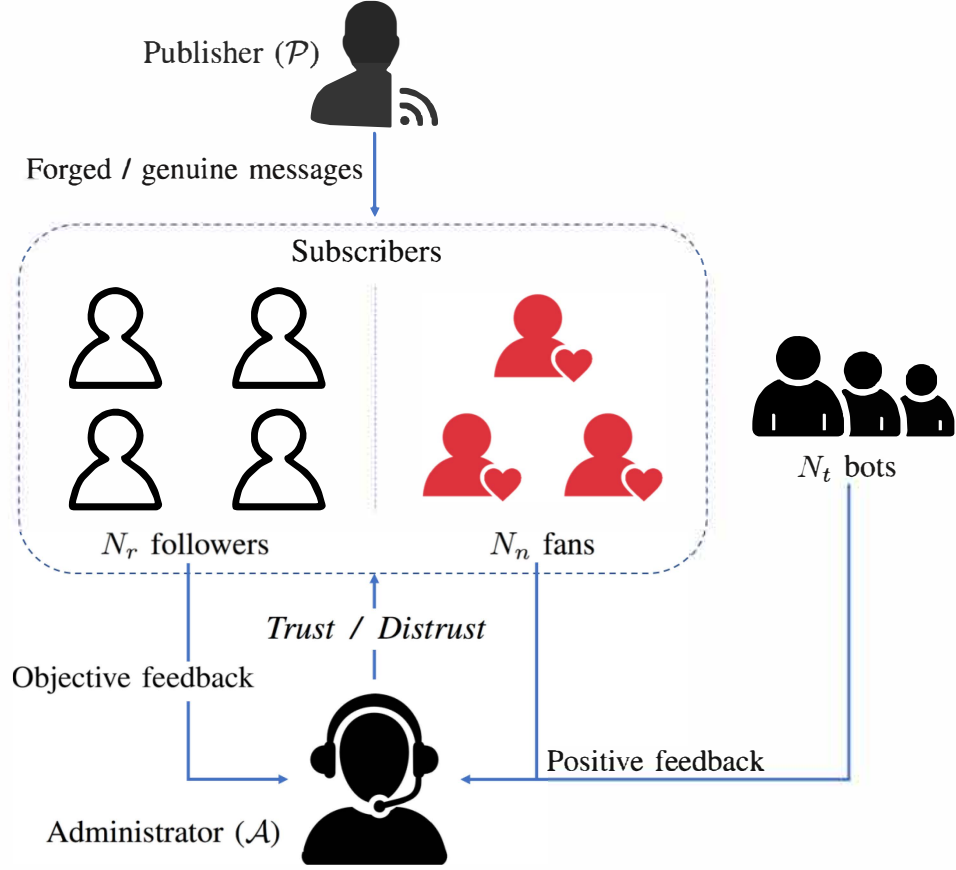


Figure 3.1 : An illustration on the subscription services in OSNs, where there are a network administrator \mathcal{A} , publisher \mathcal{P} , and subscribers (including followers, fans and bots). \mathcal{P} can publish either genuine or forged messages. The subscribers check the messages and give their feedback to \mathcal{A} . The bots are hired by \mathcal{P} to always give positive feedback to \mathcal{A} . Based on the feedback from the subscribers, \mathcal{A} can take either a *trust* or *distrust* strategy against \mathcal{P} .

N_n fans and/or N_r followers. Let $C_1 > 0$ (or $C_2 > 0$) denote the unit payoff (per message per subscriber) for \mathcal{P} to publish a genuine (or forged) message. If $C_1 \geq C_2$ (i.e., the payoff from a genuine message is higher than that from a forged message), \mathcal{P} has no incentive to publish forged messages. We are particularly interested in the case where $C_1 < C_2$ and \mathcal{P} has incentive to publish forged messages, as will be analyzed in this chapter. Let $C_3 > 0$ denote the cost of a bot to \mathcal{P} . The bots increase

the cost of \mathcal{P} , but may help increase \mathcal{P} 's payoff by misleading \mathcal{A} and reducing the probability of \mathcal{P} 's misbehaviors being detected.

We take a Facebook group for an example [115]. The group manager plays the role of the administrator \mathcal{A} which can verify every post of a publisher \mathcal{P} and alert the group members of disputable posts by using a Pin To Top announcement. As the fans (or friends) of \mathcal{P} , some group members choose to ignore \mathcal{A} 's advices and always *trust* and accept the posts. As the followers of \mathcal{P} , other group members accept \mathcal{A} 's advices, i.e., *distrust* and reject disputable posts.

By default, \mathcal{A} takes the *trust* strategy, and updates the strategy on each of \mathcal{P} 's messages based on the detection results. We set the probability that \mathcal{A} classifies a message to be genuine is equal to the ratio of positive feedback. Once a forged message is detected, \mathcal{A} takes the *distrust* strategy against \mathcal{P} and alerts the subscribers for a period of time (or in other words, \mathcal{P} is placed on a “good behavior bond” and can still publish messages during the period). The duration of the alerting period can vary under different punishment policies. For example, \mathcal{P} can be declared to be untrusted for a single round, permanently, or with exponentially increasing alert durations [116, 117]. Our analysis focuses on the exponentially increasing durations which double, every time a forged message is detected at \mathcal{A} . Thus, the alert duration in response to the k -th detected forged message of \mathcal{P} is 2^{k-1} (rounds), during which \mathcal{P} may still choose to publish another forged message at the potential consequence of a further doubled alert duration following the current one [117]. \mathcal{A} takes the *trust* strategy after the duration completes. \mathcal{P} can become aware of the strategy that \mathcal{A} takes, and adjust its own behavior accordingly within and after the duration to maximize its payoff. However, our analysis can be readily applied to the other aforementioned punishment policies, i.e., distrust for a single round or permanently.

3.3 Games of Forged Messages

In this section, we identify the critical conditions to suppress forged messages, where the interaction between \mathcal{P} and \mathcal{A} is modeled as an infinitely repeated game. We first consider a misclassification-free repeated game where genuine messages can be always correctly deduced. Sufficient conditions are established under which \mathcal{P} only publishes forged messages, or is disincentivized from publishing any forged messages. Then, we extend to more sophisticated infinitely repeated games with possible misclassification on genuine messages. The conditions, under which forged messages are disincentivized, are dependent on the cost of messages, as well as the classification results.

3.3.1 Misclassification-free Infinitely Repeated Game

We first consider a misclassification-free game, where genuine messages are always correctly classified, i.e., $p_2 = 1$. \mathcal{P} hires bots only when publishing forged messages. The payoff matrix of every message can be given by Table 3.2, where the payoffs for \mathcal{P} are listed under the specific strategy combinations. Bots are hired with the cost of C_3N_t , only when forged messages are published.

Table 3.2 : The payoff matrix to \mathcal{P} per round in the case of misclassification-free game/situations

		\mathcal{A}	
		<i>Trust</i>	<i>Distrust</i>
\mathcal{P}	<i>Benign</i>	$C_1(N_r + N_n)$	C_1N_n
	<i>Malicious</i>	$C_2(N_r + N_n) - C_3N_t$	$C_2N_n - C_3N_t$

In the case of a forged message, \mathcal{A} can collect $(N_r + N_n + N_t)$ pieces of feedback in total, $p_1N_r + N_n + N_t$ of which are positive. As a result, a forged message can be

misjudged to be genuine with probability $q_1 = \frac{p_1 N_r + N_n + N_t}{N_r + N_n + N_t}$, and correctly detected to be forged with probability $(1 - q_1)$.

Theorem 1. *In the misclassification-free infinitely repeated game, under the condition of*

$$C_2 N_n - C_3 N_t - C_1 (N_r + N_n) > 0, \quad (3.1)$$

\mathcal{P} always publishes forged messages for a higher payoff than it publishes genuine messages. \mathcal{A} cannot suppress forged messages by taking the distrust punishment. (3.1) is a sufficient condition of \mathcal{P} only publishing forged messages.

Under the condition of

$$(N_r + N_n)C_2 - ((2 - q_1)N_r + N_n)C_1 - C_3 N_t < 0, \quad (3.2)$$

\mathcal{P} has no incentive to publish forged messages; (3.2) is a sufficient condition of \mathcal{P} not publishing forged messages.

Proof. The condition (3.1) can be proved by letting the minimum payoff from a forged message be greater than the maximum payoff from a genuine message. If \mathcal{P} always takes the *malicious* strategy, it can at least get the payoff of $(C_2 N_n - C_3 N_t)$ per round. On the other hand, if \mathcal{P} always takes the *benign* strategy, it can get at most $C_1 (N_r + N_n)$ payoff per round. As a result, if $C_2 N_n - C_3 N_t - C_1 (N_r + N_n) > 0$, the better strategy is *malicious* for \mathcal{P} although it can be punished. In this case, \mathcal{A} cannot suppress forged messages by taking the *distrust* punishment.

The condition (3.2) can be proved by comparing the cumulative payoffs of a malicious publisher and a benign punisher over a sufficiently long period of time. Let τ_1 denote the cumulative rounds of punishment on the malicious publisher that

publishes f forged messages. τ_1 can be given by

$$\tau_1 = \sum_{k=0}^f (2^k - 1) \binom{f}{k} (1 - q_1)^k q_1^{f-k} \quad (3.3a)$$

$$= \sum_{k=0}^f \binom{f}{k} (2 - 2q_1)^k (q_1)^{f-k} - \sum_{k=0}^f \binom{f}{k} (1 - q_1)^k q_1^{f-k} \quad (3.3b)$$

$$= (2 - 2q_1 + q_1)^f - (1 - q_1 + q_1)^f \quad (3.3c)$$

$$= (2 - q_1)^f - 1, \quad (3.3d)$$

where (3.3a) is because k forged messages are detected with probability $\binom{f}{k}(1 - q_1)^k q_1^{f-k}$ in the total f forged messages. \mathcal{P} is to be punished for $\sum_{l=1}^k 2^{l-1} = (2^k - 1)$ rounds in total when its forged messages are detected k times. (3.3c) is obtained based on the Binomial theorem [118].

Let t denote the instant when the publication of f forged messages and the τ_1 corresponding punishments have ended. Here, $t \geq (f + \tau_1)$. Note that \mathcal{P} can publish either forged or genuine messages during the *distrust* period, resulting in different cumulative payoffs. If \mathcal{P} publishes genuine messages during the period and publishes forged messages only when the punishments are over, the cumulative payoff in t rounds, denoted by π_{a1} , can be given by

$$\pi_{a1} = f(C_2(N_r + N_n) - C_3 N_t) + \tau_1 C_1 N_n + (t - \tau_1 - f) C_1 (N_r + N_n), \quad (3.4)$$

where the payoff of each round is based on the payoff matrix in Table 3.2. The payoffs from f forged messages are $f(C_2(N_r + N_n) - C_3 N_t)$. \mathcal{P} with f forged messages is punished for τ_1 rounds on average, as given in (3.3), and can gain the payoff of $\tau_1 C_1 N_n$ during the punishment. The payoff of the remaining $(t - f - \tau_1)$ rounds is $(t - f - \tau_1) C_1 (N_r + N_n)$ where \mathcal{P} publishes genuine messages and \mathcal{A} uses the default *trust* strategy.

If the malicious \mathcal{P} publishes δ , $\delta > 0$, forged messages during the punishment and $(f - \delta)$ forged messages when it is not punished, the cumulative payoff, denoted

by π'_{a1} , can be given by

$$\begin{aligned} \pi'_{a1} = & (f - \delta)(C_2(N_r + N_n) - C_3N_t) + \delta(C_2N_n - C_3N_t) + (\tau_1 - \delta)C_1N_n \\ & + (t - \tau_1 - f + \delta)C_1(N_r + N_n). \end{aligned} \quad (3.5)$$

We have $\pi_{a1} - \pi'_{a1} = \delta(C_2 - C_1)N_r > 0$. As a result, the best strategy for \mathcal{P} is to publish genuine messages in the punishments and publish forged messages only when the punishments are over. The maximum payoff is given as π_{a1} in (3.4).

In the case that \mathcal{P} always publishes genuine messages, \mathcal{A} takes *trust* in return. The cumulative payoff of a benign \mathcal{P} in t rounds, denoted by π_{b1} , can be given by

$$\pi_{b1} = tC_1(N_r + N_n), \quad (3.6)$$

where $C_1(N_r + N_n)$ is the payoff per round in Table 3.2.

Let $g_1(f)$ denote the gap of payoff (referred to as “extra payoff”) between a malicious behavior of \mathcal{P} with f forged messages and a benign behavior, i.e.,

$$\begin{aligned} g_1(f) = & \pi_{a1} - \pi_{b1} \\ = & f(C_2(N_r + N_n) - C_3N_t) + \tau_1C_1N_n - (\tau_1 + f)C_1(N_r + N_n) \\ = & f((C_2 - C_1)(N_r + N_n) - C_3N_t) - C_1N_r((2 - q_1)^f - 1). \end{aligned} \quad (3.7)$$

Here, t is suppressed. In other words, the payoff gap is dependent of the number of forged messages and unaffected by the order of forged and genuine messages.

In the case that $g_1(1) < 0$, \mathcal{P} cannot get a higher payoff even if it publishes a single forged message. This is because $g_1(0) = 0$ and $g_1(f)$ firstly increases and then decreases, as will be proved in Theorem 2; see (3.10). The sufficient condition of \mathcal{P} not publishing forged messages can be obtained. \square

Theorem 2. \mathcal{P} gains the maximum extra payoff by publishing $\lfloor x_{m1} \rfloor$ or $\lceil x_{m1} \rceil$ forged messages. The malicious \mathcal{P} has incentive to publish less than $\lceil x_{u1} \rceil$ forged messages.

x_{m_1} and x_{u_1} can be given by

$$\begin{aligned} x_{m_1} &= -\log_{2-q_1}(\lambda_1 \ln(2-q_1)); \\ x_{u_1} &= -\frac{W_{-1}(-\lambda_1(2-q_1)^{-\lambda_1} \ln(2-q_1))}{\ln(2-q_1)} - \lambda_1; \\ \lambda_1 &= \frac{C_1 N_r}{(C_2 - C_1)(N_r + N_n) - C_3 N_t}, \end{aligned} \quad (3.8)$$

where $\lfloor \cdot \rfloor$ stands for flooring, and $\lceil \cdot \rceil$ stands for ceiling.

Proof. This theorem can be proved by relaxing the discrete function $g_1(f)$ in (3.7) to be a continuous function, denoted by $\tilde{g}_1(x)$, as given by

$$\tilde{g}_1(x) = ((C_2 - C_1)(N_r + N_n) - C_3 N_t)x - C_1 N_r ((2 - q_1)^x - 1), \quad x \geq 0. \quad (3.9)$$

The derivative of $\tilde{g}_1(x)$ can be given by

$$\frac{d\tilde{g}_1(x)}{dx} = (C_2 - C_1)(N_r + N_n) - C_3 N_t - (C_1 N_r \ln(2 - q_1)) (2 - q_1)^x. \quad (3.10)$$

As a result, $\tilde{g}_1(x)$ is a monotonically increasing function when $x < x_{m_1}$ and a monotonically decreasing function when $x > x_{m_1}$, where x_{m_1} can be given by

$$\begin{aligned} x_{m_1} &= -\log_{2-q_1}(\lambda_1 \ln(2-q_1)); \\ \lambda_1 &= \frac{C_1 N_r}{(C_2 - C_1)(N_r + N_n) - C_3 N_t}. \end{aligned} \quad (3.11)$$

This is achieved by letting $\frac{d\tilde{g}_1(x)}{dx} = 0$. Only an integer number of messages can be published. The malicious \mathcal{P} gains the maximum extra payoff, i.e., $\max(g_1(\lfloor x_{m_1} \rfloor), g_1(\lceil x_{m_1} \rceil))$, by publishing forged messages.

The publisher \mathcal{P} has incentive to publish less than $\lceil x_{u_1} \rceil$ forged messages, where

$$x_{u_1} = -\frac{W_{-1}(-\lambda_1(2-q_1)^{-\lambda_1} \ln(2-q_1))}{\ln(2-q_1)} - \lambda_1. \quad (3.12)$$

This is achieved by applying the Lambert's W Function [119] i.e., $W(\cdot)$, to $\tilde{g}_1(x) = 0$, where $-\frac{1}{e} \leq -\lambda_1(2-q_1)^{-\lambda_1} \ln(2-q_1) \leq 0$ is within the domain of Lambert W

function. By letting $h(\lambda_1) = \lambda_1(2 - q_1)^{-\lambda_1} \ln(2 - q_1)$, $\lambda_1 > 0$, we have

$$0 \leq h(\lambda_1) \leq h\left(\frac{1}{\ln(2 - q_1)}\right) = \frac{1}{e}. \quad (3.13)$$

Here, $h(\lambda_1) \geq 0$ because every item is no less than 0. The maximum value of $h(\lambda_1)$ is achieved when $\lambda_1 = \frac{1}{\ln(2 - q_1)}$. This can be obtained by letting $\frac{d \ln(h(\lambda_1))}{d \lambda_1} = 0$. $\ln(h(\lambda_1))$ is given by

$$\begin{aligned} \ln(h(\lambda_1)) &= \ln(\lambda_1(2 - q_1)^{-\lambda_1} \ln(2 - q_1)) \\ &= \ln(\lambda_1) - \lambda_1 \ln(2 - q_1) + \ln(\ln(2 - q_1)). \end{aligned} \quad (3.14)$$

Therefore, its derivative is given by

$$\frac{d \ln(h(\lambda_1))}{d \lambda_1} = \frac{1}{\lambda_1} - \ln(2 - q_1). \quad (3.15)$$

By letting $h(\lambda_1) = \lambda_1(2 - q_1)^{-\lambda_1} \ln(2 - q_1)$, $\lambda_1 > 0$, we have

$$0 \leq h(\lambda_1) \leq h\left(\frac{1}{\ln(2 - q_1)}\right) = \frac{1}{e}. \quad (3.16)$$

Here, $h(\lambda_1) \geq 0$ because every item is no less than 0. The maximum value of $h(\lambda_1)$ is achieved when $\lambda_1 = \frac{1}{\ln(2 - q_1)}$. This can be obtained by letting $\frac{d \ln(h(\lambda_1))}{d \lambda_1} = 0$. $\ln(h(\lambda_1))$ is given by

$$\begin{aligned} \ln(h(\lambda_1)) &= \ln(\lambda_1(2 - q_1)^{-\lambda_1} \ln(2 - q_1)) \\ &= \ln(\lambda_1) - \lambda_1 \ln(2 - q_1) + \ln(\ln(2 - q_1)). \end{aligned} \quad (3.17)$$

Therefore, its derivative is given by

$$\frac{d \ln(h(\lambda_1))}{d \lambda_1} = \frac{1}{\lambda_1} - \ln(2 - q_1). \quad (3.18)$$

The payoff of a malicious \mathcal{P} is no more than the payoff of a benign \mathcal{P} in the case that $f \geq \lceil x_{u_1} \rceil$ forged messages are published, i.e., $g_1(f) = \tilde{g}_1(f) \leq 0$, and $f \geq \lceil x_{u_1} \rceil$. \square

3.3.2 Infinitely Repeated Game with Message Misclassification

In a more general case that the OSN subscribers may provide incorrect feedback or comments on a genuine message due to personal opinions, the administrator can be misled and misjudge a genuine message to be a forged one. To avoid misclassification of genuine messages and reduce the detection of forged messages, \mathcal{P} hires bots for positive feedback and pays the cost of C_3N_t when publishing genuine and forged messages. As a result, the payoff matrix of a single message can be given by Table 3.3.

Table 3.3 : The payoff matrix of \mathcal{P} per round in the game/situations with misclassification

		\mathcal{A}	
		<i>Trust</i>	<i>Distrust</i>
\mathcal{P}	<i>Benign</i>	$C_1(N_r + N_n) - C_3N_t$	$C_1N_n - C_3N_t$
	<i>Malicious</i>	$C_2(N_r + N_n) - C_3N_t$	$C_2N_n - C_3N_t$

In the infinitely repeated game, every genuine message is classified as a genuine message with probability $q_2 = \frac{p_2N_r+N_n+N_t}{N_r+N_n+N_t}$. The detection probability of a forged message is $(1 - q_1)$ as given in Section 3.3.1. The confusion matrix, showing the detection and misclassification probabilities, is given by Table 3.4.

Table 3.4 : The confusion matrix for detection results

		Classification result	
		Genuine	Forged
Message	Genuine	q_2	$1 - q_2$
	Forged	q_1	$1 - q_1$

We assume that \mathcal{P} can become aware of incorrect punishments after one round, and complain to \mathcal{A} . \mathcal{A} rectifies the misclassifications in response to the complaints by stopping the punishment. Meanwhile, \mathcal{A} keeps the current punishment/alert duration for the next punishment instead of doubling it. In this way, the incorrect punishments due to misclassifications remain a single round and do not change the punishment duration of the coming forged messages. As a result, the duration of punishments on detected forged messages only depends on the number of detected forged messages and is independent of the number of genuine messages no matter whether they are misclassified or not. The malicious \mathcal{P} does not complain about the correct punishments for its publication of forged messages.

Theorem 3. *Under the condition of*

$$\frac{(1 - p_2)N_r^2 N_t}{(N_r + N_n + N_t)(N_r + N_n)} C_1 - C_3 N_t > 0, \quad (3.19)$$

\mathcal{P} hires bots for a higher payoff than it does not. (3.19) is a sufficient condition of \mathcal{P} hiring bots.

Under the condition of

$$C_2 N_n - C_1 (q_2 N_r + N_n) > 0, \quad (3.20)$$

\mathcal{P} always publishes forged messages for a higher payoff than it publishes genuine messages. \mathcal{A} cannot suppress forged messages by taking the distrust punishment. (3.20) is a sufficient condition of \mathcal{P} only publishing forged messages.

Under the condition of

$$C_2 (N_r + N_n) - C_1 ((q_1 + q_2 - 1)N_r + N_n) < 0, \quad (3.21)$$

\mathcal{P} has no incentive to publish forged messages; (3.21) is the sufficient condition of \mathcal{P} not publishing forged messages.

Proof. The condition (3.19) can be proved by comparing the payoffs from genuine messages with and without bots. In the case that a benign \mathcal{P} does not hire bots and always publishes genuine messages, \mathcal{A} can collect $(N_r + N_n)$ pieces of feedback, $p_2 N_r + N_n$ of which are positive. Thus, a genuine message is correctly classified with probability $q_3 = \frac{p_2 N_r + N_n}{N_r + N_n}$ and misclassified with probability $(1 - q_3)$. On the other hand, the punishment due to misclassification can be rectified in one round. As a result, the benign \mathcal{P} is punished with probability $(1 - q_3)$ per round. Its payoff per round, denoted by β_{b1} , can be given by

$$\begin{aligned}\beta_{b1} &= q_3 C_1(N_r + N_n) + (1 - q_3) C_1 N_n \\ &= C_1(q_3 N_r + N_n).\end{aligned}\tag{3.22}$$

Likewise, \mathcal{P} can be punished with probability $(1 - q_2)$ per round, even if it hires N_t bots and persistently publishes genuine messages, due to the misclassification of genuine messages. Its payoff per round, denoted by β_{b2} , can be given by

$$\begin{aligned}\beta_{b2} &= q_2(C_1(N_r + N_n) - C_3 N_t) + (1 - q_2)(C_1 N_n - C_3 N_t) \\ &= C_1(q_2 N_r + N_n) - C_3 N_t.\end{aligned}\tag{3.23}$$

By letting $\beta_{b2} > \beta_{b1}$, the condition under which \mathcal{P} does not hire any bot is proved.

Likewise, the condition (3.20) can be proved by letting $C_2 N_n - C_3 N_t > \beta_{b2}$, where $C_2 N_n - C_3 N_t$ is the minimum payoff from a forged message of \mathcal{P} , as given in Table 3.3.

The condition (3.21) can be proved by comparing the cumulative payoffs of a malicious publisher and a benign punisher over the same period of time. If a malicious \mathcal{P} publishes f forged messages, it is first punished for $\tau_1 = ((2 - q_1)^f - 1)$ rounds, as proved by (3.3). Note that during the punishment, \mathcal{P} can publish τ_1 number of genuine messages and be punished $(1 - q_2)\tau_1$ rounds again due to misclassifications. As a result, \mathcal{P} is able to publish τ_2 number of genuine messages during punishment

in total, where $\tau_2 = \sum_{i=0}^{\infty} \tau_1(1 - q_2)^i = \frac{\tau_1}{q_2}$. Thus, the payoff of the malicious \mathcal{P} publishing f forged messages in $f + \tau_2$ rounds, denoted by π_{a2} , can be given by

$$\pi_{a2} = (C_2(N_r + N_n) - C_3N_t)f + (C_1N_n - C_3N_t)\frac{(2 - q_1)^f - 1}{q_2}. \quad (3.24)$$

In the case that \mathcal{P} always publishes genuine messages, its cumulative payoff of $(f + \tau_2)$ rounds, denoted by π_{b2} , can be given by

$$\pi_{b2} = \beta_{b2}(f + \tau_2). \quad (3.25)$$

Let $g_2(f)$ denote the gap between the payoffs (extra payoff) of a malicious behavior of \mathcal{P} with f forged messages and a benign behavior, i.e.,

$$\begin{aligned} g_2(f) &= \pi_{a2} - \pi_{b2} \\ &= (C_2(N_r + N_n) - C_3N_t)f + (C_1N_n - C_3N_t)\frac{(2 - q_1)^f - 1}{q_2} \\ &\quad - (C_1(q_2N_r + N_n) - C_3N_t)(f + \tau_2) \\ &= f(C_2(N_r + N_n) - C_1(q_2N_r + N_n)) - C_1N_r(2 - q_1)^f + C_1N_r. \end{aligned} \quad (3.26)$$

The condition (3.21) can be proved by letting $g_2(1) < 0$. □

Theorem 4. *The publisher \mathcal{P} gains the maximum extra payoff by publishing $\lfloor x_{m_2} \rfloor$ or $\lceil x_{m_2} \rceil$ forged messages. The malicious \mathcal{P} has incentive to publish less than $\lceil x_{u_2} \rceil$ forged messages. x_{m_2} and x_{u_2} can be given by*

$$\begin{aligned} x_{m_2} &= -\log_{2-q_1}(\lambda_2 \ln(2 - q_1)); \\ x_{u_2} &= -\frac{W_{-1}(-\lambda_2(2 - q_1)^{-\lambda_2} \ln(2 - q_1))}{\ln(2 - q_1)} - \lambda_2; \\ \lambda_2 &= \frac{C_1N_r}{C_2(N_r + N_n) - C_1(q_2N_r + N_n)}. \end{aligned} \quad (3.27)$$

Proof. This theorem can be proved in the same way as Theorem 2 by replacing $g_1(f)$ with $g_2(f)$. Therefore, the proof is suppressed for brevity. □

3.4 Numerical results

Fig. 3.2 demonstrates 50 rounds of the proposed infinitely repeated games in the misclassification-free case, where $p_1 = 0.03$, $q_1 = \frac{p_1 N_r + N_n + N_t}{N_r + N_n + N_t} = 0.2$, $C_3 = 10$ and C_2 is set to be 10, 50 and 120. The upper part of the figure plots the cumulative payoffs of \mathcal{P} , where the solid line provides the payoff of a benign \mathcal{P} for reference purpose. In the case of $C_2 = 120$, the solid line with marker “+” shows the payoff of \mathcal{P} when \mathcal{A} and \mathcal{P} always take the *distrust* and *malicious* strategies, respectively. We can see that the malicious \mathcal{P} can receive a higher payoff than a benign one, even after being declared publicly to be distrusted. \mathcal{A} cannot suppress forged messages, as stated in Theorem 1. In the case of $C_2 = 10$, \mathcal{P} has no incentive to publish any forged messages at all. In the case of $C_2 = 50$, the payoffs of \mathcal{P} are compared between two cases: (a) Case 1: \mathcal{P} continues to publish forged message once an alert duration ends; and (b) Case 2: \mathcal{P} publishes forged messages only in the very beginning. In both cases, 6 forged messages are published, and the rest 44 messages are genuine. We can see that \mathcal{P} can gain a higher payoff if it sends forged messages when \mathcal{A} takes the default *trust* strategy, as stated in Theorem 1.

