SOCIAL NETWORK ANALYSIS OF MALICIOUS WEBSITES FOR DETECTION AND CHARACTERIZATION

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF INFORMATICS OF
THE MIDDLE EAST TECHNICAL UNIVERSITY
BY

MUHSİN ALDEMİR

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE
OF MASTER OF SCIENCE
IN
THE DEPARTMENT OF INFORMATION SYSTEMS

DECEMBER 2019

SOCIAL NETWORK ANALYSIS OF MALICIOUS WEBSITES FOR DETECTION AND CHARACTERIZATION

Submitted by Muhsin Aldemir in partial fulfillment of the requirements for the degree of **Master of Science in Information Systems, Middle East Technical University** by,

Prof. Dr. Deniz Zeyrek Bozşahin
Dean, **Graduate School of Informatics** _____

Prof. Dr. Sevgi Özkan Yıldırım
Head of Department, **Information Systems** _____

Assoc. Prof. Dr. Banu Günel Kılıç
Supervisor, **Information Systems, METU** _____

Examining Committee Members:

Assoc. Prof. Dr. Aysu Betin Can
Information Systems, METU _____

Assoc. Prof. Dr. Banu Günel Kılıç
Information Systems, METU _____

Assoc. Prof. Dr. Altan Koçyiğit
Information Systems, METU _____

Prof. Dr. Şeref Sağıroğlu
Computer Engineering Dept., Gazi University _____

Assoc. Prof. Dr. Tuğba Taşkaya Temizel
Information Systems, METU _____

Date: 02/12/2019

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last name : Muhsin Aldemir

Signature : _____

ABSTRACT

SOCIAL NETWORK ANALYSIS OF MALICIOUS WEBSITES FOR DETECTION AND CHARACTERIZATION

Aldemir, Muhsin

MSc., Department of Information Systems

Supervisor: Assoc. Prof. Dr. Banu Günel Kılıç

December 2019, 66 pages

Malicious websites pose major risks to users and businesses including economic damages, privacy breaches and loss of valuable data. Malicious actors use websites as a spreading medium for their motives. Analyzing the relationships between malicious websites and comparing them to benign ones can help understand the problem better, and enable detection and prevention of these websites more accurately.

This thesis focuses on detection and characterization of malicious websites using Social Network Analysis (SNA). SNA provides powerful methodologies for discovering and visualizing the relationships between actors. By utilizing the links in between and among malicious and benign websites, graphs were constituted, whose nodes were websites and ties were hyperlinks between them. For this purpose, the data which included the snapshot of the pairwise links amongst hundreds of thousands of websites, the list of malicious websites and their types were obtained from the web. First, networks of malicious websites were formed. Then, using these networks new analyses were carried out to efficiently find malicious websites and their types based on their network structures and link similarities. Results were presented showing the detection accuracies of applied methods.

Keywords: Malicious Websites, Social Network Analysis, Webgraphs, Crawler, Malware

ÖZ

ZARARLI WEBSİTELERİNİN TESPİTİ VE KARAKTERİZASYONU İÇİN SOSYAL AĞ ANALİZİ

Aldemir, Muhsin

Yüksek Lisans, Bilişim Sistemleri Bölümü

Tez Yöneticisi: Doç. Dr. Banu Günel KILIÇ

Aralık 2019, 66 sayfa

Zararlı web siteleri kullanıcılar ve şirketler için ekonomik zararlar, mahremiyet ihlalleri ve değerli veri kayıpları gibi büyük riskler oluştururlar. Kötü niyetli aktörler web sitelerini amaçları için bir yayılma aracı olarak kullanırlar. Zararlı web siteleri arasındaki ilişkiyi analiz etmek ve bunları zararsız web siteleri ile karşılaştırmak sorunu daha iyi anlamaya yardımcı olabilir ve böylece bu websitelerini daha doğru bir şekilde tespit edip önlemeye imkan sağlayabilir.

Bu tez, Sosyal Ağ Analizi (SAA) kullanarak kötü amaçlı web sitelerinin tespitine ve karakterizasyonuna odaklanmaktadır. SAA, aktörler arasındaki ilişkileri keşfetmeye ve görselleştirmeye yardımcı olan güçlü metodolojiler sunar. Zararlı ve zararsız web sayfalarının kendi aralarında ve birbirleriyle olan bağlantılarını kullanarak düğümleri web siteleri, bağları da aralarındaki linkler olan çizgeler oluşturuldu. Bu amaçla, yüz binlerce web sayfası arasındaki karşılıklı bağlantıların anlık durum görüntüsünü, zararlı web sitelerinin listesini ve onların tiplerini içeren veri webden elde edildi. Öncelikle zararlı web siteleri ağları oluşturuldu. Daha sonra, bu ağları kullanarak zararlı web sitelerini ve onların tiplerini verimli biçimde bulmak için ağ yapılarına ve bağlantı benzerliklerine dayanan yeni analizler yapıldı. Uygulanan metodların tespit doğruluğunu gösteren sonuçlar sunuldu.

Anahtar Sözcükler: Zararlı Web siteleri, Sosyal Ağ Analizi, Web Çizgeleri, Web Tarayıcı Robotları, Zararlı Yazılımlar

To My Family

ACKNOWLEDGMENTS

Firstly, I would like to express my deepest appreciation to my advisor, Assoc. Prof. Dr. Banu Günel Kılıç for her constant support, advice, and mentoring during my graduate studies and writing of this thesis.

I would like to extend my sincere thanks to my examining committee members Assoc. Prof. Dr. Aysu Betin Can, Assoc. Prof. Dr. Altan Koçyiğit, Prof. Dr. Şeref Sağıroğlu and Assoc. Prof. Dr. Tuğba Taşkaya Temizel for their valuable feedbacks and participation in my thesis defense.

I am forever grateful to my family, my mom, dad and brother for their unconditional support throughout all my life.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

AJAX	Asynchronous JavaScript and XML
API	Application Programming Interface
DNS	Domain Name System
EQ	Equation
GB	Gigabyte
HTML	Hyper Text Markup Language
IOT	Internet Of Things
LIFO	Last In First Out
OS	Operating System
RQ	Research Question
SEO	Search Engine Optimization
SNA	Social Network Analysis
TB	Terabyte
URL	Uniform Resource Locator
VM	Virtual Machine
WARC	Web Archive
WWW	World Wide Web

CHAPTER 1

INTRODUCTION

Computer and network security have always been a big concern for users. Malicious software, often called as malwares, are specifically designed programs to cause harmful or unwanted actions on computer users and computer systems. These effects could be stealing information, disrupting and damaging the computer systems, breaching user's privacy and economic losses. Malware is a broad term to generalize many types of malicious programs like adwares, bots, rootkits, ransomwares, spywares, trojan horses, viruses, and worms. Effects and motives of these programs differ from each other based on their categories. On their infected systems, some show advertisements, some steal information, some make the system unusable and some others make the system a member of a botnet, etc.

A malware can infect a system in many ways. Most common infection vectors are spam emails, infected removable drives, malicious executables and websites. They are called malware distribution channels. Malicious websites might be set up with evil motives initially or they can be legitimate websites, which are compromised later on to serve malicious content. World Wide Web (WWW) as a malware distribution channel is not a new phenomenon. Since its commercialization and opening to the masses, malware has been a huge problem for WWW. Today, the number of Internet connected devices are enormous and with the advancement in the technology and Internet of Things (IOT), malware problem is expected to grow even larger [1].

Malware detection and prevention are vast research areas. Today, both the solutions offered by the security industry and techniques to evade detection employed by malicious actors are sophisticated. Computer security industry with different cyber security solutions tries to protect users and computer systems.

Detecting malware and preventing its infection are difficult processes. Malware research is often done by analyzing the malware itself and its interactions on the host computer. However, the analysis of malware distribution channels and their mechanism is as important as the malware analysis itself.

To study malicious websites and malware distribution problem on the Web, we need a good research approach. Modeling the relationships among websites will help us to illustrate the structure of the group, so that several analyses can be carried out on them later on. Due to its well-established grounds Social Network Analysis (SNA) is a good choice to study the relationship between malicious websites. Social Network Analysis is a set of methods to define and understand a network structure. There are many problems that can be answered by utilizing the SNA approach. Applying social

network analysis on security will widen the possibilities of findings and will help us to illustrate the relationships better.

A social network consists of entities and relationships between them. These entities might be websites, people, organizations, individual states, group of companies, etc. In SNA terminology these entities are often called as nodes or vertices. Relationships between them could be one of many types such as friendship, collaboration, business ties, trust, information exchange, etc. Many objects we have in the real world and relationships between them can be classified and analyzed as a social network. Friendships among people, collaboration between academics on a conference paper, electrical power grids of a city are all social networks.

In SNA, observed attributes between actors are understood with regards to patterns or structures of ties among them. Relational ties between actors such as friendship status between them have primary status whereas the attributes of actors like their ages have secondary status [2]. Therefore an attribute would lack its exact meaning if it is taken solely without a relationship accompanying it. So, a pattern or a structure of ties and relationships are necessary to interpret the attributes correctly.

Networks and network analysis have been the subject of many academic disciplines such as mathematics, sociology, computer sciences, biology and economics. Those disciplines have some differences in terms of explaining and analyzing networks related to their respective fields. Network analysis is an interdisciplinary subject and this thesis benefits from the interdisciplinary status of the research area. Computer science constitutes the data collection and analysis efforts of the thesis. Mathematics with its graph theory paves the way for forming and understanding network structures. Also, the propagation of malware on malicious websites can be modeled similar to the propagation of an infectious disease in biological sciences.

1.1. Research Questions

This thesis tries to answer mainly two research questions.

R.Q. 1) Linked nature of the Web lets us analyze interactions between websites. Since Web is a very large domain consisting of millions of websites and our analysis capacity is not unlimited, we need to make a priority in selection of websites to analyze. How this selection can be done effectively with link analysis? By effective we mean analyzing smaller number of websites but running into more malicious ones in them.

R.Q. 2) Malicious websites have threat types in terms of malicious content they present like malware and phishing. Can we make successful predictions for their threat types by utilizing their links?

1.2. Thesis Importance and Impacts

Malicious websites pose great risks to computer users. These risks include economic damages, privacy breaches and loss of valuable data. In recent years many high scale security breaches have occurred which caused many damages [1][3].

As much as being an indispensable part of people's lives today, Web is also an important distribution medium for malicious programs. Google operates a Safe Browsing initiative which finds malicious websites on the Web. It indicates that there are currently 30,557 malware websites and 1,637,637 phishing websites on the Internet, which are deemed dangerous [4]. Also they detect thousands of new malware and phishing websites everyday [5].

By understanding the relationship between malicious websites on the Web and mapping out their interactions, we can understand the problem better. Thus, we can identify and prevent them more accurately.

1.3. Thesis Scope

Malware research is a vast and popular academic research area. There are various detection methods offered by researchers for malware detection or analysis on a website to determine whether it has any malicious content or not. This study does not aim proposing an alternative method, but a method to accompany them.

This thesis takes a social network approach and tries to understand the relationships in the network of malicious websites. Some research techniques developed in the domains of social network analysis and graph theoretic models will help us to form the networks and illustrate the relationships. As a result, we aim to provide an efficient selection method for websites that are going to be analyzed since malicious website detection is an expensive process. For each different malicious type, detection methods differ. Therefore, to find which detection techniques should be applied primarily to a suspected website we try to predict its threat type by utilizing its links.

1.4. Thesis Overview

Chapter 2 provides the related works on the subject. Chapter 3 presents the methodology we employed. In order to use SNA metrics, one needs some fundamental knowledge of the topics it covers. Therefore, this chapter begins with an introduction to SNA topics and its specific terms. Afterwards, data collection methods are presented and efforts about transitioning data to network are mentioned. In Chapter 4, finding malicious websites, we explain the prioritization process in selection of seed websites that will let us run into more malicious websites afterwards. Chapter 5, threat type identification, explains how threat type identification of a malicious website can be done with different methods based on link analysis and presents the results. Chapter 6 is the conclusions part where results

obtained in the study are summarized and contributions and future work are mentioned.

CHAPTER 2

RELATED WORKS

As detection of malicious websites is a popular research topic there are many studies in the literature. The important ones that are related to the scope of this thesis are mentioned in this chapter. Some related studies that are based on social network analysis are also summarized.

2.1 Malicious Website Detection

Studies in this section shows how traditional malicious website detection works.

Malware detection is often done by analyzing the malware itself and its interactions. These efforts include steps like static analysis, dynamic analysis, reverse engineering of malware and code analysis [6]. Static analysis tries to detect malware without running it whereas in dynamic analysis malware files are run within a secure environment.

Two notable studies made on malware detection on the web that use virtual machines were done by [7] and [8]. Virtual machine-based approaches run suspicious files in a secure, isolated environment and detect malware with a high accuracy, but they are computationally expensive.

The study [7] analyzed malicious attacks on the Web using a crawler. They limited their analysis on a specific type of malware named spyware. Their aim was to find spyware's prevalence on the Internet and a user's likelihood of encountering it. They tried to answer the questions: Which spyware is the most common on the Web and are spyware executables distributed evenly on the Web or are they concentrated in specific areas? Which websites and the site category distribute the most spyware? They initially crawled over 2500 websites and iteratively added links in them up to depth three. To identify malicious software, they downloaded the executables to Virtual Machine (VM), installed or executed it and used a commercial anti-spyware tool to identify if it was flagged as malicious or not. To identify drive-by download attacks, they visited a webpage with an unmodified browser in a VM and applied a heuristic method by using triggers in the Operating System (OS) such as observing a new library installation or creation of a new suspicious process after visiting the web page. They crawled around 18 million URLs and found spyware in 13.4% out of the 21.200 executables they come across. Also, they found drive-by download attacks in 5.9% of the web pages they visited. The most common type of spyware they found was adware or browser hijackers while others like dialers, keyloggers and trojan downloaders had relatively small prevalence. They found that spywares are not

evenly distributed and the domains which serve the most spyware programs were in the games category while news category websites had the least malicious content.

The study [8] developed an automated web patrol system to find malicious web sites that exploit browser vulnerabilities to install malware. They used a computer program called honey monkey which is similar to honey pots, but takes a more active approach in detecting threats by imitating a human's web usage. Honey monkeys running on OSes with different patch levels launch a browser instance to visit input URLs. To understand if an exploit happens, they try to find unauthorized/suspicious file creations and configuration changes. After each visit to the webpage their program generated a report containing creation or changes of files outside of the browser sandbox, OS process creation, and changes in OS registry entries, exploited vulnerability which was found based on a signature of the exploit, and visitation records based on redirection. All these operations were done in a virtual machine environment. One of the important aspects of their study was that after they found exploit sites they constructed topology graphs based on traffic redirections on them. Thus, they were able to find important actors who are responsible for these exploit webpages. Their aim was to determine both the real source of malware and other web pages that are involved in the distribution chain. They found that many exploit URLs found in the first stage did not distribute exploits themselves, but acted as a storefront, attracting large traffic and then redirecting this traffic to actual exploit distributors. They showed that redirections are big culprits in exploit distribution.

As mentioned before, Google Safe Browsing also operates scanners to find malware and phishing websites. Google Safe Browsing does not disclose its efforts in detail, but there are two publications that explain its methodology. In their study that shows how web based malware operates [9] found a large number of websites that compromise users' browsers. The study [10] also showed that how drive-by download attacks occur and how they detected them. In their study, they found over 3 million malicious URLs that trigger drive-by downloads.

2.2 Efficiency in Malicious Website Detection

Searching for malicious websites is usually a three-step process. First step is finding the URLs to analyze, which is usually done by a crawler. In the second step, these URLs are quickly inspected using fast analysis filters to dismiss possibly benign ones. These filters usually examine different features of a page such as HTML content and JavaScript functions. This step may result in imprecise decisions, but it is necessary since there are more websites than the available analysis resources. In the last step, in depth analysis of a web page is done by specialized analyzers like honey clients.

The work [11] argued that even though this three-step process is an applicable method, it is not efficient, since it requires considerable time and computing resources. In order to do it more efficiently they proposed employing a focused crawler named Malcrawler that seeks more malicious websites than the benign ones

compared to a generic crawler and as a result, increasing the toxicity of the crawled URLs. Their method focused on JavaScript based malwares. They began their crawling with a seed set of malicious URLs. For each URL they encountered they visited the webpage and extracted 10 different features such as redirections, number of dynamic code executions, bytes allocated in memory space etc. Then using already known malicious websites as a training set, they made classifications, but they also used Google Safe Browsing to cross-check the validity of their findings. As a result of classifications, links which looked like malicious were followed and others which did not look like malicious were removed and not followed. To test the effectiveness of their crawler, they made two crawls. In the first one they made a generic crawl where they followed every link, and in the second one they made a crawl with their proposed method, i.e. not crawling links which did not seem to be malicious. In both of these crawls they visited around 0.57 million URLs. In the first crawl they encountered 702 (0.123%) URLs and in the second crawl they encountered 1978 (0.348%) URLs. By applying their method, they were able to increase the number of malicious URLs they gathered from 702 to 1978.

