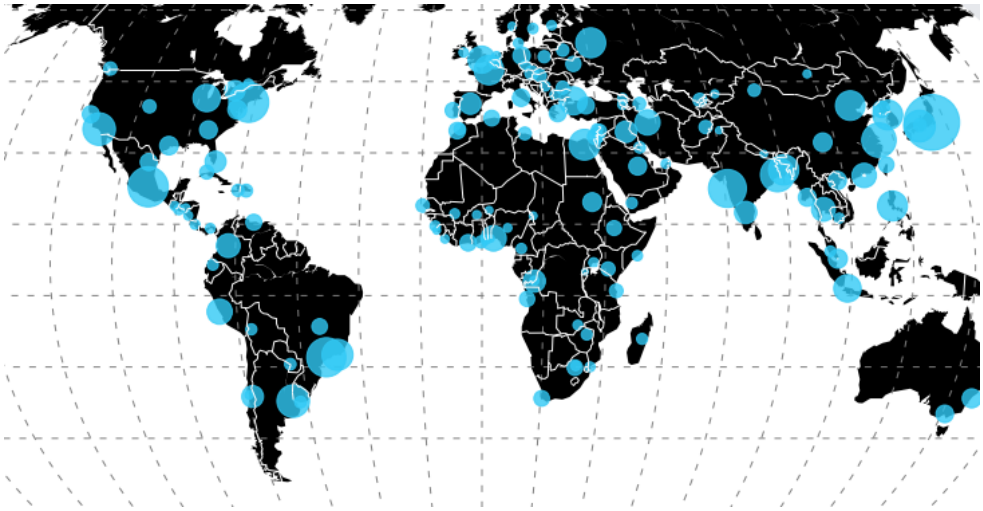
UNIVERSITY OF TWENTE

Faculty of Electrical Engineering, Mathematics and Computer Science
Design and Analysis of Communication Systems group



MASTER OF SCIENCE THESIS

Effective granularity in Internet badhood detection: Detection rate, Precision and Implementation performance

Ali Davanian

Committee:
dr.ir. Aiko Pras
A. Warmenhoven
dr. A. Sperotto
dr. G. Moura

August 1, 2017

Abstract

New malicious nodes appear everyday in Internet. Previous studies have shown that these nodes are not randomly distributed in Internet; similar to high density of criminal activities in real world bad neighborhoods, there exist Internet bad neighborhoods. Two common features to draw the local network boundaries within Internet and hence identifying the bad neighborhoods are fixed /24 IP prefix and dynamic Border Gateway Protocol (BGP) IP prefix. The main difference between these two features is the size of the underlying neighborhood and hence the granularity in the measurement of malicious activity. In this study, by analyzing a dataset of Command and Control servers and botnets, we show that BGP prefix is preferred in identifying bad neighborhoods because it offers 10% better detection rate in identifying new malicious nodes. In summary, our contributions in this study are:

1. We show that likelihood of having malicious activity from a neighborhood is a precise metric for malice prediction based on regression modeling
2. we show that likelihood of having malicious activity in adjacent /24 prefixes withing a BGP prefix is very similar with a 4% of change on average and hence the loss of granularity by using BGP is negligible
3. we show that BGP prefixes offer a detection rate improvement of 10%
4. we conclude that based on precision, granularity, detection rate and lookup performance BGP is a better feature for Internet bad neighborhood identification

Contents

Chapter 1

Introduction

Internet has become an inseparable part of our daily lives. At the same time, the security of Internet world has become one of the main concerns in the world. Only in 2016, over 100000 cyber incidents were reported targeting Health institutions, financial organizations, Industrial systems etc [1]. One of the main sources of these cyber incidents is malware. Malware is a piece of program that its purpose is to do some harm to the host system it infects. For instance, one of the malware that caused much damage in 2016 was Mirai botnet [2]. This malware could temporarily paralyze part of the Internet connectivity using a type of attack known as Distributed Denial Of Service(DDOS) attack. In this type of attack, many infected hosts at the same time send traffic to a destination and hence disrupt the service. Another instance of malware that caused a cyber incident in 2017 is Wannacry ransomware. Ransomware make the data on the infected computer inaccessible to the user and ask for a ransom in return of the data. It is estimated that this malware caused $4 billion of damage [3].

Internet is actually a collection of smaller local networks that are connected together. Evidence shows that cyber attackers are more concentrated in some of these local networks than others. In order to better understand the concept, we make an analogy to the real world so called "bad neighborhoods". In New York, law enforcement tags some neighborhoods as "bad" based on the number of crime reports. The crime rate in some neighborhoods is higher due to variety of reasons such as poor economy of the neighborhood. These neighborhoods in future are expected to foster more malicious activities, and henceforth, law enforcement would carefully monitor these neighborhoods. Similar to the real world, there exist bad neighborhoods in the Internet world.

Internet bad neighborhood concept can be used to predict attackers' source addresses and mitigate cyber threats [16]. For example, figures 1.1 shows the distribution of the infected hosts by Mirai. We can see from the figure that infected hosts are more concentrated in some regions. This is because these regions, or hereafter Internet bad neighborhoods, have poor security measures in place and infection spreads quickly within them. Given this, we can prevent DDOS attack from these infected hosts by blocking traffic from them; after see-

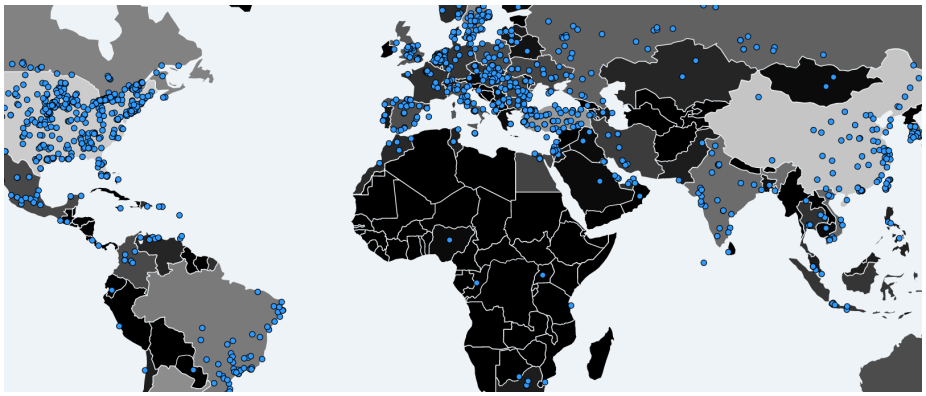


Figure 1.1: Mirai distribution of infected hosts

ing malicious traffic from a few IP addresses, we can predict the attack from the remaining IP addresses of the neighborhood and hence prevent it. *Identifying Internet bad neighborhoods enables Intrusion Detection Systems to predict attacks from neighbor nodes within a network.*

In order to identify Internet bad neighborhoods with homogeneous behavior, we need to understand the structure of Internet. Internet is composed of several Autonomous Systems (AS). An Autonomous System is a large network that is responsible for the distribution of traffic within the network. An AS, itself, consists of several smaller networks. The communication between different Autonomous Systems is through Border Gateway Protocol (BGP). Via Border Gateway Protocol, an AS announces the IP addresses of the smaller networks it controls i.e. through BGP prefix announcements. A BGP prefix is in X.Y.Z.W/N format where X,Y,Z and W are integers between 0 to 255 and N is an integer between 8 to 24. Based on this mechanism, the neighbor ASes pass the traffic to one another until it reaches the AS that contains the destination IP address.

Based on the above explanation, one may assume that an autonomous system is a neighborhood. Although that assumption is not wrong, the neighborhood in such case is too large. We can not expect a homogeneous malicious or benign behavior from such large entity because an AS is further split into smaller networks that are managed by different administration entities. For instance, many of the autonomous systems are actually Internet Service Providers (ISPs). An ISP would use a BGP prefix for DSL users, a prefix for WAN users, a prefix for servers in a rack and another for a large organization such as a university. Since these prefixes are used for different purposes and have different administration entities, we can not expect the same behavior from all of them.

Two common approaches to identify an Internet neighborhood to which an IP address belongs are based on **BGP** announcements and Classless Inter-Domain Routing(CIDR) /24 prefixes, hereafter as **fixed /24 prefixes**. The former is based on the explanation that we gave in the previous paragraph; we

can expect same behavior from the nodes within /BGP prefixes since they have the same administration policies and are probably physically neighbors [17]. The latter is based on the logic that a BGP prefix length can be as long as 24 [13]. Hence, a /24 prefix is a smaller neighborhood within a BGP prefix.

Our research goal in this paper is to **compare** the Internet bad neighborhoods based on the BGP and fixed /24 prefixes. The difference in the **size** of the neighborhoods derived from each approach affects the **error** and the **latency** in the prediction of attack from an IP address. The error can be misidentifying a traffic as malicious i.e False Positive (FP) or benign i.e False Negative (FN). The low rate of FP and FN are of high importance for an Intrusion Detection System. A high FP rate results in large number of intrusion alerts that are in fact wrong. Therefore, the human operator has to manually validate the accuracy of many alerts. A high FN means many attacks remain undetected.

Size of neighborhoods has a direct impact on both **FP and FN**. The larger a neighborhood, the lower false negative rate of Intrusion Detection. This is because we blacklist more IP addresses by enlarging neighborhoods. For instance, assume that we blacklist the entire Internet i.e the whole IP space. In this case, the FN rate is zero since we label traffic from any given IP address as malicious. On the other hand, by enlarging the neighborhood size we also increase FP rate. This is because FN and FP have reverse correlation [14]. For instance, in blacklisting the entire Internet, many of our detections are actually false because not every traffic is malicious.

Size of neighborhood also impacts the latency of the Intrusion Detection System in checking a traffic malice. To understand the effect of the neighborhood size on latency, assume that there is only one neighborhood with 256 IP addresses. In such case, only the traffic from these 256 IP addresses would be checked for Intrusion analysis. In contrast, assume that the neighborhood has 1 million IP addresses. In such case, the Intrusion Detection System has to analyze the traffic from these 1 million sources and this delays the detection.

In summary, *our research goal is to compare the Intrusion detection and prediction error and latency of Internet bad neighborhoods based on BGP and fixed /24 prefixes.* In the following section 1.1, we discuss the background of the Internet bad neighborhood topic. In section 1.2, we discuss the granularity (effect of size) of Internet bad neighborhood. In section 1.3, we present our precise research questions and our approach to answer those questions. In section 1.3.1, we explain the value of our research for Intrusion Detection systems.

1.1 Background

The concept of Internet bad neighborhood is built upon IP addresses reputation. IP addresses reputation is often one of the elements in a vector of features used for malice detection. Reputation of an IP address is defined based on its historical malicious activities. The reputation of an IP address is usually shown with 1 and 0 representing,in order, listing or not listing in a blacklist. The intuition behind IP address reputation is that if an attacker attacks once,

it will attack again in near future and we can identify the same attacker by its IP address. For instance, IP reputation, among other features, is used in [4, 5] for spam detection. In [6], IP reputation is used in a vector of features to detect malicious DNS addresses. Since reputation by definition depends on the historical records, it has limited capability in malice detection; we must have already detected malicious activity for an IP address in order to further blacklist it.

Single IP address reputation has several limitations. Firstly, IP addresses are often leased for a short period. Due to the short leasing time [7, 8], the historical malicious activity of the current IP address can not always accurately be attributed to the endpoint (source or destination of a traffic) that holds the IP address. Secondly, the IP address space is 4G large. Tracking the reputation of all these nodes is expensive in a sense that we need to have traffic data from all the 4G space in order to detect all the malicious addresses. We also need to store the reputation of these nodes that might be expensive for network appliances with limited disk space and memory size. As a result, False Negative (FN) rate of IP blacklists is high [9]. Sinha et al measure the effectiveness of blacklists for Spam Detection. They find out that 21% of the traffic that Spam detectors such as SpamAssassin can detect remain undetected by blacklists. This is mainly because a large portion of the missed sources (around 90%) are observed for just one second in the network [9]. Such a short lifetime would not allow blacklist maintainers to effectively identify and report such malicious sources.

Findings from previous works suggest that malicious activities are concentrated more in some Internet neighborhoods. We can exploit this fact to predict attacks from neighbors and reduce FN rate. Collins et al. realized that deduced IP prefixes from blacklists' entries are not randomly distributed [10]. Based on their result, it's likely to see more malicious activity from the same network of an already seen malicious host, similar to dangerous neighborhoods in the real world. [11, 12, 4, 5, 13] also reach the same findings. For instance, based on the reputation of 1.1.1.1, 1.1.1.2 and 1.1.1.3 IP addresses we can derive the reputation of 1.1.1.0/30 subnet and assume that every traffic from this subnet is malicious. Several studies confirm the effectiveness of the aggregated reputation in spam detection [10, 4, 5, 14].

1.2 Granularity

We identify Internet bad neighborhoods by aggregating the reputation of a few nodes and then attributing it to the network from where those nodes come. The aggregation of nodes reputation can be carried out with different levels of granularity and based on a feature, hereafter as aggregation feature. From a very high level view, with a low granularity, we can aggregate the reputation of the nodes within an Autonomous System (AS) and use Autonomous System Number (ASN) as the aggregation feature. Although ASN for reputation aggregation is used in [12, 15, 16] in order to find malicious Autonomous Systems, we

don't find ASN granular enough for a reputation attribution. In the following paragraphs we explain why.

Nodes within an AS are further grouped under some IP routing prefixes that may be controlled by one or more network operators. For instance, Internet Service providers may use one IP prefix for their DSL users and another for the mobile users. They may further lease an entire prefix to an organization like a university. The autonomous system identified by ASN 1103 or name SURFnet is a service provider in Netherlands that serve multiple universities and organizations under different IP prefixes. Each of these organizations has different network administration policies and hence different levels of security. Furthermore, the nodes within an organization are likely to be geographically close to each other. Henceforth, BGP prefixes are used to cluster Internet nodes in different networks [17, 11, 12, 4, 14].

Krishnamurthy and Wang coined the term network aware clustering [17]. Via traceroute and reverse DNS resolving, they find out that 90% of IP addresses can be correctly clustered in this way. Jung et al used network aware clustering to predict IP addresses that perform DOS attack in the near future. Network aware clustering has been frequently used for spam filtering [11]. [4, 14]. These works assume that BGP prefixes draw the network boundaries under the same administration. Therefore, we can assume that nodes within an administration network would expose the same behavior i.e. the IP BGP prefix would entail the neighborhood to which a node belongs. That said, the networks derived from BGP prefixes have different sizes and hence the attributed reputations have different granularities.