The lower part of Fig. 3.2 shows the corresponding number of published forged messages of \mathcal{P} with the increase of rounds, where the two aforementioned cases of \mathcal{P} ’s behaviors, i.e., Cases 1 and 2, are plotted. We can see that \mathcal{P} can obtain the maximum payoff gap when it publishes the fifth forged messages at the 13th round. The number of forged messages is achieved by first identifying the largest payoff gap between the *malicious* and *benign* strategies of \mathcal{P} in the upper part of Fig. 3.2, then mapping the corresponding round number onto the lower part of the figure (as illustrated by the dashed line), and finally checking the number of forged messages. We also see that the exponentially increasing alert durations are effective to suppress forged messages, and \mathcal{P} is disincentivized from further publishing any

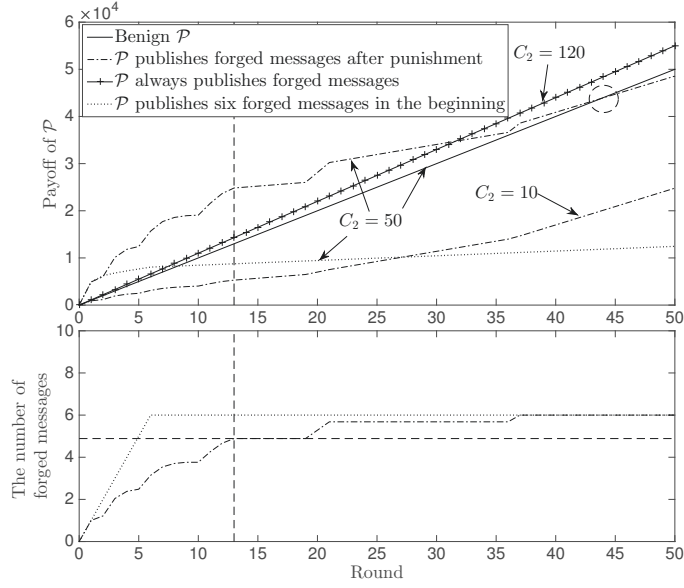
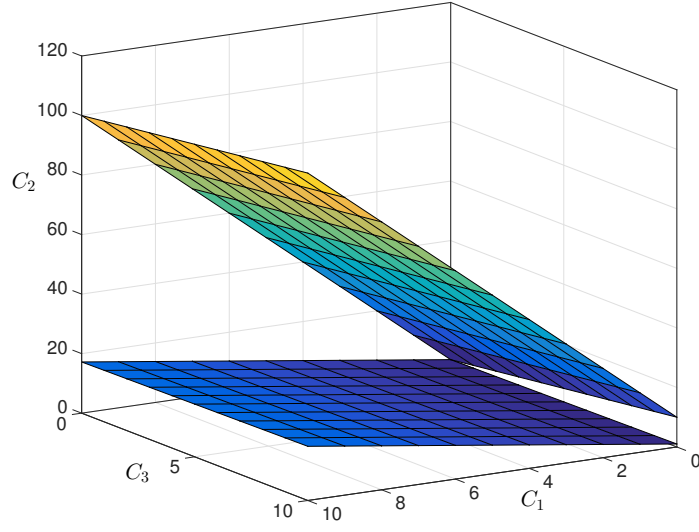


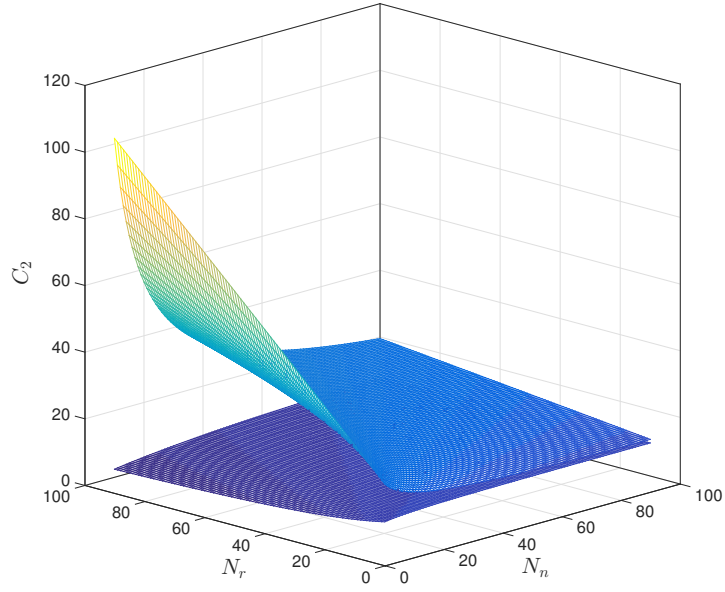
Figure 3.2 : The cumulative payoff and the number of forged messages of \mathcal{P} , where $p_1 = 0.03$, $N_r = 90$, $N_n = 10$, $N_t = 10$, $C_1 = 10$, and $C_3 = 10$. $C_2 = 10, 50$ and 120 . The payoff is the average of 5,000 independent simulations.

forged messages beyond the five forged messages. The payoff from the misbehaviors of \mathcal{P} in Case 1 with six forged messages and 38 genuine messages, is equal to the payoff of 44 genuine messages, as highlighted by a circle. The payoff from the misbehaviours can be outrun by that from genuine messages, when the number of published messages is more than 44.

Figs. 3.3(a) and 3.3(b) demonstrate Theorems 1 and 3, respectively. In the case that C_2 is between the two surfaces in each of the figures, \mathcal{P} has incentive to publish forged messages for extra payoff, but can be disincentivized by the *distrust* strategy from \mathcal{A} . Above the upper surface, \mathcal{P} is expected to always publish forged messages for high payoffs, as stated in Theorems 1 and 3. Below the lower surface, \mathcal{P} has no incentive to publish any forged messages at all, as stated in Theorems 1 and 3. From both figures, we can see that C_1 has a stronger impact on \mathcal{P} 's selection of its



(a) C_2 with the growth of C_1 and C_3 , where the upper and lower surfaces are obtained with (3.1) and (3.2) in Theorem 1, respectively. $N_r = 90$, $N_n = N_t = 10$ and $q_1 = 0.2$.



(b) C_2 with the growth of N_n and N_r , where the upper and lower surfaces are obtained with (3.20) and (3.21) in Theorem 3, respectively. $N_t = 10$, $p_1 = 0.1$, $p_2 = 0.9$, $C_1 = 10$, and $C_3 = 0$.

Figure 3.3 : The visualization of Theorems 1 and 3.

strategies than C_3 , in the case of $N_t \ll N_r + N_n$. We also see there is a peak on the upper surface where $N_r = 100$ and $N_n = 0$. In other words, \mathcal{P} would not infinitely publish forged messages when it does not have fans.

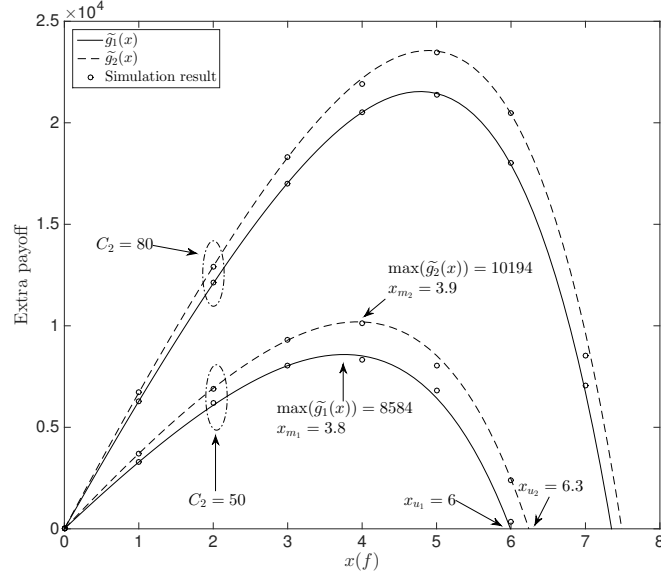


Figure 3.4 : The auxiliary continuous functions $\tilde{g}_i(x)$, indicating the extra payoffs from forged messages with the growth of x , where $N_r = 90$, $N_n = 10$, $N_t = 10$, $C_1 = 10$, $C_3 = 5$, $p_1 = 0.1$, and $p_2 = 0.5$. Two C_2 , i.e., 50 and 80, are considered. Every dot is the average extra payoff of 1,000 independent simulations.

Fig. 3.4 validates Theorems 2 and 4 by plotting the auxiliary continuous functions, i.e., $\tilde{g}_1(x)$ and $\tilde{g}_2(x)$. We find that the extra payoff of \mathcal{P} , which is the payoff gap at the end of an alert duration between *malicious* and *benign* strategies that \mathcal{P} takes (as defined in Section 3.3), first grows and then decreases rapidly. The extra payoff is upper-bounded, e.g., $\max(\tilde{g}_1(x)) = 8,584$ when $x_{m_1} = 3.8$, as shown in the figure. As revealed in Theorem 2, \mathcal{P} only has incentive to broadcast a limited number of forged messages, e.g., $x_{u_1} = 6$, for extra payoff. In the case with message misclassification, $\max(\tilde{g}_2(x)) = 10,194$ when $x_{m_2} = 3.9$, and $x_{u_2} = 6.3$, and Theorem 4 is validated. We can also see that the maximum extra payoff and

the maximum forged messages of \mathcal{P} increase with C_2 . Moreover, \mathcal{P} can gain higher payoffs in the case where the genuine messages can be misclassified, as shown by the dash curves.

We proceed to evaluate the impact of different parameters on the suppression of forged messages. Particularly, we will show that reducing the misclassification probability of genuine messages, i.e., decreasing $(1 - p_2)$, is helpful to suppress forged messages, and to incentivize the publication of genuine messages, as will be shown in Fig. 3.5. Alternatively, \mathcal{A} can encourage \mathcal{P} to publish genuine messages by improving the detection rate of forged messages, i.e., $(1 - p_1)$, as will be shown in Fig. 3.6; or increasing the reward for genuine messages, i.e., C_1 , as will be shown in Fig. 3.7.

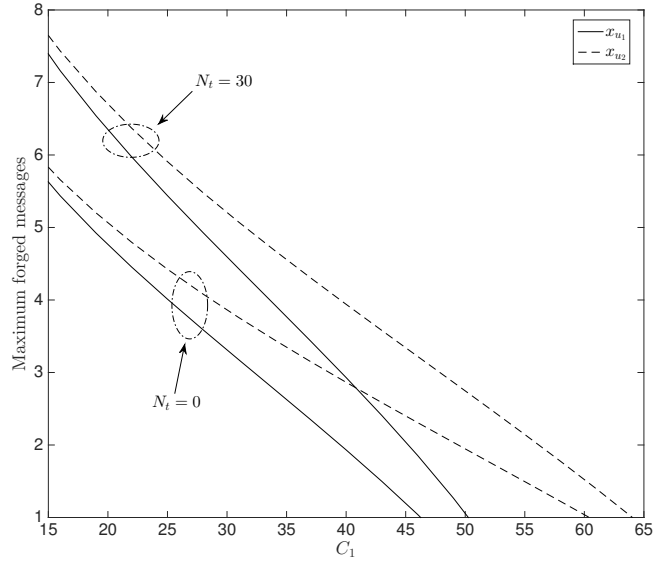


Figure 3.5 : The maximum number of forged messages of a malicious \mathcal{P} with the growth of C_1 , where $N_r = 90$, $N_n = 10$, $C_2 = 80$, $C_3 = 5$, $p_1 = 0.1$, and $p_2 = 0.5$. Two values of N_t , i.e., 0 and 30, are considered.

Fig. 3.5 shows the maximum number of forged messages with the growth of C_1 . Two values of N_t , i.e., 0 and 30, are considered. From the figure, we can see that

the maximum number of forged messages decreases with the increasing value of C_1 . The maximum number of forged messages can be less than 1, indicating that \mathcal{P} has no incentive to publish forged messages. In other words, the increasing value of C_1 can effectively suppress forged messages. We also see that the misclassification of genuine messages gives malicious \mathcal{P} chances to publish more forged messages, i.e., the dash curves are above the solid curves, especially for large C_1 . For small reward of genuine messages, e.g., $C_1 = 15$ in the figure, hiring bots helps \mathcal{P} publish more forged messages without being detected.

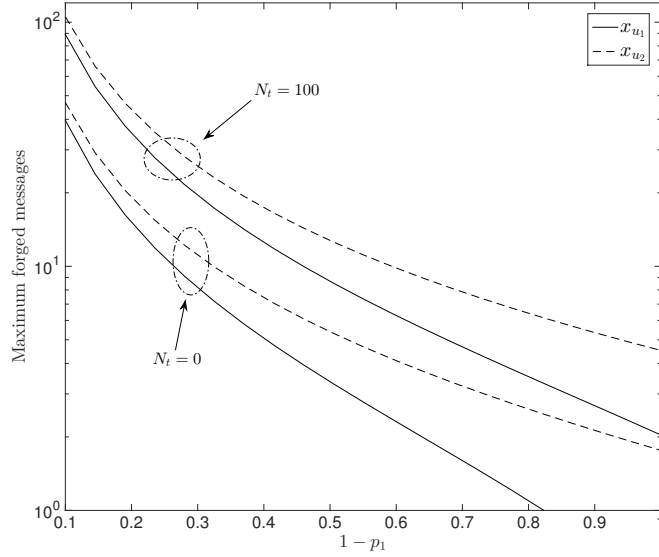


Figure 3.6 : The maximum number of forged messages of \mathcal{P} , i.e., x_{u_1} and x_{u_2} , by publishing forged messages, where $N_r = 90$, $N_n = 10$, $C_1 = 30$, $C_2 = 50$, $C_3 = 5$, and $p_2 = 0.5$. $N_t = 0$ and 100, are considered.

Fig. 3.6 plots the maximum number of forged messages, i.e., x_{u_1} and x_{u_2} , with the growth of the detection rate of forged messages, i.e., $1 - p_1$. $N_t = 0$ and 100 are considered. The y -axis is in a logarithmic scale. We can see that the improvement of the detection rate can effectively suppress forged messages when the detection rate is low, e.g., $1 - p_1 = 0.1$. We also see that that the maximum number of forged

messages can be more than one even in the case where $1 - p_1 = 1$, especially where many bots are hired, i.e., $N_t = 100$. As a result, the improvement of the detection rate cannot stop \mathcal{P} from publishing forged messages. Based on Figs. 3.5 and 3.6, \mathcal{A} can suppress forged messages by improving the detection rate in the beginning, and by increasing the reward for genuine messages.

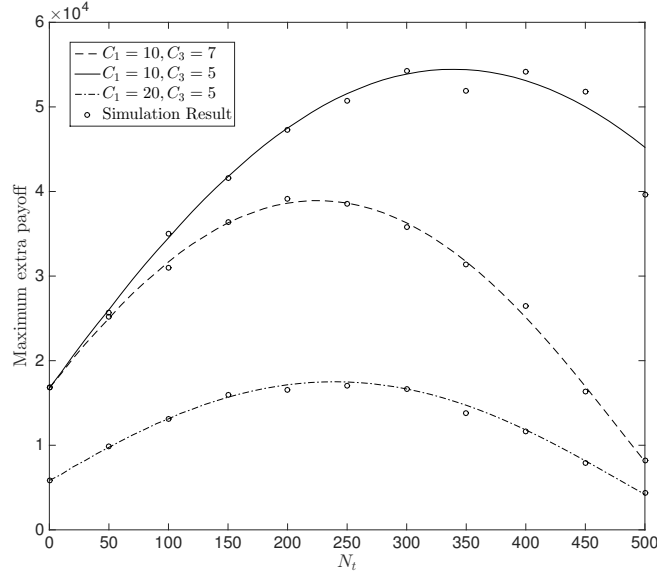


Figure 3.7 : The maximum extra payoff of \mathcal{P} in the misclassification-free game with the growth of N_t , where $N_r = 90$, $N_n = 10$, $C_2 = 50$, $p_1 = 0.5$. Every dot is an average result of 2,000 independent runs.

Fig. 3.7 plots the maximum extra payoff of \mathcal{P} with the growing number of bots in misclassification-free games. We can see that bots can increase the maximum extra payoff from the start, but decrease the maximum extra payoff later. Thus, there exists the best number of bots for the highest extra payoff. This is because the bots benefit the extra payoff by reducing the detection rate of forged messages. The detection rate is non-negative. As a result, the benefit of bots is upper bounded. However, the cost of bots increases linearly with the number of bots. In extreme cases, the benefit from the decreased detection rate cannot counteract the cost of

bots, e.g., $N_t = 500$ in the case of $C_1 = 10$ and $C_3 = 7$. It can be found that the bots are more effective in the case where genuine messages provide lower payoffs, as can be seen by comparing the curves of $C_1 = 10$ and $C_1 = 20$ under the same unit payoff of forged messages (i.e., $C_2 = 50$). We also see that low-cost bots, e.g., $C_3 = 5$ vs $C_3 = 7$, can bring higher extra payoff. As a result, the malicious \mathcal{P} has incentive to hire bots if the bots are low-cost and forged messages bring high payoffs.

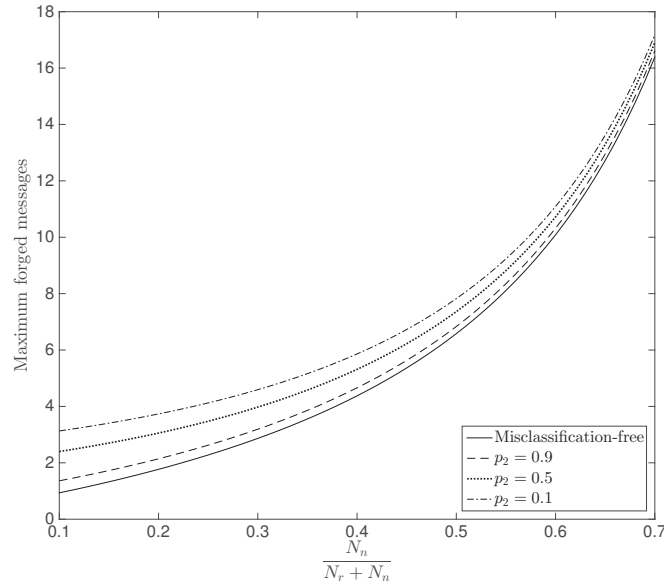


Figure 3.8 : The maximum number of forged messages with the growth of the ratio of fans, i.e., $\frac{N_n}{N_r + N_n}$, where $N_r + N_n = 100$, $N_t = 10$, $C_1 = 30$, $C_3 = 5$, $C_2 = 50$ and $p_1 = 0.1$.

Fig. 3.8 shows the maximum number of forged messages with the growing ratio of fans, i.e., $\frac{N_n}{N_r + N_n}$. We can see that \mathcal{P} can publish more forged messages with more fans, because fans persistently trust the messages from \mathcal{P} . We note that the increasing probability of misclassification, i.e., from $p_2 = 0.9$ to $p_2 = 0.1$, also offers chances for \mathcal{P} to publish more forged messages, especially in the case of a low ratio of fans. This is because the misclassification enlarges the payoff gap between forged and genuine messages by reducing rewards for genuine messages.

3.5 Conclusion

In this chapter, infinitely repeated games were developed to capture the interactions between a message publisher and the network administrator in OSNs for the suppression of forged messages. In the absence and presence of misclassification on genuine messages, sufficient conditions under which the publisher is disincentivized from publishing forged messages were identified. Closed-form expressions were derived for the maximum number of forged messages of a malicious publisher. Confirmed by simulations, our analysis indicates that forged messages can be suppressed by improving the payoffs for genuine messages, increasing the cost of bots, and/or reducing the payoffs for forged messages. The increasing detection probability of forged messages or decreasing misclassification probability of genuine messages can also have a strong impact on the suppression of forged messages.

Chapter 4

Markov-based Accurate Propagation Model

Like biological viruses, attacks in computer networks also need to propagate, so as APT attacks [18]. For example, attackers can use victim nodes as tunnels to attack their adjacent nodes in the lateral propagation stage. APT attacks can infect as many nodes inside of the internal networks as possible to move towards final targeted nodes. This is first because the targets' internal networks are unknown to attackers. Attackers need to continuously attack new nodes to collect more internal information, especially at the information leakage (attack) stage. Moreover, more attacked nodes can bring more opportunities for succeeding attacks. Note that it is difficult for defenders to completely remove ATP from internal networks due to the persistent feature of APT attack [28]. On the other hand, the victim nodes in APT would not be totally destroyed by the attackers, which gives the defenders chances to repair the nodes. As a result, there will be long-term infection and repairing transitions in the propagation of APT. The state transition of SIS epidemic model can be used to describe this process. For example, [43] uses the SIS virus propagation model to analyze the attack-defender interactions in APT. Under the above background, this chapter proposes a Markov model to capture and analyze the propagation process of APT by adopting the SIS epidemic process.

4.1 Introduction

The APT attack has become a serious risk of the present society. A model capturing impact of network topology on the complex infect-recovery interactions can be the key to predict the propagation process, evaluate defense strategies, prevent

and combat attacks.

Epidemic models, originally developed for biological infectious diseases [66], have been extensively used to provide a better understanding of computer virus propagations [67]. An element in such models can be infected with a probability β by an infected neighbor. An infected element can be cured with probability δ , typically triggered by curing actions. However, the extensions of the epidemic models to computer networks are not trivial, because the network topologies, which connect the computers (but do not exist in biological scenarios), can have a strong impact on computer virus propagations [120]. Earlier epidemic models for computer viruses were based on fully connected networks [67, 71] or statistic topologies with the degrees of individual nodes yielding a power law [73, 74]. Despite providing insights on the infection statistics, the models cannot capture specific networks. Later, Markov models were developed to accurately characterize the infection processes between connected nodes [120, 79, 35]. However, the models have exponential complexity, and subsequently, limited scalability to large networks. Recently in [79, Sec. III][35, Sec. IV], Markov models were decoupled to a set of differential equations, one per node, to linearize the complexity, where instant infection probabilities of connected nodes were uncorrelated by taking moment closure approximations. Unfortunately, the accuracy of the models degrades due to the approximation [35, Fig. 11].

4.1.1 Problem Statement

A key challenge to be addressed for modeling virus propagations is the trade-off between the modeling accuracy and complexity, which limits the extension of the models to large networks with thousands of nodes and non-trivial topologies.

The existing models either simplify analysis by implicitly assuming a complete graph or using statistically characterized topologies, limiting the practical applications of the models [66], [73]; or use Markov techniques to characterize detailed

virus spread in specific topologies, resulting in prohibitive complexity and limited scalability [35, 77, 78]. Some of the Markov models were simplified to tackle the complexity issue, using mean-field approximations, which again incurred significant losses of modeling accuracy [121, 35].

A widely approved result from the existing virus propagation models indicates that a virus dies out quickly, i.e., the mean virus lifetime is no longer than $\frac{\log(N)+1}{1-\frac{\beta}{\delta}\rho(\mathbf{A})}$, if $\frac{\beta}{\delta} < \frac{1}{\rho(\mathbf{A})}$ [120, 121, 79, 35]. N is the number of nodes in the network, \mathbf{A} is the adjacency matrix, and $\rho(\mathbf{A})$ is the largest eigenvalue of \mathbf{A} . However, the result only provides a sufficient condition for the exponential extinction of a virus. In the case that $\frac{\beta}{\delta} > \frac{1}{\rho(\mathbf{A})}$, there are still possibilities that infection dies out exponentially over time, but this has not been addressed in the existing models. In addition, the result has limited applications to large networks, as it can only accommodate $\beta \rightarrow 0$. Take a 5000-node network with a topology of 4000-regular graph (i.e., $\rho(\mathbf{A}) = 4000$) for an example. the extinction condition requires $\beta < 0.00025$.

4.1.2 Our Contributions

In this chapter, we propose a new approach to modeling the virus propagation across non-trivial topologies, which is able to leverage the modeling accuracy and complexity, hence enhancing modeling scalability to large networks. Our approach starts by modeling the virus infection of each individual node using the classical SIS model and the virus propagation among the nodes using Markov model (to be more specific, a discrete-time absorbing Markov process which is developed based on the adjacency matrix \mathbf{A}). As a result, our model, being a non-trivial extension of the SIS model, can characterize the SIS behavior of individual nodes in networks with explicit topologies. Matrix analysis techniques and Jordan decompositions are employed to study the transition matrix and the absorption rate (i.e., the virus extinction rate) of the discrete-time absorbing Markov process. By evaluating the

bounds of the second-largest eigenvalue of the transition matrix, the virus extinction rate R is proved to be bounded by $-\ln(1 - \delta^N) \leq R \leq -\ln(1 - \delta(1 - \beta + \beta\delta)^d)$. N is the network size and d is the minimum degree of the network.

We also provide a practical means to scale the bounds, to design and evaluate the lifetime and extinction of a virus in large networks with non-trivial topologies. This is based on our finding that, given any required virus extinction rate, the minimum curing probability δ_{\min} specified through the new upper bound, can ensure the gap between the required extinction rate and the actually achieved mean extinction rate fast converges with the growth of the network size. In light of this, we propose to interpret the virus extinction requirement of a large network to that of a much smaller one, and obtain the minimum curing probability of the large network efficiently through the upper bound of its smaller counterpart. Simplifications can be achieved with a negligible loss of accuracy.

Confirmed by simulations, our bounds can evaluate the virus extinction rate and lifetime across a broad spectrum of β and δ . They can also readily analyze networks with thousands of nodes in a few hours, with modeling accuracy preserved. In contrast, the Monte-Carlo simulations of such large networks would run for months and are impractical given current computing capabilities. In this sense, our analysis is of practical value.

4.1.3 Organization and Notations

The rest of this chapter is organized as follows. In Section 4.2, a new discrete-time absorbing Markov epidemic model is developed, followed by the analysis and derivations on the virus extinction rate and its bounds in Section 4.3. Numerical results are provided in Section 4.4, followed by conclusions in Section 4.5.

Notations used in the chapter are as listed in Table 4.1.

Table 4.1 : Notations Used in Chapter 4

Notation	Description
G	a network topology graph
N	network size
\mathbf{A}	the adjacency matrix of the network topology graph
$a_{i,j}$	the (i, j) -th element of \mathbf{A}
d	the minimum degree of the network topology
\mathbf{s}_i	the i -th network state
$\mathbf{s}_i(l)$	the status of the l -th node in state \mathbf{s}_i
δ	the curing probability of an infected node
β	the infection probability of a node
d_l^i	the no. of edges between node l and the infected nodes in \mathbf{s}_i
\mathbf{P}	$2^N \times 2^N$ Markov transition matrix
$p_{i,j}$	the (i, j) -th element of \mathbf{P}
\mathbf{P}_{in}	the transition matrix of infection
\mathbf{P}_{cu}	the transition matrix of curing
$\mathbf{t}^{(k)}$	$1 \times 2^{N-1}$ state probability vector
\mathbf{Q}	the $2^{N-1} \times 2^{N-1}$ bottom-right submatrix of \mathbf{P}
λ_i	the i -th largest eigenvalue of \mathbf{P}
R	virus extinction rate

4.2 Discrete-time Markov model for Virus Propagation in Computer Networks

We consider a general network topology presented by a constant, undirected, connected graph $G(\mathcal{N}, \mathcal{E})$, where \mathcal{N} is the set of vertices and \mathcal{E} is the set of edges.

The size of \mathcal{N} is N , i.e. $|\mathcal{N}| = N$, where $|\cdot|$ denotes cardinality. Let \mathbf{A} denote the adjacency matrix of the graph, where the (i, j) -th element $a_{i,j} = 1$ if vertices i and j are connected by an edge; or $a_{i,j} = 0$, otherwise. Fig. 4.1 provides an example with $N = 8$, where a Barabási-Albert graph* with minimum degree 2 is considered. Its adjacency matrix can be given by (4.1).

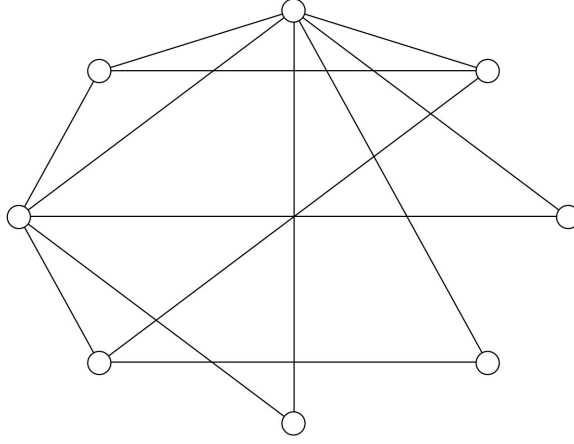


Figure 4.1 : An example of the BA-2 graph.