With the similar aim, to improve efficiency at finding URLs to analyze, the work [12] proposed a method called Evilseed. They extracted the characterizing similarities between known malicious web pages and using them generated specific search engine queries to detect other malicious pages that are similar or related to known malicious ones instead of randomly looking for malicious websites on the Web. The idea is that a feature shared by many known malicious web pages is an indication of malicious activity and other pages that have this feature are more probably to be malicious. However, different than Malcrawler, they utilized the search engines to find these websites instead of crawling themselves. They had five different feature sets as links, content, search engine optimization (SEO), domain registration and DNS queries. They regarded a web page malicious if it executed a drive-by download attack or tried to trick a user to install a fake anti-virus program. Using their features, they formed a candidate malicious URLs lists and checked whether they were malicious or not by using Google Safe Browsing, a client honeypot and a custom fake anti-virus detector. For link feature they first found web pages that link to known malicious web pages and then extracted other URLs they contained and added them as candidate malicious URLs. For content feature they extracted common terms that occur in known malicious web pages and found other pages that included the same terms and added them as candidate malicious URLs. For SEO feature they found SEO campaigns that tried to boost a malicious page's position in the search results and gathered all other pages in the same domain and added them and all the links they contained to the candidate malicious URLs. For domain registration feature they found the registration times for malicious websites and added other websites that has been registered moments before or after that website to the candidate malicious URLs. For DNS feature, they monitored the DNS queries made by a large user base and when a DNS query for a malicious web page is made, they added the other URLs whose DNS queries made in the preceding four seconds to the candidate malicious URLs. They compared their findings first with a random web search and then with a crawler. Evilseed gathered 226,140 URLs of which 3036 (1.34%) were found to be malicious whereas crawler gathered 431,428

URLs of which 604 (0.14%) were found to be malicious and random web search made on a search engine included 219 (0.34%) malicious URLs out of 64,411 URLs. Their method was able to find more malicious URLs than both random search and crawler.

2.3 Link Analysis-Based Methods

Link analysis methods can be used to select websites for potential detection. Link analysis methods on the WWW are mainly used for website classification, or also known as website categorization, which is the method of assigning one or more categories to a website. This classification could be done based on several different purposes like finding the category of a webpage such as entertainment, gaming, news etc, or to find a website's function, i.e. the role it plays like personal page or corporate page, search engine etc. For these type of classification tasks several different methods are proposed like using textual and visual features. However, since we are interested in application of link analysis, we will analyze the methods involving it. Link based classification methods on WWW are used mainly on topical categorization of websites. The work [13] used machine learning methods to classify a web page and showed that utilizing the classes of hyperlinked neighbors of the page to be classified greatly helps the classification accuracy. The work [14] showed how class categories from the neighboring web pages, pages that have link relationship with the page being analyzed, can be used for classification tasks. They used four categories of neighbors: parent, child, sibling and spouse and found out that sibling neighbors help the classification tasks a lot. By adding neighboring pages' classes into their analysis, they improved accuracy over common text classification approaches. In their study to find related web pages in WWW, the study [15] used HITS algorithm [16] to find similar web pages to a provided web page using only the link analysis. These studies show that link analysis helps in terms of classification of websites.

Link analysis has some applications in classifying websites in terms of their security status, i.e. whether they are malicious or not and detection of their attack type. Many studies that apply link analysis to security domain are done for classifying web spam websites [17][18][19][20]. Web spam websites try to increase their rankings and get higher placement in search results by deceiving search engines. As link based ranking algorithms like PageRank [21], or HITS [16] establishes ranking by utilizing a website's links and high indegree helps to get higher rank, web spam websites exploit this feature by forming links between themselves to boost their positions. Therefore, link analysis is one of the most important features for the classification of web spam.

The work [22] showed how the link structure between web pages can be used to detect spam pages. They computed statistics for a variety of features of a website, like incoming and outgoing link numbers, links per pages and number of hostnames mapping to single IP address etc. and discovered that in some of these distributions, outlier values were detected to be web spam. Their findings show statistical analysis

on link features is a decent method to find web spam. The work [23] also used link analysis to find web spam pages. They found that distribution of indegrees usually follows the power law, but 40% of spam hosts they analyzed have indegrees in a very narrow interval which led to their detection. The work [18] says that web spam pages tend to be linked by other web spam pages and by using this notion on link analysis they identified web spam.

So, can link analysis also help us to classify other malicious categories other than spam like malware and phishing? The work [24] used machine learning models for detection and attack type identification of malicious websites. They tried to detect and identify attack types of malware, phishing and spam websites. Their feature sets include lexical, link popularity, webpage content, DNS, DNS fluxiness and network communication features. For each type of malicious websites, different features provided different results. For example, lexical features were effective at detecting phishing but not as effective at detecting malware and spam. Among their other feature sets link popularity was the most related one for us, since we did not employ others. Their link popularity feature set includes 15 features such as indegrees and ratios of spam, phishing and malware links for a website. Using all six of their feature sets together they achieved 93% accuracy at detection of attack types. However, by using single feature sets alone achieved accuracies were between 63% and 85%.

The work [25] called WebCop proposed a method that includes web graphs to find malicious web pages. They defined two types of websites that were active in malicious scene, one is malware distribution sites where actual malware reside and the second one is malware landing sites which provide links to the distribution sites. First, they found malware distribution sites with author's access to a commercial telemetry service and then they created the web graphs of these websites using a crawler. Going to the reverse direction from malware distribution sites they were able to discover landing sites and additional malware executables.

One other technique that uses web graphs to find malicious websites are trust and distrust propagation methods [26][27][28]. These methods begin with forming a reliable set of malicious or clean websites first and then using them they propagate their maliciousness or cleanliness to other websites that they have link exchange. And usually websites that have scores below or above certain thresholds are predicted to be malicious or clean.

2.4 Other Related Works on Social Network Analysis

Social network analysis is a broad research topic and since we use its methods in this study some related works are presented below.

2.4.1 Web as a Small World

The notion of small world means nodes in a network are closer to each other than we thought of them and is studied by many researchers. One of the most prominent studies are Stanley Milgram's Small World experiments in the 1960s [29]. His hypothesis was that the world is small when it's seen as a network of acquaintances between people. He asked some hundreds of people who were selected randomly in the USA to send a letter to a specific person by using other people they know personally (in first name basis) as relays. When the letters that arrived at destination were analyzed, it was shown that the average path length was around 6. It showed that people live in a small world with six degrees of separation. Although there are some criticisms for Milgram's experiments [30], this study made great contributions to lay the ground for modern social network analysis by showing that the direct and indirect relationships between people are more intense than one may think.

With respect to studies showing the small world phenomenon, the study [31] says that many biological and man-made systems are also small world networks. These networks are highly clustered, but the minimum distance between any two randomly chosen vertices is short; therefore reaching from one node to any other is shorter than one may think. She argues that the World Wide Web (WWW) which is a man-made network is also a small world network. Websites are clustered between themselves according to their categories and other factors but the distance that separates one another is only a few links away. She found that for an undirected WWW network there are 3.1 and for a directed network there are 4.2 hops on average between any two connected websites.

2.4.2 Finding Hierarchy and Important Nodes

Hierarchy shows authority relationship between nodes. Finding hierarchy and important nodes help us to unfold the structure of the network. The methods applied in two studies, which are summarized below, can be applied to identify a ranking between malicious websites.

The study [32] propose a model to find the most important nodes in a graph. Their entropy model uses text mining and natural language processing to form an information theoretic model to find the most important nodes and hidden organizational structures in a graph. They tested their model on Enron email dataset as it provides a large dataset of human interaction and shows information flow in an organization. They argue that when there is a hierarchy between nodes, to disrupt a network structure one should find important nodes or leaders whose removal will have the maximum effect on the information flow. To do so they first calculated the whole graph's entropy and then they removed nodes one by one and recalculated the graph entropy for the remaining graph. When removing nodes, they also removed their adjacent edges at length=1 (directly connected nodes) and at length=2 (not directly in contact but there is an information flow through third nodes). They applied this entropy model on Enron email dataset and argued that the node whose

absence causes most change in the graph entropy was the most important one in the graph. Their results found 2 presidents, 1 manager, 1 CEO and 1 Regular Employee as the most important nodes.

The study [33] used twelve different SNA metrics along with some machine learning models to identify the hierarchy and important people in an organization. To do this they also used Enron email dataset and an email exchange dataset of a university research group. Their model successfully found managers and important people in the network. They found that five metrics out of twelve SNA metrics they gathered from literature review were more effective in identifying influential nodes. Those successful metrics were Weighted Clique Score, HITS Authority Score, Average Distance, Markov Centrality Score, and Degree Centrality Score. Despite the title of their study, they did not apply their methodology and results on subjects related with security, but focused on finding important nodes and hierarchy in a network. They concluded that their findings can be applied as a future work in many types of communication networks including Social Network Sites, Dark Net Forums and phone records.

2.4.3 Identification of Communities on the Web and Web Graphs

In a paper that presents WebGraph Framework 1, a popular tool to study the very large web graphs [34], hypertextual form of the Web was mentioned, as well as how we could utilize it to help us to design effective crawlers and detecting online communities. Web graphs are graphs whose structure consists of elements in WWW. websites are nodes and hyperlinks in them are directed edges from website A to website B if a web page in A includes a hyperlink to a web page in B. Therefore, creating web graphs to study and apply social network analysis methods on the web are helpful.

The study [35] developed a method to find Web communities. Web is decentralized and unorganized by design which makes content analysis difficult. They argued that there are millions of web pages out there with different contents and no central authority to govern their hyperlinks, but it is found that Web self organizes itself and the link structure of it helps to identify communities effectively. They described web communities as a collection of webpages where each member has more hyperlinks to other web pages within the community than outside of the community. Their algorithm begins with a seed of input web sites and crawls them up to a certain depth, aggregate links in them and determine their community membership by calculating hyperlinks inside the community and gives them a score. Then, they add highest ranked not crawled yet web sites to the seed set and iterate their procedure. They achieved to find related communities on the web by using only the hyperlink information.

CHAPTER 3

METHODOLOGY

This chapter presents how we constructed our methodology to study our research questions. It begins with difficulties regarding detecting malicious websites, how they impose a limitation and why we need a suitable research approach. Then we give a quick introduction to SNA topics in order to understand the specific terms and applications in the following chapters. Data collection methods are mentioned in detail and the steps taken for transitioning data to form networks are also given.

In this study we primarily analyze two types of malicious websites as malware and phishing. There are various detection methods offered by researchers for malicious website detection. Phishing websites target users' inability to distinguish authentic websites from counterfeit ones via social engineering. Various methods are presented for their detection like using fuzzy data mining [36], machine learning methods [37] and visual similarity based methods [38][39]. Malware detection methods are also numerous. Signature based, anomaly based and specification based techniques are presented [40].

This study does not aim proposing an alternative method, but a method to accompany them. This thesis takes a social network approach and tries to understand the relationships in the network of malicious websites. As a result, we aim to provide an efficient selection method for websites that are going to be analyzed. For each different malicious type, detection methods differ. Therefore, in order to find which detection techniques should be applied primarily to a suspected website we try to predict its threat type.

There are some reasons that make analysis of malicious websites difficult. Firstly, there are difficulties in obtaining data. The World Wide Web (WWW) is a very large research area. There are millions of websites and billions of pages operating and new ones appear every day. Registering a domain and launching a website are easy; with automation they just require seconds. Also, closing down the website when a malware campaign ends or changing the content is easy. All of these factors make analyzing malicious websites a fast-paced and complex environment. When a researcher wants to study malicious websites what he/she gets as data is simply a glimpse or a snapshot in time as the environment changes very fast. Links between websites may disappear or new links may appear during the research timespan. This dynamic nature may lead to inconsistencies in the data collection.

Web-based malware has some differences than the other types of malware. They are environment specific, targeting a specific configuration of operating system, browser and installed plugins etc. where a vulnerability is present. They show themselves

only when right conditions occur. Therefore analyses may fail if right conditions are not achieved [41].

All of these difficulties require a suitable research approach. This study takes the approach of using the linked nature of WWW. Websites have hyperlinks, or simply called links, in their content which may lead a user from a page to another page on the same website or on a different website.

Social Network Analysis (SNA) methods will be applied to understand the relationship between malicious websites. Therefore, we need a fundamental knowledge of it. It has some specific terms that denote particular phenomenon. The section below will address some important SNA topics and its vocabulary which will be helpful to present the thesis.

3.1. Introduction To Social Network Analysis

Mathematical models are important foundational blocks of Social Network Analysis. Quantitative analytical approaches are used to illustrate and understand relations between actors in a network. Graph theory, statistical theory, and algebraic models are the biggest mathematical basis for network structures. Graph theory gives a proper representation of a social network as well as set of properties that are very useful when analyzing the network [2].

Graphs and graph distribution are an important section of network analysis. Graphs are popular and convenient ways to understand and visualize social networks.

Modeled as G (V,E):

- V is the set of vertices (nodes) in the network.

- E is the set of edges (links) between them. Edges may be directed if they have a direction and in that case, they are named as arcs.

In a network which shows hyperlink exchanges between websites:

-Websites are vertices (nodes) V={Website A, Website B, Website C, Website D}

-Edges, in this case directed edges or arcs, show which website has a hyperlink to another website. E={(Website A, Website B), (Website A, Website D), (Website B, Website C), (Website B, Website D), (Website C, Website B), (Website C, Website D)}

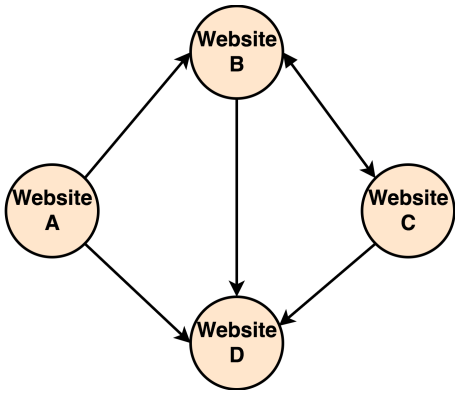

Figure 1: A four size network of hyperlinks between websites

Relationships could be symmetric if edges between two vertices simply connect them with each other and do not have a direction. Friendship networks are an example of this symmetric relationships, two people are either friends or not. Person A is a friend of Person B means that Person B is also a friend of Person A.

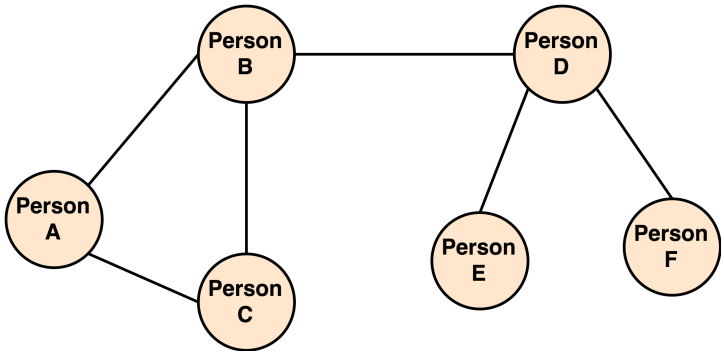

Figure 2: A symmetric network of size seven that shows friendship among a group

Asymmetric relationships occur when a node has ties to another node, but not vice versa. A relationship which shows hyperlinks from a website to others is asymmetric, because a website with a hyperlink to another website does not have to receive a hyperlink from that website as shown in Figure 1.

Each edge in a graph may have a weight which is a numerical value and shows the relevant information about the relationship. These graphs are called weighted graphs.

The term mode which is an important property of a network is related to the actor set of a network. Two types of networks exist in many real world situations [2].

One Mode Networks:

Relationships between one set of actors constitutes one mode networks. For example, the network of hyperlinks from a website to other websites is a one mode network because all actors belong to one set.

Two Mode (aka Affiliation) Networks:

Relationships between two sets of actors constitute two mode networks. For example, hyperlink from a malicious website to another website which is not malicious is a two mode network because actors do not belong to one set.

There are three main properties of a social network that define its characteristic:

1- Network Size:

Network size shows the number of nodes in a network. It is an important characteristic of a network and affects the data collection and its analysis greatly. Small networks let researchers collect and analyze data easily which can be done manually without automation. When the network size grows it is harder to collect and analyze data and an automation is required. Size also affects how relationships occur and how they are distributed in the network.