The longest BGP prefix length is 24 and hence the derived reputation from these prefixes has the highest level of granularity i.e. we generalize the reputation of only 256 nodes. Based on this, /24 prefix aggregation has been chosen to identify bad neighborhoods in [10, 18, 13]. While ASNs and BGP prefixes carry semantics and entail network boundaries, /24 prefixes are not correspondent to any specific semantics in Internet routing except that they are the smallest chunks within an ASN or BGP prefix. That said, it is unclear that fine granularity of /24 aggregation leads to any particular advantage in terms of malice detection error and latency.

Since /24 prefixes are subset of a BGP prefix, the definition of neighborhood can apply to both aggregations i.e. based on BGP prefix feature or /24 prefix feature. The difference is the granularity; BGP neighborhood is a larger or same size neighborhood in comparison to /24 prefix. The level of granularity has an effect on the false positive and false negative [14]. The larger our aggregation group, the better our detection rate and false negative. At the same time, by enlarging the aggregation size and generalizing the reputation we should expect a higher false positive rate [14]. Furthermore, the difference in granularity results in the difference in the number of stored entries in our reputation database. Such difference, in turn, affects the lookup performance when inquiring the reputation database.

Prefix CIDR	**Malicious IP addresses**	**Likelihood** = $\frac{Malicious\ IP\ addresses}{Prefix\ CIDR\ size}$
2.181.224.0/24	2	0.01
2.181.225.0/24	3	0.01
2.181.226.0/24	3	0.01
2.181.227.0/24	2	0.01
2.181.228.0/24	1	0.00
2.181.229.0/24	1	0.00
2.181.231.0/24	2	0.01
2.181.232.0/24	3	0.01
2.181.238.0/24	5	0.02
2.181.239.0/24	4	0.02
2.181.242.0/24	1	0.00
2.181.244.0/24	2	0.01
2.181.245.0/24	7	0.03
2.181.246.0/24	8	0.03
2.181.247.0/24	9	0.04
2.181.250.0/24	1	0.00
2.181.253.0/24	1	0.00
2.181.255.0/24	1	0.00

Table 1.1: /24 prefixes with malicious activity from AS31549

1.2.1 Example

The difference in the level of granularity leads to a difference in the number of black listed IP addresses. For instance, let's analyze the AS AS31549. This ASN number belongs to Aria Shatel Company Ltd. Aria Shatel Company is an Internet Service Provider in Iran. Under this AS number, there are 230 IP V4 BGP prefixes announcement of different lengths. In total, 1,214,464 IP addresses are originated from this Autonomous system. According to our dataset, around fifty /24 prefixes that have a least one malicious IP address belong to this AS number. Seventeen of these /24 prefixes fall under one BGP prefix announcement (see Table 1.1). Dividing the number of malicious IP addresses within a prefix by the size of the prefix (which is 256) we get likelihoods of malicious activity between 0.0 and 0.0.04 with a median and mean of 0.01 for these seventeen prefixes. If we aggregate the reputation of these seventeen /24 prefixes by their advertised BGP prefix, which is 2.181.224.0/19, the resulted malicious activity likelihood of this BGP prefix after aggregation is 0.01.

By keeping the reputation of this BGP prefix not only we can flag a potential malicious traffic from the seventeen /24 prefixes from which we already observed malicious IP addresses but also we expand our prediction to the adjacent prefixes of these 17 prefixes. All these 32 prefixes with a malicious activity likelihood of 0.01 are represented in routing snapshots with one single BGP entry and one organization description name "Information Technology Company (ITC)". *Our hypothesis is that, these prefixes are administrated by one single entity and we should expect the same behavior for all the nodes.* We could also use the ASN

number 31549 and give a likelihood to all IP addresses that are originated from this AS. However, we only have malicious activity with likelihood of 0.01 for only two other prefixes of length 23 within this AS except 2.181.224.0/19; other prefixes have zero likelihood of malicious activity. Precisely, the likelihood for the AS would be $\frac{106}{1214464}$ which is epsilon. In other words, we would be penalizing 1214464 IP addresses for a malicious activity of only 106 IP addresses that mostly come from one BGP prefix. In conclusion, the underlying reason that the ASN is not granular enough is that although the AS31549 represents one ISP name, we observe around 20 different organization names that use the 256 announced prefixes. In summary, we can not expect a homogeneous behavior from a large entity such as AS but our hypothesis is that we can expect same behavior from /24 prefixes represented by a single BGP entry.

1.3 Research Questions

As already stated in the beginning of this chapter, our main research question is:

Which of the two aggregations based on BGP prefix and fixed /24 prefix is better for Internet bad neighborhood identification in terms of Intrusion detection and prediction error and latency?

As noted in the previous section, there are two lines of work; one line of works uses BGP prefix for aggregation while the other uses /24. BGP offers better false negative rate while /24 offers better false positive rate (see chapter 3 for mathematical proof). Furthermore, there will be fewer BGP entries in comparison to /24 that may impact the latency. However, the interplay of the above parameters and the tradeoff between them is unknown to us. As we explain in detail in chapter 2, none of the previous works investigated the effect of finer granularity of fixed /24 prefix in comparison to BGP on the above parameters. Qian et al compare the granularity of BGP aggregation with that of DNS based clustering. Moura et al compare the different prefix sizes but not BGP with fixed prefixes [16].

To answer the main research question of this thesis, the author first extensively reviewed the state of the art (see chapter 2). The state of the art mainly focused on spam filtering whereas we need to apply bad neighborhood for malware traffic detection. Based on the application of Internet bad neighborhood in our research, an ideal comparison would be based on false positive and false negative measurement. That said, measuring these metrics require access to network traffic and a reliable ground truth. Unfortunately, computer security domain has a tangible lack of ground truth [19]. In addition, due to privacy concerns, the author of this research could not collect any network traffic regardless of tiring efforts to address the privacy concerns of the officials.

Since measuring FP/FN is not feasible for the author, he investigates other aspects of BGP and /24 prefix aggregation that facilitate answering the main research question. The research questions that we follow in the rest of this work are:

1. *How different is the granularity of BGP aggregation in comparison to /24 fixed aggregation?*
2. *How much is the difference in the detection rate of BGP in comparison to /24 fixed aggregation?*
3. *How much is the difference in the lookup performance of BGP in comparison to /24 fixed aggregation?*
4. *Which aggregation feature can preciser predict the malice of a network?*
5. *Which aggregation can better identify bad neighborhoods based on the answers to the above questions?*

The granularity, detection rate and precision aim to measure the detection error and the lookup performance aims to measure the detection latency of an aggregation feature in Intrusion detection and prevention. In chapter 3, we precisely define granularity, detection rate, lookup performance and precision. For now, granularity measures the malice of a Internet neighborhood based on its size. Detection rate measures the number of malicious IP addresses that we detect. Precision measures the malice prediction capability of an aggregation feature. Lookup performance measures the processing time required for searching the reputation of an IP address.

1.3.1 Approach

To answer the above question, we run an experiment on a dataset of Indicators of Compromise. This dataset helps us identify malicious nodes and their concentration in some neighborhoods. Since we don't have an independent ground truth, our approach would be splitting the dataset to two training and testing sets based on different time periods. Although the ground truth problem can be addressed in this way, the lack of traffic still prevents measuring false negative and false positive rates. Due to the lack of network traffic and novelty of the application of badhood in our work, we have to define new metrics to further this research.

We first formally define aggregation and reputation and present a formal framework of metrics that allow the comparison in the context of our work. We formally define malice likelihood, granularity, precision, detection rate and lookup performance metrics (see Chapter 3) that can be practically used (based on the available data) for experimental analysis. We then collect the indicators of compromise from a Malware Detection Company database. Afterwards, we extract the IP addresses and build the Internet neighborhoods based on /24 prefix feature and BGP. By using the defined metrics and the provided data, the author measures reputation in form of malice likelihood, and compare the granularity, precision, detection performance and lookup performance of the badhoods reputation extracted from one month of data. Finally, we conclude which aggregation can better identify the bad neighborhoods for malware traffic detection based on the quantitative results.

1.4 Research Value

The findings in this research will improve the prediction capability of Intrusion Detection systems as explained in section 1.2. The findings answer whether BGP or fixed /24 prefix better identifies Internet bad neighborhoods. Intrusion Detection Systems hopefully can use our results to:

1. Improve the FN, FP rate of their products without much performance compromise.

2. Automate the process of generating aggregated blacklists

As a matter of fact, the author of this research conducted this research while doing Internship in the vendor company Redsocks [1]. Such collaboration helped the author to better identify the requirements of the vendors and the end users and consider them while comparing BGP with fixed /24 prefix.

1.5 Summary

In this chapter, we explained the motivation of our research topic and the research questions we will answer through the rest of this thesis. Furthermore, we explained how we aim to answer these research questions. In the rest of this thesis, in the chapter 2, we first review the related works and present the findings of the research topic assignment that has led to the research questions of this work. In chapter 3, we formally present the definitions and the metrics that would be our framework for the comparison and answering the research questions. In chapter 4, we present two reputation lookup algorithms that would be the base for our implementation performance comparison of the two aggregation approaches. In chapter 5, we elaborate on our data collection methodology and give an overview on the data that we will use for the experimental comparison. In chapter 6, we answer the first four research questions using the experimental comparisons on the data that we collect and based on the metrics that we define. Finally, in chapter 7, we answer the last research question and conclude the research.

[1] https://www.redsocks.eu/

Chapter 2

Literature Review

In this chapter, we review the state of the art literature that somehow uses reputation, hereafter as Internet Host Reputation Systems(IHRSs), for malicious detection. Internet host reputation systems output the reputation of a host based on its historical activities. The historical malicious activities, in practice, are mainly found in public blacklists based on IP or DNS address. We use the term *reputation oracle* for any medium that the state of the art refers to for the malicious activities of connected nodes to the Internet, hereafter as Internet hosts, in past. The data from the reputation oracle is the input to a *reputation function* in IHRSs. Different works use different reputation functions and algorithms for further processing. Eventually, the *reputation output* of a reputation system can be a trained classifier or a reputation database. The former is usually an engine that takes traffic and outputs malice based on a backend reputation oracle. The latter outputs the identifiers of malicious hosts. Different works use different identifier metrics, hereafter as *aggregation feature* such as IP or DNS to identify hosts. State of the art reputation systems choose various benchmarks to evaluate their results. Figure 2.1 shows how Internet Host Reputation Systems work in practice. This figure aims to visualize the Internet Host Reputation System's structure and the relevance of our terminology, i.e the characteristics we use to categorize the state of the art, to these systems.

The works that we choose to review in this study are selected in a systematic way. First, we searched in google scholar based on Reputation system, Predictive blacklisting, Network clustering, Proactive spam detection, Internet bad neighborhood and Internet host aggregation keywords. We removed the results older than 10 years unless the work has been a break through. Next, we selected the works that have been cited more than 10 times. Afterwards, we shortly read the papers and selected the most relevant works to our research. In the rest of this section, we compare the literature based on *reputation oracle*, *reputation function*, *reputation output*, *aggregation feature*, and *Benchmarking*.

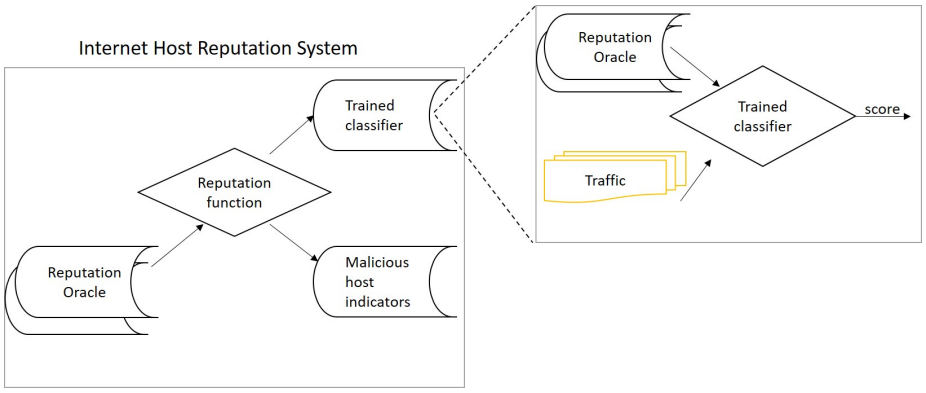


Figure 2.1: Internet host reputation systems

2.1 Reputation Oracle

In Internet host reputation systems, the reputation is measured based on previous activities of hosts. We use the term reputation oracle to refer to a database that stores the reputation of Internet hosts based on their malicious activities in past. A majority of the literature use public blacklist datasets such as SURBL[20], SBL [21], CBL [22], XBL [23], Spamcop [24], Malware Domain List [25], DNS-BH [26], Zeus tracker [27], JWSDB [28], URIBL [29], SORBS [30], DSheild [31], Phishtank [32], hpHosts [33], OpenBL [34], VERIS [35], WHID [36], WPBL [37], Support Intelligence [38], UCEPROTECT[39], APWG [40], Viruswatch [41], Malware Patrol [42] and Bot Command and Control IP addresses from ShadowServer Foundation [43]. A few works use Malware generated traffic in a controlled environment or the traffic to a Honeypot [6][18] or Spamtrap [4][14][13] as their reputation oracle. These works record the Internet host identifier (IP, DNS name or URL) that the malicious program tries to connect to. Finally, a few works use the reputation computed by another product, such as IDS, to track the reputation of a host [44].

The nature of the datasets and how data is collected is different. Several datasets are providing the IP and DNS of spammers: SBL [21], CBL [22], [24], JWSDB [28] and SORBS [30]. Some other datasets identify phishing websits: SURBL[20], Phishtank [32] and APWG [40]. Datasets such as Zeus tracker [27] and Bot Command and Control IP addresses from ShadowServer Foundation [43] identify Botnet's command and control servers. DSheild [31] provides the attackers' addresses based on firewall and IDS logs.

Different studies use different datasets based on the aim of the study and the type of malicious hosts the dataset would reveal. Nonetheless, there is a tradeoff between malicious detection capability and detection latency while choosing various datasets. Increasing the dataset size would possibly increase malicious detection performance but at the same time it would increase the processing time. Table 2.1 reports the datasets and approaches that different

Dataset	The literatures that used the dataset as their reputation Oracle								
[1]	[45]	[46]	[6]	[47]	[15]	[48]	[4]	[14]	[13]
SURBL					✓	✓			
SBL			✓		✓	✓		✓	✓
CBL						✓		✓	
XBL					✓				
Spamcop						✓		✓	
Malware Domain List			✓						
Zeus tracker			✓						
JWSDB				✓					
URIBL				✓					
SORBS								✓	
DSheild	✓	✓				✓			
Phishtank					✓	✓			✓
hpHosts						✓			
OpenBL						✓			✓
UCEPROTECT						✓			
VERIS						✓			
WHID									
WPBL						✓			
Malware Domains			✓						
Support Intelligence					✓				
APWG					✓				
Viruswatch					✓				
Malware Patrol					✓				
ShadowServer					✓				
Honeypot traffic			✓						
Spamtrap							✓	✓	✓

Table 2.1: Datasets used by literature

works use. We note that few studies use the same datasets.