Every vertex in the graph can be in one of two statuses, i.e. susceptible/uninfected and infected. Recall that β is the infection probability of each edge in a single step. A susceptible node which is connected with k infected neighbors can be infected with the probability of $(1 - (1 - \beta)^k)$, or remains susceptible with the probability of

*The steps of generating a BA- d graph (d is the minimum degree) are as follows. 1. *Initialization*: set up d nodes; 2. *Growth*: add $(N - d)$ nodes in serial as such that, for every new node, d edges are added to connect it and d existing nodes; 3. *Preferential attachment*: The probability that a new node is connected to node i is proportional to the degree of node i ; 4. *Degree check*: If the degree of node j is smaller than d , edges are added to connect node j until its degree is equal to d . The first three steps are the original BA algorithm [122] and the fourth step is newly added to ensure that the minimum degree of the generated graph.

$(1 - \beta)^k$. On the other hand, an infected node can be cured with the probability of δ in a single step, or remains infected with the probability of $(1 - \delta)$.

$$\mathbf{A}_{BA-2} = \begin{bmatrix} 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}. \quad (4.1)$$

Let a binary vector, \mathbf{s} , be the state of the entire network, collecting the infected or susceptible status of every node in the network. There are totally 2^N such states, labelled by $\mathbf{s}_i (i = 0, 1, \dots, 2^N - 1)$, $i = \sum_{k=1}^N (2^{k-1} \mathbf{s}_i(k))$, where $\mathbf{s}_i(k)$ is the k -th element of \mathbf{s}_i ($k = 1, 2, \dots, N$). If node k is infected, $\mathbf{s}_i(k) = 1$; otherwise, if the node is susceptible, $\mathbf{s}_i(k) = 0$.

In practice, the security servers of a network can disseminate patches after virus outbreaks. The infected nodes that receive the patch can cure the virus infection, and return to an uninfected state, with the probability of δ . The servers can also update the patches in response to the evolution/development of viruses, and disseminate the updated patches. For modeling convenience, we assume that the patches are disseminated throughout the network at an interval of a time step. During a time step, infections can take place only between directly linked neighbors, and a newly infected node does not infect others during the time step. This assumption is suitable for many epidemic scenarios where viruses hibernate before spreading [123]. Later, we will discuss the extension of our model to general cases, where infections can propagate instantly within a time step.

A 2^N -dimensional discrete-time absorbing Markov process can be constructed to characterize the virus propagation with the states \mathbf{s}_i , where the absorbing state is \mathbf{s}_0 . This is because the virus propagation retains the Markov property that the future state only depends on the current state, and is independent of the past states. A flow chart to obtain the transition matrix is given by Fig. 4.2. The $2^N \times 2^N$ transition matrix of the discrete-time, absorbing Markov process can be written as

$$\mathbf{P} = \mathbf{P}_{in} \mathbf{P}_{cu}, \quad (4.2)$$

where \mathbf{P}_{in} and \mathbf{P}_{cu} are the transition matrices of infection and curing, respectively.

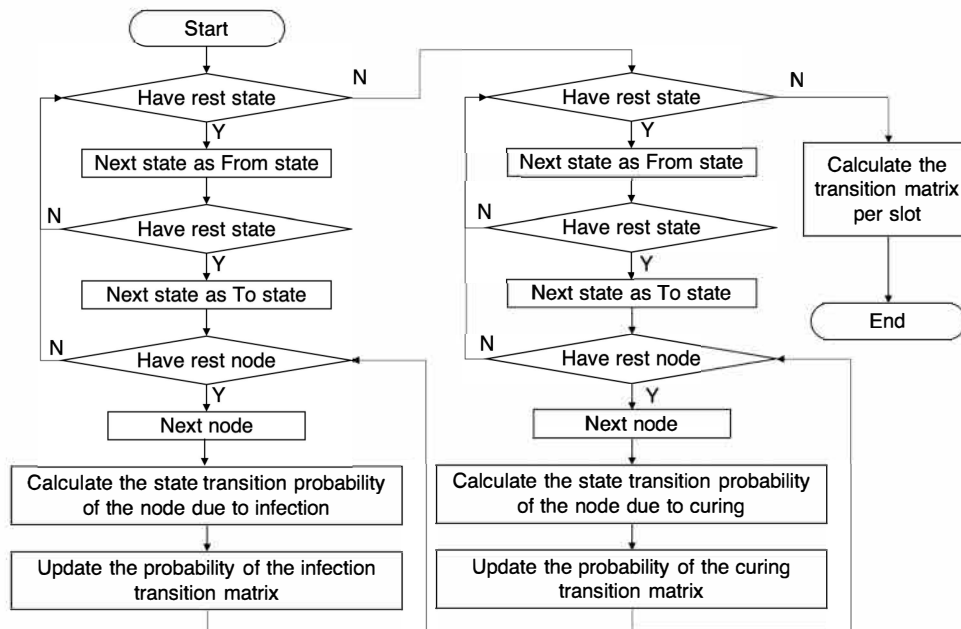


Figure 4.2 : Flow chart to construct the transition matrix of the Discrete-time Markov model

The (i, j) -th element of \mathbf{P}_{in} can be given by

$$\mathbf{P}_{in}(i, j) = \prod_{l \in \mathcal{N}} \Pr\{\mathbf{s}_j(l) | \mathbf{s}_i(l), infection\},$$

where $\mathbf{s}_i(l)$ and $\mathbf{s}_j(l)$ denote the status of node l in state \mathbf{s}_i and \mathbf{s}_j before and after the infection, respectively; $\Pr\{\mathbf{s}_j(l)|\mathbf{s}_i(l), infection\}$ is the transition probability of

node l before curing takes place at the node, and can be given by

$$\Pr\{\mathbf{s}_j(l)|\mathbf{s}_i(l), infection\} = \begin{cases} (1 - \beta)^{d_l^{(i)}} & \text{if } \mathbf{s}_j(l) = 0, \mathbf{s}_i(l) = 0; \\ 1 - (1 - \beta)^{d_l^{(i)}} & \text{if } \mathbf{s}_j(l) = 1, \mathbf{s}_i(l) = 0; \\ 0 & \text{if } \mathbf{s}_j(l) = 0, \mathbf{s}_i(l) = 1; \\ 1 & \text{if } \mathbf{s}_j(l) = 1, \mathbf{s}_i(l) = 1, \end{cases} \quad (4.3)$$

where $d_l^{(i)} = \sum_{k=1}^N (\mathbf{s}_i(k) \times a_{k,l})$ is the number of physical connections between node l and the infected nodes in \mathbf{s}_i .

Note that, during the infection stage (i.e., before curing takes place at any nodes, a susceptible node can remain susceptible with the probability of $(1 - \beta)^{d_l^{(i)}}$, or become infected with the probability of $1 - (1 - \beta)^{d_l^{(i)}}$. In other words, $\mathbf{s}_j(l) \geq \mathbf{s}_i(l) \forall l = 1, \dots, N$ for a valid state transition with a non-zero transition probability from state $\mathbf{s}_i(l)$ to $\mathbf{s}_j(l)$. The state index $j = \sum_{k=1}^N (2^{k-1} \mathbf{s}_j(k)) \geq i = \sum_{k=1}^N (2^{k-1} \mathbf{s}_i(k))$ for such transitions, as defined earlier. A state can only transit to other states with larger indices. As a result, \mathbf{P}_{in} is an upper triangular matrix.

The (i, j) -th element of \mathbf{P}_{cu} can be given by

$$\mathbf{P}_{cu}(i, j) = \prod_{l \in \mathcal{N}} \Pr\{\mathbf{s}_j(l)|\mathbf{s}_i(l), curing\},$$

where \mathbf{s}_i and \mathbf{s}_j denote the network status before and after curing takes place at the node, and $\mathbf{s}_i(l)$ and $\mathbf{s}_j(l)$ denote the status of node l in state \mathbf{s}_i and \mathbf{s}_j , respectively. The probability $\Pr\{\mathbf{s}_j(l)|\mathbf{s}_i(l), curing\}$ is the transition probability of the l -th node after curing takes place at the node, and is given by

$$\Pr\{\mathbf{s}_j(l)|\mathbf{s}_i(l), curing\} = \begin{cases} 1 & \text{if } \mathbf{s}_j(l) = 0, \mathbf{s}_i(l) = 0; \\ 0 & \text{if } \mathbf{s}_j(l) = 1, \mathbf{s}_i(l) = 0; \\ \delta & \text{if } \mathbf{s}_j(l) = 0, \mathbf{s}_i(l) = 1; \\ 1 - \delta & \text{if } \mathbf{s}_j(l) = 1, \mathbf{s}_i(l) = 1. \end{cases} \quad (4.4)$$

Note that, every instant curing takes place, an infected node can remain infected with the probability of $(1 - \delta)$, or become susceptible with the probability of δ . In other words, $\mathbf{s}_j(l) \leq \mathbf{s}_i(l) \forall l = 1, \dots, N$ for a valid state transition with a non-zero transition probability from state $\mathbf{s}_i(l)$ before the instant to $\mathbf{s}_j(l)$ after the instant. The state index $j = \sum_{k=1}^N (2^{k-1} \mathbf{s}_j(k)) \leq i = \sum_{k=1}^N (2^{k-1} \mathbf{s}_i(k))$ for such transitions. A state can only transit to other states with smaller indices. As a result, \mathbf{P}_{cu} is a lower triangular matrix.

As a result, the (i, j) -th element of \mathbf{P} , denoted by $p_{i,j}$, can be given by

$$\begin{aligned} p_{i,j} &= \sum_{k=0}^{2^N-1} (\mathbf{P}_{in}(i, k) \times \mathbf{P}_{cu}(k, j)) \\ &= \sum_{k=0}^{2^N-1} \left(\prod_{l:\mathcal{N}} \Pr\{\mathbf{s}_k(l) | \mathbf{s}_i(l), \text{infection}\} \times \prod_{l:\mathcal{N}} \Pr\{\mathbf{s}_j(l) | \mathbf{s}_k(l), \text{curing}\} \right). \end{aligned}$$

We have that $\mathbf{P}_{in}(0, 0) = 1$ and $\mathbf{P}_{in}(0, k) = 0, k \neq 0$; $\mathbf{P}_{cu}(0, 0) = 1$ and $\mathbf{P}_{cu}(0, k) = 0, k \neq 0$. Thus we have $\mathbf{P}(0, 0) = 1$, i.e., the virus-free state \mathbf{s}_0 is an absorbing state. When $\delta > 0$ and $\beta > 0$, this condition only holds for \mathbf{s}_0 , i.e., \mathbf{s}_0 is the only absorbing state for $\delta > 0$ and $\beta > 0$. As a result, \mathbf{P} has the following form

$$\mathbf{P} = \left[\begin{array}{c|ccc} 1 & 0 & \cdots & 0 \\ \hline p_{1,0} & p_{1,1} & \cdots & p_{1,2^N-1} \\ p_{2,0} & p_{2,1} & \cdots & p_{2,2^N-1} \\ \vdots & \vdots & \ddots & \vdots \\ p_{2^N-1,0} & p_{2^N-1,1} & \cdots & p_{2^N-1,2^N-1} \end{array} \right],$$

which, for illustration convenience, is rewritten as block submatrices, as given by

$$\mathbf{P} = \left(\begin{array}{c|c} \mathbf{1} & \mathbf{0} \\ \hline \mathbf{R} & \mathbf{Q} \end{array} \right).$$

Let $\mathbf{t}^{(k)} = [t_1^{(k)}, \dots, t_{2^N-1}^{(k)}]$ be the vector collecting the probabilities of \mathbf{s}_1 to \mathbf{s}_{2^N-1} in the k -th time step since the virus starts spreading, where $t_i^{(k)}$ is the probability

that the Markov process is in state \mathbf{s}_i . $\mathbf{t}^{(0)}$ is the initial state.

Clearly, $\mathbf{t}^{(k)}$ can be written as

$$\mathbf{t}^{(k)} = \mathbf{t}^{(k-1)}\mathbf{Q} = \dots = \mathbf{t}^{(0)}\mathbf{Q}^k. \quad (4.5)$$

Let $\mathbf{c} = [c_1, c_2, \dots, c_{2N-1}]$ where c_i denotes the infected population of state \mathbf{s}_i . The expected infected population after k time steps can be given by

$$\mathbb{E}\{\text{population}|\mathbf{t}^{(0)}, k\} = \mathbf{c}(\mathbf{t}^{(k)})^T = \mathbf{c}(\mathbf{t}^{(0)}\mathbf{Q}^k)^T. \quad (4.6)$$

We can also obtain the infection probability; which refers to the probability that the virus remains active, and can be calculated as the sum of $\mathbf{t}^{(k)}$, i.e.

$$\Pr\{\text{infection}|\mathbf{t}^{(0)}, k\} = \mathbf{1}(\mathbf{t}^{(k)})^T = \mathbf{1}(\mathbf{t}^{(0)}\mathbf{Q}^k)^T. \quad (4.7)$$

This discrete-time absorbing Markov model can be readily generalized to the case where infection can propagate instantly within a time step. Suppose that the infection can propagate h times within a time step, the infection process can be described as a time-homogeneous Markov chain and the transition matrix during the time step, i.e., (4.2), can be replaced by

$$\mathbf{P} = \mathbf{P}_{in}^h \mathbf{P}_{cu}.$$

In this case, the calculations of the expected infected population and the infection probabilities still follow (4.6) and (4.7).

4.3 Extrapolation of Markov Epidemic Modeling to Large Networks

In this section, we propose a new approach to reducing the complexity of Markov epidemic modeling. Different from the existing curve-fitting eigenvalue approximation [35] and the mean-field approximation[35, 79], the approach that we take is to

conduct eigenvalue analysis and Jordan decomposition on the transition matrix of the Markov process developed in Section 4.2. It enables us to present the absorption rate of the discrete-time absorbing Markov process, i.e., the extinction rate of the virus, in form of the eigenvalues, and to derive the closed-form bounds for the rate. The closed-form bounds can be computed efficiently without requirements of explicit matrix operations, e.g., matrix decompositions, and the independence assumptions or approximations.

4.3.1 Virus Extinction Rate

Given the 2^N -dimensional discrete-time absorbing Markov process developed in Section 4.2, the rate of the Markov process converging to the virus-free absorbing state \mathbf{s}_0 , or in other words, the virus extinction rate, approaches to [124]

$$R = \lim_{k \rightarrow \infty} \left(-\ln \frac{\|\mathbf{t}^{(k)}\|}{\|\mathbf{t}^{(k-1)}\|} \right),$$

where $\|\cdot\|$ denotes vector norm, and $\ln(\cdot)$ denotes the natural logarithm operation. In other words, the virus diminishes in the N -node computer network at the rate of R .

Here, we focus on the case that $\delta > 0$ and $\beta > 0$, because it corresponds to the practical and complex scenario where both the virus propagation and curing exist at the same time. In the case that the virus is undetected or eliminated completely, i.e., $\delta = 0$ or $\beta = 0$, \mathbf{P} recedes to \mathbf{P}_{in} or \mathbf{P}_{cu} . In other words, \mathbf{P} becomes a triangular matrix. The eigenvalues of \mathbf{P} can be easily obtained. The analysis on the virus extinction rate can be conducted in the same way as what follows.

To evaluate the virus extinction rate, we put forward the following theorem.

Theorem 5. *Given the 2^N -dimensional discrete-time absorbing Markov process of virus propagation, when $\delta > 0$ and $\beta > 0$, the eigenvalues of the transition matrix*

\mathbf{P} , denoted by λ_i ($i = 0, 1, \dots, 2^N - 1$), satisfy

$$1 = \lambda_0 > \lambda_1 > |\lambda_2| \geq \dots \geq |\lambda_{2^N-1}| > 0,$$

where the eigenvalues are arranged in the descent order of their absolute values.

Proof. We first prove that $\lambda_0 = 1$, where λ_0 is the largest eigenvalue of \mathbf{P} . We note that \mathbf{P} is a block triangular matrix and $\mathbf{1}$ and \mathbf{Q} are its diagonal blocks. According to [125], the eigenvalues of \mathbf{P} and \mathbf{Q} satisfy the following condition

$$\lambda(\mathbf{P}) = 1 \cup \lambda(\mathbf{Q}), \quad (4.8)$$

where $\lambda(\mathbf{P})$ and $\lambda(\mathbf{Q})$ denote the sets of the eigenvalues of \mathbf{P} and \mathbf{Q} , respectively.

According to [124, Sec 7.1.4], the largest eigenvalue of \mathbf{Q} is no larger than any matrix norm of \mathbf{Q} . We have $\max \{\lambda(\mathbf{Q})\} \leq \|\mathbf{Q}\|_\infty$, where $\|\cdot\|_\infty$ denotes infinity matrix norm, i.e., the maximum absolute row sum of the matrix.

We also have $p_{i,j} \geq 0$ and $\sum_{j=0}^{2^N-1} p_{i,j} = 1$. On the other hand, all the elements of \mathbf{R} are greater than 0, since every state has non-zero probability to transfer to the virus-free absorbing state within one time step. We have

$$\begin{aligned} \max \{\lambda(\mathbf{Q})\} &\leq \|\mathbf{Q}\|_\infty = \max \left\{ \sum_{j=1}^{2^N-1} |q_{ij}| \right\} \\ &= 1 - \min \{p_{i,0}\} < 1, \end{aligned}$$

where $q_{i,j}$ is the (i, j) -th element of \mathbf{Q} . As a result, $\lambda_0 = \max \{\lambda(\mathbf{P})\} = 1$.

Next, we prove the uniqueness of λ_1 , which is the maximal eigenvalue of \mathbf{Q} . In the case that $\beta, \delta > 0$, \mathbf{Q} is irreducible and primitive. This is because any transient state $(\mathbf{s}_1 \cdots \mathbf{s}_{2^N-1})$ is able to transfer to \mathbf{s}_{2^N-1} , since G is connected meanwhile \mathbf{s}_{2^N-1} can transfer to any state (including itself) with non-zero probabilities. Therefore, $\mathbf{Q}^m > 0$ for some $m > 0$, where $\mathbf{Q}^m > 0$ means that every element of \mathbf{Q}^m is greater than 0.

Therefore, \mathbf{Q} satisfies the Perron Frobenius theorem[124], and we have that λ_1 is real and

$$\lambda_1 > |\lambda_k|, k = 2, 3, \dots.$$

Then we proceed to prove that $0 \notin \lambda(\mathbf{P})$. Both of $|\mathbf{P}_{in}|$ and $|\mathbf{P}_{cu}|$ are larger than 0, where $|\cdot|$ denotes determinant. This is because \mathbf{P}_{in} and \mathbf{P}_{cu} are an upper and lower triangular matrices, respectively, and their diagonal elements are positive, which are self-transition probabilities. Given that $\mathbf{P} = \mathbf{P}_{in}\mathbf{P}_{cu}$, the determinant of \mathbf{P} satisfies

$$|\mathbf{P}| = |\mathbf{P}_{in}| \times |\mathbf{P}_{cu}| > 0.$$

In other words, \mathbf{P} is full rank with 2^N non-zero eigenvalues.

$$\text{As a result, } 1 = \lambda_0 > \lambda_1 > |\lambda_2| \geq \dots \geq |\lambda_{2^N-1}| > 0. \quad \square$$

Using basic results of discrete-time linear systems, the absorption rate of the 2^N -dimensional discrete-time absorbing Markov virus propagation process, i.e., the virus extinction rate, can be given by

$$R = -\ln(\lambda_1), \quad (4.9)$$

where λ_1 is the second largest eigenvalue of the transition matrix of the Markov process. The proof of (4.9) is based on Jordan decompositions.

Proof. By Theorem 5, the largest eigenvalue of \mathbf{Q} is less than 1, thus $\lim_{k \rightarrow \infty} \mathbf{Q}^k = \mathbf{0}$ and $\lim_{k \rightarrow \infty} \mathbf{t}^{(k)} = \lim_{k \rightarrow \infty} \mathbf{t}^{(0)} \mathbf{Q}^k = \mathbf{0}$. In other words, the proposed discrete-time absorbing Markov process converges to the virus-free absorbing state $\mathbf{s}_0 = (0, \dots, 0)$ with $\mathbf{t}^{(k)}$ approaching to $\mathbf{0}$.

The convergence rate relies on the structure of \mathbf{Q} . We decompose \mathbf{Q} to analyze the state transition. Specifically, we propose to decompose \mathbf{Q} by using Jordan

form [124], when \mathbf{Q} has m ($2 \leq m \leq 2^N - 1$) different eigenvalues, as given by

$$\mathbf{Q} = \mathbf{H}(\mathbf{J}_{\lambda_1} \oplus \mathbf{J}_{\lambda_2} \oplus \cdots \oplus \mathbf{J}_{\lambda_m})\mathbf{H}^{-1}, \quad (4.10)$$

where \mathbf{J}_{λ_i} is the Jordan segment associated with λ_i of \mathbf{Q} . The diagonal elements of \mathbf{J}_{λ_i} are l Jordan blocks $\mathbf{J}_*(\lambda_i)$, i.e., $\mathbf{J}_{\lambda_i} = \oplus_{k=1}^l \mathbf{J}_k(\lambda_i)$, where l is the geometric multiplicity of λ_i ; $\mathbf{J}_*(\lambda_i)$ is an upper triangular matrix with all diagonal elements equal to λ_i , the elements immediately above the main diagonal equal to 1, and all the other elements equal to 0. The numerical results of Jordan blocks $\mathbf{J}_*(\lambda)$ and \mathbf{H} are generated along with the Jordan decomposition.

Therefore, the transition matrix of k consecutive time steps is given by

$$\mathbf{Q}^k = \mathbf{H}(\mathbf{J}_{\lambda_1}^k \oplus \mathbf{J}_{\lambda_2}^k \oplus \cdots \oplus \mathbf{J}_{\lambda_m}^k)\mathbf{H}^{-1}. \quad (4.11)$$

It is proved in [124] that $\lim_{k \rightarrow \infty} \mathbf{J}_*(\lambda) = 0$ if and only if $|\lambda| < 1$. Therefore, the Jordan block associated with λ_1 converges to zero slowest, i.e., the convergence rate of \mathbf{Q} is governed by the largest eigenvalue λ_1 , because $\lim_{k \rightarrow \infty} \mathbf{J}_*(\frac{|\lambda|}{\lambda_1}) = 0$ except for $\mathbf{J}_*(\lambda_1)$. Therefore, for a large k , we have

$$\mathbf{t}^{(k)} = \mathbf{t}^{(0)}\mathbf{G}_1 \approx \lambda_1^k \mathbf{t}^{(0)}\mathbf{G}_1. \quad (4.12)$$

Since $\lambda_1 > \lambda_i$ ($i = 2, \dots, 2^N - 1$) according to Theorem 5, \mathbf{G}_1 is given by $\mathbf{G}_1 = \mathbf{xy}^*/\mathbf{y}^*\mathbf{x}$ according to the Jordan decomposition, where \mathbf{x} and \mathbf{y}^* are respectively the right and left eigenvectors of \mathbf{Q} , associated with λ_1 .

When the virus propagation stabilizes, $\mathbf{t}^{(k)}$ converges to a zero vector exponentially, and at the rate λ_1 , i.e.

$$\lim_{k \rightarrow \infty} \frac{\|\mathbf{t}^{(k)}\|}{\|\mathbf{t}^{(k-1)}\|} = \lambda_1. \quad (4.13)$$

As a result, the extinction rate is $R = -\ln(\lambda_1)$. □

4.3.2 Closed-form Bounds of R

The virus extinction rate R , given by (4.9), still requires the calculation of the largest eigenvalue of \mathbf{Q} . In the case of large computer networks, \mathbf{Q} is large and the Jordan decomposition would be computationally expensive, or even practically prohibitive. To analyze the larger networks, we proceed to derive the upper and lower bounds of R , which can be dictated in the following theorem.

Theorem 6. *The extinction rate, R , of a virus in an N -node computer network is bounded by*

$$-\ln(1 - \delta^N) \leq R \leq -\ln(1 - \delta(1 - \beta + \beta\delta)^d),$$

where d is the minimum degree of the network topology.

Proof. Based on Perron Frobenius theorem [124], λ_1 satisfies

$$\min \left\{ \sum_j q_{i,j} \right\} \leq \lambda_1 \leq \max \left\{ \sum_j q_{i,j} \right\}.$$

In other words,

$$1 - \max\{p_{i,0}\} \leq \lambda_1 \leq 1 - \min\{p_{i,0}\},$$

$$i \in [1, 2, \dots, 2^N - 1],$$

where $p_{i,0}$ is the probability from the transient state \mathbf{s}_i to the virus-free absorbing state.

Note that $\mathbf{P}_{cu}(i, 0)$ decreases with the growth of the infected population. Considering the minimum $p_{i,0}$, we have

$$\begin{aligned} p_{i,0} &= \sum_{k=0}^{2^N-1} (\mathbf{P}_{in}(i, k) \mathbf{P}_{cu}(k, 0)) \\ &\geq \sum_{k=0}^{2^N-1} (\mathbf{P}_{in}(i, k) \mathbf{P}_{cu}(2^N - 1, 0)) = \delta^N, \\ i &\in [1, 2, \dots, 2^N - 1], \end{aligned} \tag{4.14}$$

where the equality holds if and only if $i = 2^N - 1$. Therefore, the upper bound of λ_1 can be given by

$$\lambda_1 \leq 1 - \delta^N. \quad (4.15)$$

On the other hand, $p_{i,0}$ reaches its maximum when only the node with the minimum degree is infected. For a given pair of δ and β , $p_{i,0}$ is determined by the number of the infected nodes, and the degree between the infected and susceptible nodes where they are respectively denoted by c_i and d_i , in the state \mathbf{s}_i . Obviously, $p_{i,0}$ declines with the increase of c_i and/or d_i . We are now to prove that $p_{g,0}$ is the maximum value among $p_{i,0}$ ($i > 0$). \mathbf{s}_g corresponds to the case where only the node with the minimum degree $d \geq 1$ is infected.

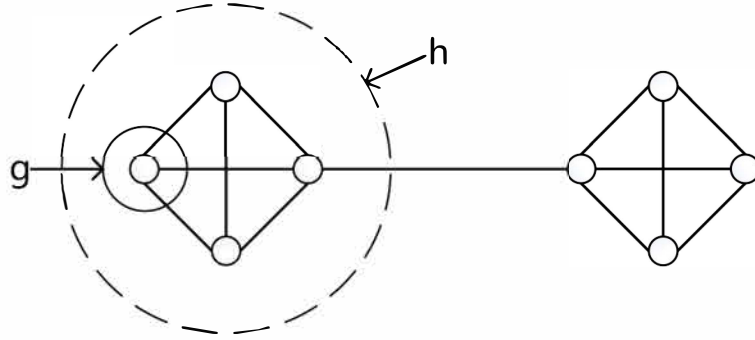


Figure 4.3 : A graph demonstrates the problem that which state has a larger $p_{i,0}$. State \mathbf{s}_g contains 1 infected node with 3 susceptible neighbors. On the contrary, state \mathbf{s}_h contains 4 infected nodes with 1 susceptible neighbor. Which one is larger, $p_{g,0}$ or $p_{h,0}$?

We first prove that $p_{g,0}$ is the maximum value among $p_{h,0}$ when $c_h = 1$, i.e., only one infected node in \mathbf{s}_h . We have,

$$c_h = 1 = c_g$$

$$d_h \geq d_g.$$

Given the same number of the infected nodes, $p_{i,0}$ declines with the increase of d_i . As a result, $p_{g,0}$ is the maximum among $p_{h,0}$, i.e., $p_{h,0} \leq p_{g,0}$.

Next, we show that $p_{g,0}$ is the maximum among $p_{h,0}$ when $c_h > 1$; in other words, there are more than one infected node in \mathbf{s}_h . Specifically, when $c_h > 1$, d_h satisfies

$$d_h \geq d \times c_h - 2 \binom{c_h}{2} = d \times c_h - c_h(c_h - 1).$$

Under this condition, there are two cases as follows.

(a) The number of the infected nodes in \mathbf{s}_h is less than $d + 1$, i.e., $1 < c_h < d + 1$.

Then, we have

$$d_h \geq d \times c_h - c_h(c_h - 1) \geq d = d_g$$

$$c_h > 1 = c_g.$$

In other words, state \mathbf{s}_h has more infected nodes and higher degrees between the infected nodes and the susceptible/uninfected nodes, than \mathbf{s}_g . As a result, $p_{g,0} > p_{h,0}$.