2-Community Structure:

Communities in a network are groupings which share more similar traits between themselves than others. Communities in the friendship networks show grouping of people by interest or communities on the web shows websites on a certain topic etc. Different networks have different community structures and their structure helps us to categorize and understand the dynamics in the network better.

3- Degree Distribution:

Degree of a node is the number of adjacent links to that node. Some nodes have high degrees, some have a few while others may even have zero and therefore be an isolate. Degree distribution shows how links are distributed between nodes of the network and indicate the ways relationships occur. Also, in some cases it is an important metric to find out the importance of a node or its hierarchy in an organization.

3.2 Data Collection Method

With the advancements in the Internet technology, social networks became more visible and the amount of social network data available to researchers skyrocketed. In the past, if a researcher had wanted to analyze someone's friend list, it required a great amount of effort, but now with Facebook and other social media sites it is easier to reach that information (with respect to privacy). Network sizes of SNA studies where data gathering was done by traditional methods were small, usually hundreds and maybe a few thousands [42] [43]. However, today online data collection methods allow researchers to access large sets of data. People's interaction with each other is more visible and easily observable by researchers than ever before thanks to the Internet. These new advancements brought new opportunities:

- Data collection efforts are easier especially with social media and online surveys.
- With bigger data it is possible to understand the network structure and relationships within it better.
- With the advancement in computational power and tools, it is easier to calculate metrics which needed to be done manually before.

Deciding which information will be gathered in the data collection is an important decision. Finding and modeling the relationships between actors are the most important steps that contribute to the success of an analysis. For a sound analysis, a boundary on data collection must be declared and then be adhered to clearly.

Different types of networks require different types of data collection efforts. For example, a friendship network among a student group is best formed by questionnaires. Also, co-authorship network among university academics are best formed by taking archival data from publishers. Therefore, data collection technique must comply with the type of the network that is targeted. To study malware distribution problem on the web, a crawler that visits web pages and collects information about its hyperlink structure would be an appropriate choice.

3.3 Collecting Hyperlinks From Websites

Websites, in order to function, send some files and codes to their visitor's web browsers. These files are then rendered and presented to the user by web browsers. These files also include hyperlinks in them, therefore when a crawler visits a webpage it can get its content and links in it. A web crawler is a computer program that automatically visits web pages, indexes them and follows hyperlinks in them to process more web pages. When a crawler visits a webpage, it is able to download the content from it. This content is mainly the source code of the website that is shown to the browsers/visitors. The main functionality of a crawler is that it begins with a seed page (usually top-level page in a website like www.example.com) and then finds hyperlinks in that page. It indexes those hyperlinks and visits them, finds new hyperlinks in them and does this process repeatedly until there's no new content left to be reached.

In HTML "<a>" element is used to define a hyperlink and shows where the linked content resides. A link found in a website looks like this where "href" part shows the address of the linked content:

Click to download our e-book

At the beginning of this study, we developed a custom crawler to find links in websites. However, after some time, we realized that this method would not provide a suitable data for our analyses. The main reason was that when a website is crawled, only outgoing links, i.e., links that originates from that website, are obtained. However, if you want to get incoming links to that website you need to crawl a large portion of the Web since a link may occur anywhere. This was not possible to be

done effectively, due to performance and resource limitations. The second problem was that crawling a website took a long time if all hyperlinks were to be obtained. Some websites have lots of pages or some of them create hyperlinks and pages dynamically and a crawler may end up in an infinite number of links to crawl. A solution we employed was only processing links up to depth=5 from main page or terminating the process after two minutes, which one happens first. The third problem was that some websites block crawlers and would not let you get their content.

Therefore, finding another solution was necessary. There are some already crawled web graph datasets online [44] [45]. However, most of them are on a specific area, outdated or small. The biggest and most trustworthy web crawl data is provided by Common Crawl. Common Crawl is an organization that maintains an online web crawl data repository since 2008. Their data is the biggest, open and free web crawl data available to researchers. They identify themselves as *"non-profit organization dedicated to providing a copy of the Internet to Internet researchers, companies and individuals at no cost for the purpose of research and analysis."* [46]. Common Crawl bots crawl websites on a monthly basis and take snapshots of the content and store links in them. The raw crawl data that includes everything related to a web page are stored in WARC archive format. They also provide metadata and plaintext extracts for different kinds of analysis [47].

Common Crawl also publishes host and domain level web graphs based on their crawls. These web graphs show which webpage has a link that points to another webpage. They are published once every three months since 2017. The most appropriate data we could use from Common Crawl is their domain level Web graphs. Web graphs consisting of February/March/April 2019 crawls are used in this thesis.

Table 1: Common Crawl data for Feb/Mar/Apr 2019 in numbers

Nodes	Arcs	Dangling Nodes
90,757,643	1,888,693,874	46,373,014

One important thing about this dataset is its dangling nodes: Dangling nodes are terminal nodes that are pointed by a crawled website, but they are not crawled yet. The domain-level Web graph has over 90 million nodes and 1.88 billion arcs where 51% of nodes are dangling nodes as shown in Table 1.

Trustability of Common Crawl Data In Terms of Representing Web:

Due to the gigantic size of the Web, it's not possible to collect every data out there for reasons of performance and lack of resources. Therefore, a sampling is necessary for analysis purposes. The exact details of data collection algorithm employed by

Common Crawl varies between crawls. Generally, websites that are visited are chosen by:

- A random sample of links from previous crawls
- A breadth-first crawl within a maximum of 4 or 6 links away from the homepages of the top hosts and domains based on traffic
- URLs extracted from sitemaps, RSS and Atom feeds
- URLs from less-represented languages in crawls so far [48] [49]

The study made by [50] analyzed the representativeness of using open source crawl data, Common Crawl, for online forums' topic modeling. Their study was on a particular car owner's forum where they made a custom crawler that collected 2.16 TB data and compared their results with those obtained from Common Crawl which is 280 GB of raw data files. They showed that although there are discrepancies between data obtained from crawls, they are similar in terms of topic proportions and word rankings and concluded that they are not statistically different from each other. They argue that in terms of data quality and completeness, Common Crawl data could be used instead of custom crawl data for topic modeling. Even though, their subject is different than ours, their findings indicate that the data from Common Crawl is representative of specifically crawled data.

3.4 Generating a List of Malicious Websites For Analysis

This subsection deals with steps taken to collect a list of malicious websites for our analyses.

3.4.1 Finding Malicious Websites

Security companies and some other initiatives publish blacklists of websites they regard malicious. We collected websites flagged as malicious from different sources. These sources are:

- abuse.ch [51]
- malwaredomains.com [52]
- SANS Internet Storm Center [53]
- phishtank.com [54]
- some other blacklists found on security forums [55]

One can argue the trustworthiness of these kind of blacklists. However, our aim here is to find as many websites as possible to further analyze, since they are going to be cross-checked with Virustotal. This checking process is explained below.

As a result of data collection efforts, we found 66,659 malicious websites using previously mentioned lists indexed by Common Crawl in February/March/April 2019 dataset.

3.4.2 Gathering Information About a Website's Security Status

Determining whether a website has malicious content is a difficult and computationally expensive process. Malicious content on the Web is an environment of a cat and mouse game between computer security industry and criminals. Due to its fast-paced nature there are some inconsistencies between identifications of malicious behavior. Malicious actors apply sophisticated concealment techniques to evade detection. By using proxies, redirection chains and other mechanisms, malicious websites can cloak their activities. Detection is hard and requires lots of efforts. As the environment changes rapidly, false positives and false negatives are important problems and as a result the identification of many different companies and organizations differ from each other.

There are differences between security products' evaluation on a website. A website may be classified as malicious by some vendors and clean by others. Different capabilities and detection techniques of the security products may be a reason for this. Also, the inspected site may not be analyzed yet by that product or at the time of analysis there may have no malicious content/action. In order to establish a trust to the malicious website list we gathered from different sources, Virustotal online security community is used.

Virustotal, which is run by Google's parent company Alphabet Inc. is an online collaborative security initiative. It aggregates over 70 antivirus and other security products' capabilities to check for malicious files and websites [56]. We used Virustotal Public API v2.0 [57] and inspected our initial 66,659 websites first. After the network creation step described below and in the analyses afterwards we totally inspected 2,136,836 websites on Virustotal. Virustotal API tells us how many products flagged the searched URL as malicious. In the creation of our network, for each domain we analyzed we searched its homepage on Virustotal and flagged it malicious if its homepage is flagged by at least two products. This procedure reduced our network size but made the seed set of malicious websites more trustworthy.

Table 2: Number of entries in our malicious websites dataset after Virustotal

Number of malicious websites indexed by Common Crawl in Feb/Mar/Apr 2019 Crawl	Number of Websites Left After Virustotal Evaluation (flagged by at least 2 products)
66,659	34,785

3.5 Transforming Data Into Network

After getting the data, we need to convert it into a network in order to apply Social Network Analysis methods. We created our network around the set of previously gathered 34,785 malicious websites. For each website in this list we find their incoming and outgoing links from Common Crawl Feb/Mar/Apr 2019 Crawl. Malicious websites and other websites which are these malicious websites' incoming and outgoing neighbors become nodes and links between them become arcs in our network.

Consider a scenario where we have 3 malicious websites: X, Y and Z. In order to create their network, we first find out all of their incoming and outgoing neighbors separately as shown in Figure 3. For website X, websites C and E are its incoming neighbors and websites A, D and G are outgoing neighbors. We do this process for websites Y and Z also. Beware that malicious website Y has malicious website X as outgoing and Z as incoming neighbors. Also website F is linked by both websites Y and Z and website C links both of websites X and Z.

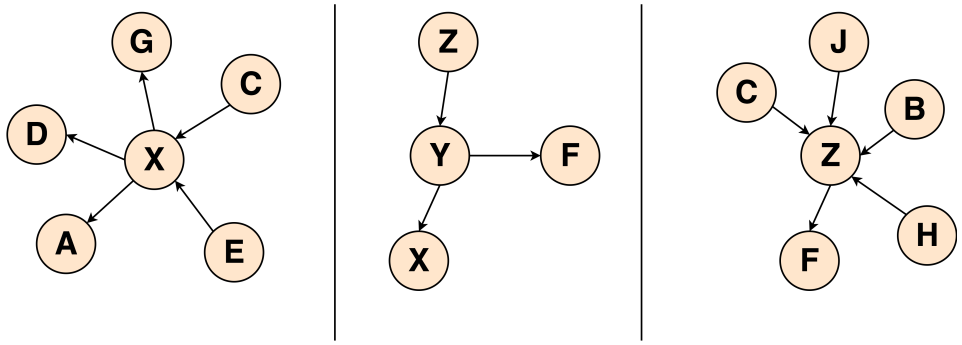

Figure 3: Separate neighbors of malicious websites X, Y and Z

After we find out every incoming and outgoing neighbors of our malicious websites we combine them together as shown in Figure 4. When we combined websites, links such as from websites Y to F and Z to F can be represented with only one F that has links from both of Y and Z.

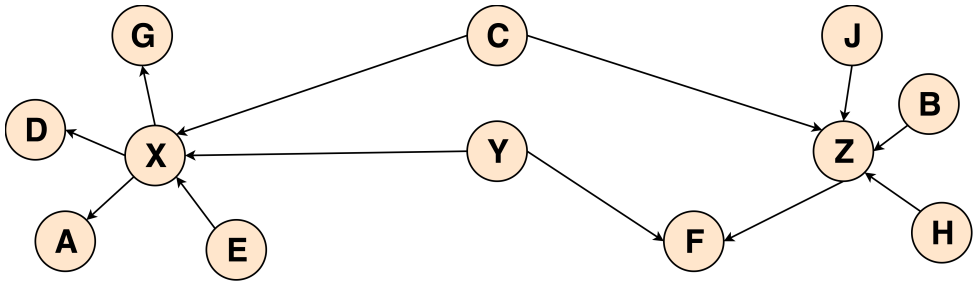

Figure 4: Network after we combine neighbors of malicious websites X, Y and Z

Beware that we built our network around the malicious websites. Website A may have a link to website D but we did not include it in our network.

To analyze the network, we used open source Pajek software developed for analyzing and visualizing networks [58]. To convert the raw data into a network, we used txt2pajek program [59].

3.6 Network Properties & Characterization

Nodes and their frequencies in our network are given in Table 3 below. We have a total of 776,230 websites in our network created from the 34,785 malicious websites and their incoming and outgoing neighbors. Out of these 34,785 malicious websites 14,543 are malware, 10,455 are phishing and 9,787 are uncategorized. These threat types are also provided by the blacklists we got these malicious websites from.

Table 3: Frequency and percentages of the threat types of websites in our network

Type	Frequency	Percentage
Malware	14,543	1.87%
Phishing	10,455	1.35%
Uncategorized	9,787	1.26%
Others	741,445	95.52%
Total	776,230	100%

Incoming and outgoing degree distributions of 34,785 malicious websites are given in Figure 5. One thing to note here is that many of malicious websites do not have any outgoing links. These websites may not have any links or for some of them this may be due to the dangling node problem in Common Crawl dataset mentioned in Chapter 3.3. Number of indegrees and outdegrees in the X axis is given in common log scale to plot the data more clearly.

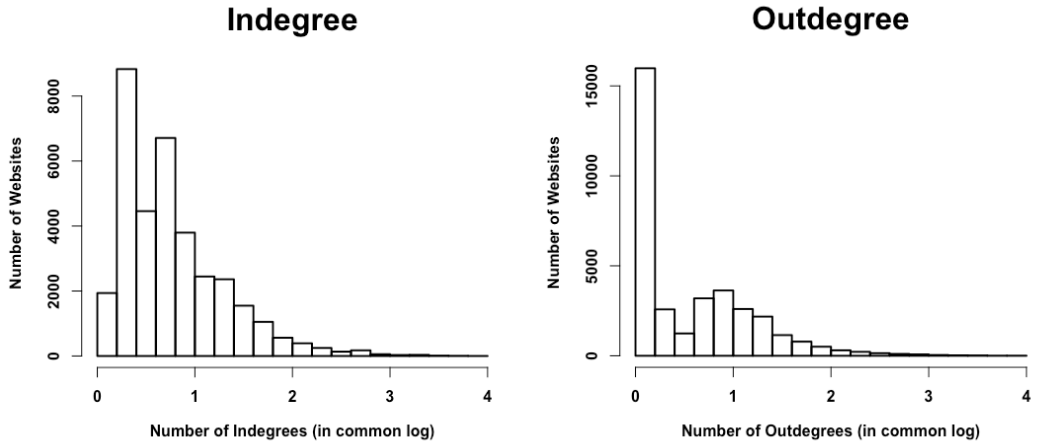

Figure 5: Indegree and outdegree distributions for 34,785 malicious websites

Analyzing the degree distributions, we see that majority of the websites have small degrees, but a little amount has orders of magnitude higher degrees. Therefore, if we plot the indegrees and outdegrees in common log-log scale we can observe power law distributions since two quantities show near linear relationships with some outliers as shown in Figure 6 below. Due to the limitations of the common log-log scale 0 values are discarded. Other studies [60][61] that uses web graphs also found power law distribution for indegrees and outdegrees.

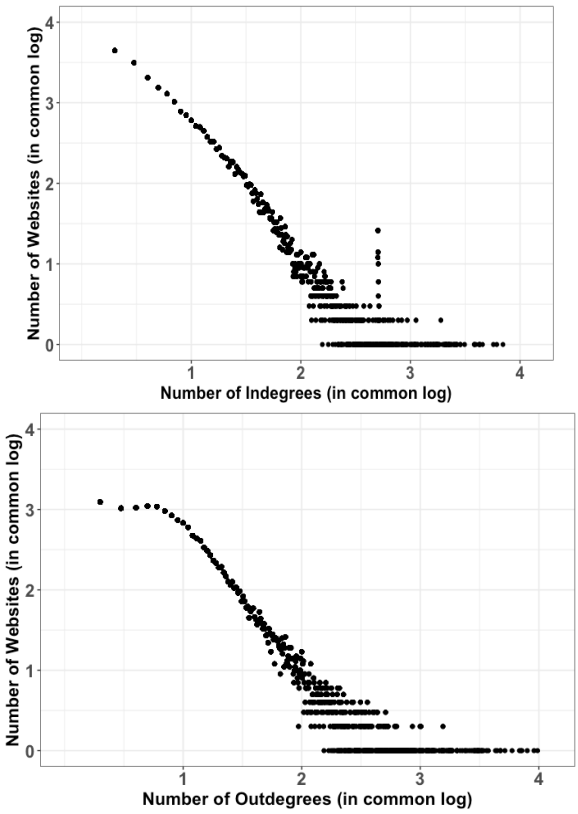

Figure 6: Indegree and outdegree distributions for 34,785 malicious websites in common log-log scale

3.7 Limitations

Common Crawl analyzes the content in the source code of the website's that is sent to the visitors. However, if hyperlinks are created dynamically in client side and not placed in source code, Common Crawl is not able to crawl them. Dynamic content can be implemented by using technologies like client side Javascript and AJAX calls. Also, links can be obfuscated in source files, but then be converted into normal links dynamically. In that scenario, again the crawler is not able to detect them correctly.