2.2 Reputation Function

Reputation function computes the reputation of an Internet host based on some historical features of the sender. Many of the state of the art works use classifiers in order to compute the reputation based on a set of features. [5] [14] [4] [13] are simple linear classifiers that label malice based on a malicious activity threshold. The threshold is usually based on a number of historical malicious activities e.g. the number of sent spams, the number of blacklists that list an IP address or DNS, the number of IP addresses in a network that are blacklisted, or spam ratio i.e. the percentage of spams to total sent emails. Several works use sophisticated Machine Leaning (ML) classifiers and label maliciousness based on an extensive set of features [44][6][48]. [44] uses RuleFit classifier and uses 13 network and traffic related features such as message length and time of the day for classification. [6] uses Logit-Boost strategy (LAD) decision tree based on 16 domain related statistical features. These features span from number of related IP addresses and domains to the number of related malicious domains

[1] The authors of the works, in order, are Soldo et al, Zhang et al, Antonakakis et al, Felegyhazi et al, Shue et al, Liu et al, Venkataraman et al, Qian et al, Moura et al

literature	category	algorithm	Number of features
Soldo et. al. [45]	Recommendation	Exponential Weighted Moving Average - Cross Validation (CA) clustering algorithm - K-Nearst neighbors	2
Zhang et. al. [46]	Recommendation	similar to Google's page rank algorithm	1
Hao et. al.[44]	Classification	RuleFit	13
Antonakakis et. al. [6]	Classification	Logit-Boost strategy (LAD	16
Liu et. al. [48]	Classification	Random Forest classifier	256
Felegyhazi et. al.[47]	Clustering	Manual	~10
Shue et. al. [15]	Classification	linear classifier	1
Venkataraman et. al. [4]	Classification	linear classifier	1
Qian et. al. [14]	Classification	linear classifier	1
Moura et. al. [13]	Classification	linear classifier	1

Table 2.2: Reputation function characterization of state of the art

and lexical features of the domain name. [48] uses Random Forest classifier and a list of 258 features that are mainly related to mismanagement issues such as open resolver in the network or mis-configured HTTPS certificate.

Other works use recommendation ML algorithms to output the reputation of an Internet host [46][45]. These systems aim to build a customized blacklist for a victim. Their input is a matrix that shows which attackers attacked which victims. They aim to find the most relevant attackers to a victim. While [46] only searches a two-dimensional matrix to find the relevance of different attackers to a victim, [45] also takes into account the temporal behavior of the attacks and searches in a 3-dimensional space. Table 2.2 reports the reputation function category, the algorithm and the number of features that different works use.

2.3 Reputation output

Internet Host Reputation Systems (IHRSs) usually output either of two general artifacts: blacklist; or a trained classifier. Regardless, the state of the art may use the blacklist or the classifier in a detection engine later on. For instance, [4, 14] deploy the result of reputation system they develop in SpamAssassin. Many public IHRS and state of the art works output a blacklist. Other IHRSs based on reputation usually output a trained classifier that may be used in a malicious detection engine. Detection engines that solely use blacklists usually have a less detection latency in comparison to the trained classifiers that work with multiple features because the processing would be limited to only querying the reputation of the node from the blacklist. On the other hand, the trained classifier output of an IHRS usually has a better detection performance accuracy; similar to solving a crime case by law enforcement, more evidences facilitate a better decision. In summary, detection engines based on blacklist artifact have lower detection performance accuracy while maintaining shorter detection latency in comparison to trained classifiers.

A blacklist would assign reputation to an Internet host based on its address

output artifact	The literatures that output such artifact										
[2]	[47]	[14]	[13]	[48]	[45]	[46]	[15]	[6]	[44]	[4]	[5]
blacklist	✓	✓	✓		✓	✓	✓			✓	✓
trained classifier				✓				✓	✓		

Table 2.3: Internet host reputation systems taxonomy based on output artifact

whereas a classifier is trained based on many attributes, including the Internet host address. In addition, a compiled blacklist is independent of the final implementation and it can be used in numerous systems e.g. firewall, IDS etc; however, a trained classifier can hardly be used anywhere else except in the designed detection engine.

A majority of state of the art output a reputation blacklist [47, 14, 13, 45, 46, 15, 4, 5]. [6, 44, 48] train a classifier based on a reputation oracle. [6] and [44] use the trained classifier in a detection engine while [48] uses the classifier to predict future cyber incidents. In addition to the previous works, there are few researches that can not be strictly characterized in one category. Table 2.3 reports the characterization of the literature based on the decision time.

2.4 Aggregation feature

Reputation is assigned to an Internet host based on a metric by which the host can be distinguished from the others. We use the general term aggregation feature to such host identifier even when there is no aggregation in practice. In such case, we consider aggregation size equal to one. Reputation is commonly assigned to Internet hosts based on *IP address* [13] [45][46][4][14]. Except [46], the rest use an aggregation of IP addresses based on IP prefix. One popular alternative to IP address is *DNS address*. There is usually a mapping between DNS host name and IP address through A record and vice versa through DNS PTR record. Since IP addresses are often dynamically assigned, DNS name sometimes maintains more stability. [6] and [47] use DNS records. [14] uses reverse Domain authoritative reverse DNS server (rANS) and reverse DNS(rDNS) as aggregation feature. In addition to DNS, *Autonomous System Number (ASN)* can be used as an aggregation feature. The mapping between IP address and ASN is not usually intuitive; the routing BGP broadcasts need to be examined for the mapping. [15] and [49] use ASN number as aggregation feature. Finally, some works use Geo location of an Internet host such as country or city to assign reputation [50, 18].

Aggregation metric may represent an individual host or a group. IP address, and DNS A records represent an individual host while the rest of the indicators represent a group of hosts and the assigned reputation applies to all the group's

[2]The authors of the works, in order, are Soldo et al, Felegyhazi et al, Qian et al, Moura et al, Liu et al, Soldo et al, Zhang et al, Shue et al, Antonakakis et al, Hao et al, Venkataraman et al, Wanrooij and Pras

Literature	IP	DNS	ASN	Geo location	Organization	Aggregated	Network-Aware
Soldo et. al.[45]	✓					✓	
Zhang et. al.[46]	✓						
Antonakakis et. al.[6]		✓					
Felegyhazi et. al.[47]		✓					
Shue et. al.[15]			✓			✓	✓
Van Polen et. al.[18]				✓		✓	
Venkataraman et. al.[4]	✓					✓	✓
Qian et. al.[14]	✓					✓	✓
Moura et. al.[13]	✓					✓	
Liu et. al.[48]					✓	✓	✓

Table 2.4: IHRSs taxonomy based on aggregation feature

members. Aggregating the Internet nodes based on a metric may be carried out in different ways. For instance, IP prefix is a common method to aggregate Internet nodes. That said, the IP prefix based on an IP addresses can be calculated either statically and based on a fixed prefix or dynamically and based on BGP prefix. The latter is called a network aware cluster because it represents a network in real world. Several studies suggest that network aware clusters have a fine granularity, and hence the precision derived from these clusters is acceptable [4, 14, 51]. On the other hand, granularity increases the entries' size that need to be stored, and hence the detection latency may increase.

Table 2.4 reports the result of categorizing the state of the art based on their aggregation feature. We note that systems which output a trained classifier are not mentioned in the table. The reason is that these works, as already mentioned, are taking a holistic approach; they don't use only one metric to track the reputation. For instance, [6] uses both the IP address prefix and the Domain name of a host for training the classifier.

2.5 Benchmarking

The literature use different benchmarking to validate the proposed reputation system. The first category of works use their reputation oracle to validate their results. In other words, they split the data from reputation oracle to two sets: training data set; and the testing data set. The testing data set, used for validation, contains the malicious activity records of later time, hereafter as *future versions* of blacklist. For instance, the records of SBL from the October of one year may be used for detection, and the records of November of the same year may be used for validation. [47] reports false positive and true positive based on presence or absence of the predicted malicious domains in JWSDB, URIBL and McAfee SiteAdvisor. Sophisticated classification solutions follow a similar approach but in a more systematic way. [44, 6, 48] use multi fold cross validation to report false positive and true positive. The second category of works use another well-known malicious detection system to compare their results. [14, 5, 4] use SpamAssassin as the ground truth to evaluate their system. What they report is basically the improvement they obtain by employing their compiled list in SpamAssassin. For instance, Qian et. al. report that their

	Ground truth	Accuracy metric	Measurement method	TP/FP value	compared work
[45]	Reputation Oracle	Hit Rate	experimental analysis	———	[46]
[46]	Reputation Oracle	Hit rate	experimental analysis	———	None
[44]	Reputation Oracle	TP/FP	10-fold cross validation	70%/0.3%	None
[6]	Reputation Oracle	TP/FP	10-fold cross validation	96.8%/0.38%	None
[48]	Reputation Oracle	TP/FP	10-fold cross validation	90%/10%	None
[47]	Reputation Oracle	TP/FP	cross validation	75%/5%	None
[14]	SpamAssassin	FN/FP	experimental analysis	10%/1% [3]	None
[13]	———	information lost - error	mathematical calculation		None
[5]	SpamAssassin	FP/FN	experimental analysis	———	None
[4]	SpamAssassin	server goodput	experimental analysis	———	None

Table 2.5: Benchmarking of the state of the art

aggregation method improves the false positive rate of SpamAssassin by 50%. Venkataraman et. al. report the percentage of the Spams that were detected when the server was overloaded. Such measurements are not comprehensive enough to compare the malicious detection performance of their underlying reputation systems. The third category, Recommendation-based solutions, do not report based on FP and NP; they concentrate on hit count [45][46]. Hit count is calculated based on the number of blacklist entries that are seen in the actual traffic. The reason to choose this measurement is the goal of such works i.e. optimizing the size and effectiveness of the public blacklists. Table 6.1 reports the characterization of the literature based on their benchmarking.

2.6 Conclusion

State of the art's findings show intrusion detection based on reputation of Internet hosts is effective. Statistical analysis of attackers' IP addresses show that attacks are likely to happen from the same malicious networks. Hence, clustering Internet hosts based on an aggregation features leads to detecting more attacks i.e. reducing False Negative rate and reducing the size of blacklist. That said, aggregation increases False Positive(FP) while reducing False Negative(FN) based on the level of granularity.

Various aggregation features have different levels of granularity i.e. different members and group sizes. IP prefix aggregation can be based on either a fixed size or announced IP BGP prefixes, and these two have different granularities. The difference in granularity results in different detection performance (in terms of FN/FP) and detection latency. Although we can hypothesize about either detection performance or detection latency advantage of one aggregation feature solely, we can not speculate about the tradeoff of these two metrics and to extent which that granularity affects these metrics.

State of the art lack a comprehensive comparison of the different aggregation features in intrusion detection in terms of detection performance and latency. According to Tab.6.1, few works use standard accuracy metrics in their work.

[3] The reported accuracy in this work differs based on threshold and the aggregation feature. Our reported value is an approximation of the optimum balance of FP/NP of DNS and IP BGP prefix clusters

Moreover, each work reports the measurement for one specific aggregation feature; an exception is [14] that investigates the difference in granularity of IP BGP prefix aggregation vs. DNS in Spam detection. Furthermore, the state of art rarely considers also the detection latency effect of an aggregation feature. Finally, the benchmarking in various works is different, making it harder for comparing one to another.

In a research, following this proposal, we will investigate if the loss of granularity by using BGP prefix aggregation in comparison to /24 fixed prefix aggregation is significantly different. Furthermore, we will investigate the detection performance, the reputation precision and the implementation performance of the two aggregation features.

Chapter 3

Problem Formulation and Definitions

In the previous chapters, we explained that generalizing the reputation of the IP addresses within an BGP prefix is a common approach. Such generalization is based on the way that BGP prefixes are managed; it is expected that nodes within a BGP prefix are managed similarly, and they together form a uniform behavior. That said, BGP prefixes have different lengths and hence the granularity of the assigned reputation will be different. Decrease in the level of granularity, attributing reputation to a larger number of nodes, will affect false positive, false negative and the implementation (or processing) performance of the reputation aggregation. The finest granularity of a BGP prefix is for the longest length that is 24. Henceforth, aggregation based on fixed 24 length to achieve high granularity is an alternative to BGP aggregation. Yet, the advantage in doing so, in terms of measurable detection and implementation performance metrics is unclear.

The objective of this chapter is to define some formal metrics that allow the comparison of BGP and /24 fixed prefix aggregation. In chapter 1, we explain that Redsocks Security, and possibly other similar vendors, is concerned about False Positive (FP), False Negative(FN) and Performance Penalty effect of a chosen aggregation feature. However, measuring FP and FN of an aggregation feature highly depends on the ground truth and the observed traffic. That said, both ground truth and traffic are rare commodities in the Network Security domain. Abt et al extensively discuss the lack of ground truth for security researches in [19].

The main problem to obtain traffic and hence ground truth is the concern about privacy. The author of the current work and the supervisors spend months of negotiation with several Dutch organizations to access network traffic and use an IDS to assign labels. Although such method would have its own limitations, it could have been a starting point in constructing a ground truth. The author and the supervisors even proposed a technical solution to the privacy concern

of the correspondents. Yet, no party gave us access to the network traffic data. Henceforth, we are not able to compare FP and FN of the aggregation features, and in the rest of section, we define metrics that we can employ using our available data.

In this chapter, we formally formulate the terms and the metrics that we will use for our comparison analysis in chapter 6. These metrics are *Precision*, *Detection Rate*, *Granularity Delta* and *Lookup Performance*. We measure these metrics based on Likelihood (see section 3.1) metric that we define. In section 3.2, we define the basic terms that we will use in the rest of this paper. In section 3.3, with recourse to the basic definitions, we formally define above metrics that allow us compare the two aggregation features and answer the research questions we earlier presented.

3.1 Likelihood

Measuring the malicious activity of a network is a subjective task. In case of spamming, spam ratio measurement is a common way in the state of the art as mentioned in the previous chapter. In case of phishing, the number of malicious URLs or domains could be an effective metric. In our case, case of C&C and bot detection, the focus is the number of malicious Internet nodes. Since there is a relation between IP and Internet nodes, there may be one or multiple nodes corresponding to one IP address, we base our measurement on the number of malicious IP addresses in a network. That said, the number itself is not representative considering the variable size of BGP prefixes. For instance, 16 malicious hosts in a prefix of length 24 does not represent the same amount of malicious activity comparing to a prefix of length 22. Henceforth, we define likelihood of malicious activity that takes into consideration the size of a prefix.