(b) The number of the infected nodes in state \mathbf{s}_h is no less than $d + 1$, i.e., $c_h \geq d + 1$. Then, we have

$$p_{h,0} = \sum_{k=0}^{2^N-1} (\mathbf{P}_{in}(h, k) \mathbf{P}_{cu}(k, 0)) \leq \sum_{k=0}^{2^N-1} (\mathbf{P}_{in}(h, k) \times \delta^{d+1}) = \delta^{d+1},$$

where the equality holds if and only if $c_h = d + 1$ and $d_h = 0$. We also have

$$p_{g,0} = \sum_{k=0}^{2^N-1} (\mathbf{P}_{in}(g, k) \mathbf{P}_{cu}(k, 0)) > \sum_{k=0}^{2^N-1} (\mathbf{P}_{in}(g, k) \times \delta^{d+1}) = \delta^{d+1}.$$

As a result, $p_{g,0} > \delta^{d+1} \geq p_{h,0}$.

To sum up, $p_{i,0}$ reaches its maximum when only the node with the minimum degree is infected, as given by

$$\begin{aligned} \max\{p_{i,0}\} &= p_{g,0} \\ &= \sum_{k=0}^d \binom{d}{k} \beta^k \delta^{k+1} (1 - \beta)^{(d-k)} \\ &= \delta(1 - \beta + \beta\delta)^d. \end{aligned}$$

The maximum $p_{i,0}$ is

$$\max\{p_{i,0}\} = \delta(1 - \beta + \beta\delta)^d.$$

Therefore, the lower bound of λ_1 can be written as

$$1 - \delta(1 - \beta + \beta\delta)^d \leq \lambda_1 \quad (4.16)$$

Combining (4.15) and (4.16), we have

$$1 - \delta(1 - \beta + \beta\delta)^d \leq \lambda_1 \leq 1 - \delta^N. \quad (4.17)$$

Substituting (4.17) into (4.9), Theorem 6 is proved. \square

From the theorem, we see that (a) the upper bound is determined by β , δ and d (or in other words, the upper bound is independent of the network size but affected by the network structure); and (b) the lower bound is determined by δ and N (or in other words, the lower bound relies on the network size).

We note that the lower bound corresponds to the case where only the node with the minimum degree d is infected across the entire network. δ is the probability that the node is cured, and $(1 - \beta + \beta\delta)^d$ is the probability that the d one-hop neighbors of the node are either not infected, or infected but soon cured before further spreading the virus. In the case of large heterogeneous networks, β can be very small and $\beta \ll \delta$. (4.16) can be further simplified as

$$1 - \delta \leq \lambda_1 \quad (4.18)$$

This corresponds to an extreme case where the virus becomes non-infectious after the node with the minimum degree d is infected. As a result, the lower bound can be simplified as the probability that the infected node stays infected.

We also note that our proposed model is discrete-time, and the bounds developed depend on δ and β , both of which are defined on per time-step. However, it can be

explicitly proved that the bounds depend on the actual temporal characteristics of the infection and cure, rather than the selection of the time-step size. The detailed proof is provided in the following.

Proof. We can define the time-step size to be the minimum time interval, denoted by τ_{\min} (in seconds), between any two consecutive infections and between any two consecutive curing instants. We use τ_{\min} to evenly discretize the time, and evaluate the probability β that an infection occurs within any interval and the probability δ that an infection is cured within the interval.

The resultant bounds for λ_1 per time step can be rewritten as per second, by taking the $\frac{1}{\tau_{\min}}$ -th power of both the upper and lower bounds, as given by

$$(1 - \delta(1 - \beta + \beta\delta)^d)^{\frac{1}{\tau_{\min}}} \leq \lambda_1 \text{ (/sec)} \leq (1 - \delta^N)^{\frac{1}{\tau_{\min}}} \quad (4.19)$$

Further decreasing the time-step size, i.e., to $\frac{\tau_{\min}}{T}$ (T is a positive integer), may result in different values of the bounds of λ_1 per second, as given by

$$(1 - \frac{\delta}{T}(1 - \frac{\beta}{T} + \frac{\beta\delta}{T^2})^d)^{\frac{T}{\tau_{\min}}} \leq \lambda_1 \text{ (/sec)} \leq (1 - \frac{\delta^N}{T^N})^{\frac{T}{\tau_{\min}}} \quad (4.20)$$

The resultant upper bound is looser, as $(1 - \delta^N)^{\frac{1}{\tau_{\min}}} \leq (1 - \frac{\delta^N}{T^N})^{\frac{T}{\tau_{\min}}} \leq (1 - \frac{\delta^N}{T^N})^{\frac{T}{\tau_{\min}}}$ based on Bernoulli's inequality (i.e., $(1 + x)^r \geq 1 + rx$ for $-1 \leq x < 0$ and $r > 0$). We still use $(1 - \delta^N)^{\frac{1}{\tau_{\min}}}$ from (4.19) as the upper bound, which is independent of the selection of time-step size (i.e., T).

The resultant lower bound in (4.20) can be evaluated by taking the limit of $T \rightarrow \infty$, as given by

$$\lim_{T \rightarrow \infty} \left(1 - \frac{\delta}{T}(1 - \frac{\beta}{T} + \frac{\beta\delta}{T^2})^d\right)^{\frac{T}{\tau_{\min}}} \leq \lim_{T \rightarrow \infty} \left(1 - \frac{\delta}{T}(1 - \beta + \beta\delta)^d\right)^{\frac{T}{\tau_{\min}}} \quad (4.21a)$$

$$= e^{-\frac{\delta(1-\beta+\beta\delta)^d}{\tau_{\min}}}, \quad (4.21b)$$

where (4.21a) is due to the fact that, when T is large enough, $\delta \leq \frac{T}{T+1} = \frac{1-\frac{1}{T}}{1-\frac{1}{T^2}} = \frac{\beta-\frac{\beta}{T}}{\beta-\frac{\beta}{T^2}}$, and in turn, $1 - \frac{\beta}{T} + \frac{\beta\delta}{T^2} \geq 1 - \beta + \beta\delta$; (4.21b) is due to that $\lim_{x \rightarrow \infty} (1 - \frac{1}{x})^x = \frac{1}{e}$.

We are particularly interested in large networks with thousands of nodes and strong network connectivity (i.e., d is large), where both analysis and simulation on virus propagations in such networks are prohibitively computationally expensive and time-consuming. In this case,

$$e^{-\frac{\delta(1-\beta+\beta\delta)^d}{\tau_{\min}}} \cong (1 - \delta(1 - \beta + \beta\delta)^d)^{\frac{1}{\tau_{\min}}} \quad (4.22)$$

In other words, the lower bound in (4.19) does not change with the increase of T .

As a result, it is proved that the upper and lower bounds in (4.19) are still valid, especially in large networks with thousands of nodes and strong network connectivity. Both of the bounds depend on the actual temporal characteristics of the infection and cure (i.e., τ_{\min}), rather than the selection of the time-step size. \square

4.3.3 Epidemic Lifetime

Given the virus extinction rate R , the virus lifetime can be measured by $\frac{1}{R}$ [126]. We give an upper bound of the lifetime in the following lemma.

Lemma 1. *The mean virus lifetime in an N -node network with the curing probability δ , denoted by $\mathbb{E}_N\{\tau\}$, is bounded:*

$$\mathbb{E}_N\{\tau\} \leq \frac{1}{\delta^N},$$

which is independent of the initial infection status and the explicit topology of the network.

Proof. The expected lifetime of an epidemic is the expected steps to the virus-free absorbing state in the Markov process, which is equal to the expected times that the Markov chain remains at transient states. With a given initial state $\mathbf{t}^{(0)}$ and

matrix \mathbf{Q} , the expected virus lifetime can be given by

$$\begin{aligned}
\mathbb{E}_N\{\tau|\mathbf{t}^{(0)}, \mathbf{Q}\} &= \sum_{k=0}^{\infty} (\mathbf{1}(\mathbf{t}^{(k)})^T) = \sum_{k=0}^{\infty} (\mathbf{1}(\mathbf{t}^{(0)}\mathbf{Q}^k)^T) \\
&\stackrel{(a)}{\leq} \sum_{k=0}^{\infty} \|(\mathbf{t}^{(0)}\mathbf{Q}^k)^T\|_1 \\
&\stackrel{(b)}{\leq} \sum_{k=0}^{\infty} \|(\mathbf{Q}^k)^T\|_1 \|(\mathbf{t}^{(0)})^T\|_1 \\
&\stackrel{(c)}{\leq} \sum_{k=0}^{\infty} \|\mathbf{Q}^T\|_1^k \stackrel{(d)}{=} \frac{1}{\delta^N}
\end{aligned} \tag{4.23}$$

where $\|\cdot\|_1$ denotes 1-norm. In the case of vector, the 1-norm is the absolute sum of the vector. In the case of matrices, the 1-norm is the maximum absolute column sum of the matrix. The inequality (a) is due to the fact that $\|(\mathbf{t}^{(0)}\mathbf{Q}^k)^T\|_1 = \mathbf{1} |(\mathbf{t}^{(0)}\mathbf{Q}^k)^T| \geq \mathbf{1}(\mathbf{t}^{(0)}\mathbf{Q}^k)^T$. The inequality (b) is because $\|\mathbf{A}\mathbf{v}\|_1 \leq \|\mathbf{A}\|_1 \|\mathbf{v}\|_1$, where \mathbf{A} is a matrix and \mathbf{v} is a vector [124]. The inequality (c) is because $\|(\mathbf{t}^{(0)})^T\|_1 = \mathbf{1}(\mathbf{t}^{(0)})^T = 1$, and $\|(\mathbf{Q}^k)^T\|_1 = \|(\mathbf{Q}^T)^k\|_1 \leq \|(\mathbf{Q}^T)\|_1^k$. The equality (d) is due to the definition of 1-norm that $\|\mathbf{Q}^T\|_1$ is the maximum absolute column sum of \mathbf{Q}^T ; or in other words, $\|\mathbf{Q}^T\|_1$ is the maximum absolute row sum of \mathbf{Q} , i.e. $\|\mathbf{Q}^T\|_1 = \max \{ \sum_j q_{i,j} \}$. From (4.14), we have $\max \{ \sum_j q_{i,j} \} = 1 - \delta^N$. As a result, $\|\mathbf{Q}^T\|_1 = 1 - \delta^N$. \square

4.3.4 Approximate Infected Population and Infection Probability

It is also interesting to evaluate the infected population and the infection probability at every time step of the epidemic lifetime. This helps understand the interactions between the infection and healing processes.

By substituting (4.12) into (4.6), the expected infected population can be approximated, as given by

$$\mathbb{E}\{population|\mathbf{t}^{(0)}, k\} \approx \lambda_1^k (\mathbf{c}(\mathbf{t}^{(0)}\mathbf{G}_1)^T). \tag{4.24}$$

By substituting (4.12) into (4.7), the expected infection probability can be ap-

proximated, as given by

$$\Pr\{infection|\mathbf{t}^{(0)}, k\} \approx \lambda_1^k (\mathbf{1}(\mathbf{t}^{(0)} \mathbf{G}_1)^T). \quad (4.25)$$

From (4.24) and (4.25), we can see that the infected population and the infection probability both decline away exponentially.

4.4 Simulation and Numerical results

In this section, simulations are conducted to validate our proposed model. The simulations are built on the modified NepidemiX 0.2[127] in a Python environment where curing takes place periodically with a time step of 10 ms and there is no consecutive infection during a time step. Our model is general, and can be applicable to other time step durations. Topologies simulated include complete graphs, denoted by \mathcal{K}_N , Barabási-Albert scale-free graphs, denoted by BA- d , ring graphs, and regular graphs, where d is the minimum degree of a BA graph.

For comparison purpose, we also plot the typical results of three existing models. The first existing model that we consider is based on the mean-field approximation, discretized from [35, eq. 10] and [121] to adapt to our discrete system setting. For an N -node network, the infection probabilities of the N nodes at any time step t can be recursively given by

$$\mathbf{V}_t = (\mathbf{V}_{t-1} + \beta(\mathbf{1} - \mathbf{V}_{t-1})\text{diag}(\mathbf{V}_{t-1}\mathbf{A})) (1 - \delta), \quad (4.26)$$

where \mathbf{V}_t is an N -dimensional vector collecting the infection probability of each node at the time step t ; $\text{diag}(\cdot)$ indicates a diagonal matrix. The average number of infected nodes in this model can be given as the column sum of \mathbf{V}_t .

The second existing model that we consider is the one which further linearises the existing model for mathematical tractability, as suggested in [121, eq. 11] and

[35]. As a result, (4.26) can be approximated as

$$\mathbf{V}_t = (1 - \delta)(\mathbf{I} + \beta\mathbf{A})\mathbf{V}_{t-1}, \quad (4.27)$$

where \mathbf{I} is the identity matrix.

The linearisation of (4.27) is important to analyze virus extinctions in our system setting where infection and curing take place in alternating time steps. The existing extinction condition $\frac{\beta}{\delta} < \frac{1}{\rho(\mathbf{A})}$, proved in [121, 35], was based on a different system setting, where infection and curing can take place in any time steps, and hence, the evolution of the infection probability was given by [121, 35]

$$\mathbf{V}_t = ((1 - \delta)\mathbf{I} + \beta\mathbf{A})\mathbf{V}_{t-1}.$$

To this end, the existing extinction condition is not directly applicable to our system setting.

Following the exactly same way the existing extinction condition was derived in [121, 35], we write the largest eigenvalue of (4.27) as $\lambda_1 = (1 - \delta)(1 + \beta\rho(\mathbf{A}))$, set $\lambda_1 < 1$ to ensure the convergence of (4.27), and subsequently, derive the equivalent of the existing extinction condition under our system setting, as given by

$$\frac{\beta(1 - \delta)}{\delta} < \frac{1}{\rho(\mathbf{A})}.$$

In turn, the equivalent upper bound of the mean virus lifetime, when the existing condition is satisfied, can be given by

$$\mathbb{E}\{\tau\} \leq \frac{\log(N) + 1}{1 - \frac{\beta(1 - \delta)}{\delta}\rho(\mathbf{A})}.$$

We also evaluate the classical SIS model [66], in which the number of infected nodes at any time step t can be recursively written, as given by

$$I_t = (I_{t-1} + \beta I_{t-1}(N - I_{t-1}))(1 - \delta), \quad (4.28)$$

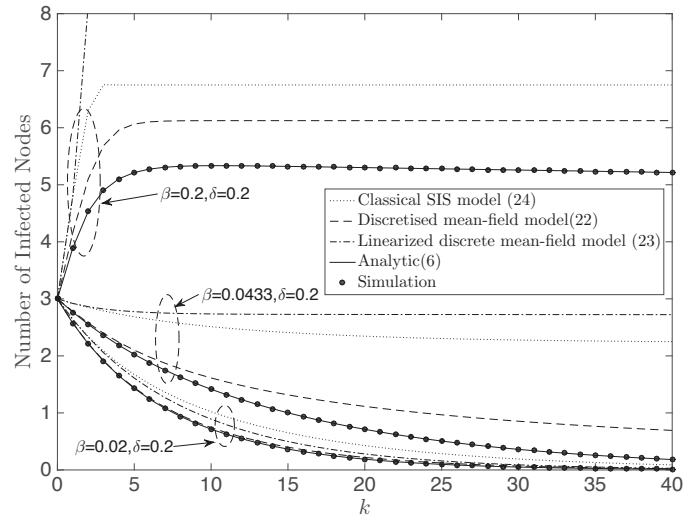
where I_t is the number of infected nodes at time step t .

4.4.1 Model Validation

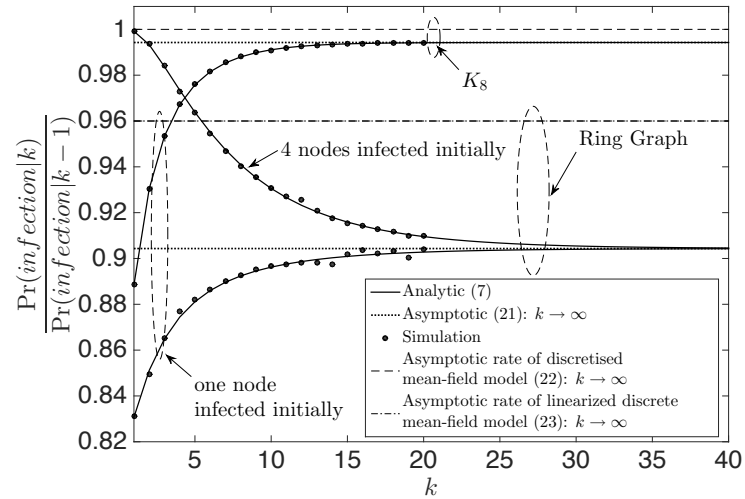
Fig. 4.4 validates the proposed discrete-time absorbing Markov modeling of virus propagations, using Monte-Carlo simulations. Essentially, our Markov model provides basic discrete linear system analysis and requires computations of 2^N by 2^N networks. The purpose of plotting Fig. 4.4 is to validate the model and compare it with the typical existing modeling methods, in terms of accuracy. The validation is important for our development of the bounds for the extinction rate. Also, based on the validated linear system analysis and bounds, we further explore the possibility of finding the extinction rate for large networks by using the results on much smaller sizes, as will be discussed in Section 4.4.2.

Fig. 4.4(a) validates the analytic exact results on the infected population from (4.6) with Monte-Carlo simulations, where there are up to 40 time steps (i.e., 40 incidents of virus propagations). The topology simulated is an 8-node BA-4 graph. The initial states of the Markov processes are the same that three nodes with the degrees of 7, 4 and 4 are initially infected. Different combinations of β and δ are considered. Fig. 4.4(b) validates the analytic results of $\frac{\Pr(\text{infection}|\mathbf{t}^{(0)},k)}{\Pr(\text{infection}|\mathbf{t}^{(0)},k-1)}$ based on (4.7) and $\lambda_1 = \lim_{k \rightarrow \infty} \frac{\Pr(\text{infection}|\mathbf{t}^{(0)},k)}{\Pr(\text{infection}|\mathbf{t}^{(0)},k-1)}$ from (4.25) with Monte-Carlo simulations, where two representative topologies, 8-node complete graph and 8-node ring graph, are simulated. For the complete graph, the initial state of the Markov processes includes one infected node randomly picked up, since the complete graph is homogeneous and each node in the graph is identical. For the ring graphs, there are two cases. In one case, the initial state is a randomly chosen infected node, due to the homogeneity of the graph. In the other case, the initial state consists of four infected nodes arranged with the susceptible nodes in an alternating manner on the ring.

Both Figs. 4.4(a) and 4.4(b) confirm that the analytical results of (4.6) and (4.7) consistently coincide the simulation results, indicating the accuracy of the



(a) Infected population where each of the simulation results is the average of 10000 independent runs.



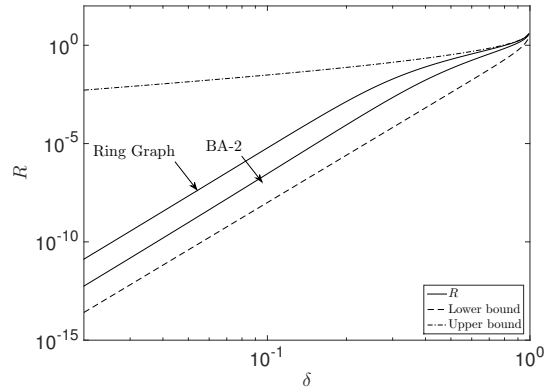
(b) Ratio of infection probability between consecutive time steps, where $\beta = 0.1$ and $\delta = 0.2$. Each of simulation results is the average of 200,000 independent runs.

Figure 4.4 : Model Validation

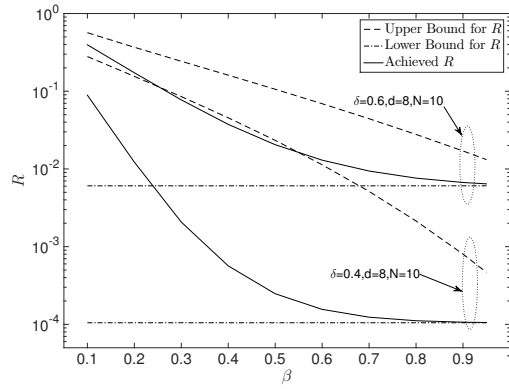
model. Figs. 4.4(a) and 4.4(b) also show that the analytic and simulated results indistinguishably approach to the results of (4.25), validating the extinction rate of $-\ln(\lambda_1)$, i.e., (4.9), under a variety of topologies and β and δ values. In contrast, the results of the existing SIS model and the mean-field approximation based models increasingly deviate from the simulation results, as β rises. Specifically, when β is small (i.e., 0.02) and satisfies the equivalent existing extinction condition under our system setting $\frac{\beta(1-\delta)}{\delta} < \frac{1}{\rho(\mathbf{A})}$, the existing models exhibit relatively accurate exponential extinction tendency. When β increases (i.e., 0.0433) and the extinction condition becomes unsatisfied, the virus still dies out exponentially over time, as shown by simulations. However, the existing models start to fail in capturing the virus propagation. The existing extinction condition is also revealed to be a sufficient condition, but not the necessary condition, to the quick extinction of a virus. When β is large (i.e., 0.2), big gaps can be observed between the existing models and the simulation results, while our model can still provide satisfactory modeling accuracy.

We also see the topology has a strong impact on the convergence rate, even under the same β and δ , as shown in Fig. 4.4(b). Given the initial state of a single infected node, we see that, the convergence rate of the complete graph is lower than that of the ring graph. In other words, λ_1 , plotted as the horizontal dotted lines in Fig. 4.4(b), increases with the network connectivity. As a result, the extinction rate $R = -\ln(\lambda_1)$ decreases as the network connectivity increases. It is further noticed that the extinction rate is independent of the initial state of the network. As shown in Fig. 4.4(b), the ring topologies with different initial infected nodes both converge to the extinction rate specified by (4.25).

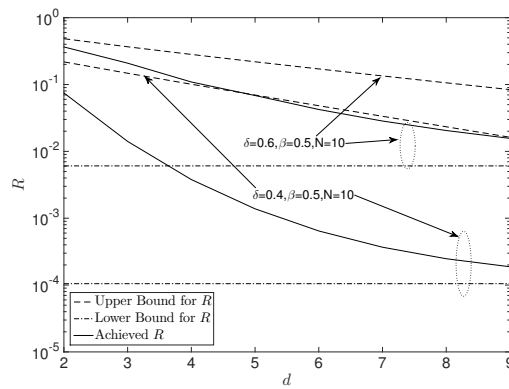
Fig. 4.5(a) validates the upper and lower bounds of the virus extinction rate, where the extinction rate is obtained from (4.9) and its bounds are obtained from Theorem 6. β is set to 0.5 and $0 < \delta \leq 1$. We consider an 8-node ring graph and an 8-node BA-2 graph (as illustrated in Fig.4.1), because the two topologies have



(a) The rate R and its bounds, as δ grows, where $\beta = 0.5$ and $0 < \delta \leq 1$.



(b) The rate R and its bounds, as β grows.



(c) The rate R and its bounds, as d grows.

Figure 4.5 : Evaluation of the proposed bounds of the virus extinction rate, where BA-2, ring and regular graphs are considered

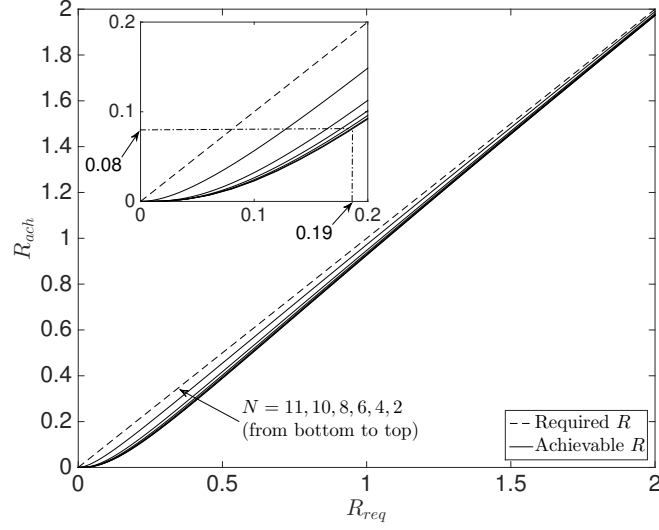


Figure 4.6 : The actual achievable convergence rate R vs. the convergence rate requirement where complete graphs of N nodes are considered ($N = 2, 4, 6, 8, 10, 11$), $\beta = 0.5$ and δ is adaptively calculated using the upper bound of Theorem 6 under any given convergence rate requirement.

the same network size and the same minimum degree, so that they have the same bounds with the benefit of visual clarity. We can see that the difference between the logarithms of the upper and lower bounds (i.e., $\ln \frac{-\ln(1-\delta(1-\beta+\beta\delta)^d)}{-\ln(1-\delta^N)}$) decreases with the growth of δ . Specifically, the upper bound is 12 orders of magnitude higher than the lower bound when $\delta = 0.02$, while the bounds converge at $\delta = 1.0$. The virus extinction rates of the topologies are both within the region specified by the upper and the lower bounds. We also observe when δ is small, the extinction rate is close to the lower bound and the ratio of R to the lower bound keeps consistent across a wide δ region. The ratio decreases as the connectivity of the graphs grows (i.e., from the ring to the BA-2 graph). When δ is approaching one, the actual virus extinction rates approach to the upper bound and become indistinguishable close to the bound.

Likewise, we also evaluate the bounds with the increase of β and d in Figs. 4.5(b) and 4.5(c), respectively, where 10-node networks with topologies of regular graph are considered. We can see that our derived bounds become increasingly tight with the growth of β and δ , where $d = 8$, as shown in Fig. 4.5(b). We also see that the bounds also become increasingly tight, as the network connectivity, i.e., network degree, grows, as corroborated in Fig. 4.5(c).

We proceed to evaluate the effectiveness of the bounds, presented by Theorem 6, in characterizing the extinction rate. Specifically, for any given rate requirement R_{req} , we first calculate the minimum δ , denoted by δ_{min} , required to meet R_{req} using the upper bound. This is done by setting R_{req} to be equal to the upper bound, i.e.,

$$R_{req} = -\ln(1 - \delta_{min}(1 - \beta + \beta\delta_{min})^d), \quad (4.29)$$

and solve δ_{min} by using a bisection method. We then substitute δ_{min} into (4.2) and (4.9) to obtain the actually achievable convergence rate R_{ach} under the minimum curing probability δ_{min} . In this sense, R_{ach} provides the lower bound to the extinction rate requirement R_{req} .

Fig. 4.6 plots the actually achievable convergence rate R_{ach} with the increase of the convergence rate requirement R_{req} , where the auxiliary dash line, $R_{ach} = R_{req}$, sets up the reference for the actually achievable rate R_{ach} . A key finding from Fig. 4.6 is that, for a given network the gap between R_{ach} and the corresponding requirement R_{req} , i.e., the reference dash line, first increases, and it soon decreases and diminishes asymptotically as R_{req} grows. The asymptotic convergence of R_{ach} to the dash line indicates the effectiveness of δ_{min} and subsequently the tightness of the upper bound provided in Theorem 6. Another important finding is that the gap between R_{ach} and R_{req} also enlarges along with the number of nodes, while the increase of the gap, resulting from the enlarged network, diminishes when the network becomes large, with δ_{min} adjusted by using the upper bound of Theorem

6. Note here that δ_{min} needs to be adjusted, since the minimum degree of the network d can change with the growth of the network. As shown in Fig. 4.6, the difference between a 10-node complete graph and an 11-node complete graph is indistinguishable. As a matter of fact, extensive simulations we carried out with larger N values show that the curves for $N \geq 11$ overlap.

4.4.2 Large Network Analysis

Our above findings from Fig. 4.6 provide a practical means to analyze virus propagations in large computer networks, where other methods become either computationally prohibitive or incur accuracy degradations. As observed in Fig. 4.6, the $R_{ach}-R_{req}$ curves become indistinguishably close for networks with $N \geq 11$. In other words, the gap between the desired extinction rate R_{req} and the actual achieved rate R_{ach} converges with the growth of the network, given δ_{min} adjusted for the size and topology of individual networks through the upper bound of the extinction rate. In light of this, we propose to use the $R_{ach}-R_{req}$ curve for $N = 11$ to present those for $N \gg 11$, and convert the target mean achievable extinction rate of a large network R_{ach} to the requirement R_{req} by mapping onto the curve. Substituting R_{req} into (4.29), we can obtain δ_{min} for the large network to actually achieve R_{ach} .