Sometimes websites publish robots.txt file usually in their top-level directory. It is a mechanism to let crawlers know that this site does not want to be crawled and indexed all of its content or some portion of it. Actually, this mechanism does not enforce a ban on crawlers. Therefore, it is just a polite request and is at the mercy of the crawlers. Common crawl states that they obey the robots.txt file limitations [46].

However, this mechanism can be abused by malicious websites in order to cloak their activities and prevent getting analyzed.

3.8 Summary

This chapter gave a detailed explanation of how data collection was held. The steps we take to construct the network are also mentioned. Degree distributions for our malicious websites are given. Lastly, the limitations regarding data collection processes were presented.

CHAPTER 4

FINDING MALICIOUS WEBSITES

This section explains the analyses carried out to find malicious websites based on their link structures. Malicious actors want people to reach their websites in order to infect them or to phish them. Therefore, they need to provide links that lead users to their websites from different sources like other websites. Malicious actors are using links since they are the medium of communication and flow on WWW.

Malicious website detection techniques are numerous. We are not proposing a new detection technique but an accompanying one that shows where should we look at to find more malicious websites. Malicious website detection is an expensive process. There are millions of websites on the WWW, even the Common Crawl dataset we use consists of 90,757,643 distinct websites and thousands are created every day. Due to the enormous number of websites a prioritization is necessary since our analysis resources are not unlimited. In this chapter we investigate which websites should be analyzed primarily to run into more malicious websites. This will let us do analysis more efficiently. By efficient we mean analyzing a smaller number of websites but running into relatively more malicious websites in them.

4.1. Core Groups

To carry out the analyses, network created in Chapter 3 is used. This network includes three types of malicious websites: malware, phishing and uncategorized. However, for the analyses in this chapter we will not make separate evaluation for each malicious type, we will group them under malicious category.

The network's total size is 776,230. It has 34,785 malicious websites and 741,445 other websites which are malicious websites' incoming and outgoing link neighbors. We do not know maliciousness status of these 741,445 websites beforehand since they are not included in our malicious websites lists. They will be analyzed from Virustotal and we define a website as malicious if it is flagged malicious by at least one product from Virustotal for the analyses in this chapter.

Let's assume that we want to detect more malicious websites and the only thing we have is 34,785 malicious websites and their incoming and outgoing links. Our aim is to find which kinds of websites we should focus our analysis on to run into more malicious websites.

Core groups in a network are special subgroups that are essential to that network or they are the ones that carry important features in a network. These features would be

based on the group members' centrality, degrees or special roles that they take in the network. In the network we will define some core groups that we consider important for malicious website detection. Then we will analyze them and show whether we could use them for an efficient analysis.

There might be many different types of core groups in a network and the definition of a core group changes according to the type of the network. For example, in a friendship network, people who are close to many members of the group might form a core group. For networks that consist of malicious websites and their links we could define the core groups as the ones that are part of the malicious activity more actively or the ones whose absence would affect the flow relatively more than the others.

We will use cohesive subgroups and betweenness centrality measures for our core group analysis. The reason for their selections as core groups are as follows:

1. Cohesive subgroups show dense pockets of nodes that stick together. They have relatively strong relationship between themselves than from the rest of the network. In the case of malicious websites, cohesive subgroups may indicate websites that are operated by the same malicious actors since cohesiveness occurs due to the connections in the network. Also, since the connectedness is formed via links, concentration of these links in some parts of the network could indicate the same attacker behind them. In order to find cohesive subgroups in a network, we can use components and k-cores.

2. There are several centrality measures defined in the networks such as degree, betweenness and closeness centralities. Each centrality measure takes a different node characteristic that let them to be ranked in order of importance. The reason we chose betweenness centrality among other types of centrality measures resides in its ability to show a node's influence over the flow of information in the network. Flow of information in this context is spreading of malicious content. Malicious actors use WWW as a distribution medium for their activities. In order to establish a good distribution mechanism, they need to employ links between nodes that takes a user to malicious content. By using betweenness centrality we measure a node's importance in this distribution channel.

4.1.1 Components

Components are the connected parts of the network. There are two types of components in a network: strongly connected and weak components. Strongly connected components are a subgroup where we can reach any other node from a node if we obey the direction of arcs. Similarly, in weak components we can reach any other node from a node if we disregard the direction of arcs.

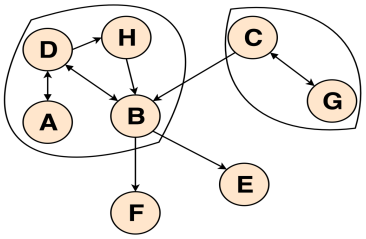

Figure 7: A sample network that shows strongly connected components inside contours

In Figure 7 above, there are two strongly connected components of size greater than one. One consists of nodes A, B, D, H and the other consists of nodes C and G. Inside these components we could reach any other node from any node following the directions. Here the largest strongly connected component is the one composed of nodes A, B, D and H since it has more members than the other component. The network in Figure 7 also has many weak components. The network as a whole for example is a weak component since we can reach any other node from a node without needing to obey the direction of arcs.

Strongly connected components are more strict than weakly connected components therefore they are usually smaller than the weak components. In order to limit the number of nodes we select and to establish a stricter criterion we will use the strongly connected components. In order to find strongly connected components, we will focus only on malicious nodes and the links between them since we are interested in the cohesive subgroups of malicious nodes. Therefore, we removed non-malicious nodes and their adjacent arcs from the network and found the largest strongly connected component between malicious websites, shown in Figure 8. These malicious websites have a relationship between themselves made by hyperlinks.

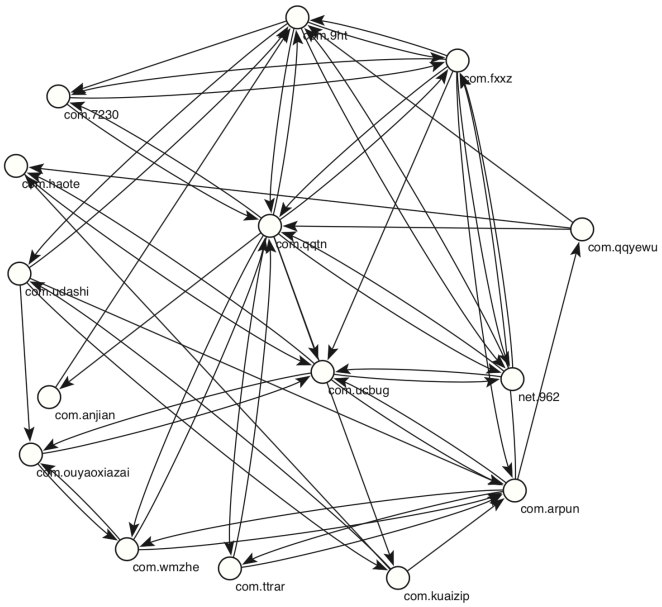

Figure 8: Largest strongly connected component with 15 members of malicious websites from the network

4.1.2 K-cores

The second type of cohesive subgroup we have is k-cores. A k-core is a maximal subnetwork where each node has at least degree k within the subnetwork. Therefore, nodes in a k-core have at least k neighbors within the group. For example, a 3-core subnetwork consists of all nodes that are connected to at least three other nodes in the subnetwork. To find the k-cores we symmetrized the network between malicious websites, i.e. transformed arcs into edges, since the application of k-cores is more suitable in the symmetrized networks.

In order to find k-cores, we will focus only on malicious nodes and the links between them since we are interested in the cohesive subgroups of malicious nodes. Therefore, we removed non-malicious nodes and their adjacent edges from the network and found three cohesive subgroups of malicious websites, 8 nodes in a 4-core and 142 nodes in a 3-core as shown in Figures 9 and 10. Beware that k-cores are nested so nodes that belong to a higher core also belong to a lower core, therefore 3-core also includes 8 members of 4-core. We dismissed the other cores below 3-core since they may not be cohesive enough.

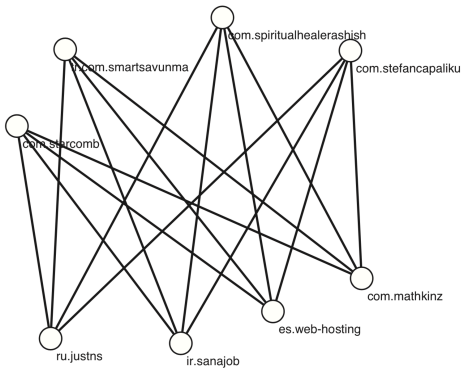

Figure 9: 8 malicious nodes in a 4-core and their network between themselves

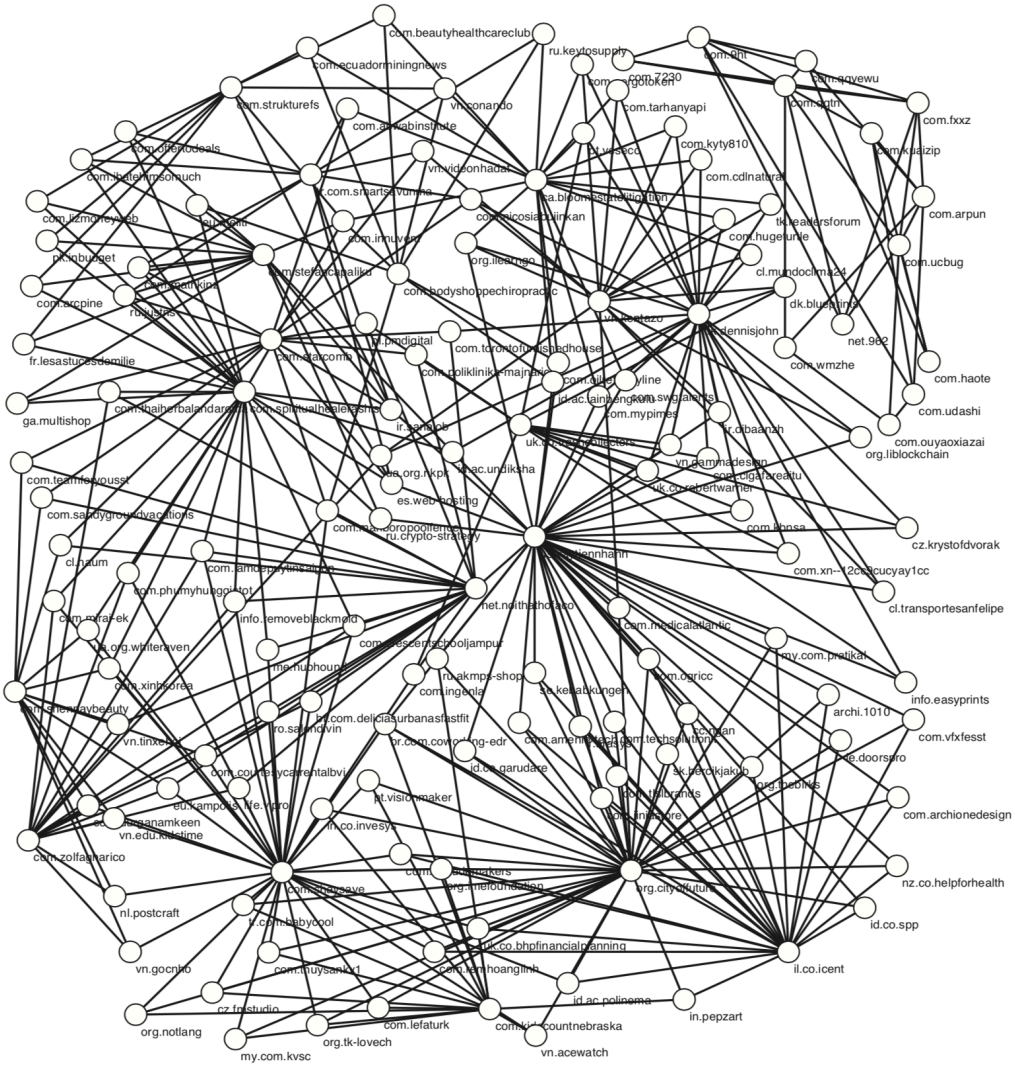

Figure 10: 142 malicious nodes in a 3-core and their network between themselves

4.1.3 Betweenness centrality

Betweenness centrality shows a node's influence over the flow of information in the network. We can say that a node is more central if it is needed to connect other nodes and whose absence would result in the total disruption of communication or would require the other nodes to take longer paths. The betweenness centrality for a node is the proportion of all shortest paths between pairs of other nodes that include this node.

Malicious nodes with higher betweenness centrality are important since they connect other malicious nodes together. Similar to components and k-cores, we removed non-malicious nodes and their adjacent edges, then we symmetrized the network between malicious websites and calculated betweenness centrality scores of malicious nodes in our network. We then selected the first 100 and 500 nodes with the largest betweenness centrality scores.

4.2. Results

After determining three different core groups in our network, we look at their link neighbors. We located their incoming and outgoing link neighbors which are not defined as malicious in our dataset before. Table 4 shows the three types of core groups in our network. We presented the size of the core group, total incoming link neighbors to the members of the core group and total outgoing link neighbors from the members of the core group. For example, in the Table 4 below, the largest strongly connected component (SCC) has 15 members. These 15 members have 8696 total incoming link neighbors where 1213 of them were found to be malicious which equates to 13.9%. Note that these link neighbors and malicious ones found in them are not included in the original malicious websites we used. They were found malicious by inspecting each of them on Virustotal.

Table 4: Core groups and analysis results of their neighbors

Core Group Selection	**SCC**	**K-Cores**		**Betweenness Centrality**	
	Largest #1	**4-Core**	**3-Core**	**TOP 100**	**TOP 500**
Core Group Node Size:	15	8	142	100	500
Malicious Incoming Link Neighbors:	1213	58	1543	5006	9370
Total Incoming Link Neighbors:	8696	431	11259	17683	51577
Malicious Percentage	**13.9%**	**13.4%**	**13.7%**	**28.3%**	**18.2%**
Malicious Outgoing Link Neighbors:	724	542	4597	4904	10362
Total Outgoing Link Neighbors	2241	2705	18557	29403	75612
Malicious Percentage	**32.3%**	**20.0%**	**24.7%**	**16.7%**	**13.7%**

Largest strongly connected component's (SCC) and k-cores' outgoing link neighbors set are discovered to include more malicious websites than their incoming link neighbors set in terms of percentage. On the other hand, for betweenness centrality core group, their incoming link neighbors provide higher percentages. The largest malicious percentage comes from the largest SC's outgoing neighbors. 15 members of the largest strongly connected component has 2241 outgoing links and 724 of them are found to be malicious after analyzing them on Virustotal.

In order to evaluate the results we gathered from core groups, we calculated two sets of websites for comparison. For the first one, we investigated what happens if we blindly select random websites from the web. Out of 90,757,643 websites present in Common Crawl dataset, we randomly selected 40,000 websites. When we inspected them on Virustotal 1,308 of them were flagged as malicious which equates to 3.27%. The reasons we selected 40,000 websites but not more and also not reiterating the

analysis many times are due to the limitations of Virustotal API which lets you to call it once in 15 seconds for each website you want to analyze. Also as it can be seen from Table 4 above, the minimum number of websites we analyzed from the core groups is 431 in 4-core incoming links and maximum number is 75,612 in betweenness centrality Top-500 therefore 40,000 is an intermediate value between them.

Since we formed a network using malicious websites as seeds, for the second comparison, we investigated what happens if we select random nodes from the network instead of blindly choosing from the Web. For this reason, we randomly selected 400, 1000, 2500, 5000, 7500, 10000, 15000, 20000, 25000, 30000, 40000, 50000, 75000, 80000 websites from the set of 741,445 nodes (34,785 malicious nodes were not included) in our network. We chose the numbers around the range of 431 to 75,612 to also compare this method with selections made from core group's link neighbors. As it can be seen from Table 4 above, these numbers are close to the sizes of core groups' incoming and outgoing link neighbors. Since we are selecting nodes randomly, we reiterated the processes 10 times for an impartial analysis and results are given in Table 5.