This likelihood, later, will be used as one of the features f_i in $V = \{f_1, f_2, ..., f_n\}$ to represent the malice of the network where the IP originates. The larger the likelihood, the higher the chance that the traffic is malicious. The reader shall note that we do not see badhood in a black and white manner, either the network of the IP address is safe or not. Our measurement is independent of the wight of the malice likelihood in the vector of features used for malware traffic detection. In other words, discussing the weight of this feature in comparison to other features is out of the scope the current work. That said, we give an example of using likelihood for malice detection. We assume that network x has a malice likelihood of 60%. If there is an outgoing traffic to this network and the connection is uploading a data of size 1GB, there is a high chance of data exfiltration by a malware. Such detection accuracy can be enhanced by other features, such as the country that the traffic flows to, the lexical characteristics of the domain and the URL and etc.

3.2 Basic definitions

In this section, we first present the basic definitions. In the end of this section, we further clarify the definitions via an example. We start our definitions from the aggregation feature labels. We define two constants BGP and $Fixed$ that will be used as labels and indexes to successively reference to aggregation based on BGP and fixed prefixes. We define a set of IP indicators of compromise as:

$$I_{s,t} = \{i_1, i_2, i_3, ..., i_n\}$$

s represents the start date of collecting indicators and t represents the number of days that the collection lasted. I, in simple terms, represents our dataset of indicators of compromise. We define an aggregation feature value domain D^f as the the set of all the values that aggregation based on a feature, $F(x)$ can have. We define $|.|$ operator that outputs the size of a set. Formally speaking:

$$D^f = \{a_1, a_2, a_3, ..., a_n\},\ f \in \{BGP, Fixed\},\ |D| = n$$
$$F(x) \in D^f$$

We define the set of all the prefixes derived from an indicator set by $A^f_{s,t}$:

$$a_i \in A^f_{s,t}\ only\ if\ \exists x \in I_{s,t}\ such\ that\ F(x) = a_i$$

A prefix a_i represents a group of IP addresses. We name this set U_{a_i}. The size of U_{a_i}, $|U_{a_i}|$, depends on the length of prefix. For Fixed prefix, the prefix length is always 24, however, for BGP the length can be anything greater or equal to 8 and less than or equal to 24. Given a prefix size l for prefix a:

$$|U_a| = 2^{32-l}$$

Similarly, we define set M_{a_i}:

$$x \in M_{a_i}\ only\ if\ x \in I\ and\ F(x) = a_i$$

We define the likelihood of having malicious activity from an IP address x by $P(x)$. This likelihood can be different in different times t since Internet hosts may be infected and cleaned. Henceforth:

$$P(x) \approx \sigma(x, t)$$

$\sigma(x, t)$ is a function that is unknown to us. By aggregation, we generalize the reputation of few hosts in a group to th whole group. Formally speaking, we are modeling $\sigma(x, t)$:

$$P(x) \approx P_{s,t}(a)\ such\ that\ F(x) = a$$

$P_{s,t}(a)$ is the probability that an IP address from a prefix a is malicious based on our training data collected from s up to t days. For the sake of simplicity, we refer to $P_{s,t}(a)$ by simply using $P(a)$ notation unless we strictly say otherwise. We define score of prefix a as S_a which is simply $|M_a|$, and we define $P(a)$ based on S_a:

$$P(a) = \frac{S_a}{|U_a|}$$

3.2.1 Example

In order to clarify the notations, we give an example using all the notations. We assume our dataset has the following entries and it has been collected from 2017-Mar-18 for 7 days:

$$I_{2017-Mar-18,10} = \{15.14.13.10, 15.14.13.11, 37.3.0.1, 156.147.2.1\}$$

D^{Fixed} in this case is all the following entries:

$$D^{Fixed} = \{0.0.0.0/24, 0.0.1.0/24, ..., 255.255.255.0/24\}\ |D^{Fixed}| = 16777216$$

D^{BGP}, however, depends on the BGP announcements between 2017-Mar-18 and 2017-Mar-25. An interesting reader can refer to [52] in order to download the appropriate dataset to generate D^{BGP}. $F(x)$ in case of fixed aggregation is a simple masking with `0xffffff00` value. In case of BGP, $F(x)$ should be implemented by checking the routing snapshots of the given period. Below is the result of aggregation:

$$A^{Fixed}_{2017-Mar-18,10} = \{15.14.13.0/24, 37.3.0.0/24, 156.147.2.0/24\}$$

$$A^{BGP}_{2017-Mar-18,10} = \{15.0.0.0/8, 37.2.0.0/15, 156.147.0.0/16\}$$

$$M_{15.14.13.0/24} = M_{15.0.0.0/8} = \{15.14.13.10, 15.14.13.11\}$$

$$M_{37.3.0.0/24} = M_{37.2.0.0/15} = \{37.3.0.1\}$$

$$M_{156.147.2.0/24} = M_{156.147.0.0/16} = \{156.147.2.1\}$$

Based on the above data, $P(15.14.13.0/24)$ is $\frac{2}{256}$. This probability means that if we observe traffic from any IP address that its masking with `0xffffff00` is 15.14.13.0/24 there is $\frac{2}{256}$ chance that this traffic is malicious. In contrast, with BGP aggregation, the probability of having malicious activity from any equivalent IP address is $\frac{2}{65536}$. The reader should notice that our example is not representative of the real world since we are only considering a dataset of only 4 IP addresses; we only gave this example to clarify the definitions and not to compare Fixed with BGP aggregation.

3.3 Metrics definition

In the following paragraphs, we define four metrics that aim to answer the first four research questions.

3.3.1 Granularity Delta

In order to answer Research Question (RQ) 1, we define Granularity Delta metric. By definition, fixed aggregation is more granular than BGP; the reputation derived from fixed aggregation is attributed to a smaller group of hosts. In other

words, we have more likelihood records in our database and this can give us a more granular view of the Internet host reputations. Formally speaking:

$$|A_{t,s}^{BGP}| <= |A_{t,s}^{Fixed}|$$

The above relation exists because all advertised prefixes in wild are less than or equal to 24, and all the prefixes in the Fixed approach are exactly 24. Although the fixed aggregation is more granular than BGP, the probabilities distribution in BGP can still neutralize the effect if:

$$P_{s,t}^{Fixed}(a) = P_{s,t}^{BGP}(a) \forall a \in A_{t,s}^{Fixed}$$

However there is not guarantee that all prefixes a exist in $A_{t,s}^{BGP}$. As a matter of fact, we first ensure this is not the case; otherwise, comparison of BGP and fixed prefix is meaningless since $A_{t,s}^{Fixed} = A_{t,s}^{BGP}$.

Nevertheless, $P_{s,t}^{BGP}(a)$ can still be derived from an a' that encompass the IP addresses that a encompass. This is because, by definition, we generalize the reputation of M_a to $\forall x \in U_a$. Since a' encompasses a, $U_a \subset U'_a$. Henceforth, the reputation of U_a members is represented by the reputation a' and $P(x)$ $x \in U_a$ is represented by $P(a')$.

In order to clarify the concept, assume that we want to lookup the reputation of 9.50.10.1 based on aggregation. In fixed approach, we need to lookup the maliciousness probability of 9.50.10.0/24 in our database. Assume that based on the routing snapshots 9.50.10.1 belongs to 9.50.10.0/23 prefix. In the BGP approach, in order to lookup the probability of 9.50.10.1 we need to look for 9.50.10.0/23 entry. If $P^{Fixed}(9.50.10.0/24) = P^{BGP}(9.50.10.0/23)$, there hasn't been indeed any granularity loss. This can happen if the adjacent \24 prefixes that comprise a BGP prefix have almost the same probability of maliciousness. For instance, in the former example, if 9.50.10.0/24 and 9.50.11.0/24 both have probability y, then $P^{BGP}(9.50.10.0/23)$ would also have the probability of y.

In order to compare the granularity of BGP and fixed aggregation, and answer RQ 1 we investigate if the above phenomenon exists i.e. the adjacent /24 prefixes based on the BGP view have the same probabilities. To perform such analysis, we define $\Delta(a)$, read as *granularity Delta of prefix a*:

$$Delta(a) = P^{Fixed}(a) - P^{BGP}(a)$$

We compute $P^{BGP}(a)$ in the same manner that we explained in the above paragraphs. We then analyze the distribution of *Delta* and analyze its statistical characters.

3.3.2 Detection Rate

In order to answer RQ 2, we define detection rate metric. We define the hits as the number of records in $A_{s,t}^f$ that also appear in $A_{s',t}^f$. Based on this definition:

$$J = A_{s,t}^f \bigcap A_{s',t'}^f \quad Hit(A_{s,t}^f, A_{s',t'}^f) = \sum_{i=1}^{|J|} |M_{a_i}| \;\; \forall a_i \in J$$

Similarly, we define *Detection Rate*:

$$Detection\ Rate = \frac{Hit(A^f_{s,t})}{|A^f_{s',t'}|}$$

In simple words, this metric shows the ability of our prefix set based on an aggregation to detect the malicious IP addresses. Of course this metric does not say anything about the confidence of the prediction; detection, here, simply means that we can have an estimation on the probability of having malicious traffic from a prefix. In order to better understand the nature ot the detection that a prefix set may have we define Cumulative Distribution Function(CDF) of Hits based on the malice probability x:

$$CDF(x) = \sum_{i=1}^{n} |M'_a|\ \forall a\ s.t.\ P(a) <= x$$

CDF graph gives us an insight about the probabilities that we report for the hits and our confidence about the malice chance. This insight will help the designer of a malicious detection solution to better understand how to employ the aggregated reputation and what to expect from the reputation.

By the detection rate metric, we aim to see the potential of each aggregation feature to identify the entire malicious IP addresses. In this regard, it is expected that BGP identifies more malicious IP addresses since:

$$\forall a \in D^{Fixed}\ , \forall b \in D^{BGP}, U_a \subset U_b$$

The above relation holds because the derived BGP prefixes from our IP indicators set are either of size 24 or larger size that encompass the /24 prefix. Since the IP space that a BGP aggregated reputation database covers is always bigger than its Fixed /24 counterpart, the Detection Rate of BGP is always equal to or greater than /24 Fixed aggregation. Larger IP space coverage, however, has a downside; the chance of having higher false positives increases by an IP space coverage growth. That said, it is the tradeoff between the Detection Rate and the space growth that can justify the usage of BGP. If the Detection Rate of BGP is meaningfully different, we can use our probability metric to signify a chance of malicious activity and expect other malicious detection features to distinguish false positives from the true positives. Otherwise, the Fixed aggregation is preferred since it will have smaller false positive rate.

3.3.3 Lookup performance

In order to answer RQ 3 and compare the implementation performance of the two aggregation features we define two lookup performance metrics. Firstly, we compare the processing overhead of each aggregation feature based on the lookup algorithm *Order value*. We present two lookup algorithms that are independent of the employed aggregation feature in chapter 4. Yet, since the number

of prefixes derived from each aggregation is different, $O(x)$ that is dependent on the number of records can be different. For instance, if the lookup algorithm has $O(n)$ the performance would be two times more if n becomes $\frac{n}{2}$.

Secondly, we compare the *Footprint* of each aggregation feature. Again, footprint depends on the implementation of the lookup algorithm and the underlying data structure; we take this into consideration and base our measurement on the two algorithms that we present in the chapter 4. We report the footprint based on the number of records and the size of the lookup algorithm data structure in Bytes on disk and in memory.

3.3.4 Precision

In order to answer RQ 4, we define precision metric that measures the capability of the prefixes likelihood metric in predicting malice of a network. In order to measure precision of BGP and fixed aggregation, we split our datasets to two subsets: training; and testing. Our training set spans from date s to $s+t$ and our testing set spans from s' to $s'+t'$ such that $s' > s+t$. We, then, measure the likelihood of malice for the observed prefixes in the training set and the testing sets. Afterwards, we extract $(P^f_{s,t}(a), P^f_{s',t'}(a)), f \in \{BGP, Fixed\}$ points from the prefixes $a \in J$:

$$J = A^f_{s,t} \bigcap A^f_{s',t'} \quad f \in \{Fixed, BGP\}$$

$P^f_{s,t}(a)$, hereafter as $P(a)$ unless stated otherwise, measures the probability that an IP address from prefix a sends malicious traffic based on our training dataset. $P^f_{s',t'}(a)$, hereafter as $P'(a)$ unless stated otherwise, measures the probability that an IP address from prefix a sends malicious traffic based on our testing dataset. In simple words, $P(a)$ is our predicted probability and $P'(a)$ is our observed probability. After constructing J, we plot it to examine the relation between P and P'. We then investigate if there is any correlation through *regression modeling*. We compare the precision of BGP and Fixed aggregation based on the standard error.

3.4 Conclusion

Aggregation based on /24 fixed prefixes lead to a subset of BGP aggregated prefixes. This inevitably leads to a more granularity of fixed aggregation in comparison to BGP prefix aggregation. On the other hand, since BGP prefix aggregation covers a larger portion of IP space the false negative rate of BGP is equal or less than /24 fixed aggregation. That said, we can not directly measure false negative or false positive rate because our efforts to obtain a reliable ground truth and traffic were impeded by privacy concerns.

Based on the available data (see chapter 5), we define several metrics in this chapter to measure the detection and implementation performance difference of BGP and /24 fixed aggregation. Firstly, we define Granularity Delta metric

to see how much /24 fixed aggregation is more granular. Secondly, we define Precision and Detection Rate metrics to compare the detection performance of the aggregation features. Finally, we define Order Value and Footprint metrics to compare the implementation performance of the aggregation features.

Chapter 4

Aggregated Reputation Lookup Implementation

In the previous chapter, we explained that implementation performance comparison of the two aggregation features depends on the reputation lookup algorithm. In this chapter, we present two algorithms that are independent of the reputation aggregation method; they only depend on the size of prefixes database. After compiling the reputation of a group of IP addresses, clustered based on a Fixed or BGP size, the database will be used for reputation lookup of single IP addresses.

Our implementations in this chapter are independent of the aggregation feature that we use to build the reputation database. In other words, we assume that the reputation lookup algorithm is unaware of the reputation compilation algorithm. In order to achieve this, we abstractly assume that the database keeps the reputations in a $NetID, PrefixLength, Probability$ tuple format. For instance, 2.16.196.0/23 prefix with a malice probability of 79% is stored as $34653184, 23, 0.79$. Therefore, in the Fixed aggregation all the entries in our database have length /24 while for BGP the length can be anything between 8 and 24. The difference in the length of prefixes leads to a difference in the number of stored records. We explain in this chapter that there is a tradeoff between the lookup performance and the footprint of the lookup implementation.

In the rest of this chapter, in Section 4.1, we present an algorithm that can do the lookup in $O(1)$ regardless of the aggregation compilation feature. In Section 4.2, we present another lookup algorithm that can do the search in $O(log(n))$ but is significantly more efficient from footprint perspective.