This process is magnified in Fig. 4.6, where a target achievable mean extinction rate is $R_{ach} = 0.08$ for a 5000-node network with topology of complete graph (i.e., $d = 4999$). As described above, we use the curve for an 11-node network to convert the target mean achievable extinction rate $R_{ach} = 0.08$ to the corresponding requirement R_{req} . The corresponding requirement is $R_{req} = 0.19$, as shown in Fig. 4.6. We then substitute $R_{req} = 0.19$ into (4.29) to calculate δ_{min} under $N = 5000$. The resultant δ_{min} can hence achieve the mean extinction rate of 0.08 in the 5000-node network.

Fig. 4.7 evaluates the effectiveness of using the process to design and analyze

virus propagations in large computer networks, where the x -axis indicates the target mean achievable extinction rate R_{ach} and the y -axis indicates the actually achieved extinction rate. The network scale is $N = 5000$. $\beta = 0.5$. Every mark in Fig. 4.7 is a simulation result, obtained by first converting R_{ach} indicated by the x -coordinate of the mark to R_{req} (as demonstrated in Fig. 4.6), adjusting δ_{min} , and then emulating the virus propagation in the network with the resultant δ_{min} and the given β . The dash auxiliary line plots R_{ach} to provide a reference. Fig. 4.7 shows that the simulation results of the actually achieved individual extinction rates are consistently distributed around the target mean extinction rate R_{ach} . The deviations of the simulation results from R_{ach} yield a Gaussian-type distribution.

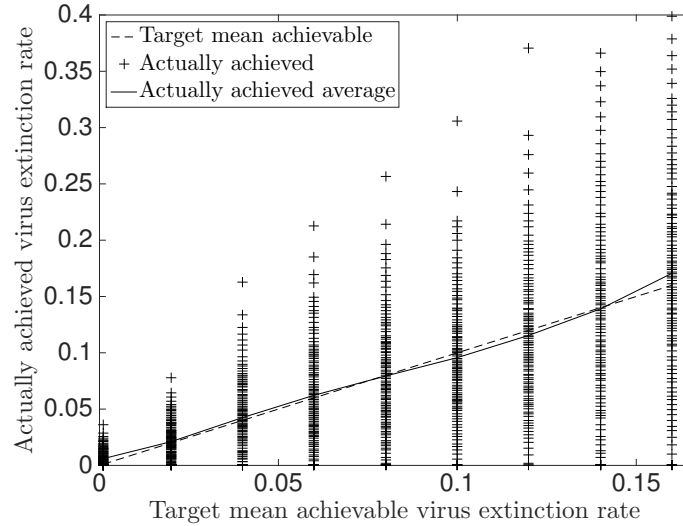


Figure 4.7 : The actually achieved extinction rate versus the target mean extinction rate, where a 5000-node network with topology of complete graph \mathcal{K}_{5000} is considered, and $\beta = 0.5$.

We note that the value of δ_{min} ranges from 0.9986 to 0.9994 in Fig. 4.7, given the target R_{ach} ranges from 0.001 to 0.16 and $\beta = 0.5$. The reason for such large δ_{min} values is because of the fairly high target mean extinction rate of $R_{ach} \geq 0.001$ in the large network with 5000 nodes. Reducing the target mean extinction rate would

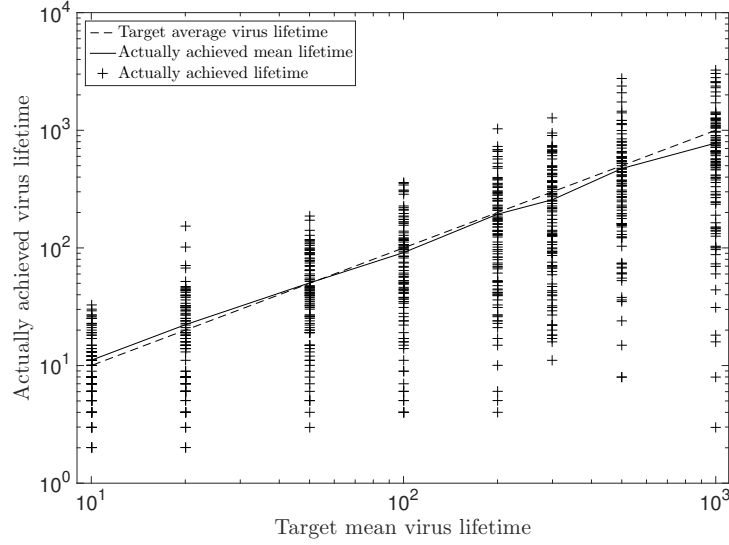


Figure 4.8 : The actually achieved virus lifetime versus the target mean virus lifetime, where a 5000-node network with topology of \mathcal{K}_{5000} is considered and $\beta = 0.5$.

of course reduce δ_{\min} . The accuracy and complexity of our large-network analysis would not be much affected by the reduced rate. However, reducing the rate would increase the virus lifetime, and in turn, substantially increase the running time of Monte-Carlo simulations to get reliable and conclusive results. Even in the case that $R_{ach} = 0.001$ and $\delta_{\min} = 0.9986$, our simulations for the 5000-node network took a few days. Let alone smaller target extinction rates or δ_{\min} . For demonstration purpose, we consider large δ_{\min} to speed up simulations.

Fig. 4.8 plots the results of the virus lifetime in the 5000-node complete graph, where the x -axis is the average number of steps (i.e., $\frac{1}{R_{ach}}$) that we plan to achieve, and y -axis is the number of steps actually achieved. As done in Fig. 4.7, we first map R_{ach} to R_{req} to calculate δ_{\min} ; and then emulate virus propagations using δ_{\min} , until the extinction of the virus. The simulation results of individual virus propagations are given in marks. We see that the simulation results are consistently distributed around the target mean virus lifetime denoted by dash line. The deviations of the simulation results from the design target also yield a Gaussian-type distribution.

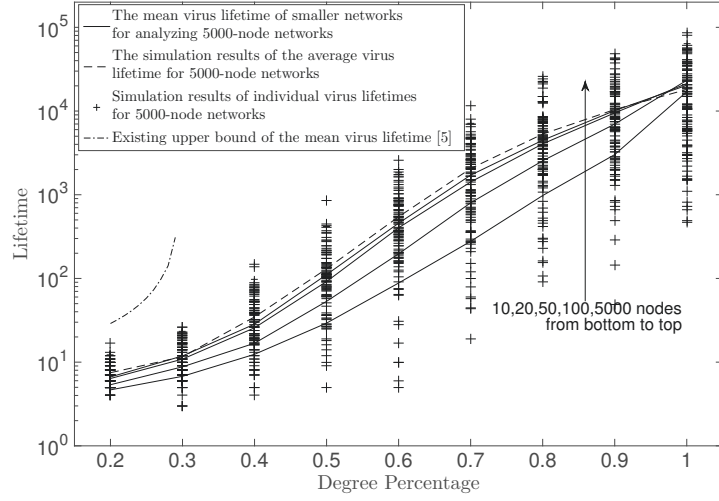


Figure 4.9 : Virus lifetime versus network connectivity, where a 5000-node network with topology of regular graph is considered, the degree of the regular graph increases from 20% to 100% of the remaining nodes (e.g., d ranges from 1000 to 4999 for 5000-node networks).

The above analysis of large networks can also be readily applied under non-trivial topologies (other than complete graphs). In Fig. 4.9, we show the effectiveness of the analysis in a 5000-node network ($N = 5000$) with topology of regular graph, where the x -axis provides the increasing degree of the network from $d = 20\%N$ to $d = N - 1$ (i.e., complete graph), $\beta = 0.3$, and $\delta = 0.9978$. The reason for choosing regular graphs is because the regular structure of the graphs can support the consistent growth of degrees without bringing in irregularity, and in turn, provide consistent results to reveal useful insights. The reason for considering such a large δ value here is the same as that for Fig. 4.7.

The marks in Fig. 4.9 provide the lifetime simulations of individual independent virus propagations given β and δ , with their averages plotted in the dash curve. Using the averaged lifetimes, we set the target mean extinction rates R_{ach} for our analysis under different degrees of the regular graph. We map the target mean

extinction rates R_{ach} onto the R_{ach} - R_{req} curves corresponding to smaller networks with $N = 10, 20, 50$, and 100 ; see Fig. 4.6. R_{req} can be obtained and substituted in (4.29) to evaluate δ_{min} . The actually mean achievable extinction rates of the smaller networks with δ_{min} , denoted by R'_{ach} to avoid confusion, are plotted in solid curves.

We can see the solid curves get increasingly close to the simulation results of the average virus lifetime (i.e., the design target of the solid curves), as N increases. A clear tendency is exhibited that the curves with N ranging from 50 to 100 start to converge to the simulation results of 5000-node networks. In other words, we can use the analytical results of a network with up to 100 nodes to evaluate the virus lifetime in large networks with thousands of nodes and non-trivial topologies.

In Fig. 4.9, we also plot the equivalent upper bound of the mean virus lifetime for our system setting $\frac{\log(N)+1}{1-\frac{\beta(1-\delta)}{\delta}\rho(\mathbf{A})}$, under the extinction condition $\frac{\beta(1-\delta)}{\delta} < \frac{1}{\rho(\mathbf{A})} = \frac{1}{d}$, by referring to [120]. We see that the typical extinction condition has limited applicability, and is only suitable for topologies with low connectivity (i.e., the degree percentage is less than 30%). Even for the topologies, to which the extinction condition is applicable, the existing upper bound of the mean virus lifetime is loose, and becomes increasingly looser as the network connectivity improves.

It is worth mentioning that some new arrangements have been implemented to facilitate plotting Fig. 4.9, under the current limitation of hardware and computing capability. As described earlier, an important step of the large-network analysis is to obtain the convergent achievable mean extinction rate R_{ach} for networks of small to medium sizes (or in Fig. 4.9, plotting R'_{ach}). The size of the network, when the gap between the required mean extinction rate R_{req} and the achievable mean extinction rate R_{ach} converges, depends on β , δ , and the topology of the network.

For complete graphs, the network size of 11 nodes sees the convergence, as shown in Fig. 4.6, and R_{ach} can be achieved by analytically evaluating the second largest

Table 4.2 : The average running time of simulating a single virus propagation process (in seconds), corresponding to Fig. 4.9

Degree d	$0.3N$	$0.6N$	$0.9N$
$N = 10$	0.0003	0.00392	0.13417
$N = 20$	0.00068	0.01531	0.54328
$N = 50$	0.00107	0.02406	0.85372
$N = 100$	0.00213	0.08243	1.9125
$N = 5000$	1.6770	78.8060	1505.1299

eigenvalue of the transition matrix of the discrete-time absorbing Markov virus propagation process, as done in Figs. 4.7 and 4.8.

For other non-trivial topologies, such as regular graphs, the network size can be tens to hundreds. The transition matrices for the networks are too large and the eigenvalue decomposition of the matrices is intractable under current computing capability. In this case, we resort to Monte-Carlo simulations to evaluate the average extinction rates of networks with from tens to hundreds of nodes. We first identify the average extinction rate which stops growing with the network size, and then use this rate as R_{ach} in our analysis on the large networks with non-trivial topologies in the same way as in Fig. 4.8. The Monte-Carlo simulations involved for networks with up to a hundred nodes are fast and can complete within a few hours (e.g., $N = 100$, $d = 90$, and 5000 runs), as opposed to directly simulate for months a large network with thousands of nodes (e.g., $N = 5000$, $d = 4500$, and 5000 runs if not more), as evident from Table 4.2.

4.5 Summary

In this chapter, we designed a discrete-time absorbing Markov process to characterize virus propagations in computer networks. Eigenvalue analysis and Jordan decompositions were carried out on the transition matrix of the Markov process to derive the bounds of the virus extinction rate. Based on the bounds, we also developed a computationally efficient approach to evaluating virus lifetimes in large networks. Specifically, we proposed to interpret the required extinction rate of a large network to that of a much smaller network, and obtain the required minimum curing rate of the large network efficiently through the upper bound of its smaller counterpart. Simulation results corroborated the effectiveness of the interpretation, as well as analytical accuracy in large computer networks.

Chapter 5

Group-based Fast Propagation Model

An accurate Markov-based propagation model was proposed in the last chapter. Although an extension method has been proposed and evaluated, the model still needs hours of evaluation time and can be improved. This chapter proposes a fast propagation model by adopting the group feature of network topology. By dividing network nodes into groups and averaging them with the mean-field approximation, the model is greatly simplified and therefore can support the propagation analysis in large-scale networks.

5.1 Introduction

Epidemic models trade the modeling accuracy for complexity reduction. Markov-chain based epidemic models are able to precisely analyze the epidemic propagation process, but require 2^N Markov states to capture the S/I states of N nodes and therefore can hardly be applied to large-scale networks [110, 35]. A number of topological epidemic models [121, 128, 35, 120, 129], decompose the 2^N -state Markov process into N small Markov processes. This is achieved by using the expected infection probability of every node instead of the actual node state. A significant result from the models is that the epidemic threshold, under which the epidemic will eventually die out, is given by $\frac{1}{\lambda_1(\mathbf{A})}$, where $\lambda_1(\mathbf{A})$ is the largest eigenvalue of the adjacency matrix of the network topology [121].

Network features, e.g., the degree distribution of scale-free networks [130], can be employed to simplify epidemic models by adopting the degree-based mean-field

approach [88]. Specifically, nodes with the same degree are assumed to be infected with the same probabilities. An interesting result of the epidemic in scale-free networks is the absence of an epidemic threshold [73]. The degree-based mean-field approach has extended the SIS process, e.g., the epidemic propagation with incubation and the epidemic with a recovery state [131, 132]. However, the degree-based epidemic model cannot capture epidemic propagations in specific networks.

This chapter presents a group-based continuous-time Markov model to quantitatively analyze the SIS process in large-scale directed networks. We start with the network modeling, where nodes are categorized into groups according to their connectivity. A collapsed adjacency matrix is proposed to describe the network topology. Based on the node groups, a continuous-time Markov model is proposed to capture the SIS-type propagation, where the state of a group is estimated by taking the mean-field approximation. Focusing on the problem of epidemic threshold, the proposed nonlinear model is linearized via omitting high-order terms around the disease-free point. The epidemic threshold is derived by performing matrix analysis on the Jacobian matrix of the linearized model and then validated by simulations. The key contributions of this chapter can be summarized as follows,

- We propose a new modeling method, which groups nodes with the same connectivity in directed networks and models the epidemic propagation of the groups by using continuous-time Markov SIS models.
- By taking the mean-field approximation, the proposed SIS model is asymptotically accurate with the decrease of effective spreading rate and/or the increase of node groups.
- Linearization and stability analysis are carried out on the proposed SIS model to deduce the epidemic threshold, under which the epidemic eventually becomes extinct.

- The epidemic threshold is proved to be dependent on network structure, and interdependent of the network scale.

Comprehensive simulations confirm the validity of the proposed mean-field epidemic model and the deduced epidemic threshold in large-scale networks.

The rest of this chapter is organized as follows. In Section 5.2, the directed network model is presented, followed by the proposed mean-field SIS model in Section 5.3. In Section 5.4, numerical and simulation results are provided, followed by conclusions in Section 5.5.

Notations used in the chapter are as listed in Table 5.1.

5.2 The Directed Network Model

We consider the SIS epidemic process in a strongly connected network with N nodes connected by directed edges. Each node in the network can be in either a susceptible (S) or an infected (I) state. An infected node can infect its susceptible neighbors along the directed edges at the rate of β per edge. Infected nodes can independently recover to be susceptible at the rate of δ .

We suppose that the N nodes in a network \mathcal{G} can be categorized into n groups, denoted by G_1, G_2, \dots, G_n , where the nodes in the same group have the same out-degrees and the same number of edges to the nodes in the same destination group. The number of G_i -nodes can be denoted by N_i (here, $\sum_i N_i = N$). Given the node groups, we define a collapsed adjacency matrix, denoted by \mathcal{A} , to describe the topology of \mathcal{G} . The (i, j) -th entry of the matrix, denoted by a_{ij} , describes the number of edges from a G_i -node pointing to G_j -nodes. As a result, \mathcal{G} can be described by the node number vector $\mathcal{N} = [N_1, N_2, \dots, N_n]$ and the collapsed adjacency matrix \mathcal{A} . Fig. 5.1 provides an example of node categorization, where $N = 7$ nodes are categorized into $n = 3$ groups, i.e., $\mathcal{N} = [2, 3, 2]$. Its collapsed adjacency matrix

Table 5.1 : Notations Used in Chapter 5

Notation	Description
N	The number of nodes in the network
n	The number of groups in the network
G_i	The i -th group
N_i	The number of nodes in G_i
\mathcal{A}	The collapsed adjacency matrix
a_{ij}	The number of edges from a G_i -node pointing to G_j -nodes
$[A]_i$	The number of G_i -nodes in state A
$[AB]_{ij}$	The number of AB_{ij} edges
$[ABC]_{ijk}$	The number of edge pairs consisting of AB_{ij} and BC_{jk}
$[ABC]'_{ijk}$	The number of edge pairs consisting of AB_{ij} and BC_{jk}
β	The infection rate per edge
δ	The curing rate of an infected node
\mathbf{J}	The Jacobian matrix of the linearization
$\lambda_1(\cdot)$	The largest eigenvalue of the operator
τ^*	the epidemic threshold

can be given by (5.1).

$$\mathcal{A} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} = \begin{bmatrix} 0 & 3 & 0 \\ 0 & 1 & 2 \\ 1 & 0 & 0 \end{bmatrix} \quad (5.1)$$

Other notations are defined as follows: $[A]_i$ ($A \in \{S, I\}$) denotes the number of G_i -nodes in state A . AB_{ij} denotes an edge starting from a G_i -node and ending at a G_j -node, where the G_i -node and the G_j -node are in states A and B , respectively.

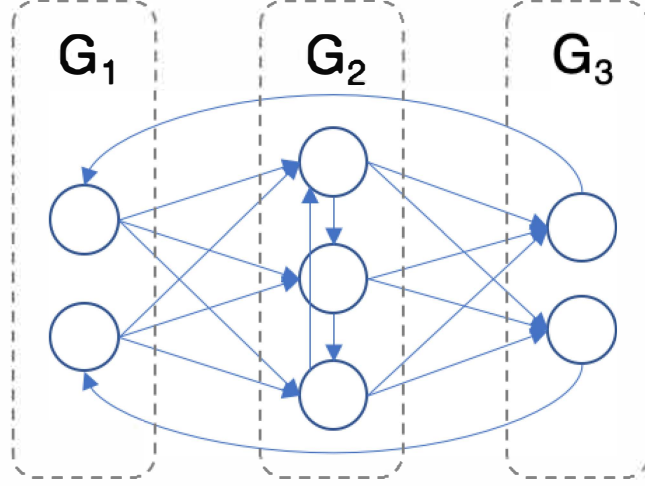


Figure 5.1 : An example of networks we considered where 7 nodes are categorized into three groups according to the connection between them.

$[AB]_{ij}$ denotes the number of AB_{ij} edges. Let $[ABC]_{ijk}$ ($[ABC]_{ijk}'$) denote the number of edge pairs consisting of AB_{ij} and BC_{jk} (AB_{ij} and CB_{kj}), as illustrated by Fig. 5.2. Numerical relationships between states of nodes and states of edges satisfy

$$N_i = [S]_i + [I]_i; \quad (5.2a)$$

$$a_{ij}N_i = [SS]_{ij} + [SI]_{ij} + [IS]_{ij} + [II]_{ij}; \quad (5.2b)$$

$$a_{ij}[S]_i = [SS]_{ij} + [SI]_{ij}; \quad (5.2c)$$

$$a_{ij}[I]_i = [IS]_{ij} + [II]_{ij}; \quad (5.2d)$$

$$[S]_i = N_i - [I]_i = N_i - \frac{1}{a_{ij}}([IS]_{ij} + [II]_{ij}). \quad (5.2e)$$

Here, (5.2a) is because any node is in either an S or I state. (5.2b) is because any edge is in one of the four states given in the right-hand side (RHS) of (5.2b). Meanwhile, edges can be classified according to the state of the starting point, as given by (5.2c) and (5.2d). (5.2e) is deduced from (5.2a), (5.2b) and (5.2d).

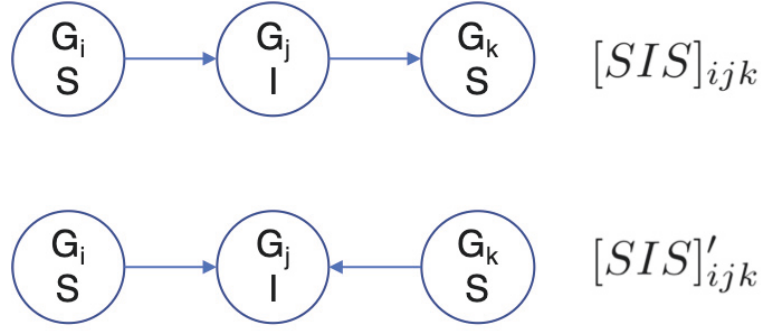


Figure 5.2 : An example of $[ABC]_{ijk}$ and $[ABC]'_{ijk}$.

5.3 Group-based Mean-field SIS Model

We propose to analyze the SIS process in directed networks by employing the mean-field approximation which uses a single average effect to approximate the effect of all the other individuals on any given individual. Thus, the same group of nodes in our model are evaluated by the same average estimation. The state transitions of nodes and edges in the SIS process can be given by

$$\frac{d[S]_i}{dt} = \delta[I]_i - \sum_j \beta[IS]_{ji}; \quad (5.3a)$$

$$\frac{d[I]_i}{dt} = -\delta[I]_i + \sum_j \beta[IS]_{ji}; \quad (5.3b)$$

$$\frac{d[SS]_{ij}}{dt} = \delta[SI]_{ij} + \delta[IS]_{ij} - \sum_k \beta[ISS]_{kij} - \sum_k \beta[SSI]'_{ijk}; \quad (5.3c)$$

$$\frac{d[SI]_{ij}}{dt} = -\delta[SI]_{ij} + \delta[II]_{ij} + \sum_k \beta[SSI]'_{ijk} - \sum_k \beta[ISI]_{kij}; \quad (5.3d)$$

$$\frac{d[IS]_{ij}}{dt} = -\delta[IS]_{ij} + \delta[II]_{ij} - \beta[IS]_{ij} + \sum_k \beta[ISS]_{kij} - \sum_k \beta[ISI]'_{ijk}; \quad (5.3e)$$

$$\frac{d[II]_{ij}}{dt} = -2\delta[II]_{ij} + \beta[IS]_{ij} + \sum_k \beta[ISI]_{kij} + \sum_k \beta[ISI]'_{ijk}. \quad (5.3f)$$

Here, (5.3a) and (5.3b) give the changing rate of susceptible and infected G_i -nodes. The RHS is because an infected G_i -node can be cured with the rate δ ; a susceptible G_i -node can be infected with the rate β per edge by an infected G_j neighbor. In the continuous-time model, the time slot is infinitesimal that the infection rate can

be summed together, i.e., $\sum_j \beta[IS]_{ji}$.

Eqs. (5.3c)-(5.3f) capture the time-varying number of links. The first two terms on the RHS of (5.3c) are because an SS_{ij} edge can transfer from an SI_{ij} or IS_{ij} edge when the infected node is cured. The last two terms on the RHS of (5.3c) capture the cases where the starting G_i -node or the ending G_j -node is infected by its infected neighbors. (5.3d) is because an SI_{ij} edge can transfer to an SS_{ij} edge in the case that the infected G_j -node is cured at the rate δ ; and to II_{ij} in the case that the susceptible G_i -node is infected by its infected neighbors. SI_{ij} can transfer from II_{ij} in the case that the infected G_i -node is cured with the rate δ or from SS_{ij} in the case that the susceptible G_j -node is infected by its infected neighbors. Different from previous SIS models in undirected networks, e.g., [73], SI_{ij} cannot transfer to II_{ij} , as the epidemic can only propagate along directed edges. (5.3e) can be similarly obtained attaching the infection process, i.e., $-\beta[IS]_{ij}$. (5.3f) is self-explanatory.

Known as the disease-free equilibrium point, the equilibrium point of interest is $([SI]_{ij}, [IS]_{ij}, [II]_{ij}) = (\mathbf{0}, \mathbf{0}, \mathbf{0})$, i.e., all nodes are susceptible. The condition of the disease-free equilibrium point can be deduced by linearizing the SIS model given by (5.3). This is because the stability of the original nonlinear system can be determined by the eigenvalues of the linearized model as stated in Lyapunov's First Method [133].

To linearize the SIS model, the number of edge pairs $[ABC]_{ijk}$ and $[ABC]'_{ijk}$ is first estimated by using the number of unpaired edges. This is achieved by applying the moment closure approximation as evaluated in [134, 135]. As a result, we have

$$[ISS]_{kij} = \frac{[IS]_{ki}[SS]_{ij}}{[S]_i} = \frac{[IS]_{ki}[SS]_{ij}}{N_i - \frac{[IS]_{ip}+[II]_{ip}}{a_{ip}}}; \quad (5.4a)$$

$$[SSI]_{ijk}' = \frac{[IS]_{kj}[SS]_{ij}}{[S]_j} = \frac{[IS]_{kj}[SS]_{ij}}{N_j - \frac{[IS]_{jq}+[II]_{jq}}{a_{jq}}}; \quad (5.4b)$$

$$[ISI]_{kij} = \frac{[IS]_{ki}[SI]_{ij}}{[S]_i} = \frac{[IS]_{ki}[SI]_{ij}}{N_i - \frac{[IS]_{ip}+[II]_{ip}}{a_{ip}}}; \quad (5.4c)$$

$$[ISI]_{ijk}' = \frac{[IS]_{ij}[IS]_{kj}}{[S]_j} = \frac{[IS]_{ij}[IS]_{kj}}{N_j - \frac{[IS]_{jq}+[II]_{jq}}{a_{jq}}}. \quad (5.4d)$$

In (5.4a), every susceptible G_i -node on average has $\frac{[SS]_{ij}}{[S]_i}$ edges pointing to G_j -nodes. $[S]_i$ is then estimated by employing (5.2e), where p and q are introduced to solve the equation. Here, p satisfies $[IS]_{ip}+[II]_{ip} > 0$ and $a_{ip} > 0$; q satisfies $[IS]_{jq}+[II]_{jq} > 0$ and $a_{jq} > 0$. (5.4b)-(5.4d) can be similarly obtained.