Table 5: Malicious websites in randomly selected websites from the network averaged for 10 experiments

		Number of Analyzed Websites														
Number of Websites Found Malicious		**400**	**500**	**1000**	**2500**	**5000**	**7500**	**10000**	**15000**	**20000**	**25000**	**30000**	**40000**	**50000**	**75000**	**80000**
	Min	37	38	99	244	525	805	1078	1592	2131	2703	3231	4212	5350	8056	8623
	Max	62	67	131	304	576	862	1126	1704	2258	2865	3355	4476	5530	8307	8868
	Average	47	53	112	264	549	826	1106	1628	2191	2766	3291	4356	5469	8213	8751
	Percentage	11.8	10.6	11.2	10.6	11.0	11.0	11.0	10.9	11.0	11.1	11.0	10.9	10.9	11.0	10.9

Figure 11 below shows the ratio between the number of analyzed websites and the number of malicious ones detected in them for three types of selection. First one is 40.000 website selected randomly out of 90,757,643 websites and shown as Random40000. The second one shows the results of randomly selecting websites from the network and shown as the line whose values are from Table 5 above. As we iterated the analysis 10 times you can see the error bars around the line. The last one is selections based on core groups and each of them are displayed with a different shape as shown on the legend.

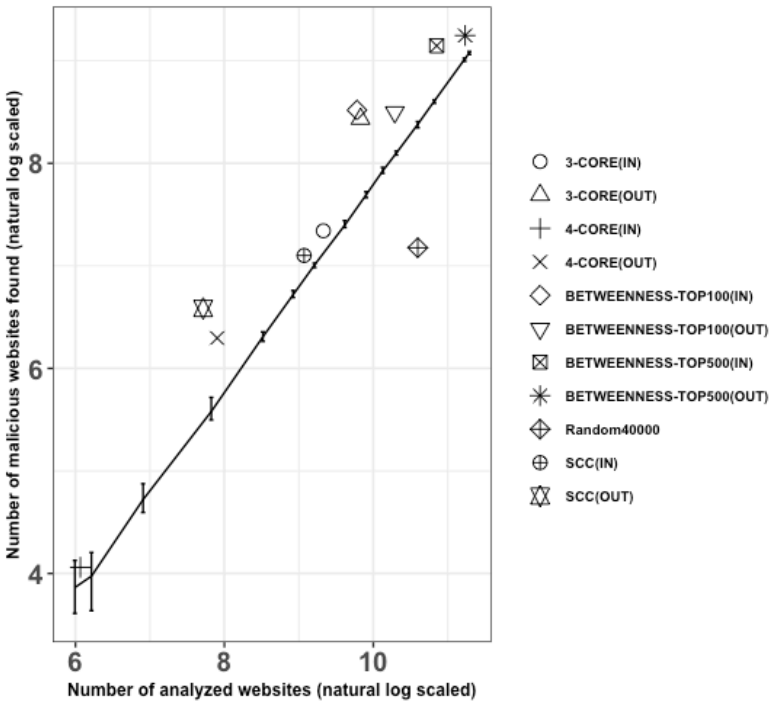

Figure 11: Number of malicious websites in number of analyzed websites for different core groups and random selections

As can be seen from the Figure 11, selection of random websites from WWW has the worst performance in terms of running into malicious websites. As shown by the line, selection of random nodes from the network, which is formed by using malicious websites as seeds and adding their incoming and outgoing neighbors with a link has better performance. This means that the chance to run into a new malicious website by inspecting link neighbors of already known malicious websites are higher than randomly choosing websites from the Web. Here we presented how we can even increase this ratio by inspecting link neighbors of core groups in the network. Selections made from core groups outperforms both of these first two methods. In the core groups, outgoing links from the largest strongly connected component and incoming links to Top-100 betweenness centrality nodes provided the best results.

First one let us find 32.3% and the second one let us find 28.3% of their links neighbors as malicious.

These findings show that analyzing the link neighbors of core groups in a network would let us find greater number of malicious websites more quickly. Since there are millions of websites and we do not have unlimited resources, we need to make a priority between the websites to be analyzed. This will let us save computational costs and find more malicious websites in a shorter amount of time.

The findings in this chapter could be used by security community as well as by academic researchers. Many studies in the web security domain require gathering a list of malicious websites. This is usually done by collecting them from published security blacklists as we did in the beginning of this study. But if one needs to find more malicious websites for a quick sampling, searching them in the neighborhood of core groups in a network like we created would provide faster results.

4.3. Comparision With Other Works

The results of our study and the other studies mentioned in the related works which try to efficiently find malicious websites are presented here. Each study applies different technique with different datasets. Therefore, it would not be reasonable to compare them exactly in terms of their performance, however presenting the results of similar studies would be beneficial to have a general grasp on the different types of methods on the topic.

Malcrawler [11] employed a crawler-based approach to find malicious web pages. Before they followed any link, they visited the web page first and extracted 10 different features. Only if the web page looked like malicious according to these features then they followed this link. They made two crawls to test their method, one with a generic crawler where every link is followed and another with their proposed method. In both of these crawls they visited around 0.57 million URLs. In the first crawl they encountered 702 (0.123%) URLs and in the second crawl they encountered 1978 (0.348%) URLs. By applying their method, they were able to increase the number of malicious URLs they encountered from 702 to 1978.

Evilseed [12] extracted the characterizing similarities between known malicious web pages and using them, they tried to find other malicious pages that are similar or related to known malicious ones. The idea is that a feature shared by many known malicious web pages is an indication of malicious activity and other pages that have this feature are more probably to be malicious. They compared their method against a random web search based and a crawler-based approach. Evilseed gathered 226,140 URLs of which 3036 (1.34%) were found to be malicious whereas the crawler gathered 431,428 URLs of which 604 (0.14%) were found to be malicious and random web search made on a search engine had 219 (0.34%) malicious URLs out of 64,411 URLs.

We compared our proposed method against a random selection of websites from the Web and random selection of websites from our created network. One of the core groups we defined let us find 724 (32.3%) malicious websites out of analyzed 2241 websites whereas selecting 40,000 websites randomly from the Web had 1308 (3.27%) malicious websites and selecting 40.000 websites randomly from our network had 4356 (10.9%) malicious websites.

We used Virustotal to check whether a website is malicious or not but Malcrawler and Evilseed used Google Safe Browsing and some custom tools. The gap between the detection numbers and percentages of our study and theirs are mainly due to this reason.

Both Malcrawler and Evilseed include a step of feature extraction. Malcrawler's method requires this for every link that it comes across whereas Evilseed requires it for its initial malicious set only. We should note that feature extraction brings computational costs and our proposed method works without any feature extraction which will let us save computational costs. Also, Evilseed uses search engine queries to find malicious web pages but search engines have a tendency to remove malicious web pages from their results to protect their users which may affect Evilseed's results profoundly.

CHAPTER 5

THREAT TYPE IDENTIFICATION

In this chapter we try to predict a malicious website's threat type for two classes, malware and phishing. As explained before, analyzing malicious websites is an expensive process. If we could predict threat type of a website, we could tailor our analysis accordingly. If it is predicted to be phishing, phishing detection techniques and if it is predicted to be malware, malware detection techniques could be used primarily which will let us save computational costs. For threat type prediction we present two methods. First one is malicious link neighbors method where we analyze the neighbors of a node to predict its threat type. The second method is link similarity method where we predict the threat type of a website based on its link similarity with other nodes.

For analyses purposes a network is created similar to the one in Chapter 3 using randomly chosen 10,000 malware and 10,000 phishing websites. Formation of this network follows the technique mentioned in Chapter 3.5. There are two reasons why we did not use the network created in Chapter 3. First reason is that the previous network includes uncategorized nodes. Since we want to make predictions between malware and phishing websites they would not be useful for our purposes. The second reason is that there were more malware nodes than the phishing nodes. For a better analysis environment here we randomly selected 10,000 malware nodes from 14,543 malware nodes and 10,000 phishing nodes from 10,455 phishing nodes. Frequencies and the size of the created network are given below.

Table 6: Frequencies and percentages of the threat types of websites in our network

Type	Frequency	Percentage
Malware	10,000	1.87%
Phishing	10,000	1.87%
Others	514,443	96.26%
Total	534,443	100%

5.1. Evaluation Metrics

In order to test the effectiveness of methods, we will use some evaluation metrics.

Coverage: Whereas other metrics provided below are generally known and have agreed upon definitions we need to use another metric in our analysis named coverage. Because of the link-based approach we take, not all measures are applicable to classify every node. For example, incoming link neighbors can only be applied to those nodes who have at least one malware or phishing incoming neighbor. Therefore, coverage of a measure is the number of nodes we could predict a type using this measure, whether it's true or not, divided by all nodes. For example, if we could make predictions for 3000 websites out of 4000 websites we analyzed, coverage is 75%.

$$Coverage\ of\ a\ Measure = \frac{\text{Number of nodes we could predict a type}}{\text{Total number of unknown malicious nodes}} \quad \text{(Eq. 1)}$$

A confusion matrix is provided in Table 7 that shows actual and predicted values for our threat type assignments. Threat type prediction we employ is actually a two-class problem and for confusion matrix we are using the two malicious types, malware and phishing, instead of generally used concept of positive and negative. The sum of the values, a+b+c+d, is equal to the number of predictions we make. Therefore, if we make 3000 predictions for 4000 websites that we analyze a+b+c+d is equal to 3000 and to show the discrepancy we give the coverage in the result tables.

Table 7: Confusion Matrix for Label Classification

		Predicted	
		Malware	Phishing
Actual	Malware	a	b
	Phishing	c	d

Accuracy: Accuracy shows the general correctness of the model. It is calculated as the sum of the correct classifications divided by the total number of classifications. Accuracy may not be a good metric for datasets where sample sets have different sizes, but since we have the same size for both classes, accuracy will be provided [62][63].

$$Accuracy = \frac{\text{Number of correct predictions}}{\text{Total number of predictions}} = \frac{\text{a+d}}{\text{a+b+c+d}} \quad \text{(Eq. 2)}$$

Precision: Precision is a measure of accuracy provided that a specific class has been predicted [62][63].

Precision for malware class gives us what proportion of predicted malware classes are truly malware and is calculated as = $\frac{a}{a+c}$

Precision for phishing class gives us what proportion of predicted phishing classes are truly phishing and is calculated as = $\frac{d}{b+d}$

Recall: Recall is a measure of the ability of a prediction model to select instances of a certain class from the dataset [62][63].

Recall for malware class gives us what proportion of actual malware classes are predicted as malware and is calculated as = $\frac{a}{a+b}$

Recall for phishing class gives us what proportion of actual phishing classes are predicted as phishing and calculated as = $\frac{d}{c+d}$

For evaluation purposes, we randomly selected 20% of malware nodes and 20% of phishing nodes in the network and made their types unknown. Rest of the 80% of both classes kept their types. For each different measure, we reiterated the calculations 10 times with different randomly chosen nodes made unknown each time. Then we calculated the averages of these 10 results by arithmetic mean and presented evaluation metrics: average accuracy, average recall, average precision and average coverage. As experiments were run for 10 times with different random data, standard deviations for each evaluation metric were provided in the tables. Our general aim was to improve accuracy and coverage at the same time as much as it is possible.

5.2. Malicious Link Neighbors Method

In malicious link neighbors method, we analyze whether links between websites could help us to classify them in terms of their threat types. A website in the network has two kinds of neighbors, direct and indirect. For a given website X, its direct neighbors are those websites that either link to X or linked by X or both. Its indirect neighbors are those that do not have a direct link relationship with website X, but are related via a third website.

In malicious link neighbors method, we build our model upon the assumption that malicious websites tend to link other malicious websites in the same class, i.e. malware websites link other malware websites, phishing websites link other phishing websites. First, we need to verify this assumption preemptively in our network, and then show how can we utilize this for attack type identification.

We define two types of neighbors as direct and indirect for a node. Afterwards we will use them in our experiments and show each of their results.

5.2.1. Direct Neighbors

Direct neighbors of a node X are the nodes that either link to X or linked by X or both. Therefore, direct neighbors are at length=1 distance away from the node X in the graph.

Let V be a set of nodes $V = \{X_1, X_2, \ldots X_n\}$ in the graph and $E = \{L_{1\to2}, L_{1\to3}, \ldots\}$ be a set of edges between nodes where $L_{i\to j}$ indicates a link from node X_i to X_j in the directed graph G =(V,E) [64]

If $S_{in}(X_i)$ is the set of incoming neighbors of node X_i :

$$S_{in}(X_i) = \{X_j \mid L_{j\to i} \in E\} \quad \text{(Eq. 3)}$$

If $S_{out}(X_i)$ is the set of outgoing neighbors of node X_i :

$$S_{out}(X_i) = \{X_j \mid L_{i\to j} \in E\} \quad \text{(Eq. 4)}$$

If $S_{in+out}(X_i)$ is the set that holds the union of incoming and outgoing neighbors of node X_i :

$$S_{in+out}(X_i) = \{X_j \mid L_{i\to j} \in E \text{ or } L_{j\to i} \in E\} \quad \text{(Eq. 5)}$$

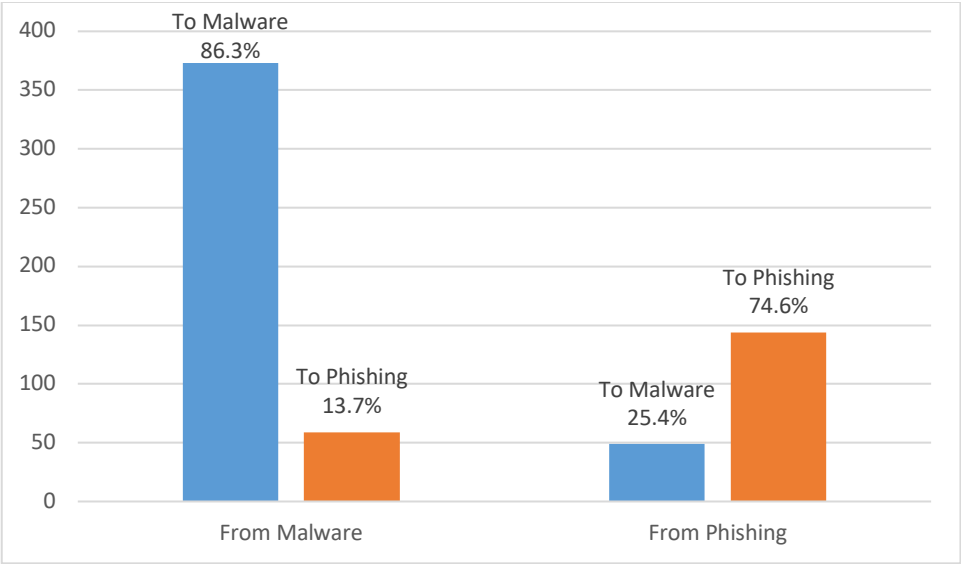

Figure 12: Links between malicious nodes in our network

In order to verify the assumption that malicious websites tend to link other malicious websites based on their threat types, we analyzed the links that occur between malicious nodes in the network we created. In Figure 12 horizontal line at the bottom shows which type of nodes the links originate from and vertical bars show where these links end in. 86.3% of all malicious links from malware nodes end in other malware nodes and 74.6% of all malicious links from phishing nodes end in other phishing nodes. These results indicate that malicious websites tend to link other malicious websites in the same class, but it also reveals something concerning about the direct neighbors. Out of 20.000 websites, i.e. the size of malware and phishing websites in our network, only 625 distinct nodes have direct link relationship with other malicious websites. Therefore, analyzing the direct neighbors would let us predict a threat type for very low number of nodes. In order to increase the number, we need another kind of neighborship.

5.2.2. Indirect Neighbors

In contrast to the direct neighbors, indirect neighbors of node X do not have any direct link relationship with it. Indirect neighborship occurs via a third node. Therefore, indirect neighbors are at length=2 distance away from node X. As shown in Figure 13, if nodes A and B are linked by the same node, they are co-cited and if they both link to a third node they are bibliographically coupled.

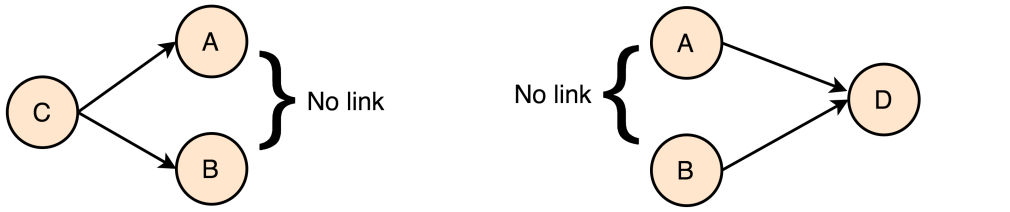

Figure 13: Co-citation and bibliographic coupling relationship

Co-citation and bibliographic coupling use citation analysis to establish a likeness between nodes. Co-citation based [65] and bibliographic coupling-based [66] techniques originated from bibliographic research community to find related scientific papers in 1960s and were shown to be useful. Although they have applications mainly in bibliographic studies they were used for many other areas where link relationships occur such as web graphs. Two different studies [15] and [67] successfully applied co-citation to find thematically similar pages and hosts on the Web graph.