4.1 Indexing Search Algorithm (ISA)

The IP reputation lookup can be implemented in a fast manner via indexing. Since IP space can be presented as a finite series, 2^{32} members, we can exploit this feature for indexing. We may use the IP address integer value as an offset

to a memory location. This memory location would store the reputation of the IP addresses. As prefixes also can be represented by integer, we can split the IP space to the prefixes of same length. Given a prefix length of l, the IP space would be split to 2^l chunks. We assign a reputation to each of these chunks and then later we can query the reputation of an IP address by finding its corresponding chunk. This approach is to implement an ideal hash table. Following hash table concept, the chunks are indeed the defined buckets in a Hash table and the hash function is a simple division of the IP by 2^{32-l} value; the reader shall note that this is an ideal hashing and each prefix is assigned to a unique bucket.

4.1.1 Data Structure

As mentioned in the previous section, our data structure is an ideal hash table. We simply use an array (see Figure 4.1). This array has 2^{24} entries; we split the IP space to buckets of length 256, or prefixes of size 24. Each element stores the reputation of the corresponding /24 Net block i.e. the start of the array address plus the result of dividing IP address with `0xff`. For instance, the reputation of 0.0.5.57 IP is stored in the 6 element of the array. For the /24 entries that we haven't recorded any malicious activity, we store 0. This value is logically correct since based on historical records, there is 0 chance that an IP from this space exposes malicious activity.

4.1.2 Initialization

In order to materialize the Indexing Search algorithm, we first need to initialize a byte array of size 2^{24} with the corresponding probabilities of the indexes i.e. /24 prefixes. Since the reputation database may contain larger net block entries than /24, $l < 24$, we first needs to process the reputation database entries and split large net blocks to /24 entries. This processing is done using Algorithm 1. The inputs of this algorithm as mentioned earlier in this chapter come from the aggregated reputation compilation. After that, Algorithm 2 initializes the Indexing array in the memory. We use by default one byte to store the reputation of a net block. In order to realize storing a float as a byte, we multiply the probability by 256. If this precision is not sufficient, the

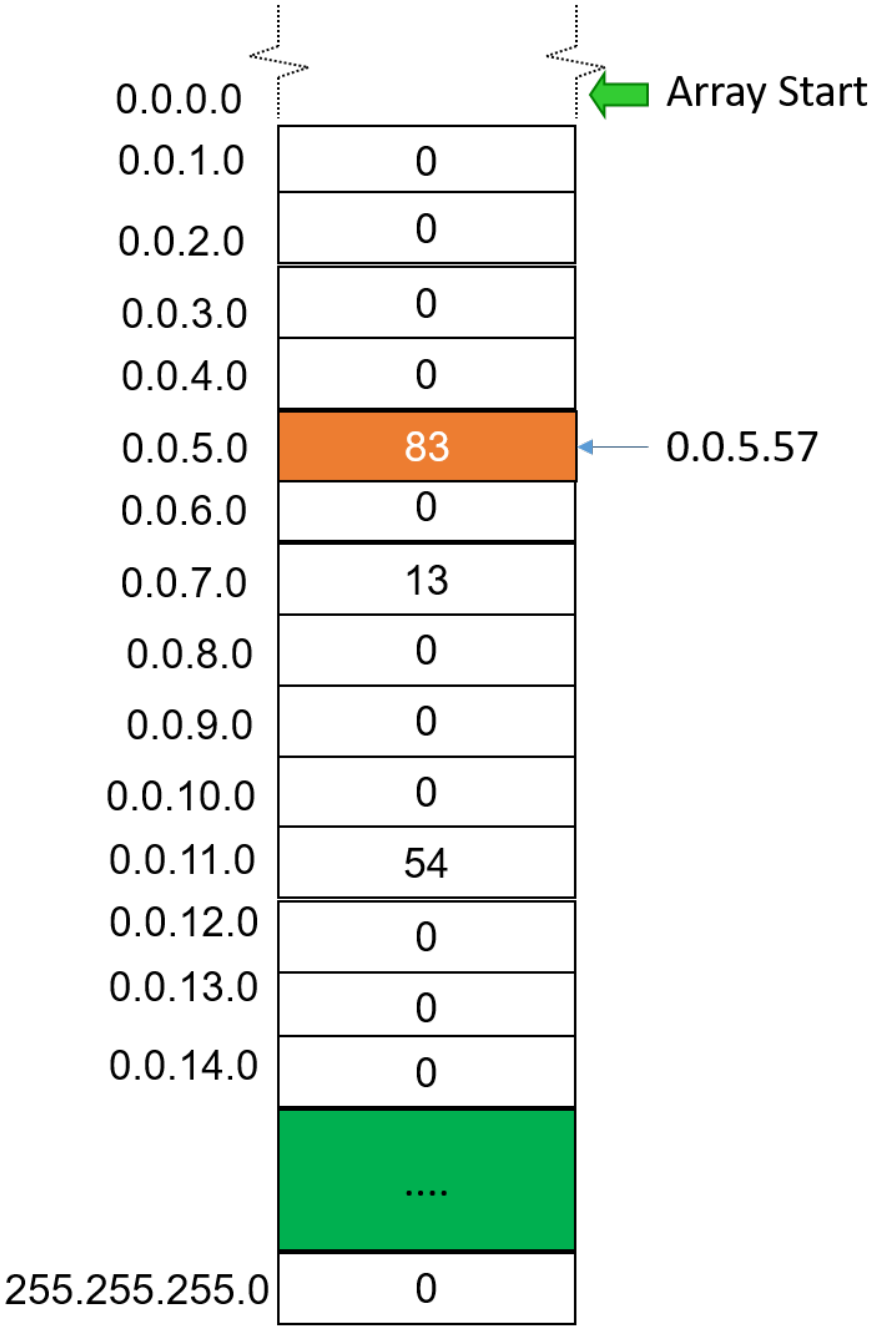


Figure 4.1: Array data structure to use for O(1) algorithm

reputation can be stored as a float; the footprint, however, increases as a result.

Algorithm 1: Covert a reputation database with entries' prefix larger than 24 to 24

```
Data: Three arrays of NetIDs, Lengths and Probabilities
Result: Two derived lists of Fixed_NetIDs and Fixed_Probabilities
Fixed_NetIDs:=List();
Fixed_Probabilities:=List();
for i:=1 to len(NetIDs) do
    start_NetID = NetID[i];
    Num__Of_Prefixes = 2^(24−Lengths[i]) ;
    for j:=1 to Num__Of_Prefixes do
        m:= (j-1) * 256;
        Fixed_NetID := start_NetID + m;
        Fixed_NetIDs.append(Fixed_NetID);
        Fixed_Probabilities.append(Probabilities[i]);
    end
end
```

Algorithm 2: Initialize the Indexing Array with the corresponding probabilities

```
Data: Two lists of Fixed_NetIDs and Fixed_Probabilities
Result: Array Indexed_NetIDs with corresponding probabilities
Indexed_NetIDs := Byte[2^24];
for i:=1 to len(Indexed_NetIDs) do
    Indexed_NetIDs := 0;
end
for j:=1 to len(Fixed_NetIDs) do
    PrefixID := Fixed_NetID.ElementAt(j);
    Index := PrefixID/256;
    Indexed_NetIDs[Index] := Fixed_Probabilities.ElementAt(j);
end
```

4.1.3 Searching

Using the Indexed array, the reputation of any IP address can be easily fetched by converting it to an index to its reputation. The reputation searching is illustrated in Algorithm 3.

Algorithm 3: Initialize the Indexing Array with the corresponding probabilities

```
Data: Indexed_NetIDs and IP
Result: Reputation
Index := IP / 256 ;
Reputation := Indexed_NetIDs[Index] ;
```

4.2 Binary Search Algorithm

For the second algorithm, we employ the classic Binary Search algorithm with a small adjustment. In order to employ Binary Search, we initialize an array with the start and the end addresses of every net block. Then, in order to find a reputation, we find the array index of the corresponding prefix starting address of an input IP, and use it as an index to the probabilities array.

4.2.1 Data structure

Our data structure for this algorithm is a sorted array. This sorted array contains the initial and end addresses of every net block for which we have a reputation. Our goal in the binary search implementation is not to find a value in the array but to find the relevant range for an IP address. For instance, given a set of $\{1.1.1.0/24, 1.1.24.0/24, 2.16.196.0/20\}$, our data structure to start the search is an array of $[16843008, 16843520, 16848896, 16849152, 34653184, 34654208]$

4.2.2 Initialization

To build our data structure, we need to load all the initial and end addresses of a net block in a sorted array. To achieve this, we read each NetID and the corresponding lengths from a sorted array, and according to its prefix size, we compute the end address of the prefix. We store the initial addresses at Odd indexes and the end addresses at even in indexes of the array. It goes unsaid that this array is as twice as the initial NetID array. Algorithm 4 illustrates the process to construct our data structure.

Algorithm 4: Initializing range array data structure for the binary search

```
Data: Sorted NetIDs and NetID_Lengths
Result: Sorted range array R_NetIDs
R_NetIDs_size := 2 * len(NetIDs) ;
R_NetIDs = Byte[R_NetIDs_size] ;
for i:=1 to len(NetIDs) do
    odd := NetIDs[i];
    even := odd + 2^(32−NetID_Lengths[i]) ;
    R_NetIDs[2i-1] := odd;
    R_NetIDs[2i] := even;
end
```

4.2.3 Searching

Our binary search algorithm has a very similar structure to the classic binary search algorithm. That said, we adjusted the algorithm to search for the range an IP address belongs to. If we have an entry for the reputation of the net block the IP address belongs to, then the IP address value falls between an odd (lower band) and an even (higher band) value in our data structure.

Otherwise, the IP address value falls between an even (lower band) and an odd (higher band) value. Following the example we presented in the beginning of this section, the result for searching 1.1.25.53 must return 0. Looking at $[16843008, 16843520, 16848896, 16849152, 34653184, 34654208]$, we learn that 16849205, integer value of 1.1.25.53, falls between element 4 and 5 of the array. Since the lower band, 4, is even we return 0. Algorithm 5 illustrates our binary search algorithm.

Algorithm 5: Binary search algorithm based on our data structure

```
Data: R_NetIDs, Probabilities and IP
Result: Reputation
low := 1 ;
high = len(R_NetIDs) ;
while low + 1 < high do
    middle := ⌈(low+high)/2⌉;
    if R_NetIDs[middle]>=IP then
        high := m;
    else
        low := m;
    end
end
if low%=1 then
    index := ⌊low/2⌋ + 1 ;
    Reputation := R_NetIDs[index];
else
    Reputation := 0;
end
```

4.3 Conclusion

In this chapter, we presented two algorithms that will be the based of our comparison for implementation performance of the two aggregation features. We adopt Hash Table and Binary Search concepts to implement our algorithms. The first algorithm that we present has O(1) and is optimized for fast processing while the second algorithm is optimized for footprint and has o(log(n)).

Chapter 5

Data Collection Methodology

In this chapter, we explain what dataset we use for our analysis. Our data comes from Redsocks company that is a cyber security solution provider in Netherlands. The data that they shared with us contains Indicator of Compromise used for malicious detection from network traffic. A majority of the data identifies command and control servers and also bots. We process the raw data that we receive and build a dataset of IP prefixes with an assigned score showing the number of malicious IP addresses within that prefix. In the rest of this chapter, in Section 5.1, we explain in detail what our raw dataset from Redsocks Security (RS) contains. In Section 5.2, we discuss the structure of the prefix dataset we would generate from raw data. In Section 5.3, we present our preprocessing steps on the raw data to construct the prefix dataset we will use later for experimental analysis. In section 5.4, we give some statistics about our prepared prefix dataset.

5.1 Redsocks Raw Dataset

Redsocks Security(RS) collects Indicators Of Compromise (IOC) on a daily basis. These IOCs are used for behavioral detection of malicious activity. RS gave us access to this database for a period of 1 month starting from 2017-04-18. Since they store IOCs with a timestamp, we fetch only the IOCs that are collected within the aforementioned period. The number of indicators for four weeks of data collection that we have are reported in Table 5.1. These values report the indicators from all types e.g. file hash, URL etc. In Section 5.3, we explain how we process these indicators and extract the data we need.

Start Date	End Date	Indicators
2017-04-18	2017-04-24	336037
2017-04-25	2017-05-01	349656
2017-05-02	2017-05-08	757027
2017-05-09	2017-05-15	329075

Table 5.1: RS IOC statistics

5.1.1 Indicators Of Compromise

An indicator Of Compromise (IOC) is a unique trace from malware that can be used to identify malware activity. The IOCs in the dataset that we had access to have many attributes. Both because of confidentiality and also lack of relevance we do not discuss all the attributes. The relevant attributes to our work are *indicator, source and timestamp*. The indicator is a string value of different length that contains the valuable data to indicate a malicious activity. IP address, DNS name, URL and file hash value, among others, are some types of IOC that one can expect from the raw dataset we had access to. Source stores the information about the source that the IOC has been collected from e.g. a malware lab. In the next subsection, we elaborate on some of the sources. Timestamp says about the date that the IOC is collected. This attribute is important to us since an IOC is not always valid; for instance, an infected workstation IP address may be an IOC in the dataset to identify a C&C but after the workstation is cleaned the corresponding IP must also be removed.

5.1.2 IOC Sources

Due to business confidentiality, we can not mention all the sources that RS uses to collect IOCs. That said, we mention the tops sources, based on count analysis, that the majority of the IOCs come from. Our count analysis shows that manual input from RS malware analysts, VirusTotal and two malware labs that RS has are the main providers of IOCs. RS also uses public datasets such as Zeus Tracker and Phishtank. However, the data from such sources is one order of magnitude less than the other sources that we mentioned.

5.1.3 How accurate is our raw data?

The raw data that we base our work on reveals the IOC based on malware analysis. This means we expect that a majority of the IP addresses are associated with command and control servers and also bots. Via manual analysis, we randomly selected some of the indicators and checked the associated IP addresses with public DNSBLs. Our analysis showed that most of the indicators are also blacklisted by DNSBLs. There are however some indicators that are not listed in DNSBLs. Redsocks Security claims that this is their added value and their database contains IOCs that can not be found in other databases. That said, because of lack of ground truth [19] in computer security domain, we can not

independently qualify our dataset and report False Positive and False Negative values for it.