By substituting (5.4) into (5.3), (5.3c)-(5.3f) can be rewritten as

$$\begin{aligned} \frac{d[SS]_{ij}}{dt} = & \delta[SI]_{ij} + \delta[IS]_{ij} - \sum_k \beta \frac{[IS]_{ki}[SS]_{ij}}{N_i - \frac{[IS]_{ip}+[II]_{ip}}{a_{ip}}} \\ & - \sum_k \beta \frac{[IS]_{kj}[SS]_{ij}}{[S]_j}; \end{aligned} \quad (5.5a)$$

$$\begin{aligned} \frac{d[SI]_{ij}}{dt} = & -\delta[SI]_{ij} + \delta[II]_{ij} + \sum_k \beta \frac{[IS]_{kj}[SS]_{ij}}{N_j - \frac{[IS]_{jq}+[II]_{jq}}{a_{jq}}} \\ & - \sum_k \beta \frac{[IS]_{ki}[SI]_{ij}}{N_i - \frac{[IS]_{ip}+[II]_{ip}}{a_{ip}}}; \end{aligned} \quad (5.5b)$$

$$\begin{aligned} \frac{d[IS]_{ij}}{dt} = & -\delta[IS]_{ij} + \delta[II]_{ij} - \beta[IS]_{ij} \\ & + \sum_k \beta \frac{[IS]_{ki}[SS]_{ij}}{N_i - \frac{[IS]_{ip}+[II]_{ip}}{a_{ip}}} - \sum_k \beta \frac{[IS]_{ij}[IS]_{kj}}{N_j - \frac{[IS]_{jq}+[II]_{jq}}{a_{jq}}}; \end{aligned} \quad (5.5c)$$

$$\begin{aligned} \frac{d[II]_{ij}}{dt} = & -2\delta[II]_{ij} + \beta[IS]_{ij} + \sum_k \beta \frac{[IS]_{ki}[SI]_{ij}}{N_i - \frac{[IS]_{ip}+[II]_{ip}}{a_{ip}}} \\ & + \sum_k \beta \frac{[IS]_{ij}[IS]_{kj}}{N_j - \frac{[IS]_{jq}+[II]_{jq}}{a_{jq}}}. \end{aligned} \quad (5.5d)$$

The terms of $[SS]_{ij}$ can be suppressed by substituting (5.2b) into (5.5). As a

result, we have

$$\begin{aligned} \frac{d[SI]_{ij}}{dt} = & -\delta[SI]_{ij} + \delta[II]_{ij} + \sum_k \beta \frac{[IS]_{kj}(a_{ij}N_i - [SI]_{ij} - [IS]_{ij} - [II]_{ij})}{N_j - \frac{1}{a_{jq}}([IS]_{jq} + [II]_{jq})} \\ & - \sum_k \beta \frac{[IS]_{ki}[SI]_{ij}}{N_i - \frac{1}{a_{ip}}([IS]_{ip} + [II]_{ip})}; \end{aligned} \quad (5.6a)$$

$$\begin{aligned} \frac{d[IS]_{ij}}{dt} = & -\delta[IS]_{ij} + \delta[II]_{ij} - \beta_{ij}[IS]_{ij} + \sum_k \beta \frac{[IS]_{ki}(a_{ij}N_i - [SI]_{ij} - [IS]_{ij} - [II]_{ij})}{N_i - \frac{1}{a_{ip}}([IS]_{ip} + [II]_{ip})} \\ & - \sum_k \beta \frac{[IS]_{ij}[IS]_{kj}}{N_j - \frac{1}{a_{jq}}([IS]_{jq} + [II]_{jq})}; \end{aligned} \quad (5.6b)$$

$$\begin{aligned} \frac{d[II]_{ij}}{dt} = & -2\delta[II]_{ij} + \beta[IS]_{ij} \\ & + \sum_k \beta \frac{[IS]_{ki}[SI]_{ij}}{N_i - \frac{1}{a_{ip}}([IS]_{ip} + [II]_{ip})} + \sum_k \beta \frac{[IS]_{ij}[IS]_{kj}}{N_j - \frac{1}{a_{jq}}([IS]_{jq} + [II]_{jq})}. \end{aligned} \quad (5.6c)$$

Near the disease-free equilibrium point, (5.6) can be linearized by suppressing all higher order terms. As a result, we have

$$\frac{d[SI]_{ij}}{dt} \approx -\delta[SI]_{ij} + \delta[II]_{ij} + \sum_k \frac{\beta a_{ij} N_i [IS]_{kj}}{N_j}; \quad (5.7a)$$

$$\frac{d[IS]_{ij}}{dt} \approx -(\delta + \beta)[IS]_{ij} + \delta[II]_{ij} + \sum_k \beta a_{ij} [IS]_{ki}; \quad (5.7b)$$

$$\frac{d[II]_{ij}}{dt} \approx -2\delta[II]_{ij} + \beta[IS]_{ij}. \quad (5.7c)$$

After the model has been linearized, the condition of the disease-free equilibrium point can be obtained by performing eigenvalue analysis on the Jacobian matrix of the linearization. The Jacobian matrix is a $3n^2 \times 3n^2$ matrix and denoted by \mathbf{J} . The nonlinear dynamic system is stable at the equilibrium point if and only if all the eigenvalues of the Jacobian matrix are negative, as stated by the Hartman-Grobman Theorem [136]. In other words, the epidemic is certainly extinct if

$$\lambda_1(\mathbf{J}) < 0, \quad (5.8)$$

where $\lambda_1(\cdot)$ is the largest eigenvalue of the operator. \mathbf{J} is the Jacobian matrix of

(5.7) and can be given by

$$\mathbf{J} = \left[\begin{array}{c|cc} \mathbf{J}^{11} & \mathbf{J}^{12} & \mathbf{J}^{13} \\ \hline \mathbf{J}^{21} & \mathbf{J}^{22} & \mathbf{J}^{23} \\ \hline \mathbf{J}^{31} & \mathbf{J}^{32} & \mathbf{J}^{33} \end{array} \right] = \left[\begin{array}{c|cc} \mathbf{J}^{11} & \mathbf{J}^{12} & \mathbf{J}^{13} \\ \hline \mathbf{0} & & \mathbf{J}^* \end{array} \right], \quad (5.9)$$

where \mathbf{J}^{11} is an $n^2 \times n^2$ diagonal matrix whose diagonal entries are $-\delta$. We note that \mathbf{J}^{21} and \mathbf{J}^{31} are zero matrices. As a result, \mathbf{J} is an upper block triangular matrix, and

$$\lambda(\mathbf{J}) = \lambda(\mathbf{J}^{11}) \cup \lambda(\mathbf{J}^*), \quad (5.10)$$

where $\lambda(\cdot)$ is the set of eigenvalues of the operator.

Here, \mathbf{J}^{23} , \mathbf{J}^{32} and \mathbf{J}^{33} are $n^2 \times n^2$ diagonal matrices. Their diagonal entries are δ , β and -2δ , respectively. The entry in the $(n(i-1)+j)$ -th row, $(n(k-1)+l)$ -th column of \mathbf{J}^{22} , denoted by $J_{ij,kl}^{22}$, is given by

$$J_{ij,kl}^{22} = \begin{cases} a_{ij}\beta - (\delta + \beta), & \text{if } ij = kl, i = l \\ -(\delta + \beta), & \text{if } ij = kl, i \neq l \\ a_{ij}\beta, & \text{if } ij \neq kl, i = l \\ 0, & \text{if } ij \neq kl, i \neq l \end{cases}. \quad (5.11)$$

By employing the Schur complement, i.e.,

$$\begin{bmatrix} \mathbf{J}^{22} & \mathbf{J}^{23} \\ \mathbf{J}^{32} & \mathbf{J}^{33} \end{bmatrix} \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ -(\mathbf{J}^{33})^{-1}\mathbf{J}^{32} & \mathbf{I} \end{bmatrix} = \begin{bmatrix} \mathbf{H} & \mathbf{J}^{23} \\ \mathbf{0} & \mathbf{J}^{33} \end{bmatrix}, \quad (5.12)$$

we have $\lambda(\mathbf{J}^*) = \lambda(\mathbf{J}^{33}) \cup \lambda(\mathbf{H})$, where $\mathbf{H} = \mathbf{J}^{22} - \mathbf{J}^{23}\mathbf{J}^{33^{-1}}\mathbf{J}^{32}$. The entry in the $(n(i-1)+j)$ -th row, $(n(k-1)+l)$ -th column of \mathbf{H} , denoted by $H_{ij,kl}$, is given by

$$H_{ij,kl} = \begin{cases} a_{ij}\beta - (\delta + \frac{\beta}{2}), & \text{if } ij = kl, i = l \\ -(\delta + \frac{\beta}{2}), & \text{if } ij = kl, i \neq l \\ a_{ij}\beta, & \text{if } ij \neq kl, i = l \\ 0, & \text{if } ij \neq kl, i \neq l \end{cases}. \quad (5.13)$$

We conclude that $\lambda(\mathbf{J}) = \lambda(\mathbf{J}^{11}) \cup \lambda(\mathbf{J}^{33}) \cup \lambda(\mathbf{H})$. Note that \mathbf{J}^{11} and \mathbf{J}^{33} are diagonal matrices, and all the diagonal entries of them are negative, i.e., all the eigenvalues of \mathbf{J}^{11} and \mathbf{J}^{33} are negative. As a result, we have

$$\lambda_1(\mathbf{J}) < 0 \Leftrightarrow \lambda_1(\mathbf{H}) < 0. \quad (5.14)$$

The matrix \mathbf{H} can be written as

$$\mathbf{H} = \mathbf{P} + \mathbf{Q}, \quad (5.15)$$

where \mathbf{Q} is a diagonal matrix and can be given by $\mathbf{Q} = \text{diag}[-(\delta + \frac{\beta}{2})]$. The entry in the $(n(i-1) + j)$ -th row, $(n(k-1) + l)$ -th column of \mathbf{P} , denoted by $P_{ij,kl}$, is given by

$$P_{ij,kl} = \begin{cases} a_{ij}\beta, & \text{if } i = l \\ 0, & \text{otherwise} \end{cases}. \quad (5.16)$$

We have that all the eigenvalues of \mathbf{Q} are $-(\delta + \frac{\beta}{2})$ and

$$\lambda(\mathbf{H}) = \lambda(\mathbf{P}) - (\delta + \frac{\beta}{2}). \quad (5.17)$$

Note that \mathbf{P} is a $n^2 \times n^2$ sparse matrix and similar with a block matrix consisting of $\beta\mathcal{A}$ and zero matrices as given by

$$\mathbf{P} = \mathbf{X}^{-1} \left[\begin{array}{c|c} \beta\mathcal{A} & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{0} \end{array} \right] \mathbf{X}. \quad (5.18)$$

This is achieved by performing matrix operations on \mathbf{P} . Thus, the eigenvalues of \mathbf{P} can be given by

$$\lambda(\mathbf{P}) = \beta \times \lambda(\mathcal{A}) \cup 0. \quad (5.19)$$

Combining (5.8), (5.14), (5.17) and (5.19), the epidemic will become extinct if $\beta\lambda_1(\mathcal{A}) - (\delta + \frac{\beta}{2}) < 0$. In other words, the epidemic threshold, denoted by τ^* , is given by

$$\tau^* = \frac{1}{\lambda_1(\mathcal{A}) - \frac{1}{2}}. \quad (5.20)$$

The epidemic dies out, if $\tau = \frac{\beta}{\delta} < \tau^*$.

5.4 Simulation and Numerical Results

In this section, numerical and simulation results are presented to validate the proposed group-based SIS model and the deduced epidemic threshold. In every run of the simulations, the groups and the network topology, i.e., the scale and structure, are first set up, according to the rules specified in Section 5.2, i.e., a_{ij} number of directed edges are added from a G_i -node to a_{ij} number of randomly selected G_j -nodes. The nodes do not connect themselves despite $a_{ii} \geq 0$. Then, the infection rate β and the curing rate δ are configured, based on the analytical epidemic threshold τ^* . The simulations are carried out on the NepidemiX [127], which is a Python library implementing simulations of epidemics. For initialization, randomly selected 10% of the nodes are infected. During a simulation run, the infected nodes can be cured at the rate of δ . Every infected node can infect its neighbors connected by edges at the rate of β . Every dot in the figures is the average of 100 independent runs under the same configurations, including the network topology, the percentage of initially infected nodes, β and δ .

We first validate our model on a 1000-node network where nodes connect each other and form a complete graph. The infection and curing rates are set to be $\beta = 0.0005$ and $\delta = 0.1$ per time slot. $\tau = \frac{\beta}{\delta} = 0.005$ is set to be larger than $\tau^* = 0.001$ to evaluate the proposed model. Ten percent of nodes are randomly chosen to be infected at $t = 0$. Fig. 5.3 plots the infection density, given by $\frac{\sum [I]_i}{N}$, from $t = 0$ to 30. The analytical results are numerically evaluated by employing (5.6) with the nodes evenly divided into 1, 2, 5, 10, 25 and 50 groups, respectively.

From the figure, we can see that the analytical results can asymptotically approach the simulation results with the increasing number of groups, e.g., from $n = 1$ to $n = 50$. The simulation results can outgrow the analytical results when the number of node groups is small. The analytical results under a single group of nodes,

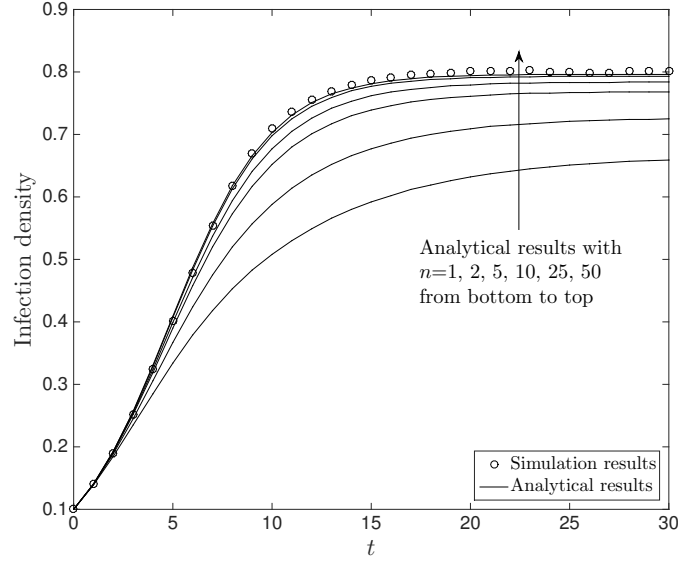


Figure 5.3 : The growth of infection density where a complete graph with 1000 nodes is considered. $\beta = 0.0005$ and $\delta = 0.1$. The analytical results are obtained based on (5.6) by evenly dividing the nodes into 1, 2, 5, 10, 25 and 50 groups.

i.e., $n = 1$, can substantially underestimate the infection density. The analytical result is only 83.8% ($\frac{0.67}{0.8}$) of the simulation result, when $t = 30$. In contrast, the analytical results under 50 groups of nodes, i.e., when $n = 50$, match the simulation results indistinguishably. This is because the proposed model is designed to decouple the state transitions of edges connecting nodes from different groups, and estimate the number of edges connecting infected nodes and subsequently the population of infected nodes. The mean-field approximation is applied to model the interplay between the averaged ratios of infectious edges connecting different pairs of node groups. With an increasing number of node groups, the averaged ratio of infectious edges connecting a specific pair of node groups can become increasingly representative with a reducing deviation. In other words, the averaged ratio becomes increasingly precise for a reducing set of edges. In the special case where every node forms a group, (i.e., the number of groups is $n = 1000$), the ratio is exactly the

probability at which an edge is infectious. The proposed model is able to capture the state transition of an edge under the averaged effect of all other individual edges, and can be fairly accurate given the large number of edges. Note that the number of groups, e.g., $n = 50$, is far less than the network size, i.e., 1000. This finding allows us to model the epidemic propagation with a small number of differential equations. The figure also shows that the analytical results can be accurate at the initial stage (or low infection densities) even with few node groups. This is because the state transitions of different edges are loosely coupled if only very few nodes are infected.

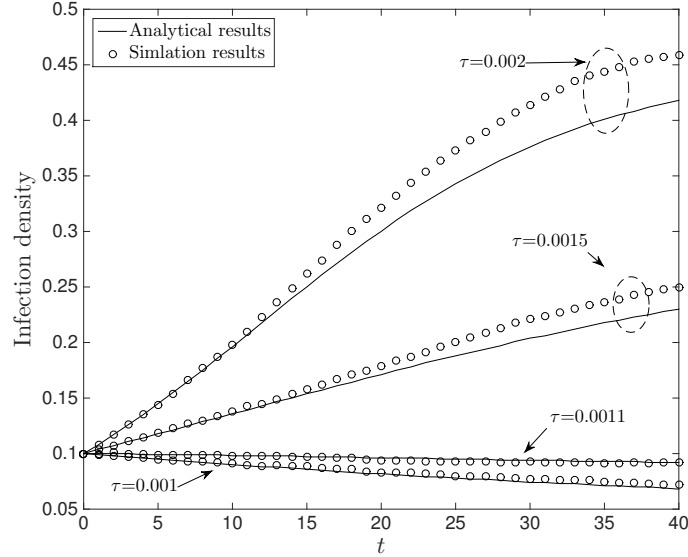


Figure 5.4 : The infection density with the growth of time where a complete graph with 1000 nodes is considered. The analytical results are obtained based on (5.6) by evenly dividing the nodes into 5 groups where $\tau = \frac{\beta}{\delta} = 0.002, 0.0015, 0.0011$ and 0.001, respectively.

We note that the proposed model has low complexity, i.e., $\mathcal{O}(n^2)$ where n is the number of groups. The complexity is significantly smaller than the model complexity in the last chapter, i.e., $\mathcal{O}(2^N)$ where N is the number of nodes. The model complexity is also small than the complexity of existing works, e.g. $\mathcal{O}(N)$ of the

model given in [121] because $n \ll N$, e.g., $n = 5$ and $N = 1000$ in our experiment.

We proceed to evaluate the model accuracy with different infection densities. This is done by adjusting the infection rate. A 1000-node network is considered where nodes connect each other and form a complete graph. The epidemic propagation with four effective spreading rates, i.e., $\tau = \frac{\beta}{\delta} = 0.002, 0.0015, 0.0011$ and 0.001 , are simulated and analyzed. The values of τ are larger than, and close to, the analytical threshold. The infection rates are obtained by adjusting β while setting δ to 0.1. The analysis is based on (6) by evenly dividing the nodes into 5 groups to explore the applicability of the model to a small number of groups.

From Fig. 5.4, we can see that the simulation results still overtake the analytical results when $\tau = 0.0015$ and 0.002 . For example, the simulation result is 0.459 in the case of $t = 40$ and $\tau = 0.002$, while the analytical result is only 0.418. However, the gap between simulation results and analytical results decreases with dropping τ , i.e., from $\tau = 0.002$ to 0.0015 . When τ further declines, i.e., $\tau = 0.0011$ and 0.001 , the analytical results are able to match the simulation results from the beginning to the end. According to (5.20), the epidemic threshold is given by $\tau^* = 0.001$. This figure reveals that the proposed mean-field model is asymptotically accurate with a decreasing τ and is able to precisely describe the epidemic propagation when the effective spreading rate is around the epidemic threshold.

We evaluate the epidemic threshold τ^* given by (5.20) in Fig. 5.5. Three networks with 500 nodes are considered, where nodes are divided into three groups (i.e., $\mathcal{N} = [100, 200, 200]$). The impact of the network topology on the threshold is evaluated by varying the number of edges in the network. Without loss of generality, their collapsed adjacency matrix \mathcal{A} is set to

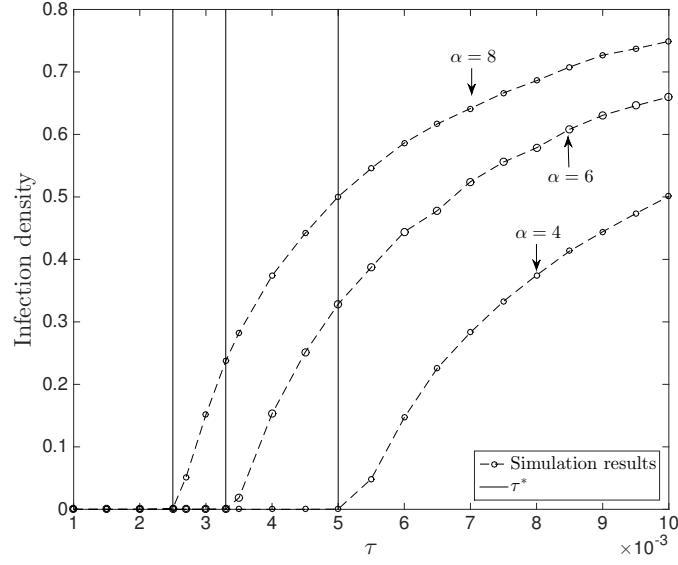


Figure 5.5 : The validation of epidemic threshold given by (5.20), where the y -axis is the infection density at $t = 1000$. Three networks with 500 nodes are considered. δ is set to be 0.1. $\alpha = 4, 6, 8$ is used to adjust the number of edges.

$$\mathcal{A} = \alpha \begin{bmatrix} 10 & 20 & 20 \\ 10 & 20 & 20 \\ 10 & 20 & 20 \end{bmatrix}, \quad (21)$$

where $\alpha = 4, 6$ or 8 . The curing rate δ is set to be 0.1. Ten percent of nodes are randomly selected to be infected at the initial state. The infection density at $t = 1000$ is used as the stable infection density, as indicated by the y -axis. Evaluated with (5.20), the epidemic threshold is $\tau^* = 0.005, 0.0033$ and 0.0025 when $\alpha = 4, 6$ and 8 , respectively. We can see from the figure that τ^* (the solid vertical lines) can precisely specify the epidemic thresholds. When $\tau < \tau^*$, the epidemic can be suppressed eventually. When $\tau > \tau^*$, the infection density grows with τ and also exhibits convexity. We can see that the network topology has a strong impact on the epidemic threshold and the infection density. Specifically, the threshold decreases

with the growth of α . For example, τ halves from 0.005 to 0.0025, when α doubles from 4 to 8 (the number of edges doubles, as well). The infection density increases with the growth of α , especially around the threshold. For example, in the case of $\alpha = 8$ and $\tau = 0.005$, the infection density is 0.5 as compared to the infection density of $\alpha = 4$.

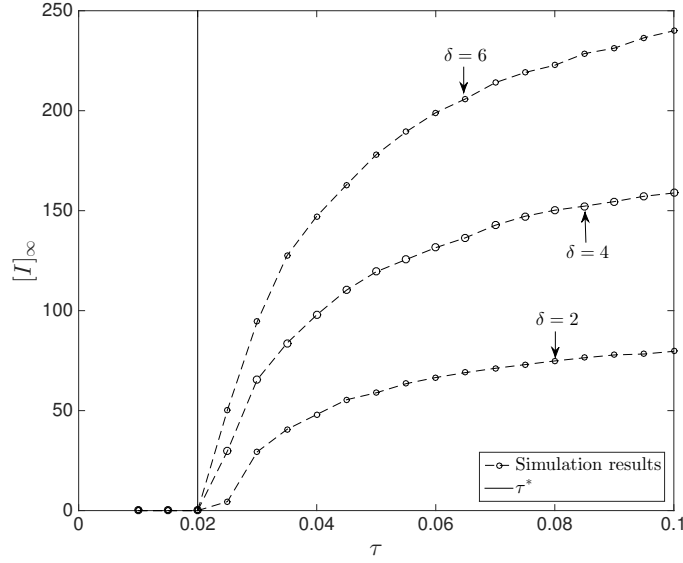


Figure 5.6 : The validation of epidemic threshold, where the y -axis is the infected population at $t = 1000$. Different scales of networks ($N=100, 200$ and 300 , respectively) are considered. δ is set to be 0.1 .

We note that the epidemic threshold, given by (5.20), is determined by the network structure \mathcal{A} , rather than the number of nodes given by \mathcal{N} . To illustrate this, we compare the number of infected populations in different scales of networks, illustrated by Fig. 5.6. To be specific, $\mathcal{N} = \delta \times [10, 20, 20]$, where $\delta = 2, 4$ and 6 , respectively. Their collapsed adjacency matrices are obtained with (21) by letting $\alpha = 1$. As a result, $\tau^* = 0.02$. Fig. 5.6 firstly confirms the accuracy of τ^* illustrated by the solid vertical line. The observation that the three networks with the same structure but different scales share the same epidemic threshold validates that the

epidemic threshold depends on the network structure rather than the scale. This finding can significantly reduce the complexity to deduce the epidemic threshold, i.e., from decomposing an N -dimensional matrix by using $\frac{1}{\lambda_1(\mathbf{A})}$, given by [129], to decomposing an n dimensional matrix ($n \ll N$). For example, $N = 1000$ and $n = 5$ in Fig. 5.4, and $N = 500$ and $n = 3$ in Fig. 5.5. It is interesting to notice that the infection densities are the same, e.g., $\frac{79.6}{100} \approx \frac{158.8}{200} \approx \frac{240}{300}$ in the case of $\tau = 0.1$, although the infected population varies in different scales of networks. This can be the reason of that the epidemic threshold depends on the network structure rather than the network scale.

5.5 Summary

In this chapter, we designed a continuous-time SIS model in large-scale networks. By categorizing nodes into groups, the model complexity was significantly reduced. The proposed epidemic model was validated to be asymptotically accurate with the decrease of the effective spreading rate and/or the increase of node groups. The epidemic threshold can be deduced with the largest eigenvalue of the collapsed adjacency matrix whose dimension is much smaller than the network scale. Simulation results corroborated the effectiveness of the model, as well as the analytical accuracy of the threshold in large-scale networks.

Chapter 6

Remote Control Modeling and Detection

The detection of APT is critical to control and defense against APT attacks. However, the detection of APT is a very challenging problem due to the advanced attack skills of APT [18]. For example, APT often attack targets with professional zero-day vulnerabilities [137], which is able to disable popular signature-based detection. The anomaly detection technology has been used in intrusion detection for a long time, where the data vectorization is a key step and has a strong impact on detection results. This chapter analyzes and models the remote control of APT, which is a distinctive stage. An anomaly detection method used to identify remote control traffic of APT is proposed and then validated.

6.1 Introduction

As a kind of novel and complicated attack, APT has not been well understood and effectively prevented. At the beginning, efforts have been put forward to analyze existing APT cases. After that, various multistage models were extended to describe APT cases, correlate alerts, rebuild attack scenes and predict attacks. Anomaly detection technology was also applied to detect APT attacks.

In this chapter, we focus on the essential stage of APT, i.e., the remote control stage. We find a new feature of remote control communications that the access of remote control domains tends to be independent, while legal web domains are accessed correlated. The feature is able to efficiently distinguish remote control domains and legal domains. To utilize this feature, we introduce a new concept,

i.e., the concurrent domains in DNS records, denoted by CODD, to measure the correlations among domains. Based on this feature, we propose a three-dimensional vector to represent the relationship between an internal host and an external domain, and apply classification algorithm to detect remote control. The method is validated on the public dataset provided by Los Alamos National Laboratory.

The rest of this chapter is organized as follows: In Section 6.2, we summarize key features of remote control in APT and propose a novel remote control detection method. Section 6.3 shows experimental and numerical results, followed by discussion and conclusion in Section 6.4.

Notations used in the chapter are as listed in Table 6.1.

Table 6.1 : Notations Used in Chapter 6

Notation	Description
r	A DNS vector
t	The DNS request time
h	The host which triggers the request
d	The queried domain
K	The number internal hosts in DNS records
H_k	The k -th host
D_l	The l -th connected domain
$\mathbf{V}_{k,l}$	The vectorized the connection between H_k and D_l
M	The number of connections between H_k and D_l
$R_{k,l,m}$	The set of CODDs with H_k and D_l with timestamp T_m
AN	The average number of CODDs
HC	The highest confidence level of CODDs

6.2 Remote Control Detection Method

In this section, we carry out analyses of remote control in APT and summarize three key features. Based on these features, we propose an effective remote control detection method.

6.2.1 Features of Remote Control in APT

Since APT is designed to evade detection, its remote control communications act like normal network traffic as possible as they can. In order to distinguish them, we analyze remote control communications in APT and identify its three key features.

1) Remote control is HTTP-based

In comparison with remote controls employing IRC or P2P, the HTTP-based remote control has remarkable advantages in penetrability and disguise and meets the requirements of APT. P2P and IRC traffic has distinct network features such as ports, package content. So, it can be easily detected and blocked. By contrast, the HTTP-based control mode is stealthier than others. Firstly, web traffic is usually labeled as legal in most enterprises. HTTP-based remote control traffic, therefore, is able to pass through security services, such as firewalls and intrusion detection systems. Secondly, remote control traffic can easily hide in the web traffic even transfers large amounts of data due to the fact that web traffic occupies a large proportion of the whole network traffic. In practice, more than 90% APT attacks employ HTTP-based remote control technology [138]. On account of these, we look for pieces of evidence of remote control in the HTTP related records.

2) Victims connect control center lowly and slowly

The main reason of this feature is the inconspicuous property of APT. To avoid detection, APT only attacks selected hosts according to the attack plan, which results in a limited scale of infected hosts compared with botnets. Consequently, the

remote control domains would not be dramatically connected by massive internal hosts. Another fact resulting from the detection evasion is slow attack processes. This is because high frequent connections left obvious evidence in the target system and increase the risk of being exposed. The slow attack process leads to that the connections between victims and remote control domains maintain low and slow [139].

However, connections between internal compromised hosts and control centers are necessary to control attack processes. Thus, the connections cannot be extremely rare.

Above all, the connections between victims and control centers exist and can be traced. But the connections are low and slow and therefore cannot be easily detected by applying large-scale group features.

3) Remote control domains are accessed independently

Since APT largely employs HTTP-based remote control, we tend to identify the difference between remote control communications and benign HTTP requests. We use concurrent domains to denote the domains that are accessed jointly, and believe that remote control domains have fewer concurrent domains than normal domains because of the independent access feature of remote control.

When users surf the Internet, a series of domains, instead of a single domain, are accessed within a time window. There are two kinds of activities resulting in this phenomenon, i.e., the loading of inline components and the continuous access. Due to the fact that a Web page consists of the main body and inline components, e.g., ads and the multimedia from other domains, the domains providing inline components are automatically accessed when the Web page is loaded. On other hand, users access a number of websites during the Web surfing. For example, a user uses a search engine to search keywords and then opens new pages to check details. On the contrary, remote control domains, which are only used to upload collected

data and download subsequent attack commands, are accessed independently and do not have inline components. As a result, remote control domain does not have concurrent domains.