For a directed graph G = (V,E) let V be a set of nodes V = $\{X_1, X_2, \ldots X_n\}$ and E = $\{L_{1\to2}, L_{1\to3}, \ldots\}$ be a set of edges between nodes where $L_{i\to j}$ indicates a link from node X_i to X_j [64].

If $S_{cocit}(X_i)$ is the set of co-cited neighbors of node X_i :

$S_{cocit}(X_i) = \{X_j, \mid L_{k\rightarrow i} \in E, L_{k\rightarrow j} \in E$ and $X_i \neq X_j$ and where X_k is a third node that links to both X_i and $X_j\}$ (Eq. 6)

If $S_{bibco}(X_i)$ is the set of bibliographic coupling neighbors of node X_i :

$S_{bibco}(X_i) = \{X_j, \mid L_{i\rightarrow k} \in E, L_{j\rightarrow k} \in E$ and $X_i \neq X_j$ and where X_k is a third node that both X_i and X_j links} (Eq. 7)

5.2.3. Usage of Direct and Indirect Neighbors for Threat Type Detection

Proposed threat type detection algorithm exploits the threat types of neighbors. Before we begin the calculations, we make 20% of malware and 20% of phishing nodes unknown for evaluation purposes. The rest of the 80% of each class keep their types as explained in Chapter 5.1. Set $U = \{u_1, u_2 \ldots u_N\}$ holds the N nodes whose types are unknown and set $K = \{k_1, k_{2,\ldots} k_M\}$ holds the M nodes whose types are known.

We assign a maliciousness probability scores $R_x(m)$ and $R_x(p)$ for each node X in the graph. $R_x(m)$ is the probability of node X to be malware and $R_x(p)$ is to be phishing. In the initial assignments step, we assign maliciousness probability scores $R_x(m)$ and $R_x(p)$ for each node X in the graph based on their threat types as given in Table 8 below. Since we know their threat types a priori, for malware nodes $R_x(m)=1$ and $R_x(p)=0$ and for phishing nodes $R_x(m)=0$ and $R_x(p)=1$. Since for type assignment purposes we are only interested in malicious nodes, $R_x(m)$ and $R_x(p)$ probabilities of other nodes in the graph are set to 0. We also set $R_x(m)=0$ and $R_x(p)=0$ to type unknown nodes in the beginning but these values are expected to change during the following steps.

Table 8: Types of nodes and their initial maliciousness probabilities

Node Type	R
Known malware	$R_x(m) = 1, R_x(p) = 0$
Known phishing	$R_x(m) = 0, R_x(p) = 1$
Unknown type	$R_x(m) = 0, R_x(p) = 0$
Others	$R_x(m) = 0, R_x(p) = 0$

$R_x(m)$ and $R_x(p)$ scores for unknown type nodes are calculated as follows:

$$R_x(m) = \frac{\text{Sum of malware probabilities of its neighbors}}{Total\ number\ of\ malicious\ and\ unknown\ neighbors} \quad \text{(Eq. 8)}$$

$$R_x(p) = \frac{\text{Sum of phishing probabilities of its neighbors}}{Total\ number\ of\ malicious\ and\ unknown\ neighbors} \quad \text{(Eq. 9)}$$

The neighbors mentioned in Equations 8 and 9 could be a kind of direct neighbor or indirect neighbor depending on which set we decided to use.

For the rest of this method, firstly the simple explanation of the proposed algorithm is given. Then we mention the steps taken in detail and lastly the pseudocode is presented.

The proposed algorithm simply tries to calculate above mentioned maliciousness probability scores $R_x(m)$ and $R_x(p)$ for unknown type nodes in the network using different set of neighbors. First it calculates these values for unknown nodes in random order. But since the order affects the results, we get into an iterative process that make nodes trigger other nodes to recalculate their maliciousness probabilities when theirs change. After the algorithm converges or we reach the maximum number of iterations for each node, we finish the iterative process and we make predictions for nodes based on their latest $R_x(m)$ and $R_x(p)$ values. We assign the threat type between malware and phishing based on whichever's probability is higher.

The detailed explanation of the algorithm and steps we take to make algorithm converge and limit the number of iterations in reasonable amounts are mentioned below.

After we assign maliciousness probability scores for already known malware and phishing nodes, we begin to calculate $R_x(m)$ and $R_x(p)$ scores for every unknown node in our graph. This process can be modeled similar to a graph labeling problem and two examples are shown in Figures 14 and 15 below.

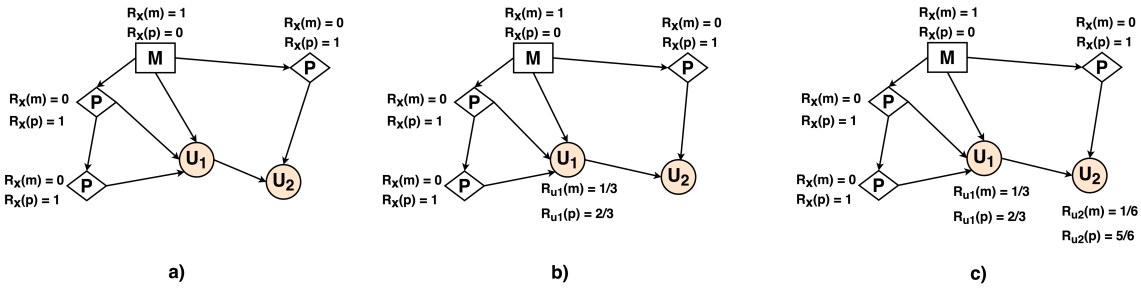

Figure 14: Type assignment simulated as graph labeling with steps a to c and beginning with node U_1

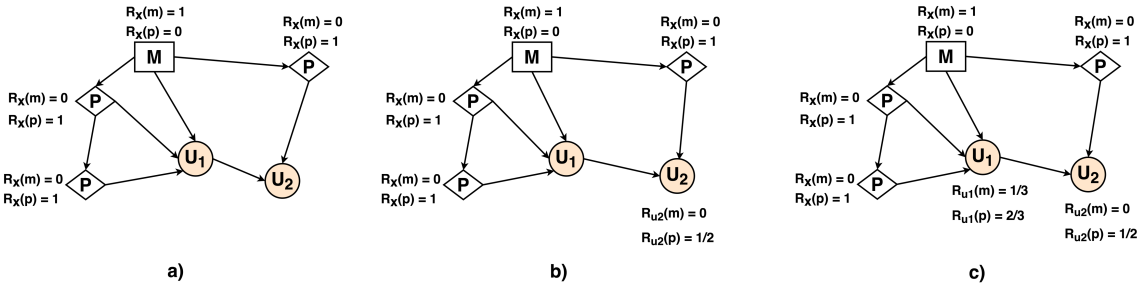

Figure 15: Type assignment simulated as graph labeling with steps a to c and beginning with node U_2

In Figures 14 and 15, rectangular objects show malware, diamond objects show phishing and circle objects show unknown class nodes. Beginning from steps a to c, $R_x(m)$ and $R_x(p)$ scores for unknown nodes are calculated as given in Equations 8 and 9 before. Here we use the incoming link neighbors measure as an example. Therefore, we calculate maliciousness probability scores based on the node's incoming neighbors. Probabilities can be calculated based on other direct and indirect neighbors as well. In Figure 14 we started the calculations with node U_1. Since U_1 has three incoming link neighbors it gets $R_{u1}(m) = (1+0+0) / 3 = 1/3$ and $R_{u1}(p) = (0+1+1) / 3 = 2/3$ in step b. In step c we do calculations for U_2 which has two incoming link neighbors and it gets $R_{u2}(m) = (1/3+0) / 2 = 1/6$ and $R_{u2}(p) = (2/3+1) / 2 = 5/6$.

In Figure 15 which is the same network as in Figure 14, we started the application with node U_2, did the similar calculations but get a different probability for it than in Figure 14. This shows that the order we start calculating the maliciousness probabilities of unknown nodes actually affects the results. There are some options to tackle this problem. First one is changing the ordering of the nodes that are being processed with some metric and reporting results accordingly for different orderings. The second one would be iteratively calculating maliciousness probabilities for nodes. We adopted the second option and implemented an iterative process to the algorithm.

In the iterative process, we let changes in one node's maliciousness probabilities trigger other nodes to recalculate their maliciousness probabilities. This only occurs when changed node is in a neighborhood of other nodes for the chosen neighbor set.

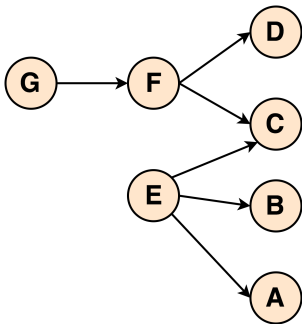

Figure 16: An example network

If we assume to use incoming neighbors set in Figure 16 above, when nodes A, B, C and D's maliciousness probabilities are changed no other node will be triggered because they are not any other nodes' incoming neighbor. Therefore, they do not contribute to any other nodes' probability calculations. However, when node G's probabilities are changed node F will be triggered, since node G contributes to node F's probabilities as an incoming neighbor. When node F's probability changes it triggers nodes C and D. But their change will not affect any other node as explained before. Also if a change in node E's probabilities occur, it triggers nodes A, B and C, but they also will not trigger any other nodes.

If we assume to use co-citation neighbors measure in Figure 16, when node E, F or G's maliciousness probabilities are changed, no other node is triggered since they are not co-citation neighbor of any other node. However, when node A's maliciousness probabilities are changed, this will trigger nodes B and C to recalculate their maliciousness probabilities, since they are co-citation neighbors of node A and node A contributes to the calculations of node B and C's maliciousness probabilities. Then when node B's probabilities are changed, this triggers node A and C to recalculate their probabilities etc. These kinds of situations may end in endless loops if certain precautions are not taken. So we will mention them shortly.

As seen above, when a change occurs in one node's probabilities this may require changes in lots of other nodes. In order to keep the nodes we are going to process, we defined a last-in-first-out (LIFO) stack that keeps unknown nodes that are waiting for their maliciousness probabilities to be recalculated. When a node's probabilities are changed, we push the other nodes it triggered to the stack, if there is no affected node nothing is pushed.

When we finish the calculations on a node and find the other nodes it triggered and pushed them to the stack, our job is done for that node for the time being. Then we pop the top node from the stack and do the same operations on it continuously until there is no node left to be analyzed in the stack. When the stack is empty, we check our set U if there are any remaining unanalyzed unknown node left. If yes, we push it to the stack and if no, we stop the iterative step and continue to the type prediction. In this step we make predictions for nodes based on their latest $R_x(m)$ and $R_x(p)$ values. We assign the threat type between malware and phishing based on whichever's probability is higher. In the case of evenness we let the type remain unknown since we have a metric called coverage that shows the rate of a measure's applicability to classify nodes for these kinds of incidents.

We had to take some measures for the algorithm to converge and limit the number of iterations in reasonable amounts.

1) We defined a metric called confidence for each node which is the number of unknown link neighbors that needs to be recalculated if this node's probabilities are changed. The lesser the value the more confidence a node has. When we need to select a node from a set of nodes, we select the ones with the largest confidence. By doing this we make the algorithm to reach higher number of analyzed unknown nodes faster by not spending time in iterations.

2) We defined a scalar delta (Δ) value. If absolute value of the difference between malware and phishing probabilities of a node is greater than the delta, $|R_x(m) - R_x(p)| > \Delta$, we say that these ratios could be trusted to predict a type. Whichever is higher among $R_x(m)$ and $R_x(p)$ is set to 1 and the other is set to 0. We also make these nodes' status fixed, and their probabilities will not be recalculated and changed in iterations anymore. This will limit the number of operations we need to make. We do this calculation before we begin iterative process and in every iteration afterwards. $\Delta = 0.15, 0.30, 0.40, 0.50$ and 0.70 are tested.

3) In the iterative process, changes in one node's maliciousness probabilities may trigger others to recalculate their maliciousness probabilities. Especially for indirect neighbors this may result in endless loops. If node A is co-citation neighbor of node B, the other way around is also true. Therefore, for co-citation neighborhood when node A's maliciousness probabilities change it triggers node B to recalculate its maliciousness probabilities which triggers node A in return and it continues like that. To prevent endless loops, we let one specific node to trigger other specific node only one time.

4) To limit the total number of iterations in reasonable amounts there are some options. One is employing a global iteration count and when it is reached stopping the algorithm. Here we employed a total number of 30 changes per node. Afterwards, that node's status is fixed and its probabilities will not be recalculated and changed in iterations afterwards.

The pseudocode of the algorithm is as follows:

Algorithm 1 Iterative threat type predictor

Given:
$K = \{k_1, k_2, \ldots k_N\}$ //Set of known type nodes
$U = \{u_1, u_2, \ldots u_M\}$ //Set of unknown type nodes
$V = K \cup U$ //Set V is the union of sets K and U
$E = \{L_{1\to2}, L_{1\to3}, \ldots\}$ //Set of links in the network
G (V,E)
Value Delta (Δ)

Step1.1 Initial Assignments
For i=1 to N
    If ($T(k_i)$ = malware) //If node's type is malware
        $R_{ki}(m) = 1$, $R_{ki}(p) = 0$
    Else if ($T(k_i)$ = phishing)
        $R_{ki}(m) = 0$, $R_{ki}(p) = 1$
    Else if ($T(k_i)$ = others or $T(k_i)$ = unknown)
        $R_{ki}(m) = 0$, $R_{ki}(p) = 0$

For i=1 to M
    Calculate $R_{ui}(m)$ and $R_{ui}(p)$

Step1.2. Preparations
For i=1 to M
    If ($|R_{ui}(m) - R_{ui}(p)| > \Delta$ and $R_{ui}(m) > R_{ui}(p)$)
        $R_{ui}(m) = 1$ $R_{ui}(p) = 0$
        Make u_i fixed
    Else if ($|R_{ui}(m) - R_{ui}(p)| > \Delta$ and $R_{ui}(m) < R_{ui}(p)$)
        $R_{ui}(m) = 0$ $R_{ui}(p) = 1$
        Make u_i fixed
    Else
        Continue
Define stack S
Select the unknown node u_i from the set U having the largest confidence
Push u_i to the stack S

Step 2. Iterative Step
While(S is not empty)
{
x = Popped element from the S
Calculate $R_x(m)$ and $R_x(p)$
If ($R_x(m)$ and $R_x(p)$ are changed)
    $R_x(m)$, $R_x(p)$ = new $R_x(m)$, new $R_x(p)$ // Update $R_x(m)$ and $R_x(p)$
    LN = set of neighbors of x whose probabilities need to be recalculated due to changes in node x if rules allow
    Sort LN according to the confidence ratios ascending

```
        Push LN to the stack beginning with the node having the lowest confidence
If (S is empty)
        Select the not analyzed unknown node u having the largest confidence from
        the set U
        If (no unanalyzed node left in set U)
                Break
        Else
                Push u to the S
}
```

Step 3. Type Prediction

```
For i = 1 to M          // Assign types to nodes in Set U
        If (Rui(m) > Rui(p))
                T(ui) = malware
        Else if (Rui(m) < Rui(p))
                T(ui) = phishing
        Else
                T(ui) = unknown
```

5.3. Link Similarity Method

In this method we predict the threat type of a node by finding the similarity scores between the nodes based on their link neighbors. If node A and node B have the same link neighbors, we could say that they are somewhat related. Here the question is could we use this relatedness to predict their threat type?

We use the same network and analysis techniques created in the beginning of this chapter. We make 20% of malware and 20% of phishing nodes unknown and 80% of malware and 80% of phishing nodes keep their types. Let there be two sets of malicious nodes $U = \{u_1, u_2 \ldots u_M\}$ and $K = \{k_1, k_{2,\ldots} k_N\}$ where U holds the nodes whose types are unknown and K holds the nodes whose types are known and M and N are the sizes of their respective sets.