5.2 Prefix Dataset Structure

In order to analyze /24 Fixed aggregation or BGP aggregation, we need to analyze the distribution of IP addresses in their corresponding prefixes. Henceforth, we build a dataset of prefixes from indicators showing the number of malicious indicators (IPs) in each prefix. Each row in our dataset has the following attributes:

- *NetID*: a 32 bit integer that is numeric representation of a net block ID. For instance, 188.201.133.0 is represented with 3167323392 value
- *PrefixLength*: Prefix length is the number of bits used to represent the NetID of an IP address. For instance, in CIDR representation 188.201.133.0/24, 24 is the PrefixLength.
- *IndicatorsCount*: This attribute reports the number of malicious IPs within the network of the NetID.
- *BGPID*: This attribute reports the CIDR that the NetID belongs to according to the routing snapshots. For instance, our previous example belongs to 188.200.0.0/14.
- *timestamp*: Since indicators and BGPIDs are valid only for a period, we store the date that we compile the record.

5.3 Preprocessing

We process the raw data from RS and then compile our prefix dataset. We perform this preprocessing in three steps (see Figure 5.1). The steps in details are:

1. *IP extraction*: Since not all the indicators are IP, we need to extract the IP indicators or try to map an indicator to an IP address. A mapping can be drawn between URL and domain indicators and an IP address. For URLs, we extract the host name part of the URL and then resolve it to an IP address. For domains, we again query it and try to resolve it to an IP address. We note that not all host names are resolvable; in such cases, we discard the indicator. We also note that an IP address may be pointed by multiple indicators i.e. different URLs, domain names etc. In our analysis, such IP addresses are treated same as other IP addresses; considering the weight for such IP addresses and analyzing the effect is beyond the scope of the current work. Finally, we note that not the same IP addresses can be derived at different points of time from the same dataset of indicators; domains and URLs may point to different IP addresses at different points

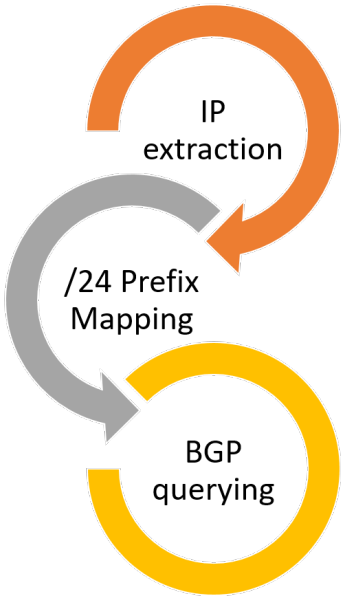


Figure 5.1: Preprocessing

of time. That said, the closest time to the time when an indicator is collected is the best time to query it for the malicious IP address.

2. /24 *prefix mapping*: In our analysis, a /24 prefix is the smallest prefix (from the number of hosts point of view) that an IP address can belong to; for Fixed aggregation, this prefix is indeed the one and for BGP the prefix is either of size 24 or larger. Henceforth, we first mask every IP address with `0xffffff00` value and derive the prefix it belongs to. If it is not already listed in our dataset, we include that prefix. Otherwise, we increase the IndicatorsCount value by one. The PrefixLength as the name of this step implies is always 24; analysis based on other sizes would change this value.

3. *BGP querying*: To compare Fixed aggregation with that of BGP, we need to know how IP addresses are distributed in BGP prefixes. We map each /24 Fixed prefix to a BGP one. The mapping is done using [53]. To have an accurate mapping, we load the relevant BGP announcements according to indicators date from [52] into *pyasn* script. To compute the number of malicious IP addresses in a BGP prefix, we have to aggregate the score of all the /24 prefixes that it encompasses. To expedite analysis, we store the result of such aggregation separately.

5.4 Generated Prefix Datasets

The result of our data collection and the final dataset that we base our analysis is accessible from [54]. This dataset contains the prefixes that we observe malicious activities from the Indicators of Compromises Redsocks Security shared with us from 2017-04-18 to 2017-05-18. Each BGP and /24 prefix has an assigned score that reflects the number of malicious IP addresses from that prefix. The dataset is in SQL format and it has been implemented in a MySQL database. The data from each week and based on the aggregation feature (fixed or BGP) is reported in a different table. Furthermore, aggregation based on different training lengths (1,2 and 3 weeks) is also reported in a different table.

Table 5.2 reports the data that the database that we will use[54] has. The number of IP addresses, /24 prefixes and BGP prefixes that we observed in each week is reported in Table 5.3.

Start Date	End Date	name	aggregation feature
2017-04-18	2017-04-24	fixed_week_0418	Fixed
2017-04-18	2017-04-24	bgps_week_0418	BGP
2017-04-25	2017-05-01	fixed_week_0425	Fixed
2017-04-25	2017-05-01	bgps_week_0425	BGP
2017-05-02	2017-05-08	fixed_week_0502	Fixed
2017-05-02	2017-05-08	bgps_week_0502	BGP
2017-05-09	2017-05-15	fixed_week_0509	Fixed
2017-05-09	2017-05-15	bgps_week_0509	BGP
2017-04-18	2017-05-01	fixed_2weeks_20170418	Fixed
2017-04-18	2017-05-01	bgps_2weeks_20170418	BGP
2017-04-18	2017-05-08	fixed_3weeks_20170418	Fixed
2017-04-18	2017-05-08	bgps_3weeks_20170418	BGP

Table 5.2: [54] dataset that we will use for analysis

Start Date	End Date	Unique IP addresses	/24 prefixes	BGP prefixes
2017-04-18	2017-04-24	113357	61222	26197
2017-04-25	2017-05-01	103298	71866	30274
2017-05-02	2017-05-08	154939	83018	34782
2017-05-09	2017-05-15	128475	62557	26768

Table 5.3: Processed IOC and prefixes statistics

Chapter 6

Experimental Results

In chapter 3 and 4, we presented the groundwork for quantitative comparison of BGP and /24 fixed aggregation. In chapter 6, we presented our data collection methodology and an overview of our data. In this chapter, we employ the metrics we defined in chapter 3 and present quantitative values based on our data. Our evaluation of the data is based on splitting the data to two training and testing sets. For all the experiments, testing period is one week immediately after the training end date.

In section 6.1, we analyze the distribution of prefix lengths in BGP aggregation. We expect a significant portion of the prefixes to have lengths different than 24; otherwise, both aggregation features lead to the same view of the Internet neighborhoods and further analysis is not required. In section 6.2, we analyze the precision of each aggregation feature in predicting the likelihood of having malicious activity from a given prefix and answer RQ4. In section 6.3, we answer RQ1 and see whether better granularity of /24 fixed aggregation leads to a significantly different view of the likelihood of having malicious activity. In section 6.4, we investigate the detection rate of each aggregation feature and answer RQ2. In section 6.5, we answer RQ3 and compare the reputation lookup performance of each aggregation feature.

6.1 BGP Prefix Length Distribution

We measure the distribution of the BGP prefixes and IP indicators over different lengths. Figure 6.1 shows such distributions. This graph gives us two insights. First, BGP aggregation leads to a significant difference in the length of the prefixes. Second, there is not a a correlation between BGP prefix length and the number of hits; it may be assumed that since a larger prefix length encompasses a larger space, it should also have higher hits while this is not the case in reality. Third, there is not any correlation between the number of prefix entries and the number of hits. In other words, more entries for a specific length does not result in the identification of more malicious IP addresses. In conclusion, we learn from

the distribution of the prefixes with malicious activity over different lengths in BGP aggregation that aggregation based on BGP would lead to a different view of the prefixes and grouping of Internet hosts.

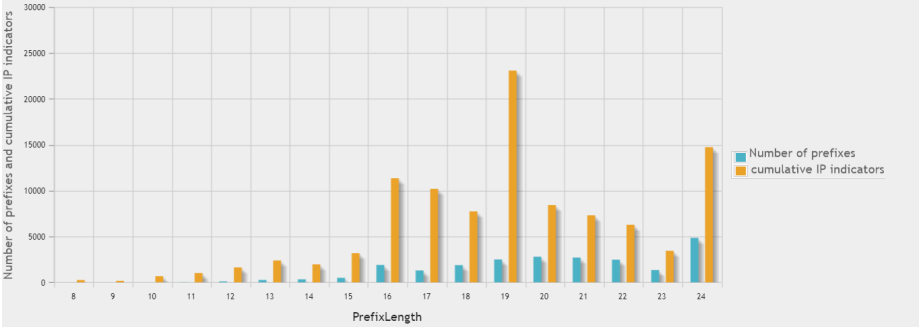


Figure 6.1: Distribution of IP indicators and prefixes over different lengths

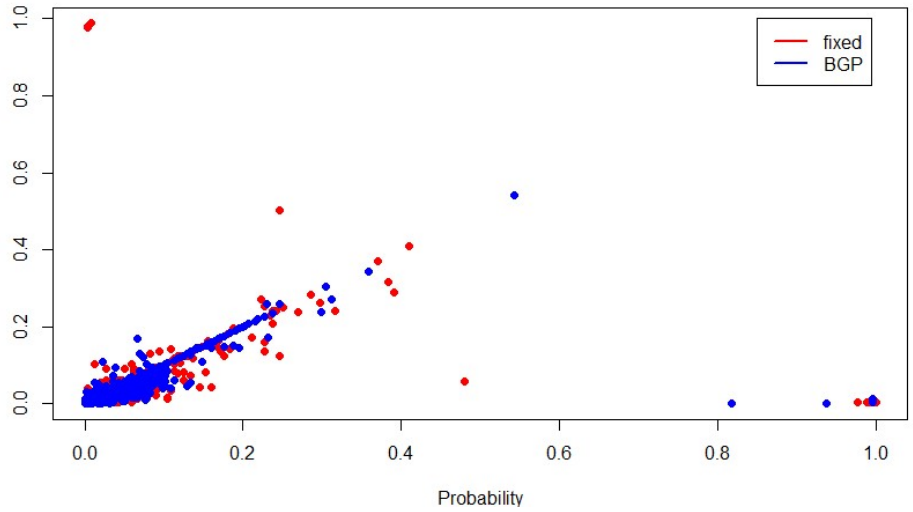
6.2 Precision

As we discussed in 3.3.4, we plot the points in J, prefixes that appear both in training and testing period, to see if there is a correlation between P, the probability of malice based on the training set on the x axis, and P', the probability of malice based on the testing set on the y axis. The graphs (see Figure 6.2a, 6.2b and 6.2c) show a strong correlation for both approaches. In order to measure the precision we will rely on numeric values.

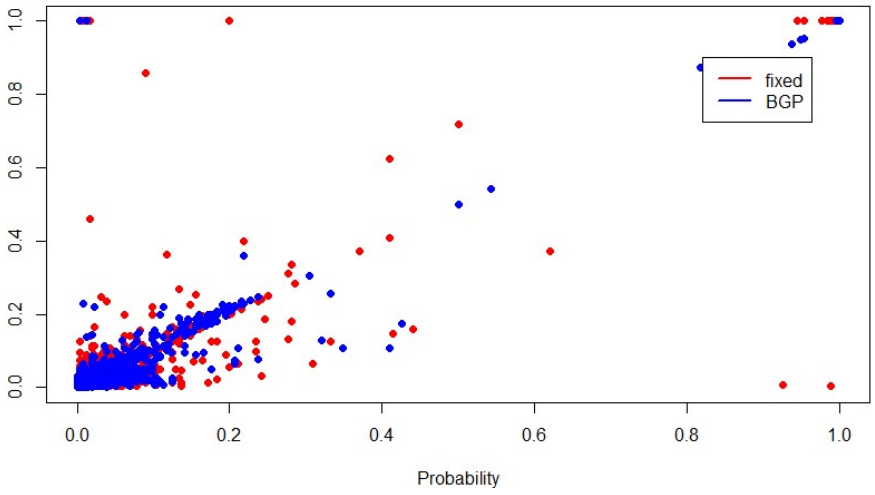
Table 6.1 and 6.2 respectively illustrate the result of linear regression modeling of P' based on P for Fixed and BGP aggregation. By analyzing the values we conclude the followings:

- In both cases, there is a very strong correlation between P and P'. This means that we can safely use P to predict the probability of maliciousness with a small error based on either approaches
- Two weeks of training for both aggregations lead to a finer modeling of P'. Longer training has diminutive effect on both approaches but relatively better effect on BGP
- Based on the R squared value, we can see that BGP will have a better modeling for larger portion of prefixes as the training time increases
- Based on 1 week of training, BGP can better predict P'

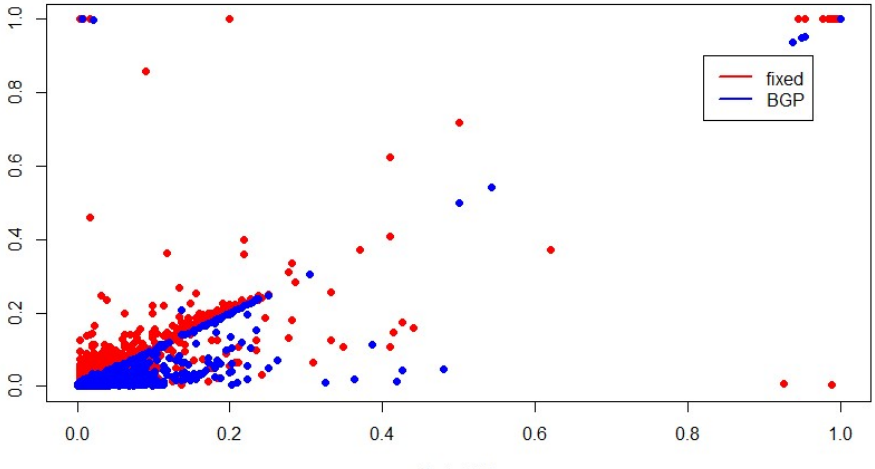
The result of our analysis, in this section, shows that BGP and Fixed aggregation are both adequately precise in the prediction of malice probability. This analysis, however, does not say anything about the probability values effect on our final prediction. To illustrate the concept, let's review an example. Assume



(a) Training(P) and Testing(P') Probabilities based on 1 week of training



(b) PP' graph based on two weeks of training



training length	Coefficient	Standard Error	P value	Residual std er	Multiple R squared
1 week	0.8808146	0.0105130	$< 2 * 10^{-16}$	0.03193	0.4109
2 weeks	0.8857	0.001517	$< 2 * 10^{-16}$	0.01493	0.8602
3 weeks	0.9692774	0.0020672	$< 2 * 10^{-16}$	0.02034	0.7987

Table 6.1: Linear Regression modeling of P' based on P for Fixed aggregation

training length	Coefficient	Standard Error	P value	Residual std er	Multiple R squared
1 week	1.0140342	0.0084435	$< 2 * 10^{-16}$	0.01912	0.5872
2 weeks	0.7635	0.002823	$< 2 * 10^{-16}$	0.01354	0.7416
3 weeks	0.9429	0.002527	$< 2 * 10^{-16}$	0.01285	0.8501

Table 6.2: Linear Regression modeling of P' based on P for BGP aggregation

that a /20 prefix in BGP has 20 malicious indicators based on our training dataset. The probability of compromise for all the members in this /20 prefix is $\frac{20}{2^{12}}$. If all these indicators are accumulated in a /24 prefix X, the probability of malice for members of X is $\frac{20}{2^8}$. Since for the adjacent prefixes of X we don't have any record the probability of malice for the those prefixes is zero. Now assume that the same values hold also for the testing set. In such situation, P' is no different than P. What our analysis revealed now is that based on the testing set, the probability for all the members of the set (either BGP or fixed) remains almost the same; in this example, the probability for the /20 prefix does not grow or drop significantly, and the same holds for the /24 prefix. That said, the probability of some /24 prefixes based on Fixed aggregation is zero but based on BGP is $\frac{20}{2^{12}}$.