DNS, translating domain names to numerical IP addresses, reflects HTTP requests and provides enough information to detect remote control connections. We use CODD to denote the COncurrent Domain in DNS records and define it by

Definition 1. *CODDs are the domains that are quired by the same host and appear within a given time window in the domain name server records.*

Although concurrent domains contribute to the CODDs, CODDs are not limited to the concurrent domains. For the concurrent domains analyzed before, there exist hyperlinks connecting them, i.e., they are connected rather than simply appear together. DNS records do not collect web page content and, therefore, cannot reveal the connectivity between concurrent domains. However, DNS records are still able to show the access sequence of concurrent domains. On the other hand, separate domains can become CODDs just because they are accessed together accidentally. For example, a user opens several favorite pages together, which makes these separate domains become CODDs. This also works for remote control domains because remote control domains can be connected while the benign domains are queried. This concurrence increases the number of CODDs of remote control domains and has a negative impact on the discrimination between remote control domains and benign domains. The number of CODDs, came from separate domains, can be treated as random error and reduced by re-observation.

Benign domains have a number of CODDs, while remote control domains have significantly fewer CODDs. Fig. 6.1 illustrates the number of CODDs of some popular benign domains, where these domains are independent accessed once and their CODDs are counted; the time window is set as 5s. It confirms that these

popular benign domains have an amount of CODDs. We also see that the more complicated (e.g., www.cnn.com) the webpage is the more CODDs it has.

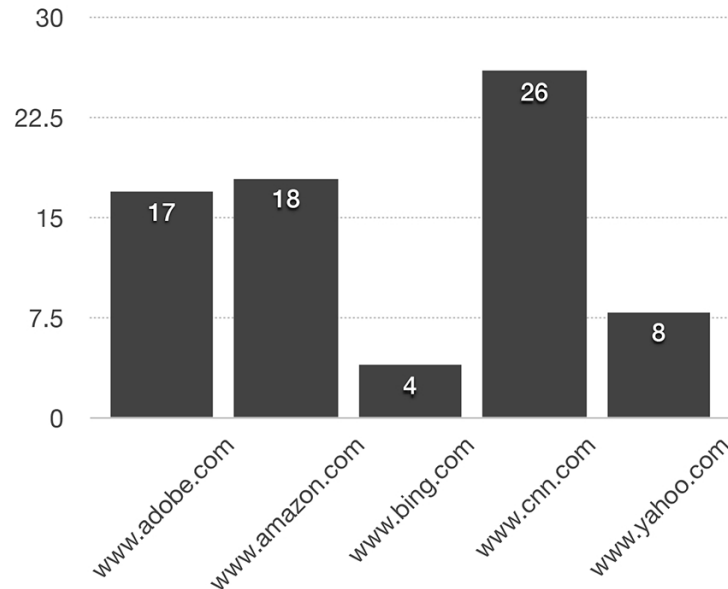


Figure 6.1 : The number of CODDs of some popular domains

6.2.2 Remote Control Detection Details

Based on the independent access feature of remote control, we propose to detect remote control in APT with the DNS records by the following steps.

Converting DNS records

Raw DNS records are converted into triples to simplify processing.

DNS records generated by internal DNS servers are excluded, since they do not provide more information for the detection. Then we extract the type-A DNS records, which translate given URLs into numerical addresses, because remote control communication in APT appears like normal traffic and mainly uses the type-A request. Finally, we use a triple, denoted by r , to present a DNS record which is

given by

$$r = \langle t, h, d \rangle, \quad (6.1)$$

where t denotes the request time; h denotes the host triggering the request and d denotes the queried domain.

Establishing sets of DNS record

Our remote control detection starts with a given host, which can be selected from internal important hosts. A set of DNS records for every host-domain pair is set up.

Suppose there are K internal hosts in the DNS records and the host H_k ($k = 1, 2 \dots K$) has connected L domains. The number of connections between internal host H_k and domain D_l ($l = 1, 2 \dots L$) is denoted by M . With a given time window $2 \times \delta$, let $R_{k,l,m}$ denote the set of CODDs associated with the m -th connection between host H_k and domain D_l with timestamp T_m . In other words, $R_{k,l,m}$ is the set of DNS records where the element r satisfies one of the two limitations in (6.2).

$$r(h) = H_k, \quad r(d) \in D, \quad T_m - \delta \leq r(t) < T_m; \quad (6.2)$$

$$r(h) = H_k, \quad r(d) \in D, \quad T_m < r(t) \leq T_m + \delta,$$

where $r(\cdot)$ is the element of r . We exclude the record whose $r(t)$ is equal to T_m to filter out the targeted DNS record itself.

Then we set up a set of $R_{k,l,m}$, $1 \leq m \leq M$, which is denoted by $C_{k,l}$ and used to present the CODDs of the connection between the internal host H_k and the queried domain D_l during the observation. The $C_{k,l}$ is given by

$$C_{k,l} = \{R_{k,l,i}\}_{i=1,2,\dots,M}. \quad (6.3)$$

Extracting features

After obtaining $C_{k,l}$, we use a 1×3 vector $\mathbf{V}_{k,l} = [M, AN, HC]$ to represent the connection between host H_k and domain D_l , where M is the connection times

between host H_k and domain D_l ; AN is short for the average number of CODDs; HC is short for the highest confidence level.

AN measures the quantity of CODDs and can be given by

$$AN = \frac{\sum_{i=1}^M |R_{k,l,i}|}{M}, \quad (6.4)$$

where $|\cdot|$ denotes cardinality. As analyzed before, CODDs exist for different reasons. We use the confidence level to measure the relationship between a given domain and its concurrent domains. The confidence level is numerically equal to the number of concurrent times. For a given domain D_w and a set $C_{k,l}$, the confidence level of D_w , denoted by $CI(D_w)$, is given by

$$CI(D_w) = |S_{k,l}|, \quad (6.5)$$

where $S_{k,l}$ is the subset of $C_{k,l}$ which contains the DNS record r whose $r(d)$ is the given domain D_w . Then HC is obtained by choosing the maximum $CI(D_w)$ and can be given by

$$HC = \max_{D_w \in D} (CI(D_w)). \quad (6.6)$$

HC is used to measure the correlation between the target domain and its CODDs. The CODDs generated by the inline components have high confidence levels.

Figure 6.2 illustrates the generation of the feature vector $\mathbf{V}_{k,l}$. As shown in the figure, host H_k connects domain D_l four times, so $M = 4$. $AN = 2.5$ since domain D_l has ten CODDs in total, i.e., $AN = \frac{10}{4} = 2.5$. Among the CODDs of D_l , domain 1 appears 4 times, which is the maximum appear times. As a result, $HC = 4$.

Finding Remote Control domains

After modeling the connections between the internal compromised hosts and remote control domains, our goal is identifying remote control domains based on classification. In the experiment, we use RIPPER [140] to perform classification

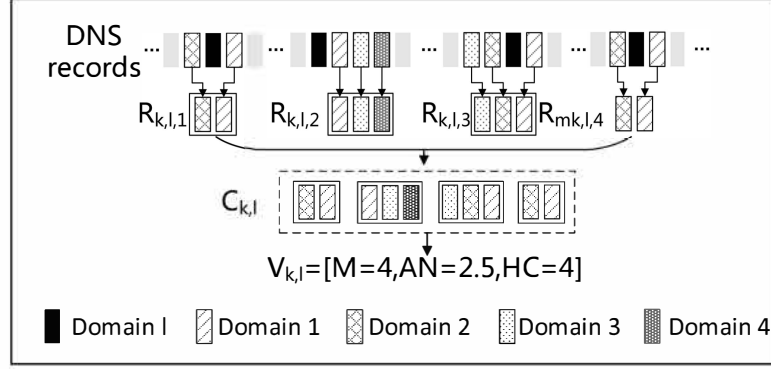


Figure 6.2 : Generative process of the feature vector

and detect remote control domains. Other classification algorithms can also be used in the classification-based detection. Among the variables in $\mathbf{V}_{k,l}$, M is used to measure active level and narrow search area. AN and HC reflect the independent access feature of the remote control connections and identify whether domain D_l is a remote control domain. As analyzed before, the domains with smaller AN and HC tend to be remote control domains.

6.3 Performance Evaluation

6.3.1 Data Set

To evaluate the performance of our detection method, we applied our remote control detection approach to the dataset of APT infection discovery challenge sponsored by Los Alamos National Laboratory (LANL) [141]. Specifically designed for the APT detection challenge, the dataset collected 1.25 TB DNS data in 49 days.

According to LANL, simulated APT was divided into three main stages: initial compromise, downloading additional malware, and establishing remote control communications. They simulated several APT attacks that contain whole or parts of the attack and provided two months of anonymized DNS records collected from a large site. The data of the first month was used as training data. The data of

the second month was mixed with simulated APT traffic and used as test data. In this data set, real attacks might exist, because the data came from a real site rather than network traffic generators. A hint file was provided for each day in the second month, which contains hint hosts and websites used in each phase of APT.

When our experiments were implemented, just part of the data of the second month was available on their FTP server. As far as we know, this data set was the only public and widely used dataset created for APT detection. The data set also provided enough data to test our mechanism. Therefore, we carried out experiments on this data set.

6.3.2 Experiment Details

We first simplified DNS records as described earlier. Meanwhile, we applied a whitelist to the DNS records to reduce them, which was set up based on the top 100 second-level domains (SLDs). After done that, the reduced data was about one percent of the original uncompressed data. In the dataset, we only used the data containing remote control communications and excluded the data simulating the initial compromised only. Finally, we used only the DNS records launched by the hosts given in the hint files and got 27,383 records of ten days in the second month.

We set the time window, δ , to be 5 seconds, which is a proper value to cover possible situations. We parsed the original dataset and extracted instances of $\mathbf{V}_{k,l}$ in a Python environment. In the end, we got 6486 instances of $\mathbf{V}_{k,l}$. There were 15 records labeled as remote control communications and 6471 records labeled as normal, according to the given hints. Among classification algorithms, we chose RIPPER for classification.

We carried out the 10-fold cross-validation ten times using Weka's [142] classify function with ten different random seeds. During this process, RIPPER would generate a set of classification rules. Since the real attacks might exist in the background

traffic, the domains not listed in the hint files might be remote control domains as well. We also analyzed the generated classification rules and the classification results.

6.3.3 Experiment results

Table 6.2 : Remote Control Detection Rules

Rule 1	$M \geq 3, AN \leq 0.4$
Rule 2	$M \geq 3, AN \leq 1, HC \leq 1$

In the experiment, we obtain two rules with RIPPER to filter remote control domains out, shown in Table 6.2. As we can see, the generated rules corroborate our analyses in Section 6.2.1 that remote control domains tend to have small AN and HC . The value of M indicates that the compromised hosts are always under the control of adversaries. There is a special phenomenon in Rule 1 that HC is not needed when $AN < 0.4$. This is because the domains, which only have rare CODDs and are considered as remote control domains, do not have a CODD with high HC . On the other hand, normal domains must have a certain amount of CODDs. As for the Rule 2, though the value of AN is larger than that of Rule 1, it remains a low level, which reveals that the remote control domains are accessed independently most of the time but connected along with benign requests accidentally. The value of HC contributes to the detection in Rule 2, which confirms that remote control domains barely have concurrent domains.

Table 6.3 illustrates the detection results of the generated rules, where each number refers to the quantity of host-domain pairs. Based on the rules, our detection method could find out all the documented 15 remote control domains without missing but classify 3 documented benign domains as remote control domains. However,

Table 6.3 : Confusion Matrix of Classification

	Classified as Remote control	Classified as Normal
Documented Remote control	15	0
Documented Normal	3	6468

due to the fact that real attacks might exist in the data set, we carry out further analyses on the misclassified domains.

The numerical \mathbf{V} of the three misclassified classified records can be given by $\mathbf{V}_1 = [172, 0.2558, 6]$, $\mathbf{V}_2 = [4, 0, 0]$ and $\mathbf{V}_3 = [9, 0.56, 1]$, respectively. We examined these domains based on the periodic connection feature used in [100], which assumes remote control domains are connected periodically. We found that the domain of \mathbf{V}_1 is connected every 3 minutes; the domain of \mathbf{V}_3 is connected almost every 1 hour; while the domain of \mathbf{V}_2 is connected irregularly. It indicates that the domains of \mathbf{V}_1 and \mathbf{V}_3 can be undocumented remote control domains. \mathbf{V}_2 is so outstanding with $AN = 0, HC = 0$ that we regard it as a non-periodic undocumented remote control launched by skillful attackers.

Besides of [100], the periodic connection feature was also applied in [143] with a different evaluation method. In their APT detection mechanisms, the remote control detection lays a foundation of precursor and successor detections. Our remote control detection results were compared with theirs in the help of hints. In our ten times 10-fold cross-validation, we are able to find out 89.3% remote control connections in average. According to the result in [100, Table. I], the number of undocumented but labeled as remote control domains is larger than the number of detected real remote control domains, which is an unusual phenomenon even though there might exist undocumented attacks in the dataset. Based on the result in [143],

their detection method could find out 79% remote control connections in the case of applying the feature of periodic connection only. Although the feature of periodic connection can be used to detect remote control domains, it can be easily evaded in practice. For example, the adversaries can configure non-periodic connections, which do not rely on host states, i.e., does not leave additional pieces of evidence and trigger other detection systems.

We performed further statistical analyses on the dataset to explain the generated rules. During the statistics, we removed the domains connected by the same host less than three times, i.e., let $M \geq 3$, which is the same configuration as [143]. Because the communications between the compromised hosts and control centers cannot be extremely rare as analyzed and validated before, while legal HTTP requests can. Finally, we got 1364 instances of $\mathbf{V}_{k,l}$.

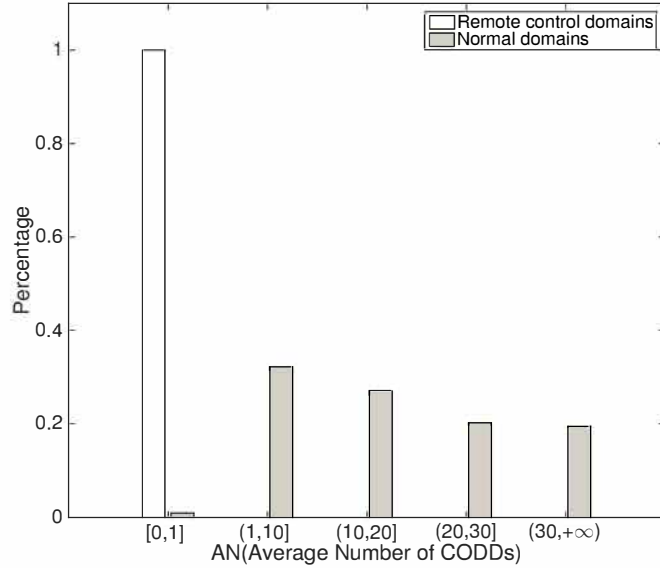


Figure 6.3 : Distribution of Average Number

Figure 6.3 compares the difference between the distribution of AN of remote control domains and normal domains. A noticeable distinction is that all of re-

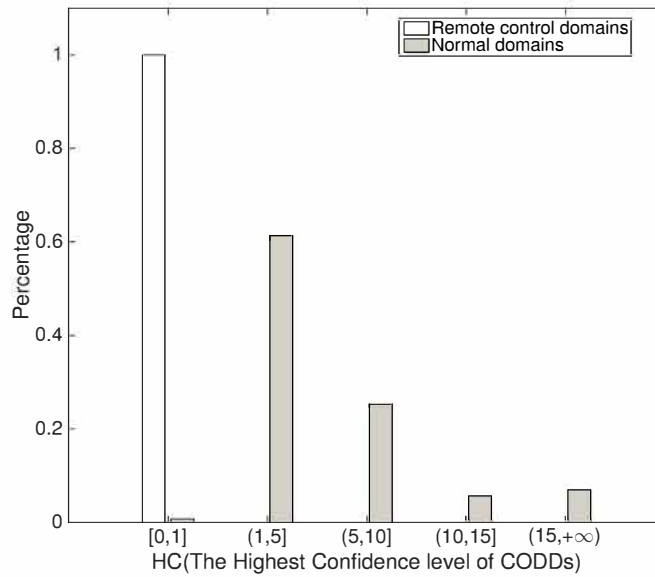


Figure 6.4 : Distribution of the Highest Confidence level

remote control domains' AN are no larger than 1 while AN of normal domains are distributed throughout a wide range. This suggests that normal domains have a number of domains which are concurrent with them, but remote control domains are accessed independently and barely have CODDs.

Figure 6.4 compares the distribution of HC of remote control domains and normal domains. Similarly, all of remote control domains' HC are no larger than 1, and rare normal domains' HC is less than 1. Near 61% normal domains have CODDs whose the highest confidence level ranges from 1 to 5. Combining the statistics result that almost the same (59%) normal domains are connected from 3 to 5 times, it corroborates that normal domains have fixed concurrent domains. Both of Figure 6.3 and Figure 6.4 confirm our analyses that remote control domains are accessed independently, and their CODDs are rare and mainly caused by accidentally together access. On the contrary, normal domains have a number of concurrent domains as well as CODDs.

6.4 Discussions and Summary

Although the proposed detection method is able to find out remote control domains in the LANL dataset, it can be evaded by the following strategies: a) The compromised hosts issue a series of HTTP requests while connecting remote control domains; b) The compromised hosts connect remote control domains while users are surfing the Web. However, these strategies make the remote control connection anomaly in other aspects. For the first strategy, the compromised hosts may change the behaviors of web usage, e.g., connecting the users' unfamiliar domains, which are anomaly actions and will trigger other anomaly detection services. In the case of the second strategy, the adversaries have to monitor the users' internet usage, which increases the cost of invasion. On the other hand, network monitoring is a sensitive function and highly regulated by security services. Therefore, this strategy also makes the adversaries exposed.

The proposed detection method does not rely on the widely used features, such as traffic content and the correlation between domain names. It provides a new insight of remote control. Applying the independent access feature, our novel detection method is effective in the context of APT attacks that skillful adversaries utilize information hiding, targeted attack, and random calling-back to evade existing detections. The proposed remote control detection method demonstrates the usage of the independent access feature. In practice, this feature can be applied with other modeling methods or combined with other detection technologies, such as traffic-based detection and the group similarity, to improve the performance.

The proposed remote control detection method is able to handle large-scale traffic. First of all, the proposed detection method just uses the DNS traffic which takes a small ratio of the total network traffic. Secondly, the proposed detection method analyzes the connections between internal hosts and external domains rather than

every individual DNS requests, which can significantly reduce the amount of data to be analyzed. Then, the proposed detection method uses a tuple to represent a connection between an internal host and an external domain rather than the complex raw DNS data. Lastly, filters, such as the white list used in this chapter, can effectively reduce data as well.

Cyber security is an endless competition between adversaries and defenders, especially in APT. The adversaries can continuously update their attack skills and adjust attack strategies. Thus, there is no permanent solution for the defenders to ensure information security. The proposed independent access feature helps the defenders to achieve a leading superiority in the competition with attackers.

In this chapter, we analyzed and concluded the features of remote control in APT, which were quite different from the remote control in botnets. We proposed a novel feature of remote control communication in APT named independent access. Based on the feature, a DNS records analysis and remote control detection method was implemented, which was validated on public datasets.

Chapter 7

Future Works

With a series of proposed models, this thesis analyzes key stages of APT cyber-attacks and deduces meaningful results. In this chapter, possible future research directions are discussed based on the works in this thesis.

7.1 Future Works on Propagation Models

7.1.1 New Propagation Processes

New propagation processes can be proposed to study the propagation of specific cyber-attacks. The homogeneous propagation model is studied in this thesis, where all the nodes share the same infection and curing rate, can be extended to heterogeneous propagations where every node has a unique curing rate and infection rate. This is based on the fact that nodes can be at different risk levels according to their heterogeneous security settings, functions, and operators. Note that the complexity of heterogeneous models would dramatically higher than that of homogeneous models. Thus, efficient analysis methods, such as mean-field and linearization, can be developed to tackle the complexity issue. Another possible research direction is the modeling of propagation processes with new statuses. Traditional epidemic models, including the SIS model analyzed in this thesis, can hardly capture the versatile APT attacks. New attack behaviors, such as being scanned and information collection, can be captured by models with new statuses. The status transition can be described by a directed graph rather than simply susceptible-infected-susceptible looping transition in SIS models. As a result, the models can be analyzed from new

viewpoints beyond the lifetime and extinction threshold as adopted in this thesis.

7.1.2 Propagation in Adaptive Networks

New propagation models can be proposed to study the propagation of cyber-attacks in adaptive/dynamic networks. This thesis studies the virus propagation in static networks where both the nodes and edges remain unchanged. Practical networks, however, can be adaptive where nodes can join and leave and edges can be broken and rewired according to given rules. For example, programmable infrastructure technology can dynamically interpret a single physical network into a variety of logical networks. The adaptive networks allow administrators to control propagation by isolating infected nodes and, on the other hand, might accelerate the propagation because infected nodes have the potential to interact with more susceptible nodes. As a result, new control strategies can be proposed to suppress the propagation of viruses by designing adaptive rules. If adaptive rules are given, new propagation models can be developed to evaluate the propagation of viruses under the adaptive rules.

7.1.3 Propagation in Multi-layer Networks

New propagation models can be proposed to study the propagation of cyber-attacks across multi-layer networks. As analyzed in the first two sections, APT can be across multiple networks, e.g., social network, computer network and industrial control network, where a single node can have different connections and behave differently in the networks. This thesis only studies the propagation of attacks in a single layer network, which can be extended to the propagation in multi-layer networks. The key research can be the description of multi-layer networks, the propagation process in every individual network and the interaction among multi-layer networks.

7.2 Cyber-Attack Detection

7.2.1 Detection of New Attacks

Detection technology can be developed to identify new attacks. With the development of information and communication technology, cyber-attacks have been extended to various networks, e.g., social network, vehicle network and cyber-physical network, and a variety of devices, such as mobile phones and autonomous vehicles. The cyber-attacks can be traced in different logs and hardly be detected with a single data source and method. As a result, detection technology can be developed to address the challenge of advanced attacks. The key research point is the description of attacks. To be specific, dynamic behavior process and static code samples can be collected and then abstracted to attack models with graph models, e.g., Petri net, and vectorization.

7.2.2 Artificial Intelligence-based Detection

The latest artificial intelligence technologies have the potential to improve the detection performance, such as detection rate, false alarm rate, and efficiency. Traditional artificial intelligence technologies, e.g., classification algorithms and cluster algorithms, have been widely adopted in abnormal detection. The artificial intelligence technologies have been greatly developed and have solved many practical complex issues. For example, the long short-term memory and the convolutional neural network deep learning technology have been developed for natural language processing and image processing tasks and achieved significant outcomes, respectively. Such technologies give new viewpoints to analyze security data and can be adopted in attack detection. For example, the security data in an enterprise network can be treated as graph signals and then be processed with the image processing technologies, which is a novel detection method and has the potential to provide meaningful global results.

7.3 Blockchain-based Security

7.3.1 Introduction of Blockchain

Being a distributed and tamper-resistant ledger database, Blockchain has become a hot topic in security research and has the potential to address the critical security issues, particularly on data integrity and reliability. Blockchain allows software applications to send and record transactions in a trustworthy and distributed (peer-to-peer) manner. Blockchain is rapidly gaining popularity and used extensively for applications including smart contracts, distributed storage, and digital assets. With the Blockchain technologies, the security requirement can be fulfilled. The following prominent features of Blockchain can contribute to data integrity and enhance security:

- *Decentralization:* Blockchains can record transactions between multiple parties without central coordination. This can provide flexible network configurations, and reduce the risks of single-point failures.
- *Integrity:* Blockchains are able to keep transactions permanently in a verifiable way. Specifically, the signatures of the senders in transactions can guarantee the integrity and non-repudiation of the transactions. The hash chain structure of Blockchains ensures that any recorded data cannot be updated, even partly. The consensus protocols of Blockchains can guarantee valid and consistent records. The protocols can also tolerate failures and attacks, e.g., attackers with less than $\frac{1}{2}$ hash power in Proof of Work (PoW), or less than $\frac{1}{3}$ of nodes in Practical Byzantine Fault Tolerance (PBFT) consensus protocol.
- *Anonymity:* Blockchains can use changeable public keys as users' identities to preserve anonymity and privacy. This is attractive to many applications, especially those which need to keep confidential identities and privacy.

7.3.2 Blockchain-based Security Service

Blockchain can be employed to provide robust and reliable services against the damage from cyber-attacks. This is based on the fact that Blockchain runs on a peer-to-peer network and can resist single-point failure benefited from its consensus protocol. However, Blockchain is dramatically different from centralized web services and has its unique features, such as the smart contract technology and token system, which are initially designed for cryptocurrency. It is necessary to develop the Blockchain technologies for the compatibility with the systems beyond cryptocurrency. On the other hand, current Blockchain technologies require massive storage and heavy computing and can hardly replace the cloud-based services. A potential research point is the consensus protocol for flexible and efficient Blockchains. Business-based consensus protocols can be a research point for next-generation Blockchains, where the consensus protocols drive practical businesses besides generating blocks.

Blockchain can be employed to realize trusted and privacy-preserving services against information leakage attacks in APT. This is based on the fact that Blockchain is a decentralized system and widely employs cryptography to secure data on it. However, Blockchain is transparent where all data can be public verified. A key research is to build a flexible and privacy-preserving data sharing system over the Blockchain. Recent advanced cryptographic algorithms, e.g., proxy re-encryption, multiparty computing, and homomorphic encryption, can be developed to tackle the challenge, where data can be encrypted with the algorithms before being uploaded to Blockchain.

Chapter 8

Contributions

In this chapter, we first summarize the research in this thesis, followed by contributions of chapters.

In this thesis, a series of models were proposed to model and analyze key stages of APT. The proposed models include game theoretic social attack-defense models, a Markov theoretic propagation model, a mean-field theoretic propagation model, and a command and control model. To be specific, the novelty and contributions of this thesis can be summarized as follows.

Game theoretic social attack-defense models were proposed to capture social attacks in the initial attack stage of APT. The infinitely repeated games evaluated the effect of risk alerts on the inhibition of attack messages in the presence of multiple types of subscribers and possible miss-detection and false alarm of forged messages. The proposed games captured the interactions between a message publisher and the network administrator in an online social network. In the absence and presence of misclassification on genuine messages, sufficient conditions under which the publisher is disincentivized from publishing forged messages were identified. Closed-form expressions were derived for the maximum number of forged messages of a malicious publisher. Confirmed by simulations, our analysis indicates that forged messages can be suppressed by improving the payoffs for genuine messages, increasing the cost of bots, and/or reducing the payoffs for forged messages. The increasing detection probability of forged messages or decreasing misclassification probability of genuine messages can also have a strong impact on the suppression of forged messages.

A discrete-time absorbing Markov process was designed to characterize SIS-type propagations in the lateral propagation stage of APT. Eigenvalue analysis and Jordan decompositions were carried out on the transition matrix of the Markov process to derive the bounds of the virus extinction rate. Based on the bounds, a computationally efficient approach was developed to evaluating virus lifetimes in large networks. Specifically, the required extinction rate of a large network was interpreted to that of a much smaller network to deduce the required minimum curing rate of the large network efficiently through the upper bound of its smaller counterpart. Simulation results corroborated the effectiveness of the interpretation, as well as analytical accuracy in large computer networks.

A continuous-time SIS model was proposed to analyze propagations in large-scale networks, corresponding to the lateral propagation stage of APT. By categorizing nodes into groups, the model complexity was significantly reduced. The proposed epidemic model was validated to be asymptotically accurate with the decrease of the effective spreading rate and/or the increase of node groups. The epidemic threshold can be deduced with the largest eigenvalue of the collapsed adjacency matrix whose dimension is much smaller than the network scale. Simulation results corroborated the effectiveness of the model, as well as the analytical accuracy of the threshold in large-scale networks.

The command and control in APT was analyzed in this thesis that keeps a low and slow communication pattern. The independent access feature was identified to describe the command and control in APT and distinguish it from that in botnets. Based on the feature, a tuple was proposed base on DNS records to detect APT-type command and control. The detection method was validated on a public dataset and then discussed.