If we want to calculate the similarity between nodes u_i and k_j considering their link neighbors, we should count the number of separate and common link neighbors for both nodes. There are different kinds of proximity and similarity metrics used in the literature such as Euclidian distance, Minkowski distance, dot product and Jaccard index [68]. For the data we have which are link neighbors of websites, best choice would be using Jaccard index since it can calculate the degree of overlap between two sets. It compares members of sets to find which members are shared and which members are distinct [69]. The mathematical representation of Jaccard index where $sim(u_i,k_j)$ is the similarity score between nodes u_i and k_j could be written like in Equation 10 where LN shows the link neighbor set of u_i and k_j [69].

$$sim(u_i, k_j) = \frac{|\, LN_{ui} \cap LN_{kj} |}{|\, LN_{ui} \cup LN_{kj} \,|}\ , 0 \leq sim(u_i,k_j) \leq 1 \quad \text{(Eq. 10)}$$

Equation 10 calculates the size of the intersection divided by the size of the union of sets. The similarity values range from 0 to 1 and two sets with totally different members will have 0 similarity whereas two sets where they share all the members will have the similarity score of 1.

We calculate the similarity score for each node in set U with every node in set K using the Equation 10 and construct a size MxN matrix. The values at every intersection of rows and columns show the similarity score between the nodes in that row and column.

Table 9: Size MxN matrix showing similarity scores between nodes

	$\mathbf{k_1}$	$\mathbf{k_2}$	…	$\mathbf{k_N}$
$\mathbf{u_1}$	$sim(u_1,k_1)$	$sim(u_1,k_2)$		$sim(u_1,k_N)$
$\mathbf{u_2}$	$sim(u_2,k_1)$	$sim(u_2,k_2)$		$sim(u_2,k_N)$
⋮	⋮	⋮		⋮
$\mathbf{u_M}$	$sim(u_M,k_1)$	$sim(u_M,k_2)$	…	$sim(u_M,k_N)$

Similar to the application in Section 5.2, for threat type prediction we assign maliciousness probability scores $R_x(m)$ and $R_x(p)$ to each node X in the graph. $R_x(m)$ is the probability of node X to be malware and $R_x(p)$ is to be phishing. For each known malware nodes, we set their maliciousness probabilities $R_x(m) = 1$, $R_x(p) = 0$ and for known phishing nodes $R_x(m) = 0$, $R_x(p) = 1$ as shown in Table 10.

Table 10: Types of nodes and their initial assigned maliciousness probabilities

Node Type	R
Known malware	$R_x(m) = 1$, $R_x(p) = 0$
Known phishing	$R_x(m) = 0$, $R_x(p) = 1$

Then for each node in set U we calculate its $R_{ui}(m)$ and $R_{ui}(p)$ probabilities with Equations 11 and 12 given below. These equations takes a weighted approach for calculation of R values. With this weighted method we take the similarity score between nodes u_i and k_j into effect when calculating the node u_i's maliciousness probabilities. Node k_j's contribution to the total maliciousness probability of node u_i's is directly proportional to its similarity to node u_i. If the similarity is higher, its effect will be higher. Likewise if similarity score is zero ,meaning they do not share any number of links, k_j will not affect u_i at all.

$$R_{ui}(m) = \sum_{j=1}^{N} sim(u_i, k_j) R_{k_j}(m) \quad \text{(Eq. 11)}$$

$$R_{ui}(p) = \sum_{j=1}^{N} sim(u_i, k_j) R_{k_j}(p) \quad \text{(Eq. 12)}$$

After we calculate the probability values, we assign a type to node u_i. If its malware probability is higher, we assign malware, if its phishing probability is higher, we assign phishing and if there is a tie between probabilities, its type remains unknown. In the case of equality some kind of heuristic approach could be used such as assigning the type of k_j that has the largest $sim(u_i,k_j)$ value. However, since we have a metric called coverage that shows the rate of a measure's applicability to classify nodes for these kinds of incidents, we let the type remain unknown in this case.

5.4. Results

In this section results are presented for two methods mentioned in Chapters 5.2 and 5.3. For each different neighbor sets experiments were run for 10 times with different random data and standard deviations for each evaluation metric are provided in the tables. Our general aim is to improve accuracy and coverage at the same time as much as it is possible, but it is shown that sometimes we need to make concessions from accuracy in order to increase coverage.

5.4.1. Malicious Link Neighbors Method's Results

Linked nature of the Web made it possible to form our network as a directed graph. Because of the links, a node has other nodes that it is in relationship with. We illustrated this relationship in five different neighborhoods around a node. Therefore, a node has 5 different link neighbors: incoming, outgoing, incoming + outgoing, co-citation and bibliographic coupling neighbors. Here we showed which neighbors provide the greatest value for threat type predictions. Each delta value had different results for accuracy and coverage. A smaller delta value let more nodes reach fixed status earlier and minimize the number of iterations. As we want to limit the number of iterations a delta value equals to 0.30 is used for the results presented in this chapter. Algorithm 1 – Iterative threat type predictor is run with different neighbor sets and results are presented below.

Table 11: Malicious link neighbors method results for using direct neighbors set

	Incoming Neighbors	Outgoing Neighbors	Incoming + Outgoing Neighbors
Avg. Accuracy	84.82 (σ=3.44)	88.87 (σ=3.01)	84.31 (σ=1.63)
Avg. Recall Malware Phishing	90.14 (σ=4.14) 73.85 (σ=2.96)	88.62 (σ=4.99) 88.34 (σ=4.71)	89.00 (σ=1.70) 74.49 (σ=4.84)
Avg. Precision Malware Phishing	86.99 (σ=14.37) 79.39 (σ=29.35)	91.41 (σ=3.06) 85.15 (σ=4.30)	87.92 (σ=2.70) 76.15 (σ=4.97)
Avg. Coverage	2.57 (σ=0.35)	1.51 (σ=0.33)	3.46 (σ=0.33)

Table 12: Confusion matrix for direct neighbors sets

Used Set / (N=4000)		Incoming Neighbors		Outgoing Neighbors		Incoming + Outgoing Neighbors	
		Predicted		Predicted		Predicted	
		Malware	Phishing	Malware	Phishing	Malware	Phishing
Actual	Malware	62 (σ=7.15)	6 (σ=1.15)	30 (σ=1.41)	4 (σ=1.00)	84 (σ=5.25)	10 (σ=1.18)
	Phishing	8 (σ=0.91)	26 (σ=2.48)	2 (σ=0.91)	24 (σ=5.55)	12 (σ=1.27)	34 (σ=2.90)

As seen from the Table 11, incoming and outgoing neighbors sets have high accuracy. However, their coverage is very low. At most they can classify up to 3.46% of nodes in the case we merge these two neighbors in incoming + outgoing links neighbors. However, since their accuracies are very high, they should be used whenever possible. Their low coverages are due to the fact that most nodes do not have any malware or phishing direct neighbor.

In order to improve the coverage we presented another type of neighbors called indirect neighbors. Incoming and outgoing neighbors only consider direct edges, but co-citation and bibliographic coupling neighbors utilize the graph structure that extend link relationships from only direct neighbors to new kind of neighbors formed via third nodes.

Table 13: Malicious link neighbors method results for using indirect neighbors set

	Co-citation Neighbors	Bibliographic Coupling Neighbors
Avg. Accuracy	69.64 (σ=0.77)	60.24 (σ=3.23)
Avg. Recall Malware Phishing	56.17 (σ=1.49) 84.69 (σ=0.78)	54.34 (σ=5.60) 66.31 (σ=4.11)
Avg. Precision Malware Phishing	80.40 (σ=0.64) 63.37 (σ=1.18)	62.24 (σ=3.41) 58.77 (σ=3.41)
Avg. Coverage	68.41 (σ=1.03)	52.11 (σ=0.56)

Table 14: Confusion Matrix for Indirect Neighbors Sets

Used Set / (N=4000)		Co-citation Neighbors		Bibliographic Coupling Neighbors	
		Predicted		Predicted	
		Malware	Phishing	Malware	Phishing
Actual	Malware	810 (σ=7.93)	634 (σ=15.4)	572 (σ=26.71)	482 (σ=32.7)
	Phishing	198 (σ=5.20)	1094 (σ=13.54)	348 (σ=20.87)	684 (σ=22.29)

Co-citation is shown to be a better measure to predict threat types than bibliographic coupling. It surpassed bibliographic coupling both in terms of accuracy and coverage. By using co-citation we are able to predict threat type with 69.64% accuracy.

Indirect neighbors, co-citation and bibliographic coupling, have increased coverage than direct neighbors. Therefore, by using them we could make predictions for more nodes. However, their accuracies are not as high as direct neighbors. In this case a hybrid method could be used. If direct neighbors can be applied to a node, we can use them and if it is not possible, we can use indirect neighbors methods.

Results for bibliographic coupling has higher standard deviations. This is due to the fact that many malicious websites in our network have zero outdegree as mentioned in Chapter 3.3 and since bibliographic coupling relationships are formed via outgoing links from malicious websites results vary for the randomly selected nodes in each experiment.

5.4.2. Link Similarity Method's Results

As mentioned earlier, similarity score $sim(u_i, k_j)$ between two nodes is calculated based on their links. We apply the similarity score for its incoming links, outgoing links and incoming + outgoing links.

Table 15: Results for Link Similarity Methods

	Used Link Set for Similarity Calculation		
	Incoming Links	Outgoing Links	Incoming + Outgoing Links
Avg. Accuracy	68.92 (σ= 0.68)	58.97 (σ= 0.98)	64.84 (σ= 0.65)
Avg. Recall Malware Phishing	61.23 (σ= 1.06) 77.53 (σ= 0.90)	67.29 (σ= 3.30) 50.22 (σ= 4.11)	66.68 (σ= 1.34) 62.89 (σ= 1.08)
Avg. Precision Malware Phishing	75.37 (σ= 0.53) 64.06 (σ= 0.83)	58.89 (σ= 1.43) 59.24 (σ= 1.60)	65.57 (σ= 0.69) 64.05 (σ= 0.83)
Avg. Coverage	69.74 (σ= 0.69)	52.03 (σ= 0.65)	85.59 (σ= 0.27)

Table 16: Confusion matrix for direct neighbors sets

Used Set / (N=4000)		Incoming Neighbors		Outgoing Neighbors		Incoming + Outgoing Neighbors	
		Predicted		Predicted		Predicted	
		Malware	Phishing	Malware	Phishing	Malware	Phishing
Actual	Malware	904 (σ=23.62)	572 (σ=10.91)	719 (σ=22.67)	351 (σ= 42.02)	1175 (σ=22.63)	587 (σ= 24.59)
	Phishing	295 (σ= 9.75)	1019 (σ=24.35)	503 (σ=42.25)	508 (σ=42.11)	617 (σ=19.24)	1045 (σ= 16.77)

With link similarity method we can predict a threat type with 68.92% of accuracy in incoming link set. Also by combining incoming and outgoing link set we can make predictions for 85.59% of analyzed nodes with an accuracy of 64.84%.

When analyzing the data in the initial trials we realized that some of the nodes have high indegrees and outdegrees. For example, search engines like Google, Yahoo, Bing and social media sites like Twitter and Facebook have both high indegrees and outdegrees. For search engines other websites tend to link them for custom searches and search engines should provide link towards them as a result of users' search queries. Similarly including a Facebook Like button or Twitter Share button to a website creates a link towards them. Many websites also use analytics tools such as Google Analytics, or use Googleapis service for importing JavaScript libraries which

adds a link to these services. Consider a scenario where a websites u and k have a Facebook Like button in their sources but no other incoming and outgoing links. In this case their Jaccard similarity will be 1, the maximum possible value. These links are neither placed for related content nor for endorsement. Therefore, they may possibly skew the results. We detected 100 of these websites where edges that includes them happen because of their functionality and added to a whitelist that can be seen in Appendix-1 and removed their edges from our graph and reiterated our analysis again.

Table 17: Results for Link Similarity Methods after removing 100 functional links

	Used Link Neighbor Set for Similarity Calculation		
	Incoming Links	Outgoing Links	Incoming + Outgoing Links
Avg. Accuracy	70.04 (σ= 1.40)	60.78 (σ= 0.74)	67.18 (σ=0.64)
Avg. Recall Malware Phishing	 67.75 (σ= 2.58) 72.63 (σ= 1.11)	 71.59 (σ= 0.84) 49.19 (σ= 2.23)	 71.35 (σ=1.10) 62.68 (σ=1.25)
Avg. Precision Malware Phishing	 73.66 (σ= 1.29) 66.61 (σ= 1.80)	 60.19 (σ= 1.04) 61.73 (σ= 0.82)	 67.35 (σ=0.71) 66.99 (σ=0.86)
Avg. Coverage	66.82 (σ= 0.57)	42.24 (σ= 0.76)	81.19 (σ= 0.76)

Table 18: Confusion matrix for direct neighbors sets after removing 100 functional links

Used Set / (N=4000)		Incoming Links		Outgoing Links		Incoming + Outgoing Links	
		Predicted		Predicted		Predicted	
		Malware	Phishing	Malware	Phishing	Malware	Phishing
Actual	Malware	945 (σ=36.38)	450 (σ=38.82)	615 (σ= 9.64)	244 (σ= 9.15)	1182 (σ=21.81)	475 (σ=19.11)
	Phishing	338 (σ=14.81)	896 (σ=11.11)	407 (σ=20.94)	394 (σ=21.56)	573 (σ=21.29)	963 (σ=20.86)

By removing websites where the links around them occur due to their functionalities, we were able to increase accuracies for each neighbor sets. We also passed 70% accuracy in incoming links neighbors. Coverages however were decreased. Since we deleted some edges that prevented possible formation of neighbors, this was an expected outcome.

For threat type identification, we managed to achieve at most 88.87% of accuracy but with limited coverages in link neighbors method. In link similarity method we achieved at most 70.04% of accuracy with relatively larger coverage. Results in the threat type identification are above the possible random assignment accuracy result of 50% but may not seem significantly high on their own. However, the methods

here only include link analysis. If they could be combined with other methods when analyzing malicious websites such as text mining, visual similarity etc. they proved to have the potential to provide contribution. Therefore, results obtained in this chapter shows us that link analysis is promising for threat type identification.

CHAPTER 6

CONCLUSIONS

This chapter discusses the results regarding the analysis made in the previous chapters. Results for malicious website and threat type identification is presented in their respective chapters.

Detecting malicious websites is a popular subject of study in the literature. What we presented is pairing it with social network analysis and investigating methods for finding malicious websites and threat type identification more effectively using the linked nature of WWW.

Social network analysis methods were used throughout this study. We created the networks of malicious websites and our analyses were built upon them. Using techniques mentioned in finding malicious websites chapter, we could form a priority list of suspicious websites whose analyses would let us run into more malicious websites in them. Combining these techniques mentioned in threat type identification chapter we could predict these suspicious websites' threat types and tailor our analysis accordingly. If a website is predicted to be phishing, we can apply phishing detection techniques first, or if it is predicted as malware, we can apply malware detection techniques. This is due to the fact that malicious website detection is an expensive process. There are lots of different detection techniques for both malware and phishing. As an example, dynamic analysis methods for malware, runs the executables in a sandboxed environment and analyze the behavior of them. Consider websites with lots of executables in them. In order to truly classify them as malicious, one needs to download and analyze every file in them which has high computational costs. However, with a pre filtering mechanism, both the selection of the websites to be analyzed and their possible threat types could be predicted beforehand in order to save computational costs. This will also help us to minimize the opportunity cost, the loss of other options when one option is chosen, since with limited resources when you analyze some websites, you do not analyze others therefore they may evade the detection.

We revealed that malicious websites tend to link other malicious websites in the same maliciousness category as them, i.e. malware websites tend to link other malware websites, phishing websites tend to link other phishing websites than other categories. This notion is studied and proven correct mainly for web spam websites in the literature. For malware and phishing types this is the second study that investigate this notion.

We showed that focusing our analysis on the core groups in a network of malicious websites would help us to find other malicious websites more efficiently. By

efficient we mean analyzing a smaller number of websites but running into relatively more malicious websites in them.

The most prominent difference from the previous research and hence our contribution lies in our usage of co-citation and bibliographic coupling for attack type identification of malicious websites. Co-citation and bibliographic coupling are applied to many areas from bibliographic studies to topic similarity. We applied them for the first time to attack type prediction of malicious websites.

We showed that link analysis is a promising area for effective malicious website detection and threat type identification. However, the experiments we carried uses only the link analysis. For better results they can be used alongside with other methods since a hybrid method that includes link analysis would provide better results. For example, to detect phishing websites link analysis methods could be used with textual features and visual similarity methods. The findings here could also be used as one of the features to train a machine learning model.

REFERENCES

[1] Accenture. Ninth Annual Cost of Cybercrime Study [Internet]. 2017. Available from: https://www.accenture.com/t20170926t072837z__w__/us-en/_acnmedia/pdf-61/accenture-2017-costcybercrimestudy.pdf

[2] Wasserman S, Faust Katherine. Social Networks Analysis: Methods and Applications. Cambridge: Cambridge University Press; 1994.