6.3 Granularity

What our precision analysis misses to consider is the granularity of the probabilities. Following our previous example, fixed aggregation gives a probability of $\frac{20}{2^8}$ to all the X members while BGP gives the probability of $\frac{20}{2^{12}}$ to these members that is 16 times larger! Of course this is an example; however if in reality BGP aggregation indeed leads to such scenario then Fixed aggregation accuracy is preferred. In order to compare the granularity of the two aggregations and see if the above phenomenon exists we analyze the distribution of $Delta(p)$. Since P^{BGP} can be small, due to the BGP prefix size, we normalize $Delta(p)$ by dividing it to P^{BGP}. This normalization allows us to understand how many times larger the probability can be if we use fixed aggregation instead of BGP. Figure 6.3 shows the distribution of normalized $Delta(p)$ value. We can instantly notice that most probabilities have the same value for both approaches. Furthermore, except some outliers the normalized $Delta(p)$ is less than 10. In order to precisely analyze the granularity we report the statistical features of this distribution, excluding the outliers, in Table 6.3. As illustrated in Table 6.3, 75% of the data has no difference in probability and the mean of difference is 4% change with a small variance. The conclusion is that although fixed aggregation is inevitably more granular, the loss of information by using

BGP is negligible. *The underlying could be homogeneous distribution of maliciousness in the /24 prefixes of a BGP; in other words, nodes withing a BGP prefix behave similarly.*

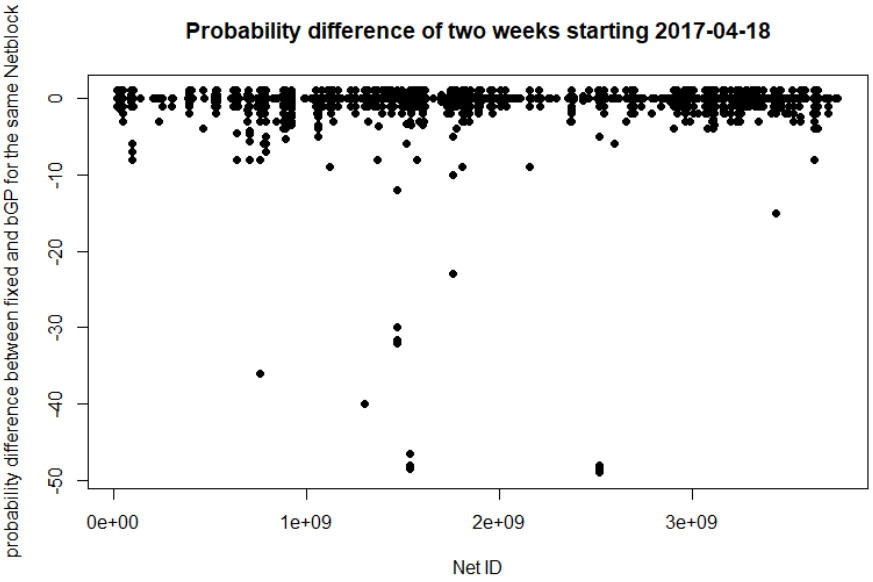


Figure 6.3: Normalized $Delta(p)$ distribution

3rd Quartile	Mean	Standard Deviation	Variance
0	0.04401	0.8731013	0.7623059

Table 6.3: Normalized Delta distribution statistics

6.4 Detection Rate

In order to see if BGP has a meaningful higher Detection Rate, we compare the number of hits and the Detection Rate of fixed and BGP aggregation successively in Table 6.4 and 6.5. A quick glance reveals that Detection Rate of BGP is meaningfully higher than Fixed aggregation. We note that after three weeks of training, the Detection Rate of fixed aggregation enhances and becomes very close to that of BGP. This can be explained by the growth in the number of stored fixed prefixes in fixed aggregation; the coverage of IP addresses by the two methods become very similar. The number of stored BGP prefixed, however, is 40% of Fixed prefixes.

Although longer training enhances the detection rate, we note that the number of entries without any hit also increases (see Table 6.4 and 6.5 last column).

Training length	# of hit prefixes	# of hit indicators	Detection rate	Testing prefixes Size	Testing indicators Size	Training prefixes size	% of prefixes without hit
1 week	53014	83445	80%	71866	103293	61222	14%
2 weeks	55396	115562	74%	83018	154930	80072	31%
3 weeks	54835	107655	83%	62557	128470	107692	49%

Table 6.4: Detection rate and values for Fixed aggregation

Training length	# of hit prefixes	# of hit indicators	Detection rate	Testing prefixes Size	Testing indicators Size	Training prefixes size	% of prefixes without hit
1 week	23430	91291	88%	30274	100061	26197	11%
2 weeks	25494	133202	86%	34782	149771	33041	23%
3 weeks	24535	110365	86%	26768	122207	42328	42%

Table 6.5: Detection rate and values for BGP aggregation

The increase in number of such entries can result in false positives if there is traffic from such prefixes to a monitored source. Therefore, based on the high detection rate and also the low percentage of unused entries(entries without hit) of BGP, we conclude that BGP has a better detection rate; it can faster learn and predict the net blocks that indeed expose malicious activity.

The detection concept that we presented above only says that we can report the reputation of the IP address, which appears in the traffic, through the reputation of its net block. It goes unsaid that the higher the probability that we report, the higher the chance that the traffic is indeed malicious. This implicitly means that for lower probabilities we need to rely more on other features to detect malicious activity. In order to give an insight about the probabilities that we would report based on the hits we employ CDF metric that we defined in Section 3.3.2 for two weeks of training. Figure 6.4 illustrates the CDF of hits for both aggregation features. We note that for almost 80% of the data we report probabilities less than 20%. Furthermore, for around 60% of hits we report numbers close to zero. This implies that a designer of an IDS must treat such traffic with low probabilities of malice with discretion; on one hand, networks with low probabilities can not be treated as benign since there are indeed malicious traffic from such networks. On the other hand, if other detection features can not distinguish harmful from benign traffic in these networks, a lot of false positives will be raised that is not desired. For other 20% of the hits, we report relatively high probabilities. For the entire data, Fixed aggregation reports higher probabilities and the difference is especially significant for the top left of Figure 6.4. From Section 6.2 results, we know the reported probabilities are adequately precise for both aggregation features.

We can conclude that by only using prefix reputation, we can not comprehensively identify the entire malicious traffic. This is because in such case, we should have a cutoff threshold to alert malicious activity; such threshold must be chosen as high as possible to avoid false positive. Our CDF analysis revealed that by having a probability threshold of greater than or equal to 85% we can identify 10% of the hits in BGP aggregation. In contrast, by having a threshold of around 95% in Fixed aggregation, we can identify around 20% of the hits. Combining this result with our precision analysis, we conclude that after two

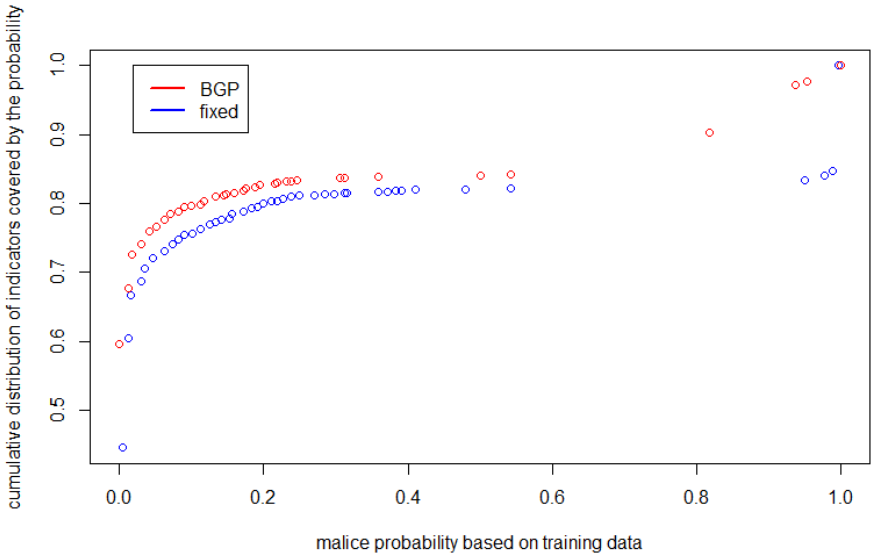


Figure 6.4: CDF of hits based on probability of malice

weeks of training, Fixed aggregation with a threshold of $> 95\%$ *can precisely detect only about 20% of the malicious traffic*

6.5 Lookup performance

In Chapter 4, we presented two algorithms that are the base for our comparison in this section. Indexing Search Algorithm (see Algorithm 3) is optimized for fast processing with $O(1)$. There is, however, a memory footprint penalty with this algorithm. In contrast, Binary Search Algorithm (see Algorithm 5) is optimized for memory footprint with a processing performance penalty that results in $O(log(n))$. In this section, we compare Fixed with BGP aggregation based on each of this algorithm.

For Indexing Search Algorithm, the lookup order value and footprint for both approaches are the same. Since the Order of this algorithm is $O(1)$ the lookup order value for both approaches is 1. The memory (and Disk) footprint for both Fixed and BGP aggregation is $16MB$ (based on 2^{24} prefixes of length /24 and $1B$ for score storage). This may come as a surprise to the reader since BGP has less entries than /24 Fixed prefixes. In 2, however, we processed the BGP entries and derived the probabilities of child /24 prefixes. Therefore, for both cases our point of reference for fetching the malice probability of an IP address is its /24 prefix that is computed by masking the IP value with `ffffff00`. This entails that we have the same values for all the adjacent /24 prefixes in parent

b

Training length	Performance metric	/24 Fixed aggregation	BGP aggregation
1 week	Number of prefixes(n)	61222	26197
	Lookup order value	O(log(61222))=O(15.90)	O(log(26197))=O(14.68)
	Footprint	0.55MB	0.24MB
2 weeks	Number of prefixes(n)	80072	33041
	Lookup order value	O(log(80072))=O(16.29)	O(log(33041))=O(15.01)
	Footprint	0.72MB	0.30MB
3 weeks	Number of prefixes(n)	107692	42328
	Lookup order value	O(log(107692))=O(16.72)	O(log(42328))=O(15.37)
	Footprint	0.97	0.38MB

Table 6.6: Lookup performance comparison based on BSA with $O(log(n))$

BGP prefix.

For Binary Search Algorithm (BSA), the lookup order value and footprint is different for each approach since n (the number of prefixes) is different. Table 6.6 reports the result of our computation for each approach. Footprint column is calculated based on Algorithm 5. In the implementation of this algorithm, two arrays are required; one for storing the lower and higher band of each prefix and another for storing the scores. Since IP is 32 bits long we assume that the size of the first array is $4 * n * 2 = 8n\ Bytes$ where n is the number of prefixes. 4 in the calculation represents a $4B$ integer and 2 represents the lower and higher band for each prefix. The second array stores a score of one byte long for each prefix. Hence, the footprint for n prefixes is $9n$ Bytes.

According to Table 6.6, the lookup order value for BGP is $O(1)$ faster for the order of n that our dataset has. It is hard to measure the exact computation time performance since it depends on the CPU power and also the number of IPs that are looked up in a unit of time. The footprint for BGP is 2 to 2.5 times less than Fixed aggregation given n based on our datasets.

6.6 Conclusion

In summary, our analysis shows that aggregation of reputation based on BGP announcements is better than aggregation based on Fixed /24 prefixes. Firstly, BGP is equally precise and sometime preciser than fixed aggregation in predicting the malice likelihood of the neighborhood where an IP address is originated. Secondly, although fixed aggregation is always more granular by definition, the loss of likelihood granularity by BGP aggregation is negligible. Thirdly, BGP has a better detection rate given a short training time; this possibly leads to less false positives. Finally, in terms of lookup performance BGP has equal or better lookup performance in terms of lookup Order and footprint.

Chapter 7

Conclusion

IP address reputation is an important element in detecting malware malicious activity from traffic. Individual IP address reputation, however, has a high false negative rate. Findings from analysis of malicious IP addresses distribution suggest that so called "Bad Internet neighborhoods" with high density of malicious activity exists. Based on these findings, aggregating reputation and attributing reputation to Internet neighborhoods can improve the prediction and hence the false negative rate.

Two important aggregation approaches based on BGP and /24 IP prefixes differ in their level of granularity. The level of granularity in reputation aggregation impacts the criteria that are important for Malware Threat Detection vendors; these criteria are low False Positive (FP), low False Negative (FN) and low processing overhead. Finer granularity is expected to reduce FP while increasing FN. That said, investigating the effect of finer granularity of fixed /24 prefix aggregation on FP/FN in comparison to BGP aggregation is hindered by lack of ground truth and live traffic data.

Based on available data, the author of this thesis defined substitute metrics (to FP and FN) to compare the detection and implementation performance of fixed /24 with BGP prefix aggregation. Fixed /24 prefix aggregation is more granular and has less detection rate than BGP since /24 prefixes are a subset of BGP prefixes. That said, investigating the loss of granularity by BGP aggregation and the gain in detection rate need experimental analysis. Furthermore the processing overhead of each aggregation approach should be considered based on the underlying reputation lookup algorithm.

The author in this thesis measured precision of malice likelihood, malice likelihood granularity loss, detection rate, Reputation lookup Order value and footprint of BGP and fixed /24 aggregation. The author used a dataset of Indicators of Compromise for a period of one month. For evaluation, the author split the dataset to different length training and testing sets.

The experimental analysis shows that the loss of granularity in malice likelihood view of the IP space by using BGP is negligible, around 4% of difference on average, while the detection rate improvement is substantial, almost 10%.

Furthermore, malice likelihoods derived from either approach are adequately precise in predicting the probability of having malicious activity; the p-value based on regression analysis in both cases is almost zero. Moreover, the lookup performance and footprint of BGP prefixes are slightly better than fixed /24 prefixes; lookup performance of BGP is O(1) better and its footprint is half of fixed /24 prefix. In summary, BGP prefixes better identify the neighborhood to which an IP belongs and hence are preferred for reputation aggregation.