Bibliography

- [1] 360. 2018 global advanced persistent threat report. <https://ti.360.net/uploads/2019/01/02/56e5630023fe905b2a8f511e24d9b84a.pdf>, 2019.
- [2] Internet world stats. <https://www.internetworldstats.com/stats.htm>, 2018.
- [3] State of the iot 2018: Number of iot devices now at 7b market accelerating. <https://iot-analytics.com/state-of-the-iot-update-q1-q2-2018-number-of-iot-devices-now-7b/>, 2018.
- [4] Dipankar Raychaudhuri and Mario Gerla. *Emerging wireless technologies and the future mobile internet*. Cambridge University Press, 2011.
- [5] Hannes Hartenstein and Kenneth Laberteaux. *VANET: vehicular applications and inter-networking technologies*, volume 1. John Wiley & Sons, 2009.
- [6] Ahmad-Reza Sadeghi, Christian Wachsmann, and Michael Waidner. Security and privacy challenges in industrial internet of things. In *2015 52nd ACM/EDAC/IEEE Design Automation Conference (DAC)*, pages 1–6. IEEE, 2015.
- [7] Jiong Jin, Jayavardhana Gubbi, Slaven Marusic, and Marimuthu Palaniswami. An information framework for creating a smart city through internet of things. *IEEE Internet of Things journal*, 1(2):112–121, 2014.

- [8] Constantinos Kolias, Georgios Kambourakis, Angelos Stavrou, and Jeffrey Voas. Ddos in the iot: Mirai and other botnets. *Computer*, 50(7):80–84, 2017.
- [9] Nikos Virvilis and Dimitris Gritzalis. The big four-what we did wrong in advanced persistent threat detection? In *Availability, Reliability and Security (ARES), 2013 Eighth International Conference on*, pages 248–254. IEEE, 2013.
- [10] Gaoqi Liang, Steven R Weller, Junhua Zhao, Fengji Luo, and Zhao Yang Dong. The 2015 ukraine blackout: Implications for false data injection attacks. *IEEE Transactions on Power Systems*, 32(4):3317–3318, 2017.
- [11] Cybercrime legislation worldwide. https://unctad.org/en/Pages/DTL/STI_and_ICTs/ICT4D-Legislation/eCom-Cybercrime-Laws.aspx, 2018.
- [12] National vulnerability database. <https://nvd.nist.gov/general>, 2018.
- [13] Trend Micro. Mapping the future. <https://documents.trendmicro.com/assets/rpt/rpt-mapping-the-future.pdf>, 2018.
- [14] Richard Bejtlich. Air force cyberspace report. <http://taosecurity.blogspot.com/2007/10/air-force-cyberspace-report.html>, 2007.
- [15] Google. New approach to china. <http://googleblog.blogspot.com/2010/01/new-approach-to-china.html>.
- [16] NIST. Advanced Persistent Threat (APT). <https://csrc.nist.gov/glossary/term/advanced-persistent-threat>, 2011.
- [17] FireEye. M-trends 2018. <https://www.fireeye.com/content/dam/fireeye-www/services/pdfs/ig-mtrends-2018.pdf>, 2018.

- [18] A. Alshamrani, S. Myneni, A. Chowdhary, and D. Huang. A survey on advanced persistent threats: Techniques, solutions, challenges, and research opportunities. *IEEE Communications Surveys Tutorials*, pages 1–1, 2019.
- [19] Nicolas Falliere, Liam O Murchu, and Eric Chien. W32. stuxnet dossier. *White paper, Symantec Corp., Security Response*, 5(6):29, 2011.
- [20] Boldizsár Bencsáth, Gábor Pék, Levente Buttyán, and Márk Félegyházi. The cousins of stuxnet: Duqu, flame, and gauss. *Future Internet*, 4(4):971–1003, 2012.
- [21] Symantec Security Response. Flamer: Highly sophisticated and discreet threat targets the middle east.
<https://www.symantec.com/connect/blogs/flamer-highly-sophisticated-and-discreet-threat-targets-middle-east>.
- [22] Arthur W Coviello. Open letter to rsa customers. *RSA [database online]*, 2011.
- [23] Katharina Krombholz, Heidelinde Hobel, Markus Huber, and Edgar Weippl. Advanced social engineering attacks. *Journal of Information Security and Applications*, 22:113 – 122, 2015. Special Issue on Security of Information and Networks.
- [24] Ari Juels and Ting Fang Yen. Sherlock holmes and the case of the advanced persistent threat. In *Proceedings of the 5th USENIX conference on Large-Scale Exploits and Emergent Threats*, pages 2–2, 2012.
- [25] Command Five Pty Ltd. Advanced persistent threats: A decade in review. 2011.
- [26] A. Paradise, A. Shabtai, R. Puzis, A. Elyashar, Y. Elovici, M. Roshandel, and C. Peylo. Creation and management of social network honeypots for

- detecting targeted cyber attacks. *IEEE Transactions on Computational Social Systems*, 4(3):65–79, Sep. 2017.
- [27] Aditya K Sood and Richard J Enbody. Targeted cyberattacks: a superset of advanced persistent threats. *IEEE security & privacy*, 11(1):54–61, 2013.
- [28] M. Ussath, D. Jaeger, F. Cheng, and C. Meinel. Advanced persistent threats: Behind the scenes. In *2016 Annual Conference on Information Science and Systems (CISS)*, pages 181–186, March 2016.
- [29] Lena Laribee, David S Barnes, Neil C Rowe, and Craig H Martell. Analysis and defensive tools for social-engineering attacks on computer systems. In *2006 IEEE Information Assurance Workshop*, pages 388–389, June 2006.
- [30] Ping Chen, Lieven Desmet, and Christophe Huygens. A study on advanced persistent threats. In Bart De Decker and André Zúquete, editors, *Communications and Multimedia Security*, pages 63–72, Berlin, Heidelberg, 2014. Springer Berlin Heidelberg.
- [31] Aleksandr Matrosov, Eugene Rodionov, David Harley, and Juraj Malcho. Stuxnet under the microscope. *ESET LLC (September 2010)*, 2010.
- [32] Kate Munro. Deconstructing flame: the limitations of traditional defences. *Computer Fraud & Security*, 2012(10):8 – 11, 2012.
- [33] Antoine Lemay, Joan Calvet, Francois Menet, and Jos M. Fernandez. Survey of publicly available reports on advanced persistent threat actors. *Computers & Security*, 72:26 – 59, 2018.
- [34] R. Heartfield, G. Loukas, and D. Gan. You are probably not the weakest link: Towards practical prediction of susceptibility to semantic social engineering attacks. *IEEE Access*, 4:6910–6928, 2016.

- [35] Piet Van Mieghem, Jasmina Omic, and Robert Kooij. Virus spread in networks. *IEEE/ACM Transactions on Networking*, 17(1):1–14, 2009.
- [36] Bruce Schneier. Attack trees. *Dr. Dobbs journal*, 24(12):21–29, 1999.
- [37] Paul Giura and Wei Wang. A context-based detection framework for advanced persistent threats. In *2012 International Conference on Cyber Security*, pages 69–74. IEEE, 2012.
- [38] Sebastian Roschke, Feng Cheng, and Christoph Meinel. Using vulnerability information and attack graphs for intrusion detection. In *2010 Sixth International Conference on Information Assurance and Security*, pages 68–73. IEEE, 2010.
- [39] Wentao Zhao, Pengfei Wang, and Fan Zhang. Extended petri net-based advanced persistent threat analysis model. In *Computer Engineering and Networking*, pages 1297–1305. Springer, 2014.
- [40] Marten van Dijk, Ari Juels, Alina Oprea, and RonaldL. Rivest. Flipit: The game of “stealthy takeover”. *Journal of Cryptology*, 26(4):655–713, 2013.
- [41] L. Xiao, D. Xu, N. B. Mandayam, and H. V. Poor. Attacker-centric view of a detection game against advanced persistent threats. *IEEE Transactions on Mobile Computing*, 17(11):2512–2523, Nov 2018.
- [42] M. Min, L. Xiao, C. Xie, M. Hajimirsadeghi, and N. B. Mandayam. Defense against advanced persistent threats in dynamic cloud storage: A colonel blotto game approach. *IEEE Internet of Things Journal*, 5(6):4250–4261, Dec 2018.
- [43] Pengdeng Li, Xiaofan Yang, Qingyu Xiong, Junhao Wen, and Yuan Yan Tang. Defending against the advanced persistent threat: An optimal control approach. *Security and Communication Networks*, 2018, 2018.

- [44] M. Egele, G. Stringhini, C. Kruegel, and G. Vigna. Towards detecting compromised accounts on social networks. *IEEE Trans. Depend. Sec. Comput.*, 14(4):447–460, July 2017.
- [45] X. Ruan, Z. Wu, H. Wang, and S. Jajodia. Profiling online social behaviors for compromised account detection. *IEEE Trans. Inf. Forensics Security*, 11(1):176–187, Jan 2016.
- [46] C. Chen, Y. Wang, J. Zhang, Y. Xiang, W. Zhou, and G. Min. Statistical features-based real-time detection of drifted twitter spam. *IEEE Trans. Inf. Forensics Security*, 12(4):914–925, April 2017.
- [47] C. Yang, R. Harkreader, and G. Gu. Empirical evaluation and new design for fighting evolving twitter spammers. *IEEE Trans. Inf. Forensics Security*, 8(8):1280–1293, Aug 2013.
- [48] S. Shehnepoor et al. Netspam: A network-based spam detection framework for reviews in online social media. *IEEE Trans. Inf. Forensics Security*, 12(7):1585–1595, July 2017.
- [49] F. Benevenuto, T. Rodrigues, A. Veloso, J. Almeida, M. Goncalves, and V. Almeida. Practical detection of spammers and content promoters in online video sharing systems. *IEEE Trans. Syst. Man, Cybern. B, Cybern.*, 42(3):688–701, June 2012.
- [50] Victoria L Rubin. Deception detection and rumor debunking for social media. *The SAGE Handbook of Social Media Research Methods*, page 342, 2017.
- [51] T. Zhu et al. Beating the artificial chaos: Fighting osn spam using its own templates. *IEEE/ACM Trans. New.*, 24(6):3856–3869, December 2016.

- [52] M. Nitti, R. Girau, and L. Atzori. Trustworthiness management in the social internet of things. *IEEE Trans. Knowl. Data Eng.*, 26(5):1253–1266, May 2014.
- [53] S. Sedhai and A. Sun. Semi-supervised spam detection in twitter stream. *IEEE Trans. Comput. Social Syst.*, 5(1):169–175, March 2018.
- [54] Xu Wang, Bo Song, Wei Ni, et al. Group-based susceptible-infectious-susceptible model in large-scale directed networks. *Security and Communication Networks*, 2019(1657164):9, 2019.
- [55] L. Liu and H. Jia. Trust evaluation via large-scale complex service-oriented online social networks. *IEEE Trans. Syst., Man, Cybern., Syst.*, 45(11):1402–1412, Nov 2015.
- [56] M. Sirivianos, K. Kim, and X. Yang. SocialFilter: Introducing social trust to collaborative spam mitigation. In *Proc. INFOCOM*, pages 2300–2308, April 2011.
- [57] J. D. Falk and M. S. Kucherawy. Battling spam: The evolution of mail feedback loops. *IEEE Internet Comput.*, 14(6):68–71, Nov 2010.
- [58] P. Heymann, G. Koutrika, and H. Garcia-Molina. Fighting spam on social web sites: A survey of approaches and future challenges. *IEEE Internet Comput.*, 11(6):36–45, Nov 2007.
- [59] X. Zha, W. Ni, K. Zheng, R. P. Liu, and X. X. Niu. Collaborative authentication in decentralized dense mobile networks with key predistribution. *IEEE Transactions on Information Forensics and Security*, 12:2261–2275, October 2017.
- [60] K. Zhang et al. PIF: A personalized fine-grained spam filtering scheme with

- privacy preservation in mobile social networks. *IEEE Trans. Comput. Social Syst.*, 2(3):41–52, Sept 2015.
- [61] How is facebook addressing false news through third-party fact-checkers?
https://www.facebook.com/help/1952307158131536?locale=en_US,
 2018.
- [62] Manoj Parameswaran, Huaxia Rui, and S Sayin. A game theoretic model and empirical analysis of spammer strategies. In *Collaboration, Electronic Messaging, AntiAbuse and Spam Conf. (CEAS '10)*, volume 7. Citeseer, 2010.
- [63] F. Zhou, R. J. Jiao, and B. Lei. Bilevel game-theoretic optimization for product adoption maximization incorporating social network effects. *IEEE Trans. Syst., Man, Cybern., Syst.*, 46(8):1047–1060, Aug 2016.
- [64] H. Abbass, G. Greenwood, and E. Petraki. The N -player trust game and its replicator dynamics. *IEEE Trans. Evol. Comput.*, 20(3):470–474, June 2016.
- [65] L. Yang, P. Li, Y. Zhang, X. Yang, Y. Xiang, and W. Zhou. Effective repair strategy against advanced persistent threat: A differential game approach. *IEEE Transactions on Information Forensics and Security*, 14(7):1713–1728, July 2019.
- [66] Roy M Anderson. *Infectious diseases of humans: dynamics and control*, volume 28. Wiley Online Library, 1992.
- [67] Jeffrey O Kephart and Steve R White. Directed-graph epidemiological models of computer viruses. In *Research in Security and Privacy, 1991. Proceedings., 1991 IEEE Computer Society Symposium on*, pages 343–359, 1991.

- [68] J T Jackson and S Creese. Virus propagation in heterogeneous Bluetooth networks with human behaviors. *Dependable and Secure Computing*, 2012.
- [69] Chao Gao and Jiming Liu. Modeling and restraining mobile virus propagation. *Mobile Computing, IEEE Transactions on*, 12(3):529–541, March 2013.
- [70] P. De, Yonghe Liu, and S.K. Das. An epidemic theoretic framework for vulnerability analysis of broadcast protocols in wireless sensor networks. *Mobile Computing, IEEE Transactions on*, 8(3):413–425, March 2009.
- [71] J O Kephart and S R White. Measuring and modeling computer virus prevalence. In *Research in Security and Privacy, 1993. Proceedings., 1993 IEEE Computer Society Symposium on*, pages 2–15. IEEE, 1993.
- [72] M. Vojnovic and A.J. Ganesh. On the race of worms, alerts, and patches. *Networking, IEEE/ACM Transactions on*, 16(5):1066–1079, Oct 2008.
- [73] Romualdo Pastor-Satorras and Alessandro Vespignani. Epidemic Spreading in Scale-Free Networks. *Physical Review Letters*, 86(14):3200–3203, April 2001.
- [74] R. Pastor-Satorras and A. Vespignani. Epidemic dynamics in finite size scale-free networks. *Physical Review E*, 2002.
- [75] C.C. Zou, D. Towsley, and Weibo Gong. Modeling and simulation study of the propagation and defense of internet e-mail worms. *Dependable and Secure Computing, IEEE Transactions on*, 4(2):105–118, April 2007.
- [76] Cong Li, Ruud van de Bovenkamp, and Piet Van Mieghem. Susceptible-infected-susceptible model: A comparison of n -intertwined and heterogeneous mean-field approximations. *Phys. Rev. E*, 86:026116, August 2012.

- [77] E Cator and P Van Mieghem. Susceptible-infected-susceptible epidemics on the complete graph and the star graph: Exact analysis. *Physical Review E*, 87(1):012811, January 2013.
- [78] P L Simon, M Taylor, and I Z Kiss. Exact epidemic models on graphs using graph-automorphism driven lumping. *Journal of mathematical biology*, 2011.
- [79] Deepayan Chakrabarti, Yang Wang, Chenxi Wang, Jurij Leskovec, and Christos Faloutsos. Epidemic thresholds in real networks. *ACM Transactions on Information and System Security*, 10(4):1–26, January 2008.
- [80] D. Chakrabarti, J. Leskovec, C. Faloutsos, S. Madden, C. Guestrin, and Michalis Faloutsos. Information survival threshold in sensor and p2p networks. In *INFOCOM 2007. 26th IEEE International Conference on Computer Communications. IEEE*, pages 1316–1324, May 2007.
- [81] Sheng Wen, Wei Zhou, Jun Zhang, Yang Xiang, Wanlei Zhou, Weijia Jia, and C.C. Zou. Modeling and analysis on the propagation dynamics of modern email malware. *Dependable and Secure Computing, IEEE Transactions on*, 11(4):361–374, July 2014.
- [82] Sheng Wen, Wei Zhou, Jun Zhang, Yang Xiang, Wanlei Zhou, and Weijia Jia. Modeling Propagation Dynamics of Social Network Worms. *Parallel and Distributed Systems, IEEE Transactions on*, 24(8):1633–1643, 2013.
- [83] F D Sahneh, C Scoglio, and P Van Mieghem. Generalized Epidemic Mean-Field Model for Spreading Processes Over Multilayer Complex Networks. *Networking, IEEE/ACM Transactions on*, 21(5):1609–1620, 2013.
- [84] Shouhuai Xu, Wenlian Lu, and Zhenxin Zhan. A stochastic model of multivirus dynamics. *Dependable and Secure Computing, IEEE Transactions on*, 9(1):30–45, Jan 2012.

- [85] P Van Mieghem and J Omic. In-homogeneous virus spread in networks. *arXiv.org*, 2013.
- [86] W. K. Chai and G. Pavlou. Path-based epidemic spreading in networks. *IEEE/ACM Transactions on Networking*, 25(1):565–578, February 2017.
- [87] Jooho Kim and Makarand Hastak. Social network analysis: Characteristics of online social networks after a disaster. *International Journal of Information Management*, 38(1):86–96, 2018.
- [88] Romualdo Pastor-Satorras, Claudio Castellano, Piet Van Mieghem, and Alessandro Vespignani. Epidemic processes in complex networks. *Reviews of modern physics*, 87(3):925, 2015.
- [89] Lauren Ancel Meyers, MEJ Newman, and Babak Pourbohloul. Predicting epidemics on directed contact networks. *Journal of theoretical biology*, 240(3):400–418, 2006.
- [90] Cong Li, Huijuan Wang, and Piet Van Mieghem. Epidemic threshold in directed networks. *Phys. Rev. E*, 88(6):062802, December 2013.
- [91] Ali Khanafer, Tamer Başar, and Bahman Ghahesifard. Stability of epidemic models over directed graphs: A positive systems approach. *Automatica*, 74:126–134, 2016.
- [92] Trend Micro. Taxonomy of botnet threats. *Whitepaper*, November, 2006.
- [93] Sang-Kyun Noh, Joo-Hyung Oh, Jae-Seo Lee, Bong-Nam Noh, and Hyun-Cheol Jeong. Detecting p2p botnets using a multi-phased flow model. In *Digital Society, 2009. ICDS'09. Third International Conference on*, pages 247–253. IEEE, 2009.

- [94] James R Binkley and Suresh Singh. An algorithm for anomaly-based botnet detection. In *Proceedings of USENIX Steps to Reducing Unwanted Traffic on the Internet Workshop (SRUTI)*, pages 43–48, 2006.
- [95] Hyunsang Choi, Hanwoo Lee, Heejo Lee, and Hyogon Kim. Botnet detection by monitoring group activities in dns traffic. In *Computer and Information Technology, 2007. CIT 2007. 7th IEEE International Conference on*, pages 715–720. IEEE, 2007.
- [96] G Gu, J Zhang, and W Lee. Botsniffer: Detecting botnet command and control channels in network traffic. In *Proceedings of the 15th Annual Network and Distributed System Security Symposium, NDSS*, 2008.
- [97] Guofei Gu, Roberto Perdisci, Junjie Zhang, and Wenke Lee. Botminer: Clustering analysis of network traffic for protocol- and structure-independent botnet detection. In *Proceedings of the 17th conference on Security symposium*, pages 139–154. USENIX Association, 2008.
- [98] Shun-Te Liu, Yi-Ming Chen, and Shiou-Jing Lin. A novel search engine to uncover potential victims for apt investigations. In *Network and Parallel Computing*, pages 405–416. Springer, 2013.
- [99] Marco Balduzzi, Vincenzo Ciangaglini, and Robert McArdle. Targeted attacks detection with sponge. In *Privacy, Security and Trust (PST), 2013 Eleventh Annual International Conference on*, pages 185–194. IEEE, 2013.
- [100] H Van Dyke Parunak, Alex Nickels, and Richard Frederiksen. An agent-based framework for dynamical understanding of dns events (dude). In *Proceedings of the 1st International Workshop on Agents and CyberSecurity*, page 6. ACM, 2014.

- [101] Nikita Spirin and Jiawei Han. Survey on web spam detection: principles and algorithms. *ACM SIGKDD explorations newsletter*, 13(2):50–64, 2012.
- [102] Emilio Ferrara, Onur Varol, Clayton Davis, Filippo Menczer, and Alessandro Flammini. The rise of social bots. *Commun. ACM*, 59(7):96–104, June 2016.
- [103] C. Chen, K. Wu, V. Srinivasan, and X. Zhang. Battling the internet water army: Detection of hidden paid posters. In *Proc. IEEE/ACM Int. Conf. Advances in Social Networks Analysis and Mining (ASONAM '13)*, pages 116–120, Aug 2013.
- [104] Danah M Boyd and Nicole B Ellison. Social network sites: Definition, history, and scholarship. *J. Computer-mediated Comm.*, 13(1):210–230, 2007.
- [105] Hunt Allcott and Matthew Gentzkow. Social media and fake news in the 2016 election. *J. Econ. Perspectives*, 31(2):211–36, 2017.
- [106] Michael Luca and Georgios Zervas. Fake it till you make it: Reputation, competition, and yelp review fraud. *Manage. Science*, 62(12):3412–3427, 2016.
- [107] Fabrcio Benevenuto, Gabriel Magno, Tiago Rodrigues, and Virglio Almeida. Detecting spammers on twitter. In *Proc. 7th Annu. Collaboration, Electronic messaging, Anti-Abuse and Spam Conf. (CEAS '10)*, 2010.
- [108] Justin Malbon. Taking fake online consumer reviews seriously. *J. Consumer Policy*, 36(2):139–157, Jun 2013.
- [109] Y. Wang, A. V. Vasilakos, J. Ma, and N. Xiong. On studying the impact of uncertainty on behavior diffusion in social networks. *IEEE Trans. Syst., Man, Cybern., Syst.*, 45(2):185–197, Feb 2015.

- [110] X. Wang, W. Ni, K. Zheng, R. P. Liu, and X. Niu. Virus propagation modeling and convergence analysis in large-scale networks. *IEEE Trans. Inf. Forensics Security*, 11(10):2241–2254, Oct 2016.
- [111] R. Wald, T. M. Khoshgoftaar, A. Napolitano, and C. Sumner. Predicting susceptibility to social bots on twitter. In *Proc. IEEE 14th Int. Conf. Inform. Reuse Integration (IRI '13)*, pages 6–13, Aug 2013.
- [112] Garrett Brown et al. Social networks and context-aware spam. In *Proc. ACM Conf. Computer Supported Cooperative Work (CSCW '08)*, pages 403–412, New York, NY, USA, 2008. ACM.
- [113] X. Zha, W. Ni, X. Wang, R. P. Liu, Y. J. Guo, X. Niu, and K. Zheng. The impact of link duration on the integrity of distributed mobile networks. *IEEE Trans. Inf. Forensics Secur.*, 13(9):2240–2255, Sep 2018.
- [114] Arjun Mukherjee, Vivek Venkataraman, Bing Liu, and Natalie S Glance. What yelp fake review filter might be doing? In *Proc. Int. AAAI Conf. Weblogs and Social Media (ICWSM '13)*, pages 409–418, 2013.
- [115] Groups. <https://www.facebook.com/help/1629740080681586>, 2018.
- [116] Ehud Kalai and Ehud Lehrer. Rational learning leads to nash equilibrium. *Econometrica*, 61(5):1019–1045, 1993.
- [117] T. G. Papaioannou and G. D. Stamoulis. An incentives' mechanism promoting truthful feedback in peer-to-peer systems. In *Proc. CCGrid*, volume 1, pages 275–283, May 2005.
- [118] Julian L Coolidge. The story of the binomial theorem. *The American Mathematical Monthly*, 56(3):147–157, 1949.

- [119] R. M. Corless, G. H. Gonnet, D. E. G. Hare, D. J. Jeffrey, and D. E. Knuth. On the Lambert W function. *Advances in Computational Mathematics*, 5(1):329–359, Dec 1996.
- [120] A. Ganesh, L. Massoulié, and D. Towsley. The effect of network topology on the spread of epidemics. In *INFOCOM 2005. 24th Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings IEEE*, volume 2, pages 1455–1466 vol. 2, March 2005.
- [121] Yang Wang, D Chakrabarti, Chenxi Wang, and C Faloutsos. Epidemic spreading in real networks: an eigenvalue viewpoint. In *Reliable Distributed Systems, 2003. Proceedings. 22nd International Symposium on*, pages 25–34. IEEE Comput. Soc, 2003.
- [122] R Albert and A L Barabási. Statistical mechanics of complex networks. *Reviews of modern physics*, 2002.
- [123] Yang Wang and Chenxi Wang. *Modeling the Effects of Timing Parameters on Virus Propagation*. WORM '03. ACM, New York, NY, USA, 2003.
- [124] Carl D Meyer. *Matrix analysis and applied linear algebra*. Siam, 2000.
- [125] James W Demmel. *Applied numerical linear algebra*. Siam, 1997.
- [126] Stephen Boyd, Persi Diaconis, and Lin Xiao. Fastest Mixing Markov Chain on a Graph. *Siam Review*, 46:667–689, January 2004.
- [127] Lukas Ahrenberg. Nepidemix. <http://nepidemix.irmacs.sfu.ca/>.
- [128] Leo P Kadanoff. More is the same; phase transitions and mean field theories. *Journal of Statistical Physics*, 137(5-6):777, 2009.

- [129] P. Van Mieghem and R. van de Bovenkamp. Non-markovian infection spread dramatically alters the susceptible-infected-susceptible epidemic threshold in networks. *Phys. Rev. Lett.*, 110:108701, March 2013.
- [130] ALBERT-LSZL BARABSI and ERIC BONABEAU. Scale-free networks. *Scientific American*, 288(5):60–69, 2003.
- [131] Huiyan Kang and Xinchu Fu. Epidemic spreading and global stability of an sis model with an infective vector on complex networks. *Communications in Nonlinear Science and Numerical Simulation*, 27(1):30 – 39, 2015.
- [132] Tao Li, Xiongding Liu, Jie Wu, Chen Wan, Zhi-Hong Guan, and Yuanmei Wang. An epidemic spreading model on adaptive scale-free networks with feedback mechanism. *Physica A: Statistical Mechanics and its Applications*, 450:649 – 656, 2016.
- [133] Aleksandr Mikhailovich Lyapunov. The general problem of the stability of motion. *Int. J. Control*, 55:531–534, June 1992.
- [134] M. J. Keeling. The effects of local spatial structure on epidemiological invasions. *Proceedings of the Royal Society of London B: Biological Sciences*, 266(1421):859–867, 1999.
- [135] Kieran J. Sharkey, Carmen Fernandez, Kenton L. Morgan, Edmund Peeler, Mark Thrush, James F. Turnbull, and Roger G. Bowers. Pair-level approximations to the spatio-temporal dynamics of epidemics on asymmetric contact networks. *Journal of Mathematical Biology*, 53(1):61–85, July 2006.
- [136] David Arrowsmith and Colin M Place. *Dynamical systems: differential equations, maps, and chaotic behaviour*. CRC Press, Chapman and Hall, London, 1992.

- [137] N. A. S. Mirza, H. Abbas, F. A. Khan, and J. Al Muhtadi. Anticipating advanced persistent threat (apt) countermeasures using collaborative security mechanisms. In *2014 International Symposium on Biometrics and Security Technologies (ISBAST)*, pages 129–132, Aug 2014.
- [138] DeepEnd Research. Malware traffic patterns.
<https://docs.google.com/spreadsheet/ccc?key=0AjvsQV3iSLa1dDFfWHduQ1A5THBRd081eFhsZThwU1E#gid=3>, 2013 11.
- [139] davey winder. Persistent and evasive attacks uncovered. *Infosecurity*, 2011.
- [140] William W Cohen. Fast effective rule induction. In *Proceedings of the Twelfth International Conference on Machine Learning, Lake Tahoe, California*, 1995.
- [141] Paul S Ferrell. Apt infection discovery using dns data. Technical report, Los Alamos National Laboratory (LANL), 2013.
- [142] Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H Witten. The weka data mining software: an update. *ACM SIGKDD explorations newsletter*, 11(1):10–18, 2009.
- [143] Scott McMillan, Naomi Anderson, Jonathan Chu, Derrick Karimi, Andrew Mellinger, David Shepard, and Eric Werner. C3E DNS Challenge Report. In *C3E workshop report*, January 2013.