[3] Accenture. Cost of Cybercrime Study 2017 [Internet]. 2017. Available from: https://www.accenture.com/t20170926t072837z__w__/us-en/_acnmedia/pdf-61/accenture-2017-costcybercrimestudy.pdf

[4] Transparency Report G. Safe Browsing: malware and phishing – Google Transparency Report [Internet]. 2019 [cited 2019 Nov 28]. Available from: https://transparencyreport.google.com/safe-browsing/overview?unsafe=dataset:1;series:malwareDetected,phishingDetected;start:1543881600000;end:1573516799999&lu=unsafe

[5] Transparency Report G. Safe Browsing: malware and phishing – Google Transparency Report [Internet]. 2019 [cited 2019 Nov 28]. Available from: https://transparencyreport.google.com/safe-browsing/overview?unsafe=dataset:0;series:malware,phishing;start:1148194800000;end:1573977600000&lu=unsafe

[6] Sikorski M, Honig A. Practical malware analysis: the hands-on guide to dissecting malicious software. No starch Press. 2012.

[7] Moshchuk A, Bragin T, Gribble SD, Levy HM. A Crawler-based Study of Spyware on the Web. Proc 2006 Netw Distrib Syst Secur Symp. 2006;

[8] Wang Y-M, Beck D, Jiang X, Roussev R. Automated Web Patrol with Strider HoneyMonkeys : Finding Web Sites That Exploit Browser Vulnerabilities. Proc Symp Netw Distrib Syst Secur. 2005;0–11

[9] Provos N, McNamee D, Mavrommatis P, Wang K, Modadugu N. The Ghost in the Browser Analysis of Web-based Malware. Proc First Conf First Work Hot Top Underst Botnets [Internet]. Berkeley, CA, USA: USENIX Association; 2007. p. 4. Available from: http://dl.acm.org/citation.cfm?id=1323128.1323132

[10] Provos N, Mavrommatis P, Rajab MA, Monrose F. All Your iFRAMEs Point to Us. Proc 17th Conf Secur Symp [Internet]. Berkeley, CA, USA: USENIX Association; 2008. p. 1–15. Available from: http://dl.acm.org/citation.cfm?id=1496711.1496712

[11] Singh AK, Goyal N. Malcrawler: A crawler for seeking and crawling malicious websites. Lect Notes Comput Sci (including Subser Lect Notes Artif Intell Lect Notes Bioinformatics). 2017

[12] Invernizzi L, Comparetti PM, Benvenuti S, Kruegel C, Cova M, Vigna G. EvilSeed: A guided approach to finding malicious web pages. Proc - IEEE Symp Secur Priv. 2012

[13] Chakrabarti S, Dom B, Indyk P. Enhanced hypertext categorization using hyperlinks. SIGMOD Rec. 1998

[14] Qi X, Davison BD. Knowing a web page by the company it keeps. Int Conf Inf Knowl. Manag Proc. 2006

[15] Dean J, Henzinger MR. Finding related pages in the World Wide Web. Comput Networks. 1999

[16] Kleinberg JM. Authoritative sources in a hyperlinked environment. J ACM. 1999

[17] Wu B, Davison BD. Identifying link farm spam pages. 14th Int World Wide Web Conf WWW2005. 2005

[18] Castillo C, Donato D, Gionis A, Murdock V, Silvestri F. Know your neighbors: Web spam detection using the web topology. Proc 30th Annu Int ACM SIGIR Conf Res Dev Inf Retrieval, SIGIR'07. 2007

[19] Gibson D, Kumar R, Tomkins A. Discovering large dense subgraphs in massive graphs. VLDB 2005 - Proc 31st Int Conf Very Large Data Bases. 2005

[20] Gyöngyi Z, Garcia-Molina H. Link spam alliances. VLDB 2005 - Proc 31st Int Conf Very Large Data Bases. 2005

[21] Page L, Brin S, Motwani R, Winograd T. The PageRank Citation Ranking: Bringing Order to the Web. World Wide Web Internet Web Inf Syst. 1998

[22] Fetterly D, Manasse M, Najork M. Spam, damn spam, and statistics: Using statistical analysis to locate spam web pages. ACM Int Conf Proceeding Ser. 2004

[23] Becchetti L, Castillo C, Donato D, Baeza-Yates R, Leonardi S. Link analysis for Web spam detection. ACM Trans Web. 2008

[24] Choi H, Zhu BB, Lee H. Detecting malicious web links and identifying their attack types. WebApps. 2011

[25] Stokes JW, Andersen R, Seifert C, Chellapilla K. WebCop: Locating neighborhoods of malware on the web. LEET 2010 - 3rd USENIX Work Large-Scale Exploit Emergent Threat Botnets, Spyware, Worms, More. 2010

[26] Wu B, Goel V, Davison BD. Propagating trust and distrust to demote web spam. CEUR Workshop Proc. 2006

[27] Gyongyi Z, Garciamolina H, Pedersen J. Combating Web Spam with TrustRank. Proc 2004 VLDB Conf. 2004

[28] Guha R, Raghavan P, Kumar R, Tomkins A. Propagation of trust and distrust. Thirteen Int World Wide Web Conf Proceedings, WWW2004. 2004

[29] Travers J, Milgram S. An Experimental Study of the Small World Problem. Sociometry [Internet]. [American Sociological Association, Sage Publications, Inc.]; 1969;32:425–43. Available from: http://www.jstor.org/stable/2786545

[30] Kleinfeld JS. The small world problem. Society. 2002

[31] Adamic LA. The small world web. Lect Notes Comput Sci (including Subser Lect Notes Artif Intell Lect Notes Bioinformatics). 1999

[32] Shetty J, Adibi J. Discovering important nodes through graph entropy the case of enron email database. 3rd Int Work Link Discov LinkKDD 2005 - conjunction with 10th ACM SIGKDD Int Conf Knowl Discov Data Min. 2005

[33] Phillips E, Nurse JRC, Goldsmith M, Creese S. Applying Social Network Analysis to Security. 2015 [cited 2019 Aug 26]. Available from: http://www.ferc.gov/

[34] Boldi P, Vigna S. The Webgraph Framework I: Compression Techniques. Proc 13th Int Conf World Wide Web [Internet]. New York, NY, USA: ACM; 2004. p. 595–602. Available from: http://doi.acm.org/10.1145/988672.988752

[35] Flake GW, Lawrence S, Lee Giles C, Coetzee FM. Self-organization and identification of web communities. Computer (Long Beach Calif). 2002

[36] Aburrous M, Hossain MA, Dahal K, Thabtah F. Intelligent phishing detection system for e-banking using fuzzy data mining. Expert Syst Appl. 2010

[37] Abu-Nimeh S, Nappa D, Wang X, Nair S. A comparison of machine learning techniques for phishing detection. ACM Int Conf Proceeding Ser. 2007

[38] Medvet E, Kirda E, Kruegel C. Visual-similarity-based phishing detection. Proc 4th Int Conf Secur Priv Commun Networks, Secur. 2008

[39] Chen TC, Dick S, Miller J. Detecting visually similar web pages: Application to phishing detection. ACM Trans Internet Technol. 2010

[40] Idika N, Mathur AP. A Survey of Malware Detection Techniques. SERC Tech Reports. 2007

[41] Kolbitsch C, Livshits B, Zorn B, Seifert C. Rozzle: De-cloaking Internet malware. Proc - IEEE Symp Secur Priv. 2012. p. 443–57

[42] Zachary WW. An Information Flow Model for Conflict and Fission in Small Groups. J Anthropol Res. 1977

[43] Girvan M, Newman MEJ. Community structure in social and biological networks. Proc Natl Acad Sci U S A. 2002

[44] Stanford Large Network Dataset Collection [Internet]. [cited 2019 Aug 26]. Available from: https://snap.stanford.edu/data/#web

[45] Laboratory for Web Algorithmics [Internet]. [cited 2019 Aug 26]. Available from: http://law.di.unimi.it/datasets.php

[46] Frequently Asked Questions – Common Crawl [Internet]. [cited 2019 Aug 26]. Available from: https://commoncrawl.org/big-picture/frequently-asked-questions/

[47] Get started – Common Crawl [Internet]. [cited 2019 Aug 26]. Available from: https://commoncrawl.org/the-data/get-started/

[48] May 2019 crawl archive – Common Crawl [Internet]. [cited 2019 Aug 26]. Available from: https://commoncrawl.org/2019/05/may-2019-crawl-archive-now-available/

[49] July 2019 crawl archive – Common Crawl [Internet]. [cited 2019 Aug 26]. Available from: https://commoncrawl.org/2019/07/july-2019-crawl-archive-now-available/

[50] Du Y, Herzog A, Luckow A, Nerella R, Gropp C, Apon A. Representativeness of latent dirichlet allocation topics estimated from data

samples with application to common crawl. Proc - 2017 IEEE Int Conf Big Data, Big Data 2017. 2018

[51] URLhaus | Malware URL exchange [Internet]. [cited 2019 Aug 26]. Available from: https://urlhaus.abuse.ch/

[52] DNS-BH – Malware Domain Blocklist by RiskAnalytics [Internet]. [cited 2019 Aug 26]. Available from: http://www.malwaredomains.com/

[53] Suspicious Domains - SANS Internet Storm Center [Internet]. [cited 2019 Aug 26]. Available from: https://isc.sans.edu/suspicious_domains.html

[54] PhishTank | Join the fight against phishing [Internet]. [cited 2019 Aug 26]. Available from: https://www.phishtank.com/

[55] Github. GitHub - HorusTeknoloji/TR-PhishingList: Türkiye'ye Yönelik Zararlı Bağlantı Erişim Engelleme Listesi [Internet]. [cited 2019 Dec 20]. Available from: https://github.com/HorusTeknoloji/TR-PhishingList

[56] How it works – VirusTotal [Internet]. [cited 2019 Aug 26]. Available from: https://support.virustotal.com/hc/en-us/articles/115002126889-How-it-works

[57] Developers - VirusTotal [Internet]. [cited 2019 Aug 26]. Available from: https://developers.virustotal.com/

[58] Pajek / PajekXXL / Pajek3XL [Internet]. [cited 2019 Nov 28]. Available from: http://mrvar.fdv.uni-lj.si/pajek/

[59] Pfeffer J, Mrvar A, Batagelj V. txt2pajek: Creating Pajek Files from Text Files. 2013

[60] Kumar R, Raghavan P, Rajagopalan S, Sivakumar D, Tomkins A, Upfal E. Stochastic models for the web graph. Annu Symp Found Comput Sci - Proc. 2000

[61] Barabási AL, Albert R. Emergence of scaling in random networks. Science (80-). 1999

[62] Davis J, Goadrich M. The relationship between precision-recall and ROC curves. ACM Int Conf Proceeding Ser. 2006

[63] Wikipedia. Confusion matrix - Wikipedia [Internet]. [cited 2019 Dec 20]. Available from: https://en.wikipedia.org/wiki/Confusion_matrix

[64] Lu Q, Getoor L. Link-based Classification. Proceedings, Twent Int Conf Mach Learn. 2003

[65] Small H. Co-citation in the scientific literature: A new measure of the relationship between two documents. J Am Soc Inf Sci. 1973

[66] Kessler MM. Bibliographic coupling between scientific papers. Am Doc. 1963

[67] Bharat K, Chang BW, Henzinger M, Ruhl M. Who links to whom: Mining linkage between web sites. Proc - IEEE Int Conf Data Mining, ICDM. 2001

[68] Han J, Kamber M, Pei J. Data Mining: Concepts and Techniques. Data Min. Concepts Tech. 2012

[69] Niwattanakul S, Singthongchai J, Naenudorn E, Wanapu S. Using of jaccard coefficient for keywords similarity. Lect Notes Eng Comput Sci. 2013

APPENDICES

Appendix A: 100 websites whose links occur due to their functionalities:

Website Name	Functionality	Website Name	Functionality
addthis.com	Bookmarking	histats.com	Website statistic
amazon.co.jp	Shopping	hotmail.com	Web Portal
amazon.co.uk	Shopping	hstatic.net	Content Delivery
amazon.com	Shopping	imageshack.com	Photo Sharing
amazonaws.com	Cloud	imgur.com	Photo Sharing
amzn.to	Link Shortening	instagram.com	Social Media
aol.com	Web Portal	joomla.org	Website Creator
apple.com	Business	jquery.com	Programming
archive.org	Web Archive	jqueryui.com	Programming
azurewebsites.net	Cloud	linkedin.com	Social Media
baidu.com	Search Engine	medium.com	Blogging
bit.ly	Link Shortening	microsoft.com	Business
blogspot.com	Blogging	msn.com	Web Portal
bootstrapcdn.com	Content Delivery	opendns.com	Network
cdninstagram.com	Content Delivery	paypal.com	Payment Processor
cloudflare.com	Web Protection	people.com.cn	Web Portal
cpanel.com	Web Hosting	pinterest.com	Photo Sharing
dailymotion.com	Video Hosting	rawgit.com	Programming
disqus.com	Website Comment	reddit.com	Social Media
dropbox.com	Cloud	sharethis.com	Bookmarking
duckduckgo.com	Search Engine	shopify.com	Shopping
ebay.com	Shopping	ssl-images-amazon.com	Content Delivery
facebook.com	Social Media	statcounter.com	Website statistic
facebook.net	Social Media	stripe.com	Payment Processor
fb.me	Link Shortening	stumbleupon.com	Bookmarking
flickr.com	Photo Sharing	t.co	Link Shortening
fontawesome.com	Web Programming	t.me	Link Shortening
github.com	Programming	taobao.com	Shopping
godaddy.com	Cloud	tinypic.com	Photo Sharing

goo.gl	Link Shortening	tinyurl.com	Link Shortening
google-analytics.com	Analytics	tumblr.com	Social Media
google.at	Search Engine	twitter.com	Social Media
google.co.in	Search Engine	vimeo.com	Video Hosting
google.co.jp	Search Engine	vk.com	Social Media
google.com	Search Engine	w.org	Blogging
google.com.br	Search Engine	weebly.com	Programming
google.com.hk	Search Engine	whatsapp.com	Social Media
google.com.tr	Search Engine	wikimedia.org	Website Creator
google.de	Search Engine	wikipedia.org	Website Creator
google.es	Search Engine	wordpress.com	Website Creator
google.hu	Search Engine	wordpress.org	Website Creator
googleapis.com	Programming	wp.com	Website Creator
googlecode.com	Programming	yahoo.com	Search Engine
googledrive.com	Cloud	yandex.com	Search Engine
googlegroups.com	Social Media	yandex.net	Search Engine
googlesyndication.com	Programming	yandex.ru	Search Engine
googletagmanager.com	Programming	yelp.com	Business Directory
googleusercontent.com	Programming	youtu.be	Link Shortening
gravatar.com	Photo Sharing	youtube.com	Social Media
gstatic.com	Content Delivery	ytimg.com	Content Delivery

TEZ İZİN FORMU / THESIS PERMISSION FORM

ENSTİTÜ / INSTITUTE

Fen Bilimleri Enstitüsü / Graduate School of Natural and Applied Sciences [ ]

Sosyal Bilimler Enstitüsü / Graduate School of Social Sciences [ ]

Uygulamalı Matematik Enstitüsü / Graduate School of Applied Mathematics [ ]

Enformatik Enstitüsü / Graduate School of Informatics [X]

Deniz Bilimleri Enstitüsü / Graduate School of Marine Sciences [ ]

YAZARIN / AUTHOR

Soyadı / Surname : ALDEMİR
Adı / Name : MUHSİN
Bölümü / Department : BİLİŞİM SİSTEMLERİ / INFORMATION SYSTEMS

TEZİN ADI / TITLE OF THE THESIS (**İngilizce** / English) : SOCIAL NETWORK ANALYSIS OF MALICIOUS WEBSITES FOR DETECTION AND CHARACTERIZATION

TEZİN TÜRÜ / DEGREE: **Yüksek Lisans** / Master [X] **Doktora** / PhD [ ]

1. **Tezin tamamı dünya çapında erişime açılacaktır.** / Release the entire work immediately for access worldwide. [ ]
2. **Tez iki yıl süreyle erişime kapalı olacaktır.** / Secure the entire work for patent and/or proprietary purposes for a period of **two year. *** [ ]
3. **Tez altı ay süreyle erişime kapalı olacaktır.** / Secure the entire work for period of **six months**. ***** [X]

***** *Enstitü Yönetim Kurulu Kararının basılı kopyası tezle birlikte kütüphaneye teslim edilecektir.*
A copy of the Decision of the Institute Administrative Committee will be delivered to the library together with the printed thesis.

Yazarın imzası / Signature ..... **Tarih** / Date /12/2019