7.1 Relation to State of The Art

Our work is similar to [17, 11, 10, 4, 5, 14, 16, 12] but it has three major differences. Firstly, the application of "Internet bad neighborhood", "Network aware clustering" or uncleanliness [1] to exploit the networks structure characteristics is novel in our work. Jung et al use network aware clustering, aggregation based on BGP announcements, to detect DOS attacks[11]. [12, 4, 5, 14, 55] propose a spam mitigation technique based on IP aggregation. Except [55], the rest of the works examine also the content in conjunction with the IP aggregation filter. Because spam can be identified based on the content, measuring false positive is straightforward. Based on the same reason, the usage of IP aggregation is less prone to false positive since content itself has a very low false positive. On the other hand, we aim to identify malware traffic activity, and malware infection can not be easily identified while there is not still any published signature. Identifying malicious activity via network traffic without deep packet inspection is even harder since a malicious traffic can look exactly like a legit traffic.

Secondly, in the current work, we introduce likelihood of having malicious activity from the source network of an IP address as a malice detection feature. In contrast, previous works mainly see aggregated reputation in a black and white manner; the IP is either malicious based on its originating network or benign. Qian et al define spam ratio of the originating network and then use a threshold of spam ratio to blacklist the entire IP space of the prefix [14]. They measure the FP/FN pair of different spam ratios and then find the best ratio that balances this pair. We maintain that aggregation can not be seen in a black and white state for malware traffic detection. [16] reports the top autonomous systems and organizations with highest amount of spam activity. Although such report provides valuable insight about the poor security measures in some Internet Service Providers the underlying data is too coarse to be used for malicious detection.

Thirdly, none of the previous works investigated the loss of granularity by using BGP announcements in comparison to fixed /24 prefix. There are two school of thoughts in the literature. The first school of though uses /24 prefix to aggregate reputations [10, 5, 55]. The second school of thought uses BGP announcements for aggregation and grouping Internet hosts [11, 12, 4, 14, 16]. The first group believes that /24 has the finest granularity since BGP prefixes can be as long as /24. The second group believes that the clustering of the

[1] In the literature, all refer to the same concept

Internet hosts must be based on the administration boundaries and BGP draws such administration boundary. In this work, thanks to the likelihood of having malicious activity concept, we show that the granularity loss by using BGP is negligible while the detection rate gain is substantial (around 10%). Qian et al also examine the granularity of BGP prefixes [14]. That said, Qian et al do not compare BGP with /24, and rather try to refine the BGP prefixes granularity using DNS clustering. Based on their finding, 10% of BGP prefixes can be refined and coarse prefixes are usually shorter than /16. That said, we haven't observed any entry in our aggregation with prefix length shorter than 16; the malice likelihood of those entries are zero because of their large size. Once again, such finding shows that likelihood is a fine metric and can neutralize the effect of size growth of coarse BGP prefixes.

7.2 Future Work

Two directions can be pursued following this work. The first direction is investigating the application of aggregated reputation of BGP prefixes for different purposes. For instance, the effect of adding badhood likelihood feature to network traffic classifiers' vector of features on detection accuracy can be investigated. Another instance of badhood application is using badhood concept for network massive scanning. In such a work, scanning the Internet to identify a command and control server based on a special signature can be prioritized based on likelihood of having malicious activity.

The second line of work can improve the current aggregation scheme. For instance, in chapter 6, we reported that for few BGP entries the granularity loss can be substantial. The reason is aggregation of routing entries in the routing tables, or the lack of enough BGP announcements to the outside world from the autonomous systems. For these BGP entries, individual /24 entries within the BGP prefix can be instead used. The aggregation can be further enhanced by merging the entries that have similar likelihoods; the goal is to draw the administration boundary that is invisible from the outside of the autonomous system.

7.3 Lessons and Final words

The research behind this thesis has been conducted in the last six months of the author MS program. These six months significantly developed the author both scientifically and socially. In the scientific context, the author learned that Ground Truth is a must for any field including computer security to grow. Unfortunately, as a young scientist, I used to think that ground truth is a given. This lack is felt not only in academia but also industry. Henceforth, both the state of the art and security products are very diverse while the true effectiveness of the methods in terms of measurable and comparable quantitative metrics is unknown.

Furthermore, the author learned that there exist a privacy phobia in the world that appears to the advantage of the attackers. Based on the code of conducts, most security researches would not use traffic data and other available resources for analysis without permission. The best security researchers can do to further their research is to design methods and tools to make processing possible whilst addressing the privacy concern. Yet, the officials would not understand the designed technology and do not allow the analysis on the sensitive data. On the other hand, attackers would use the same resources to improve their attacks and tools.

In the social context, the author learned invaluable lessons. First of all, the author learned that the most difficult part of any research or development is communicating the motivation and the findings. Unless the author finds a way to communicate effectively, there would not be any contribution. This is because in high tech studies, there are many details that only a few people fully understand. Therefore, the audience would not understand the importance of the work and would not further go into the details of the work in case of ineffective communication. In consequence, it is only the author who is aware of the advancement and the work is lost.

Second of all, the author learned that the world is not neither fair nor rational all the time. The author used to think that an action has a predictable consequence; the prediction may be wrong but this is an error of the predictor nor the environment. During this research, the author learned that consequences may not be always predictable because the humans' actions are not always logical. The lack of logic is due to the lack of unbiased truth. In other words, everybody sees the world through his own frame and this causes inconsistency in our logics.

Last but not the least, the author learned that the only way to beat the mishaps and succeed is not to give up. The world can be unfair, the actions may be unreasonable but not always! The only way to ensure success is to try again and again. As long as there is an effort, there is hope. We may not know but every failure makes us only closer to the final goal. In conclusion, I would like to say it is the journey that matters not the destination!

Bibliography

[1] 2016 data breach investigations report. http://www.verizonenterprise.com/resources/reports/rp_DBIR_2016_Report_en_xg.pdf, 2016. Accessed: 2017-08-02.

[2] Ddos attack that disrupted internet was largest of its kind in history, experts say. https://www.theguardian.com/technology/2016/oct/26/ddos-attack-dyn-mirai-botnet, 2016. Accessed: 2017-08-02.

[3] "wannacry" ransomware attack losses could reach $4 billion. http://www.cbsnews.com/news/wannacry-ransomware-attacks-wannacry-virus-losses, 2017. Accessed: 2017-08-02.

[4] Shobha Venkataraman, Subhabrata Sen, Oliver Spatscheck, Patrick Haffner, and Dawn Song. Exploiting network structure for proactive spam mitigation. In *Usenix Security*, 2007.

[5] Ward Van Wanrooij and Aiko Pras. Filtering spam from bad neighborhoods. *International Journal of Network Management*, 20(6):433–444, 2010.

[6] Manos Antonakakis, Roberto Perdisci, David Dagon, Wenke Lee, and Nick Feamster. Building a dynamic reputation system for dns. In *USENIX security symposium*, pages 273–290, 2010.

[7] Giovane CM Moura, Carlos Ganán, Qasim Lone, Payam Poursaied, Hadi Asghari, and Michel van Eeten. How dynamic is the isps address space? towards internet-wide dhcp churn estimation. In *IFIP Networking Conference (IFIP Networking), 2015*, pages 1–9. IEEE, 2015.

[8] Yinglian Xie, Fang Yu, Kannan Achan, Eliot Gillum, Moises Goldszmidt, and Ted Wobber. How dynamic are ip addresses? In *ACM SIGCOMM Computer Communication Review*, volume 37, pages 301–312. ACM, 2007.

[9] Sushant Sinha, Michael Bailey, and Farnam Jahanian. Shades of grey: On the effectiveness of reputation-based "blacklists". In *Malicious and Unwanted Software, 2008. MALWARE 2008. 3rd International Conference on*, pages 57–64. IEEE, 2008.

[10] M Patrick Collins, Timothy J Shimeall, Sidney Faber, Jeff Janies, Rhiannon Weaver, Markus De Shon, and Joseph Kadane. Using uncleanliness to predict future botnet addresses. In *Proceedings of the 7th ACM SIGCOMM conference on Internet measurement*, pages 93–104. ACM, 2007.

[11] Jaeyeon Jung, Balachander Krishnamurthy, and Michael Rabinovich. Flash crowds and denial of service attacks: Characterization and implications for cdns and web sites. In *Proceedings of the 11th international conference on World Wide Web*, pages 293–304. ACM, 2002.

[12] Anirudh Ramachandran and Nick Feamster. Understanding the network-level behavior of spammers. In *ACM SIGCOMM Computer Communication Review*, volume 36, pages 291–302. ACM, 2006.

[13] Giovane CM Moura, Ramin Sadre, Anna Sperotto, and Aiko Pras. Internet bad neighborhoods aggregation. In *Network Operations and Management Symposium (NOMS), 2012 IEEE*, pages 343–350. IEEE, 2012.

[14] Zhiyun Qian, Zhuoqing Morley Mao, Yinglian Xie, and Fang Yu. On network-level clusters for spam detection. In *NDSS*, 2010.

[15] Craig A Shue, Andrew J Kalafut, and Minaxi Gupta. Abnormally malicious autonomous systems and their internet connectivity. *IEEE/ACM Transactions on Networking (TON)*, 20(1):220–230, 2012.

[16] Giovane César Moura. *Internet bad neighborhoods*. Number 12. University of Twente, 2013.

[17] Balachander Krishnamurthy and Jia Wang. On network-aware clustering of web clients. *ACM SIGCOMM Computer Communication Review*, 30(4):97–110, 2000.

[18] Matthijs GT Van Polen, Giovane CM Moura, and Aiko Pras. Finding and analyzing evil cities on the internet. In *IFIP International Conference on Autonomous Infrastructure, Management and Security*, pages 38–48. Springer, 2011.

[19] Sebastian Abt and Harald Baier. Are we missing labels? a study of the availability of ground-truth in network security research. In *Building Analysis Datasets and Gathering Experience Returns for Security (BADGERS), 2014 Third International Workshop on*, pages 40–55. IEEE, 2014.

[20] Surbl. http://www.surbl.org/. Accessed: 2017-03-22.

[21] Spamhaus project, spamhaus block list (sbl). https://www.spamhaus.org/sbl/. Accessed: 2017-03-22.

[22] Composite blocking list. http://www.abuseat.org/. Accessed: 2017-03-24.

[23] Spamhaus project, exploits block list (xbl). https://www.spamhaus.org/xbl/. Accessed: 2017-03-22.

[24] Ironport systems' spamcop blacklists. https://www.spamcop.net/bl.shtml/. Accessed: 2017-03-24.

[25] Malware domain list. https://www.malwaredomainlist.com/. Accessed: 2017-03-24.

[26] Dns-bh – malware domain blocklist. http://www.malwaredomains.com/. Accessed: 2017-03-24.

[27] Zeus tracker. https://zeustracker.abuse.ch/. Accessed: 2017-03-24.

[28] Joe wein spam domain blacklist. http://www.joewein.de/sw/blacklist.htm/. Accessed: 2017-03-24.

[29] Uribl. http://uribl.com/. Accessed: 2017-03-24.

[30] Sorbs. http://www.sorbs.net/. Accessed: 2017-03-24.

[31] Dsheild. https://www.dshield.org. Accessed: 2017-03-24.

[32] Opendns, phishtank. http://www.phishtank.com/. Accessed: 2017-03-22.

[33] hphosts for your pretection. http://hosts-file.net/. Accessed: 2017-03-29.

[34] Openbl. http://www.openbl.org/. Accessed: 2017-03-24.

[35] Veris community database. http://veriscommunity.net/vcdb.html/. Accessed: 2017-03-24.

[36] Web-hacking-incident-database. http://projects.webappsec.org/w/page/13246995/Web-Hacking-Incident-Database. Accessed: 2017-03-24.

[37] Wpbl: Weighted private block list. http://www.wpbl.info/. Accessed: 2017-03-29.

[38] Support intelligence, llc. http://www.support-intelligence.com/. Accessed: 2017-03-22.

[39] Uceprotector network. http://www.uceprotect.net/. Accessed: 2017-03-29.

[40] Apwg, anti-phishing working group. http://www.antiphishing.org. Accessed: 2017-03-22.

[41] Netpilot gmbh, viruswatch mailing list. http://lists.clean-mx.com/cgi-bin/mailman/listinfo/viruswatch/. Accessed: 2017-03-22.

[42] Malware patrol, malware block list. http://www.malwarepatrol.net/. Accessed: 2017-03-22.

[43] Shadowserver foundation. https://www.shadowserver.org/. Accessed: 2017-03-22.

[44] Shuang Hao, Nadeem Ahmed Syed, Nick Feamster, Alexander G Gray, and Sven Krasser. Detecting spammers with snare: Spatio-temporal network-level automatic reputation engine. In *USENIX security symposium*, volume 9, 2009.

[45] Fabio Soldo, Anh Le, and Athina Markopoulou. Blacklisting recommendation system: Using spatio-temporal patterns to predict future attacks. *IEEE Journal on Selected Areas in Communications*, 29(7):1423–1437, 2011.

[46] Jian Zhang, Phillip A Porras, and Johannes Ullrich. Highly predictive blacklisting. In *USENIX Security Symposium*, pages 107–122, 2008.

[47] Mark Felegyhazi, Christian Kreibich, and Vern Paxson. On the potential of proactive domain blacklisting. *LEET*, 10:6–6, 2010.

[48] Yang Liu, Armin Sarabi, Jing Zhang, Parinaz Naghizadeh, Manish Karir, Michael Bailey, and Mingyan Liu. Cloudy with a chance of breach: Forecasting cyber security incidents. In *USENIX Security*, pages 1009–1024, 2015.

[49] Michel Van Eeten, Johannes M Bauer, Hadi Asghari, Shirin Tabatabaie, and David Rand. The role of internet service providers in botnet mitigation an empirical analysis based on spam data. In *Proceedings of the 9th Workshop on the Economics of Information Security (WEIS)*, 2010.

[50] Niels Provos Panayiotis Mavrommatis and Moheeb Abu Rajab Fabian Monrose. All your iframes point to us. In *USENIX Security Symposium*.

[51] Jing Zhang, Zakir Durumeric, Michael Bailey, Mingyan Liu, and Manish Karir. On the mismanagement and maliciousness of networks. In *NDSS*, 2014.

[52] RouteViews Project University of Oregon Advanced Network Technology Center. Route views project. http://www.routeviews.org/. Accessed: 2017-03-22.

[53] Hadi Asghari. Python ip address to autonomous system number lookup using archived bgp dumps. https://github.com/hadiasghari/pyasn. Accessed: 2017-07-13.

[54] Ali Davanian. Prefix reputation dataset. https://github.com/adavanian/badhood. Accessed: 2017-07-13.

[55] Giovane CM Moura, Ramin Sadre, and Aiko Pras. Internet bad neighborhoods: the spam case. In *Network and Service Management (CNSM), 2011 7th International Conference on*, pages 1–8. IEEE, 2011.